Probabilistic Linear Algebra

Dissertation
der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Simon Bartels
aus Pasewalk

Tübingen
2019

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	20.10.2020
Stellvertretender Dekan:	Prof. Dr. József Fortágh
1. Berichterstatter:	Prof. Dr. Philipp Hennig
2. Berichterstatter:	Prof. Dr. Matthias Hein

Probabilistic Linear Algebra

Simon Bartels
Eberhard Karls Universität Tübingen

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich, Simon Bartels, die vorliegende Arbeit selbstständig angefertigt, keine anderen als die angegebenen Hilfsmittel benutzt und alle Stellen, die dem Wortlaut oder Sinne nach anderen Werken entnommen sind, durch Angabe der Quellen als Entlehnung kenntlich gemacht habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Ort, Datum

Unterschrift

Abstract

Linear algebra operations are at the core of many computational tasks. For example, evaluating the density of a multivariate normal distribution requires the solution of a linear equation system and the determinant of a square matrix. Frequently, and in particular in machine learning, the size of the involved matrices is too large to compute exact solutions, and necessitate approximation. Building upon recent work (Hennig and Kiefel 2012; Hennig 2015) this thesis considers numerical linear algebra from a probabilistic perspective.

Part iii establishes connections between approximate linear solvers and Gaussian inference, with a focus on projection methods. One result is the observation that solution-based inference (Cockayne, Oates, Ipsen, and Girolami 2018) is subsumed in the matrix-based inference perspective (Hennig and Kiefel 2012). Part iv shows how the probabilistic viewpoint leads to a novel algorithm for kernel least-squares problems. A Gaussian model over kernel functions is proposed that uses matrix-multiplications computed by conjugate gradients to obtain a low-rank approximation of the kernel. The derived algorithm *kernel machine conjugate gradients* provides empirically better approximations than conjugate gradients and, when used for Gaussian process regression, additionally provides estimates for posterior variance and log-marginal likelihood, without the need to rerun. Part v is concerned with the approximation of kernel matrix determinants. Assuming the inputs to the kernel are independent and identically distributed, a stopping condition for the Cholesky decomposition is presented that provides probably approximately correct (PAC) estimates of the log-determinant with only little overhead.

Zusammenfassung

Operationen der linearen Algebra bilden häufig die Basis vieler anderer Algorithmen und mathematischen Probleme. Beispielsweise, um die Dichte einer multivariaten Gauß-Verteilung zu berechnen, benötigt man die Lösung eines lineares Gleichungssystem und die Determinante einer quadratischen Matrix. Häufig, und insbesondere in Anwendungen des Maschinellen Lernens, ist die Größe der involvierten Matrizen zu groß, um exakte Lösungen berechnen zu können und man muss auf Approximationsmethoden zurückgreifen. Aufbauend auf kürzlich veröffentlichten Arbeiten (Hennig und Kiefel 2012; Hennig 2015), betrachtet diese Arbeit, Approximationsmethoden der linearen Algebra aus einer probabilistischen Perspektive.

Part iii zeigt Verbindungen auf zwischen Approximationsalgorithmen für lineare Gleichungssysteme und Gaußscher Inferenz, mit einem Fokus auf Projektionsmethoden. Ein Resultat ist die Beobachtung, dass lösungsfokussierte Inferenz (Cockayne, Oates, Ipsen und Girolami 2018) enthalten ist in der matrixfokussierten Perspektive (Hennig und Kiefel 2012). Part iv zeigt, wie sich die probabilistische Perspektive nutzen lässt, um neue Approximationsalgorithmen zu entwickeln, hier für die Lösungen von Normalgleichungssystemen für Kernmethoden. Mit einem speziellen Gauß-Prozess Modell über die Kernfunktion und unter Verwendung von Matrixmultiplikationen mit der Methode der konjugierten Gradienten erhält man eine näherungsweise Kernfunktion von niedrigem Rang. Der resultierende Algorithmus *kernel machine conjugate gradients* gibt empirisch bessere Approximationen als die Methode der konjugierten Gradienten, und, im Fall von Gauß-Prozess Regression, gibt zudem Schätzungen für Unsicherheit und Evidenz, ohne Neustart des Algorithmus. Part v beschäftigt sich der Approximation von Determinanten für Kernmatrizen. Unter der Annahme, dass die Argumente der Kernfunktion unabhängig und identisch verteilt sind, beschreibt dieser Teil eine Stopstrategie für die Cholesky-Dekomposition, die Näherungslösungen der Log-Determinante mit gewünschter Präzision und Wahrscheinlichkeit ausgibt, mit nur geringem Mehraufwand.

Acknowledgments

I am most grateful for the endless patience, guidance and trust of my adviser Prof. Dr. Philipp Hennig, for allowing me to deal with the ups and downs of working towards a Ph.D. in my own way. As a colleague once put it: discussing research with Philipp is like pursuing a Ferrari on a bicycle. It was an honor to write my dissertation under the supervision of such a brilliant and at the same time humble and understanding researcher.

I am thanking Prof. Dr. Matthias Hein for the time and effort spent to evaluate this thesis. Further, Prof. Dr. Kay Nieselt and Prof. Dr. Philipp Behrens have my gratitude for sparing valuable time for my defense.

I am grateful for inspiration and support at all times from the probabilistic numerics group: thank you, Edgar Klenske, Maren Mahsereci, Michael Schober, Hans Kersting, Alexandra Gessner, Lukas Balles, Filip de Roos, Motonobu Kanagawa, Frank Schneider, Matthias Werner, Felix Dangel, Susanne Zabel, Frederik Künstner, Agustinus Kristiadi, Jonathan Wenger and Nicholas Krämer.

Many people have patiently listened to me while I was rambling about the mathematical problems on the way to Part v. Thank you, Gabriele Abbati, Damien Garreau, Motonobu Kanagawa, Hans Kersting, Jonas Kübler, Simon Julien-Lacoste, Krikamol Muandet, Alexander Neitz, Giambatista Parascandolo, Michael Perrot, Carl Rasmussen, Luca Rendsburg, Michael Schober, Sebastian Weichwald and everyone else I may have forgotten.

Special thanks go to Inna Zeitler for listening when I had a first idea and because it was most inconvenient at the time. Special thanks go to Maja Rudolph for pointing me to the works of David Freedman which eventually lead to discovering the theorem by Fan, Grama, and Liu (2012). Special thanks go to Damien Garreau for the most difficult first proof-read of Part v.

Thank you Jon Cockayne and Ilse Ipsen for inspiring discussions about probabilistic views on linear solvers.

Thank you Pablo Garcia Moreno, Javier Gonzalez and Neil Lawrence for your supervision at Amazon Cambridge.

On my path towards this degree, I am (in chronological order) grateful for the time and effort which Prof. Karsten Wolf, Prof. Dietlinde Lau, Dr. Roland Ewald, Dr. Jan Himmelspach, Prof. Dr. Clemens Câp, Prof. Dr. Frank Hutter, Manuel Blum and Prof. Hans Rudolf Lerche invested into teaching and supervision.

Thank you Jonas Schöley for pushing me just the right way while writing this thesis.

Thank you Dr. Simon Donné for your couch the night before my defense.

Contents

List of Figures

List of Tables

List of Algorithms

Notation

Table 1: overview over frequently used letters and their context

i, j, k, l, m	discrete index variables
d, D	number of dimensions
n, N	number of datapoints
$\boldsymbol{\mu}$	mean vector
$\boldsymbol{\Sigma}$	covariance matrix
$\boldsymbol{I}$	identity matrix, $(\boldsymbol{I})_{ij} = \delta_{ij}$
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	$\boldsymbol{x}$ is distributed multivariate normal with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
$K_m(\boldsymbol{A}, \boldsymbol{b})$	Krylov space of order m generated by the matrix $A \in \mathbb{R}^{d \times d}$ and the vector $\boldsymbol{b} \in \mathbb{R}^d$
$f^{-1}(\delta)$	$\arg\sup_{\epsilon \in \mathbb{R}}\{f(\epsilon) \leq \delta\}$
$\sigma(\boldsymbol{x}_1, ..., \boldsymbol{x}_N)$	the σ-Algebra generated from random variables $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$
$\mathbf{1}_A$	indicator function for a set A

Vectors are denoted with small bold letters $\boldsymbol{x}$, matrices with capital bold letters $\boldsymbol{C}$. For a symmetric positive-definite (s.p.d.) matrix $\boldsymbol{M} \in \mathbb{R}^{d \times d}$ and two vectors $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^d$, I write $\langle \boldsymbol{v}, \boldsymbol{w} \rangle_M = \boldsymbol{v}^\top \boldsymbol{M} \boldsymbol{w}$ for the inner product induced by $\boldsymbol{M}$, and $\|\boldsymbol{v}\|_M^2 = \langle \boldsymbol{v}, \boldsymbol{v} \rangle_M$ for the corresponding norm. If $\langle \boldsymbol{v}, \boldsymbol{w} \rangle_M = 0$ these vectors are called $\boldsymbol{M}$-conjugate.

I will slightly abuse notation to describe shifted and scaled subspaces of $\mathbb{R}^d$: Let $\mathbb{S}$ be an m-dimensional linear subspace of $\mathbb{R}^d$ with basis $\{\boldsymbol{s}_1, \dots, \boldsymbol{s}_m\}$. Then for a vector $\boldsymbol{v} \in \mathbb{R}^d$ and a matrix $M \in \mathbb{R}^{d \times d}$, let

$$\boldsymbol{v} + M\mathbb{S} = \mathrm{span}(\boldsymbol{v} + M\boldsymbol{s}_1, \dots, \boldsymbol{v} + M\boldsymbol{s}_m).$$

Part I

Prologue

> “I have heard the heartbeat of the universe. I know the answers to many questions. Ask me.” The apprentice gave him a bleary look. It was too early in the morning for it to be early in the morning. That was the only thing he currently knew for sure. “Er... what does master want for breakfast?” he said. Wen looked down on their camp and across the snowfields and purple mountains to the golden daylight creating the world, and mused upon certain aspects of humanity. “Ah,” he said. “One of the difficult ones.”

—Terry Pratchett, *Thief of Time*

1 Introduction

“‘What is the chance of an earthquake?’” Stark and Freedman (2003)

In their article, Stark and Freedman (2003) discuss how to interpret the probability predicted by the U.S. Geological Survey that a large earthquake will occur within the next 30 years in the San Francisco Bay area. One concern (among many) raised in that article is that the predicted probability of 0.7 ± 0.1 relies on simulations involving numerical approximation techniques. The complaint is that the parameters for the approximation algorithms have been set *ad hoc*, such that it is not clear how reliable the outcomes of the simulations actually are.

> There is no straightforward interpretation of the USGS probability forecast. Many steps involve models that are largely untestable; modeling choices often seem arbitrary. (Stark and Freedman 2003, p. 9)

Computationally intricate models for which numerical approximation is inevitable are prevalent across sciences[1]. Ideally, one would like to run approximation algorithms only as long as is necessary for sufficiently accurate solutions, since computation costs time, energy and hence, money.

The problem translates into the two questions: how sensitive is the model to approximation and how stable is the approximation. To reason about the second question, the field of probabilistic numerics (Hennig, Osborne, and Girolami 2015) investigates *probabilistic numerical methods* (PNM) which return probability distributions, instead of only point-estimates.

Probabilistic interpretations provide an alternative perspective on numerical algorithms, and can also provide extensions such as the ability to exploit noisy or corrupted observations (Hennig, Osborne, and Girolami 2015; Cockayne, Oates, Sullivan, and Girolami 2017). Of particular interest are those PNM whose estimate coincides with approximation methods from classic numerical analysis. The relationship between PNM and classic methods has been explored for integration (e.g. Karvonen and Sarkka 2017), ODE-solvers (Schober, Duvenaud, and Hennig 2014; Schober, Särkkä, and Hennig 2018; Kersting, Sullivan, and Hennig 2018; Tronarp, Kersting, Särkkä, and Hennig 2019) and PDE solvers (Cockayne, Oates, Sullivan, and Girolami 2016) in some generality. Concerning linear algebra, attention has thus far been restricted to the conjugate gradient (CG) method (Hennig 2015; Cockayne, Oates, Ipsen, and Girolami 2018). CG is but a single member of

[1] Roeckner, Bäuml, Bonaventura, Brokopf, Esch, and Giorgetta 2003; Arras, Knollmüller, Junklewitz, and Enßlin 2018; Nille, Toussaint, Sieglin, and Faitsch 2018.

a larger class of iterative solvers, and applicable only if the matrix A is symmetric and positive-definite. Broadening the understanding of the connections between PNMs and classic linear solvers is the concern of Part iii.

Linear algebra operations appear frequently across mathematical problems: the omnipresent multivariate Gaussian probability distribution requires the evaluations of a determinant and a quadratic form, Newton's method for root finding and second order optimization requires solving linear equation systems, to name two examples. In particular, for Gaussian process inference, the size of modern machine learning datasets makes approximation necessary, and therefore Gaussian process inference will be the main application studied in this thesis. Part iv describes a novel method arising from the probabilistic perspective, particularly tailored for kernel least-squares problems of which Gaussian process regression is an instance. Part v addresses the question how statistical structure in numerical problems can be exploited in classic numerical algorithms. The particular problem under consideration is the computation of kernel-matrix determinants.

2 Publications

Some of the results presented in this thesis have been devised, developed, and published in collaboration. Of relevance for this thesis are the publications

- S. Bartels and P. Hennig (2016). "Probabilistic Approximate Least-Squares." In: *Proceedings of Artificial Intelligence and Statistics (AISTATS)* for Part iii,
- S. Bartels, J. Cockayne, I. C. F. Ipsen, and P. Hennig (2019). "Probabilistic Linear Solvers: A Unifying View." In: *Statistics and Computing* 29.6, pp. 1249–1263 also for Part iii, and
- S. Bartels and P. Hennig (2019). "Conjugate Gradients for Kernel Machines." In: *ArXiv e-prints* 1911.06048. arXiv:1911.06048 for Part iv.

The results of Part v have not yet been published.

The most credit for the publication "Probabilistic Approximate Least-Squares" goes to Philipp Hennig. The scientific ideas, the writing, as well as the analysis are mainly his work. My main contribution was the data generation.

"Probabilistic Linear Solvers: A Unifying View" has been initiated by Jon Cockayne and me. Scientific ideas have been contributed equally by all four authors. In particular, Propositions 8, 15 and 16 and Corollary 9 in this thesis (which are taken from the *op. cit.*) are contributions by my collaborators. In due place, I will be giving exact credit. The figures in Section 8.2.3 (also taken from *op. cit.*) have been generated by me, based on Jon Cockayne's simulation study from Cockayne, Oates, Ipsen, and Girolami (2018). The analysis has been performed by the two of us. All four authors have been contributing equally to the scientific writing, everyone working in every part. The final wording is more due to Jon Cockayne and Philipp Hennig.

Part iv corresponds to the publication Bartels and Hennig (2019) which is currently under its second revision at the *Journal of Machine Learning Research* (JMLR). I was the primary author and performed the principal analysis and work. Philipp Hennig provided initial ideas, direction and supervision.

For Part v, ideas, data generation, analysis and writing are my own. The results are not yet published but will be submitted to the *International Conference on Machine Learning* (ICML).

Part II

Preliminaries

> Bevor ich fortfahre, will ich feierlich versichern, daß von jetzt ab das Fäkalthema so erledigt ist, wie Chopin schon auf Seite 189 war. Auch mit der Schilderung erzieherischer Maßnahmen bei militärischen Organisationen bin ich am Ende. Es könnte zu leicht der Verdacht entstehen, dieses Erzählwerk wäre antimilitaristisch oder gar abrüstungsfreundlich bzw. aufrüstungsfreindlich. O nein, es geht um Höheres, um – wie jeder unvoreingenommene Leser längst weiß – um die Liebe und um die Unschuld. Daß die Umstände, unter, die Details, mit denen ich beides hier zu schildern versuche, die Erwähnung gewisser Formationen, Organisationen, Institutionen notwendig macht, ist nicht meine Schuld, sondern die eines Schicksals, mit dem jeder hadern mag, soviel er Lust hat.

—Heinrich Böll, *Entfernung von der Truppe*

3 Introduction

Each part in this thesis has its own “Preliminaries” chapter containing background material specific to that part. This part contains background material that is of relevance for all or several of the parts to follow. The following elaborations assume the reader to be familiar with basics concepts of probability, such as σ-Algebra, Bayes rule and expectation. Otherwise, the work by DeGroot and Schervish (2012) is an excellent starting point. For an overview on (numerical) linear algebra, Golub and Van Loan (2013) contains most information relevant for this thesis. The definitions and properties of multivariate Gaussian random variables and Gaussian processes are fundamental to this dissertation, but likely familiar to the reader and are therefore in Appendix B.

The exposition starts with Gaussian process regression and its connection to kernel least-squares. The purpose of that chapter is to provide the reader with an example application that all parts of this thesis contribute to. Thereafter, two popular solvers for linear equation systems are introduced: the generalized minimal residual (GMRES) method and the method of conjugate gradients (CG). Part iii will provide probabilistic interpretations of these solvers and Part iv shows how to use them in combination with certain approximation methods for kernel machines. The last chapter, Chapter 6, presents the Kronecker product and its symmetric sibling which will be a major tool for Parts iii and iv.

4 Gaussian Process Regression and Regularized Least-Squares

One of the main applications, for all parts of this work, are machine learning algorithms involving operations with kernel matrices. Exemplary we will consider here regularized least-squares regression. Regularized least-squares is known under a variety of names such as kernel ridge regression (Hoerl and Kennard 1970), spline regression (e.g. Wahba (1990)), Kriging (e.g. Matheron (1973)) and Gaussian process (GP) regression (e.g. Rasmussen and Williams (2006)). The common principle is the estimation of a regression function from a reproducing kernel Hilbert space (RKHS) $f : \mathbb{X} \to \mathbb{R}$ over some domain $\mathbb{X}$ that minimizes the regularized loss (Rasmussen and Williams 2006, Eq. (6.19))

$$\mathcal{L}(f) = \frac{1}{2}\|f\|_k^2 + \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - f(x_i))^2,$$

where $(\boldsymbol{x}_i, y_i) \in \mathbb{X} \times \mathbb{R}$, $i = 1, \ldots, N$ are observations, $\sigma^2 \in \mathbb{R}^+$ is a regularization parameter, k is the corresponding kernel and $\|\cdot\|_k$ is the RKHS norm of f.

The minimizer of this loss has a closed-form solution that coincides with the posterior mean of the Gaussian process $p(f \mid \boldsymbol{X}, \boldsymbol{y}) = \mathcal{GP}(f; \bar{f}, \bar{c})$ under a zero-mean prior $p(f) = \mathcal{GP}(f; 0, k)$ and likelihood $p(\boldsymbol{y} \mid \boldsymbol{f}(\boldsymbol{X})) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{f}(\boldsymbol{X}), \sigma^2\boldsymbol{I})$ (Kimeldorf and Wahba 1970; Wahba 1990; Rasmussen and Williams 2006):

$$\bar{f}(\boldsymbol{x}_*) = \boldsymbol{k}_*^\intercal(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}, \tag{1}$$

$$\bar{c}(\boldsymbol{x}_*, \boldsymbol{x}_{**}) = k(\boldsymbol{x}_*, \boldsymbol{x}_{**}) - \boldsymbol{k}_*^\intercal(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{k}_{**} \tag{2}$$

where $\boldsymbol{K}_{ij} = k(x_i, x_j)$, and $\boldsymbol{k}_{*,i} = k(\boldsymbol{x}_*, \boldsymbol{x}_i)$. For medium-size datasets,[1] the standard approach to solve Equations (1) and (2) is to compute a Cholesky decomposition $\boldsymbol{C}$ (Benoit 1924) of the symmetric and positive definite (s. p. d.) $\boldsymbol{K} + \sigma^2\boldsymbol{I}$ at a cubic cost $\mathcal{O}(N^3)$. For larger datasets, a number of approximate algorithms have been proposed that yield an approximation $\hat{f}$ to $\bar{f}$ in linear time.[2] Comparative empirical studies like those of Chalupka, Williams, and Murray (2013) or Quiñonero-Candela and Rasmussen (2005) indicate that some of these methods can provide good approximations in a reasonable amount of time, although there is no conclusive 'best practice' among these choices.

The machine learning community tends to prefer the methods above over the linear solvers that will be introduced in Chapter 5. In compar-

[1] For currently modern machines this means roughly $N \sim 5 \cdot 10^4$ observations.

[2] Zhu, Williams, Rohwer, and Morciniec 1998; Csató and Opper 2002; Snelson and Ghahramani 2007; Walder, Kim, and Schölkopf 2008; Rahimi and Recht 2009; Titsias 2009; Lázaro-Gredilla, Quiñonero-Candela, Rasmussen, and Figueiras-Vidal 2010; Yan and Qi 2010; Le, Sarlos, and Smola 2013; Solin and Särkkä 2014; Wilson and Nickisch 2015; Hensman, Durrande, and Solin 2018.

ison the latter are computationally more expensive, and a linear solver needs to run again for new test inputs when computing the posterior uncertainty (Eq. (2)). Furthermore, Gaussian process regression often requires the evaluation of the log marginal likelihood

$$\ln p(\boldsymbol{y}) = -\frac{1}{2}\boldsymbol{y}^\intercal(\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1}\boldsymbol{y} + \frac{1}{2}\ln|2\pi(\boldsymbol{K}+\sigma^2\boldsymbol{I})|^{-1}, \tag{3}$$

e.g. for model selection and linear solvers do not provide estimates for the determinant, off-the-shelf.[3]

[3] Conjugate gradients can be used to estimate $|\boldsymbol{K}|$ (Filippone and Engler 2015), yet also requiring several runs.

5 Classic Linear Solvers

Consider the linear equation system

$$Ax^* = b \tag{LES}$$

where $A \in \mathbb{R}^{d \times d}$ is no longer s.p.d. but more generally, an invertible matrix, $b \in \mathbb{R}^d$ is a given vector and $x^* \in \mathbb{R}^d$ is an unknown to be determined. The problem can be solved exactly using for example Gaussian elimination. The costs for solving such systems in general scale as $\mathcal{O}(d^3)$. This thesis is concerned with linear equation systems so large that running exact algorithms is precluded.

Two approximation algorithms will be of particular interest: the conjugate gradient (CG) method (Hestenes and Stiefel 1952) and the generalized minimal residual (GMRES) method (Saad and Schultz 1986). The following presentation follows Saad (2003). Both methods can be derived from Arnoldi's procedure as outlined below.

5.1 Arnoldi's Procedure

Arnoldi's procedure (Saad 2003, Section 6.3) is used to construct orthonormal bases for Krylov spaces $K_m(A, r_0) = \operatorname{span}(r_0, Ar_0, ..., A^{m-1}r_0)$ of general, nonsingular matrices A and vectors r_0. Starting with $q_1 = r_0 / \|r_0\|_2$, for some vector r_0, at each iteration j, the algorithm multiplies the previous q_{j-1} by A and then orthonormalizes the resulting vector $w_j = Aq_{j-1}$ against all previous q_i, using the Gram-Schmidt procedure (cf. Algorithm 1).

Define

$$\begin{aligned} Q_m &:= \begin{bmatrix} q_1 & \dots & q_m \end{bmatrix} \in \mathbb{R}^{d \times m}, \\ h_{ij} &:= \langle Aq_j, q_i \rangle \quad 1 \leq i \leq j \leq m \text{ and} \\ h_{j+1,j} &:= \|w_i\|. \end{aligned}$$

The basis vectors satisfy the relations

$$Q_{m+1}\tilde{H}_m = AQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^\intercal \text{ and} \tag{4}$$

$$Q_m^\intercal A Q_m = H_m, \tag{5}$$

where the *upper Hessenberg* matrix $\boldsymbol{H}_m$ is defined as

$$\boldsymbol{H}_m = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1,m-1} & h_{1m} \\ h_{21} & h_{22} & h_{23} & \dots & h_{2,m-1} & h_{2m} \\ 0 & h_{32} & h_{33} & \dots & h_{3,m-1} & h_{3m} \\ \vdots & 0 & h_{43} & \dots & h_{4,m-1} & h_{3m} \\ \vdots & & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & \dots & 0 & h_{m,m-1} & h_{mm} \end{bmatrix} \in \mathbb{R}^{m\times m}$$

and

$$\tilde{\boldsymbol{H}}_m = \begin{bmatrix} \boldsymbol{H}_m \\ h_{m+1,m}\boldsymbol{e}_m^\intercal \end{bmatrix} \in \mathbb{R}^{(m+1)\times m}.$$

Algorithm 1: Arnoldi's procedure (Saad 2003, Algorithm 6.2)

```
$\boldsymbol{r}_0 \leftarrow \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$, $\beta \leftarrow \|\boldsymbol{r}_0\|_2$, $\boldsymbol{q}_1 \leftarrow \boldsymbol{r}_0/\beta$
for $j = 1, \dots, m$ do
        $h_{ij} \leftarrow \langle \boldsymbol{A}\boldsymbol{q}_j, \boldsymbol{q}_i \rangle$
        $\boldsymbol{w}_j \leftarrow \boldsymbol{A}\boldsymbol{q}_j - \sum_{i=1}^{j} h_{ij}\boldsymbol{q}_i$
    $h_{j+1,j} \leftarrow \|\boldsymbol{w}_j\|_2$
    if $h_{j+1,j} = 0$ then
        Stop
    end if
    $\boldsymbol{q}_{j+1} \leftarrow \boldsymbol{w}_j/h_{j+1,j}$
end for
Define $\tilde{\boldsymbol{H}}_m \in \mathbb{R}^{(m+1)\times m}$ with elements $h_{ij}$
```

5.2 Conjugate Gradients

For general matrices $\boldsymbol{A}$, conjugate gradients is known as Arnoldi's method for linear systems, or Full Orthoganalization Method (FOM) (Saad 2003, p. 165). For some initial guess $\boldsymbol{x}_0$, FOM computes the iterate

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{Q}_m\boldsymbol{c}_m$$

where $\boldsymbol{c}_m$ is chosen s.t. the residual $\boldsymbol{r}_m := \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m$ is orthogonal to $\boldsymbol{Q}_m$. This implies

$$\begin{aligned} \mathbf{0} &= \boldsymbol{r}_m^\intercal \boldsymbol{Q}_m \\ &= \boldsymbol{b}^\intercal \boldsymbol{Q}_m - \boldsymbol{x}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{Q}_m \\ &= \boldsymbol{b}^\intercal \boldsymbol{Q}_m - \boldsymbol{x}_0^\intercal \boldsymbol{A}^\intercal \boldsymbol{Q}_m - \boldsymbol{c}_m^\intercal \boldsymbol{Q}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{Q}_m \\ &\quad /\!/ \textit{ definition of } \boldsymbol{x}_m \end{aligned} \tag{6}$$

$$
\begin{aligned}
&= \boldsymbol{r}_0^\top \boldsymbol{Q}_m - \boldsymbol{c}_m^\top \boldsymbol{H}_m^\top \\
&\text{// \textit{definition of} } \boldsymbol{H}_m \\
&= ||\boldsymbol{r}_0||_2 \boldsymbol{e}_1 - \boldsymbol{c}_m^\top \boldsymbol{H}_m^\top \\
&\text{// \textit{by choice of} } \boldsymbol{q}_1
\end{aligned}
$$

and hence, $\boldsymbol{c}_m = ||\boldsymbol{r}_0||_2 \boldsymbol{H}_m^{-1} \boldsymbol{e}_1$. For $\boldsymbol{x}_m$ this implies

$$
\begin{aligned}
\boldsymbol{x}_m &= \boldsymbol{x}_0 + ||\boldsymbol{r}_0||_2 \boldsymbol{Q}_m^\top \boldsymbol{H}_m^{-1} \boldsymbol{e}_1, \text{ or} \\
&= \boldsymbol{x}_0 + \boldsymbol{Q}_m (\boldsymbol{Q}_m^\top \boldsymbol{A} \boldsymbol{Q}_m)^{-1} \boldsymbol{Q}_m^\top \boldsymbol{r}_0 \qquad \text{(FOM)} \\
&\text{// \textit{when stopping simplification at Eq. (6)}}
\end{aligned}
$$

where Eq. (FOM) will be relevant in Section 7.2. After running m steps of Arnoldi's procedure, solving the $m \times m$ linear equation system is possible in $\mathcal{O}(m^2)$ (Golub and Van Loan 2013, p. 179) s.t. $\boldsymbol{x}_m$ can be evaluated in $\mathcal{O}(d + m^2)$.

If the matrix $\boldsymbol{A}$ is symmetric and positive definite, the Hessenberg matrix $\boldsymbol{H}_m = \boldsymbol{Q}_m^\top \boldsymbol{A} \boldsymbol{Q}_m$ is tridiagonal which allows to simplify Arnoldi's procedure, then called symmetric Lanczos algorithm, and FOM can be simplified to conjugate gradients.

Conjugate gradients is the method that produces in each step the minimizer $\boldsymbol{x}_m$ of the function $\phi(\boldsymbol{x}) := 1/2\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{b}$ where $\boldsymbol{x} \in \boldsymbol{x}_0 + K_m(\boldsymbol{A}, \boldsymbol{r}_0)$ (Nocedal and Wright 1999, Section 5) and shown in Algorithm 2. Alternatively $\boldsymbol{x}_m$ can be written as

$$
\boldsymbol{x}_m = \operatorname*{arg\,min}_{\boldsymbol{x} \in \boldsymbol{x}_0 + K_m(\boldsymbol{A}, \boldsymbol{r}_0)} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_{\boldsymbol{A}},
$$

which is interesting for a comparison to GMRES, following soon (*c.f.* Eq. (8)).

Algorithm 2: the conjugate gradients algorithm (Hestenes and Stiefel 1952; Saad 2003, p. 199f), adapted in notation and presentation to this dissertation.

```
$r_0 \leftarrow b - Ax_0$                         // The initial residual ...
$s_0 \leftarrow r_0$                              // ... is the first search direction.
$i \leftarrow 0$
while $||r_i||_2 > \epsilon$ do
    $z_i \leftarrow As_i$                         // the most expensive step: $\mathcal{O}(d^2)$ matrix-multiplication
    $\alpha_i \leftarrow \frac{r_i^\top r_i}{s_i^\top z_i}$   // optimal linesearch along $s_i$ for $\phi(x) := x^\top Ax - 2x^\top b$
    $x_{i+1} \leftarrow x_i + \alpha_i s_i$       // update to the solution
    $r_{i+1} \leftarrow r_i - \alpha_i z_i$       // analogue update to the residual
    $s_{i+1} \leftarrow r_{i+1} + \frac{r_{i+1}^\top r_{i+1}}{r_i^\top r_i} s_i$   // Gram-Schmidt applied to the new residual
    $i \leftarrow i + 1$
end while
return $x_i$
```

The transition from FOM to CG takes the pages 196 to 200 in the book by Saad (2003). The process itself is not relevant for this work. Relevant is the fact that CG is a special case of FOM, when considering probabilistic interpretations of conjugate gradients in Sections 8.2.1 and 8.3.2. Here, we will just show that the two algorithms produce the same solutions in each step.

Proposition 1. *If $\boldsymbol{A}$ is symmetric and positive definite, FOM and CG produce the same solutions in each step.*

Proof. Denote with $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{m-1}$ the conjugate gradients directions which span the vector space $K_m(\boldsymbol{A}, \boldsymbol{r}_0)$ (Nocedal and Wright 1999, Theorem 5.3). The proof of Theorem 5.2 by Nocedal and Wright (1999, p. 106) states that $\boldsymbol{x}_m = \arg\min_{\boldsymbol{x} \in \boldsymbol{x}_0 + K_m(\boldsymbol{A}, \boldsymbol{r}_0)} \phi(\boldsymbol{x})$ iff $\boldsymbol{r}_m^\intercal \boldsymbol{s}_i$ for $i = 0, \ldots, m-1$. By construction through Arnoldi's procedure, $\boldsymbol{Q}_m$ forms a basis of $K_m(\boldsymbol{A}, \boldsymbol{r}_0)$, and by definition of FOM $\boldsymbol{Q}_m^\intercal \boldsymbol{r}_m = \boldsymbol{0}$. □

5.3 Generalized Minimal Residual

Another method derived from the Arnoldi procedure is the *Generalized Minimal Residual Method* (Saad 2003, Section 6.5). GMRES computes the iterate

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{Q}_m \boldsymbol{c}_m$$

where $\boldsymbol{c}_m$ is chosen to satisfy the optimality condition in Eq. (7).

$$\begin{aligned} \|\boldsymbol{r}_m\|_2 &= \min_{\boldsymbol{x} \in K_m(\boldsymbol{A}, \boldsymbol{r}_0)} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{r}_0\|_2 && (7) \\ &= \min_{\boldsymbol{x} \in \boldsymbol{x}_0 + K_m(\boldsymbol{A}, \boldsymbol{r}_0)} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2. && (8) \end{aligned}$$

That is, at iteration m, GMRES minimises the residual over the vector space $\boldsymbol{x}_0 + K_m(\boldsymbol{A}, \boldsymbol{r}_0)$. Hence,

$$\begin{aligned} \boldsymbol{c}_m &= \arg\min_{\boldsymbol{c} \in \mathbb{R}^m} \|\boldsymbol{A}\boldsymbol{Q}_m \boldsymbol{c} - \boldsymbol{r}_0\|_2 && (9) \\ &= \left((\boldsymbol{A}\boldsymbol{Q}_m)^\intercal (\boldsymbol{A}\boldsymbol{Q}_m)\right)^{-1} (\boldsymbol{A}\boldsymbol{Q}_m)^\intercal \boldsymbol{r}_0. \end{aligned}$$

and

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{Q}_m \left(\boldsymbol{Q}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{A} \boldsymbol{Q}_m\right)^{-1} \boldsymbol{Q}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{r}_0, \qquad \text{(GMR)}$$

where the above equation will be relevant in Section 7.2. The least-squares problem in Eq. (9) can be solved exactly in $\mathcal{O}(d + m^3)$ but GMRES solves it more efficiently via Arnoldi's method. To this end, express the starting vector in the Krylov basis,

$$\boldsymbol{r}_0 = \|\boldsymbol{r}_0\|_2 \boldsymbol{q}_1 = \|\boldsymbol{r}_0\|_2 \boldsymbol{Q}_{m+1} \boldsymbol{e}_1,$$

and exploit the Arnoldi recursion from Eq. (4),

$$\boldsymbol{A}\boldsymbol{Q}_m \boldsymbol{c} - \boldsymbol{r}_0 = \boldsymbol{Q}_{m+1} \left(\tilde{\boldsymbol{H}}_{m+1} \boldsymbol{c} - \|\boldsymbol{r}_0\|_2 \boldsymbol{e}_1\right),$$

followed by the unitary invariance of the two-norm,

$$\|\boldsymbol{A}\boldsymbol{Q}_m\boldsymbol{c} - \boldsymbol{r}_0\|_2 = \|\tilde{\boldsymbol{H}}_m\boldsymbol{c} - \|\boldsymbol{r}_0\|_2\,\boldsymbol{e}_1\|_2.$$

Thus, instead of solving the least squares problem Equation (9), GMRES solves

$$\boldsymbol{c}_m = \underset{\boldsymbol{c}\in\mathbb{R}^m}{\arg\min}\,\|\tilde{\boldsymbol{H}}_m\boldsymbol{c} - \|\boldsymbol{r}_0\|_2\,\boldsymbol{e}_1\|_2. \tag{10}$$

Solving above systems costs only $\mathcal{O}(m^2)$ (Golub and Van Loan 2013, p. 179) instead of the $\mathcal{O}(m^3)$ for the naive solution of Eq. (9).

6 Kronecker Calculus

The Kronecker product will be a vital component throughout Parts iii and iv. The Kronecker product and its symmetric version have been studied, among others, by Loan (2000) and Magnus and Neudecker (1980). The definitions used in this work slightly differ from the authors above and instead follow Hennig (2015).

The Kronecker product for two arbitrary matrices $\boldsymbol{A} \in \mathbb{R}^{N_1 \times N_2}$, $\boldsymbol{B} \in \mathbb{R}^{N_3 \times N_4}$ is defined as

$$[\boldsymbol{A} \otimes \boldsymbol{B}]_{ij,kl} := \boldsymbol{A}_{ik} \boldsymbol{B}_{jl}$$

where $i \in \{1, ..., N_1\}$, $j \in \{1, ..., N_3\}$, $k \in \{1, ..., N_2\}$ and $l \in \{1, ..., N_4\}$, and ij is not a product but a double-index. The following identities about Kronecker products and the vectorization operator can be found in Hennig and Kiefel (2013), and are restated here for the convenience of the reader:

$$\begin{aligned}
(\boldsymbol{A} \otimes \boldsymbol{B}) \operatorname{vec}(\boldsymbol{C}) &= \operatorname{vec}(\boldsymbol{A}\boldsymbol{C}\boldsymbol{B}^\intercal) && \text{(K1)}\\
(\boldsymbol{A} \otimes \boldsymbol{B})(\boldsymbol{C} \otimes \boldsymbol{D}) &= (\boldsymbol{A}\boldsymbol{C}) \otimes (\boldsymbol{B}\boldsymbol{D}) && \text{(K2)}\\
(\boldsymbol{A} \otimes \boldsymbol{B})^{-1} &= \boldsymbol{A}^{-1} \otimes \boldsymbol{B}^{-1} && \text{(K3)}\\
(\boldsymbol{A} \otimes \boldsymbol{B})^\intercal &= \boldsymbol{A}^\intercal \otimes \boldsymbol{B}^\intercal && \text{(K4)}\\
(\boldsymbol{A} + \boldsymbol{B}) \otimes \boldsymbol{C} &= \boldsymbol{A} \otimes \boldsymbol{C} + \boldsymbol{B} \otimes \boldsymbol{C} && \text{(K5)}
\end{aligned}$$

where[1] $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \in \mathbb{R}^{N \times N}$, and $\boldsymbol{A}$ and $\boldsymbol{B}$ are assumed to be invertible. An appealing property of Kronecker-structured matrices is their interaction with vectorized matrices. For a square matrix $\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_1 & \dots & \boldsymbol{a}_N \end{bmatrix}^\intercal \in \mathbb{R}^{N \times N}$, the operator $\operatorname{vec}(\,) : \mathbb{R}^{N \times N} \to \mathbb{R}^{N^2}$ stacks the rowsof $\boldsymbol{A}$ into one vector:

$$\operatorname{vec}(\boldsymbol{A}) = \begin{bmatrix} \boldsymbol{a}_1 \\ \vdots \\ \boldsymbol{a}_N \end{bmatrix}, \quad \text{with} \quad [\operatorname{vec}(\boldsymbol{A})]_{(ij)} = [\boldsymbol{A}]_{ij}$$

and $\operatorname{mat}(\,) : \mathbb{R}^{N^2} \to \mathbb{R}^N \times \mathbb{R}^N$ transforms an N^2 vector into an $N \times N$ matrix, s.t. $\operatorname{mat}(\operatorname{vec}(\boldsymbol{A})) = \boldsymbol{A}$. A vector product of vectorized matrices corresponds to the trace of their product:

$$\operatorname{vec}(\boldsymbol{A})^\intercal \operatorname{vec}(\boldsymbol{B}) = \operatorname{tr} \boldsymbol{A}\boldsymbol{B}^\intercal. \qquad \text{(V1)}$$

[1] The conditions can be more general but for ease of exposition, we assume all matrices are square and of equal size.

Proof.

$$\operatorname{tr} \boldsymbol{A}\boldsymbol{B}^\intercal = \sum_i [\boldsymbol{A}\boldsymbol{B}^\intercal]_{ii} = \sum_{i,j} \boldsymbol{A}_{ij}\boldsymbol{B}^\intercal_{ji} = \sum_{i,j} \boldsymbol{A}_{ij}\boldsymbol{B}_{ij} = \operatorname{vec}(\boldsymbol{A})^\intercal \operatorname{vec}(\boldsymbol{B})$$

□

The *symmetric* Kronecker product for two square matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{N \times N}$ of equal size is defined as

$$\boldsymbol{A} \circledast \boldsymbol{B} := \boldsymbol{\Gamma}_N(\boldsymbol{A} \otimes \boldsymbol{B})\boldsymbol{\Gamma}_N$$

where $[\boldsymbol{\Gamma}_N]_{ij,kl} := {}^1\!/_2\delta_{ik}\delta_{jl} + {}^1\!/_2\delta_{il}\delta_{jk}$ satisfies

$$\boldsymbol{\Gamma} \operatorname{vec}(\boldsymbol{C}) = {}^1\!/_2 \operatorname{vec}(\boldsymbol{C}) + {}^1\!/_2 \operatorname{vec}(\boldsymbol{C}^\intercal)$$

for all square-matrices $\boldsymbol{C} \in \mathbb{R}^{N \times N}$. Equivalently, one can write

$$(\boldsymbol{A} \circledast \boldsymbol{B})_{ij,kl} = \frac{1}{4}\left(\boldsymbol{A}_{ik}\boldsymbol{B}_{jl} + \boldsymbol{A}_{il}\boldsymbol{B}_{jk} + \boldsymbol{B}_{ik}\boldsymbol{A}_{jl} + \boldsymbol{B}_{il}\boldsymbol{A}_{jk}\right).$$

The symmetric Kronecker product inherits some of the desirable properties of the Kronecker product. Some of the following identities can, again, be found in Hennig (2015), some are due to Loan (2000) and Magnus and Neudecker (1980) and some are novel. The proof gives exact credit.

Proposition 2. *Let* $\boldsymbol{V}, \boldsymbol{W} \in \mathbb{R}^{N \times N}$ *be square matrices and* $\boldsymbol{A}^\intercal, \boldsymbol{B} \in \mathbb{R}^{N \times M}$ *be rectangular.*

$$\boldsymbol{W} \circledast \boldsymbol{W} = \boldsymbol{\Gamma}_N(\boldsymbol{W} \otimes \boldsymbol{W}) \tag{SK1}$$

$$\boldsymbol{\Gamma}_M(\boldsymbol{A} \otimes \boldsymbol{A}) = (\boldsymbol{A} \otimes \boldsymbol{A})\boldsymbol{\Gamma}_N \tag{SK2}$$

$$\boldsymbol{V} \circledast \boldsymbol{W} = \boldsymbol{W} \circledast \boldsymbol{V} \tag{SK3}$$

$$(\boldsymbol{A} \otimes \boldsymbol{A})(\boldsymbol{W} \circledast \boldsymbol{W})(\boldsymbol{B} \otimes \boldsymbol{B}) = (\boldsymbol{A}\boldsymbol{W}\boldsymbol{B}) \circledast (\boldsymbol{A}\boldsymbol{W}\boldsymbol{B}) \tag{SK4}$$

$$\boldsymbol{W} \circledast \boldsymbol{W} - \boldsymbol{V} \circledast \boldsymbol{V} = (\boldsymbol{W} + \boldsymbol{V}) \circledast (\boldsymbol{W} - \boldsymbol{V}) \tag{SK5}$$

$$(\boldsymbol{W} \circledast \boldsymbol{W})^{-1} = (\boldsymbol{W}^{-1} \circledast \boldsymbol{W}^{-1}). \tag{SK6}$$

The interpretation of Eq. (SK6) *requires some care: symmetric Kronecker product matrices are rank deficient. Eq.* (SK6) *is to be read in the sense that for symmetric* $\boldsymbol{Y} \in \mathbb{R}^{N \times N}$, i.e. $\boldsymbol{Y} = \boldsymbol{Y}^\intercal$, $\boldsymbol{X} := \operatorname{mat}\left((\boldsymbol{W}^{-1} \circledast \boldsymbol{W}^{-1}) \operatorname{vec}(\boldsymbol{Y})\right)$ *satisfies* $\operatorname{vec}(\boldsymbol{Y}) = (\boldsymbol{W} \circledast \boldsymbol{W}) \operatorname{vec}(\boldsymbol{X})$ *and* $\boldsymbol{X}$ *is the unique symmetric solution.*

The proof is part of Appendix C.

Part III

Probabilistic Linear Solvers

> Spätestens hier wird der kluge Leser wissen, was wir, er und ich, nun auch dem weniger klugen Leser nicht länger vorenthalten sollten: daß dieses Erzählwerk wirklich eine reine Idylle werden soll, in der Kloakendüfte dieselbe Funktion haben wie anderswo Rosendüfte, in der die Auseinandersetzung mit dem Krieg vermieden oder zumindest sehr reduziert wird, die Nazi-Angelegenheit wie etwas zwischen Schnupfen und Schwefelregen abgetan werden soll, [...]

—Heinrich Böll, *Entfernung von der Truppe*

7 Preliminaries

The first part of this thesis is concerned with the question how classic linear solvers are connected to probabilistic numerical methods (PNM). The motivation to answer this question is to gain a deeper understanding of linear solvers; which assumptions are implicitly encoded and under which circumstances, which solvers are preferable.

7.1 Introduction

The first publications, that show how to construct probabilistic linear solvers are those by Hennig and Kiefel (2012), Hennig and Kiefel (2013) and Hennig (2015). Their focus is mainly on the connection between Gaussian inference and optimization—the only linear solver analyzed is the conjugate gradient (CG) method. Later Cockayne, Oates, Ipsen, and Girolami (2018) proposed a (seemingly) different approach, also with a focus on CG. CG is but a single member of a larger class of iterative solvers, and applicable only if the matrix $\boldsymbol{A}$ is symmetric and positive-definite. This part is based on the publication "Probabilistic Linear Solvers: A Unifying View" (Bartels, Cockayne, Ipsen, and Hennig 2019) which explored the connection for a larger class of solvers, in particular, projection methods. As a step towards a probabilistic understanding of projection methods, the goal of that publication was, for a given projection method, to find possible prior assumptions and information[1] s.t. the posterior mean estimate coincides with the solution of the projection method.

Projection methods will be introduced in Section 7.2. Section 7.3 presents existing approaches to construct probabilistic linear solvers. Thereafter, Chapter 8 shows connections between projection methods and probabilistic linear solvers.

"Probabilistic Linear Solvers: A Unifying View" is a joint publication, and Propositions 8, 15 and 16 and Corollary 9 are contributions by my collaborators, which I will point out again in due time.

7.2 Projection Methods

Recall Eq. (LES) which describes the system of linear equations

$$\boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{b} \tag{LES}$$

[1] The meaning of "information" will be defined properly Section 7.3.

where $\boldsymbol{A} \in \mathbb{R}^{d\times d}$ is an invertible matrix, $\boldsymbol{b} \in \mathbb{R}^d$ is a given vector and $\boldsymbol{x}^* \in \mathbb{R}^d$ is an unknown to be determined. Denote with $\boldsymbol{x}_0$ an initial guess and with $\boldsymbol{r}_m := \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m$ the residual for an approximation $\boldsymbol{x}_m$.

Many iterative methods for linear systems belong to the class of projection methods (Saad 2003, p. 130f.), including popular linear solvers such as the conjugate gradients (CG) (see Section 5.2) method and the generalized minimal residual (GMRES) method (see Section 5.3). Saad describes a projection method as an iterative scheme in which, at each iteration, a solution vector $\boldsymbol{x}_m$ is constructed by projecting $\boldsymbol{x}^*$ into a *solution space* $\mathbb{X}_m \subset \mathbb{R}^d$, subject to the restriction that the residual $\boldsymbol{r}_m = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_m$ is orthogonal to a *constraint space* $\mathbb{U}_m \subset \mathbb{R}^d$.

More formally, each iteration of a projection method is defined by two matrices $\boldsymbol{X}_m, \boldsymbol{U}_m \in \mathbb{R}^{d\times m}$, and by a starting point $\boldsymbol{x}_0$. The matrices $\boldsymbol{X}_m$ and $\boldsymbol{U}_m$ each encode the solution and constraint spaces as $\mathbb{X}_m = \mathsf{range}(\boldsymbol{X}_m)$ and $\mathbb{U}_m = \mathsf{range}(\boldsymbol{U}_m)$. The projection method then constructs $\boldsymbol{x}_m$ as $\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{X}_m\boldsymbol{\alpha}_m$ with $\boldsymbol{\alpha}_m \in \mathbb{R}^m$ determined by the constraint $\boldsymbol{U}_m^\intercal \boldsymbol{r}_m = \mathbf{0}$. If $\boldsymbol{U}_m^\intercal \boldsymbol{A}\boldsymbol{X}_m$ is nonsingular, one obtains

$$\begin{aligned}\boldsymbol{\alpha}_m &= (\boldsymbol{U}_m^\intercal \boldsymbol{A}\boldsymbol{X}_m)^{-1}\boldsymbol{U}_m^\intercal \boldsymbol{r}_0, \text{ and thus}\\ \boldsymbol{x}_m &= \boldsymbol{x}_0 + \boldsymbol{X}_m(\boldsymbol{U}_m^\intercal \boldsymbol{A}\boldsymbol{X}_m)^{-1}\boldsymbol{U}_m^\intercal \boldsymbol{r}_0. \end{aligned} \tag{P}$$

Eq. (P) is tagged with a special letter as it will be referenced frequently in Chapter 8 when examining probabilistic interpretations of projection methods. Observe that another way to express Eq. (P) is

$$\begin{aligned}\boldsymbol{x}_m &= \boldsymbol{x}_0 + \boldsymbol{P}(\boldsymbol{x}^* - \boldsymbol{x}_0), \text{ where}\\ \boldsymbol{P} &:= \boldsymbol{X}_m(\boldsymbol{U}_m^\intercal \boldsymbol{A}\boldsymbol{X}_m)^{-1}\boldsymbol{U}_m^\intercal \boldsymbol{A},\end{aligned}$$

and note further that $\boldsymbol{P}$ is idempotent, *i.e.* $\boldsymbol{P}\boldsymbol{P} = \boldsymbol{P}$. Thus $\boldsymbol{P}$ is a linear projection, hence the name, projection method.

Differing from the presentation of FOM/CG and GMRES in Chapter 5, in the projection method perspective, both algorithms perform only a single step with the size of the Krylov subspace m fixed and determined in advance. For FOM the spaces are $\mathbb{U}_m = \mathbb{X}_m = K_m(\boldsymbol{A}, \boldsymbol{b})$, while for GMRES they are $\mathbb{X}_m = K_m(\boldsymbol{A}, \boldsymbol{b})$ and $\mathbb{U}_m = AK_m(\boldsymbol{A}, \boldsymbol{b})$ (compare Eq. (P) with Eqs. (FOM) and (GMR) on page 15 and page 16, respectively).

7.3 Constructing Probabilistic Linear Solvers

Hennig (2015) and Cockayne, Oates, Ipsen, and Girolami (2018) proposed two seemingly different approaches to probabilistic linear solvers. In the matrix-based inference (MBI) approach of Hennig (2015), a probability measure is constructed on the matrix $\boldsymbol{A}^{-1}$, while the solution-based inference (SBI) method of Cockayne, Oates, Ipsen, and Girolami (2018) constructs a measure on the solution vector $\boldsymbol{x}^*$.

7.3.1 Solution-Based Inference

To phrase the solution of Eq. (LES) as a form of probabilistic inference, Cockayne, Oates, Ipsen, and Girolami (2018) consider a Gaussian prior over the solution $\boldsymbol{x}^*$, and condition on observations provided by a set of *search directions* $\boldsymbol{s}_1, \dots, \boldsymbol{s}_m$, $m < d$. Let $\boldsymbol{S}_m \in \mathbb{R}^{d \times m}$ be given by $\boldsymbol{S}_m := [\boldsymbol{s}_1, \dots, \boldsymbol{s}_m]$, and let information be given by $\boldsymbol{y}_m := \boldsymbol{S}_m^\intercal \boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{S}_m^\intercal \boldsymbol{b}$. Since the information is a linear projection of $\boldsymbol{x}^*$, the posterior distribution is a Gaussian distribution on $\boldsymbol{x}^*$ (*c.f.* Lemma 35 in Appendix B):

Lemma 3 (Cockayne, Oates, Ipsen, and Girolami (2018))**.** *Assume that the columns of S_m are linearly independent. Consider the prior*

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0).$$

The posterior from SBI is then given by

$$p(\boldsymbol{x} \mid \boldsymbol{y}_m) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_m, \boldsymbol{\Sigma}_m)$$

where

$$\begin{aligned} \boldsymbol{x}_m &= \boldsymbol{x}_0 + \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{S}_m (\boldsymbol{S}_m^\intercal \boldsymbol{A} \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{S}_m)^{-1} \boldsymbol{S}_m^\intercal \boldsymbol{r}_0 \\ \boldsymbol{\Sigma}_m &= \boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{S}_m (\boldsymbol{S}_m^\intercal \boldsymbol{A} \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{S}_m)^{-1} \boldsymbol{S}_m^\intercal \boldsymbol{\Sigma}_0, \end{aligned} \tag{SBI}$$

and $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$.

The naive evaluation of Eq. (SBI) requires $\mathcal{O}(m^3)$ floating point operations, due to the inversion of $\boldsymbol{Y}_m^\intercal \boldsymbol{W}_0 \boldsymbol{Y}_m$, which is still less than for the original problem Eq. (LES). If $\boldsymbol{S}_m$ is chosen $\boldsymbol{A}^\intercal \boldsymbol{\Sigma}_0 \boldsymbol{A}$-orthogonal, *e.g.* by applying Gram-Schmidt orthogonalization, the solution can be updated progressively from the last guess, avoiding the $\mathcal{O}(m^3)$ costs in each step.

7.3.2 Matrix-Based Inference

In contrast to SBI, the MBI approach of Hennig (2015) treats the matrix inverse $\boldsymbol{A}^{-1}$ as the unknown in the inference procedure. As in the previous section, search directions $\boldsymbol{S}_m$ yield matrix-vector products $\boldsymbol{Y}_m \in \mathbb{R}^{d \times m}$. In Hennig (2015) these arise from *right*-multiplying[2] $\boldsymbol{A}$ with $\boldsymbol{S}_m$, *i.e.* $\boldsymbol{Y}_m = \boldsymbol{A}\boldsymbol{S}_m$. Note that

$$\boldsymbol{S}_m = \boldsymbol{A}^{-1}\boldsymbol{Y}_m, \text{ or, equivalently } \operatorname{vec}(\boldsymbol{S}_m) = (\boldsymbol{I} \otimes \boldsymbol{Y}_m^\intercal) \operatorname{vec}\left(\boldsymbol{A}^{-1}\right). \tag{11}$$

[2] This work also considers a model class that explicitly encodes *symmetry* of $\boldsymbol{A}$, such that the distinction between left- and right- multiplication vanishes. Proposition 15 in Section 8.3.2 will make use of this model class.

Thus $\boldsymbol{S}_m$ is a linear transformation of $\boldsymbol{A}^{-1}$ and the posterior is again distributed Gaussian:

Lemma 4 (Lemma 2.1 in Hennig (2015))**.** *Consider the prior*

$$p\left(\operatorname{vec}\left(\boldsymbol{A}^{-1}\right)\right) = \mathcal{N}\left(\operatorname{vec}\left(\boldsymbol{A}^{-1}\right); \operatorname{vec}\left(\boldsymbol{A}_0^{-1}\right), \boldsymbol{\Sigma}_0 \otimes \boldsymbol{W}_0\right). \tag{12}$$

Then the posterior given the observations $\operatorname{vec}(\boldsymbol{S}_m) = \boldsymbol{A}^{-1}\boldsymbol{Y}_m$ *is given by*

$$p\left(\operatorname{vec}\left(\boldsymbol{A}^{-1}\right) \,\middle|\, \operatorname{vec}(\boldsymbol{S}_m)\right) = \mathcal{N}\left(\operatorname{vec}\left(\boldsymbol{A}^{-1}\right); \operatorname{vec}\left(\boldsymbol{A}_m^{-1}\right), \boldsymbol{\Sigma}_0 \otimes \boldsymbol{W}_m\right)$$

with

$$\begin{aligned}\boldsymbol{A}_m^{-1} &= \boldsymbol{A}_0^{-1} + (\boldsymbol{S}_m - \boldsymbol{A}_0^{-1}\boldsymbol{Y}_m)(\boldsymbol{Y}_m^\top \boldsymbol{W}_0 \boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\top \boldsymbol{W}_0\\ \boldsymbol{W}_m &= \boldsymbol{W}_0 - \boldsymbol{W}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\top \boldsymbol{W}_0 \boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\top \boldsymbol{W}_0.\end{aligned}$$

For linear solvers, the object of interest is $\boldsymbol{x}^* = \boldsymbol{A}^{-1}\boldsymbol{b}$. Writing $\boldsymbol{A}^{-1}\boldsymbol{b} = (\boldsymbol{I} \otimes \boldsymbol{b}^\top)\operatorname{vec}\left(\boldsymbol{A}^{-1}\right)$, and using Lemma 34 in Appendix B, the associated marginal is also Gaussian, and given by

$$p(\boldsymbol{x} \mid \boldsymbol{S}, \boldsymbol{Y}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{A}_m^{-1}\boldsymbol{b}, \boldsymbol{b}^\top \boldsymbol{W}_m \boldsymbol{b} \cdot \boldsymbol{\Sigma}_0). \tag{MBI}$$

Again, the naive evaluation of Eq. (MBI) requires $\mathcal{O}(m^3)$ floating point operations, which can be avoided with Gram-Schmidt orthogonalization.

The Kronecker structure of the prior covariance matrix in Eq. (12) is by no means the only option that facilitates tractable inference. For example, a diagonal covariance matrix would allow efficient inference, as well. However, in absence of literature exploring other approaches within MBI, throughout MBI will refer to the use of a prior covariance matrix with Kronecker structure.

Since the typical approximate linear solver does not construct a matrix inverse, the SBI view appears to be preferable. Also, MBI has more model parameters than SBI. However, unlike in SBI, the information obtained in MBI need not be specific to a particular solution vector $\boldsymbol{x}^*$ and thus can be propagated and recycled over several linear problems, similar to the notion of subspace recycling (Soodhalter, Szyld, and Xue 2014). Furthermore, MBI is able to utilize the information $\boldsymbol{y}_m := \boldsymbol{S}_m^\top \boldsymbol{b}$, as well. In fact, this observation will be the key in the following chapter, where it is shown that SBI is subsumed in MBI.

8

Probabilistic Interpretation of Projection Methods

The insights presented in Bartels, Cockayne, Ipsen, and Hennig (2019) can be summarized into three contributions. First, solution-based inference is a special case of matrix-based inference (Section 8.1). Second, the derivations in that section reveal that two points of view are useful to reason about probabilistic interpretations of projection methods: whether information stems from left-multiplication (Section 8.2)

$$\boldsymbol{Y}^{\intercal} := \boldsymbol{S}^{\intercal}\boldsymbol{A}, \tag{L}$$

or right-multiplication (Section 8.3)

$$\boldsymbol{Y} := \boldsymbol{A}\boldsymbol{S}, \tag{R}$$

with $\boldsymbol{A}$.

To evaluate the expression for a projection method approximation (Eq. (**P**)) both forms of multiplication are possible. In practice, the implementations of GMRES and CG only perform right-multiplications, yet, astonishingly, it was easier to find results presented in the right-multiplied perspective and these are more general than for left-multiplication. The third contribution is a probabilistic understanding of preconditioning (Section 8.4).

8.1 Matrix-based Inference and Solution-based Inference

One might suspect SBI and MBI to be equivalent, yet the posterior from Lemma 4 is structurally different to the posterior in Lemma 3. For the former, $\boldsymbol{x}_m \in \boldsymbol{x}_0 + \operatorname{span}(\boldsymbol{S}_m - \boldsymbol{A}_0^{-1}\boldsymbol{A}\boldsymbol{S}_m)$ whereas for the latter $\boldsymbol{x}_m \in \boldsymbol{x}_0 + \operatorname{span}(\boldsymbol{\Sigma}_0\boldsymbol{A}^{\intercal}\boldsymbol{S}_m)$. However, the posterior over the solution vector from MBI can be made to coincide with the posterior from SBI, if one considers observations in MBI as $\boldsymbol{S}_m^{\intercal} = \boldsymbol{Y}_m^{\intercal}\boldsymbol{A}^{-1}$.

Proposition 5. *Consider a Gaussian MBI prior*

$$p(\boldsymbol{A}^{-1}) = \mathcal{N}(\boldsymbol{A}^{-1}; \operatorname{vec}\left(\boldsymbol{A}_0^{-1}\right), \boldsymbol{\Sigma}_0 \otimes \boldsymbol{W}_0),$$

conditioned on the left-multiplied information of Eq. (L). The associated marginal on $\boldsymbol{x}$ (Eq. (MBI)) is identical to the SBI posterior on $\boldsymbol{x}$ arising in Lemma 3 from $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0)$, under the conditions

$$\boldsymbol{A}_0^{-1}\boldsymbol{b} = \boldsymbol{x}_0 \quad \textit{and} \quad \boldsymbol{b}^{\intercal}\boldsymbol{W}_0\boldsymbol{b} = 1.$$

The proof can be found in Appendix D. The first condition can be fulfilled for arbitrary $\boldsymbol{x}_0 \neq \mathbf{0}$ by defining $\boldsymbol{A}_0^{-1} := \boldsymbol{x}_0(\boldsymbol{x}_0^\top \boldsymbol{b})^{-1}\boldsymbol{x}_0^\top$, or if $\boldsymbol{x}_0 = \mathbf{0}$ by definining $\boldsymbol{A}_0^{-1} := \mathbf{0}$. The second condition can be enforced for an arbitrary covariance $\bar{\boldsymbol{W}}_0$ by setting $\boldsymbol{W}_0 := (\boldsymbol{b}^\top \bar{\boldsymbol{W}}_0 \boldsymbol{b})^{-1}\bar{\boldsymbol{W}}_0$. The result in Proposition 5 shows that any result proven for SBI applies immediately to MBI with left-multiplied observations.

8.2 Left-information Views

In this section we first show, in Proposition 6, that the conditional mean from SBI after m steps corresponds to some projection method. Then, in Proposition 7 we prove the converse: that each projection method is also the posterior mean of a probabilistic method, for some prior covariance and choice of information. After these general results, the focus will be on left-information views on CG and GMRES.

Proposition 6 (SBI defines projection methods)**.** *Let the columns of $\boldsymbol{S}_m$ be linearly independent. Consider SBI under the prior*

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0),$$

and with observations $\boldsymbol{y}_m = \boldsymbol{S}_m^\top \boldsymbol{b}$. Then the posterior mean $\boldsymbol{x}_m$ in Lemma 3 is identical to the iterate from a projection method defined by the matrices $\boldsymbol{U}_m = \boldsymbol{S}_m$ and $\boldsymbol{X}_m = \boldsymbol{\Sigma}_0 \boldsymbol{A}^\top \boldsymbol{S}_m$, and the starting vector $\boldsymbol{x}_0$.

Proof. Substituting $\boldsymbol{U}_m = \boldsymbol{S}_m$ and $\boldsymbol{X}_m = \boldsymbol{\Sigma}_0 \boldsymbol{A}^\top \boldsymbol{S}_m$ into Lemma 3 gives Eq. (**P**), as required. □

The converse to this also holds:

Proposition 7. *Consider a projection method defined by the matrices $\boldsymbol{X}_m, \boldsymbol{U}_m \in \mathbb{R}^{d \times m}$, each with linearly independent columns, and the starting vector $\boldsymbol{x}_0 \in \mathbb{R}^d$. Then the iterate $\boldsymbol{x}_m$ in Eq. (**P**) is identical to the SBI posterior mean in Lemma 3 under the prior*

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{X}_m \boldsymbol{X}_m^\top) \tag{13}$$

when search directions $\boldsymbol{S}_m = \boldsymbol{U}_m$ are used.

Proof. Abbreviate $\boldsymbol{Z} = \boldsymbol{X}_m^\top \boldsymbol{A}^\top \boldsymbol{U}_m$ and write the projection method iterate from Eq. (**P**) as

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{X}_m \boldsymbol{Z}^{-\top} \boldsymbol{U}_m^\top \boldsymbol{r}_0.$$

Multiply the middle matrix by the identity,

$$\begin{aligned} \boldsymbol{Z}^{-\top} &= \boldsymbol{Z}\boldsymbol{Z}^{-1}\boldsymbol{Z}^{-\top} = \boldsymbol{Z}(\boldsymbol{Z}^\top \boldsymbol{Z})^{-1} \\ &= \boldsymbol{X}_m^\top \boldsymbol{A}^\top \boldsymbol{U}_m (\boldsymbol{U}_m^\top \boldsymbol{A} \boldsymbol{\Sigma}_0 \boldsymbol{A}^\top \boldsymbol{U}_m)^{-1}, \end{aligned}$$

and insert this into the expression for $\boldsymbol{x}_0$,

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{U}_m (\boldsymbol{U}_m^\intercal \boldsymbol{A} \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{U}_m)^{-1} \boldsymbol{U}_m^\intercal \boldsymbol{r}_0.$$

Setting $\boldsymbol{U}_m = \boldsymbol{S}_m$ gives the mean in Lemma 3. □

A direct way to enforce the posterior occupying the solution space is by placing a prior on the coefficients $\boldsymbol{\alpha}$ in $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{X}_m \boldsymbol{\alpha}$. Under a unit Gaussian prior $\boldsymbol{\alpha} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, the implied prior on $\boldsymbol{x}$ naturally has the form of Eq. (13).

However, this prior is unsatisfying since it requires the solution space to be specified *a-priori*, precluding adaptivity in the algorithm and perhaps more worryingly, the posterior uncertainty over the solution is a matrix of zeros even though the solution is not fully identified. Again taking $\boldsymbol{Z} = \boldsymbol{X}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{U}_m$:

$$\begin{aligned}
\boldsymbol{\Sigma}_m &= \boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{U}_m (\boldsymbol{U}_m^\intercal \boldsymbol{A} \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{U}_m)^{-1} \boldsymbol{U}_m^\intercal \boldsymbol{A} \boldsymbol{\Sigma}_0 \\
&= \boldsymbol{X}_m \boldsymbol{X}_m^\intercal - \boldsymbol{X}_m \boldsymbol{Z} (\boldsymbol{Z}^\intercal \boldsymbol{Z})^{-1} \boldsymbol{Z}^\intercal \boldsymbol{X}_m^\intercal \\
&= \boldsymbol{X}_m \boldsymbol{X}_m^\intercal - \boldsymbol{X}_m \boldsymbol{X}_m^\intercal \\
&= \boldsymbol{0}.
\end{aligned}$$

Concerning this issue, Hennig (2015) and Bartels and Hennig (2016) each proposed to adding additional uncertainty in the null space of $\boldsymbol{X}_m$. This empirical uncertainty calibration step has not yet been analyzed in detail. Such analysis is left for future work.

Including the solution space $\boldsymbol{X}_m$ in the prior covariance matrix requires it to be specified *a-priori*. For solvers like CG and GMRES which construct $\boldsymbol{X}_m$ adaptively this assumption may appear problematic—a probabilistic interpretation should use for inference only quantities that have already been computed. From a projection method perspective, the computation of $\boldsymbol{X}_m$ is part of the initialization, but practically such methods choose m adaptively by examining the norm of the residual.[1] Nevertheless, the proposition provides a probabilistic view for *arbitrary* projection methods and does not involve $\boldsymbol{A}^{-1}$, unlike the results presented in Hennig (2015) and Cockayne, Oates, Sullivan, and Girolami (2017).

The above prior is not unique. The next proposition establishes probabilistic interpretations of projection methods under priors that are indepedent of solution- and constraint-space, albeit under more restrictive conditions. The benefit of this is that m need not be fixed *a-priori*. The following proposition is a contribution by Jon Cockayne.

Proposition 8. *Consider a projection method defined by* $\boldsymbol{X}_m, \boldsymbol{U}_m \in \mathbb{R}^{d \times m}$ *and the starting vector* $\boldsymbol{x}_0$. *Further suppose that* $\boldsymbol{U}_m = \boldsymbol{R}\boldsymbol{X}_m$ *for*

[1] Sometimes m is fixed *a-priori*, due to memory or computation time limits.

some invertible $\boldsymbol{R} \in \mathbb{R}^{d\times d}$*, and that* $\boldsymbol{A}^\intercal \boldsymbol{R}$ *is symmetric positive-definite. Then under the prior*

$$p(\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{x}_0, (\boldsymbol{A}^\intercal \boldsymbol{R})^{-1}\right)$$

and the search directions $\boldsymbol{S}_m = \boldsymbol{U}_m = \boldsymbol{R}\boldsymbol{X}_m$*, the iterate in the projection method is identical to the posterior mean in Lemma 3.*

Proof. First substitute $\boldsymbol{X}_m = \boldsymbol{R}^{-1}\boldsymbol{U}_m$ into Eq. (**P**) to obtain

$$\begin{aligned}\boldsymbol{x}_m &= \boldsymbol{x}_0 + \boldsymbol{R}^{-1}\boldsymbol{U}_m(\boldsymbol{U}_m^\intercal \boldsymbol{A}\boldsymbol{R}^{-1}\boldsymbol{U}_m)^{-1}\boldsymbol{U}_m^\intercal \boldsymbol{r}_0 \\ &= \boldsymbol{x}_0 + \boldsymbol{R}^{-1}\boldsymbol{A}^{-\top}\boldsymbol{A}^\intercal \boldsymbol{U}_m(\boldsymbol{U}_m^\intercal \boldsymbol{A}\boldsymbol{R}^{-1}\boldsymbol{A}^{-\top}\boldsymbol{A}^\intercal \boldsymbol{U}_m)^{-1}\boldsymbol{U}_m^\intercal \boldsymbol{r}_0 \\ &= \boldsymbol{x}_0 + \boldsymbol{\Sigma}_0\boldsymbol{A}^\intercal \boldsymbol{U}_m(\boldsymbol{U}_m^\intercal \boldsymbol{A}\Sigma_0\boldsymbol{A}^\intercal \boldsymbol{U}_m)^{-1}\boldsymbol{U}_m^\intercal \boldsymbol{r}_0.\end{aligned}$$

The third line uses $\boldsymbol{\Sigma}_0 = (\boldsymbol{A}^\intercal \boldsymbol{R})^{-1} = \boldsymbol{R}^{-1}\boldsymbol{A}^{-\intercal}$. This is equivalent to the posterior mean in Eq. (SBI) with $\boldsymbol{S}_m = \boldsymbol{U}_m$. □

A corollary which provides further insight arises when one considers the *polar decomposition* of $\boldsymbol{A}$. Recall that an invertible matrix $\boldsymbol{A}$ has a unique polar decomposition $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{H}$, where $\boldsymbol{P} \in \mathbb{R}^{d\times d}$ is orthogonal and $\boldsymbol{H} \in \mathbb{R}^{d\times d}$ is symmetric positive-definite. The following corollary is a contribution by Ilse Ipsen.

Corollary 9. *Consider a projection method defined by* $\boldsymbol{X}_m, \boldsymbol{U}_m \in \mathbb{R}^{d\times m}$ *and the starting vector* $\boldsymbol{x}_0$*, and suppose that* $\boldsymbol{U}_m = \boldsymbol{P}\boldsymbol{X}_m$*, where* $\boldsymbol{P}$ *arises from the polar decomposition* $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{H}$*. Then under the prior*

$$p(\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{H}^{-1}\right)$$

and the search directions $\boldsymbol{S}_m = \boldsymbol{U}_m = \boldsymbol{P}\boldsymbol{X}_m$*, the iterate in the projection method is identical to the posterior mean in Lemma 3.*

Proof. This follows from Proposition 8. Setting $\boldsymbol{R} = \boldsymbol{P}$ aligns the search directions in Corollary 9 with those in Proposition 8. Since $\boldsymbol{P}$ is orthogonal, $\boldsymbol{P}^{-1} = \boldsymbol{P}^\intercal$, and since $\boldsymbol{H}$ is symmetric positive-definite, $\boldsymbol{A}^\intercal \boldsymbol{P} = \boldsymbol{P}^\intercal \boldsymbol{A} = \boldsymbol{H}$ by definition of the polar decomposition, which gives the prior covariance required for Proposition 8. □

This is an intuitive analogue of similar results by Hennig (2015) and Cockayne, Oates, Sullivan, and Girolami (2017) which show that CG is recovered under certain conditions involving a prior $\boldsymbol{\Sigma}_0 = \boldsymbol{A}^{-1}$.

When $\boldsymbol{A}$ is not symmetric and positive definite it cannot be used as a prior covariance. This corollary suggests a natural way to select a prior covariance still linked to the linear system, though this choice is still not computationally convenient. Furthermore, in the case that $\boldsymbol{A}$ is symmetric positive-definite, this recovers the prior which replicates CG: note that each of $\boldsymbol{H}$ and $\boldsymbol{P}$ can be stated explicitly as $\boldsymbol{H} = (\boldsymbol{A}^\intercal \boldsymbol{A})^{\frac{1}{2}}$ and $\boldsymbol{P} = \boldsymbol{A}(\boldsymbol{A}^\intercal \boldsymbol{A})^{-\frac{1}{2}}$. Thus in the case of symmetric positive-definite $\boldsymbol{A}$ we have that $\boldsymbol{H} = \boldsymbol{A}$ and $\boldsymbol{P} = \boldsymbol{I}$, so that the prior covariance $\boldsymbol{\Sigma}_0 = \boldsymbol{A}^{-1}$ arises naturally from this interpretation.

8.2.1 Conjugate Gradients

Recall from Section 5.2 that conjugate gradients can be seen as projection method with $\boldsymbol{X}_m = \boldsymbol{U}_m = \boldsymbol{Q}_m$, where $\boldsymbol{Q}_m$ is a basis of $K_m(\boldsymbol{A}, \boldsymbol{r}_0)$. A left-multiplied probabilistic interpretation for CG has been presented by Cockayne, Oates, Ipsen, and Girolami (2018) for the choice $\boldsymbol{\Sigma}_0 := \boldsymbol{A}^{-1}$ and $\boldsymbol{S}_m$ being the conjugate gradients search directions. This is now a consequence of Corollary 9. Novel is the interpretation $\boldsymbol{\Sigma}_0 := \boldsymbol{Q}_m \boldsymbol{Q}_m^\intercal$ following from Proposition 7.

Corollary 10. *Under the prior*

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0) \qquad \textit{where} \quad \boldsymbol{\Sigma}_0 = \boldsymbol{Q}_m \boldsymbol{Q}_m^\intercal,$$

and with observations $\boldsymbol{y}_m = \boldsymbol{Q}_m^\intercal \boldsymbol{b}$, the SBI posterior mean Eq. (SBI) is identical to the CG iterate $\boldsymbol{x}_m$ in Eq. (FOM).

Above prior also gives a probabilistic interpretation of FOM, the generalization of CG, when $\boldsymbol{A}$ is not a valid covariance matrix.

8.2.2 Generalized Minimal Residual

Now follow probabilistic linear solvers with posterior means that coincide with the solution estimate from GMRES.

Corollary 11. *Under the SBI prior*

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0) \qquad \textit{where} \quad \boldsymbol{\Sigma}_0 = (\boldsymbol{A}^\intercal \boldsymbol{A})^{-1}$$

and the search directions $\boldsymbol{U}_m = \boldsymbol{A}\boldsymbol{Q}_m$, the posterior mean is identical to the GMRES iterate $\boldsymbol{x}_m$ in Eq. (GMR).

Proof. Substitute $\boldsymbol{R} = \boldsymbol{A}$ and $\boldsymbol{U}_m = \boldsymbol{A}\boldsymbol{Q}_m$ into Proposition 8 and compare to Eq. (GMR). □

This interpretation exhibits an interesting duality with CG for which $\boldsymbol{\Sigma}_0 = \boldsymbol{A}^{-1}$ and $\boldsymbol{U}_m = \boldsymbol{Q}_m$. Another probabilistic interpretation follows from Proposition 7.

Corollary 12. *Under the prior*

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0) \qquad \textit{where} \quad \boldsymbol{\Sigma}_0 = \boldsymbol{Q}_m \boldsymbol{Q}_m^\intercal, \tag{14}$$

and with observations $\boldsymbol{y}_m = \boldsymbol{Q}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{b}$, the SBI posterior mean Eq. (SBI) is identical the GMRES iterate $\boldsymbol{x}_m$ in Eq. (GMR).

Note that Proposition 11 has a posterior covariance which is not practical, as it involves $\boldsymbol{A}^{-1}$. Cockayne, Oates, Sullivan, and Girolami (2017) proposed replacing $\boldsymbol{A}^{-1}$ in the prior covariance with a preconditioner, which does yield a practically computable posterior, but this extension was not explored here. Furthermore, that approach yields unsatisfactorily calibrated posterior uncertainty, as described in that work. Corollary 12 does not have this drawback, but as mentioned there, the posterior covariance is a matrix of zeroes.

8.2.3 Simulation Study

In this section the simulation study of Cockayne, Oates, Ipsen, and Girolami (2018) will be replicated to demonstrate that the uncertainty produced from GMRES in Proposition 11 is similarly poorly calibrated to CG, owing to the dependence of the Arnoldi directions $\boldsymbol{Q}_m$ on $\boldsymbol{x}^*$ by way of its dependence on $\boldsymbol{b}$. Throughout the size of the test problems is set to $d = 100$. The eigenvalues of $\boldsymbol{A}$ were drawn from an exponential distribution with parameter $\gamma = 10$, and eigenvectors were drawn uniformly from the Haar-measure over rotation-matrices (see Diaconis and Shahshahani (1987)). In contrast to Cockayne, Oates, Ipsen, and Girolami (2018) the entries of $\boldsymbol{b}$ are drawn from a standard Gaussian distribution, rather than $\boldsymbol{x}^*$. In that case, $\boldsymbol{x}^*$ is a draw from the GMRES prior $\mathcal{N}(\boldsymbol{0}, (\boldsymbol{A}^\intercal \boldsymbol{A})^{-1})$ by Lemma 34 in Appendix B. Hence, the prior is perfectly calibrated for this scenario, and one would expect that the posterior should be equally well-calibrated for $m \geq 1$.

Cockayne, Oates, Ipsen, and Girolami (2018) argue that if the uncertainty is well-calibrated, then $\boldsymbol{x}^*$ can be considered as a draw from the posterior. Under this assumption, *i.e.* $\boldsymbol{\Sigma}_m^{-1/2}(\boldsymbol{x}^* - \boldsymbol{x}_m) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ they derive the test statistic:

$$Z(\boldsymbol{x}^*) := \|\boldsymbol{\Sigma}_m^{-1/2}(\boldsymbol{x}^* - \boldsymbol{x}_m)\| \sim \chi^2_{d-m}.$$

Figure 1 shows the convergence of GMRES and below, the convergence rate of the trace of the posterior covariance matrix Σ_m. Figure 2 displays the test statistic. It can be seen that the same poor uncertainty quantification occurs; even after just 10 iterations, the empirical distribution of the test statistic exhibits a profound left-shift, indicating an overly conservative posterior distribution. Producing well-calibrated posteriors remains an open issue in the field of probabilistic linear solvers.

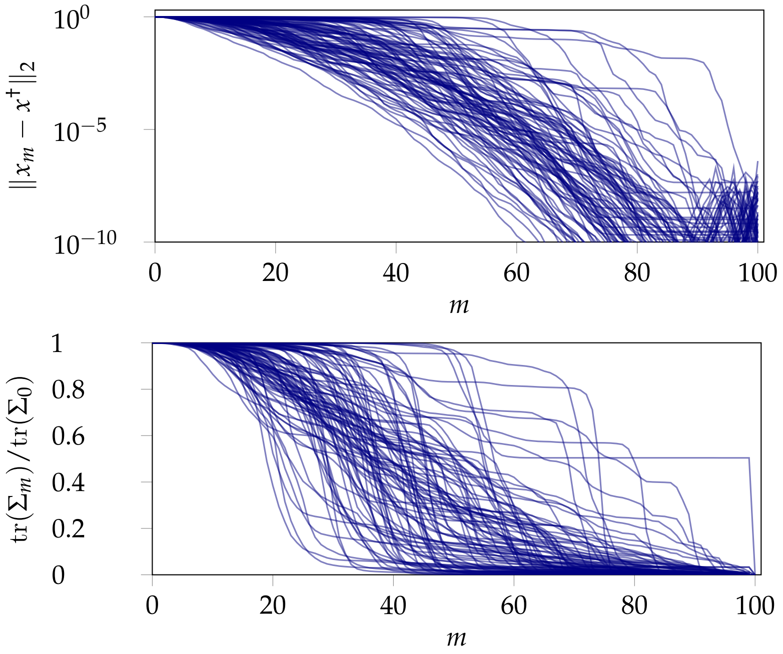


Figure 1: convergence of posterior mean (top plot) and trace of the posterior covariance matrix (bottom plot) of the probabilistic interpretation of GMRES from Proposition 11.

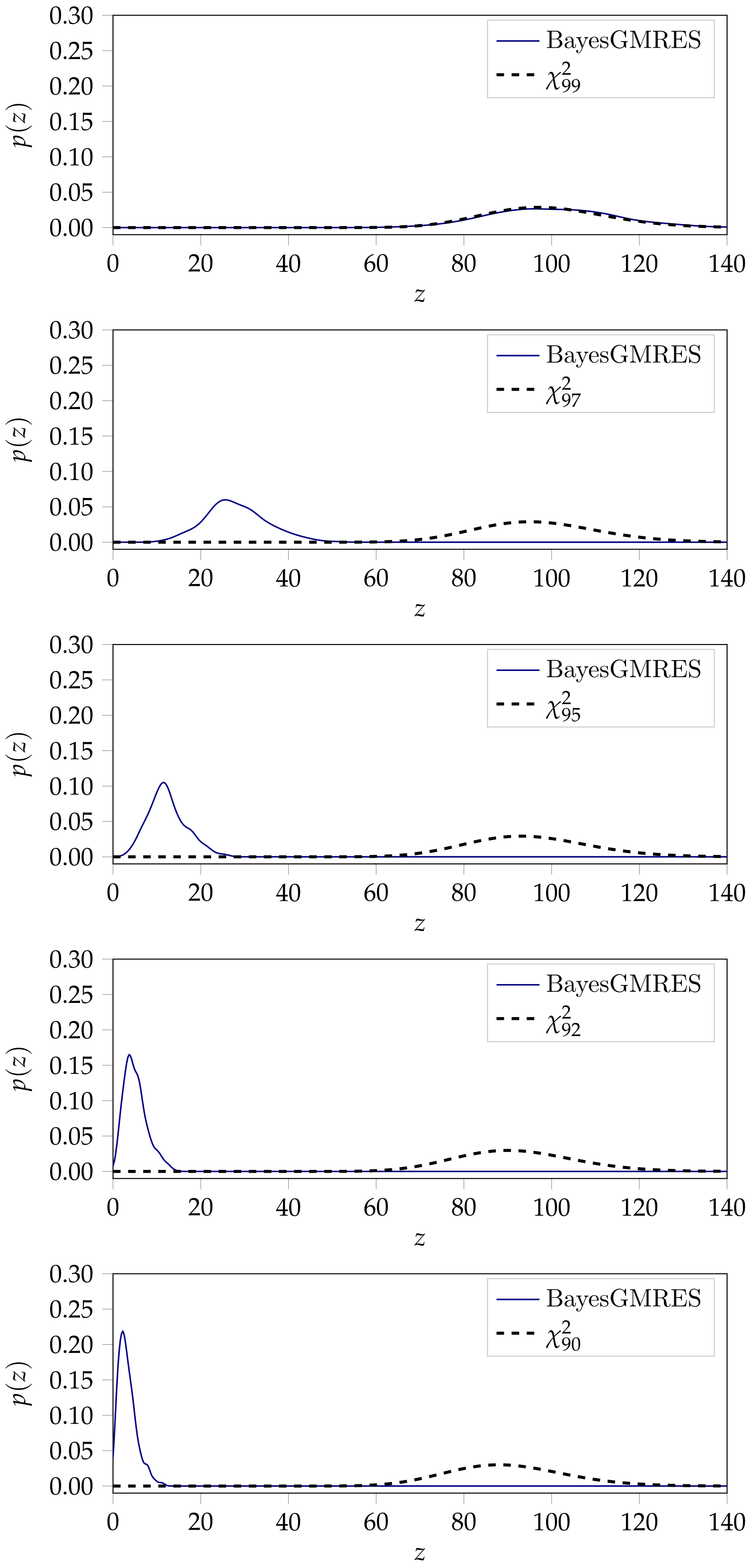


Figure 2: assessment of the uncertainty quantification. Plotted are kernel density estimates for the statistic Z based on 500 randomly sampled test problems for steps $m = \{1, 3, 5, 8, 10\}$. These are compared with the theoretical distribution of Z when the posterior distribution is well-calibrated.

8.3 Right-information Views

Surprisingly, it is harder to find probabilistic interpretations that use right-multiplied observations. A general result comparable to Proposition 7 remains yet to be found. The following proposition has not been published in Bartels, Cockayne, Ipsen, and Hennig (2019) but is a result of considerations during the writing process of this dissertation.

Proposition 13. *Consider a projection method defined by the matrices* $\boldsymbol{X}_m, \boldsymbol{U}_m \in \mathbb{R}^{d\times m}$*, each with linearly independent columns, and the starting vector* $\boldsymbol{x}_0 \in \mathbb{R}^d$ *and assume that* $\boldsymbol{X}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{U}_m$ *is invertible. Assume further that*

$$\boldsymbol{U}_m^\intercal \boldsymbol{A}\boldsymbol{x}_0 = \boldsymbol{0} \tag{15}$$

and if $\boldsymbol{x}_0 \neq \boldsymbol{0}$*, that there exists a* $\boldsymbol{v} \neq \boldsymbol{0}$*, s.t.*

$$\boldsymbol{X}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{v} = \boldsymbol{0}. \tag{16}$$

Then the iterate $\boldsymbol{x}_m$ *in Eq.* (**P**) *is identical to the projected MBI posterior mean in Eq.* (MBI) *under the prior*

$$p(\boldsymbol{A}^{-1}) = \mathcal{N}\left(\boldsymbol{A}^{-1}; \frac{1}{\boldsymbol{v}^\intercal \boldsymbol{b}}\boldsymbol{x}_0\boldsymbol{v}^\intercal, \boldsymbol{V}\otimes \boldsymbol{U}_m\boldsymbol{U}_m^\intercal\right) \tag{17}$$

when search directions $\boldsymbol{S}_m = \boldsymbol{X}_m$ *are used and where* $\boldsymbol{V}$ *is an arbitrary s.p.d. matrix.*

Proof.

$$\begin{aligned}
&\boldsymbol{A}_m^{-1}\boldsymbol{b}\\
&= \boldsymbol{A}_0^{-1}\boldsymbol{b} + (\boldsymbol{S}_m - \boldsymbol{A}_0^{-1}\boldsymbol{A}\boldsymbol{S}_m)(\boldsymbol{S}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{A}\boldsymbol{S}_m)^{-1}\boldsymbol{S}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{b}\\
&\textit{// using Eq. (MBI)}\\
&= \boldsymbol{x}_0 + (\boldsymbol{S}_m - \frac{1}{\boldsymbol{v}^\intercal\boldsymbol{b}}\boldsymbol{x}_0\boldsymbol{v}^\intercal\boldsymbol{A}\boldsymbol{S}_m)(\boldsymbol{S}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{A}\boldsymbol{S}_m)^{-1}\boldsymbol{S}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{b}\\
&\textit{// using } \boldsymbol{A}_0^{-1} = \frac{1}{\boldsymbol{v}^\intercal\boldsymbol{b}}\boldsymbol{x}_0\boldsymbol{v}^\intercal\\
&= \boldsymbol{x}_0 + (\boldsymbol{X}_m - \frac{1}{\boldsymbol{v}^\intercal\boldsymbol{b}}\boldsymbol{x}_0\boldsymbol{v}^\intercal\boldsymbol{A}\boldsymbol{X}_m)(\boldsymbol{X}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{A}\boldsymbol{X}_m)^{-1}\boldsymbol{X}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{b}\\
&\textit{// using definition of } \boldsymbol{S}_m\\
&= \boldsymbol{x}_0 + \boldsymbol{X}_m(\boldsymbol{X}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{A}\boldsymbol{X}_m)^{-1}\boldsymbol{X}_m^\intercal\boldsymbol{A}^\intercal\boldsymbol{U}_m\boldsymbol{U}_m^\intercal\boldsymbol{b}\\
&\textit{// using Eq. (16)}\\
&= \boldsymbol{x}_0 + \boldsymbol{X}_m(\boldsymbol{U}_m^\intercal\boldsymbol{A}\boldsymbol{S}_m)^{-1}\boldsymbol{U}_m^\intercal\boldsymbol{b}\\
&\textit{// simplifying}\\
&= \boldsymbol{x}_0 + \boldsymbol{X}_m(\boldsymbol{U}_m^\intercal\boldsymbol{A}\boldsymbol{S}_m)^{-1}\boldsymbol{U}_m^\intercal(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)\\
&\textit{// using Eq. (15)}
\end{aligned}$$

□

Observe the symmetry with respect to Proposition 7 where the roles of $\boldsymbol{X}_m$ and $\boldsymbol{U}_m$ in prior and search directions are exchanged. Equations (15) and (16) are trivially fulfilled when $\boldsymbol{x}_0 = \boldsymbol{0}$. If $\boldsymbol{X}_m = \boldsymbol{U}_m$ and $\boldsymbol{A}$ is s.p.d. one can choose $\boldsymbol{v} := \boldsymbol{x}_0$. This choice imposes the search directions to be $\boldsymbol{A}$-conjugate to the initial solution. For conjugate gradients this is generally *not* the case.

8.3.1 Generalized Minimal Residual

Now follows a probabilistic interpretation of GMRES. Again, a general result comparable to those in Section 8.2.2 remains future work.

Proposition 14. *Assume that $\boldsymbol{x}_0 = \boldsymbol{0}$. Under the prior*

$$p(\boldsymbol{A}^{-1}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma} \otimes \boldsymbol{I})$$

and given $\boldsymbol{Y}_m = \boldsymbol{A}\boldsymbol{Q}_m$, where $\boldsymbol{Q}_m$ are the Arnoldi directions, the implied posterior mean over the solution given by $\boldsymbol{A}_m^{-1}\boldsymbol{b}$ is equivalent to the GMRES solution.

Proof. Under this prior, $\boldsymbol{b}$ applied to the posterior mean is

$$\begin{aligned}\boldsymbol{A}_m^{-1}\boldsymbol{b} =& \boldsymbol{A}_0^{-1}\boldsymbol{b} + (\boldsymbol{Q}_m - \boldsymbol{A}_0^{-1}\boldsymbol{Y}_m)(\boldsymbol{Y}_m^\intercal \boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\intercal \boldsymbol{b}\\ =& \boldsymbol{Q}_m(\boldsymbol{Y}_m^\intercal \boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\intercal \boldsymbol{b}\\ =& \boldsymbol{Q}_m(\boldsymbol{Q}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{A}\boldsymbol{Q}_m)^{-1}\boldsymbol{Q}_m^\intercal \boldsymbol{A}^\intercal \boldsymbol{b}\end{aligned}$$

which is the GMRES projection (Eq. (GMR)) if $\boldsymbol{x}_0 = \boldsymbol{0}$. □

If above proposition is true for $\boldsymbol{x}_0 \neq \boldsymbol{0}$, it provides structural insights into GMRES. For the choice $\boldsymbol{\Sigma} := \boldsymbol{I}$, one could say that running GMRES is equivalent to inference over $\boldsymbol{A}^{-1}$ or $\boldsymbol{x}^*$ under the assumption that all entries are independent standard normal.

8.3.2 Conjugate Gradients

The following interpretation of conjugate gradients provides an explanation of how the search directions can be motivated probabilistically. The following proposition is a contribution by Philipp Hennig.

Proposition 15. *Consider the prior*

$$p(\boldsymbol{A}^{-1}) = \mathcal{N}(\operatorname{vec}\left(\boldsymbol{A}^{-1}\right); \operatorname{vec}(\alpha \boldsymbol{I}), (\beta \boldsymbol{I} + \gamma \boldsymbol{W}) \otimes (\beta \boldsymbol{I} + \gamma \boldsymbol{W}))$$

where $\boldsymbol{W} := \boldsymbol{A}^{-1}$. For all choices $\alpha \in \mathbb{R} \setminus \{0\}$ and $\beta, \gamma \in \mathbb{R}_{+,0}$ with $\beta + \gamma > 0$, Algorithm 3 is equivalent to CG, in the sense that it produces the same search directions $\boldsymbol{s}_i$ (scaled).

Proof. The proof is extensive and can be found in Appendix D. □

In general, the solution estimate $\boldsymbol{x}_m$ produced by Algorithm 3 differs from the posterior over $\boldsymbol{x}$ in Eq. (MBI), since the CG estimate is corrected by the step size computed in line 6. Fixing this rank-1 discrepancy would complicate the exposition of Algorithm 3 and yield a more cumbersome algorithm. Proposition 15 is remarkable since, for the case when $\gamma = 0$, it is the only result with a rank d prior matrix, that does *not* involve $\boldsymbol{A}^{-1}$.

```
1  $x_0 \leftarrow A_0^{-1} b$                                  // initial guess
2  $r_0 \leftarrow A x_0 - b$
3  for $i = 1, \dots, m$ do
4        $d_i \leftarrow -A_{i-1}^{-1} r_{i-1}$                // compute optimization direction
5        $z_i \leftarrow A d_i$                                 // observe
6        $\alpha_i \leftarrow -\frac{d_i^\intercal r_{i-1}}{d_i^\intercal z_i}$   // optimal step-size
7        $s_i \leftarrow \alpha_i d_i$                          // re-scale step
8        $y_i \leftarrow \alpha_i z_i$                          // re-scale observation
9        $x_i \leftarrow x_{i-1} + s_i$                         // update estimate for $x$
10       $r_i \leftarrow r_{i-1} + y_i$                         // new gradient at $x_i$
11     $A_i^{-1} \leftarrow \mathbb{E}_{p(A^{-1}|S,Y)} A^{-1}$  // estimate $A^{-1}$
12 end for
13 return $x_m$
```

Algorithm 3: The algorithm referred to by Proposition 15, which reproduces the search directions from CG.

8.4 Preconditioning

This section discusses probabilistic views on preconditioning. Preconditioning is a technique used to accelerate the convergence of iterative methods (Saad 2003, Sections 9 and 10). A preconditioner $\boldsymbol{P}$ is a nonsingular matrix satisfying two requirements:

1. linear systems $\boldsymbol{P}\boldsymbol{z} = \boldsymbol{c}$ can be solved at low computational cost

2. and $\boldsymbol{P}$ is "close" to $\boldsymbol{A}$ in some sense.

Hence, solving systems based upon a preconditioner can be viewed as approximately inverting $\boldsymbol{A}$, and indeed many preconditioners are constructed based upon this intuition. One distinguishes between *right preconditioners* $\boldsymbol{P}_r$ and *left preconditioners* $\boldsymbol{P}_l$, depending on whether they act on $\boldsymbol{A}$ from the left or the right. Two-sided preconditioning with nonsingular matrices $\boldsymbol{P}_l$ and $\boldsymbol{P}_r$ transforms implicitly Eq. (LES) into a new linear problem

$$\boldsymbol{P}_l \boldsymbol{A} \boldsymbol{P}_r\, \boldsymbol{z}^* = \boldsymbol{P}_l \boldsymbol{b}, \qquad \text{with} \quad \boldsymbol{x}^* = \boldsymbol{P}_r \boldsymbol{z}^*. \tag{18}$$

The preconditioned system can then be solved using arbitrary projection methods as described in Section 7.2, from the starting point $\boldsymbol{z}_0$ defined by $\boldsymbol{x}_0 = \boldsymbol{P}_r \boldsymbol{z}_0$. The probabilistic view can be used to create a nuanced description of preconditioning as a form of prior information. In the SBI framework, Proposition 16 below shows that solving a right-preconditioned system is equivalent to modifying the prior, while in Proposition 17 shows that left-preconditioning is equivalent to making a different choice of observations. The following proposition is a contribution by Philipp Hennig.

Proposition 16 (Right preconditioning)**.** *Consider the right-preconditioned system*

$$\boldsymbol{A}\boldsymbol{P}_r\boldsymbol{z}^* = \boldsymbol{b} \qquad \textit{where} \quad \boldsymbol{x}^* = \boldsymbol{P}_r\boldsymbol{z}^*. \tag{19}$$

SBI on Eq. (19) *under the prior*

$$\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{z}; \boldsymbol{z}_0, \boldsymbol{\Sigma}_0) \tag{20}$$

is equivalent to solving Eq. (LES) *under the prior*

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{P}_r\boldsymbol{z}_0, \boldsymbol{P}_r\boldsymbol{\Sigma}_0\boldsymbol{P}_r^\intercal). \tag{21}$$

Proof. Define $\boldsymbol{B} := \boldsymbol{A}\boldsymbol{P}_r$ and $\hat{\boldsymbol{r}}_0 = \boldsymbol{b} - \boldsymbol{B}\boldsymbol{z}_0$. Consider the prior defined in Eq. (21). Lemma 3 implies that after observing information from search directions $\boldsymbol{S}_m$, the posterior mean equals

$$\boldsymbol{x}_m = \boldsymbol{P}_r\boldsymbol{z}_0 + \boldsymbol{P}_r\boldsymbol{\Sigma}_0\boldsymbol{B}^\intercal\boldsymbol{S}_m(\boldsymbol{S}_m^\intercal\boldsymbol{B}\boldsymbol{\Sigma}_0\boldsymbol{B}^\intercal\boldsymbol{S}_m)^{-1}\boldsymbol{S}_m^\intercal\hat{\boldsymbol{r}}_0.$$

Left multiplying by $\boldsymbol{P}_r^{-1}$ shows that this is equivalent to

$$\begin{aligned}\boldsymbol{z}_m &:= \boldsymbol{P}_r^{-1}\boldsymbol{x}_m \\ &= \boldsymbol{z}_0 + \boldsymbol{\Sigma}_0\boldsymbol{B}^\intercal\boldsymbol{S}_m(\boldsymbol{S}_m^\intercal\boldsymbol{B}\boldsymbol{\Sigma}_0\boldsymbol{B}^\intercal\boldsymbol{S}_m)^{-1}\boldsymbol{S}_m^\intercal\hat{\boldsymbol{r}}_0.\end{aligned}$$

Now note that $\boldsymbol{z}_m$ is the posterior mean from the prior Eq. (20) after observing search directions $\boldsymbol{S}_m$, when inferring the solution over of the system $\boldsymbol{B}\boldsymbol{z}^* = \boldsymbol{b}$. □

Proposition 17 (Left preconditioning)**.** *Consider the left-preconditioned system*

$$\boldsymbol{P}_l\boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{P}_l\boldsymbol{b} \tag{22}$$

And the SBI prior

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0).$$

Then the posterior from SBI on Eq. (22) *under search directions* $\boldsymbol{S}_m$ *is equivalent to the posterior from SBI applied to the system Eq.* (LES) *under search directions* $\boldsymbol{P}_l^\intercal\boldsymbol{S}_m$*.*

Proof. Lemma 3 implies that after observing search directions $\boldsymbol{T}_m$, the posterior mean over the solution of Eq. (LES) equals

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{T}_m (\boldsymbol{T}_m^\intercal \boldsymbol{A} \boldsymbol{\Sigma}_0 \boldsymbol{A}^\intercal \boldsymbol{T}_m)^{-1} \boldsymbol{T}_m^\intercal \boldsymbol{r}_0$$

where $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$. Setting $\boldsymbol{T}_m = \boldsymbol{P}_l^\intercal \boldsymbol{S}_m$ gives

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{\Sigma}_0 \boldsymbol{B}^\intercal \boldsymbol{S}_m (\boldsymbol{S}_m^\intercal \boldsymbol{B} \boldsymbol{\Sigma}_0 \boldsymbol{B}^\intercal \boldsymbol{S}_m)^{-1} \boldsymbol{S}_m^\intercal \boldsymbol{P}_l \hat{\boldsymbol{r}}_0$$

where $\boldsymbol{B} := \boldsymbol{P}_l \boldsymbol{A}$ and $\hat{\boldsymbol{r}}_0 = \boldsymbol{P}_l \boldsymbol{b} - \boldsymbol{P}_l \boldsymbol{A} \boldsymbol{x}_0$. Thus, $\boldsymbol{x}_m$ is the posterior mean of the system $\boldsymbol{B}\boldsymbol{x}^* = \boldsymbol{P}_l \boldsymbol{b}$ after observing search directions $\boldsymbol{S}_m$. □

If a probabilistic linear solver has a posterior mean which coincides with a projection method, the Propositions 16 and 17 show how to obtain a probabilistic interpretation of the *preconditioned* version of that algorithm. With Proposition 5 these results carry over to MBI.

This interpretation of preconditioning is *not* unique. When using left-multiplied observations, the same reasoning can be used to show that left-preconditioning corresponds to a change in the prior belief, while right-preconditioning corresponds to collecting different observations.

9 Discussion

9.1 Summary

This part showed that solution-based inference is contained within matrix-based inference, which allows to transfer results from SBI to MBI with left-multiplied information. This correspondence motivates the classification of probabilistic interpretations as left-multiplied and right-multiplied. Connections between probabilistic linear solvers and the class of projection methods have been presented, and a probabilistic interpretation of preconditioning.

9.2 Future Directions

Posterior uncertainty calibration remains a challenge for probabilistic linear solvers. Direct probabilistic interpretations of CG and GMRES yield posterior covariance matrices which are not always computable, and even when the posterior can be computed, the uncertainty remains poorly calibrated. Mitigating this issue without sacrificing the rate of convergence provided by Krylov methods remains an important goal.

Another natural question to ask is whether other classes of linear solvers such as stationary iterative methods (Saad 2003, Chapter 4) have probabilistic interpretations. As an outlook for a non-Bayesian probabilistic interpretation consider the following relation between Jacobi-iteration (Saad 2003, pp. 105) and Gibbs sampling (Geman and Geman 1984; Bishop 2006, pp. 542).

If all components of $\boldsymbol{x}^*$ are identified, except for one, say the i-th component, then one can identify this last component, by rearranging the i-th equation for $\boldsymbol{x}_i$

$$\boldsymbol{x}_i = \frac{1}{\boldsymbol{A}_{ii}} \left(\boldsymbol{b}_i - \sum_{\substack{j=1 \\ j \neq i}}^{d} \boldsymbol{A}_{ij} \boldsymbol{x}_j \right), \tag{23}$$

and has found the solution $\boldsymbol{x}^*$ (assuming $\boldsymbol{A}_{ii} \neq 0$). The last sentence can be expressed probabilistically as

$$p(\boldsymbol{x}_i \mid \boldsymbol{x}_1, \dots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \dots, \boldsymbol{x}_d) = \delta \left(\boldsymbol{x}_i - \frac{1}{\boldsymbol{A}_{ii}} \left(\boldsymbol{b}_i - \sum_{\substack{j=1 \\ j \neq i}}^{d} \boldsymbol{A}_{ij} \boldsymbol{x}_j \right) \right).$$

Given an initial guess $\boldsymbol{x}_0$, and a likelihood of the form above, Gibbs sampling cycles through all variables in order and updates each by a sample from the likelihood, where in this case the samples are deterministic and updated according to Eq. (23). At the same time, Eq. (23) is the update rule for Jacobi-iteration. Hence, when using the last sample as estimate, Gibbs sampling and Jacobi-iteration produce the same output in each step.

Part IV

Probabilistic Solvers for Kernel Least-Squares Problems

> Hier soll der geduldige Leser verschnaufen. Ich schweife nicht ab, sondern zurück, verspreche feierlich: das Fäkalienthema ist noch nicht ganz erschöpft, Chopin aber endgültig erledigt, jedenfalls in seiner Qualität, als Quantität werde ich, schon aus kompositorischen Gründen, mich seiner noch einige Male bedienen müssen. Es wird nicht mehr vorkommen. Reuevoll schlage ich mir an die Brust, jene, deren Quantität bei meinem Hemdenschneider zu erfahren wäre, deren Qualität aber so schwer zu definieren ist.

—Heinrich Böll, *Entfernung von der Truppe*

10

Preliminaries

The previous part compared and related probabilistic numerical methods and classic appproaches to approximation. The purpose of this part is to present a possibility, to use the probabilistic perspective to develop novel approximation algorithms. It is based on the publication S. Bartels and P. Hennig (2019). "Conjugate Gradients for Kernel Machines." In: *ArXiv e-prints* 1911.06048. arXiv:1911.06048, currently under revision. I was the primary author and performed the principal analysis and work, Philipp Hennig provided initial ideas and direction.

10.1 Introduction

Regularized least-squares (kernel-ridge / Gaussian process) regression is a fundamental algorithm of statistics and machine learning. Because generic algorithms for the exact solution have cubic complexity in the number of datapoints, large datasets require to resort to approximations. In the following, the computation of the least-squares prediction is itself treated as a probabilistic inference problem. The key is a structured Gaussian regression model on the kernel function that uses projections of the kernel matrix to obtain a low-rank approximation of the kernel and the matrix. This leads to an enhanced way to use the method of conjugate gradients for the specific setting of least-squares regression as encountered in machine learning dubbed *kernel machine conjugate gradients* (KMCG). KMCG improves the approximation of the kernel ridge regressor / Gaussian process posterior mean over vanilla conjugate gradients and, when used in Gaussian process models, allows computation of the posterior variance and the log marginal likelihood (evidence) without further overhead. Recall the regularized least-squares problem introduced in Chapter 4:

$$\bar{f}(\boldsymbol{x}_*) = \boldsymbol{k}_*^\intercal(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}. \tag{1}$$

Instead of providing an approximation solely to the vector $(\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}$, the approach uses the MVMs performed by CG to learn an approximation directly to the function k. This will allow to approximate Eqs. (2) and (3) (restated below) as well, without rerunning CG.

$$\bar{c}(\boldsymbol{x}_*, \boldsymbol{x}_{**}) = k(\boldsymbol{x}_*, \boldsymbol{x}_{**}) - \boldsymbol{k}_*^\intercal(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{k}_{**} \tag{2}$$

$$\ln p(\boldsymbol{y}) = -\frac{1}{2}\boldsymbol{y}^\intercal(\boldsymbol{K} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y} + \frac{1}{2}\ln|2\pi(\boldsymbol{K} + \sigma^2\boldsymbol{I})|^{-1} \tag{3}$$

Chapter 11 proposes a model template that can be used to learn finite-rank approximations to kernel functions. Then, Chapter 12 shows how conjugate gradients can be used in combination.

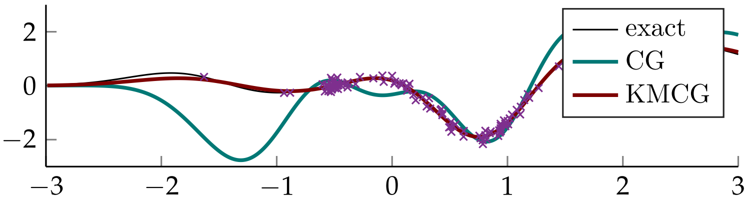


Figure 3: the algorithm KMCG in comparison to CG on a toy setup. The dataset consists of one hundred data-points where the targets are a draw from a zero-mean Gaussian process with squared exponential kernel (Eq. (43) with $\mathbf{\Lambda} = 0.25$ and $\theta_f = 2$). The thin, black line is the posterior mean of that Gaussian process (Eq. (1)). The light-green line is the mean prediction produced by conjugate gradients after $P = 7$ steps and the dark-red line is the mean prediction of KMCG (where the number of inducing inputs $M = N$).

10.2 Finite-rank Kernel

An M-rank approximation to a kernel is a factorization of the form

$$k(\boldsymbol{x}, \boldsymbol{z}) \approx \boldsymbol{\phi}(\boldsymbol{x})^* \mathbf{\Sigma}^{-1} \boldsymbol{\phi}(\boldsymbol{z}) \tag{24}$$

where $\boldsymbol{\phi}(\boldsymbol{x}) : \mathbb{X} \to \mathbb{C}^M$, $\boldsymbol{\phi}^*$ denotes the conjugate transpose, and $\mathbf{\Sigma}$ is an $M \times M$ Hermitian and positive definite matrix. Given such an expansion one can use the matrix-inversion, and matrix-determinant lemmata to approximate Equations (1) to (3) with the expressions below

$$\bar{f}(\boldsymbol{x}_*) \approx \boldsymbol{\phi}(\boldsymbol{x}_*)^* \boldsymbol{A}^{-1} \mathbf{\Phi} \boldsymbol{y} \tag{25}$$

$$\bar{c}(\boldsymbol{x}_*, \boldsymbol{z}_*) \approx \sigma^2 \boldsymbol{\phi}(\boldsymbol{x}_*)^* \boldsymbol{A}^{-1} \boldsymbol{\phi}(\boldsymbol{z}_*) \tag{26}$$

$$\ln p(\boldsymbol{y}) \approx -\frac{1}{2} \boldsymbol{y}^{\intercal} \mathbf{\Phi}^* \boldsymbol{A}^{-1} \mathbf{\Phi} \boldsymbol{y} - \frac{1}{2} \ln |\boldsymbol{A}| - \frac{N}{2} \ln(2\pi\sigma^2) \tag{27}$$

where $\boldsymbol{\phi}(\boldsymbol{x}_*)_j = \phi_j(\boldsymbol{x}_*)$, $\mathbf{\Phi}_{ij} = \phi_i(\boldsymbol{X}_j)$ and $\boldsymbol{A} := \mathbf{\Phi}\mathbf{\Phi}^* + \sigma^2 \mathbf{\Sigma}$. Typically $M \ll N$ and therefore the computational costs to evaluate Equations (25) to (27) reduce from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$, *i.e.* linear in N. The dominant factor is the matrix-matrix product $\mathbf{\Phi}\mathbf{\Phi}^*$.

An example for a finite-rank kernel that will become important later is the *Subset of Regressors* (SoR) approximation (Quiñonero-Candela and Rasmussen 2005)

$$k_{SoR}(\boldsymbol{x}, \boldsymbol{z}) = k(\boldsymbol{x}, \boldsymbol{X}_U) k(\boldsymbol{X}_U, \boldsymbol{X}_U)^{-1} k(\boldsymbol{X}_U, \boldsymbol{z}) \tag{28}$$

where $\boldsymbol{X}_U$ is a set of M so called inducing inputs. The method proposed in this work (KMCG) is related to SoR. Readers familiar with SoR will be aware of the associated flaws, and methods to remedy them (Quiñonero-Candela and Rasmussen 2005; Titsias 2009). For stationary kernels and tests points far away from the data, the predictive uncertainty (Eq. (26)) goes to zero. The *Deterministic Training Conditional (DTC)* approximation alleviates this issue by using the exact kernel for the prior uncertainty over the test inputs (Quiñonero-Candela and Rasmussen 2005). In effect this is a substitution of Eq. (26) for Eq. (29) below.

$$\bar{c}(\boldsymbol{x}_*, \boldsymbol{z}_*) \approx k(\boldsymbol{x}_*, \boldsymbol{x}_{**}) - \boldsymbol{\phi}(\boldsymbol{x}_*)^* \left(\mathbf{\Phi}\mathbf{\Phi}^* + \sigma^2 \boldsymbol{I} \right)^{-1} \boldsymbol{\phi}(\boldsymbol{z}_*) \tag{29}$$

We will apply the same substitution for our method KMCG.

11

Model

To approximate Equations (1) to (3), we will approximate the kernel and, to this end, present a probabilistic estimation rule for k. The idea is to treat the kernel as unknown and to choose prior and likelihood such that the posterior mean k_M is efficient to evaluate and yields a kernel of finite rank. Substituting for this finite-rank kernel in Equations (1) to (3) then allows to compute these expressions faster. The following sections describe a prior over k, possible likelihoods and resulting posteriors. Fig. 4 on page 48 shows a schematic summary of this chapter.

11.1 Prior

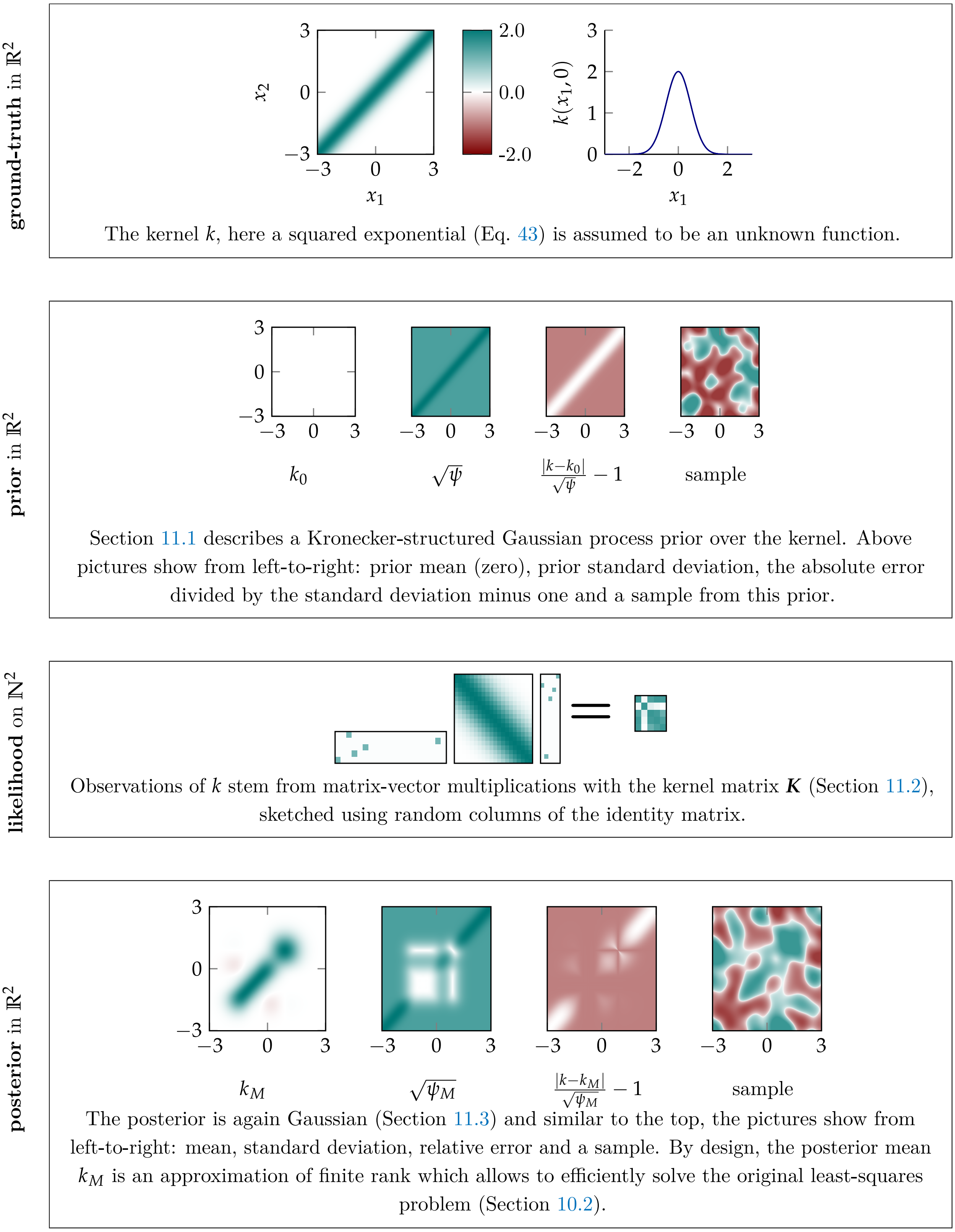
Consider a Gaussian process prior over bivariate functions

$$k \sim \mathcal{GP}(k_0, \gamma\psi) \tag{30}$$

where $\psi : \mathbb{X}^2 \times \mathbb{X}^2 \to \mathbb{R}$ is a covariance function over kernel and $\gamma \in \mathbb{R}^+$ is a scaling parameter. Since the posterior mean is meant to be a substitution for the exact kernel, this is an exchange of one least-squares problem for another. Without further assumptions, calculating the posterior over k is more expensive than computing the equations of interest (Eqs. (1) to (3)). Efficient inference is rendered possible by imposing the following structure on ψ

$$\psi(k(\boldsymbol{a},\boldsymbol{b}), k(\boldsymbol{c},\boldsymbol{d})) := \frac{1}{2}w(\boldsymbol{a},\boldsymbol{c})w(\boldsymbol{b},\boldsymbol{d}) + \frac{1}{2}w(\boldsymbol{a},\boldsymbol{d})w(\boldsymbol{b},\boldsymbol{c}) \tag{31}$$

for $\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},\boldsymbol{d} \in \mathbb{X}$ and where w is a covariance function on the domain $\mathbb{X}$. Consider the first addend. It states that the similarity between $k(\boldsymbol{a},\boldsymbol{b})$ and $k(\boldsymbol{c},\boldsymbol{d})$ depends on the similarity of $\boldsymbol{a}$ and $\boldsymbol{c}$, and $\boldsymbol{b}$ and $\boldsymbol{d}$–a natural assumption for kernel matrices. The second addend is a symmetrization of the first. Observe that each addend is a product kernel of two pairs of inputs and recall that a product kernel produces Kronecker product matrices. The sum of the two products leads to covariance matrices that have a *symmetric* Kronecker product form, *i.e.* $\forall \boldsymbol{A},\boldsymbol{B} \in \mathbb{R}^{N\times N} : \psi(\boldsymbol{A},\boldsymbol{B}) = \boldsymbol{A} \circledS \boldsymbol{B} \in \mathbb{R}^{N^2\times N^2}$ (recall Chapter 6). As in the previous part, this will allow a sufficiently efficient evaluation of the posterior. Fig. 4 visualizes the variance and shows samples from this prior for the toy setup from Fig. 3.



The kernel k, here a squared exponential (Eq. 43) is assumed to be an unknown function.

Section 11.1 describes a Kronecker-structured Gaussian process prior over the kernel. Above pictures show from left-to-right: prior mean (zero), prior standard deviation, the absolute error divided by the standard deviation minus one and a sample from this prior.

Observations of k stem from matrix-vector multiplications with the kernel matrix $\boldsymbol{K}$ (Section 11.2), sketched using random columns of the identity matrix.

The posterior is again Gaussian (Section 11.3) and similar to the top, the pictures show from left-to-right: mean, standard deviation, relative error and a sample. By design, the posterior mean k_M is an approximation of finite rank which allows to efficiently solve the original least-squares problem (Section 10.2).

Figure 4: schematic summary of the proposed kernel approximation method.

This choice of prior offers a trade-off between efficient tractable inference and the desire to encode as much prior structural information about the kernel as possible. One desirable property to encode is symmetry, and indeed, matrix-valued functions sampled from this prior distribution are symmetric (*c.f.* Fig. 4 for examples, Appendix E.1 for formal proof). Kernel functions are also positive definite. Unfortunately, since the positive definite cone is not a linear sub-space of the vector-space of real matrices, this property can not be encoded in a Gaussian prior in closed form.[1] However, it *is* possible to guarantee positive-definiteness of the posterior mean point estimate through the specific choice of prior parameters $k_0 = 0$ (proof in Appendix E.2.1). For this reason, $k_0 := 0$ for the remainder. There are other properties of certain kernels that would be desirable to encode, but which are not feasible within the chosen framework without also sacrificing fast computability at the same time. For example, stationarity of the kernel can not be represented by a prior with Kronecker structure in the covariance since $\boldsymbol{a}$ and $\boldsymbol{b}$ (and symmetrically $\boldsymbol{c}$ and $\boldsymbol{d}$) do not appear together as arguments to w.

The question remains how to choose w. Recall that w should reflect the similarity between $k(\boldsymbol{a}, \boldsymbol{b})$ and $k(\boldsymbol{c}, \boldsymbol{d})$ which depends on the similarity of $\boldsymbol{a}$ and $\boldsymbol{c}$, and $\boldsymbol{b}$ and $\boldsymbol{d}$. To measure the relationship between inputs is exactly the purpose of the kernel k and we therefore set

$$w := k$$

for the remainder. Even if k fails to capture similarity between inputs, as choice for w it still captures the similarity between the kernel values. Second, samples from the approximate kernel will be a function of w and lastly, this choice is convenient computationally as expressions simplify.

11.2 Likelihood

Having specified a prior over k, we will now be concerned with how to obtain observations. We can use matrix-vector products with the kernel matrix for learning a low-rank version of the kernel by introducing the linear operator

$$\boldsymbol{T_p} : k \mapsto \operatorname{vec}\left(\left[\iint k(\boldsymbol{x}, \boldsymbol{z}) p_i(\boldsymbol{x}) p_j(\boldsymbol{z}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{z}\right]_{ij}\right) \tag{32}$$

where $i, j = 1...P$, $\boldsymbol{p} = [p_1, ..., p_P]$ are densities or distributions.

Example 18 (Matrix-vector multiplication)**.** *Define* $\boldsymbol{T_p}$ *with*

$$p_i(\boldsymbol{x}) = \sum_{j=1}^{M} s_{ij} \delta(\boldsymbol{x} - \boldsymbol{x}_{u_j}). \tag{33}$$

[1] For example Hennig (2015) discusses this problem and possible solutions.

Then the evaluation of $\boldsymbol{T_p}k$ reduces to a matrix vector product, that is $\operatorname{mat}\left(\boldsymbol{T_p}k\right) = \boldsymbol{S}^\intercal k(\boldsymbol{X}_U, \boldsymbol{X}_U)\boldsymbol{S}$ *where* $\boldsymbol{S}_{ij} = s_{ij}$*,* $\boldsymbol{X}_U = [\boldsymbol{x}_{u_1}, ..., \boldsymbol{x}_{u_M}]$*.*

The $\boldsymbol{x}_{u_j}$ can be datapoints or arbitrary elements of the domain $\mathbb{X}$. In Chapter 12 we will use the conjugate gradients search directions.

Example 19 (Integrals with Eigenfunctions)**.** *Let ϕ_i $i = 1, ..., P$ be orthogonal Eigenfunctions of k with respect to a density ν on $\mathbb{X}$,* i.e.

$$\int k(\boldsymbol{x}, \boldsymbol{z})\phi_i(\boldsymbol{z})\nu(\boldsymbol{z})\ \mathrm{d}\boldsymbol{z} = \lambda_i\phi_i(\boldsymbol{x})$$
$$\int \phi_i(\boldsymbol{z})\phi_j(\boldsymbol{z})\nu(\boldsymbol{z})\ \mathrm{d}\boldsymbol{z} = \delta_{ij}$$

where $\lambda_i \in \mathbb{R}_+$ and δ_{ij} is the Kronecker delta (compare Rasmussen and Williams (2006, p. 96)). Then for

$$p_i(\boldsymbol{x}) = \phi_i(\boldsymbol{x})\nu(\boldsymbol{x})$$

the observations $[\operatorname{mat}\left(\boldsymbol{T_p}k\right)]_{ij} = \delta_{ij}\lambda_i$ *are spectral values of the kernel.*

In essence, this example shows another possibility to express prior knowledge over the kernel. This likelihood leads to the *Projected Bayes Regressor* (Trecate, Williams, and Opper 1999), which is a historical, deterministic precursor to the more widely known random Fourier feature expansion of Rahimi and Recht 2008. For the purposes of this thesis, Example 19 is a motivation for the generality of the observation operator in Eq. (32). The example will not be considered further.

11.3 Posterior

The observation operator $\boldsymbol{T_p}$ is linear, and hence transforms the Gaussian prior into an also Gaussian posterior. Given the prior (Eq. (30)) and any likelihood of the previous section, the posterior is Gaussian with:

$$p(k \mid \boldsymbol{Y}, \boldsymbol{T_p}) = \mathcal{N}(k_M, w_M)$$
$$k_M = k_0 + (\boldsymbol{T_p}\psi)^\intercal(\boldsymbol{T_p}(\boldsymbol{T_p}\psi)^\intercal)^{-1}(\operatorname{vec}(\boldsymbol{Y}) - \boldsymbol{T_p}k_0) \tag{34}$$
$$\psi_M = \psi - (\boldsymbol{T_p}\psi)^\intercal(\boldsymbol{T_p}(\boldsymbol{T_p}\psi)^\intercal)^{-1}\boldsymbol{T_p}\psi \tag{35}$$

The concrete posterior depends on the choice of $\boldsymbol{T_p}$. The following propositions presents an approximation method that has a view as GP inference with low-rank kernel and how it arises in our framework.

Proposition 20 (Subset of Regressors)**.** *Consider the prior of Eq.* (30) *with $k_0 := 0$ and $w := k$ and the likelihood defined in Eq.* (33) *with $s_{ij} = \delta_{ij}$. Then the posterior mean k_M is equivalent to that of SoR:*

$$k_M(\boldsymbol{x}, \boldsymbol{z}) = k_{SoR} = k(\boldsymbol{x}, \boldsymbol{X}_U)k(\boldsymbol{X}_U, \boldsymbol{X}_U)^{-1}k(\boldsymbol{X}_U, \boldsymbol{z})$$

where $\boldsymbol{X}_U$ *are inducing inputs, not necessarily part of* $\boldsymbol{X}$*.*

The proof is part of Appendix E.2. An example of this posterior distribution is shown in Fig. 4. Fig. 5 visualizes the progression of the posterior for the KMCG algorithm, presented in the next chapter.

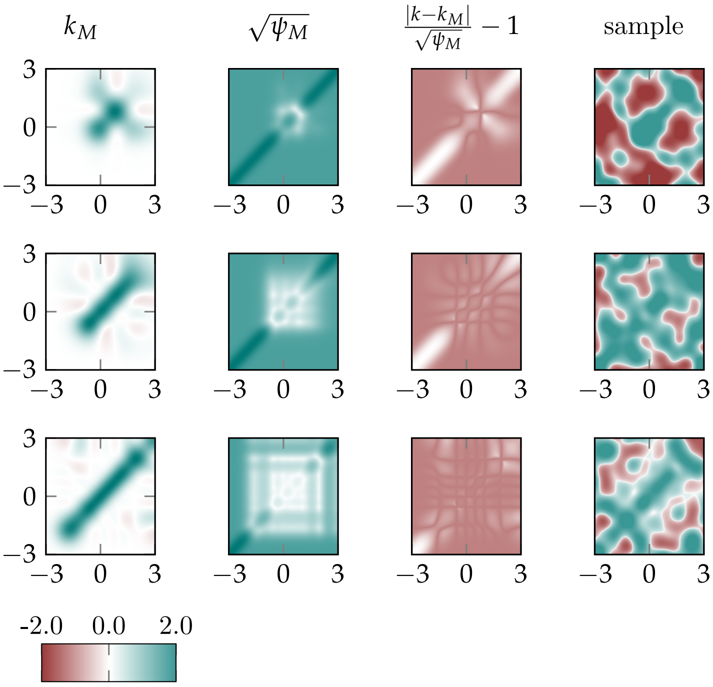


Figure 5: progression of the posterior (Eq. (37)) for KMCG on the toy example from Fig. 3 for $P = 2, 4$ and 8 conjugate gradients steps. The columns show from left to right: mean, standard deviation, standardized error (white refers to perfect calibration, green to overconfidence and red to underconfidence) and a sample.

12

Conjugate Gradients for Kernel Machines

The previous chapter introduced a probabilistic estimation rule for the kernel k. This section presents another data-collection approach using conjugate gradients that leads to a new approximation algorithm: *kernel machine conjugate gradients* (KMCG).

The interest to use conjugate gradients for kernel machines goes back to more than 25 years (Skilling 1993) and is still continuing (Davies 2015; Filippone and Engler 2015). Albeit quadratic costs per step, CG has advantages over many of the approximation methods referenced in the introduction. CG has only one parameter, the desired precision, which is more natural than *e.g.* the number of inducing inputs for inducing point methods (Quiñonero-Candela and Rasmussen 2005). Thus the computational budget of CG is not fixed in advance but varies as necessary for the problem at hand.

The approach is to run conjugate gradients for P steps on a kernel matrix of size M and to treat the matrix multiplications ($\boldsymbol{z}_i$ in Algorithm 2) as observations in the model presented in Chapter 11. Formally the likelihood is defined similar to the SoR likelihood (Example 18) albeit scaled.

Definition 21 (Conjugate-gradients likelihood)**.** *Choose a subset $\boldsymbol{X}_M$ of size M from $\boldsymbol{X}$ and denote as $\boldsymbol{y}_M \in \mathbb{R}^M$ the vector that contains the corresponding entries of $\boldsymbol{y}$. Run conjugate gradients (Algorithm 2 on p. 15) with $\boldsymbol{x}_0 := \boldsymbol{0}$, $\boldsymbol{A} = k(\boldsymbol{X}_M, \boldsymbol{X}_M)$, $\boldsymbol{b} = \boldsymbol{y}_M$ and $\epsilon := 0.01||\boldsymbol{b}||_2$. In Eq. (32) set*

$$p_i(\boldsymbol{x}) := \sum_{j=1}^{M} s_j \delta(\boldsymbol{x} - \boldsymbol{x}_j) \tag{36}$$

where s_j is the j-th entry of vector $\boldsymbol{s}_i$ in iteration i of the CG algorithm.

Remark 22. *KMCG uses only the CG search directions $\boldsymbol{s}_1, ..., \boldsymbol{s}_P$ and* not *the solution $\hat{\boldsymbol{x}}$.*

Using this likelihood, the resulting approximate kernel (Eq. (34)) and approximate Equations are (cf. Proposition 41):

$$\hat{k}_M(\boldsymbol{x}_*, \boldsymbol{x}_{**}) = k(\boldsymbol{x}_*, \boldsymbol{X}_M)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{K}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal k(\boldsymbol{X}_M, \boldsymbol{x}_{**}) \tag{37}$$

$$\hat{f}(\boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{X}_M)\boldsymbol{S}(\boldsymbol{R}^\intercal \boldsymbol{R} + \sigma^2 \boldsymbol{S}^\intercal \boldsymbol{K}_M \boldsymbol{S})^{-1}\boldsymbol{R}^\intercal \boldsymbol{y} \tag{38}$$

$$\begin{aligned}\hat{c}(\boldsymbol{x}_*, \boldsymbol{x}_{**}) = {} & k(\boldsymbol{x}_*, \boldsymbol{x}_{**}) \\ & - k(\boldsymbol{x}_*, \boldsymbol{X}_M)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{K}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal k(\boldsymbol{X}_M, \boldsymbol{x}_{**})\end{aligned} \tag{39}$$

$$+\sigma^2 k(\boldsymbol{x}_*, \boldsymbol{X}_M)\boldsymbol{S}\left(\boldsymbol{R}^\top\boldsymbol{R}+\sigma^2\boldsymbol{S}^\top\boldsymbol{K}_M\boldsymbol{S}\right)^{-1}\boldsymbol{S}^\top k(\boldsymbol{X}_M, \boldsymbol{x}_{**})$$

$$\begin{aligned}\ln\hat{Z} &= \frac{1}{2\sigma^2}(\boldsymbol{y}^\top\boldsymbol{y}-\boldsymbol{y}^\top\boldsymbol{R}(\boldsymbol{R}^\top\boldsymbol{R}+\sigma^2\boldsymbol{S}^\top\boldsymbol{K}_M\boldsymbol{S})^{-1}\boldsymbol{R}^\top\boldsymbol{y}) \qquad (40)\\ &+\frac{1}{2}\ln|\boldsymbol{R}^\top\boldsymbol{R}+\sigma^2\boldsymbol{S}^\top\boldsymbol{K}_M\boldsymbol{S}|-\frac{1}{2}|\boldsymbol{S}^\top\boldsymbol{K}_M\boldsymbol{S}|\\ &+\frac{1}{2}(N-P)\ln\sigma^2+\frac{1}{2}N\ln 2\pi\end{aligned}$$

where $\boldsymbol{S} := [\boldsymbol{s}_1, \dots, \boldsymbol{s}_P], \boldsymbol{R} := k(\boldsymbol{X}_M, \boldsymbol{X})\boldsymbol{S}$ and P is the number of CG iterations.

Algorithm 4: Kernel Machine Conjugate Gradients

```
procedure KMCG($k$, $\boldsymbol{X}$, $\boldsymbol{y}$, $\sigma^2$, $\epsilon$)
    ▷ (W.l.o.g.) assume that the inducing inputs are a subset of $\boldsymbol{X}$.
    ▷ Denote this subset by $\boldsymbol{X}_M$.
    ▷ Let $\boldsymbol{y}_M$ the be corresponding entries of $\boldsymbol{y}$.
    Conjugate Gradients$(k(\boldsymbol{X}_M, \boldsymbol{X}_M), \boldsymbol{y}_M, \epsilon)$        // ignore solution $\hat{\boldsymbol{x}}$
    $\boldsymbol{S} \leftarrow [\boldsymbol{s}_1, ..., \boldsymbol{s}_P]$        // collect CG search directions
    $\boldsymbol{Z} \leftarrow [\boldsymbol{z}_1, ..., \boldsymbol{z}_P]$        // $\boldsymbol{Z} = \boldsymbol{K}_M\boldsymbol{S}$
    if $M < N$ then
        $\boldsymbol{R} \leftarrow k(\boldsymbol{X}, \boldsymbol{X}_M)\boldsymbol{S}$
    else
        $\boldsymbol{R} \leftarrow \boldsymbol{Z}$    // When $\boldsymbol{X}_M = \boldsymbol{X}$ above matrix multiplication is not necessary.
    end if
    $\boldsymbol{L}_1 \leftarrow \mathsf{chol}(\boldsymbol{S}^\top\boldsymbol{Z})$        // precompute required Choleskies
    $\boldsymbol{L}_2 \leftarrow \mathsf{chol}(\sigma^2\boldsymbol{S}^\top\boldsymbol{Z} + \boldsymbol{R}^\top\boldsymbol{R})$
    evaluate Eqs. (38) to (40)
end procedure
```

12.1 Properties

Fig. 5 shows how the approximation to the kernel progresses for the toy example from Fig. 3. Computing the Cholesky of $\boldsymbol{R}^\top\boldsymbol{R}+\sigma^2\boldsymbol{S}^\top\boldsymbol{K}_M\boldsymbol{S}$ costs $\mathcal{O}(NMP)$. After that evaluating the mean prediction is possible in $\mathcal{O}(M)$ and the variance in $\mathcal{O}(MP)$.

In case $P = M$, KMCG reduces to SoR since all occurrences of $\boldsymbol{S}$ in Eq. (37) cancel and what remains is the SoR kernel (Eq. (28)). If $\boldsymbol{K}_M$ has a favorable distribution of eigenvalues such that conjugate gradients terminates in less than M steps (see Section 5.2), KMCG can be used to speed up SoR.[1] In practice, this kind of advantage can only be expected to be beneficial when realized in low-level code. The level of efficiency of existing low-level linear algebra routines makes it challenging to evaluate this area.

Recall that the computational complexity of CG for the solution of Eq. (30) in P iterations is $\mathcal{O}(N^2P)$, that of inducing point methods with M inducing inputs is $\mathcal{O}(NM^2)$, and KMCG running for P iterations

[1] The same applies to related methods such as *DTC* (Quiñonero-Candela and Rasmussen 2005) and Titsias' method (Titsias 2009).

on M inducing points has complexity $\mathcal{O}(NMP)$. While the main point of the present paper is to "fix" problems of CG in kernel machines, this structure hints at an interesting side-observation: Restricting the number of steps P in advance can then allow to increase the number of inducing points M beyond what would otherwise be feasible with standard inducing input methods. The subsequent evaluation section is dedicated to the case $M = N$, *i.e.* using the whole dataset which places KMCG in direct competition to plain conjugate gradients.

12.1.1 Relationship to the Nadaraya-Watson estimator

Taking only one step ($P = 1$) implies $\boldsymbol{S} = \boldsymbol{y}_M$ and Eq. (38) takes the following form

$$\hat{f}(\boldsymbol{x}_*) = \alpha \sum_{m=1}^{M} k(\boldsymbol{x}_m, \boldsymbol{x}_*) y_m$$

where $\alpha = \frac{\boldsymbol{y}_M^\intercal \boldsymbol{K}_M \boldsymbol{y}_M}{\sigma^2 \boldsymbol{y}_M^\intercal \boldsymbol{K}_M \boldsymbol{y}_M + \boldsymbol{y}_M^\intercal \boldsymbol{K}_M \boldsymbol{K}_M \boldsymbol{y}_M}$. The equation bears resemblance to the Nadaraya-Watson estimator (Bishop 2006, p. 301f): a sum over all training targets weighted by the similarity of the corresponding input to the test input. However, the scaling-factor α is different.

12.1.2 Uncertainty

In addition to the posterior mean k_M, the Gaussian formulation of the approximation problem also provides a posterior variance ψ_M. It is a natural question to which degree this object can be interpreted as a notion of uncertainty or, more specifically, as an estimate of the square error $(k - k_M)^2$. This section provides an analysis of this covariance for KMCG, showing it to be an outer bound on the true error. Fig. 5 visualizes this for the toy dataset from Fig. 3.

Proposition 23 (relative error bound)**.** *The relative size of estimation error and error estimate is bounded from above by 2.*

$$\frac{(k(\boldsymbol{x}, \boldsymbol{z}) - k_M(\boldsymbol{x}, \boldsymbol{z}))^2}{\psi_M(k(\boldsymbol{x}, \boldsymbol{z}), k(\boldsymbol{x}, \boldsymbol{z}))} \leq 2 \tag{41}$$

Proof. Define $\boldsymbol{k}_x^\intercal := k(\boldsymbol{x}, \boldsymbol{X})$ and $\boldsymbol{G} := \boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{K} \boldsymbol{S})^{-1} \boldsymbol{S}^\intercal$. For KMCG posterior mean and variance evaluate to (Appendix E.2):

$$\begin{aligned} k_M(\boldsymbol{x}, \boldsymbol{z}) &= \boldsymbol{k}_x^\intercal \boldsymbol{G} \boldsymbol{k}_z, \\ \psi_M(k(\boldsymbol{x}, \boldsymbol{z}), k(\boldsymbol{x}, \boldsymbol{z})) &= \frac{1}{2}\left(k(\boldsymbol{x}, \boldsymbol{x}) k(\boldsymbol{z}, \boldsymbol{z}) + k(\boldsymbol{x}, \boldsymbol{z})^2\right) \\ &\quad - \frac{1}{2}\left(\boldsymbol{k}_x^\intercal \boldsymbol{G} \boldsymbol{k}_x \boldsymbol{k}_z^\intercal \boldsymbol{G} \boldsymbol{k}_z + (\boldsymbol{k}_x^\intercal \boldsymbol{G} \boldsymbol{k}_z)^2\right) \\ &= \frac{1}{2}\left(k(\boldsymbol{x}, \boldsymbol{x}) k(\boldsymbol{z}, \boldsymbol{z}) + k(\boldsymbol{x}, \boldsymbol{z})^2\right) \end{aligned}$$

$$- \left(k_M(\boldsymbol{x},\boldsymbol{x}) k_M(\boldsymbol{z},\boldsymbol{z}) - k_M(\boldsymbol{x},\boldsymbol{z})^2 \right).$$

As a variance $\psi_M(k(\boldsymbol{x},\boldsymbol{x}), k(\boldsymbol{x},\boldsymbol{x}))$ is always larger than 0 which implies $k(\boldsymbol{x},\boldsymbol{x}) \geq k_M(\boldsymbol{x},\boldsymbol{x})$ for all $\boldsymbol{x}$. Thus $\psi_M(k(\boldsymbol{x},\boldsymbol{z}), k(\boldsymbol{x},\boldsymbol{z}))$ is bounded from below by $\frac{1}{2}k(\boldsymbol{x},\boldsymbol{z})^2 - \frac{1}{2}k_M(\boldsymbol{x},\boldsymbol{z})^2$ from which we can conclude

$$\begin{aligned}
\frac{(k(\boldsymbol{x},\boldsymbol{z}) - k_M(\boldsymbol{x},\boldsymbol{z}))^2}{\psi_M(k(\boldsymbol{x},\boldsymbol{z}), k(\boldsymbol{x},\boldsymbol{z}))} &\leq 2\frac{(k(\boldsymbol{x},\boldsymbol{z}) - k_M(\boldsymbol{x},\boldsymbol{z})^2}{k(\boldsymbol{x},\boldsymbol{z})^2 - k_M(\boldsymbol{x},\boldsymbol{z})^2} \\
&= 2\frac{(k(\boldsymbol{x},\boldsymbol{z}) - k_M(\boldsymbol{x},\boldsymbol{z})^2}{(k(\boldsymbol{x},\boldsymbol{z}) - k_M(\boldsymbol{x},\boldsymbol{z}))(k(\boldsymbol{x},\boldsymbol{z}) + k_M(\boldsymbol{x},\boldsymbol{z}))} \\
&= 2\frac{|k(\boldsymbol{x},\boldsymbol{z}) - k_M(\boldsymbol{x},\boldsymbol{z})|}{k(\boldsymbol{x},\boldsymbol{z}) + k_M(\boldsymbol{x},\boldsymbol{z})} \\
&\leq 2.
\end{aligned}$$

□

12.2 Related Work

In terms of using conjugate gradients for kernel machines there is related work by Filippone and Engler (2015). Their algorithm ULISSE is aimed at the estimation of the marginal likelihood $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ where $\boldsymbol{\theta}$ are hyper-parameters of the kernel k. They use a randomized conjugate gradients to estimate gradients of the log-marginal likelihood (Eq. (3)) which in combination with Stochastic Gradient Langevin Dynamics (SGLD) (Welling and Teh 2011) allows to sample from $p(\boldsymbol{\theta} \mid \boldsymbol{y})$. Our work is complementary to ULISSE. While running CG the matrix multiplications the inference perspective in Chapter 11 can be used to build a low-rank approximation of the kernel matrix which can serve as preconditioner for the next SGLD step.

Using the Kronecker product for efficient inference has been explored before for example in the KISS-GP framework (Wilson and Nickisch 2015). The difference to this work is that Wilson and Nickisch (2015) factorize the kernel matrix $\boldsymbol{K}$ into a Kronecker-product where here it is the covariance matrix of the prior $\psi(\boldsymbol{K}, \boldsymbol{K})$ over the kernel that has Kronecker structure (cf. Eq. (30)). A synergy between their and our approach is hard to imagine. However, the follow-up work by Pleiss, Gardner, Weinberger, and Wilson (2018) uses Lanczos iteration to build a low-rank approximation of a kernel matrix $\boldsymbol{C}$ for the variance prediction. Presumably, one could use instead KMCG.

13 Empirical Comparison of CG and KMCG

This section elaborates the conceptual differences between CG and KMCG and then compares both algorithms with numerical experiments. Consider Eq. (1) restated below for convenience.

$$\bar{f}(\boldsymbol{x}_*) = \boldsymbol{k}_*^\intercal(\boldsymbol{K}+\sigma^2\boldsymbol{I})^{-1}\boldsymbol{y} \tag{1}$$

CG computes an approximation to $(\boldsymbol{K}+\sigma^2 I)^{-1}\boldsymbol{y}$ and uses the exact $\boldsymbol{k}_*$. In contrast, KMCG computes an approximation to k and substitutes $\boldsymbol{k}_*$ as well. That the systematic replacement of the kernel can be of importance has been noted before by Rasmussen and Williams (2006, p. 177) when comparing SoR and the Nyström method (Williams and Seeger 2001). The SoR method approximates k with the kernel in Eq. (28). In contrast Nyström uses the exact $\boldsymbol{k}_*$ such that the predictive variance (Eq. (2)) can become negative. They further observed that for large M, Nyström and SoR have a similar performance, yet for small M Nyström performs poorly. We will make the same observations for CG and KMCG in the following comparison.

Conjugate gradients is used to solve the equations $(\boldsymbol{K}+\sigma^2\boldsymbol{I})\boldsymbol{\alpha}=\boldsymbol{y}$. In contrast, since the goal of KMCG is to learn an approximation to the kernel, the algorithm runs conjugate gradients on $\boldsymbol{K}\boldsymbol{\alpha}=\boldsymbol{y}$, *i.e.* without noise term. Both methods were evaluated in terms of the average relative error

$$\epsilon_f := \frac{1}{n_*}\sum_{k=1}^{n_*}\left|\frac{\bar{f}(\boldsymbol{x}_{*,k})-\hat{f}(\boldsymbol{x}_{*,k})}{\bar{f}(\boldsymbol{x}_{*,k})}\right|, \tag{42}$$

where $\boldsymbol{x}_{*,k}$ is a test-input not part of the training set.

The text-book version of conjugate gradients in Algorithm 2 is known to be numerically unstable which is demonstrated in Appendix E.3.3, and there exist different strategies to cope with this problem (Golub and Van Loan 2013, p. 635). To explore the potential of KMCG, we bypass this implementation issue using the slowest[1] yet most stable solution: complete reorthogonalization Golub and Van Loan 2013, p. 564 and the explicit projection-method formulation, Eq. (FOM), to compute $\boldsymbol{\alpha}$. Therefore the following comparison will be conceptually, *i.e.* over the number of conjugate gradient steps. For completeness, Appendix E.3.1 contains results how KMCG performs in wall-clock time. Often the baseline methods converge faster since block-matrix multiplication is faster than looped matrix-vector multiplication. Baseline methods are the Fully Independent Training Conditional (FITC) approximation

[1] Computing the exact solution is actually faster.

(Quiñonero-Candela and Rasmussen 2005) and the Variational Free Energy (VFE) method (Titsias 2009) with inducing inputs randomly selected from the dataset as recommended by Chalupka, Williams, and Murray (2013). The baseline runs were repeated 10 times and besides the average, each figure will show also the progressive minimum and maximum over all runs to take into account for more elaborate inducing-input selection schemes.

In all our experiments, we used two popular stationary kernel functions: automatic relevance determination (ARD) Squared Exponential (Eq. (43)) and ARD Matérn 5/2 (Rasmussen and Williams 2006, p. 83f, p. 106),

$$k_{SE}(d(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\Lambda})) = \theta_f \exp\left(-\frac{1}{2}d^2\right) \tag{43}$$

$$k_{52}(d(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\Lambda})) = \theta_f \left(1 + \sqrt{5}d + \frac{5}{3}d^2\right) \exp\left(-\sqrt{5}d\right) \tag{44}$$

where $d = d(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\Lambda}) = ||\boldsymbol{x} - \boldsymbol{z}||_{\boldsymbol{\Lambda}}$ and $\boldsymbol{\Lambda}$ is a diagonal matrix. All experiments were executed with Matlab R2019a on an Intel i7 CPU with 32 Gigabytes of RAM running Ubuntu 18.04.

13.1 Common Regression Datasets

The datasets chosen are small such that computation of the exact GP is still feasible. Origin and purpose the datasets can be found in Table 3 in Appendix A. Each dataset has been shuffled and split into two sets, using one for training and the other for testing. For each dataset we optimized the kernel parameters running Carl Rasmussen's `minimize` function[2] for 100 optimization-steps, where initially all kernel hyper-parameters are set to 1.

Fig. 6 shows how the average relative error develops for the described setup. Since the Matérn kernel experiments look very similar, these figures are part of Appendix E.3.2. The upper x-axis displays the number of conjugate gradients steps, the lower x-axis, the number of inducing inputs. During early iterations the performance of CG is not as reliable as KMCG and the latter also improves more consistently. For the baselines, the number of inducing inputs M was set to $M = \sqrt{NP}$ such that $\mathcal{O}$-notation costs are equivalent to KMCG. (Since KMCG uses multiplications with $\boldsymbol{K}$ for observations, the costs per CG-step are $\mathcal{O}(N^2)$.) In comparison to the baselines, KMCG often provides a worse approximation to start with but exhibits a faster convergence rate.

In contrast to plain conjugate gradients, KMCG naturally provides estimates for variance (Eq. (2)) and evidence (Eq. (3)). Define the average relative errors ϵ_{var} and ϵ_{ev} analogously to Eq. (42), respectively. Figs. 7 and 8 show the average relative error of these estimates in comparison to the baselines. For all datasets one can observe that the

[2] This method is part of the GPML toolbox (Rasmussen and Nickisch 2010), see http://www.gaussianprocess.org/gpml/code/matlab/doc.

approximation quality of KMCG for the evidence (Eq. (3)) is improving at first and then worsening. KMCG is better at approximating the quadratic form than the determinant. Therefore, the approximation often 'overshoots'.

The baselines clearly outperform KMCG in these experiments. A possible explanation is that the baselines provide a better overall-approximation to the kernel matrix: After P CG-steps, the KMCG kernel is of rank P whereas using M inducing inputs, the VFE kernel is of rank M (so is the FITC kernel, putting the diagonal correction aside). Since $M = \sqrt{NP}$, the baselines can afford more inducing inputs M than KMCG can afford CG-steps P. Overall, when it comes to real-time, the baselines are preferable over KMCG. The picture changes when matrix-multiplication is less expensive than $\mathcal{O}(N^2)$ which is investigated in the next section.

13.2 Grid-structured Datasets

In the previous section the baselines are the preferable estimators over KMCG. This changes when matrix-multiplication costs less than $\mathcal{O}(N^2)$. For example when the kernel is a product kernel (such as squared exponential) and the dataset has grid-structure, the costs for matrix-multiplication are almost linearly in the number of data-points (Wilson, Gilboa, Nehorai, and Cunningham 2014) such that the number of CG-steps KMCG can take, matches the number of baseline inducing inputs.

13.2.1 Artificial Datasets

The datasets considered in the following are artificial multi-dimensional grids. For the training set[3], along each axis, G points are equally spaced in $[-G/4, G/4]$ distorted by Gaussian noise $\mathcal{N}(0, 10^{-3})$. One-hundred test inputs are uniformly distributed over the $[-G/4, G/4]$ cube. Targets are drawn from a Gaussian process with squared exponential kernel (length scales and amplitude equal to 1). The number of inducing inputs had to be capped at 500 due to memory limitations.

Fig. 9 shows how the approximation error to mean, variance and likelihood term evolves, zoomed in on the first 100 steps. In Appendix E.3.1, Fig. 16 shows the same comparison over time for the whole 500 steps, stopping KMCG when it becomes slower than the baselines. For reference, we include a 10 × 10 dataset to give an idea how each method would evolve when investing more computational power would be feasible.

On these datasets KMCG dominates the baseline methods. After already one-hundred CG-steps, KMCG provides a useful approximation

[3] Computing the exact solution is feasible exploiting the Kronecker structure of the kernel matrix which we use to evaluate the quality of the approximation methods. However, we may imagine datapoints missing, s.t. matrix-vector multiplication is fast but computing the exact solution is not.

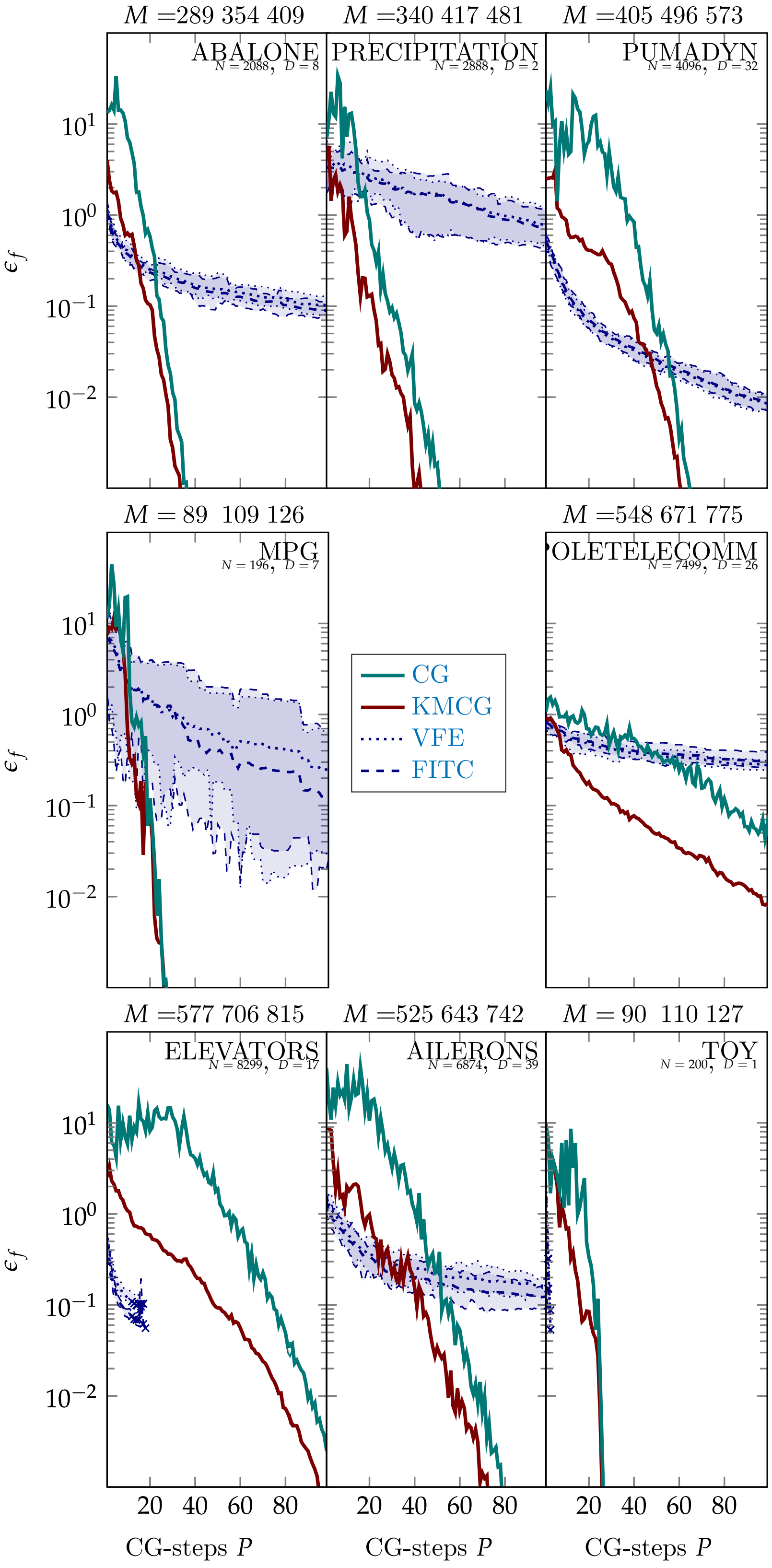


Figure 6: progression of the relative error ϵ_f as a function of the number of iterations of CG and KMCG for different datasets using the squared-exponential kernel (Eq. (43)). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

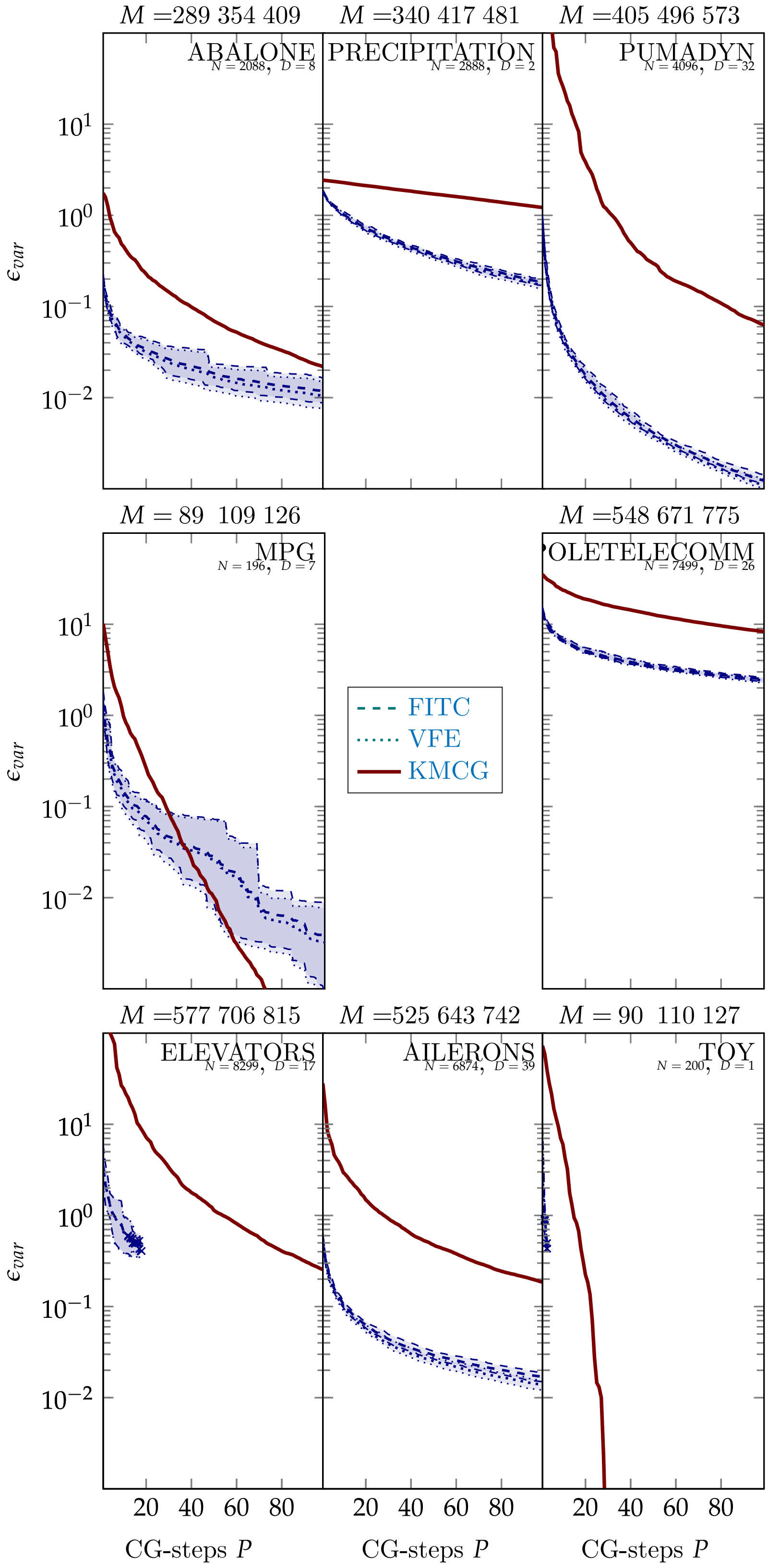


Figure 7: progression of the relative error of the variance ϵ_{var} as a function of the number of iterations of KMCG and baseline for different datasets using the squared-exponential kernel (Eq. (43)). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

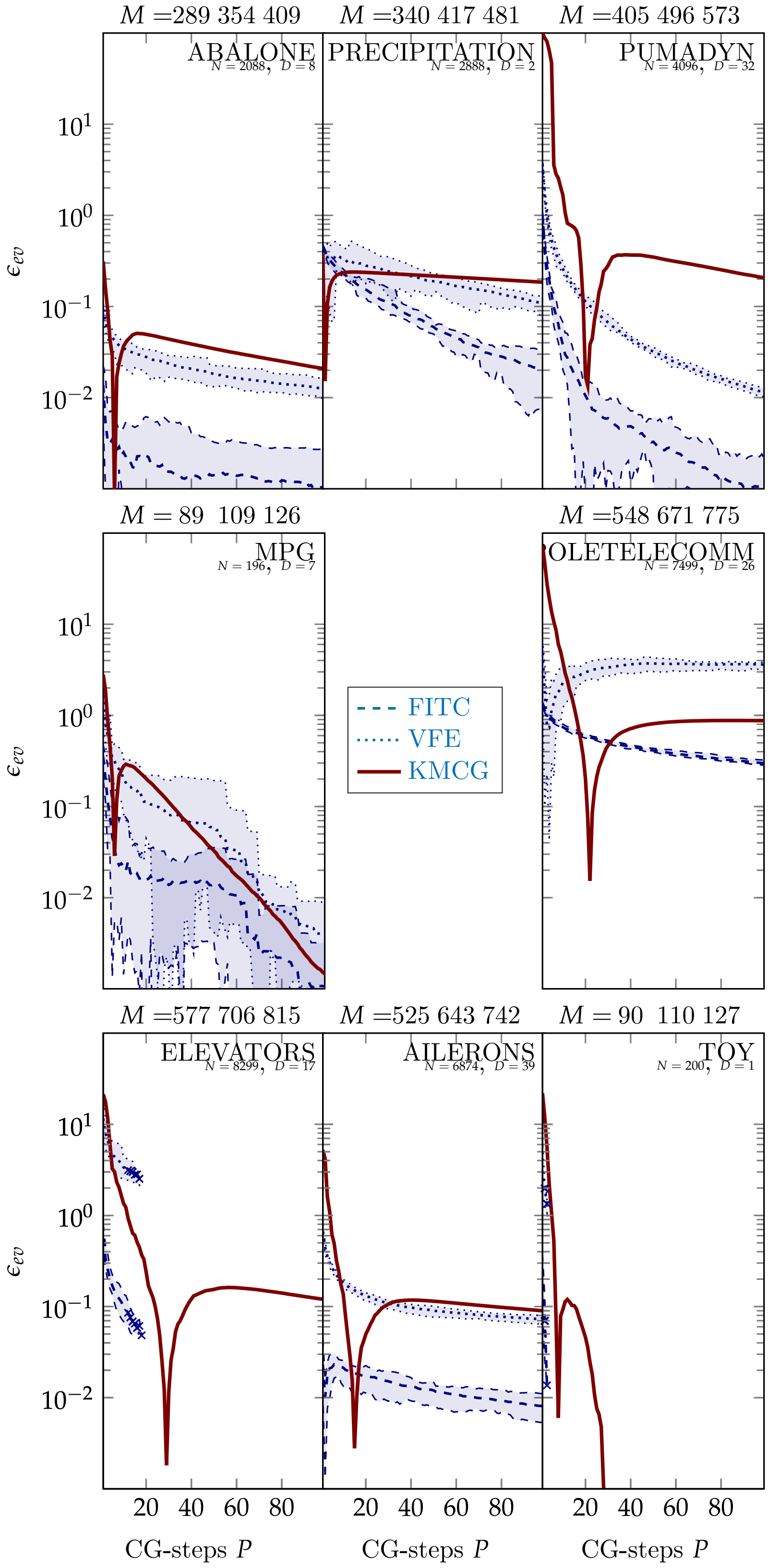


Figure 8: progression of the relative error of the evidence ϵ_{ev} as a function of the number of iterations of baseline and KMCG for different datasets using the squared-exponential kernel (Eq. (43)). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run. The small spikes in the plots where KMCG appears to be close to the solution correspond to changes of the estimate from too small to too large.

to the posterior mean whereas the baselines hardly show any progress. For the variance, the same computational effort is not enough. Though the baselines find better solutions, all methods essentially fail to arrive at a satisfactory solution of a relative error below one. The issue is that all methods overestimate the posterior variance by two orders of magnitude. The picture is similar for the evidence, albeit the approximations are closer to the truth and KMCG performs slightly better on average.

13.2.2 Natural Sound Modeling

For a real-world example of a grid-structured dataset, we repeat the Natural Sound Modeling experiment considered by Turner (2010), Wilson and Nickisch (2015), and Dong, Eriksson, Nickisch, Bindel, and Wilson (2017). Given the intensity of a sound signal recorded over time, the objective is to recover the signal in missing regions. All inputs (*i.e.* including missing) are equidistant and hence the kernel matrix (over all inputs) is Toeplitz for stationary kernel. The kernel matrix over the given inputs is not Toeplitz, which forbids to use this structure for exact inference. Nevertheless matrix-vector-multiplication can be performed in linear time. We use the squared-exponential kernel with the hyper-parameters used by Dong, Eriksson, Nickisch, Bindel, and Wilson (2017). Since the exact posterior is infeasible to compute we report only the standardized mean squared-error:

$$SMSE := \frac{1}{\mathbb{V}[\boldsymbol{y}]} \sum_{j=1}^{N_*} (\boldsymbol{y}_{*,j} - \hat{f}(\boldsymbol{x}_{*,j}))^2.$$

To conform with the original experiment, we added for each baseline a run where the inducing inputs where chosen to be on a regular grid. The result of this run correspond to the minimum. Fig. 10 confirms the observations of the previous section that KMCG arrives at satisfactory solutions faster than baseline, if matrix-vector multiplication is not an issue.

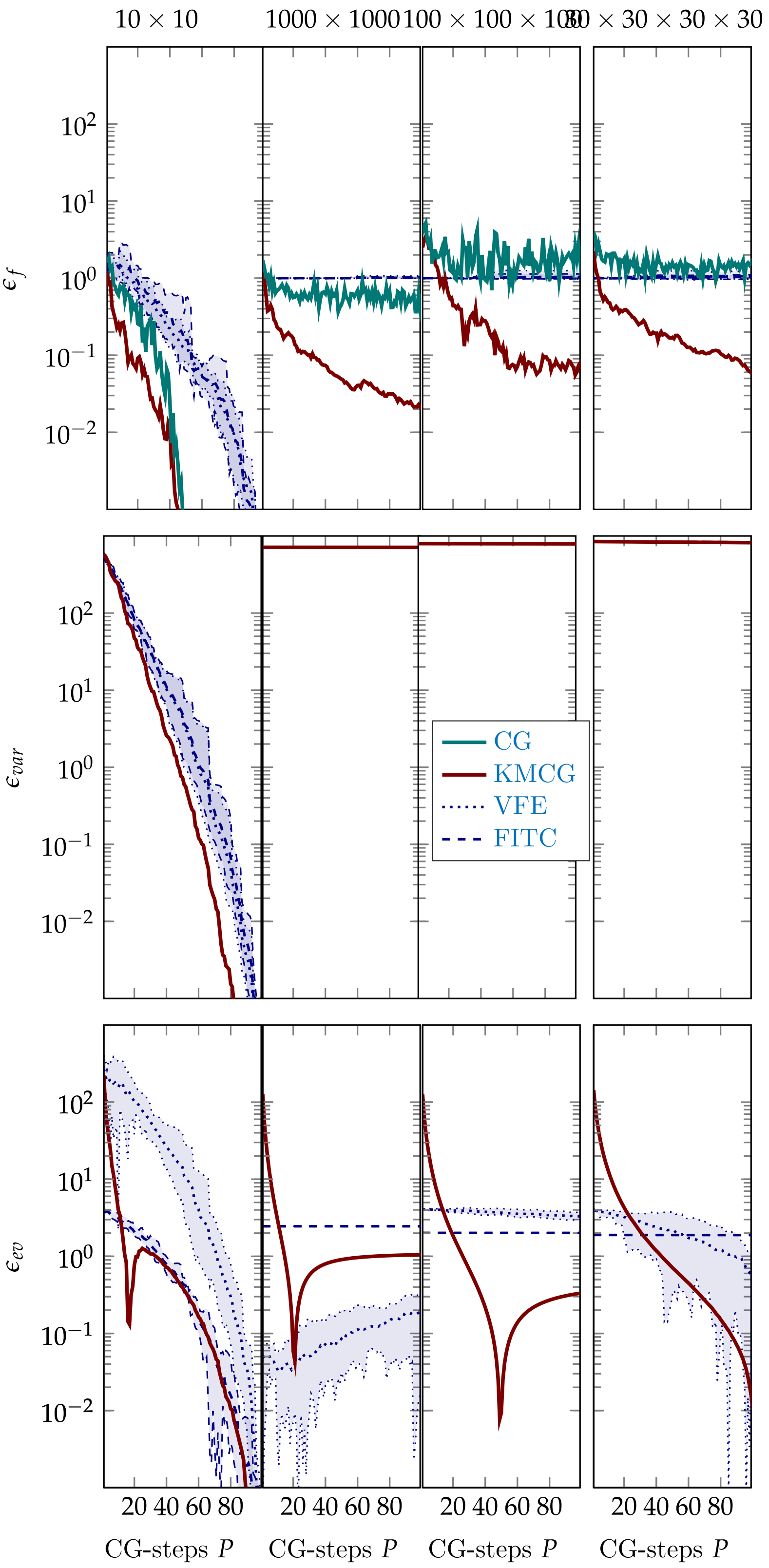


Figure 9: comparison of baseline and KMCG on grid-structured datasets using the squared exponential kernel (Eq. (43)). The shaded area visualizes minimum and maximum over all baseline runs.

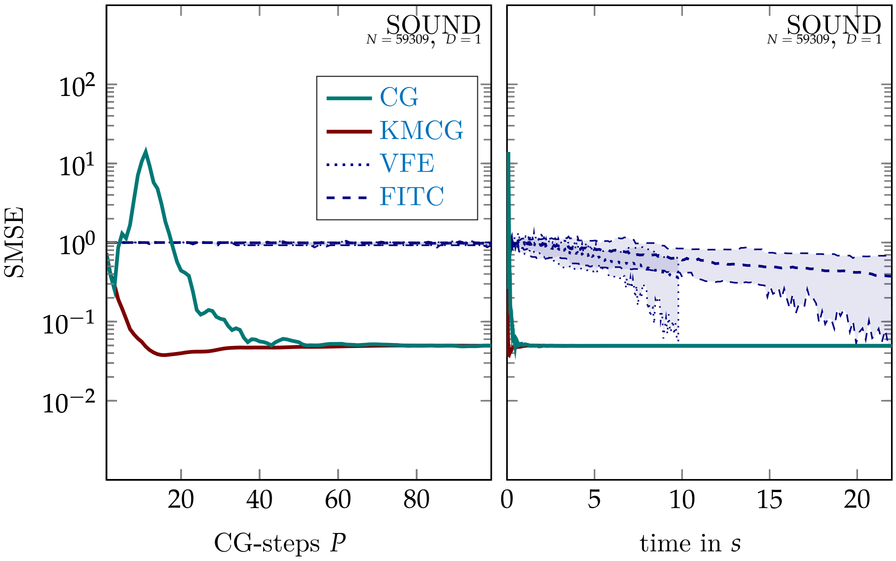


Figure 10: comparison of KMCG and CG on the SOUND dataset using the squared exponential kernel (Eq. (43)) with the hyper-parameters from Dong, Eriksson, Nickisch, Bindel, and Wilson (2017). The shaded area visualizes minimum and maximum over all baseline runs.

14 Discussion

14.1 Summary

This part presented a new approximate inference method for kernel machines that showed how linear solvers can be used in combination with low-rank kernel approximations. The approach is based on a probabilistic numerics viewpoint: the kernel k is treated as a latent quantity and conjugate gradients is used for collecting observations of k. By design, the resulting approximate kernel is of low rank and is plugged into the nonparametric least-squares problem. The approach is not restricted to least-squares problems but applicable in any scenario where the bottleneck is the inversion of a large kernel matrix.

Kernel machine conjugate gradients (KMCG) consistently outperforms plain conjugate gradients in numerical experiments. This does not change the fact that standard dense kernel least-squares problems are often more efficiently solved by inducing point methods. However, as demonstrated in Section 13.2, in the settings which allow fast multiplication with the kernel matrix, the new algorithm can improve upon the state of the art.

14.2 Future Directions

A hope associated with KMCG was that the probabilistic approach would allow to reason about the approximation error. However, this endeavor turned out to be more challenging than anticipated. For example, for an uncertainty over the approximate posterior mean, Eq. (1), this requires propagating the uncertainty over the kernel matrix through a matrix inverse operation. However, even if x is only univariate Gaussian, only the 0-th moment for $^1/_x$ exists. Hence, to derive a meaningful uncertainty, requires the additional step of formalizing another ground-truth. If there exist a lower bound or an upper bound on x which guarantee that x is necessarily larger or smaller than 0, one can use a truncated Gaussian. In that case higher moments of x^{-1} exist. Generalizing this approach to the multivariate case is more challenging.

Putting aside the issue of defining $p(\boldsymbol{K}^{-1})$, evaluating this expression will likely require approximation. Given a belief over $\boldsymbol{K}$, one direction could be to use a consequence of the Cayley–Hamilton theorem: the inverse of an invertible $d \times d$ matrix $\boldsymbol{A}$ can be written as

$$\boldsymbol{A}^{-1} = \sum_{j=0}^{d-1} c_j \boldsymbol{A}^j,$$

where the coefficients c_j also depend on $\boldsymbol{A}$. Using above expansion for $\boldsymbol{K}$ and $\hat{\boldsymbol{K}}$, results in the following expression for the inverse:

$$\boldsymbol{K}^{-1} = \hat{\boldsymbol{K}}^{-1} + \sum_{j=0}^{N-1} (c_j \boldsymbol{K}^j - \hat{c}_j \hat{\boldsymbol{K}}).$$

Under the assumption that second order and higher polynomials contribute little, one can use the approximation

$$\boldsymbol{K}^{-1} \approx \hat{\boldsymbol{K}}^{-1} + (c_0 - \hat{c}_0)\boldsymbol{I} + c_1 \boldsymbol{K} - \hat{c}_1 \hat{\boldsymbol{K}}.$$

Treating c_0, c_1 as constant, above expression contains only $\boldsymbol{K}$ as latent variable, and one obtains:

$$\mathbb{E}\boldsymbol{K}^{-1} \approx \hat{\boldsymbol{K}}^{-1} + (c_0 - \hat{c}_0)\boldsymbol{I} + (c_1 - \hat{c}_1)\hat{\boldsymbol{K}} \text{ and}$$
$$\mathbb{V}\boldsymbol{K}^{-1} \approx c_1^2 \mathbb{V}\boldsymbol{K}.$$

To examine the usefulness of this idea, next steps would be to test the assumption if second order and above polynomials can be ignored, to find means of estimating $c_0 - \hat{c}_0$, $c_1 - \hat{c}_1$ and c_1^2 with reasonable effort, and to examine the relationship to a first-order Taylor expansion.

Part V

Probabilistic Kernel-Matrix Determinant Estimation

> “This is just what you need, old pal. Me and you on a bender with a few beautiful ladies. I’m going over there.” “No.” “Oh, yes. I may be tiny, but I’ve got a certain *je ne sais quoi*.” “A certain what?” “I don’t know what,” admitted Zaphod, “But that’s never stopped me before.”

—Eoin Colfer, *And Another Thing...*

15 Preliminaries

15.1 Introduction

Whereas the previous parts have been concerned with the questions how new numerical approximation algorithms can be derived from a statistical perspective, this part deals with exploiting statistical properties of the numerical problem. The particular problem under consideration is the estimation of log-determinants of kernel matrices, a vital component of inference with models like determinantal point processes or Gaussian processes (*c.f.* Chapter 4). Given a set of $\boldsymbol{x}_1, ..., \boldsymbol{x}_N \in \mathbb{X}$ inputs and a kernel function k, the task is to compute

$$\ln |\boldsymbol{K}_N|$$

where

$$\boldsymbol{K}_N := \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & & \vdots \\ \vdots & & \ddots & \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & \dots & & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix}.$$

Fig. 11 shows the progression of $\ln |\boldsymbol{K}_n + 0.001\boldsymbol{I}|$ with increasing n for different benchmark datasets and the squared-exponential kernel (Eq. (43)). The figure demonstrates that there exist cases in which a linear extrapolation from some $\ln |\boldsymbol{K}_n|$, $n < N$, is sufficient to estimate $\ln |\boldsymbol{K}_N|$. The success of such a strategy depends two factors: the kernel and the inputs $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$.

The goal of this part is to develop a stopping strategy that recognizes such "easy" cases while computing the log-determinant and terminates the calculation if possible. Early stopping would be helpful during hyper-parameter tuning for Gaussian processes. For example, consider the case of a kernel matrix generated from an ARD squared-exponential kernel (Eq. (43)) with lengthscales far too long for the dataset. In that case, a subset is sufficient to make accurate predictions and the progression of the determinant is easy to predict. An algorithm capable of early stopping, allows to try these kernel parameters "cheaply". Contrarily, not early stopping can be a sign that the amount of data is not sufficient with respect to the flexibility of the model.

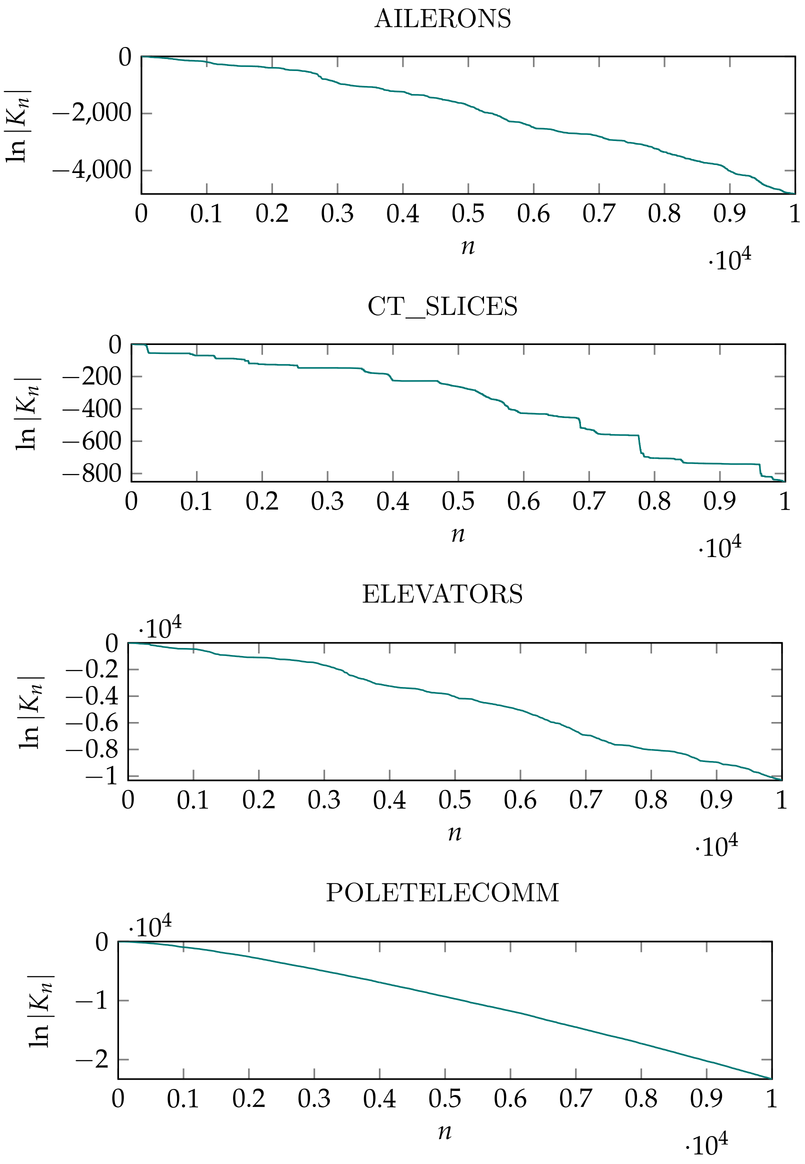


Figure 11: evolution of the log determinant $\ln|\boldsymbol{K}_n + 0.001\boldsymbol{I}|$ for common benchmark problems using the squared-exponential kernel Eq. (43). The datasets have been centered and standardized but are otherwise unprocessed. Note the different magnitudes of the y-axis. The figure demonstrates that depending on the kernel and the distribution of the dataset, there are cases in which a linear extrapolation from some $\ln|\boldsymbol{K}_n|$ is sufficient to estimate $\ln|\boldsymbol{K}_N|$ with high relative precision. More details about the datasets can be found in Appendix A.

15.2 The Cholesky Decomposition

To evaluate $\ln|\boldsymbol{K}_N|$ one usually computes the Cholesky decomposition $\boldsymbol{C}$ which is a triangular matrix satisfying $\boldsymbol{C}\boldsymbol{C}^\intercal = \boldsymbol{K}_N$. Given the Cholesky, one obtains the log-determinant by using $\ln|\boldsymbol{K}_N| = 2\sum_{j=1}^{N} \ln \boldsymbol{C}_{jj}$ (Lemma 49, p. 130). In the following, we will focus on an implementation of the Cholesky decomposition that proceeds column-wise over the elements of $\boldsymbol{K}_N$, Algorithm 5. The motivation being, that a blocked version (Golub and Van Loan 2013, p. 170) of this algorithm is used in *OpenBLAS* (Wang, Zhang, Zhang, and Yi 2013). Presumably, closed source libraries, such as *Intel MKL*, use the same implementation. Observe that the costs per step j scale as $\mathcal{O}(j + (N-j)\cdot j)$, *i.e.* the costs increase until $j = {}^{N}/_{2}$ and then decrease again. Hence, the saved time when stopping "late" could be insignificant. However, in the algorithm as proposed by Cholesky (Benoit 1924), the effort per step scales as $\mathcal{O}(j^2)$, s.t. even when stopping only in one of the last steps, the saved effort is considerable.

Note that computing $\boldsymbol{C}_{jj}$ requires access only to the first $\boldsymbol{x}_1, ..., \boldsymbol{x}_j$ datapoints. Under the assumption that the $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$ are independent and identically distributed, one can show that the $\boldsymbol{C}_{jj}$ decrease in expectation, *i.e.* $\mathbb{E}[\boldsymbol{C}_{j+1,j+1}] \leq \mathbb{E}[\boldsymbol{C}_{jj}]$ (see Lemma 32, p. 91). This allows to design a stopping condition that interrupts the calculation of the Cholesky decomposition and returns an estimate that is correct up to a relative error with high probability. In the following Chapter 16, we will consider a more general problem.

```
procedure CHOLESKY($\boldsymbol{A}$)
    for $j = 1, \dots, N$ do
        for $k = 1, \dots, j-1$ do
            $\boldsymbol{A}_{jj} \leftarrow \boldsymbol{A}_{jj} - \boldsymbol{A}_{jk}\boldsymbol{A}_{jk}$
        end for
        $\boldsymbol{A}_{jj} \leftarrow \sqrt{\boldsymbol{A}_{jj}}$
        for $\ell = j+1, \dots, N$ do
            for $k = 1, \dots, j-1$ do
                $\boldsymbol{A}_{\ell j} \leftarrow \boldsymbol{A}_{\ell j} - \boldsymbol{A}_{ik}\boldsymbol{A}_{jk}$
            end for
            $\boldsymbol{A}_{\ell j} \leftarrow {}^{\boldsymbol{A}_{\ell j}}/_{\boldsymbol{A}_{jj}}$
        end for
    end for
end procedure
```

Algorithm 5: Cholesky decomposition according to Meister (2015, p. 48). The letters of the indices have been adapted to fit with the notation here. A blocked version of the algorithm can be found in Golub and Van Loan (2013, p. 170).

15.3 Martingales and Stopping Times

The following elaborations require some notation and terminology. For a more thorough introduction, these terms are discussed by Grimmett and Stirzaker (2001), and Davidson (1994).

For a monotonically increasing function $f : \mathbb{R} \mapsto \mathbb{R}$ and $\delta \in \mathbb{R}$, define $f^{-1}(\delta) := \arg\sup_{\epsilon \in \mathbb{R}} \{f(\epsilon) \leq \delta\}$. The following definitions can be more general, but for our purposes, discrete and positive indices in $\mathbb{N}$ will be sufficient. A *filtration* is a sequence $(\mathcal{F}_j)_{j\in\mathbb{N}}$ of increasing σ-algebras, *i.e.* $\mathcal{F}_j \subseteq \mathcal{F}_{j+1}$ for all $j \in \mathbb{N}$. For random variables $X_1, ..., X_N$ denote by $\sigma(X_1, ..., X_N)$ the generated σ-Algebra. A sequence of random variables $(X_j)_{j\in\mathbb{N}}$ is called *adapted* to a filtration, if X_j is $\mathcal{F}_j$-measurable for all $j \in \mathbb{N}$. Let $(X_j)_{j\in\mathbb{N}}$ be a sequence of random variables adapted to a filtration $\mathcal{F}_{j\in\mathbb{N}}$. A sequence $(X_j, \mathcal{F}_j)_{j\in\mathbb{N}}$ is called a *martingale* if

filtration

adapted

martingale

$$\begin{aligned} \mathbb{E}|X_j| &< \infty, \text{ and} \\ \mathbb{E}[X_{j+1} \mid \mathcal{F}_j] &= X_j \text{ almost surely,} \end{aligned}$$

for all $j \in \mathbb{N}$. A sequence $(X_j, \mathcal{F}_j)_{j\in\mathbb{N}}$ is called a *martingale difference*, if

martingale difference

$$\begin{aligned} \mathbb{E}|X_j| &< \infty, \text{ and} \\ \mathbb{E}[X_j \mid \mathcal{F}_{j-1}] &= 0 \text{ almost surely,} \end{aligned}$$

for all $j \in \mathbb{N}$. A random variable τ is called a *stopping time* (w.r.t. a filtration), if it takes values in $\mathbb{N}$ and $\{\tau = j\} \in \mathcal{F}_j$ for all $j \in \mathbb{N}$.

stopping time

16 A Probably Approximately Correct Bound

16.1 Problem Definition

Let $(\Omega, \mathcal{F}, P)$ be a probability space and $(\mathcal{F}_j)_{j\in\{1,...,N\}}$ be a filtration. Furthermore, let $(f_j)_{j\in\mathbb{N}} \in [C^-, C^+]$ be a sequence of $\mathcal{F}_j$-measurable random variables that decrease conditionally in expectation,

$$\mathbb{E}[f_{j+1} \mid \mathcal{F}_j] \le \mathbb{E}[f_j \mid \mathcal{F}_{j-1}] \tag{$*$}$$

for $j \in \{1, ..., N-1\}$, where $\mathcal{F}_0 := \{\emptyset, \mathbb{R}\}$, and define

$$D_N := \sum_{j=1}^{N} f_j. \tag{45}$$

Given a desired precision $r \in (0, 1)$ and failure chance $\delta \in (0, 1)$, the problem is to device a strategy that, being presented sequentially with the $f_1, f_2, ...$, decides in each step whether to continue or to stop, and if stopping, provides an estimator $\hat{D}_\tau$, s.t. its relative error is less than r with probability $1 - \delta$. Formally, the goal is to device a stopping time τ and an estimator $\hat{D}_\tau$, s.t.

$$P\left(\mathsf{abs}\left(\frac{D_N - \hat{D}_\tau}{D_N}\right) > r\right) \le \delta. \tag{46}$$

Remark 24. *A trivial solution is to define $\tau := N$ and $\hat{D}_\tau := D_N$.*

In absence of related work (*c.f.* Section 16.4), formal evaluation criteria for τ and $\hat{D}_\tau$ will not be developed. The goal will be to design a strategy that satisfies Eq. (46), and for at least one scenario: $\tau < N$.

16.2 Stopping Condition

The design of the stopping conditions are based on the following lemma.

Lemma 25 (Bound on Relative Error)**.** *Let $D, \hat{D} \in [\mathcal{L}, \mathcal{U}]$, and assume $\mathsf{sign}(\mathcal{L}) = \mathsf{sign}(\mathcal{U}) \neq 0$. Then the relative error of the estimator $\hat{D}$ can be bounded as*

$$\frac{\mathsf{abs}(D - \hat{D})}{\mathsf{abs}(D)} \le \frac{\max(\mathcal{U} - \hat{D}, \hat{D} - \mathcal{L})}{\min(\mathsf{abs}(\mathcal{L}), \mathsf{abs}(\mathcal{U}))}.$$

Proof. The proof is part of Appendix F. □

Remark 26. *The bound is minimized for the choice* $\hat{D} := {}^{1}\!/_{2}(\mathcal{L} + \mathcal{U})$*. In that case*

$$\frac{\mathrm{abs}(D - \hat{D})}{\mathrm{abs}(D)} \leq \frac{\mathcal{U} - \mathcal{L}}{2\min(\mathrm{abs}(\mathcal{L}), \mathrm{abs}(\mathcal{U}))}.$$

In the following, we will device lower bounds $\mathcal{L}_n$ and upper bounds $\mathcal{U}_n$ to D_N, respectively, for $n = 1, ..., N$. The lower bounds $\mathcal{L}_n$ will be deterministic, whereas $D_N \leq \mathcal{U}_n$ only with a certain probability. The stopping time τ will monitor these bounds and stop if they are large in magnitude (away from zero) and close enough that the relative error can not exceed the desired precision r.

Describing stopping strategy and estimator formally, will necessitate a number of definitions. The reader can quickly access all of them in Table 2 on p. 82. Define $C := C^{+} - C^{-}$ and $l_j := f_j - C^{-}$. Observe that l_j is bounded from above by C and from below by 0. The latter property is the reason for introducing this definition.

Set

$$\mathcal{L}_n := NC^{-} + \sum_{j=1}^{n} l_j$$

and define an empirical mean of the "last" m elements, ending in n:

$$\hat{\mu}_n := \frac{1}{m} \sum_{j=n-m+1}^{n},$$

where $m \in \{1, ..., N-1\}$ is a user defined nuisance parameter. Our estimator will be

$$\hat{D}_n := {}^{1}\!/_{2}(\mathcal{L}_n + \mathcal{U}_n). \tag{47}$$

To construct $\mathcal{U}_n$, define the function

$$H_N(x) := H(x, N) = \mathbf{1}_{\{x \leq N\}} \sqrt{\left(\frac{N}{N+x}\right)^{N+x} \left(\frac{N}{N-x}\right)^{N-x}}$$

where $H(x, N)$ is defined in Theorem 29 (p. 84). The function $H_N(x)$ will be used to bound the probability that the upper bound $\mathcal{U}_n$ fails. Fig. 12 visualizes this function for different N. Define the error tolerance for the mean $\hat{\mu}_n$ as $\epsilon_\mu := {}^{C}\!/_{m} H_m(\frac{\delta}{2|S|})$, where $S \subseteq \{m+1, ..., N-1\}$ is another nuisance parameter. The set S defines possible stopping points. It is an artifact from the proof of Theorem 27 which may become obsolete (see Section 19.2). Define the error tolerance for the upper bound $\epsilon_n := (N - n)\epsilon_\mu + CH_N^{-1}({}^{\delta}\!/_{2})$, and the upper bound

$$\mathcal{U}_n := \mathcal{L}_n + \min((N-n)\hat{\mu}_n + \epsilon_n, (N-n)C).$$

$\mathcal{L}_n + (N-n)C$ is a deterministic upper bound to D_N and taking the minimum ensures that $\mathcal{U}_n$ is never worse than that. At last, define the stopping time

$$\tau := \min \left\{ n \in S \,\middle|\, \operatorname{sign}(\mathcal{U}_n) = \operatorname{sign}(\mathcal{L}_n) \neq 0, \text{ and} \right. \tag{C1}$$

$$\left. \frac{\mathcal{U}_n - \mathcal{L}_n}{2\min(\operatorname{abs}(\mathcal{U}_n), \operatorname{abs}(\mathcal{L}_n))} \leq r \right\} \cup \{N\}. \tag{C2}$$

Note that the quantities in the stopping conditions are $\mathcal{F}_n$-measurable, *i.e.* τ defines indeed a stopping time.

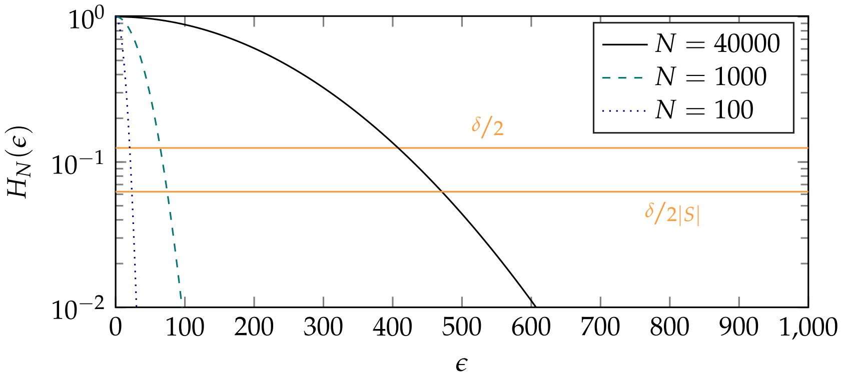


Figure 12: the function $H_N(x)$ for different N. Increasing N requires a larger ϵ for $H_N(\epsilon)$ to fall below a certain threshold. However, in relation to N the increase is small. The thresholds $\delta/2$ and $\delta/2|S|$ become relevant for Fig. 14.

Theorem 27. *Let $(\Omega, \mathcal{F}, P)$ be a probability space and $(\mathcal{F}_j)_{j\in\{1,...,N\}}$ be a filtration. Furthermore, let $(f_j)_{j\in\mathbb{N}} \in [C^-, C^+]$ be a sequence of $\mathcal{F}_j$-measurable random variables that decrease conditionally in expectation,*

$$\mathbb{E}[f_{j+1} \mid \mathcal{F}_j] \leq \mathbb{E}[f_j \mid \mathcal{F}_{j-1}] \tag{$*$}$$

for $j \in \{1, ..., N-1\}$, where $\mathcal{F}_0 := \{\emptyset, \mathbb{R}\}$, and define

$$D_N := \sum_{j=1}^{N} f_j.$$

For $r, \delta \in (0,1)$, the probability that the relative error of the estimator $\hat{D}_\tau$ defined by Eqs. (47),(C1) and (C2) is larger than r is less than δ.

$$P\left(\operatorname{abs}\left(\frac{D_N - \hat{D}_\tau}{D_N}\right) > r\right) \leq \delta \tag{48}$$

The proof will be presented in Chapter 17.

Theorem 28. *Assume $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{X}$ are independent and identically distributed. Denote with P the law of the $\boldsymbol{x}_1, .., \boldsymbol{x}_N$ and with $\boldsymbol{C}$ the Cholesky decomposition of $\boldsymbol{K}_N + \sigma^2\boldsymbol{I}$, where $\sigma^2 > 0$. Define* *the probability space* $(\mathbb{X}, \sigma(\boldsymbol{x}_1, ..., \boldsymbol{x}_N), P)$ *and the canonical filtration* $\mathcal{F}_j := \sigma(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j)$ *for* $j = 1, ..., N$. *Further, define*

$$\begin{aligned} f_j &:= 2 \ln |\boldsymbol{C}_{jj}|, \text{ with} \\ C^- &:= \ln \sigma^2 \text{ and} \\ C^+ &:= \max_{j=1,...,N} \ln(k(\boldsymbol{x}_j, \boldsymbol{x}_j) + \sigma^2). \end{aligned}$$

Then, using the definitions of Theorem 27,

$$P\left(\frac{\operatorname{abs}(|\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}| - \hat{D}_\tau)}{\operatorname{abs}(|\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}|)} > r\right) \leq \delta.$$

The proof will be presented in Chapter 18.

16.3 Discussion

For Theorem 28 the quantities required for the stopping time are inexpensive to compute compared to the elements of the Cholesky. I modified the *OpenBLAS* (Wang, Zhang, Zhang, and Yi 2013) Cholesky decomposition to accommodate the stopping rule. Fig. 13 shows the relative overhead when computing the log-determinant of identity matrices of different size. The identity matrix is a special case, where early stopping is impossible—since $\ln |\boldsymbol{I}| = 0$, there exists no $\delta < 1$ for which a finite relative error could be guaranteed. One can see that with increasing dataset size the overhead becomes less than one percent on average. In that sense, Theorem 28 offers an "almost-free lunch".

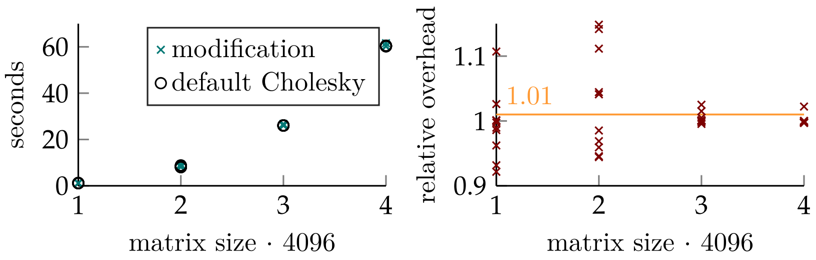


Figure 13: relative overhead of the OpenBLAS Cholesky decomposition with early stopping against without. Both algorithms compute the Cholesky decomposition of identity matrices. The nuisance parameters were set to $m := 256$ and $S := m\lfloor\{m, ..., N-1\}/m\rfloor$. The parameters r and δ are in this case irrelevant for the performance, since early stopping is impossible by design. With increasing matrix size, the measurement noise is decreasing and more runs remain below the one percent overhead mark. The experiments were executed on an Intel i7 CPU with 32 Gigabytes of RAM running Ubuntu 18.04.

Theorem 27 requires the user to define two nuisance parameters m and S. Fig. 14 demonstrates their influence on the bound $\mathcal{U}_n$ and whether the stopping conditions can be met. The parameter m sets the number of elements l_j preceding the current step, used in the empirical mean $\hat{\mu}_n$ to build the upper bound $\mathcal{U}_n$. When setting the parameter too small, the target probability δ could be unattainable. On the other hand, for larger m, it is more likely that the upper bound $\mathcal{U}_n$ is more conservative, hence delaying termination.

Furthermore, the user needs to select a subset $S \subseteq \{m, ..., N-1\}$ of possible stopping points. This is due to a union bound in the proof of

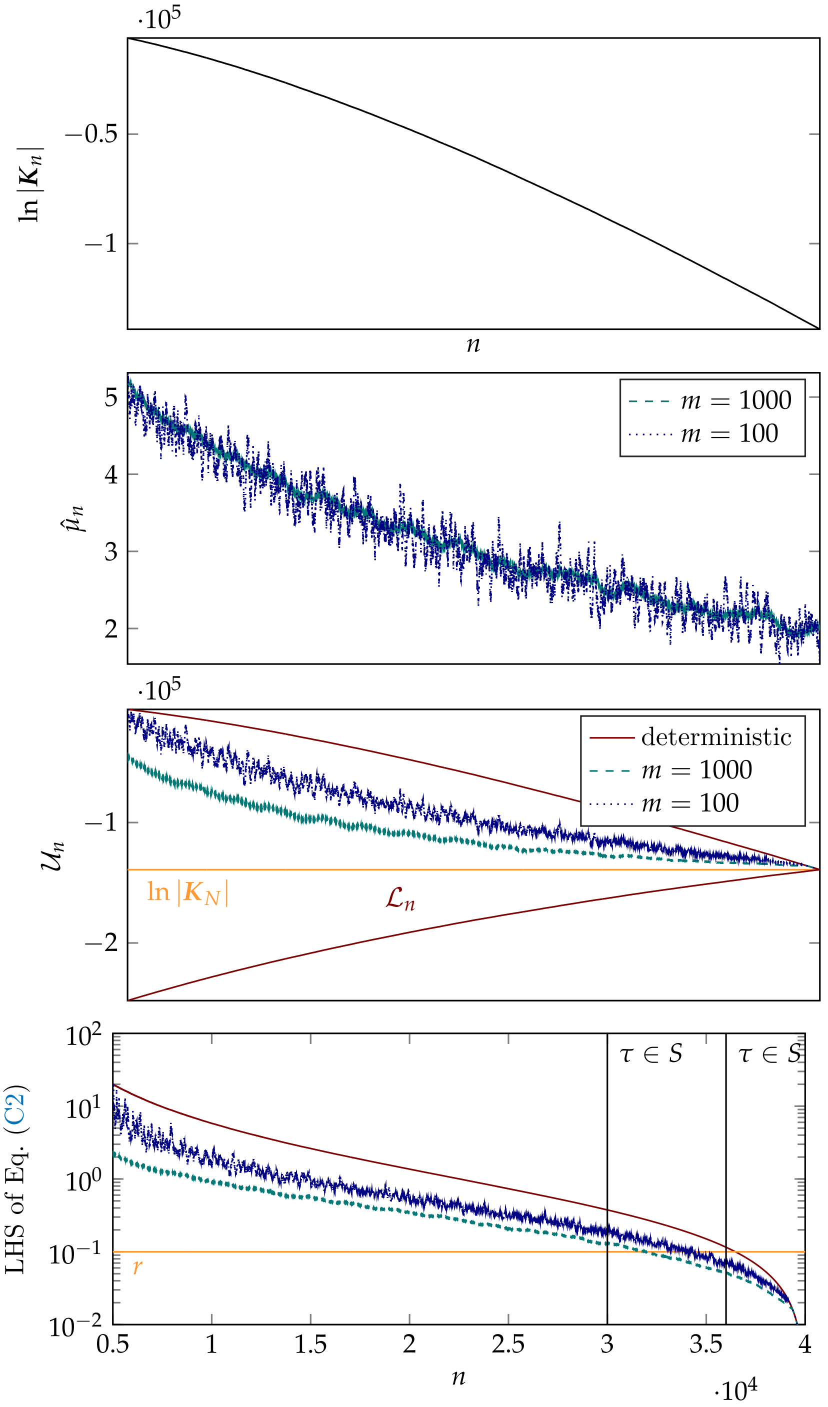


Figure 14: visualization of the quantities of Theorem 28 for one random shuffling of the SARCOS dataset ($N = 40000$), the squared-exponential kernel (Eq. (43)) and $\sigma^2 := 0.001$. The parameters have been set to $r := 0.1, \delta := 0.25, m := 100, 1000$ and $S := \{0.75N, 0.9N\}$. The deterministic upper bound is $\mathcal{U}_n := \mathcal{L}_n + (N-n)C^+$. It may seem surprising that the upper bound for $m = 100$ is higher than for $m = 1000$. However, recall that ϵ_μ has a term C/m. In the last picture, both stopping conditions are fulfilled for $\tau = 0.9N$, for both m. This figure demonstrates that Theorem 28 is not trivial. Arguably, it is desirable to achieve early stopping for less conservative parameters than $|S| = 2$ and $r = 0.1$. Section 19.2 discusses possibilities how to obtain more practical versions of Theorem 27.

Theorem 27 (Eq. (54)). If $|S|$ is too large, it may not be possible to stay below the desired probability δ. On the other hand, choosing fewer stopping points implies fewer opportunities to terminate. Section 19.2 discusses options how the union-bound could be avoided.

Fig. 14 demonstrates that Theorem 28 is not trivial: there exist computationally easy settings, it is possible to recognize these settings during runtime, and to guarantee a desired precision. However, the parameter configurations used in the experiments for Fig. 14 are pathological. The reader will agree that a relative precision of 0.1 and a failure chance of 25% can hardly be considered practically relevant values. In that sense, Theorem 27 should rather be taken as an encouraging feasibility study. Section 19.2 discusses possibilities how to obtain more practical versions.

16.4 Related Work

Most closely related is the work by Mnih, Szepesvári, and Audibert (2008) and references therein. They propose an algorithm called *EBStop* that returns an estimate of the mean of a sum of i.i.d. random variables. Similar to Theorem 27, they are able to guarantee a relative precision with high probability. However, Mnih, Szepesvári, and Audibert (2008) assume that the addends are independent and identically distributed, whereas Theorem 27 is more general and assumes only a (non-strict) decrease in expectation (Eq. ($*$)). Their approach is in a sense more sophisticated as they also monitor the empirical variance of the addends (*c.f.* remarks on future directions in Section 19.2). Bardenet, Doucet, and Holmes (2014) propose a related approach to stop Metropolis-Hastings sampling, but the algorithm is not applicable for the problem described in Section 16.1. Zhao, Zhou, Sabharwal, and Ermon (2016) present concentration inequalities for arbitrary stopping times, yet they also assume that the addends are independent and identically distributed.

I assume that the problem described in Section 16.1 has been approached by someone else before. However, I was unable to find literature that makes the same or less restrictive assumptions, as in the case described here. It appears as if the bandit and reinforcement learning community has not yet considered this scenario. A literature search on optimal stopping, empirical stopping, adaptive stopping, sequential analysis and racing, did not yield any further leads. An indicator why this problem may not have been considered yet is that the main tool for the proof of Theorem 27, Theorem 29 by Fan, Grama, and Liu (2012), has been published fairly recently.

The problem of estimating determinants has been studied extensively especially for symmetric and positive definite matrices (Skilling 1989; Dorn and Enßlin 2015; Fitzsimons, Granziol, Cutajar, Osborne, Filippone, and Roberts 2017; Fitzsimons, Cutajar, Osborne, Roberts, and Filippone 2017; Saibaba, Alexanderian, and Ipsen 2017; Boutsidis, Drineas, Kambadur, Kontopoulou, and Zouzias 2017). Yet, none of the aforementioned work considers kernel matrices in particular. In comparison, Theorem 28 offers an “almost-free lunch”. If early stopping is not possible, the algorithm returns the exact solution with negligible overhead. To my knowledge, Theorem 28 is the first distribution-free and deterministic approach to quantify uncertainty over approximate determinant estimation for kernel matrices.

17 Proof of Theorem 27

Theorem 27 from page 77 is restated here for the readers convenience. Recall that all definitions introduced in Section 16.2 are summarized in Table 2 on page 82.

Theorem 27. *Let $(\Omega, \mathcal{F}, P)$ be a probability space and $(\mathcal{F}_j)_{j\in\{1,\ldots,N\}}$ be a filtration. Furthermore, let $(f_j)_{j\in\mathbb{N}} \in [C^-, C^+]$ be a sequence of $\mathcal{F}_j$-measurable random variables that decrease conditionally in expectation,*

$$\mathbb{E}[f_{j+1} \mid \mathcal{F}_j] \leq \mathbb{E}[f_j \mid \mathcal{F}_{j-1}] \qquad (*)$$

for $j \in \{1,\ldots,N-1\}$, where $\mathcal{F}_0 := \{\emptyset, \mathbb{R}\}$, and define

$$D_N := \sum_{j=1}^{N} f_j.$$

For $r, \delta \in (0,1)$, the probability that the relative error of the estimator $\hat{D}_\tau$ defined by Eqs. (47),(C1) and (C2) is larger than r is less than δ.

$$P\left(\text{abs}\left(\frac{D_N - \hat{D}_\tau}{D_N}\right) > r\right) \leq \delta \qquad (48)$$

PROOF OUTLINE The steps to bound the left hand side of Eq. (48) can be split into three parts. Section 17.1 contains the first part, where Eq. (48) is split into two cases. In the first case, D_N is smaller than the probabilistic upper bound $\mathcal{U}_\tau$ such that by the conditions of the stopping time τ, the probability of failure is zero. The second case is the main proof, which is to show that the probability $P(\mathcal{U}_\tau < D_N)$ is small. Part two, in Section 17.2, shows that $P(\mathcal{U}_\tau < D_N)$ is bound from above by two terms. The first term is essentially the probability that D_N is "much larger" than its expected value and we will show that this probability is below $\delta/2$. Section 17.3, the last part, is concerned with the second term which is the probability that the upper bound $\mathcal{U}_\tau$ is less than the expected value of D_N. There we will make use of Eq. $(*)$ to show that this probability is also less than $\delta/2$.

Proof. As a preliminary observation note that

$$D_N = \sum_{j=1}^{N} f_j$$

// *by definition*

$$= NC^- + \sum_{j=1}^{N} l_j$$

Symbol	Definition	Intuition
l_j	$f_j - C^-$	the (positive) random elements
$\hat{\mu}_n$	$\frac{1}{m}\sum_{j=n-m+1}^{n} l_j$	mean estimate
$\mathcal{L}_n$	$NC^- + \sum_{j=1}^{n} l_j$	lower bound on D_N (deterministic)
C	$C^+ - C^-$	upper bound on l_j
$H_N(x)$	$\mathbf{1}_{\{x\leq N\}}\sqrt{\left(\frac{N}{N+x}\right)^{N+x}\left(\frac{N}{N-x}\right)^{N-x}}$	bounding function for failure probability (definition of $H(x, N)$ in Theorem 29, p. 84)
ϵ_μ	${}^{C}\!/\!{}_{m} H_m^{-1}\left(\frac{\delta}{2\lvert S\rvert}\right)$	errror tolerance for the mean
ϵ_n	$(N-n)\epsilon_\mu + CH_N^{-1}({}^{\delta}\!/\!{}_{2})$	error tolerance for the upper bound
$\mathcal{U}_n$	$\mathcal{L}_n + \min((N-n)\hat{\mu}_n + \epsilon_n, (N-n)C)$	upper bound on D_N (probabilistic)
$\hat{D}_n$	${}^{1}\!/\!{}_{2}(\mathcal{L}_n + \mathcal{U}_n)$	estimate for D_N
Eq. (C1)	$\operatorname{sign}(\mathcal{U}_n) = \operatorname{sign}(\mathcal{L}_n) \neq 0$	first condition for τ
Eq. (C2)	$\frac{\mathcal{U}_n - \mathcal{L}_n}{2\min(\operatorname{abs}(\mathcal{U}_n),\operatorname{abs}(\mathcal{L}_n))} \leq r$	second condition for τ

Table 2: compact overview of all the definitions of Part v.

// *using the definition of* l_j

$$= \mathcal{L}_n + \sum_{j=n+1}^{N} l_j \text{ for all } n = 0, ..., N \tag{49}$$

// *using the definition of* $\mathcal{L}_n$

and hence, for all $n = 0, ..., N$, $\mathcal{L}_n$ is a (deterministic) lower bound to D_N, since $l_j \geq 0$ by definition.

17.1 Using the Stopping Conditions

This section is the first part of the proof in which we split Eq. (48) into two cases and resolve the first case. We will make use of Lemma 25 (p. 75).

$$P\left(\mathsf{abs}\left(\frac{D_N - \hat{D}_\tau}{D_N}\right) > r\right)$$

$$= P\left(\frac{\mathsf{abs}\left(D_N - \hat{D}_\tau\right)}{\mathsf{abs}\left(D_N\right)} > r, D_N \leq \mathcal{U}_\tau\right)$$

$$+ P\left(\frac{\mathsf{abs}\left(D_N - \hat{D}_\tau\right)}{\mathsf{abs}\left(D_N\right)} > r, D_N > \mathcal{U}_\tau\right)$$

// *sum rule*

$$\leq P\left(\frac{\mathsf{abs}\left(D_N - \hat{D}_\tau\right)}{\mathsf{abs}\left(D_N\right)} > r, D_N \leq \mathcal{U}_\tau\right) + P\left(D_N > \mathcal{U}_\tau\right) \tag{50}$$

// *upper-bounding joint by marginal*

Consider the first term. Recall that $\hat{D}_\tau \in [\mathcal{L}_\tau, \mathcal{U}_\tau]$.

$$P\left(\frac{\mathsf{abs}\left(D_N - \hat{D}_\tau\right)}{\mathsf{abs}\left(D_N\right)} > r, D_N \leq \mathcal{U}_\tau\right)$$

$$\leq P\left(\frac{\mathsf{abs}\left(D_N - \hat{D}_\tau\right)}{\mathsf{abs}\left(D_N\right)} > r, D_N \leq \mathcal{U}_\tau, \mathsf{sign}(\mathcal{L}_\tau) = \mathsf{sign}(\mathcal{U}_\tau) \neq 0\right)$$

$$+ P\left(\mathsf{sign}(\mathcal{U}_\tau \neq \mathcal{L}_\tau)\right)$$

// *sum rule and upper bounding joint by marginal*

$$= P\left(\frac{\mathsf{abs}\left(D_N - \hat{D}_\tau\right)}{\mathsf{abs}\left(D_N\right)} > r, D_N \leq \mathcal{U}_\tau, \mathsf{sign}(\mathcal{L}_\tau) = \mathsf{sign}(\mathcal{U}_\tau) \neq 0\right)$$

// *using the first stopping condition of* τ*, Eq. (C1)*

$$= P\left(\frac{\mathsf{abs}\left(D_N - \hat{D}_\tau\right)}{\mathsf{abs}\left(D_N\right)} > r, \mathcal{L}_\tau \leq D_N \leq \mathcal{U}_\tau, \mathsf{sign}(\mathcal{L}_\tau) = \mathsf{sign}(\mathcal{U}_\tau) \neq 0\right)$$

// *since* $\mathcal{L}_\tau$ *is a deterministic lower bound to* D_N*, Eq. (49)*

$$\leq P\left(\frac{\mathsf{max}\left(\mathcal{U}_\tau - \hat{D}_\tau, \hat{D}_\tau - \mathcal{L}_\tau\right)}{\mathsf{min}(\mathsf{abs}(\mathcal{L}_\tau), \mathsf{abs}(\mathcal{U}_\tau))} > r, \mathcal{L}_\tau \leq D_N \leq \mathcal{U}_\tau, \mathsf{sign}(\mathcal{L}_\tau) = \mathsf{sign}(\mathcal{U}_\tau) \neq 0\right)$$

// *by Lemma 25*

$$= P\left(\frac{\mathcal{U}_\tau - \mathcal{L}_\tau}{2\min(\mathrm{abs}(\mathcal{L}_\tau), \mathrm{abs}(\mathcal{U}_\tau))} > r, \mathcal{L}_\tau \leq D_N \leq \mathcal{U}_\tau, \mathrm{sign}(\mathcal{L}_\tau) = \mathrm{sign}(\mathcal{U}_\tau) \neq 0\right)$$

// plugging in definition of $\hat{D}_\tau$

$$= 0$$

// by the second condition of τ*, Eq.* (C2)

17.2 D_N is probably close to its Expected Value

In this section we will consider the remaining term from Eq. (50). Again, this term will be split into two cases, where this section resolves the first and the second is taken care of in the next section. The following parts of the proof rely on a recent theorem presented by Fan, Grama, and Liu (2012).

Theorem 29 (Hoeffding's inequality for supermartingales (Fan, Grama, and Liu 2012))**.** *Assume that* $(\xi_j, \mathcal{F}_j)_{j=1,\ldots,N}$ *are supermartingale differences satisfying* $\xi_j \leq 1$*. Then, for any* $x \geq 0$ *and* $v > 0$*,*

$$P\left(\sum_{j=1}^{n} \xi_j \geq x \text{ and } \sum_{j=1}^{n} \mathbb{V}[\xi_j \mid \mathcal{F}_{j-1}] \leq v \text{ for some } n \in [1, N]\right) \leq H_N(x, v),$$

where

$$H_N(x,v) := \mathbf{1}_{\{x \leq N\}} \left\{ \left(\frac{v}{v+x}\right)^{v+x} \left(\frac{N}{N-x}\right)^{N-x} \right\}^{\frac{N}{N+v}}.$$

To apply Theorem 29 define $Z'_j := l_j - \mathbb{E}[l_j \mid \mathcal{F}_{j-1}]$ and $Z_j := Z'_{\tau+j}$. Since $(Z'_j, \mathcal{F}_j)_{j\in\{1,N\}}$ is a martingale difference,

$$\left(Z_{\min(j,N)}, \mathcal{F}_{\min(\tau+j,N)}\right)_{j\in\mathbb{N}_0},$$

is a martingale difference as well (Corollary 48 in Appendix F).[1] The random variables Z_j/C are bounded from below by -1 and from above by 1. Now consider the second, remaining term in Eq. (50) from page 83.

$$P(D_N > \mathcal{U}_\tau)$$

$$= P\left(L_\tau + \sum_{j=\tau+1}^{N} l_j > L_\tau + \min((N-\tau)\hat{\mu}_\tau + \epsilon_\tau, (N-\tau)C)\right)$$

// using Eq. (49) *and definition of* $\mathcal{U}_\tau$

$$= P\left(\sum_{j=\tau+1}^{N} l_j > \min((N-\tau)\hat{\mu}_\tau + \epsilon_\tau, (N-\tau)C)\right)$$

// simplifying

$$= P\left(\sum_{j=\tau+1}^{N} l_j > (N-\tau)\hat{\mu}_\tau + \epsilon_\tau \text{ or } \sum_{j=\tau+1}^{N} l_j > (N-\tau)C\right)$$

[1] This follows from a result that is sometimes referred to as Doob's Optional Sampling Theorem (Theorem 46 in Appendix F).

$$
\begin{aligned}
&\textit{// exchanging } \min \textit{ for logical or}\\
=&P\left(\sum_{j=\tau+1}^{N} l_j > (N-\tau)\hat{\mu}_\tau + \epsilon_\tau\right)\\
&\textit{// since } l_j \le C\\
=&P\left(\sum_{j=1}^{N-\tau} \left[Z_j + \mathbb{E}[l_{\tau+j} \mid \mathcal{F}_{\tau+j-1}]\right] > (N-\tau)\hat{\mu}_\tau + \epsilon_\tau\right)\\
&\textit{// definition of } Z_j\\
\le&P\left(\sum_{j=1}^{N-\tau} Z_j + \sum_{j=\tau+1}^{N} \mathbb{E}[l_j \mid \mathcal{F}_{j-1}] > (N-\tau)\hat{\mu}_\tau + \epsilon_\tau,\right. \qquad (51)\\
&\qquad\left.\sum_{j=\tau+1}^{N} \mathbb{E}[l_j \mid \mathcal{F}_{j-1}] \le (N-\tau)(\hat{\mu}_\tau + \epsilon_\mu)\right)\\
&\quad + P\left(\sum_{j=\tau+1}^{N} \mathbb{E}[l_j \mid \mathcal{F}_{j-1}] > (N-\tau)(\hat{\mu}_\tau + \epsilon_\mu)\right)\\
&\textit{// sum rule and upper-bounding joint by marginal}
\end{aligned}
$$

Consider the first term in Eq. (51). We want to apply Theorem 29.

$$
\begin{aligned}
&P\left(\sum_{j=1}^{N-\tau} Z_j + \sum_{j=\tau+1}^{N} \mathbb{E}[l_j \mid \mathcal{F}_{j-1}] > (N-\tau)\hat{\mu}_\tau + \epsilon_\tau,\right.\\
&\qquad\left.\sum_{j=\tau+1}^{N} \mathbb{E}[l_j \mid \mathcal{F}_{j-1}] \le (N-\tau)(\hat{\mu}_\tau + \epsilon_\mu)\right)\\
&\le P\left(\sum_{j=1}^{N-\tau} Z_j + (N-\tau)(\hat{\mu}_\tau + \epsilon_\mu) > (N-\tau)\hat{\mu}_\tau + \epsilon_\tau\right)\\
&\textit{// combining the two events}\\
&= P\left(\sum_{j=1}^{N-\tau} Z_j > \epsilon_\tau - (N-\tau)\epsilon_\mu\right)\\
&\textit{// simplifying}\\
&= P\left(\sum_{j=1}^{N-\tau} \frac{Z_j}{C} > H_N^{-1}(\delta/2)\right)\\
&\textit{// definition of } \epsilon_\tau \textit{ and dividing by } C\\
&\le P\left(\sum_{j=1}^{n} \frac{Z_j}{C} > H_N^{-1}(\delta/2) \text{ for some } n \in \{1, \ldots, N\}\right) \qquad (52)\\
&\textit{// enlargening the event}\\
&= P\left(\sum_{j=1}^{n} \frac{Z_j}{C} > H_N^{-1}(\delta/2), \sum_{j=1}^{n} \mathbb{V}[Z_j/C \mid \mathcal{F}_{j-1}] \le N \text{ for some } n \in \{1, \ldots, N\}\right)\\
&\textit{// by Popoviciu's inequality (Theorem 44 on p. 129): } \mathbb{V}[Z_j/C \mid \mathcal{F}_{j-1}] \le 1.\\
&\le H(H_N^{-1}(\delta/2), N) \qquad (53)\\
&\textit{// by Theorem 29, where } H \textit{ is defined in that theorem}\\
&= H_N(H_N^{-1}(\delta/2)) \le \delta/2.
\end{aligned}
$$

// *definition of* H_N

17.3 $\mathcal{U}_\tau$ is probably large enough

This part takes care of the second term in Eq. (51). In essence, we will show that $\mathcal{U}_\tau$ is large enough with high probability. Now we will make use of the assumption that the l_j decrease in expectation: Eq. ($*$). We will again apply Theorem 29.

$$P\left(\sum_{j=\tau+1}^{N}\mathbb{E}[l_j \mid \mathcal{F}_{j-1}] > (N-\tau)(\hat{\mu}_\tau+\epsilon_\mu)\right)$$

$$\leq P\left(\mathbb{E}[l_{\tau+1} \mid \mathcal{F}_\tau] - \hat{\mu}_\tau > \epsilon_\mu\right)$$

// *using Eq.* ($*$)

$$= P\left(\mathbb{E}[l_{\tau+1} \mid \mathcal{F}_\tau] - \frac{1}{m}\sum_{j=\tau-m+1}^{\tau} l_j > \epsilon_\mu\right)$$

// *definition of* $\hat{\mu}_\tau$

$$= P\left(\sum_{j=\tau-m+1}^{\tau}(\mathbb{E}[l_{\tau+1} \mid \mathcal{F}_\tau] - l_j) > m\epsilon_\mu\right)$$

// *multiplication by* m *and moving the conditional expectation into the sum*

$$\leq P\left(\sum_{j=\tau-m+1}^{\tau}(\mathbb{E}[l_j \mid \mathcal{F}_{j-1}] - l_j) > m\epsilon_\mu\right)$$

// *using again Eq.* ($*$)

$$= \sum_{s\in S} P\left(\sum_{j=\tau-m+1}^{\tau}(\mathbb{E}[l_j \mid \mathcal{F}_{j-1}] - l_j) > m\epsilon_\mu, \tau = s\right)$$

// *sum rule*

$$= \sum_{s\in S} P\left(\sum_{j=s-m+1}^{s}(\mathbb{E}[l_j \mid \mathcal{F}_{j-1}] - l_j) > CH_m^{-1}\left(\frac{\delta}{2|S|}\right)\right) \tag{54}$$

// *definition of* ϵ_μ

$$= \sum_{s\in S} P\left(\sum_{j=s-m+1}^{s} -\frac{Z'_j}{C} > H_m^{-1}\left(\frac{\delta}{2|S|}\right)\right)$$

// *definition of* Z'_j

$$\leq \sum_{s\in S} P\left(\sum_{j=1}^{m'} -\frac{Z'_{s-m+j}}{C} > H_m^{-1}\left(\frac{\delta}{2|S|}\right) \text{ for some } m' \in \{1,\dots,m\}\right)$$

// *enlargening the event*

Changing the sign does not change the martingale difference property and hence, $(-Z'_j, \mathcal{F}_j)_{j\in\{s-m+1,\dots,s\}}$ is a martingale difference for all $s \in S$. We can apply the same argument as in Eq. (53).

$$\sum_{s\in S} P\left(\sum_{j=1}^{m'} -\frac{Z'_{s-m+j}}{C} > H_m^{-1}\left(\frac{\delta}{2|S|}\right) \text{ for some } m' \in \{1,\dots,m\}\right)$$

$$\leq \sum_{s \in S} H\left(H_m^{-1}\left(\frac{\delta}{2|S|}\right), m\right)$$

// *using the same argument as in Eq.* (53)

$$\leq \sum_{s \in S} \frac{\delta}{2|S|} = \frac{\delta}{2}$$

To see that this completes the proof, we recall below the proof outline from page 81. In Section 17.1 we showed that in order to proof

$$P\left(\mathsf{abs}\left(\frac{D_N - \hat{D}_\tau}{D_N}\right) > r\right) \leq \delta,$$

using the definition of the stopping time τ, it is sufficient to show $P(D_N > \mathcal{U}_\tau) \leq \delta$. In Section 17.2, more specifically in Eq. (51), the term $P(D_N > \mathcal{U}_\tau)$ was split into a sum of two terms. The remainder of Section 17.2 showed that the first term is less than $\delta/2$. This section showed above that the second term is less than $\delta/2$. Hence, the proof is complete.

□

18 Application to Kernel Matrix Determinant Estimation

This chapter shows that the stopping time proposed in Theorem 27 can be used to terminate the computation of kernel matrix determinants. We will make the mild assumption that $\boldsymbol{x}_1, ..., \boldsymbol{x}_n, ..., \boldsymbol{x}_N \in \mathbb{X}$ are independent and identically distributed. This assumption is not always fulfilled, *e.g.* when the inputs are sorted. By shuffling the dataset, the assumption can be established.

In the following, we will consider the problem of evaluating $\ln |\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}_N|$ for $\sigma^2 > 0$. This is a common expression required for example in Gaussian process regression where σ^2 represents the observational noise (see *e.g.* Rasmussen and Williams (2006, p. 13ff.)). In case $\sigma^2 = 0$ one needs another, strictly positive, lower bound on the smallest eigenvalue of $\boldsymbol{K}_N$.

Theorem 28. *Assume $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{X}$ are independent and identically distributed. Denote with P the law of the $\boldsymbol{x}_1, .., \boldsymbol{x}_N$ and with $\boldsymbol{C}$ the Cholesky decomposition of $\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}$, where $\sigma^2 > 0$. Define the probability space $(\mathbb{X}, \sigma(\boldsymbol{x}_1, ..., \boldsymbol{x}_N), P)$ and the canonical filtration $\mathcal{F}_j := \sigma(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j)$ for $j = 1, ..., N$. Further, define*

$$\begin{aligned} f_j &:= 2 \ln |\boldsymbol{C}_{jj}|, \text{ with} \\ C^- &:= \ln \sigma^2 \text{ and} \\ C^+ &:= \max_{j=1,...,N} \ln(k(\boldsymbol{x}_j, \boldsymbol{x}_j) + \sigma^2). \end{aligned}$$

Then, using the definitions of Theorem 27,

$$P\left(\frac{\operatorname{abs}(|\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}| - \hat{D}_\tau)}{\operatorname{abs}(|\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}|)} > r\right) \leq \delta.$$

Proof. The proof follows from Theorem 27. To apply the theorem we need to show that for all $j = 1, ...N$, the $\boldsymbol{C}_{jj}$ are functions of $\boldsymbol{x}_1, ..., \boldsymbol{x}_j$ (Lemma 30, p. 89), that $f_j \in [C^-, C^+]$ (Lemma 31, p. 90), and that $\mathbb{E}[f_{j+1} \mid \mathcal{F}_j] \leq \mathbb{E}[f_j \mid \mathcal{F}_{j-1}]$ (Lemma 32, p. 91). □

To proof the three lemmata referenced above, define

$$\begin{aligned} \boldsymbol{k}_j(\boldsymbol{x}) &:= [k(\boldsymbol{x}, \boldsymbol{x}_1), ..., k(\boldsymbol{x}, \boldsymbol{x}_j)]^\intercal \in \mathbb{R}^j \text{ and} \\ \boldsymbol{k}_{j+1} &:= \boldsymbol{k}_j(\boldsymbol{x}_j) \in \mathbb{R}^j. \end{aligned}$$

Lemma 30. *Denote with $\boldsymbol{C}_N$ the Cholesky decomposition of $\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}_N$. The n-th diagonal element of $\boldsymbol{C}_N$, squared, is the posterior variance of a GP conditioned on the previous data-points:*

$$[\boldsymbol{C}_N]^2_{nn} = k(\boldsymbol{x}_n, \boldsymbol{x}_n) + \sigma^2 - \boldsymbol{k}_n^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_n.$$

Proof. With abuse of notation, define $\boldsymbol{C}_1 := \sqrt{k(\boldsymbol{x}_1, \boldsymbol{x}_1)}$ and

$$\boldsymbol{C}_N := \begin{bmatrix} \boldsymbol{C}_{N-1} & \boldsymbol{0} \\ \boldsymbol{k}_N^\intercal \boldsymbol{C}_{N-1}^{-\intercal} & \sqrt{k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2 - \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N} \end{bmatrix}.$$

We will show that the lower triangular matrix $\boldsymbol{C}_N$ satisfies $\boldsymbol{C}_N \boldsymbol{C}_N^\intercal = \boldsymbol{K}_N + \sigma^2 \boldsymbol{I}_N$. Since the Cholesky decomposition is unique (Golub and Van Loan 2013, Theorem 4.2.7), $\boldsymbol{C}_N$ must be the Cholesky decomposition of $\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}_N$. Furthermore, by definition of $\boldsymbol{C}_N$, $[\boldsymbol{C}_N]^2_{NN} = k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2 - \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N$. The statement then follows by induction.

To remain within the text margins, define

$$x := \boldsymbol{k}_N^\intercal \boldsymbol{C}_{N-1}^{-\intercal} \boldsymbol{C}_{N-1}^{-1} \boldsymbol{k}_N + k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2 - \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N.$$

We want to show that $\boldsymbol{C}_N \boldsymbol{C}_N^\intercal = \boldsymbol{K}_N + \sigma^2 \boldsymbol{I}_N$.

$$\begin{aligned} \boldsymbol{C}_N \boldsymbol{C}_N^\intercal &= \begin{bmatrix} \boldsymbol{C}_{N-1} & \boldsymbol{0} \\ \boldsymbol{k}_N^\intercal \boldsymbol{C}_{N-1}^{-\intercal} & \sqrt{k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2 - \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N} \end{bmatrix} \\ &\quad \cdot \begin{bmatrix} \boldsymbol{C}_{N-1}^\intercal & \boldsymbol{C}_{N-1}^{-1} \boldsymbol{k}_N \\ \boldsymbol{0}^\intercal & \sqrt{k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2 - \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{C}_{N-1} \boldsymbol{C}_{N-1}^\intercal & \boldsymbol{C}_{N-1} \boldsymbol{C}_{N-1}^{-1} \boldsymbol{k}_N \\ \boldsymbol{k}_N^\intercal \boldsymbol{C}_{N-1}^{-\intercal} \boldsymbol{C}_{N-1}^\intercal & x \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{K}_{N-1} + \sigma^2 \boldsymbol{I}_{N-1} & \boldsymbol{k}_N \\ \boldsymbol{k}_N^\intercal & x \end{bmatrix} \end{aligned}$$

Also x can be simplified further.

$$\begin{aligned} x &= \boldsymbol{k}_N^\intercal \boldsymbol{C}_{N-1}^{-\intercal} \boldsymbol{C}_{N-1}^{-1} \boldsymbol{k}_N + k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2 - \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N \\ &= \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N + k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2 - \boldsymbol{k}_N^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_N \\ &= k(\boldsymbol{x}_N, \boldsymbol{x}_N) + \sigma^2. \end{aligned}$$

□

Lemma 31. *The $f_j := 2 \ln \boldsymbol{C}_{jj}$ are bounded between $C^- := \ln \sigma^2$ and $C^+ := \max_{j=1,\ldots,N} \ln \left(k(\boldsymbol{x}_j, \boldsymbol{x}_j) + \sigma^2\right)$.*

Proof. By Lemma 30, $\boldsymbol{C}_{jj}^2 = k(\boldsymbol{x}_j, \boldsymbol{x}_j) + \sigma^2 - \boldsymbol{k}_j^\intercal (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I}_{j-1})^{-1} \boldsymbol{k}_j$ which is the posterior variance of a Gaussian process. The third term in the posterior variance is always positive since $(\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I}_{j-1})^{-1}$ is an s.p.d. matrix. Hence, $k(\boldsymbol{x}_j, \boldsymbol{x}_j) + \sigma^2$ is an upper bound to $\boldsymbol{C}_{jj}$. On the other hand, since k is a kernel, $k(\boldsymbol{x}_j, \boldsymbol{x}_j) - \boldsymbol{k}_j^\intercal (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I}_{j-1})^{-1} \boldsymbol{k}_j$ can not be negative and σ^2 is a therefore a lower bound to $\boldsymbol{C}_{jj}^2$. Since both values are positive and the logarithm is an increasing function on the positive real axis, the proof is complete. □

Lemma 32. *The $f_j := 2 \ln \boldsymbol{C}_{jj}$ decrease in expectation:*

$$\mathbb{E}[f_{j+1} \mid \sigma(\boldsymbol{x}_1, ..., \boldsymbol{x}_j)] \leq \mathbb{E}[f_j \mid \sigma(\boldsymbol{x}_1, ..., \boldsymbol{x}_{j-1})].$$

Proof. By Lemma 30, $\boldsymbol{C}_{jj}^2 = k(\boldsymbol{x}_j, \boldsymbol{x}_j) + \sigma^2 - \boldsymbol{k}_n^\intercal (\boldsymbol{K}_{n-1} + \sigma^2 \boldsymbol{I}_{n-1})^{-1} \boldsymbol{k}_n$ which is the posterior variance of a Gaussian process. Since a variance is always positive and the logarithm is an increasing function on the positive real axis it is sufficient to show that the $\boldsymbol{C}_{jj}^2$ decrease in expectation. Define the two shorthands $p_j := \boldsymbol{C}_{jj}^2$ and $q_j(\boldsymbol{x}) := \boldsymbol{k}_j(\boldsymbol{x})^\intercal (\boldsymbol{K}_j + \sigma^2 \boldsymbol{I}_j)^{-1} \boldsymbol{k}_j(\boldsymbol{x})$. We will show below, in Eq. (56), that $q_j(\boldsymbol{x}) = q_{j-1}(\boldsymbol{x}) + r_{j-1}(\boldsymbol{x})$ where $r_{j-1}(\boldsymbol{x}) \geq 0$.

$$\mathbb{E}[p_{j+1} \mid \sigma(\boldsymbol{x}_1, ..., \boldsymbol{x}_j)]$$

$$= \int k(\boldsymbol{x}, \boldsymbol{x}) + \sigma^2 - \boldsymbol{k}_j(\boldsymbol{x})^\intercal (\boldsymbol{K}_j + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_j(\boldsymbol{x}) \; P(\,\mathrm{d}\boldsymbol{x} \mid \boldsymbol{x}_1, ..., \boldsymbol{x}_j)$$

// property of conditional expectation

$$= \int k(\boldsymbol{x}, \boldsymbol{x}) + \sigma^2 - q_j(\boldsymbol{x}) \; P(\,\mathrm{d}\boldsymbol{x} \mid \boldsymbol{x}_1, ..., \boldsymbol{x}_j)$$

// definition of $q_j(\boldsymbol{x})$

$$= \int k(\boldsymbol{x}, \boldsymbol{x}) + \sigma^2 - q_{j-1}(\boldsymbol{x}) - r_{j-1}(\boldsymbol{x}) \; P(\,\mathrm{d}\boldsymbol{x} \mid \boldsymbol{x}_1, ..., \boldsymbol{x}_j)$$

// using Eq. (55)

$$= \int k(\boldsymbol{x}, \boldsymbol{x}) + \sigma^2 - q_{j-1}(\boldsymbol{x}) \; P(\,\mathrm{d}\boldsymbol{x} \mid \boldsymbol{x}_1, ..., \boldsymbol{x}_j) - \int r_{j-1}(\boldsymbol{x}) \; P(\,\mathrm{d}\boldsymbol{x} \mid \boldsymbol{x}_1, ..., \boldsymbol{x}_j)$$

// splitting the integral

$$\leq \int k(\boldsymbol{x}, \boldsymbol{x}) + \sigma^2 - q_{j-1}(\boldsymbol{x}) \; P(\,\mathrm{d}\boldsymbol{x} \mid \boldsymbol{x}_1, ..., \boldsymbol{x}_j)$$

// using Eq. (56)

$$= \int k(\boldsymbol{x}, \boldsymbol{x}) + \sigma^2 - q_{j-1}(\boldsymbol{x}) \; P(\,\mathrm{d}\boldsymbol{x} \mid \boldsymbol{x}_1, ..., \boldsymbol{x}_{j-1})$$

// with Fubini's theorem

$$= \mathbb{E}[p_j \mid \sigma(\boldsymbol{x}_1, ..., \boldsymbol{x}_{j-1})]$$

// property of conditional expectation

It remains to show $q_j(\boldsymbol{x}) = q_{j-1}(\boldsymbol{x}) + r_{j-1}(\boldsymbol{x})$ where $r_{j-1}(\boldsymbol{x}) \geq 0$. Define $\boldsymbol{v}_{\boldsymbol{x}} := (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{j-1}(\boldsymbol{x})$. First note, that using block-matrix inversion we can write

$$(\boldsymbol{K}_j + \sigma^2 \boldsymbol{I})^{-1} = \begin{bmatrix} (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I})^{-1} + \boldsymbol{v}_x p_j^{-1} \boldsymbol{v}_x^\intercal & -\boldsymbol{v}_x p_j^{-1} \\ -\boldsymbol{v}_x^\intercal p_j^{-1} & p_j^{-1} \end{bmatrix}.$$

Using above observation, we can transform $q_j(\boldsymbol{x})$.

$$\begin{aligned} q_j(\boldsymbol{x}) &= \begin{bmatrix} \boldsymbol{k}_{j-1}(\boldsymbol{x})^\intercal & k(\boldsymbol{x}, \boldsymbol{x}_j) \end{bmatrix} \\ &\quad \cdot \begin{bmatrix} (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I})^{-1} + \boldsymbol{v}_x p_j^{-1} \boldsymbol{v}_x^\intercal & -\boldsymbol{v}_x p_j^{-1} \\ -\boldsymbol{v}_x^\intercal p_j^{-1} & p_j^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{k}_{j-1}(\boldsymbol{x}) \\ k(\boldsymbol{x}, \boldsymbol{x}_j) \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{k}_{j-1}(\boldsymbol{x})^\intercal & k(\boldsymbol{x}, \boldsymbol{x}_j) \end{bmatrix} \\ &\quad \cdot \begin{bmatrix} (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}_{j-1}(\boldsymbol{x}) + \boldsymbol{v}_x p_j^{-1} \boldsymbol{v}_x^\intercal \boldsymbol{k}_{j-1}(\boldsymbol{x}) - \boldsymbol{v}_x p_j^{-1} k(\boldsymbol{x}, \boldsymbol{x}_j) \\ -\boldsymbol{v}_x^\intercal \boldsymbol{k}_{j-1}(\boldsymbol{x}) p_j^{-1} + p_j^{-1} k(\boldsymbol{x}, \boldsymbol{x}_j) \end{bmatrix} \\ &= \boldsymbol{k}_{j-1}(\boldsymbol{x})^\intercal (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I}) \boldsymbol{k}_{j-1}(\boldsymbol{x}) + p_j^{-1} (\boldsymbol{v}_x^\intercal \boldsymbol{k}_{j-1}(\boldsymbol{x}))^2 \\ &\quad - 2 \boldsymbol{v}_x^\intercal \boldsymbol{k}_{j-1}(\boldsymbol{x}) p_j^{-1} k(\boldsymbol{x}, \boldsymbol{x}_j) + p_j^{-1} k(\boldsymbol{x}, \boldsymbol{x}_j)^2 \\ &= \boldsymbol{k}_{j-1}(\boldsymbol{x})^\intercal (\boldsymbol{K}_{j-1} + \sigma^2 \boldsymbol{I}) \boldsymbol{k}_{j-1}(\boldsymbol{x}) + p_j^{-1} (k(\boldsymbol{x}, \boldsymbol{x}_j)^2 - \boldsymbol{v}_x^\intercal \boldsymbol{k}_{j-1}(\boldsymbol{x}))^2 \\ &= q_{j-1}(\boldsymbol{x}) + p_j^{-1} (k(\boldsymbol{x}, \boldsymbol{x}_j)^2 - \boldsymbol{v}_x^\intercal \boldsymbol{k}_{j-1}(\boldsymbol{x}))^2 \\ &= q_{j-1}(\boldsymbol{x}) + p_j^{-1} (k(\boldsymbol{x}, \boldsymbol{x}_j)^2 - q_{j-1}(\boldsymbol{x}))^2 \end{aligned}$$

This shows that

$$q_j(\boldsymbol{x}) = q_{j-1}(\boldsymbol{x}) + r_{j-1}(\boldsymbol{x}) \text{ , where} \tag{55}$$

$$r_{j-1}(\boldsymbol{x}) := p_j^{-1} (k(\boldsymbol{x}, \boldsymbol{x}_j)^2 - q_{j-1}(\boldsymbol{x}))^2 \geq 0. \tag{56}$$

□

19 Discussion

19.1 Summary

This part presented a stopping strategy for the Cholesky decomposition that provides probably approximately correct kernel matrix determinants, under the assumptions that the dataset inputs are independent and identically distributed. Fig. 14 demonstrated that Theorem 27, though not trivial, needs further improvement. Nevertheless, Fig. 13 demonstrated that the practical realization of this stopping strategy is feasible and promising. And, though, behind expectations, using the stopping strategy is an “almost free lunch.”

19.2 Future Directions

19.2.1 Improving the Proof

The need to select a subset $S \subseteq \{m, ..., N-1\}$ of possible stopping points, is due to the union bound in Eq. (54). The proof of Theorem 29 by Fan, Grama, and Liu (2012) avoids a union bound with a technique they call the *conjugate probability measure*. It should be possible to apply the same technique.

The union bound may be unnecessary, altogether. Theorem 29 is a generalization of Hoeffding’s inequality (Fan, Grama, and Liu 2012). Schölkopf and Smola (2002, p. 181f.) proof with Hoeffding’s inequality that a subsample from a larger sum of i.i.d. random variables can be used to estimate the latter. Their proof does not require to measure separately how much estimate and remaining terms deviate from their expectation (*c.f.* Eq. (51)). Yet, in the case described here the summands are neither identically distributed nor independent, and there comes additional difficulty from the stopping condition.

Boucheron, Lugosi, and Massart (2013) describe a number of concentration inequalities for functions of random variables that are called self-bounding. It turns out, that the log-determinant of a kernel matrix falls into this category (Lemma 33, below). Self-bounding functions have the property that their variance is bounded by their expected value. Theorem 29 provides stronger guarantees when providing better bounds on the variance. Originally, my plan was to monitor the empirical variance as well, to obtain a Bernstein-like bound. For the examples from Figure 11 in Section 15.1 the empirical variance of the $^{l_j}/_{C}$ is lower than the bound by Popoviciu's inequality (*c.f.* Eq. (52)). One can expect a stopping rule that takes the variance into account to be more efficient than one which does not. Furthermore, the concentration inequalities for self-bounding functions often allow to reason about the probability of the function falling *below* its expectation. In the proof, the lower-bound for the determinant is deterministic. An estimated, probabilistic lower-bound could be less conservative.

Another direction to obtain sharper bounds is to stipulate further assumptions in Theorem 27. Properties of the kernel, such as Lipschitzness and differentiability could be bequeathed to the f_j.

Lemma 33. *Denote with* $\boldsymbol{C}$ *the Cholesky decomposition of* $\boldsymbol{K}_N$*. Assume that there exist constants* $C^-, C^+ \in \mathbb{R}$*, s.t.* $2 \ln \boldsymbol{C}_{jj} \in [C^-, C^+]$ *for all* $j \in \{1, \ldots, N\}$ *(cf. Lemma 31). Then the function*

$$f(\boldsymbol{x}_1, ..., \boldsymbol{x}_N) := \frac{\ln |\boldsymbol{K}_N| - NC^-}{C^+ - C^-}$$

is self-bounding (definition in proof).

Proof. For the definition of self-bounding (Boucheron, Lugosi, and Massart 2013, p. 60) we have to show that there exist functions $f_i(\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_N)$ s.t. for all $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$:

$$0 \leq f(\boldsymbol{x}_1, ..., \boldsymbol{x}_N) - f_i(\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_N) \leq 1, \tag{57}$$

and

$$\sum_{i=1}^{N} (f(\boldsymbol{x}_1, ..., \boldsymbol{x}_N) - f_i(\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_N)) \leq f(\boldsymbol{x}_1, ..., \boldsymbol{x}_N). \tag{58}$$

Choose

$$f_i(\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_N) := \frac{\ln |\boldsymbol{K}_{N\setminus i}| - (N-1)C^-}{C^+ - C^-}$$

where $\boldsymbol{K}_{N\setminus i}$ is obtained from $\boldsymbol{K}_N$ by deleting the i-th row and column. Note that, the order of the inputs does not matter. Swapping two rows or columns changes the sign of the determinant. Swapping the order of the datapoints i and j corresponds to a swap of the corresponding columns and rows, and the sign change cancels. Denote with $\boldsymbol{K}_N^i$ the resulting kernel matrix when moving the i-th datapoint to the end of the dataset, and denote with $\boldsymbol{C}^i$ the Cholesky decomposition. This allows to write rewrite Eq. (57) as follows:

$$\begin{aligned} & f(\boldsymbol{x}_1, ..., \boldsymbol{x}_N) - f_i(\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_N) \\ & = \frac{\ln |\boldsymbol{K}_N| - \ln |\boldsymbol{K}_{N\setminus i}| - C^-}{C^+ - C^-} \\ & \text{// } \textit{definition of } f \textit{ and } f_i \end{aligned}$$

$$= \frac{\ln |\boldsymbol{K}_N^i| - \ln |\boldsymbol{K}_{N-1}^i| - C^-}{C^+ - C^-}$$

// using the determinant's invariance

$$= \frac{2 \ln \boldsymbol{C}_{NN}^i - C^-}{C^+ - C^-}$$

// by Lemma 49 (p. 130)

By assumption on $\boldsymbol{C}$, which carries over to $\boldsymbol{C}^i$, the last expression is bounded from below by 0 and from above by 1. This shows Eq. (57). Extending above argument, we obtain for Eq. (58):

$$\sum_{i=1}^{N} (f(\boldsymbol{x}_1, ..., \boldsymbol{x}_N) - f_i(\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_N))$$

$$= \frac{1}{C^+ - C^-} \sum_{i=1}^{N} (2 \ln \boldsymbol{C}_{NN}^i - C^-)$$

$$\leq \frac{1}{C^+ - C^-} \sum_{i=1}^{N} (2 \ln \boldsymbol{C}_{ii} - C^-)$$

// using Lemma 30, see below

$$= \frac{\ln |\boldsymbol{K}_N| - NC^-}{C^+ - C^-}$$

// by Lemma 49 (p. 130)

$$= f(\boldsymbol{x}_1, ..., \boldsymbol{x}_N)$$

// definition of f

Recall that Lemma 30 relates the diagonal elements of a Cholesky decomposition, to the posterior variance of a Gaussian process: $(\boldsymbol{C}_{NN}^i)^2$ is the posterior variance in $\boldsymbol{x}_i$ given $\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, ..., \boldsymbol{x}_N$, whereas $\boldsymbol{C}_{ii}^2$ is the posterior variance in $\boldsymbol{x}_i$ given only $\boldsymbol{x}_1, ..., \boldsymbol{x}_{i-1}$. The posterior variance of a GP can never increase with more datapoints and therefore $\boldsymbol{C}_{NN}^i \leq \boldsymbol{C}_{ii}$. □

19.2.2 Beyond Determinant Estimation

Often (e.g. for GP regression) it is necessary to evaluate a quadratic form $\boldsymbol{y}^\top (\boldsymbol{K}_N + \sigma^2 \boldsymbol{I}) \boldsymbol{y}$ and the determinant $|\boldsymbol{K}_N|$ at the same time. The analysis for the quadratic form is similar and it would be possible to compute both terms in parallel, stopping prematurely if one or both can be sufficiently approximated. In a conversation, Carl Rasmussen pointed out that a combination of above ideas with variational approximation methods for GPs would be interesting. For example to decide the number of necessary inducing inputs.

Part VI

Epilogue

> Ich klage mich an, bekenne mich ohne Einschränkung schuldig, und vielleicht senken sich jetzt die schon zu verzweifeltem Ringen erhobenen Hände, glätten sich die gerunzelten Stirnen wieder und wischt sich der eine oder andere den Schaum aus den Mundwinkeln. Ich verspreche hiermit feierlich, daß ich am Schluß dieses Erzählwerks ein umfassendes Geständnis ablegen, eine fix und fertige Moral liefern werde, auch eine Interpretation, die allen Interpreten vom Obertertianer bis zum Meisterinterpreten im Oberseminar Seufzen und Nachdenken ersparen wird. Sie wird so abgefaßt sein, daß auch der einfache, der unbefangene Leser sie “mit nach Hause nehmen kann”, weit weniger kompliziert als die Anleitung zur Ausfüllung des Antrags auf Lohnsteuerjahresausgleich. Geduld, Geduld, wir sind noch nicht am Ende.

—Heinrich Böll, *Entfernung von der Truppe*

20 Conclusions

This thesis contains three main parts, each concerned with a different aspect of probabilistic numerics. Part iii examined the connection between probabilistic numerical methods and classic linear solvers. Part iv asked the question how probabilistic numerics can be used to develop novel methods. Part v showed how structure induced by probability can be exploited in classic numerical algorithms. This chapter summarizes the discussions and future directions that have been presented in Chapters 9, 14 and 19 for each part, respectively.

20.1 Discussion

Part iii showed that solution-based inference (SBI) proposed by Cockayne, Oates, Ipsen, and Girolami (2018) always constitutes a projection method (Proposition 6) and that for a given projection method there exist different SBI interpretations. Furthermore, SBI was shown to be subsumed in the matrix-based inference perspective, motivating a classification of probabilistic perspectives into left-multiplied and right-multiplied information. Hence, when reasoning about linear solvers, the unification allows to switch between perspectives as appropriate or convenient. The probabilistic perspective on preconditioning, though not unique, is the first and justifiably intuitive: when using left-multiplied information, right preconditioning corresponds to a change in prior, whereas left preconditioning corresponds to a change in information. When using right-multiplied information, the converse holds. The probabilistic interpretations of GMRES and FOM are currently a rather theoretical contribution as posterior variances are not available in practice.

Equipped with the insights from Part iii, Part iv presented a probabilistic numerical method to solve kernel least-squares problems. In essence, the approach is based on a finite-rank inducing Gaussian process prior over kernel functions and using conjugate gradients matrix-vector products with the kernel matrix to collect observations. The *kernel machine conjugate gradients* (KMCG) approximation consistently outperforms plain conjugate gradients in numerical experiments. This improvement does not change the fact, that standard dense kernel least-squares problems are often more efficiently solved by inducing point methods. Nevertheless, one contribution is a principled procedure to use linear solvers in the context of kernel least-squares problems.

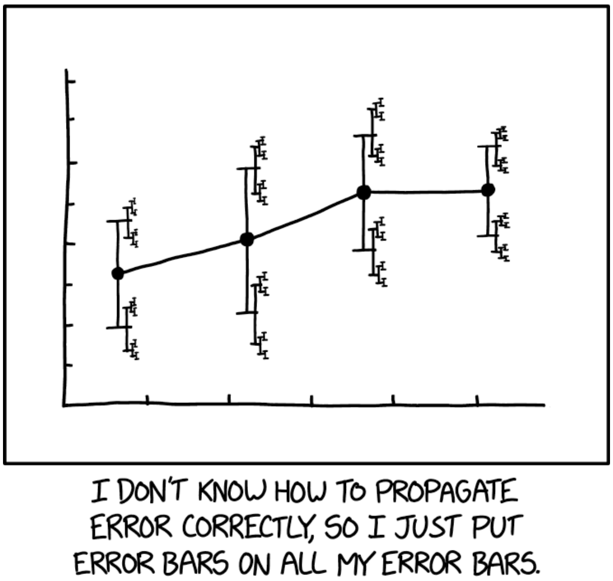
Part v presented a new approach to the approximation of kernel matrix determinants. More generally, Part v posed the problem of approximating the sum of a sequence of N elements that decrease in expectation from a subsequence of eligible length, s.t. a certain precision is reached with a desired probability. Section 16.2 presented a non-trivial solution and the proof of Theorem 27 showed that the combination of proposed stopping strategy and estimator provide a prediction that has indeed the relative precision with the desired probability. Theorem 28 showed that Theorem 27 can be used to terminate the Cholesky decomposition to approximate log-determinants of kernel matrices. When the stopping condition is not triggered, the implementation delivers the exact solution with negligible overhead. In that sense, the stopping strategy is an "almost free lunch." Arguably, the stopping conditions are more conservative than necessary, s.t. the scenarios in which a termination could occur, are rare in practice. Nevertheless, this is the first contribution regarding a distribution-free and deterministic approach to approximate determinants of kernel matrices. To the best of my knowledge, there is no literature that focuses on the approximation of *specifically* kernel-matrix determinants—usually the more general case of symmetric and positive definite matrices is considered.

20.2 Future Work

This thesis left the question how to interpret and evaluate posterior uncertainty provided by probabilistic linear solvers unanswered. The χ^2 statistic proposed by Cockayne, Oates, Ipsen, and Girolami (2018) is a first step but is restricted to Gaussian linear solvers and not applicable for the probabilistic interpretation of Jacobi-iteration from Section 9.2.

A hope associated with KMCG from Part iv was that the probabilistic approach would allow to reason about the approximation error. Propagating probability measures through inverse operations remains an open challenge. First, this requires defining rigorously a ground-truth that can be aimed for. Then the question remains how to approximate the inversion operation. The Cayley–Hamilton theorem might provide an answer.

The work presented in Part v can be extended in different directions. Besides improving the bound in Theorem 27 with other mathematical tools, another option is to change the assumptions. Often Lipschitzness and differentiability of the kernel are known properties and using these could give less conservative bounds. The analysis could be extended to other operations associated with the Cholesky decomposition, *e.g.* estimation of the quadratic form $\boldsymbol{y}^\intercal \boldsymbol{K}^{-1} \boldsymbol{y}$. In a more distant future, the stopping strategy could be a starting point for a modified Cholesky decomposition that constructs a low-rank approximations to $\boldsymbol{K}_N$ with probabilistic guarantees. When treating inducing input variational inference methods (*c.f.* Titsias (2009)) as such low-rank approximations, the considerations in Part v might be useful to decide the number of necessary inducing inputs, as Carl Rasmussen pointed out in a personal conversation.


I DON'T KNOW HOW TO PROPAGATE
ERROR CORRECTLY, SO I JUST PUT
ERROR BARS ON ALL MY ERROR BARS.

R. Munroe (2019). *Error Bars*. URL: https://xkcd.com/2110/.
License: Creative Commons 2.5 BY-NC-SA

Part VII

Appendix

> "You know," said Arthur, "it's at times like this, when I'm trapped in a Vogon airlock with a man from Betelgeuse, and about to die of asphyxiation in deep space that I really wish I'd listened to what my mother told me when I was young." "Why, what did she tell you?" "I don't know, I didn't listen." "Oh." Ford carried on humming.

—Douglas Adams, *The Hitchhiker's Guide to the Galaxy*

A

Benchmark Datasets

Table 3 describes purpose and origin of standard benchmark datasets used for Gaussian process regression. More information on PRECIPITATION can be found at http://www.image.ucar.edu/Data/US.monthly.met/. It appears that the datasets AILERONS, ELEVATORS and POLETELECOMM are no longer available under the link https://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html.

name	reference	url	description
ABALONE	Nash, Sellers, Talbot, Cawthorn, and Ford (1994), Waugh (1995), and Dua and Graff (2019)	https://archive.ics.uci.edu/ml/datasets/Abalone	age prediction of abalone from physical measurements
AILERONS	Camachol (1998)	n/a	control action prediction on the ailerons of an F16 aircraft
CT_SLICES	Graf, Kriegel, Schubert, Pölsterl, and Cavallaro (2011) and Dua and Graff (2019)	https://archive.ics.uci.edu/ml/datasets/Relative+location+of+CT+slices+on+axial+axis	prediction of relative location of a computer tomographic image in the human body in polar coordinates
ELEVATORS	Camachol (1998)	n/a	control action prediction on the elevators of an F16 aircraft
MPG	Quinlan (1993) and Dua and Graff (2019)	https://archive.ics.uci.edu/ml/datasets/auto+mpg	fuel consumption prediction in miles per gallon for different attributes of cars
POLETELECOMM	Weiss and Indurkhya (1995)	n/a	commercial telecommunication application–no further information
PRECIPITATION	Vanhatalo and Vehtari (2008)	github.com/gpstuff-dev/gpstuff/blob/master/gp/demo_regression_ppcs.m	US annual precipitation prediction for the year 1995
PUMADYN	Snelson and Ghahramani (2006)	ftp://ftp.cs.toronto.edu/pub/neuron/delve/data/tarfiles/pumadyn-family/pumadyn-32nm.tar.gz	acceleration prediction one of the arm links given angles, positions and velocities of other links of a *Puma560* robot
SARCOS	Vijayakumar and Schaal (2000)	http://www.gaussianprocess.org/gpml/data/	torque prediction for the seven degrees-of-freedom SARCOS anthropomorphic robot arm
SOUND	Turner (2010) and Wilson and Nickisch (2015)	https://github.com/kd383/GPML_SLD/blob/master/demo/sound/audio_data.mat	sound intensity prediction of a signal recorded over time for missing regions
TOY	Bartels and Hennig (2019)	n/a	targets are a draw from a zero-mean Gaussian process with squared exponential kernel (Eq. (43) with $\mathbf{\Lambda} = 0.25$ and $\theta_f = 2$), inputs stem in equal parts from a Gaussian mixture $(\mathcal{N}(0,1)+\mathcal{N}(1,0.1)+\mathcal{N}(-0.5,0.05))$ and the uniform distribution over $[0,1]$

Table 3: descriptions and sources for all datasets considered in this work.

B

Gaussian Processes

The density of an l-dimensional standard normal random vector $\boldsymbol{z}$ is defined as

$$p(\boldsymbol{z}) := \frac{1}{|2\pi|^{l/2}} \exp\left(-\frac{1}{2}\boldsymbol{z}^\intercal \boldsymbol{z}\right).$$

A real d-dimensional random vector $\boldsymbol{x}$ is said to be distributed Gaussian $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ iff there exist $l \in \mathbb{N}$ and $\boldsymbol{A} \in \mathbb{R}^{d\times l}$ s.t. $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{\mu}$, where $\boldsymbol{z}$ is distributed standard normal (Gut 2009, p. 454). These parameters also define mean and covariance of $\boldsymbol{x}$ with $\mathbb{E}\boldsymbol{x} = \boldsymbol{\mu}$ and $\operatorname{cov}\boldsymbol{x} = \boldsymbol{\Sigma}$. Gaussian random variables exhibit convenient properties as described in the two lemmata below.

Lemma 34. *Let $\boldsymbol{x} \in \mathbb{R}^d$ be Gaussian distributed with density $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma})$ for $\boldsymbol{x}_0 \in \mathbb{R}^d$ and $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ a positive semi-definite matrix. Let $\boldsymbol{M} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{z} \in \mathbb{R}^n$. Then $\boldsymbol{v} = \boldsymbol{M}\boldsymbol{x} + \boldsymbol{z}$ is also Gaussian, with*

$$p(\boldsymbol{v}) = \mathcal{N}(\boldsymbol{v}; \boldsymbol{M}\boldsymbol{x}_0 + \boldsymbol{z}, \boldsymbol{M}\boldsymbol{\Sigma}\boldsymbol{M}^\intercal).$$

Lemma 35. *Let $\boldsymbol{x} \in \mathbb{R}^d$ be distributed as in Lemma 34, and let observations $\boldsymbol{y} \in \mathbb{R}^n$ be generated from the conditional density*

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{M}\boldsymbol{x} + \boldsymbol{z}, \boldsymbol{\Lambda})$$

with $\boldsymbol{M} \in \mathbb{R}^{n\times d}$, $\boldsymbol{z} \in \mathbb{R}^n$, and $\Lambda \in \mathbb{R}^{n\times n}$ again positive-semidefinite. Then the associated conditional distribution on $\boldsymbol{x}$ after observing $\boldsymbol{y}$ is again Gaussian, with

$$\begin{aligned} p(\boldsymbol{x} \mid \boldsymbol{y}) = \mathcal{N}(\boldsymbol{x}; \bar{\boldsymbol{x}}, \bar{\boldsymbol{\Sigma}}) \qquad & \textit{where} \\ \bar{\boldsymbol{x}} &= \boldsymbol{x}_0 + \boldsymbol{\Sigma}\boldsymbol{M}^\intercal(\boldsymbol{M}\boldsymbol{\Sigma}\boldsymbol{M}^\intercal + \boldsymbol{\Lambda})^{-1}(\boldsymbol{y} - \boldsymbol{M}\boldsymbol{x}_0 - \boldsymbol{z}) \\ \bar{\boldsymbol{\Sigma}} &= \boldsymbol{\Sigma} - \boldsymbol{\Sigma}\boldsymbol{M}^\intercal(\boldsymbol{M}\boldsymbol{\Sigma}\boldsymbol{M}^\intercal + \boldsymbol{\Lambda})^{-1}\boldsymbol{M}\boldsymbol{\Sigma}). \end{aligned}$$

This formula also applies if $\boldsymbol{\Lambda} = 0$, i.e. *observations are made without noise, with the caveat that if $\boldsymbol{M}\boldsymbol{\Sigma}\boldsymbol{M}^\intercal$ is singular, the inverse should be interpreted as a pseudo-inverse.*

The following definition of a Gaussian process follows Rasmussen and Williams (2006, p. 13). A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution. It is completely specified by a mean function $\mu : \mathbb{R}^D \to \mathbb{R}$ and a covariance function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$. Hence, mean vector and covariance matrix remain implicit. The indices of the variables change from natural numbers to input locations of the function.

C

Additional Material for Part ii

Proposition 2. *Let* $\boldsymbol{V}, \boldsymbol{W} \in \mathbb{R}^{N \times N}$ *be square matrices and* $\boldsymbol{A}^\intercal, \boldsymbol{B} \in \mathbb{R}^{N \times M}$ *be rectangular.*

$$\boldsymbol{W} \circledast \boldsymbol{W} = \boldsymbol{\Gamma}_N (\boldsymbol{W} \otimes \boldsymbol{W}) \tag{SK1}$$

$$\boldsymbol{\Gamma}_M (\boldsymbol{A} \otimes \boldsymbol{A}) = (\boldsymbol{A} \otimes \boldsymbol{A}) \boldsymbol{\Gamma}_N \tag{SK2}$$

$$\boldsymbol{V} \circledast \boldsymbol{W} = \boldsymbol{W} \circledast \boldsymbol{V} \tag{SK3}$$

$$(\boldsymbol{A} \otimes \boldsymbol{A})(\boldsymbol{W} \circledast \boldsymbol{W})(\boldsymbol{B} \otimes \boldsymbol{B}) = (\boldsymbol{A}\boldsymbol{W}\boldsymbol{B}) \circledast (\boldsymbol{A}\boldsymbol{W}\boldsymbol{B}) \tag{SK4}$$

$$\boldsymbol{W} \circledast \boldsymbol{W} - \boldsymbol{V} \circledast \boldsymbol{V} = (\boldsymbol{W} + \boldsymbol{V}) \circledast (\boldsymbol{W} - \boldsymbol{V}) \tag{SK5}$$

$$(\boldsymbol{W} \circledast \boldsymbol{W})^{-1} = (\boldsymbol{W}^{-1} \circledast \boldsymbol{W}^{-1}). \tag{SK6}$$

The interpretation of Eq. (SK6) *requires some care: symmetric Kronecker product matrices are rank deficient. Eq.* (SK6) *is to be read in the sense that for symmetric* $\boldsymbol{Y} \in \mathbb{R}^{N \times N}$*,* i.e. $\boldsymbol{Y} = \boldsymbol{Y}^\intercal$, $\boldsymbol{X} := \operatorname{mat}\left((\boldsymbol{W}^{-1} \circledast \boldsymbol{W}^{-1}) \operatorname{vec}(\boldsymbol{Y})\right)$ *satisfies* $\operatorname{vec}(\boldsymbol{Y}) = (\boldsymbol{W} \circledast \boldsymbol{W}) \operatorname{vec}(\boldsymbol{X})$ *and* $\boldsymbol{X}$ *is the unique symmetric solution.*

Proof. The proofs for Eqs. (SK1) and (SK2) can be found in Magnus and Neudecker 1999[p. 46-50]. In the notation of Magnus and Neudecker 1999 $\boldsymbol{\Gamma} = N_n = D_n D_n^+$ and $K = 2\boldsymbol{\Gamma} - 2\boldsymbol{I}$. Eq. (SK1) is Theorem 13 (a). Eq. (SK2) follows from Theorem 9 (a).

To show $(\boldsymbol{W} \circledast \boldsymbol{V}) = (\boldsymbol{V} \circledast \boldsymbol{W})$, let $\boldsymbol{X} \in \mathbb{R}^{N \times N}$ be an arbitrary matrix.

$$\begin{aligned}
(\boldsymbol{V} \circledast \boldsymbol{W}) \operatorname{vec}(\boldsymbol{X}) &= \boldsymbol{\Gamma}(\boldsymbol{V} \otimes \boldsymbol{W}) \boldsymbol{\Gamma} \operatorname{vec}(\boldsymbol{X}) \\
&= \frac{1}{2} \boldsymbol{\Gamma}(\boldsymbol{V} \otimes \boldsymbol{W}) \operatorname{vec}(\boldsymbol{X} + \boldsymbol{X}^\intercal) \\
&= \frac{1}{2} \boldsymbol{\Gamma} \operatorname{vec}(\boldsymbol{V}(\boldsymbol{X} + \boldsymbol{X}^\intercal)\boldsymbol{W}^\intercal) \\
&= \frac{1}{4} \operatorname{vec}(\boldsymbol{V}(\boldsymbol{X} + \boldsymbol{X}^\intercal)\boldsymbol{W}^\intercal + \boldsymbol{W}(\boldsymbol{X} + \boldsymbol{X}^\intercal)\boldsymbol{V}^\intercal) \\
&= \frac{1}{2} \boldsymbol{\Gamma} \operatorname{vec}(\boldsymbol{W}(\boldsymbol{X} + \boldsymbol{X}^\intercal)\boldsymbol{V}^\intercal) \\
&= \frac{1}{2} \boldsymbol{\Gamma}(\boldsymbol{W} \otimes \boldsymbol{V}) \operatorname{vec}(\boldsymbol{X} + \boldsymbol{X}^\intercal) \\
&= \boldsymbol{\Gamma}(\boldsymbol{W} \otimes \boldsymbol{V}) \boldsymbol{\Gamma} \operatorname{vec}(\boldsymbol{X}) \\
&= (\boldsymbol{W} \circledast \boldsymbol{V}) \operatorname{vec}(\boldsymbol{X})
\end{aligned}$$

To show Eq. (SK4), use (SK2).

$$\begin{aligned}
(\boldsymbol{A} \otimes \boldsymbol{A})(\boldsymbol{W} \circledast \boldsymbol{W})(\boldsymbol{B} \otimes \boldsymbol{B}) &= (\boldsymbol{A} \otimes \boldsymbol{A}) \boldsymbol{\Gamma}_N (\boldsymbol{W} \otimes \boldsymbol{W}) \boldsymbol{\Gamma}_N (\boldsymbol{B} \otimes \boldsymbol{B}) \\
&= \boldsymbol{\Gamma}_M (\boldsymbol{A} \otimes \boldsymbol{A})(\boldsymbol{W} \otimes \boldsymbol{W})(\boldsymbol{B} \otimes \boldsymbol{B}) \boldsymbol{\Gamma}_M \\
&= \boldsymbol{\Gamma}_M (\boldsymbol{A}\boldsymbol{W}\boldsymbol{B} \otimes \boldsymbol{A}\boldsymbol{W}\boldsymbol{B}) \boldsymbol{\Gamma}_M \\
&= \boldsymbol{A}\boldsymbol{W}\boldsymbol{B} \circledast \boldsymbol{A}\boldsymbol{W}\boldsymbol{B}
\end{aligned}$$

The proof of Eq. (SK5) uses (SK3).

$$\begin{aligned}(\boldsymbol{A}+\boldsymbol{B})\circledast(\boldsymbol{A}-\boldsymbol{B}) &= \boldsymbol{\Gamma}(\boldsymbol{A}+\boldsymbol{B})\otimes(\boldsymbol{A}-\boldsymbol{B})\boldsymbol{\Gamma}\\ &= \boldsymbol{\Gamma}(\boldsymbol{A}\otimes\boldsymbol{A}-\boldsymbol{A}\otimes\boldsymbol{B}+\boldsymbol{B}\otimes\boldsymbol{A}-\boldsymbol{B}\otimes\boldsymbol{B})\boldsymbol{\Gamma}\\ &= \boldsymbol{A}\circledast\boldsymbol{A}-\boldsymbol{A}\circledast\boldsymbol{B}+\boldsymbol{B}\circledast\boldsymbol{A}-\boldsymbol{B}\circledast\boldsymbol{B}\\ &= \boldsymbol{A}\circledast\boldsymbol{A}-\boldsymbol{B}\circledast\boldsymbol{A}+\boldsymbol{B}\circledast\boldsymbol{A}-\boldsymbol{B}\circledast\boldsymbol{B}\\ &= \boldsymbol{A}\circledast\boldsymbol{A}-\boldsymbol{B}\circledast\boldsymbol{B}\end{aligned}$$

It remains to prove Eq. (SK6). Assume $\boldsymbol{Z}$ satisfies $(\boldsymbol{W}\circledast\boldsymbol{W})\operatorname{vec}(\boldsymbol{Z}) = \operatorname{vec}(\boldsymbol{Y})$ and $\boldsymbol{Z} = \mathbf{Z}^{\intercal}$. Then,

$$\begin{aligned}\operatorname{vec}(\boldsymbol{Y}) &= (\boldsymbol{W}\circledast\boldsymbol{W})\operatorname{vec}(\boldsymbol{Z})\\ &= (\boldsymbol{W}\otimes\boldsymbol{W})\boldsymbol{\Gamma}_N\operatorname{vec}(\boldsymbol{Z})\\ &\textit{// using Eq. (SK1) and Eq. (SK2)}\\ &= (\boldsymbol{W}\otimes\boldsymbol{W})\operatorname{vec}(\boldsymbol{Z})\\ &\textit{// since } \mathbf{Z} = \mathbf{Z}^{\intercal}\end{aligned}$$

and hence, $\operatorname{vec}(\boldsymbol{Z}) = (\boldsymbol{W}\otimes\boldsymbol{W})^{-1}\operatorname{vec}(\boldsymbol{Y})$. Using Eq. (K3) and again Eq. (SK1),

$$\begin{aligned}\boldsymbol{Z} &= (\boldsymbol{W}^{-1}\otimes\boldsymbol{W}^{-1})\operatorname{vec}(\boldsymbol{Y})\\ &\textit{// by Eq. (K3)}\\ &= (\boldsymbol{W}^{-1}\otimes\boldsymbol{W}^{-1})\boldsymbol{\Gamma}_N\operatorname{vec}(\boldsymbol{Y})\\ &\textit{// since } \boldsymbol{Y} = \boldsymbol{Y}^{\intercal}\\ &= (\boldsymbol{W}^{-1}\circledast\boldsymbol{W}^{-1})\operatorname{vec}(\boldsymbol{Y})\\ &\textit{// by Eq. (SK2) and Eq. (SK1)}\end{aligned}$$

which is the definition of $\boldsymbol{X}$. □

D
Additional Material for Part iii

D.1 Proof of Proposition 5

Proposition 5. *Consider a Gaussian MBI prior*

$$p(\boldsymbol{A}^{-1}) = \mathcal{N}(\boldsymbol{A}^{-1}; \operatorname{vec}\left(\boldsymbol{A}_0^{-1}\right), \boldsymbol{\Sigma}_0 \otimes \boldsymbol{W}_0),$$

conditioned on the left-multiplied information of Eq. (L)*. The associated marginal on* $\boldsymbol{x}$ *(Eq.* (MBI)*) is identical to the SBI posterior on* $\boldsymbol{x}$ *arising in Lemma 3 from* $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{\Sigma}_0)$*, under the conditions*

$$\boldsymbol{A}_0^{-1}\boldsymbol{b} = \boldsymbol{x}_0 \quad \textit{and} \quad \boldsymbol{b}^\intercal \boldsymbol{W}_0 \boldsymbol{b} = 1.$$

Proof of Proposition 5. The proof is analogous to the proof of Lemma 2.1 in Hennig (2015). Let $\boldsymbol{H} := \boldsymbol{A}^{-1}$ and let $\boldsymbol{H}_0 := \boldsymbol{A}_0^{-1}$. First note that by right-multiplying the information in Eq. (L) by $\boldsymbol{H}$:

$$\begin{aligned}
\boldsymbol{Y}_m^\intercal \boldsymbol{H} &= \boldsymbol{S}_m^\intercal \\
&\quad /\!/ \textit{ Eq. } (\mathrm{L}) \\
\implies \operatorname{vec}(\boldsymbol{Y}_m^\intercal \boldsymbol{H}) &= \operatorname{vec}(\boldsymbol{S}_m^\intercal) \\
&\quad /\!/ \textit{ applying } \operatorname{vec}(\ ) \textit{ on both sides} \\
\implies (\boldsymbol{Y}_m \otimes \boldsymbol{I}) \operatorname{vec}(\boldsymbol{H}) &= \operatorname{vec}(\boldsymbol{S}_m^\intercal) \\
&\quad /\!/ \textit{ from Eq. } (\mathrm{K1})
\end{aligned}$$

Let $\boldsymbol{\Omega}_0 = \boldsymbol{\Sigma}_0 \otimes \boldsymbol{W}_0$ and let $\boldsymbol{P} = \boldsymbol{Y}_m^\intercal \otimes \boldsymbol{I}$. Now the implied posterior on $\operatorname{vec}(\boldsymbol{H})$ can be computed using the standard laws of Gaussian conditioning (Lemma 35): $p(\boldsymbol{H} \mid \boldsymbol{S}_m, \boldsymbol{Y}_m) = \mathcal{N}(\operatorname{vec}(\boldsymbol{H}), \operatorname{vec}(\boldsymbol{H}_m), \boldsymbol{\Omega}_m)$ where

$$\begin{aligned}
\operatorname{vec}(\boldsymbol{H}_m) &= \operatorname{vec}(\boldsymbol{H}_0) + [\boldsymbol{P}\boldsymbol{\Omega}_0]^\intercal [\boldsymbol{P}\boldsymbol{\Omega}_0\boldsymbol{P}^\intercal]^{-1} (\operatorname{vec}(\boldsymbol{S}_m^\intercal) - \operatorname{vec}(\boldsymbol{Y}_m^\intercal \boldsymbol{H}_0)) \\
\boldsymbol{\Omega}_m &= \boldsymbol{\Omega}_0 - [\boldsymbol{P}\boldsymbol{\Omega}_0]^\intercal [\boldsymbol{P}\boldsymbol{\Omega}_0\boldsymbol{P}^\intercal]^{-1} (\boldsymbol{P}\boldsymbol{\Omega}_0)
\end{aligned}$$

Using Eq. (K2), $\boldsymbol{P}\boldsymbol{\Omega}_0 = (\boldsymbol{Y}_m^\intercal \otimes \boldsymbol{I})(\boldsymbol{\Sigma}_0 \otimes \boldsymbol{W}) = (\boldsymbol{Y}_m^\intercal \boldsymbol{\Sigma}_0) \otimes \boldsymbol{W}$ which implies $(P\boldsymbol{\Omega}_0)^\intercal = (\boldsymbol{\Sigma}_0 \boldsymbol{Y}_m) \otimes \boldsymbol{W}$ by Eq. (K4). Thus

$$\begin{aligned}
\boldsymbol{P}\boldsymbol{\Omega}_0\boldsymbol{P}^\intercal &= (\boldsymbol{Y}_m^\intercal \otimes \boldsymbol{I})(\boldsymbol{\Sigma}_0 \otimes \boldsymbol{W}_0)(\boldsymbol{Y}_m^\intercal \otimes \boldsymbol{I})^\intercal \\
&\quad /\!/ \textit{ definitions of } \boldsymbol{P} \textit{ and } \boldsymbol{\Omega}_0 \\
&= (\boldsymbol{Y}_m^\intercal \boldsymbol{\Sigma}_0 \boldsymbol{Y}_m) \otimes \boldsymbol{W}_0 \\
&\quad /\!/ \textit{ using Eqs. } (\mathrm{K2}) \textit{ and } (\mathrm{K4}) \\
\implies (P\boldsymbol{\Omega}_0 P^\intercal)^{-1} &= (\boldsymbol{Y}_m^\intercal \boldsymbol{\Sigma}_0 \boldsymbol{Y}_m)^{-1} \otimes \boldsymbol{W}_0^{-1} \\
&\quad /\!/ \textit{ using Eq. } (\mathrm{K3})
\end{aligned}$$

which allows us to conclude that

$$\begin{aligned}
&(\boldsymbol{P}\boldsymbol{\Omega}_0)^\intercal(\boldsymbol{P}\boldsymbol{\Omega}_0\boldsymbol{P}^\intercal)^{-1} \\
&\quad = [(\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)\otimes\boldsymbol{W}][(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}\otimes\boldsymbol{W}^{-1}] \\
&\quad = (\boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1})\otimes\boldsymbol{I} \\
\implies &(\boldsymbol{P}\boldsymbol{\Omega}_0)^\intercal(\boldsymbol{P}\boldsymbol{\Omega}_0\boldsymbol{P}^\intercal)^{-1}(\boldsymbol{P}\boldsymbol{\Omega}_0) \\
&\quad = (\boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0)\otimes\boldsymbol{W}_0.
\end{aligned}$$

From these expressions one can simplify the expressions for $\operatorname{vec}(\boldsymbol{H}_m)$:

$$\begin{aligned}
\operatorname{vec}(\boldsymbol{H}_m) &= \operatorname{vec}(\boldsymbol{H}_0) + (\boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}\otimes\boldsymbol{I})(\operatorname{vec}(\boldsymbol{S}_m^\intercal) - \operatorname{vec}(\boldsymbol{Y}_m^\intercal\boldsymbol{H}_0)) \\
&= \operatorname{vec}\Big(\boldsymbol{H}_0 + \boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}(\boldsymbol{S}_m^\intercal - \boldsymbol{Y}_m^\intercal\boldsymbol{H}_0)\Big) \\
&\quad \textit{// by Eq. (K1).}
\end{aligned}$$

Analogously, for $\boldsymbol{\Omega}_m$:

$$\begin{aligned}
\boldsymbol{\Omega}_m &= \boldsymbol{\Sigma}_0\otimes\boldsymbol{W} - (\boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0)\otimes\boldsymbol{W}_0 \\
&= (\boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0)\otimes\boldsymbol{W}_0 \\
&\quad \textit{// by Eq. (K5)}
\end{aligned}$$

It remains to project the posterior into $\mathbb{R}^d$ by performing the matrix-vector product $\boldsymbol{H}\boldsymbol{b} = \boldsymbol{x} = (\boldsymbol{I}\otimes\boldsymbol{b}^\intercal)\boldsymbol{H}$ using Eq. (K1). Thus, the implied posterior is $\boldsymbol{x}\sim\mathcal{N}(\bar{\boldsymbol{x}}_m, \bar{\boldsymbol{\Sigma}}_m)$, with

$$\begin{aligned}
\bar{\boldsymbol{x}}_m &= (\boldsymbol{I}\otimes\boldsymbol{b}^\intercal)\operatorname{vec}(\boldsymbol{H}_0 + \boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}(\boldsymbol{S}_m^\intercal - \boldsymbol{Y}_m^\intercal\boldsymbol{H}_0)) \\
&= \operatorname{vec}\Big(\boldsymbol{H}_0\boldsymbol{b} + \boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}(\boldsymbol{S}_m^\intercal\boldsymbol{b} - \boldsymbol{Y}_m^\intercal\boldsymbol{H}_0\boldsymbol{b})\Big) \\
&= \boldsymbol{x}_0 + \boldsymbol{\Sigma}_0 A^\intercal\boldsymbol{S}_m(\boldsymbol{S}_m^\intercal A\boldsymbol{\Sigma}_0 A^\intercal\boldsymbol{S}_m)^{-1}\boldsymbol{S}_m^\intercal(\boldsymbol{b} - A\boldsymbol{x}_0). \\
&\quad \textit{// since } \boldsymbol{H}_0\boldsymbol{b} = \boldsymbol{x}_0 \textit{ and } \boldsymbol{Y}_m = \boldsymbol{A}^\intercal\boldsymbol{S}_m
\end{aligned}$$

Furthermore

$$\begin{aligned}
\bar{\boldsymbol{\Sigma}}_m &= (\boldsymbol{I}\otimes\boldsymbol{b}^\intercal)\cdot\Big[(\boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0)\otimes\boldsymbol{W}_0\Big]\cdot(\boldsymbol{I}\otimes\boldsymbol{b}^\intercal)^\intercal \\
&= (\boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_0\boldsymbol{Y}_m(\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0\boldsymbol{Y}_m)^{-1}\boldsymbol{Y}_m^\intercal\boldsymbol{\Sigma}_0)\times\boldsymbol{b}^\intercal\boldsymbol{W}_0\boldsymbol{b} \\
&\quad \textit{// using Eq. (K2) and that } \boldsymbol{b}^\intercal\boldsymbol{W}_0\boldsymbol{b} \textit{ is scalar} \\
&= \boldsymbol{\Sigma}_0 - \boldsymbol{\Sigma}_0 A^\intercal\boldsymbol{S}_m(\boldsymbol{S}_m^\intercal A\boldsymbol{\Sigma}_0 A^\intercal\boldsymbol{S}_m)^{-1}\boldsymbol{S}_m^\intercal A\boldsymbol{\Sigma}_0 \\
&\quad \textit{// since } \boldsymbol{b}^\intercal\boldsymbol{W}_0\boldsymbol{b} = 1 \textit{ and } \boldsymbol{Y}_m = \boldsymbol{A}^\intercal\boldsymbol{S}_m
\end{aligned}$$

Now note that $\boldsymbol{x}_m = \bar{\boldsymbol{x}}_m$ and $\boldsymbol{\Sigma}_m = \bar{\boldsymbol{\Sigma}}_m$, as in Lemma 3. Thus, the proof is complete. □

D.2 Proof of Theorem 15

Theorem 36 (Theorem 2.3 in Hennig (2015))**.** *Let $\boldsymbol{W}\in\mathbb{R}^{d\times d}$ be symmetric and positive definite. Assume a Gaussian prior of symmetric mean $\boldsymbol{A}_0^{-1}$ and covariance $\boldsymbol{W}\otimes\boldsymbol{W}$ on the elements of a symmetric matrix $\boldsymbol{A}^{-1}$. After m linearly independent noise-free observations* *of the form* $\boldsymbol{S} = \boldsymbol{A}^{-1}\boldsymbol{Y}$*,* $\boldsymbol{Y} \in \mathbb{R}^{d\times m}, \mathrm{rk}(\boldsymbol{Y}) = m$*, the posterior belief over* $\boldsymbol{A}^{-1}$ *is a Gaussian with mean*

$$\begin{aligned}\boldsymbol{A}_m^{-1} = \boldsymbol{A}_0^{-1} &+ (\boldsymbol{S} - \boldsymbol{A}_0^{-1}\boldsymbol{Y})\boldsymbol{G}\boldsymbol{Y}^\intercal\boldsymbol{W} \\ &+ \boldsymbol{W}\boldsymbol{Y}\boldsymbol{G}(\boldsymbol{S} - \boldsymbol{A}_0^{-1}\boldsymbol{Y})^\intercal \\ &+ \boldsymbol{W}\boldsymbol{Y}\boldsymbol{G}\boldsymbol{Y}^\intercal(\boldsymbol{S} - \boldsymbol{A}_0^{-1}\boldsymbol{Y})\boldsymbol{G}\boldsymbol{Y}^\intercal\boldsymbol{W} \end{aligned} \tag{59}$$

and posterior covariance

$$\boldsymbol{V}_m = (\boldsymbol{W} - \boldsymbol{W}\boldsymbol{Y}\boldsymbol{G}\boldsymbol{Y}^\intercal\boldsymbol{W}) \otimes (\boldsymbol{W} - \boldsymbol{W}\boldsymbol{Y}\boldsymbol{G}\boldsymbol{Y}^\intercal\boldsymbol{W})$$

where $\boldsymbol{G} := (\boldsymbol{Y}^\intercal\boldsymbol{W}\boldsymbol{Y})^{-1}$.

The following proofs are due to Philipp Hennig. My contribution are simplifications that reduce the length by approximately one page.

Proof of Theorem 15 by Philipp Hennig. Denote by $\boldsymbol{x}_i^{CG}$ the conjugate gradient estimate in iteration i and with $\boldsymbol{p}_i$ the search direction in that iteration. From one iteration to the next, the update to the solution can be written as (Nocedal and Wright 1999, p. 108)

$$\boldsymbol{x}_{i+1}^{\mathrm{CG}} = \boldsymbol{x}_i^{\mathrm{CG}} + \frac{\boldsymbol{r}_i^\intercal \boldsymbol{p}_i}{\boldsymbol{p}_i^\intercal \boldsymbol{A}\boldsymbol{p}_i}\boldsymbol{p}_i.$$

Comparing this update to lines 7 to 10 in Algorithm 3 it is sufficient to show that $\boldsymbol{d}_i \propto \boldsymbol{p}_i$ which follows from Lemma 37. □

Lemma 37. *Assume that CG does not terminate before d iterations. Using the prior of Theorem 15 in Algorithm 3, the directions* $\boldsymbol{d}_i$ *are scaled conjugate gradients search directions,* i.e.

$$\boldsymbol{d}_i = \gamma_i \boldsymbol{p}_i^{CG}$$

where $\boldsymbol{p}_i^{CG}$ *is the CG search direction in iteration i and* $\gamma_i \in \mathbb{R} \setminus \{0\}$*.*

Proof. The proof proceeds by induction. Throughout we will suppress the superscript CG on the CG search directions, *i.e.* $\boldsymbol{p}_i^{\mathrm{CG}} = \boldsymbol{p}_i$. For $i = 1$, $\boldsymbol{A}_{i-1}^{-1} = \alpha I$ by assumption and therefore $\boldsymbol{d}_i = \alpha \boldsymbol{r}_0$ which is the first CG search direction scaled by $\gamma_1 = \alpha \neq 0$.

For the inductive step, suppose that the search directions $\boldsymbol{s}_1, ..., \boldsymbol{s}_{i-1}$ are scaled CG directions and that the vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_{i-1}$ are the same as the first $i-1$ solution estimates produced by CG. We will prove that $\boldsymbol{s}_i$ is the i^{th} CG search direction, and that $\boldsymbol{x}_i$ is the i^{th} solution estimate from CG. Lemma 39 states that $\boldsymbol{d}_i$ can be written as

$$\boldsymbol{d}_i = \boldsymbol{A}_{i-1}^{-1}\boldsymbol{r}_{i-1} = \sum_{j<i} \nu_j \boldsymbol{s}_j + \nu_i \boldsymbol{r}_{i-1}.$$

where $\nu_j \in \mathbb{R}, j = 1, \dots, i$. Under the prior, the posterior mean $\boldsymbol{A}_i^{-1}$ is always symmetric. This allows application of Lemma 38, so that $\{\boldsymbol{s}_1, \dots, \boldsymbol{s}_{i-1}, \boldsymbol{d}_i\}$ is an A-conjugate set. Thus we have, for $\ell < i$:

$$0 = \boldsymbol{s}_\ell^\intercal \boldsymbol{A}\boldsymbol{d}_i = \nu_\ell \boldsymbol{s}_\ell^\intercal \boldsymbol{A}\boldsymbol{s}_\ell + \nu_i \boldsymbol{s}_\ell^\intercal \boldsymbol{A}\boldsymbol{r}_{i-1}$$

$$= \nu_\ell \boldsymbol{s}_\ell^\intercal \boldsymbol{A} \boldsymbol{s}_\ell + \nu_i \boldsymbol{y}_\ell^\intercal \boldsymbol{r}_{i-1}. \tag{60}$$

Now note that

$$\boldsymbol{y}_\ell^\intercal \boldsymbol{r}_{i-1} = (\boldsymbol{r}_\ell - \boldsymbol{r}_{\ell-1})^\intercal \boldsymbol{r}_{i-1}.$$

This follows from Line 10 of Algorithm 3, from which it is clear that $\boldsymbol{y}_\ell = \boldsymbol{r}_\ell - \boldsymbol{r}_{\ell-1}$. Recall that the CG residuals $\boldsymbol{r}_j$ are orthogonal Nocedal and Wright 1999, p. 109, and that from the inductive assumption, Algorithm 3 is equivalent to CG up to iteration $i-1$). Thus, for $\ell < i-1$ we have that

$$\begin{aligned} & \boldsymbol{y}_\ell^\intercal \boldsymbol{r}_{i-1} = 0 \\ \implies \; & \boldsymbol{s}_\ell \boldsymbol{A} \boldsymbol{d}_i = \nu_\ell \boldsymbol{s}_\ell^\intercal \boldsymbol{A} \boldsymbol{s}_\ell = 0 \qquad \forall\, \ell < i-1 \end{aligned}$$

where the second line is from application of the first line in Eq. (60). However, $\boldsymbol{A}$ is positive definite and by assumption the algorithm has not converged, so $\boldsymbol{d}_\ell \neq \boldsymbol{0}$. Furthermore clearly $\boldsymbol{s}_\ell^\intercal \boldsymbol{A} \boldsymbol{s}_\ell \neq 0$. Hence we must have that

$$\nu_\ell = 0 \qquad \forall\, j < i-1.$$

Equation (D.2) thus simplifies to

$$\boldsymbol{d}_i = \nu_{i-1} \boldsymbol{s}_{i-1} + \nu_i \boldsymbol{r}_{i-1} = \nu_{i-1} \alpha_{i-1} \boldsymbol{d}_{i-1} + \nu_i \boldsymbol{r}_{i-1}.$$

Now, again by Lemma 38, $\boldsymbol{d}_i$ must be conjugate to $\boldsymbol{s}_{i-1}$ which implies $\nu_i \neq 0$. Pre-multiplying Eq. (D.2) by $\boldsymbol{s}_{i-1}^\intercal \boldsymbol{A}$ gives

$$\begin{aligned} 0 &= \nu_{i-1} \alpha_{i-1} \boldsymbol{s}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{d}_{i-1} + \nu_i \boldsymbol{s}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{r}_{i-1} \\ \implies \; \nu_{i-1} \alpha_{i-1} &= -\nu_i \frac{\boldsymbol{s}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{r}_{i-1}}{\boldsymbol{s}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{d}_{i-1}}. \end{aligned}$$

Thus, $\boldsymbol{d}_i$ can be written as

$$\begin{aligned} \boldsymbol{d}_i &= \nu_i \left(\boldsymbol{r}_{i-1} - \frac{\boldsymbol{s}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{r}_{i-1}}{\boldsymbol{s}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{d}_{i-1}} \boldsymbol{d}_{i-1} \right) \\ &= \nu_i \left(\boldsymbol{r}_{i-1} - \frac{\boldsymbol{p}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{r}_{i-1}}{\boldsymbol{p}_{i-1}^\intercal \boldsymbol{A} \boldsymbol{p}_{i-1}} \boldsymbol{p}_{i-1} \right) \end{aligned} \tag{61}$$

where the second line again applies the inductive assumption, that $\boldsymbol{d}_{i-1}$ and $\boldsymbol{s}_{i-1}$ are proportional to the CG search direction $\boldsymbol{p}_{i-1}$, noting that the proportionality constants on numerator and denominator cancel. The term inside the brackets is precisely the i^{th} CG search direction. This completes the result. □

Lemma 38. *If the belief over $\boldsymbol{A}_m^{-1}$ is symmetric for all $m = 0, \dots, d$ and $\boldsymbol{A}$ is symmetric and positive definite, then Algorithm 3 produces $\boldsymbol{A}$-conjugate directions.*

Proof. The proof is by induction. Note that the case $i = 1$ is irrelevant since a set consisting of one element is trivially A-conjugate. On many occasions the proof relies on the consistency of the MBI belief, *i.e.* $\boldsymbol{A}_i^{-1}\boldsymbol{z}_k = \boldsymbol{d}_k$ for $k \leq i$ and by symmetry $\boldsymbol{z}_k^\intercal \boldsymbol{A}_i^{-1} = \boldsymbol{d}_k^\intercal$. Thus, for the base case $i = 2$ we have:

$$\begin{aligned}
\boldsymbol{d}_1^\intercal \boldsymbol{A}\boldsymbol{d}_2 &= -\boldsymbol{d}_1^\intercal \boldsymbol{A}(\boldsymbol{A}_1^{-1}\boldsymbol{r}_1) \\
&= -\boldsymbol{d}_1^\intercal \boldsymbol{A}(\boldsymbol{A}_1^{-1}(\boldsymbol{y}_1 + \boldsymbol{r}_0)) \\
&= -\boldsymbol{d}_1^\intercal \boldsymbol{A}(\boldsymbol{s}_1 + \boldsymbol{A}_1^{-1}\boldsymbol{r}_0)
\end{aligned}$$

where the second line is by Line 10 of Algorithm 3. Now recall that $\alpha_1 = -\boldsymbol{d}_1^\intercal \boldsymbol{r}_0 / \boldsymbol{d}_1^\intercal \boldsymbol{A}\boldsymbol{d}_1$ to give:

$$\begin{aligned}
\boldsymbol{d}_1^\intercal \boldsymbol{A}\boldsymbol{d}_2 &= -\alpha_1 \boldsymbol{d}_1^\intercal \boldsymbol{A}\boldsymbol{d}_1 - \boldsymbol{d}_1^\intercal \boldsymbol{A}\boldsymbol{A}_1^{-1}\boldsymbol{r}_0 \\
&= \boldsymbol{d}_1^\intercal \boldsymbol{r}_0 - \boldsymbol{d}_1^\intercal \boldsymbol{A}\boldsymbol{A}_1^{-1}\boldsymbol{r}_0 \\
&= \boldsymbol{d}_1^\intercal \boldsymbol{r}_0 - \boldsymbol{z}_1^\intercal \boldsymbol{A}_1^{-1}\boldsymbol{r}_0 \\
&= \boldsymbol{d}_1^\intercal \boldsymbol{r}_0 - \boldsymbol{d}_1^\intercal \boldsymbol{r}_0 \qquad (62) \\
&= 0.
\end{aligned}$$

Here, the symmetry of the estimator $\boldsymbol{A}_i^{-1}$ is used in Eq. (62). For the inductive step, assume $\{\boldsymbol{d}_0, \dots, \boldsymbol{d}_{i-1}\}$ are pairwise $\boldsymbol{A}$-conjugate. For any $k < i$ we have:

$$\begin{aligned}
\boldsymbol{d}_k^\intercal \boldsymbol{A}\boldsymbol{d}_i &= -\boldsymbol{d}_k^\intercal \boldsymbol{A}(\boldsymbol{A}_i^{-1}\boldsymbol{r}_i) \\
&= -\boldsymbol{d}_k^\intercal \boldsymbol{A}\boldsymbol{A}_i^{-1}\left(\sum_{j \leq i} \boldsymbol{y}_j + \boldsymbol{r}_0\right)
\end{aligned}$$

where the second line follows from the fact that $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} + \boldsymbol{y}_i$. Thus, we have:

$$\begin{aligned}
\boldsymbol{d}_k^\intercal \boldsymbol{A}\boldsymbol{d}_i &= -\boldsymbol{d}_k^\intercal \boldsymbol{A}\left(\sum_{j \leq i} \boldsymbol{s}_j + \boldsymbol{A}_i^{-1}\boldsymbol{r}_0\right) \\
&= -\boldsymbol{d}_k^\intercal \boldsymbol{A}\left(\sum_{j \leq i} \alpha_j \boldsymbol{d}_j + \boldsymbol{A}_i^{-1}\boldsymbol{r}_0\right).
\end{aligned}$$

Now, applying the conjugacy from the inductive assumption:

$$\begin{aligned}
\boldsymbol{d}_k^\intercal \boldsymbol{A}\boldsymbol{d}_i &= -\alpha_k \boldsymbol{d}_k^\intercal \boldsymbol{A}\boldsymbol{d}_k - \boldsymbol{d}_k^\intercal \boldsymbol{A}(\boldsymbol{A}_i^{-1}\boldsymbol{r}_0) \\
&= \boldsymbol{d}_k^\intercal \boldsymbol{r}_{k-1} - \boldsymbol{d}_k^\intercal \boldsymbol{r}_0 \\
&= \boldsymbol{d}_k^\intercal \left(\sum_{j<k} \boldsymbol{y}_j + \boldsymbol{r}_0\right) - \boldsymbol{d}_k^\intercal \boldsymbol{r}_0 = 0 \\
&= \sum_{j<k} \alpha_j \boldsymbol{d}_k^\intercal \boldsymbol{A}\boldsymbol{d}_j = 0.
\end{aligned}$$

where the second line rearranges line 6 of the algorithm to obtain $\alpha_i \boldsymbol{d}_i^\intercal \boldsymbol{z}_i = -\boldsymbol{d}_i^\intercal \boldsymbol{r}_{i-1}$. The third line again uses that $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} + \boldsymbol{y}_i$, while the fourth line is from the assumed conjugacy. □

Lemma 39. *Under the prior in Theorem 15 and given scaled CG search directions* $\boldsymbol{p}_1, \dots, \boldsymbol{p}_i$, *it holds that* $\boldsymbol{A}_i^{-1}\boldsymbol{r}_i \in \operatorname{span}\{\boldsymbol{p}_1, \dots, \boldsymbol{p}_i, \boldsymbol{r}_i\}$.

Proof. Recall first that under the prior in Theorem 15, $\boldsymbol{A}_0^{-1} = \alpha I$. Then by inspection of Eq. (59) we have $\boldsymbol{A}_i^{-1}\boldsymbol{r}_i \in \mathcal{S}$ where

$$\mathcal{S} = \mathrm{span}\{\boldsymbol{r}_i, \boldsymbol{p}_1, ..., \boldsymbol{p}_i, \boldsymbol{y}_1, ..., \boldsymbol{y}_i, \boldsymbol{W}\boldsymbol{y}_1, ..., \boldsymbol{W}\boldsymbol{y}_i\}$$

By choice of $\boldsymbol{W} = \beta\boldsymbol{I} + \gamma\boldsymbol{A}^{-1}$, $\mathcal{S} = \mathrm{span}\{\boldsymbol{r}_i, \boldsymbol{p}_1, ..., \boldsymbol{p}_i, \boldsymbol{y}_1, ..., \boldsymbol{y}_i\}$. From line 10 of Algorithm 3 $\boldsymbol{y}_i = \boldsymbol{r}_i - \boldsymbol{r}_{i-1}$ and therefore $\mathcal{S} = \mathrm{span}\{\boldsymbol{r}_1, ..., \boldsymbol{r}_i, \boldsymbol{p}_1, ..., \boldsymbol{p}_i\}$. By Theorem 5.3 in Nocedal and Wright 1999, p. 109 the span of the conjugate gradients residuals and search directions are equivalent. Therefore $\mathcal{S} \subseteq \{\boldsymbol{r}_i, \boldsymbol{p}_1, ..., \boldsymbol{p}_i\}$. □

E Additional Material for Part iv

E.1 Sampling from a Gaussian with Symmetric Kronecker Covariance

To sample matrices from the KMCG posterior (Eq. 34) the following proposition will be useful.

Proposition 40. *Let* $\boldsymbol{W}, \boldsymbol{W}_M \in \mathbb{R}^{N \times N}$ *be symmetric and positive semi-definite matrices s.t.* $\boldsymbol{W} - \boldsymbol{W}_M$ *is symmetric positive-semidefinite as well. Further let* $\operatorname{vec}(\boldsymbol{Y}) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W} \circledast \boldsymbol{W} - \boldsymbol{W}_M \circledast \boldsymbol{W}_M)$*, denote with* $\boldsymbol{L}_+$ *the Cholesky of* $\boldsymbol{W} + \boldsymbol{W}_M$*, with* $\boldsymbol{L}_-$ *the Cholesky of* $\boldsymbol{W} - \boldsymbol{W}_M$ *and let* $\operatorname{vec}(\boldsymbol{X}) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{N^2})$*, then* $\boldsymbol{\Gamma}(\boldsymbol{L}_1 \otimes \boldsymbol{L}_2) \operatorname{vec}(\boldsymbol{X})$ *and* $\operatorname{vec}(\boldsymbol{Y})$ *have the same distribution and* $\boldsymbol{Y} = \boldsymbol{Y}^\intercal$*.*

Proof. As $\operatorname{vec}(\boldsymbol{X})$ is standard normal, $\boldsymbol{\Gamma}(\boldsymbol{L}_+ \otimes \boldsymbol{L}_-) \operatorname{vec}(\boldsymbol{X})$ is distributed Gaussian with mean $\boldsymbol{0}$ and covariance matrix $\boldsymbol{\Gamma}(\boldsymbol{L}_+ \otimes \boldsymbol{L}_-)(\boldsymbol{\Gamma}(\boldsymbol{L}_+ \otimes \boldsymbol{L}_-))^\intercal$.

$$\begin{aligned}\boldsymbol{\Gamma}(\boldsymbol{L}_+ \otimes \boldsymbol{L}_-)\left[\boldsymbol{\Gamma}(\boldsymbol{L}_+ \otimes \boldsymbol{L}_-)\right]^\intercal &= \boldsymbol{\Gamma}(\boldsymbol{L}_+ \otimes \boldsymbol{L}_-)(\boldsymbol{L}_+^\intercal \otimes \boldsymbol{L}_-^\intercal)\boldsymbol{\Gamma} \\ &= (\boldsymbol{L}_+\boldsymbol{L}_+^\intercal) \circledast (\boldsymbol{L}_-\boldsymbol{L}_-^\intercal) \\ &= (\boldsymbol{W} + \boldsymbol{W}_M) \circledast (\boldsymbol{W} - \boldsymbol{W}_M)\end{aligned}$$

According to Equation (SK5): $(\boldsymbol{W} + \boldsymbol{W}_M) \circledast (\boldsymbol{W} - \boldsymbol{W}_M) = \boldsymbol{W} \circledast \boldsymbol{W} - \boldsymbol{W}_M \circledast \boldsymbol{W}_M$. $\boldsymbol{Y}$ is symmetric due to the application of the $\boldsymbol{\Gamma}$-operator. □

E.2 Proofs

This section contains the proof of Proposition 20:

Proposition 20 (Subset of Regressors)**.** *Consider the prior of Eq. (30) with* $k_0 := 0$ *and* $w := k$ *and the likelihood defined in Eq. (33) with* $s_{ij} = \delta_{ij}$*. Then the posterior mean* k_M *is equivalent to that of SoR:*

$$k_M(\boldsymbol{x}, \boldsymbol{z}) = k_{SoR} = k(\boldsymbol{x}, \boldsymbol{X}_U) k(\boldsymbol{X}_U, \boldsymbol{X}_U)^{-1} k(\boldsymbol{X}_U, \boldsymbol{z})$$

where $\boldsymbol{X}_U$ *are inducing inputs, not necessarily part of* $\boldsymbol{X}$*.*

The mentioned prior is $p(k) := \mathcal{N}(0, \psi)$ where $\psi(k(\boldsymbol{a}, \boldsymbol{b}), k(\boldsymbol{c}, \boldsymbol{d})) := \frac{1}{2} k(\boldsymbol{a}, \boldsymbol{c}) k(\boldsymbol{b}, \boldsymbol{d}) + \frac{1}{2} k(\boldsymbol{a}, \boldsymbol{d}) k(\boldsymbol{b}, \boldsymbol{c})$. Denote with $\boldsymbol{Y}$ the noise-free observations of k, $\boldsymbol{Y} = \operatorname{mat}\left(\boldsymbol{T}_{\boldsymbol{p}} k\right)$ where

$$\boldsymbol{T}_{\boldsymbol{p}} : k \mapsto \operatorname{vec}\left(\left[\iint k(\boldsymbol{x}, \boldsymbol{z}) p_i(\boldsymbol{x}) p_j(\boldsymbol{z}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{z}\right]_{ij}\right) \tag{32}$$

$$p_i(\boldsymbol{x}) = \sum_{j=1}^{M} s_{ij} \delta(\boldsymbol{x} - \boldsymbol{x}_{u_j}) \tag{33}$$

which implies the likelihood $p(\boldsymbol{Y} \mid \boldsymbol{T_p}, k) = \delta(\boldsymbol{Y} - \boldsymbol{T_p} k)$. Proposition 20 follows from the more general Proposition 41.

Proposition 41. *Consider the prior of Eq.* (30) *(without the restriction $w = k$) and the likelihood defined in Eq.* (33)*. The posterior over k is $p(k \mid \boldsymbol{Y} = \boldsymbol{T_p} k) = \mathcal{N}(k; k_M, \psi_M)$ with posterior mean*

$$\begin{aligned} k_M(\boldsymbol{a},\boldsymbol{b}) = k_0(\boldsymbol{a},\boldsymbol{b}) &+ w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal \boldsymbol{K}_M \boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{b}) \\ &- w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal k_0(\boldsymbol{X}_U,\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{b}) \end{aligned} \tag{63}$$

and posterior variance

$$\begin{aligned} \psi_M(k(\boldsymbol{a},\boldsymbol{b}), k(\boldsymbol{c},\boldsymbol{d})) = &\frac{1}{2}w(\boldsymbol{a},\boldsymbol{c})w(\boldsymbol{b},\boldsymbol{d}) + \frac{1}{2}w(\boldsymbol{a},\boldsymbol{d})w(\boldsymbol{b},\boldsymbol{c}) \\ &- \frac{1}{2}w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{c})w(\boldsymbol{b},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{d}) \\ &- \frac{1}{2}w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{d})w(\boldsymbol{b},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{c}) \end{aligned} \tag{64}$$

where $\boldsymbol{W}_M = w(\boldsymbol{X}_U, \boldsymbol{X}_U)$.

Proof. Proofing the proposition is tedious linear Algebra which is what follows now. If prior and likelihood are Gaussian, so is the posterior with mean and variance:

$$\begin{aligned} k_M(\boldsymbol{a},\boldsymbol{b}) &= k_0(\boldsymbol{a},\boldsymbol{b}) - (\boldsymbol{T_p}\psi(k(\boldsymbol{a},\boldsymbol{b}),\cdot))^\intercal(\boldsymbol{T_p}(\boldsymbol{T_p} w)^\intercal)^{-1}\operatorname{vec}(\boldsymbol{Y} - \boldsymbol{S}^\intercal k_0(\boldsymbol{X}_U,\boldsymbol{X}_U)\boldsymbol{S}), \\ \psi_M(k(\boldsymbol{a},\boldsymbol{b}),k(\boldsymbol{c},\boldsymbol{d})) &= \psi((\boldsymbol{a},\boldsymbol{b}),(\boldsymbol{c},\boldsymbol{d})) - (\boldsymbol{T_p}\psi((\boldsymbol{a},\boldsymbol{b}),(\cdot,\cdot)))^\intercal(\boldsymbol{T_p}(\boldsymbol{T_p}\psi)^\intercal)^{-1}\boldsymbol{T_p}\psi(\boldsymbol{c},\boldsymbol{d}),(\cdot,\cdot)) \end{aligned}$$

Lemma 42 allows to write

$$\begin{aligned} &(\boldsymbol{T_p}\psi(k(\boldsymbol{a},\boldsymbol{b}),\cdot))^\intercal(\boldsymbol{T_p}(\boldsymbol{T_p} w)^\intercal)^{-1} \\ &= \frac{1}{2}\operatorname{vec}(\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{a})w(\boldsymbol{b},\boldsymbol{X}_U) + w(\boldsymbol{X}_U,\boldsymbol{b})w(\boldsymbol{a},\boldsymbol{X}_U))\boldsymbol{S})^\intercal \left((\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1} \otimes (\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\right) \\ &= \frac{1}{2}\operatorname{vec}\left((\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{a})w(\boldsymbol{b},\boldsymbol{X}_U) + w(\boldsymbol{X}_U,\boldsymbol{b})w(\boldsymbol{a},\boldsymbol{X}_U))\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\right)^\intercal \end{aligned}$$

and thus for Eq. (63)

$$\begin{aligned} k_M(\boldsymbol{a},\boldsymbol{b}) = k_0&(\boldsymbol{a},\boldsymbol{b}) \\ &+ \frac{1}{2}\operatorname{tr}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{a})w(\boldsymbol{b},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}(\boldsymbol{Y} - k_0(\boldsymbol{X}_U,\boldsymbol{X}_U)) \\ &+ \frac{1}{2}\operatorname{tr}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{b})w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}(\boldsymbol{Y} - k_0(\boldsymbol{X}_U,\boldsymbol{X}_U)) \\ &= k_0(\boldsymbol{a},\boldsymbol{b}) + w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal \boldsymbol{K}_M \boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{b}) \\ &- w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal k_0(\boldsymbol{X}_U,\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{b}) \end{aligned}$$

The derivation for Eq. (64) follows analogously. □

Lemma 42. *Let $\boldsymbol{T_p}$ be as in Eq.* (36)*.*

$$\boldsymbol{T_p} w(k(\boldsymbol{a},\boldsymbol{b}),\cdot) = \frac{1}{2}\operatorname{vec}\left(\boldsymbol{S}^\intercal\left(w(\boldsymbol{X}_U,\boldsymbol{a})w(\boldsymbol{b},\boldsymbol{X}_U) + w(\boldsymbol{X}_U,\boldsymbol{b})w(\boldsymbol{a},\boldsymbol{X}_U)\right)\boldsymbol{S}\right) \tag{65}$$

$$\boldsymbol{T_p}(\boldsymbol{T_p} w(\cdot,\cdot))^\intercal = (\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S}) \otimes (\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S}) \tag{66}$$

Proof. Denote with $\mathsf{mat}\,(\;)$ the complement of the vectorization operator, *i.e.* $\mathsf{mat}\,(\mathsf{vec}\,(\boldsymbol{A})) = \boldsymbol{A}$. Define the matrix $\boldsymbol{S} \in \mathbb{R}^{N\times M}$ as $\boldsymbol{S}_{ij} = s_{ij}$ and denote with $\boldsymbol{S}_l$ the l-th column of $\boldsymbol{S}$. Also recall that by Eq. (31) $\psi(k(\boldsymbol{a},\boldsymbol{b}),k(\boldsymbol{x},\boldsymbol{z})) = \frac{1}{2}(w(\boldsymbol{a},\boldsymbol{x})w(\boldsymbol{b},\boldsymbol{z}) + w(\boldsymbol{a},\boldsymbol{z})w(\boldsymbol{b},\boldsymbol{x}))$.

$$
\begin{aligned}
&[\mathsf{mat}\,(\boldsymbol{T_p}[\psi(k(\boldsymbol{a},\boldsymbol{b}),k(\cdot,\cdot))])]_{ij} \\
&= \iint \psi(k(\boldsymbol{a},\boldsymbol{b}),k(\boldsymbol{x},\boldsymbol{z})) \left(\sum_{l=1}^{M} s_{il}\delta(\boldsymbol{x}-\boldsymbol{u}_l)\right)\left(\sum_{l=1}^{M} s_{jl}\delta(\boldsymbol{z}-\boldsymbol{u}_l)\right) \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{z} \\
&= \iint \frac{1}{2}(w(\boldsymbol{a},\boldsymbol{x})w(\boldsymbol{b},\boldsymbol{z}) + w(\boldsymbol{a},\boldsymbol{z})w(\boldsymbol{b},\boldsymbol{x})) \left(\sum_{l=1}^{M} s_{il}\delta(\boldsymbol{x}-\boldsymbol{u}_l)\right)\left(\sum_{l=1}^{M} s_{jl}\delta(\boldsymbol{z}-\boldsymbol{u}_l)\right) \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{z} \\
&= \frac{1}{2}\sum_{m=1}^{M}\sum_{l=1}^{M} \boldsymbol{S}_{im}\boldsymbol{S}_{jl}(w(\boldsymbol{a},\boldsymbol{u}_m)w(\boldsymbol{b},\boldsymbol{u}_l) + w(\boldsymbol{a},\boldsymbol{u}_l)w(\boldsymbol{b},\boldsymbol{u}_m)) \\
&= \frac{1}{2}\,[\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{a})w(\boldsymbol{b},\boldsymbol{X}_U)\boldsymbol{S} + \boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{b})w(\boldsymbol{a},\boldsymbol{X}_U)\boldsymbol{S}]_{ij} \\
&= \frac{1}{2}\,[\boldsymbol{S}^\intercal (w(\boldsymbol{X}_U,\boldsymbol{a})w(\boldsymbol{b},\boldsymbol{X}_U) + w(\boldsymbol{X}_U,\boldsymbol{b})w(\boldsymbol{a},\boldsymbol{X}_U))\boldsymbol{S}]_{ij}
\end{aligned}
$$

which shows Eq. (65)

$$
\begin{aligned}
&= [\mathsf{mat}\,((\boldsymbol{S}^\intercal \otimes \boldsymbol{S}^\intercal)\boldsymbol{\Gamma}\,\mathsf{vec}\,(w(\boldsymbol{X}_U,\boldsymbol{a})w(\boldsymbol{b},\boldsymbol{X}_U)))]_{ij} \\
&= [\mathsf{mat}\,((\boldsymbol{S}^\intercal \otimes \boldsymbol{S}^\intercal)\boldsymbol{\Gamma}(w(\boldsymbol{X}_U,\boldsymbol{a}) \otimes w(\boldsymbol{X}_U,\boldsymbol{b})))]_{ij}
\end{aligned}
$$

Repeating above derivations shows the second statement:

$$
\begin{aligned}
\boldsymbol{T_p}(\boldsymbol{T_p}\psi)^\intercal &= (\boldsymbol{S}^\intercal \otimes \boldsymbol{S}^\intercal)\boldsymbol{\Gamma}(w(\boldsymbol{X}_U,\boldsymbol{X}_U) \otimes w(\boldsymbol{X}_U,\boldsymbol{X}_U))\boldsymbol{\Gamma}(\boldsymbol{S}\otimes\boldsymbol{S}) \\
&= (\boldsymbol{S}\otimes\boldsymbol{S})^\intercal(w(\boldsymbol{X}_U,\boldsymbol{X}_U)\circledast w(\boldsymbol{X}_U,\boldsymbol{X}_U))(\boldsymbol{S}\otimes\boldsymbol{S}) \\
&= (\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})\circledast(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S}) \\
&\quad /\!/ \textit{ Eq. } (\text{SK4})
\end{aligned}
$$

□

E.2.1 Positive Semi-definiteness of the Approximate Kernel

This section shows that the KMCG kernel (Eq. 34) is always positive semi-definite.

Proposition 43. *If $k_0 = 0$, $\boldsymbol{S}$ has rank M and k and w are positive definite kernel functions then the posterior mean in Equation (63) is symmetric and positive semi-definite.*

Proof. With $k_0 = 0$ the expression for k_M simplifies to

$$k_M(\boldsymbol{x},\boldsymbol{z}) = w(\boldsymbol{x},\boldsymbol{X}_U)\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal \boldsymbol{K}_M \boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal w(\boldsymbol{X}_U,\boldsymbol{z})$$

The function k_M is symmetric since k is symmetric. The bivariate function k_M is said to be positive (semi-)definite iff for all $n \in \mathbb{N}$ and for all $\boldsymbol{Z} \in \mathbb{X}$, $k_M(\boldsymbol{Z},\boldsymbol{Z})$ is a positive (semi-)definite matrix. Since $k(\boldsymbol{X}_U,\boldsymbol{X}_U)$ is an s.p.d. matrix, so is $\boldsymbol{S}^\intercal k(\boldsymbol{X}_U,\boldsymbol{X}_U)\boldsymbol{S}$ for arbitrary $\boldsymbol{S}$. The same argument holds for $\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S}$. Since $\boldsymbol{S}$ is rank M, $(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}$ exists and the inverse of an s.p.d. matrix is s.p.d. as well. Therefore $\boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}^\intercal \boldsymbol{K}_M \boldsymbol{S}(\boldsymbol{S}^\intercal \boldsymbol{W}_M \boldsymbol{S})^{-1}\boldsymbol{S}$ is symmetric and positive semi-definite. This completes the proof. □

E.3 Additional Experiments and Results

This section consists of figures showing the results of the experiments-section (Section 13) for the Matérn kernel, real-time experiments and experiments with the textbook version of conjugate gradients. All figures have been trimmed to the slowest baseline method.

E.3.1 Real-time Results

This section shows the same results as in Section 13 but over training-time instead of CG-steps. Figure 15 shows how the relative error ϵ_f develops over time for the squared exponential kernel and Figure 16 shows the same for experiments over grid-structured datasets from Section 13.2. For the x-axis values we took the median of all measurements and fitted a quadratic function to these.

E.3.2 Matérn Kernel Results

The figures in this section show the results for the Matérn $5/2$ kernel (Eq. (44)) for the experiment setup described Section 13. Figure 17 shows the results for the relative error ϵ_f, Figure 18 and Figure 19 the results for ϵ_{var} and ϵ_{ev}, respectively. Figure 20 displays the relative error over time.

E.3.3 Instability of Textbook Conjugate Gradients

The experiments in Section 13, where carried out by running conjugate gradients with full reorthogonalization. Figure 21 demonstrates that for the problems under consideration, the textbook version of conjugate gradients is not sufficiently numerically stable. With vanilla conjugate gradients in the background, KMCG can run only for a couple of steps before the necessary Cholesky decompositions fail to be computable.

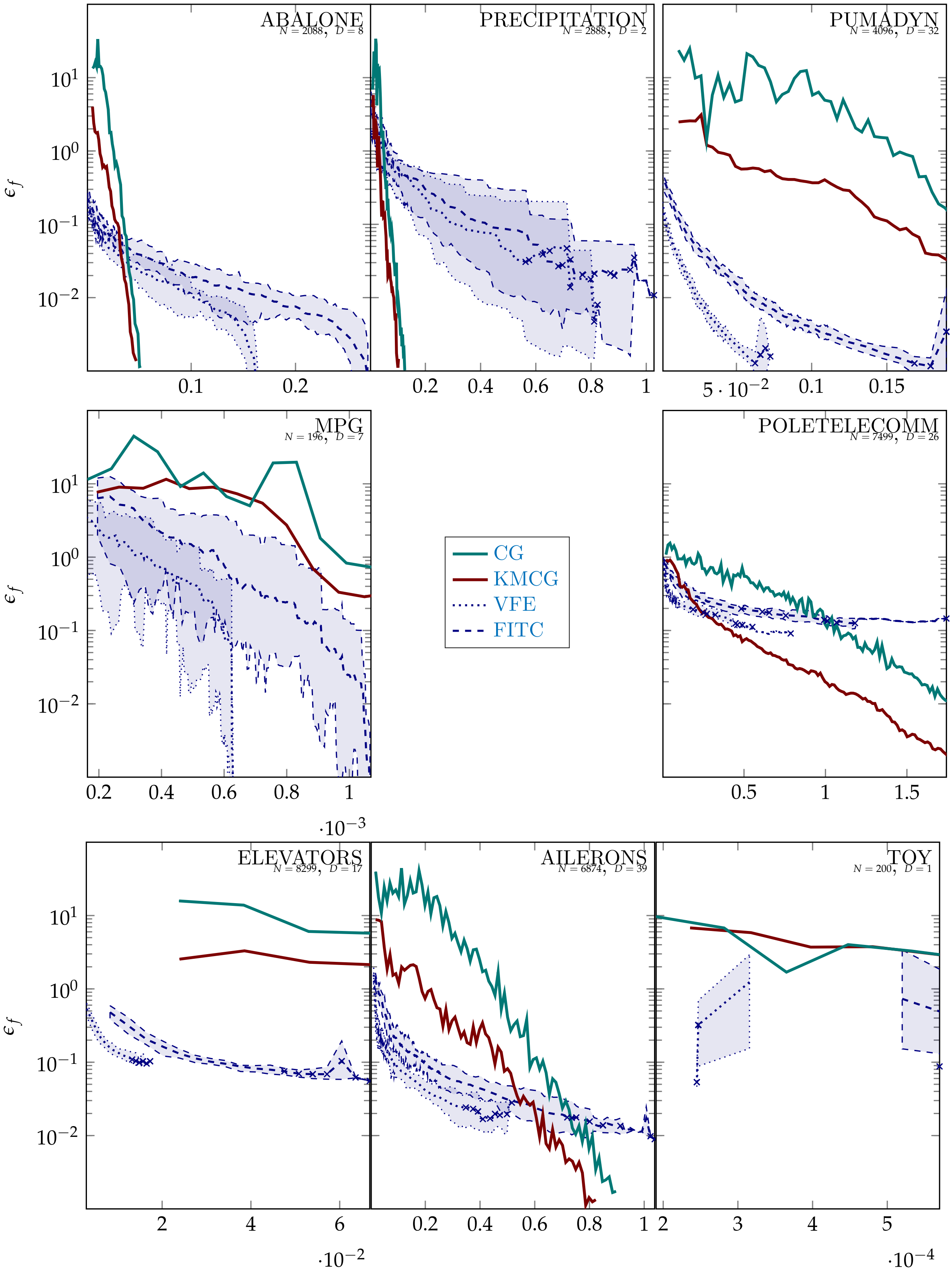


Figure 15: progression of the relative error ϵ_f over training time for different datasets using the squared-exponential kernel (Eq. 43). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

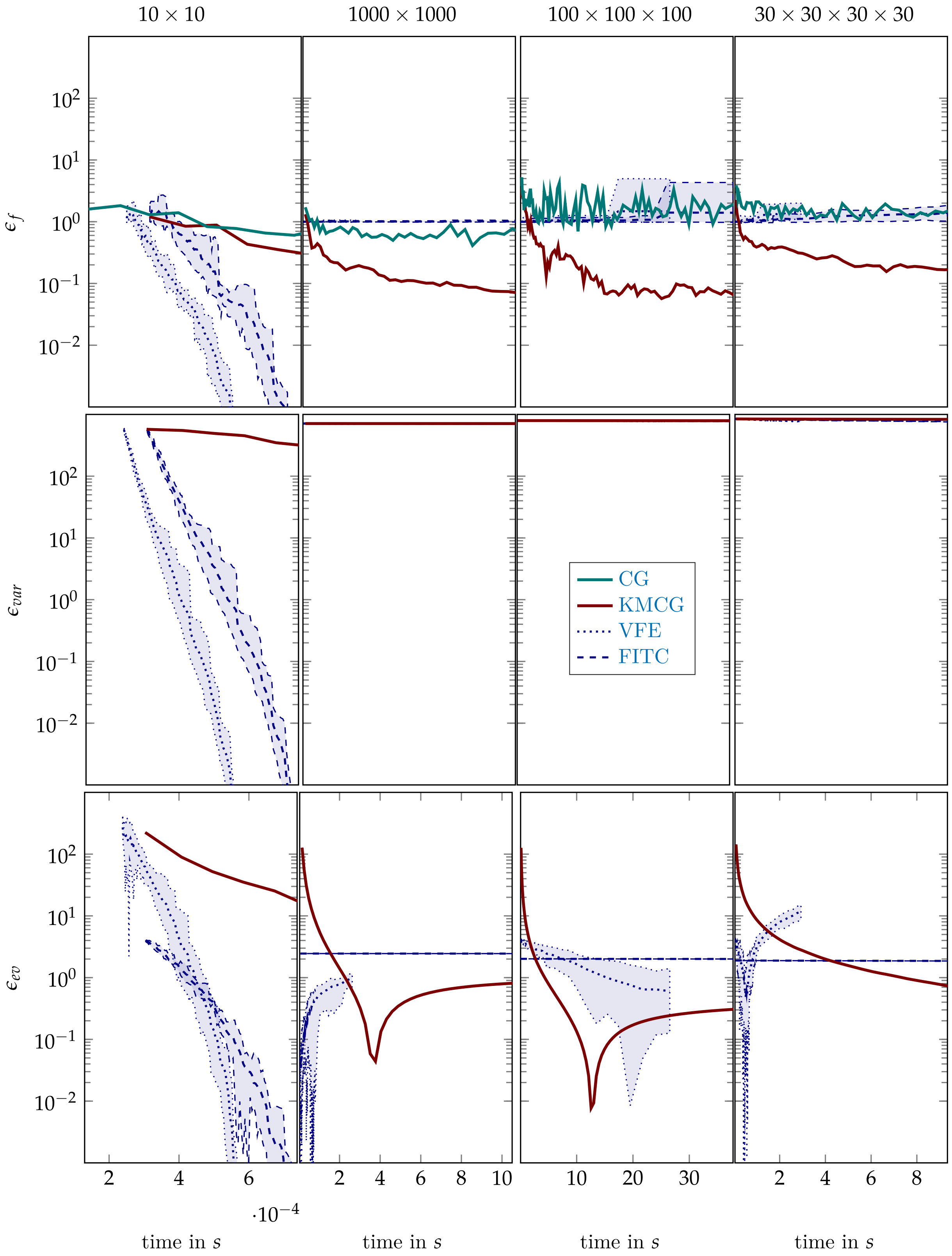


Figure 16: comparison of baseline and KMCG on grid-structured datasets using the squared exponential kernel (Eq. 43). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run. It may seem surprising that the runs on the 100 × 100 × 100 dataset take more than twice as long. By chance the dataset contains more extreme values in the kernel matrix, *i.e.* smaller than $1e^{-50}$. Multiplication with these elements takes more time.

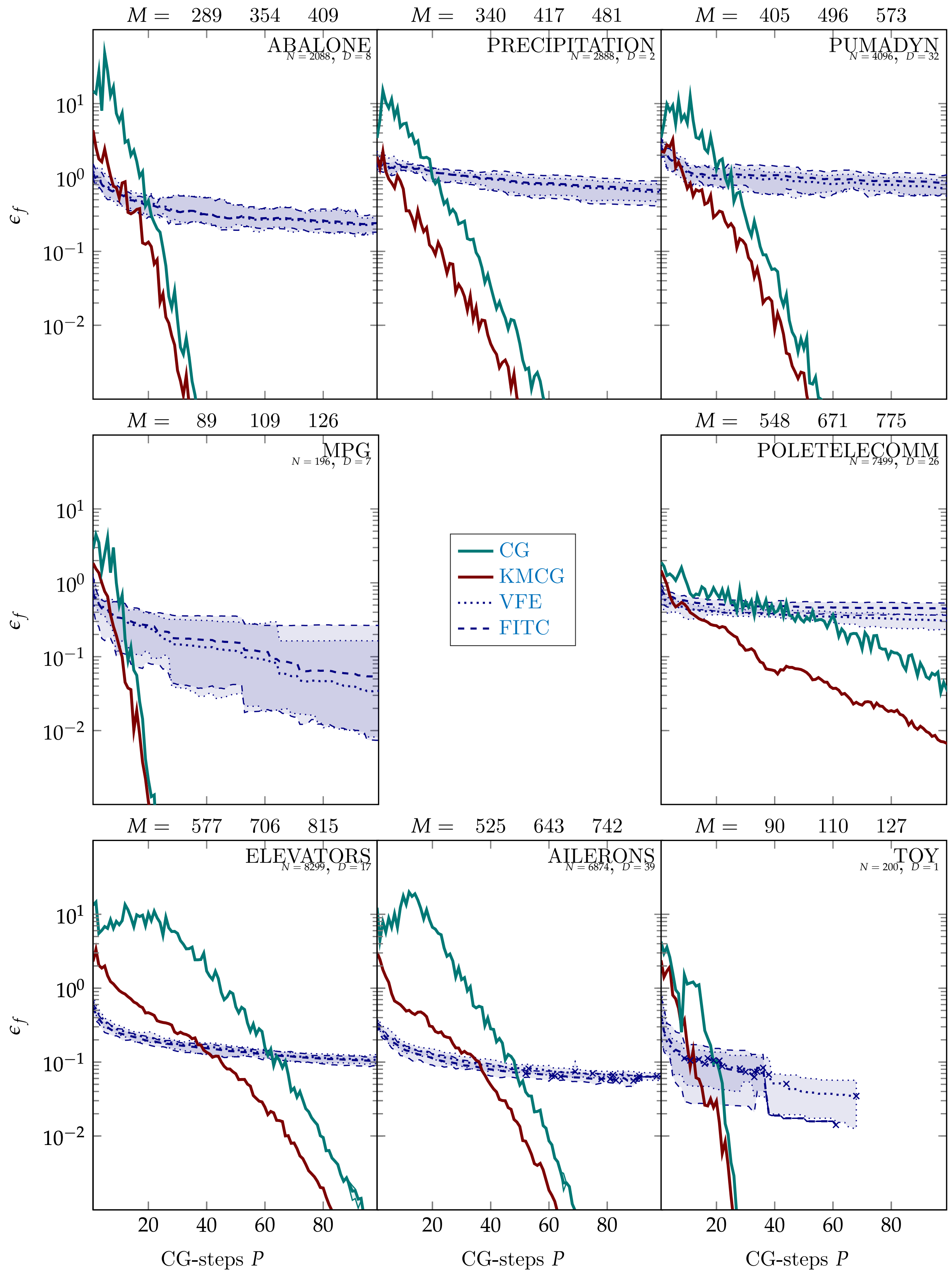


Figure 17: progression of the relative error ϵ_f as a function of the number of iterations of CG and KMCG for different datasets using the Matérn kernel (Eq. 44). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

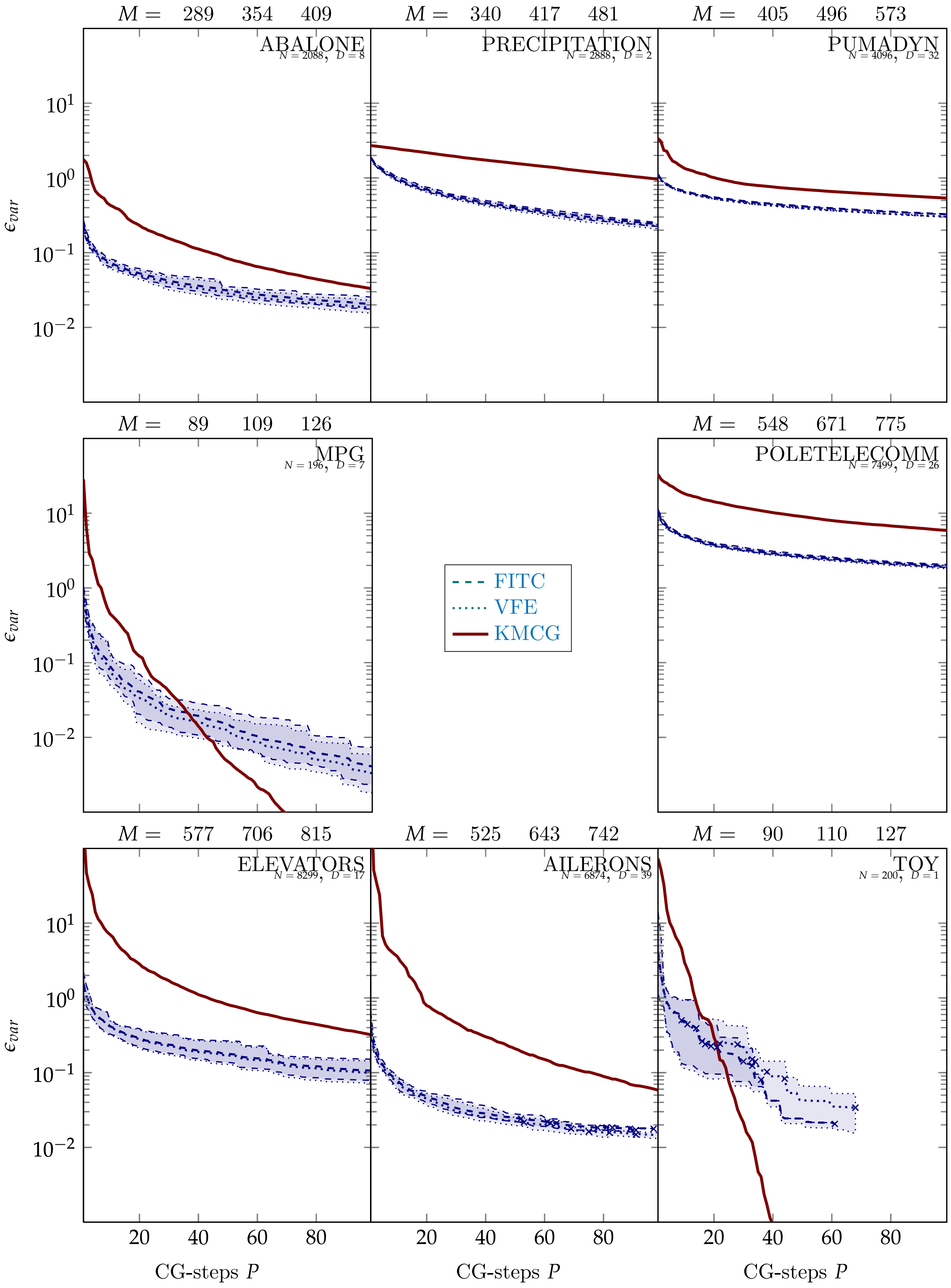


Figure 18: progression of the relative error of the variance as a function of the number of iterations of KMCG and baseline for different datasets using the Matérn kernel (Eq. 44). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

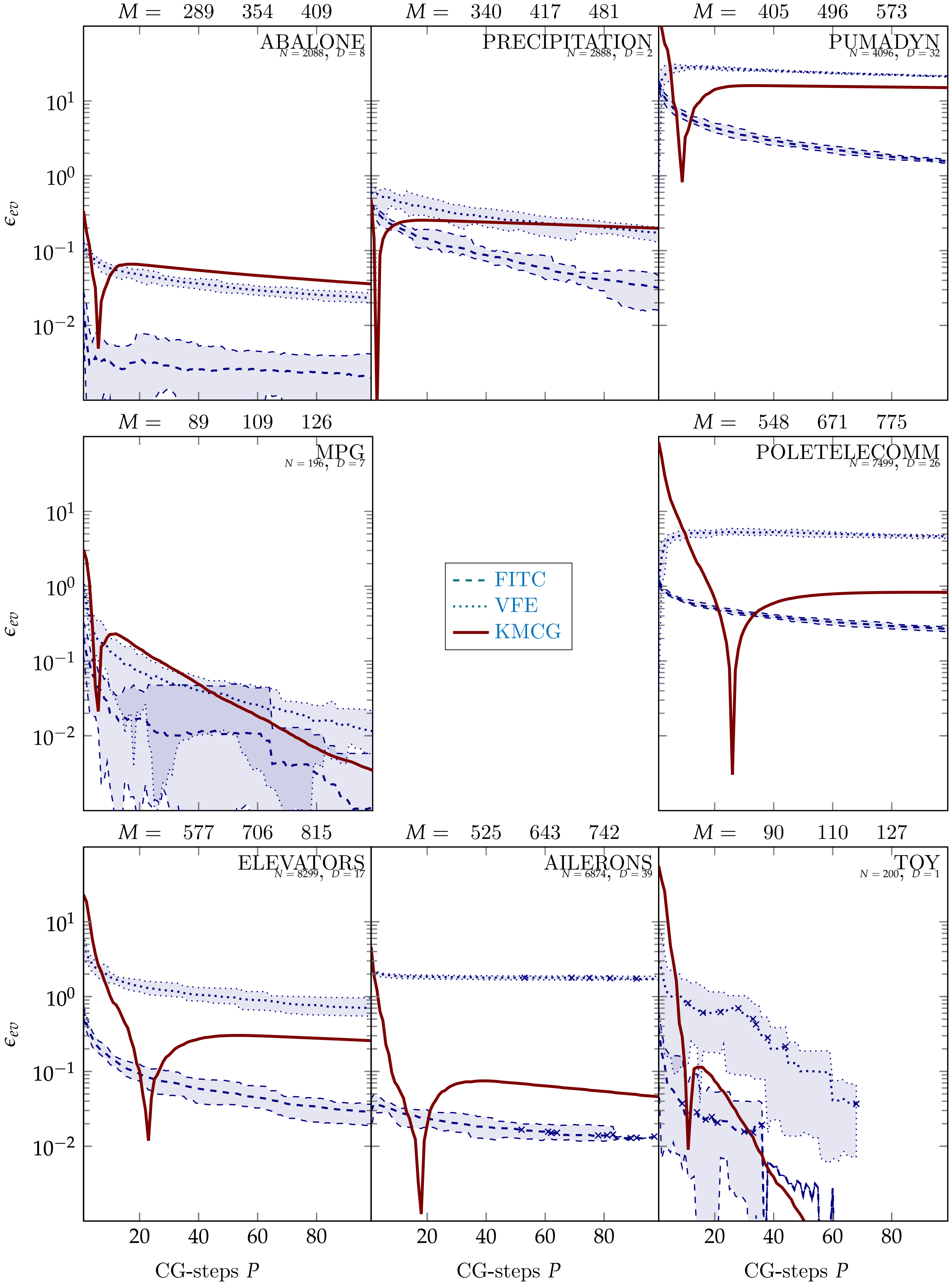


Figure 19: progression of the relative error of the evidence as a function of the number of iterations of baseline and KMCG for different datasets using the Matérn kernel (Eq. 44). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

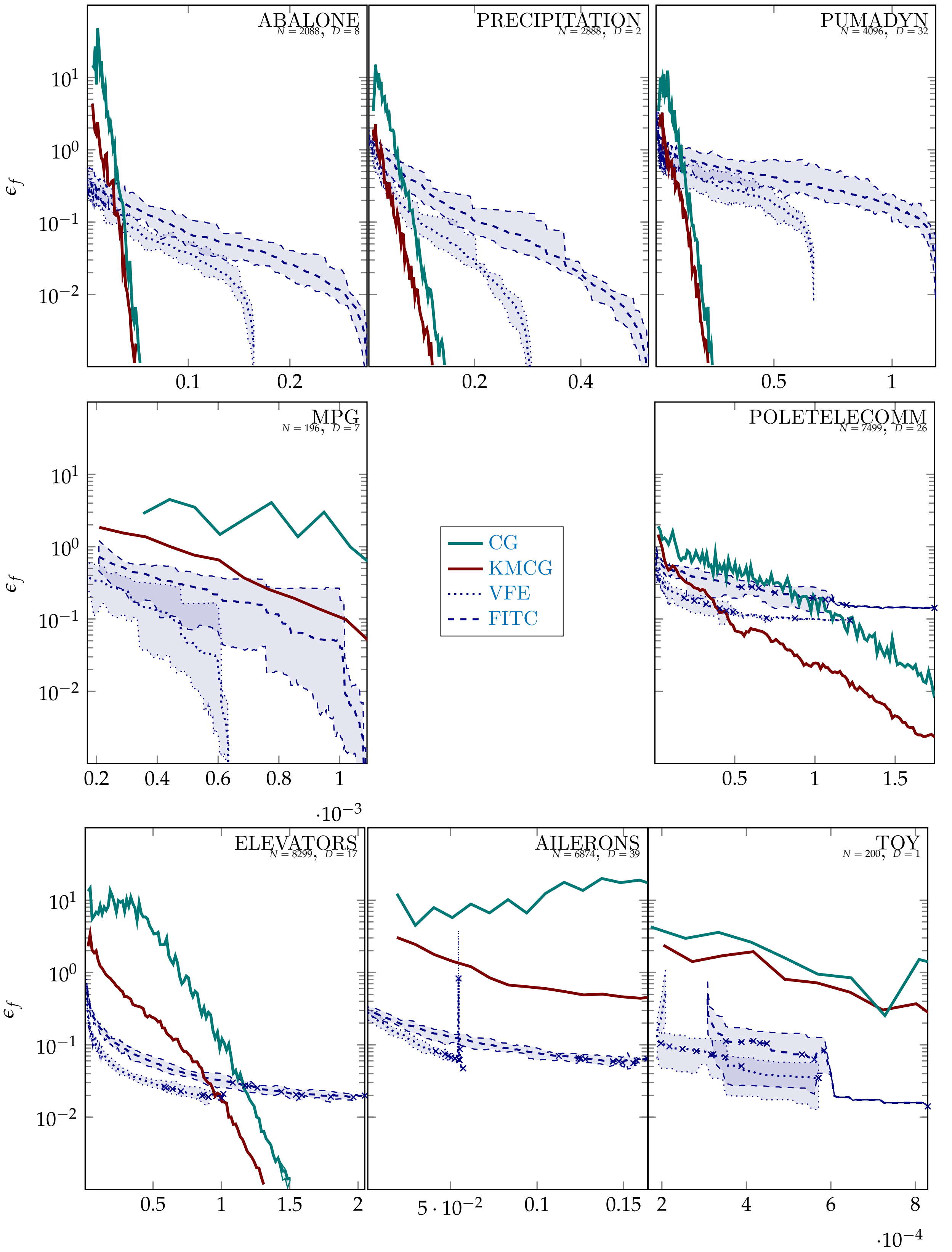


Figure 20: progression of the relative error ϵ_f over training time for different datasets using the Matérn 5/2 kernel (Eq. 44). The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

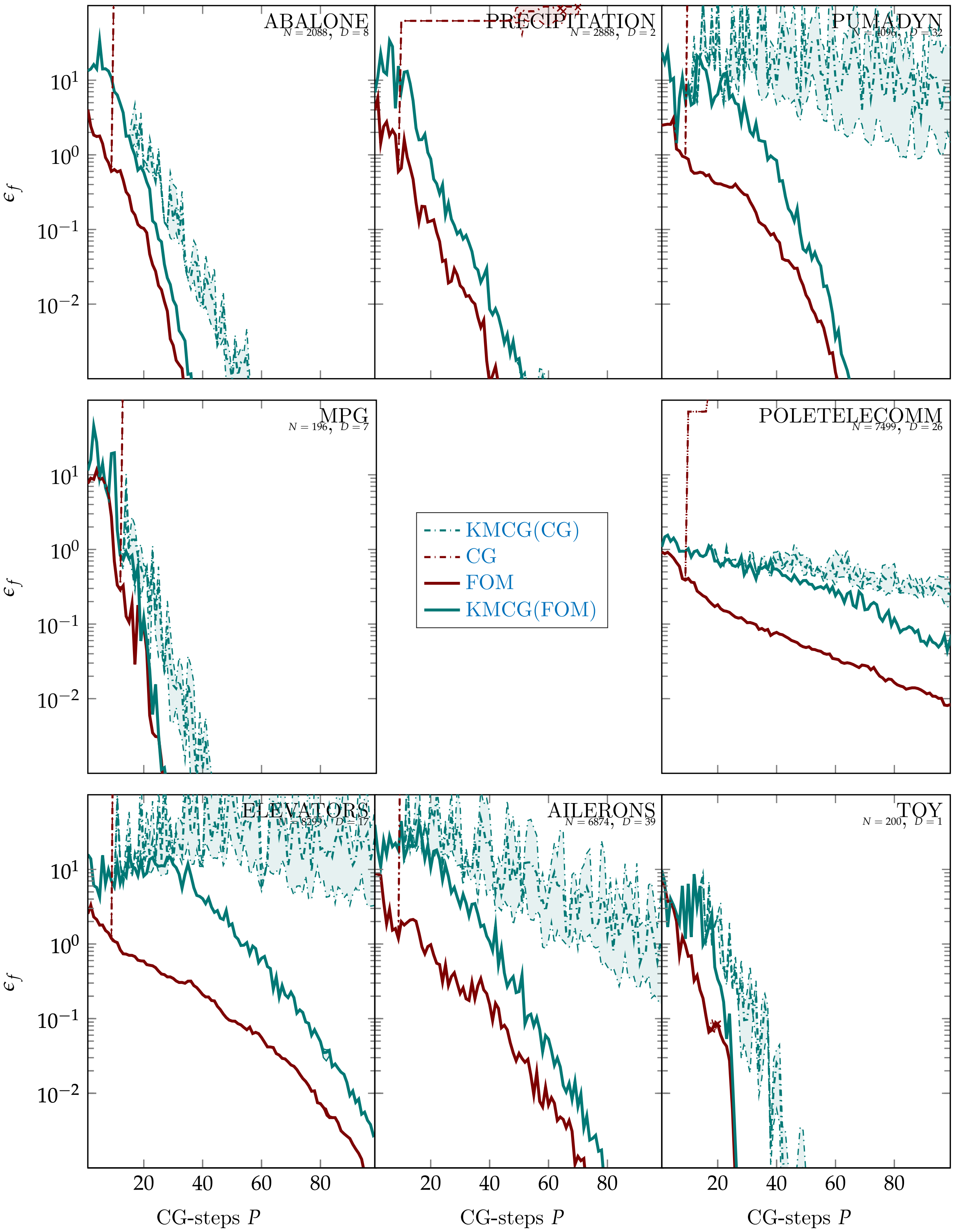


Figure 21: progression of the relative error ϵ_f over 100 CG-steps for different datasets using the squared exponential kernel (Eq. 43), comparing CG and FOM. The shaded area visualizes minimum and maximum over all baseline runs. A cross denotes the end of a crashed run.

F Additional Material for Part v

Theorem 44 (Popoviciu (1935) and Sharma, Gupta, and Kapoor (2010))**.** *For a sequence of real numbers $x_1, ..., x_n \in [m, M]$, define $\mu := \frac{1}{n}\sum_{j=1}^{n} x_j$ and $\sigma^2 := \frac{1}{n}\sum_{j=1}^{N}(x_j - \mu)^2$, then*

$$\sigma^2 \leq 1/4(M-m)^2.$$

Remark 45. *Above theorem can be used for a bound on the (conditional) variance as well. Let $x_1, ..., x_n \sim P(\cdot \mid \mathcal{F})$ be independent. Then,*

$$\begin{aligned}\mathbb{V}[X \mid \mathcal{F}] &= \mathbb{E}[(X - \mathbb{E}[X \mid \mathcal{F}])^2 \mid \mathcal{F}] \\ &= \frac{n}{n-1}\mathbb{E}[\sigma^2 \mid \mathcal{F}] \\ &\quad /\!/ \text{ using Bessel's correction} \\ &\leq \frac{n}{4(n-1)}(M-m)^2\end{aligned}$$

which holds for all $n \in \mathbb{N}$. Hence, $\mathbb{V}[X \mid \mathcal{F}] \leq 1/4(M-m)^2$.

Theorem 46 (Doob's Optional Sampling Theorem (Grimmett and Stirzaker 2001, p. 489))**.** *Let $(X_j, \mathcal{F}_j)_{j\in\mathbb{N}}$ be a submartingale and $\tau_1 \leq \tau_2 \leq ...$ be a sequence of stopping times s.t. $P(\tau_j \leq n_j) = 1$ for some deterministic real sequence n_j, then the stopped process $(X_{\tau_j}, \mathcal{F}_{\tau_j})_{j\in\mathbb{N}}$ is also a submartingale.*

Remark 47. *By exchanging X_j for $-X_j$ the theorem can be shown to hold for supermartingales as well.*

Corollary 48. *Let $(\xi_j, \mathcal{F}_j)_{j\in\mathbb{N}}$ be a submartingale-difference and let τ be a stopping time, then the stopped process $(\xi_{\min(j,\tau)}, \mathcal{F}_{\min(j,\tau)})_{j\in\mathbb{N}}$ is also a submartingale-difference.*

Proof. Define $X_l := \sum_{j=1}^{l} \xi_j$ and observe that this defines a submartingale. By Theorem 46 $(X_{\min(j,\tau)}, \mathcal{F}_{\min(j,\tau)})_{j\in\mathbb{N}}$ is a submartingale. Then $X_{\min(j,\tau)} - X_{\min(j,\tau)-1} = \xi_{\min(j,\tau)}$ is again a submartingale-difference. □

The following lemma appaers as well in Mnih (2008). It was relieving to see that other researchers have the same ideas and I took this as a sign to be on the right track.

Lemma 25 (Bound on Relative Error)**.** *Let $D, \hat{D} \in [\mathcal{L}, \mathcal{U}]$, and assume $\mathsf{sign}(\mathcal{L}) = \mathsf{sign}(\mathcal{U}) \neq 0$. Then the relative error of the estimator $\hat{D}$ can be bounded as*

$$\frac{\mathsf{abs}(D - \hat{D})}{\mathsf{abs}(D)} \leq \frac{\max(\mathcal{U} - \hat{D}, \hat{D} - \mathcal{L})}{\min(\mathsf{abs}(\mathcal{L}), \mathsf{abs}(\mathcal{U}))}.$$

Proof. First observe that if $D_N > \hat{D}$ then $\mathsf{abs}(D_N - \hat{D}) = D_N - \hat{D} \leq \mathcal{U} - \hat{D}$. If not, then $\mathsf{abs}(D_N - \hat{D}) = \hat{D} - D_N \leq \hat{D} - \mathcal{L}$. Hence,

$$\mathsf{abs}(D_N - \hat{D}) \leq \max(\mathcal{U} - \hat{D}, \hat{D} - \mathcal{L}).$$

Case $\mathcal{L} > 0$: This implies $\mathsf{abs}(D_N) = D_N$.

$$\frac{\mathsf{abs}(D_N - \hat{D})}{\mathsf{abs}(D_N)} \leq \frac{\mathsf{abs}(D_N - \hat{D})}{\mathcal{L}}$$

Case $\mathcal{U} < 0$: In that case $\mathsf{abs}(\mathcal{L}) \geq \mathsf{abs}(D_N) \geq \mathsf{abs}(\mathcal{U})$.

$$\frac{\mathsf{abs}(D_N - \hat{D})}{\mathsf{abs}(D_N)} \leq \frac{\mathsf{abs}(D_N - \hat{D})}{\mathsf{abs}(\mathcal{U})}$$

Since we assumed $\mathsf{sign}(\mathcal{L}) = \mathsf{sign}(\mathcal{U})$ these were all cases that required consideration. Note that, in the first case $1/\mathcal{L} \geq 1/\mathsf{abs}(\mathcal{U})$, and in the second case, the $1/\mathcal{L} \leq 1/\mathsf{abs}(\mathcal{U})$, since $\mathcal{L} < 0$. Combining all observations gives

$$\frac{\mathsf{abs}(D_N - \hat{D})}{\mathsf{abs}(D_N)} \leq \max(\mathcal{U} - \hat{D}, \hat{D} - \mathcal{L}) \max\left(\frac{1}{\mathsf{abs}(\mathcal{U})}, \frac{1}{\mathcal{L}}\right)$$

If $\mathcal{L} > 0$, $\max(\frac{1}{\mathsf{abs}(\mathcal{U})}, \frac{1}{\mathcal{L}}) = \min(\mathsf{abs}(\mathcal{U}), \mathcal{L}) = \min(\mathsf{abs}(\mathcal{U}), \mathsf{abs}(\mathcal{L}))$. In the other case, $\mathcal{U} < 0$, we can write

$$\max\left(\frac{1}{\mathsf{abs}(\mathcal{U})}, \frac{1}{\mathcal{L}}\right) = \max\left(\frac{1}{\mathsf{abs}(\mathcal{U})}, \frac{1}{\mathsf{abs}(\mathcal{L})}\right) = \min(\mathsf{abs}(\mathcal{U}), \mathsf{abs}(\mathcal{L})).$$

□

Lemma 49. *Denote with $\boldsymbol{C}$ the Cholesky decomposition of a symmetric and positive definite matrix $\boldsymbol{K}$. The log-determinant of $\boldsymbol{K}$ equals two times the sum over the logarithm of the diagonal elements of the Cholesky decomposition $\boldsymbol{C}$:*

$$\ln|\boldsymbol{K}| = 2\sum_{j=1}^{N} \ln \boldsymbol{C}_{jj}.$$

Proof.

$$\begin{aligned}
\ln|\boldsymbol{K}| &= \ln|\boldsymbol{C}\boldsymbol{C}^\intercal| \\
&\text{// \textit{using} } \boldsymbol{K} = \boldsymbol{C}\boldsymbol{C}^\intercal \\
&= \ln(|\boldsymbol{C}| \cdot |\boldsymbol{C}^\intercal|) \\
&\text{// \textit{property of the determinant}} \\
&= \ln(|\boldsymbol{C}|^2) \\
&\text{// \textit{transposition does not affect the determinant}} \\
&= \ln(\prod_{j=1}^{N} \boldsymbol{C}_{jj})^2 \\
&\text{// \textit{property of triangular matrices}} \\
&= 2\sum_{j=1}^{N} \ln \boldsymbol{C}_{jj} \\
&\text{// \textit{property of the logarithm}}
\end{aligned}$$

□

Bibliography

Arras, P., J. Knollmüller, H. Junklewitz, and T. A. Enßlin (2018). “Radio Imaging with Information Field Theory.” In: *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2683–2687.

Bardenet, R., A. Doucet, and C. Holmes (2014). “Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach.” In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by E. P. Xing and T. Jebara. Vol. 32. Proceedings of Machine Learning Research 1, pp. 405–413.

Bartels, S., J. Cockayne, I. C. F. Ipsen, and P. Hennig (2019). “Probabilistic Linear Solvers: A Unifying View.” In: *Statistics and Computing* 29.6, pp. 1249–1263.

Bartels, S. and P. Hennig (2016). “Probabilistic Approximate Least-Squares.” In: *Proceedings of Artificial Intelligence and Statistics (AISTATS)*.

– (2019). “Conjugate Gradients for Kernel Machines.” In: *ArXiv e-prints* 1911.06048. arXiv:1911.06048.

Benoit (1924). “Note sûre une méthode de résolution des équations normales provenant de l’application de la méthode des moindres carrés a un système d’équations linéaires en nombre inférieure a celui des inconnues. Application de la méthode a la résolution d’un système défini d’équations linéaires. (Procédé du Commandant Cholesky).” In: *Bulletin Geodesique* 7.1, pp. 67–77.

Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Springer.

Boucheron, S., G. Lugosi, and P. Massart (2013). *Concentration Inequalities: A Nonasymptotic Theory of Independence*. 1st. Oxford University Press.

Boutsidis, C., P. Drineas, P. Kambadur, E.-M. Kontopoulou, and A. Zouzias (2017). “A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix.” In: *Linear Algebra and its Applications* 533, pp. 95 –117.

Camachol, R. (1998). “Inducing models of human control skills.” In: *Machine Learning: ECML-98*. Ed. by C. Nédellec and C. Rouveirol, pp. 107–118.

Chalupka, K., C. K. I. Williams, and I. Murray (2013). “A Framework for Evaluating Approximation Methods for Gaussian Process Regression.” In: *Journal of Machine Learning Research* 14.1, pp. 333–350.

Cockayne, J., C. Oates, I. C. F. Ipsen, and M. Girolami (2018). “A Bayesian Conjugate Gradient Method.” In: *ArXiv e-prints* 1801.05242. arXiv:1801.05242.

Cockayne, J., C. Oates, T. J. Sullivan, and M. Girolami (2016). “Probabilistic Numerical Methods for Partial Differential Equations and Bayesian Inverse Problems.” In: *ArXiv e-prints* 1605.07811. arXiv:1605.07811.

– (2017). “Bayesian Probabilistic Numerical Methods.” In: *ArXiv e-prints* 1702.03673. arXiv:1702.03673.

Csató, L. and M. Opper (2002). “Sparse On-line Gaussian Processes.” In: *Neural Computation* 14.3, pp. 641–668.

Davidson, J. (1994). *Stochastic Limit Theory: An Introduction for Econometricians*. Oxford University Press.

Davies, A. (2015). “Effective implementation of Gaussian process regression for machine learning.” PhD thesis. University of Cambridge.

DeGroot, M. H. and M. J. Schervish (2012). *Probability and Statistics.* 4th. Pearson.

Diaconis, P. and M. Shahshahani (1987). “The Subgroup Algorithm for Generating Uniform Random Variables.” In: *Probability in the Engineering and Informational Sciences* 1.01, p. 15. DOI: 10.1017/s0269964800000255.

Dong, K., D. Eriksson, H. Nickisch, D. Bindel, and A. G. Wilson (2017). “Scalable Log Determinants for Gaussian Process Kernel Learning.” In: *Advances in Neural Information Processing Systems*, pp. 6330–6340.

Dorn, S. and T. A. Enßlin (2015). “Stochastic determination of matrix determinants.” In: *Physical Review E* 92 (1), p. 013302.

Dua, D. and C. Graff (2019). *UCI Machine Learning Repository.* URL: http://archive.ics.uci.edu/ml.

Fan, X., I. Grama, and Q. Liu (2012). “Hoeffding’s inequality for supermartingales.” In: *Stochastic Processes and their Applications* 122.10, pp. 3545–3559.

Filippone, M. and R. Engler (2015). “Enabling scalable stochastic gradient-based inference for Gaussian processes by employing the Unbiased LInear System SolvEr (ULISSE).” In: *Proceedings of the 32nd International Conference on Machine Learning.* Lille, France, pp. 1015–1024.

Fitzsimons, J., K. Cutajar, M. Osborne, S. Roberts, and M. Filippone (2017). “Bayesian Inference of Log Determinants.” In: *Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, August 11-15, 2017, Sydney, Australia.* Ed. by G. Elidan, K. Kersting, and A. T. Ihler.

Fitzsimons, J., D. Granziol, K. Cutajar, M. Osborne, M. Filippone, and S. Roberts (2017). “Entropic Trace Estimates for Log Determinants.” In: *Machine Learning and Knowledge Discovery in Databases.* Ed. by M. Ceci, J. Hollmén, L. Todorovski, C. Vens, and S. Džeroski, pp. 323–338.

Geman, S. and D. Geman (1984). “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6, pp. 721–741.

Golub, G. and C. Van Loan (2013). *Matrix computations.* 4th ed. Johns Hopkins Univ Pr.

Graf, F., H.-P. Kriegel, M. Schubert, S. Pölsterl, and A. Cavallaro (2011). “2D Image Registration in CT Images Using Radial Image Descriptors.” In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2011.* Ed. by G. Fichtinger, A. Martel, and T. Peters, pp. 607–614.

Grimmett, G. and D. Stirzaker (2001). *Probability and Random Processes.* 3rd. Oxford University Press.

Gut, A. (2009). *An Intermediate Course in Probability.* 2nd. Springer.

Hennig, P. (2015). “Probabilistic Interpretation of Linear Solvers.” In: *SIAM J on Optimization* 25.1, pp. 210–233.

Hennig, P. and M. Kiefel (2012). “Quasi-Newton methods – a new direction.” In: *International Conference on Machine Learning (ICML).*

– (2013). “Quasi-Newton Methods – a new direction.” In: *Journal of Machine Learning Research* 14, pp. 834–865.

Hennig, P., M. Osborne, and M. Girolami (2015). “Probabilistic numerics and uncertainty in computations.” In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 471.2179.

Hensman, J., N. Durrande, and A. Solin (2018). “Variational Fourier Features for Gaussian Processes.” In: *Journal of Machine Learning Research* 18.151, pp. 1–52.

Hestenes, M. and E. Stiefel (1952). “Methods of conjugate gradients for solving linear systems.” In: *Journal of Research of the National Bureau of Standards* 49.6, pp. 409–436.

Hoerl, A. E. and R. W. Kennard (1970). “Ridge regression: Biased estimation for nonorthogonal problems.” In: *Technometrics* 12.1, pp. 55–67.

Karvonen, T. and S. Sarkka (2017). “Classical quadrature rules via Gaussian processes.” In: *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE.

Kersting, H., T. J. Sullivan, and P. Hennig (2018). “Convergence Rates of Gaussian ODE Filters.” In: *ArXiv e-prints* 1807.09737. arXiv:1807.09737.

Kimeldorf, G. S. and G. Wahba (1970). “A correspondence between Bayesian estimation on stochastic processes and smoothing by splines.” In: *The Annals of Mathematical Statistics*, pp. 495–502.

Lázaro-Gredilla, M., J. Quiñonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal (2010). “Sparse Spectrum Gaussian Process Regression.” In: *Journal of Machine Learning Research* 11, pp. 1865–1881.

Le, Q., T. Sarlos, and A. Smola (2013). “Fastfood - Computing Hilbert Space Expansions in loglinear time.” In: *Proceedings of the 28th International Conference on Machine Learning*, pp. 244–252.

Loan, C. F. V. (2000). “The ubiquitous Kronecker product.” In: *Journal of Computational and Applied Mathematics* 123.1. Numerical Analysis 2000. Vol. III: Linear Algebra, pp. 85 –100.

Magnus, J. R. and H. Neudecker (1980). “The Elimination Matrix: Some Lemmas and Applications.” In: *SIAM Journal on Algebraic Discrete Methods* 1.4, pp. 422–449. DOI: 10.1137/0601049.

Magnus, J. R. and H. Neudecker (1999). *Matrix Differential Calculus with Applications in Statistics and Econometrics*. Second. John Wiley.

Matheron, G. (1973). “The intrinsic random functions and their applications.” In: *Advances in applied probability*, pp. 439–468.

Meister, A. (2015). *Numerik Linearer Gleichungssysteme*. 5th. Springer Fachmedien Wiesbaden.

Mnih, V. (2008). “Efficient stopping rules.” MA thesis. University of Alberta, Canada.

Mnih, V., C. Szepesvári, and J. Audibert (2008). “Empirical Bernstein stopping.” In: ed. by A. McCallum and S. Roweis, pp. 672–679.

Munroe, R. (2019). *Error Bars*. URL: https://xkcd.com/2110/. License: Creative Commons 2.5 BY-NC-SA.

Nash, W. J., T. L. Sellers, S. R. Talbot, A. J. Cawthorn, and W. B. Ford (1994). *The Population biology of abalone (Haliotis species) in Tasmania. 1, Blacklip abalone (H. rubra) from the north coast and the islands of Bass Strait*. Tech. rep. 48. Sea Fisheries Division, Marine Research Laboratories - Taroona, Department of Primary Industry and Fisheries, Tasmania. URL: https://trove.nla.gov.au/work/11326142.

Nille, D., U. von Toussaint, B. Sieglin, and M. Faitsch (2018). “Probabilistic Inference of Surface Heat Flux Densities from Infrared Thermography.” In: *Bayesian Inference and Maximum Entropy Methods in Science and Engineering*. Ed. by A. Polpo, J. Stern, F. Louzada, R. Izbicki, and H. Takada, pp. 55–64.

Nocedal, J. and S. Wright (1999). *Numerical Optimization*. Springer Verlag.

Pleiss, G., J. Gardner, K. Weinberger, and A. G. Wilson (2018). “Constant-Time Predictive Distributions for Gaussian Processes.” In: *Proceedings of the 35th International Conference on Machine Learning*, pp. 4114–4123.

Popoviciu, T. (1935). “Sur les equations algebriques ayanttoutes leurs racines reelles.” In: *Mathematica (Cluj)* 9, pp. 129–145.

Quinlan, J. R. (1993). “Combining Instance-based and Model-based Learning.” In: *Proceedings of the Tenth International Conference on International Conference on Machine Learning*. ICML’93, pp. 236–243.

Quiñonero-Candela, J. and C. E. Rasmussen (2005). “A unifying view of sparse approximate Gaussian process regression.” In: *Journal of Machine Learning Research* 6, pp. 1939–1959.

Rahimi, A. and B. Recht (2008). “Random Features for Large-Scale Kernel Machines.” In: *Advances in Neural Information Processing Systems 20*. Ed. by J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, pp. 1177–1184.

– (2009). “Weighted Sums of Random Kitchen Sinks: Replacing minimization with randomization in learning.” In: *Advances in Neural Information Processing Systems 23*, pp. 1313–1320.

Rasmussen, C. E. and H. Nickisch (2010). “Gaussian Processes for Machine Learning (GPML) Toolbox.” In: *Journal of Machine Learning Research* 11, pp. 3011–3015.

Rasmussen, C. and C. Williams (2006). *Gaussian Processes for Machine Learning*. MIT.

Roeckner, E., G. Bäuml, L. Bonaventura, R. Brokopf, M. Esch, and M. Giorgetta (2003). *The atmospheric general circulation model ECHAM 5. part I: Model description*. Tech. rep. MPI für Meteorologie.

Saad, Y. and M. H. Schultz (1986). “GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems.” In: *SIAM Journal on Scientific and Statistical Computing* 7.3, pp. 856–869.

Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems*. 2nd. Society for Industrial and Applied Mathematics.

Saibaba, A. K., A. Alexanderian, and I. C. F. Ipsen (2017). “Randomized matrix-free trace and log-determinant estimators.” In: *Numerische Mathematik* 137.2, pp. 353–395.

Schober, M., D. Duvenaud, and P. Hennig (2014). “Probabilistic ODE Solvers with Runge-Kutta Means.” In: *Advances in Neural Information Processing Systems 27*, pp. 739–747.

Schober, M., S. Särkkä, and P. Hennig (2018). “A probabilistic model for the numerical solution of initial value problems.” In: *Statistics and Computing*.

Schölkopf, B. and A. Smola (2002). *Learning with Kernels*. MIT Press.

Sharma, R., M. Gupta, and G Kapoor (2010). “Some better bounds on the variance with applications.” In: *Journal of Mathematical Inequalities* 4, pp. 355–363.

Skilling, J. (1989). “The Eigenvalues of Mega-dimensional Matrices.” In: *Maximum Entropy and Bayesian Methods: Cambridge, England, 1988*. Ed. by J. Skilling, pp. 455–466.

– (1993). “Bayesian Numerical Analysis.” In: *Physics and Probability: Essays in Honor of Edwin T. Jaynes.* Ed. by J. W. T. Grandy and P. W. Milonni, pp. 207–222.

Snelson, E. and Z. Ghahramani (2006). “Sparse Gaussian Processes using Pseudo-inputs.” In: *Advances in Neural Information Processing Systems 18.* Ed. by Y. Weiss, B. Schölkopf, and J. C. Platt, pp. 1257–1264.

– (2007). “Local and global sparse Gaussian process approximations.” In: *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, pp. 524–531.

Solin, A. and S. Särkkä (2014). “Hilbert Space Methods for Reduced-Rank Gaussian Process Regression.” In: *ArXiv e-prints* 1401.5508. arXiv:1401.5508.

Soodhalter, K. M., D. B. Szyld, and F. Xue (2014). “Krylov subspace recycling for sequences of shifted linear systems.” In: *Applied Numerical Mathematics* 81, pp. 105–118. DOI: 10.1016/j.apnum.2014.02.006. URL: https://doi.org/10.1016/j.apnum.2014.02.006.

Stark, P. B. and D. A. Freedman (2003). “What is the chance of an earthquake?” In: *Earthquake Science and Seismic Risk Reduction*, pp. 201–216.

Titsias, M. (2009). “Variational Learning of Inducing Variables in Sparse Gaussian Processes.” In: *Artificial Intelligence and Statistics (AISTATS), JMLR W&CP 5.*

Trecate, G. F., C. K. I. Williams, and M. Opper (1999). “Finite-dimensional Approximation of Gaussian Processes.” In: *Advances in Neural Information Processing Systems 2*, pp. 218–224.

Tronarp, F., H. Kersting, S. Särkkä, and P. Hennig (2019). “Probabilistic solutions to ordinary differential equations as nonlinear Bayesian filtering: a new perspective.” In: *Statistics and Computing* 29.6, pp. 1297–1315.

Turner, R. E. (2010). “Statistical Models for Natural Sounds.” PhD thesis. University College London.

Vanhatalo, J. and A. Vehtari (2008). “Modelling local and global phenomena with sparse Gaussian processes.” In: *UAI 2008, Twenty-Fourth Conference on Uncertainty in Artificial Intelligence, Helsinki, Finland, July 9-12, 2008.* Ed. by D. McAllester and P. Myllymäki.

Vijayakumar, S. and S. Schaal (2000). “Locally Weighted Projection Regression: An $\mathcal{O}(n)$ Algorithm for Incremental Real Time Learning in High Dimensional Space.” In: *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pp. 1079–1086.

Wahba, G. (1990). *Spline models for observational data.* CBMS-NSF Regional Conferences series in applied mathematics 59. SIAM.

Walder, C., K. I. Kim, and B. Schölkopf (2008). “Sparse Multiscale Gaussian Process Regression.” In: *Proceedings of the 25th International Conference on Machine Learning*, pp. 1112–1119.

Wang, Q., X. Zhang, Y. Zhang, and Q. Yi (2013). “AUGEM: Automatically Generate High Performance Dense Linear Algebra Kernels on x86 CPUs.” In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis.* SC ’13, 25:1–25:12.

Waugh, S. (1995). “Extending and benchmarking Cascade-Correlation.” PhD thesis. University of Tasmania.

Weiss, S. M. and N. Indurkhya (1995). “Rule-based Machine Learning Methods for Functional Prediction.” In: *Journal of Artificial Intelligence Research* 3.1, pp. 383–403.

Welling, M. and Y. W. Teh (2011). "Bayesian Learning via Stochastic Gradient Langevin Dynamics." In: *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 681–688.

Williams, C. and M. Seeger (2001). "Using the Nyström Method to Speed Up Kernel Machines." In: *Advances in Neural Information Processing Systems 13*.

Wilson, A. and H. Nickisch (2015). "Kernel Interpolation for Scalable Structured Gaussian Processes (KISS-GP)." In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by F. Bach and D. Blei. Vol. 37. Proceedings of Machine Learning Research, pp. 1775–1784.

Wilson, A. G., E. Gilboa, A. Nehorai, and J. P. Cunningham (2014). "Fast Kernel Learning for Multidimensional Pattern Extrapolation." In: *Advances in Neural Information Processing Systems 27*, pp. 3626–3634.

Yan, F. and Y. Qi (2010). "Sparse Gaussian Process Regression via l1 Penalization." In: *Proceedings of the 27th International Conference on Machine Learning*, pp. 1183–1190.

Zhao, S., E. Zhou, A. Sabharwal, and S. Ermon (2016). "Adaptive Concentration Inequalities for Sequential Decision Problems." In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, pp. 1343–1351.

Zhu, H., C. K. I. Williams, R. J. Rohwer, and M. Morciniec (1998). "Gaussian regression and optimal finite dimensional linear models." In: *Neural Networks and Machine Learning*.