Numerical Integration as and for Probabilistic Inference

Dissertation

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

vorgelegt von

Alexandra Gessner

aus Friedrichshafen

Tübingen
2021

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

Tag der mündlichen Qualifikation	21.03.2022
Dekan	Prof. Dr. Thilo Stehle
1. Berichterstatter	Prof. Dr. Philipp Hennig
2. Berichterstatter	Prof. Dr. Robert Bamler

Abstract

Numerical integration or *quadrature* is one of the workhorses of modern scientific computing and a key operation to perform inference in intractable probabilistic models. The epistemic uncertainty about the true value of an analytically intractable integral identifies the integration task as an inference problem itself. Indeed, numerical integration can be cast as a probabilistic numerical method known as Bayesian quadrature (BQ). BQ leverages structural assumptions about the function to be integrated via properties encoded in the prior. A posterior belief over the unknown integral value emerges by conditioning the BQ model on an actively selected point set and corresponding function evaluations. Iterative updates to the Bayesian model turn BQ into an *adaptive* quadrature method that quantifies its uncertainty about the solution of the integral in a principled way.

This thesis traces out the scope of probabilistic integration methods and highlights types of integration tasks that BQ excels at. These arise when sample efficiency is required and encodable prior knowledge about the integration problem of low to moderate dimensionality is at hand.
The first contribution addresses transfer learning with BQ. It extends the notion of active learning schemes to cost-sensitive settings where cheap approximations to an expensive-to-evaluate integrand are available. The degeneracy of acquisition policies in simple BQ is lifted upon generalization to the multi-source, cost-sensitive setting. This motivates the formulation of a set of desirable properties for BQ acquisition functions.
A second application considers integration tasks arising in statistical computations on Riemannian manifolds that have been learned from data. Unsupervised learning algorithms that respect the intrinsic geometry of the data rely on the repeated estimation of expensive and structured integrals. Our custom-made active BQ scheme outperforms conventional integration tools for Riemannian statistics.
Despite their unarguable benefits, BQ schemes provide limited flexibility to construct suitable priors while keeping the inference step tractable. In a final contribution, we identify the ubiquitous integration problem of computing multivariate normal probabilities as a type of integration task that is structurally taxing for BQ. The instead proposed method is an elegant algorithm based on Markov chain Monte Carlo that permits both sampling from and estimating the normalization constant of linearly constrained Gaussians that contain an arbitrarily small probability mass.

As a whole, this thesis contributes to the wider goal of advancing integration algorithms to satisfy the needs imposed by contemporary probabilistic machine learning applications.

Zusammenfassung

Numerische Integration oder *Quadratur* ist eine wichtige Methode in wissenschaftlichen computergestützten Modellen und eine essenzielle Operation für die Inferenz in probabilistischen Modellen. Die epistemische Unsicherheit über den wahren Wert eines analytisch unlösbaren Integrals lässt eine Formulierung der numerischen Integration als Inferenzproblem zu. Die resultierende probabilistische numerische Methode nennt sich *Bayes'sche Quadratur* (BQ). BQ nutzt strukturelle Annahmen über die zu integrierende Funktion, die in der *A-priori* Verteilung berücksichtigt werden, um durch Konditionierung auf einen Satz aktiv ausgewählter Punkte und Funktionsauswertungen eine *A-posteriori* Verteilung über den unbekannten Integralwert zu konstruieren. Mittels einer iterativen Anpassung des Bayes'schen Modells ist BQ eine *adaptive* Quadraturmethode, die eine Quantifizierung ihrer Unsicherheit bezüglich der Lösung des Integrals ermöglicht.

In dieser Dissertation werden die Stärken probabilistischer Integrationsmethoden aufgezeigt und spezifische Anwendungen hervorgehoben, für die BQ besonders geeignet ist. Zu derartigen Anwendungen gehören Integrale moderater Dimensionalität, die mittels begrenzter Ressourcen geschätzt werden müssen und über die kodierbares Vorwissen vorhanden ist.
Der erste Beitrag befasst sich mit dem Transferlernen mittels BQ. Darin wird das aktive Lernen auf Situationen mit limitiertem Budget übertragen, in denen Auswertungen des Integranden teuer sind, aber günstigere Approximationen als Informationsquelle zur Verfügung stehen. Zur Verallgemeinerung der typischen Akquisitionsfunktionen auf kostenabhängige Raten, die mehrere Funktionen berücksichtigen, bedarf es aufgrund pathologischer Fälle besonderer Vorsicht. Dies motiviert die Formulierung wünschenswerter Eigenschaften einer BQ-Akquisitionsfunktion.
In einer weiteren Anwendung werden Integrationsaufgaben untersucht, die bei statistischen Berechnungen auf dateninduzierten riemannschen Mannigfaltigkeiten auftreten. Unüberwachte Lernalgorithmen, die die intrinsische Geometrie der Daten berücksichtigen, sind auf die wiederholte Berechnung teurer und strukturierter Integrale angewiesen. Unser maßgeschneiderter aktiver BQ-Algorithmus übertrifft herkömmliche Integrationsmethoden im Bereich riemannscher Statistik.
Trotz einiger unbestreitbarer Vorteile limitiert die eingeschränkte Flexibilität bei der Konstruktion geeigneter *A-priori* Wahrscheinlichkeiten unter Beibehaltung der geschlossenen Form des Integrationsschritts den Anwendungsrahmen von BQ. Die Berechnung von Integralen linear begrenzter multivariater Normalverteilungen ist ein allgegenwärtiges Integrationsproblem, welches eine ungünstige Struktur für BQ aufweist. Die alternativ vorgeschlagene Methode ist ein eleganter Algorithmus, der auf einem Markov-Chain-Monte-Carlo-Verfahren basiert und sowohl die Stichprobenziehung als auch die Schätzung der Masse von linear beschränkten Gauß-Integralen ermöglicht.

Im weitesten Sinne leistet diese Arbeit einen Beitrag zur Weiterentwicklung pratikabler Integrationsalgorithmen für Anwendungen des maschinellen Lernens.

Acknowledgements

First and foremost, I would like to thank Philipp Hennig for his guidance, support, and trust throughout my PhD. His creativity and passion for research have been invaluable ingredients for my scientific growth. Further I wish to thank my TAC Ulrike von Luxburg and Philipp Berens, as well as Robert Bamler for evaluating this thesis.
I would not have embarked on this journey without a nudge by Maren Mahsereci, who has been a mentor and friend ever since, and I thank her very much for that.

MPI-IS, the University of Tübingen, and IMPRS-IS have provided a fantastic research environment and I thank the associated colleagues and friends for inspiring discussions over much appreciated high-quality coffee. In particular, I would like to thank my fantastic colleagues from the *Methods of Machine Learning* group, Maren Mahsereci, Michael Tiemann, Motonobu Kanagawa, Simon Bartels, Hans Kersting, Lukas Balles, Filip de Roos, Frank Schneider, Matthias Werner, Felix Dangel, Susanne Zabel, Agustinus Kristiadi, Jonathan Wenger, Nicholas Krämer, Filip Tronarp, Thomas Gläßle, Katharina Ott, Nathanael Bosch, Julia Grosse, Marius Hobbhahn, Emilia Magnani, Marvin Pförtner, and Jonathan Schmidt. A warm thank you also goes to my students Christian Fröhlich, Philipp Hummel, and Mila Gorecki. Franziska Weiler deserves my special gratitude for not only sorting every conceivable administrative matter, but also all sorts of life problems.

It was a great pleasure to work with my collaborators Georgios Arvanitidis, Bernhard Schölkopf, and Oindrila Kanjilal. Also, I want to thank Maren Mahsereci, Javier González and the `emukit` team for a fantastic internship at Amazon Research Cambridge, and Andrei Paleyes from who taught me a lot about software development during the stay. This turned out useful for my involvement in `probnum` that greatly hinged on joint efforts with Toni Karvonen on *Coding Tuesdays*, and support by Jonathan Wenger and, yet again, Maren. I further thank the entire probabilistic numerics community for the warm and friendly atmosphere as well as numerous inspiring workshops and discussions, in particular with Motonobu Kanagawa, Toni Karvonen, and François-Xavier Briol.

Advancing research and wrapping up a thesis has not always been easy in the era of a pandemic. The regular exchange with Filip de Roos has been of extraordinary (if not critical) help in this time. Thanks for proofreading (parts of) this manuscript Filip de Roos, Maren Mahsereci, Nicholas Krämer, Nathanael Bosch, Seth Axen, and Amin Charusaie! Great thanks also goes to my new team Álvaro Tejero Cantero, Seth Axen, and Elena Sizana.

For the good times in Tübingen I thank the current and previous members of the Jürgensen-WG, in particular Naomi Dura and Andrea Aquino. Finally I wish to thank my family for their constant support.

Tübingen 2022

Contents

List of Figures

List of Tables

Notation

General Notation and Preliminaries

$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \ldots$	A vector
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \ldots$	A matrix
$a, b, c, \ldots$	A scalar
α	An acquisition function $\alpha : \mathcal{X} \to \mathbb{R}$
$\mathbb{C}[\cdot, \cdot]$	Covariance
$\mathcal{D}$	Data, usually $\{\boldsymbol{x}_n, y_n\}_{n=1}^N = \{\boldsymbol{X}, \boldsymbol{y}\}$ the default dataset
$\mathcal{D}_\star$	Prospective, yet unobserved data $\{\boldsymbol{X}_\star, \boldsymbol{y}_\star\}$
D	Input dimension
$D_{\mathrm{KL}}(p \mid\mid q)$	The Kullback-Leibler divergence between p and q
e	Euler's constant
ϵ_n	Noise variable of the n^{th} data point
erf	The error function
$\mathbb{E}[\cdot]$	Expectation
f	short form of $f_{\boldsymbol{X}}$

$\boldsymbol{f}_{\boldsymbol{X}}$	N stacked function evaluations of f, $\boldsymbol{f}_{\boldsymbol{X}} = [f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)]^\top$
f	$\mathsf{f} : \mathbb{R}^D \to \mathbb{R}, \boldsymbol{x} \mapsto \mathsf{f}(\boldsymbol{x})$ a random process
$\mathbf{f}$	The random process restricted to nodes $\boldsymbol{X}$, short for $\mathbf{f}_{\boldsymbol{X}}$
f	$f : \mathbb{R}^D \to \mathbb{R}, \boldsymbol{x} \mapsto f(\boldsymbol{x})$ a real function
Γ	The Gamma function
$\mathcal{GP}$	A Gaussian process
$\boldsymbol{G}$	$\boldsymbol{K} + \sigma^2 \boldsymbol{I}$ Kernel Gram matrix with additive noise $(N \times N)$
$\boldsymbol{G}_\star$	$\boldsymbol{K}_\star + \sigma^2 \boldsymbol{I}$ Kernel Gram matrix with additive noise $(N_\star \times N_\star)$ at new nodes
H	Entropy
$\boldsymbol{I}$	An identity matrix (when the size is clear from the context)
$\boldsymbol{I}_D$	A $D \times D$ identity matrix
$\mathbb{1}[\cdot]$	Indicator function $\mathbb{1}[X] = 1$ if X is true, otherwise $\mathbb{1}[X] = 0$
$\mathbf{1}$	A vector $[1, \ldots, 1]^\top$
I	Mutual information
$\boldsymbol{K}$	$k(\boldsymbol{X}, \boldsymbol{X})$ Kernel Gram matrix—no index defaults to evaluation at $\boldsymbol{X}$
$\boldsymbol{K}_\star$	$k(\boldsymbol{X}_\star, \boldsymbol{X}_\star)$ Kernel Gram matrix at prospective points $\boldsymbol{X}_\star$
k	A positive semi-definite kernel $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R},\ \boldsymbol{x}, \boldsymbol{x}' \mapsto k(\boldsymbol{x}, \boldsymbol{x}')$
$k(\boldsymbol{X}, \boldsymbol{X}')$	The kernel evaluated at given pairs $(\boldsymbol{x}_i, \boldsymbol{x}'_j)$, $k : \mathbb{R}^{D \times N} \times \mathbb{R}^{D \times N'} \to \mathbb{R}^{N \times N'}$
$k_{\mathcal{D}}$	The posterior variance of a GP
κ	$\kappa : \mathbb{R}^D \to \mathbb{R},\ \boldsymbol{x} \mapsto \kappa(\boldsymbol{x}) = \int_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}') \nu(\boldsymbol{x}')\, d\boldsymbol{x}'$ Kernel mean
$\boldsymbol{\kappa}$	$\kappa(\boldsymbol{X}) \in \mathbb{R}^N$ Kernel mean at locations $\boldsymbol{X}$
$\boldsymbol{\kappa}_\star$	$\kappa(\boldsymbol{X}_\star) \in \mathbb{R}^N$ Kernel mean at prospective locations $\boldsymbol{X}_\star$
λ	A scalar lengthscale $\lambda \in \mathbb{R}_+$
$\boldsymbol{L}$	A lower triangular matrix (Cholesky factor)
$\boldsymbol{\Lambda}$	A lengthscale matrix $\boldsymbol{\Lambda} \in \mathbb{R}_+^{D \times D}$
$\boldsymbol{m}$	$= \boldsymbol{m}_{\boldsymbol{X}}$, the prior mean function at locations $\boldsymbol{X}$
$\boldsymbol{\mu}$	A mean vector
m	A prior mean function $m : \mathcal{X} \to \mathbb{R}$
$m_{\mathcal{D}}$	The posterior GP mean function $m_{\mathcal{D}} : \mathcal{X} \to \mathbb{R}$
$\mathfrak{m}$	The prior BQ mean, $m = \int_{\mathcal{X}} m(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x})$
$\mathfrak{m}_{\mathcal{D}}$	The posterior BQ mean $m = \int_{\mathcal{X}} m_{\mathcal{D}}(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x})$
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	A Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\nu(\boldsymbol{x})$	An integration measure on $\mathcal{X}$
N	Number of observations
$N_\star$	Number of prospective observations
p, q	Probability density functions
$Q(f; \boldsymbol{X}, \boldsymbol{w})$	A quadrature rule with nodes $\boldsymbol{X}$ and weights $\boldsymbol{w} \in \mathbb{R}^N$
$Q_k(f; \boldsymbol{X})$	A kernel quadrature rule
$\mathbb{R}$	The real numbers
$r_{\boldsymbol{\Lambda}}$	The Mahalanobis distance w.r.t. the matrix $\boldsymbol{\Lambda}$
σ^2	A (noise) variance
$\boldsymbol{\Sigma}$	A covariance matrix
Θ	A parameter space

$\mathsf{\theta}$	A random variable for unknown parameters
$\boldsymbol{\theta}$	A parameter realization $\boldsymbol{\theta} \in \Theta$
θ	A kernel output scale
$\mathcal{T}$	Some transformation operator
ϑ	An angle $\vartheta \in [0, 2\pi]$
$\mathbb{V}[\cdot]$	Variance
$\mathfrak{v}$	The prior BQ variance
$\mathfrak{v}_{\mathcal{D}}$	The posterior BQ variance
$\boldsymbol{w}$	Weights, $\boldsymbol{w} = [w_1, \ldots, w_N]^\top \in \mathbb{R}^N$
$\Phi(\cdot)$	The cumulative Gaussian
$\boldsymbol{X}$	N input locations $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$, $\boldsymbol{X} \subset \mathcal{X}$
$\boldsymbol{X}_\star$	$N_\star$ new input locations $[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_\star}]$
$\mathcal{X}$	Input domain $\mathcal{X} \subseteq \mathbb{R}^D$
$\boldsymbol{x}$	Input location, $\boldsymbol{x} \in \mathcal{X}$
$\boldsymbol{x}_\star$	A single new input location $\boldsymbol{x}_\star \in \mathcal{X}$
$\boldsymbol{x}_n$	The n^{th} input location
$\boldsymbol{x}_{1:N}$	N input locations, synonymous for $\boldsymbol{X}$
x	Input location when $D = 1$, *i.e.*, $\mathcal{X} \subseteq \mathbb{R}$
y	A random variable representing yet unobserved evaluations $\boldsymbol{y}$
$\boldsymbol{y}$	Function evaluations $\boldsymbol{y} = [y_1, \ldots, y_N]^\top$
y_n	A single function evaluations $y_n = f(\boldsymbol{x}_n) + \epsilon_n$
$y_{1:N}$	Synonymous for $\boldsymbol{y}$
Z	A random variable representing the unknown integral Z
Z	An integral $Z = \int_{\mathcal{X}} f(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x})$

Chapter 5 (Active Multi-Information Source Bayesian Quadrature)

$\alpha_{\boldsymbol{\ell}_\star}(\boldsymbol{X}_\star)$	A non-myopic acquisition function, $\alpha_{l_\star}(\boldsymbol{x}_\star)$ in the myopic case
$\boldsymbol{B}$	A coregionalization matrix ($L \times L$)
c_l	The cost of source l, in general $c : \mathcal{L} \times \mathcal{X} \to [\delta, 1]$
δ	$0 < \delta \leq 1$, the lower bound on the cost
ϵ_l	Noise of source l
$\boldsymbol{f}(\boldsymbol{x})$	$f : \mathbb{R}^D \to \mathbb{R}^L, \boldsymbol{x} \mapsto \boldsymbol{f}(\boldsymbol{x})$ a vector-valued function
$\mathbf{f}(\boldsymbol{x})$	A vector-valued random process
$f_l(\boldsymbol{x})$	The l^{th} information source
$\boldsymbol{G}_{\boldsymbol{\ell}}$	$= \boldsymbol{K}_{\boldsymbol{\ell}\boldsymbol{\ell}}(\boldsymbol{X}, \boldsymbol{X}) + \boldsymbol{\Sigma}_{\boldsymbol{\ell}} \in \mathbb{R}^{N \times N}$ the noise-corrected Gram matrix
$\boldsymbol{K}$	The $L \times L$ matrix-valued covariance function with entries $k_{ll'}(\boldsymbol{x}, \boldsymbol{x}')$
$\boldsymbol{K}_{\boldsymbol{\ell}\boldsymbol{\ell}}(\boldsymbol{X}, \boldsymbol{X})$	The Gram matrix, $\boldsymbol{K}_{\boldsymbol{\ell}\boldsymbol{\ell}}(\boldsymbol{X}, \boldsymbol{X}) \in \mathbb{R}^{N \times N}$
$\mathcal{L}$	The set of information sources $\mathcal{L} = \{1, \ldots, L\}$
$\boldsymbol{\ell}$	The sources that have been evaluated, $\boldsymbol{\ell} = \{l_n\}_{n=1}^N$
$\boldsymbol{\ell}_\star$	The sources that are to be evaluated, $\boldsymbol{\ell} = \{l_n\}_{n=1}^{N_\star}$
L	Number of information sources, indexed by l
l	The index of one information source $l \in \mathcal{L}$, $l = 1$ is the primary source
$\boldsymbol{m}$	Prior mean function $\boldsymbol{m} \in \mathbb{R}^L$

$\mathcal{X}_\star$	Locations of prospective observation triplets
$N_\star$	Number of prospective observation triplets
$\boldsymbol{\Sigma}_{\boldsymbol{\ell}}$	$= \mathrm{diag}(\sigma^2_{l_1}), \ldots, \sigma^2_{l_N}$
$\boldsymbol{y}_{\boldsymbol{\ell}_\star}$	Prospective noisy function evaluations at sources $\boldsymbol{\ell}_\star$
$\boldsymbol{y}_{\boldsymbol{\ell}}$	Noisy function evaluations $\boldsymbol{y}_{\boldsymbol{\ell}} = [f_{l_1}(\boldsymbol{x}_1) + \epsilon_{l_1}, \ldots, f_{l_N}(\boldsymbol{x}_N) + \epsilon_{l_N}]^\top$
Z_1	The random variable representing Z_1
Z_1	The integral over the primary source, $Z_1 = \int_{\mathcal{X}} f_1(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x})$

Chapter 6 (Bayesian Quadrature on Riemannian Data Manifolds)

α	Acquisition function (here: US)
$\bar{\alpha}$	DCV acquisition function
β	Scaling factor of a unit tangent vector
$\mathcal{C}$	The normalization constant of the Gaussian on the manifold
$\mathcal{D}$	Data collected by BQ, $\mathcal{D} = \{\boldsymbol{v}_n, f_{\boldsymbol{\mu}}(\boldsymbol{v}_n)\}_{n=1}^{N_{\mathrm{BQ}}}$
δ	A small positive offset
$\mathrm{Exp}_{\boldsymbol{\mu}}(\cdot)$	Exponential map $\mathrm{Exp}_{\boldsymbol{\mu}}(\cdot) : \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M} \to \mathcal{M}$
$\boldsymbol{\eta}$	A stochastic function for construction of the metric tensor $\boldsymbol{M}$
$E(\gamma)$	The energy functional
f	The random process associated with $f_{\boldsymbol{\mu}}$
$f_{\boldsymbol{\mu}}$	The integrand, $f_{\boldsymbol{\mu}} : \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M} \to \mathbb{R}_+$
g	An auxiliary GP s.t. $\mathsf{f} = 1/2\mathsf{g}^2 + \delta$
γ	A curve $\gamma : [0, 1] \to \mathcal{M}$
g	Auxiliary function $g = \sqrt{2(f_{\boldsymbol{\mu}} - \delta)}$
$\boldsymbol{J}_{\boldsymbol{\eta}}$	The Jacobian of $\boldsymbol{\eta}$
K	The number of components in a LAND mixture
$\mathrm{Log}_{\boldsymbol{\mu}}(\cdot)$	Logarithmic map $\mathrm{Log}_{\boldsymbol{\mu}}(\cdot) : \mathcal{M} \to \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$
$\mathscr{L}$	The Lagrangian
$L(\gamma)$	The length of a curve γ
$\boldsymbol{\mu}$	The LAND mean and reference point for computations on the tangent space
$\boldsymbol{\mu}_k$	The mean of the k^{th} LAND component
$\mathcal{M}$	A Riemannian manifold
$\boldsymbol{\mu}^2_{\{\phi,\theta\}}$	Mean of the VAE encoder (ϕ) and decoder (θ)
$\boldsymbol{M}$	A metric tensor
N	Size of the dataset
N_{BQ}	Number of BQ nodes
π_k	Weight of the k^{th} LAND component
p_θ	A decoder of a VAE with parameters θ
q_ϕ	An encoder of a VAE with parameters ϕ
ρ	A parameter that determines the asymptotic value of the metric
r_{nk}	The responsibility of the k^{th} LAND component for the n^{th} datum
σ^2	The lengthscale of the kernel metric
$\boldsymbol{\Sigma}$	Euclidean covariance of the LAND model
$\boldsymbol{\sigma}^2_{\{\phi,\theta\}}$	Variance vector of the VAE encoder (ϕ) and VAE decoder θ

$\boldsymbol{\Sigma}_k$	The covariance of the k^{th} LAND component
S	Number of MC samples
$\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$	The tangent space to $\mathcal{M}$ at $\boldsymbol{\mu}$
$\hat{\boldsymbol{v}}$	A tangent vector with unit norm
$\boldsymbol{v}$	A tangent vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$
w_n	The weight of the n^{th} point in a kernel metric
$\boldsymbol{x}$	A location on the manifold $\boldsymbol{x} \in \mathcal{M}$
$\boldsymbol{x}_n$	The n^{th} datapoint
x_d	d^{th} component of a location $\boldsymbol{x} \in \mathcal{M}$
x_{nd}	d^{th} component of the n^{th} datapoint $\boldsymbol{x}_n$

Chapter 7 (Inference With Gaussians Under Linear Domain Constraints)

$\boldsymbol{A}$	Matrix containing the linear constraints, $\boldsymbol{A} \in \mathbb{R}^{D \times M}$
$\boldsymbol{a}_m$	The m^{th} linear constraint
$\boldsymbol{b}$	The shift vector of the linear constraints
b_m	The m^{th} shift of the linear constraints, $\gamma_T = 0$
γ_t	Shift of the t^{th} subset
$\mathcal{L}$	The integration domain where $\prod_{m=1}^{M} \boldsymbol{a}_m^\top \boldsymbol{x} + b_m > 0$
$\mathcal{L}_t$	The t^{th} subset defined through γ_t
λ	An arbitrary parameter
M	Number of linear constraints
N	Number of samples per subset
$\hat{\rho}_t$	The estimated t^{th} conditional probability for HDR
ρ	$\rho \in [0, 1]$ conditional probability for subset simulation
ρ_t	t^{th} conditional probability for HDR
T	Number of subsets
$\hat{Z}$	The MC estimator for the integral
Z	The integral $\int_{\mathcal{L}} \mathrm{d}\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$

Acronyms

AMSBQ	active multi-information source Bayesian quadrature
BO	Bayesian optimization
BQ	Bayesian quadrature
DPP	determinantal point process
EP	expectation propagation
ESS	elliptical slice sampling
GP	Gaussian process
HDR	Holmes-Diaconis-Ross algorithm
HMC	Hamiltonian Monte Carlo
ICM	intrinsic coregionalization model

i.i.d.	independent and identically distributed
IP	integral precision
IPI	integral precision increase
IVR	integral variance reduction
KRR	kernel ridge regression
LAND	locally adaptive normal distribution
LIN-ESS	elliptical slice sampling with linear constraints
MC	Monte Carlo
MCMC	Markov chain Monte Carlo
MI	mutual information
NIV	negative integral variance
ODE	ordinary differential equation
PDF	probability density function
PNM	probabilistic numerical method
QMC	quasi-Monte Carlo
RKHS	reproducing kernel Hilbert space
SMC	sequential Monte Carlo
ADK	adenylate kinase
BVP	boundary value problem
CDF	cumulative distribution function
CPU	central processing unit
DCV	directional cumulative variance
GMM	Gaussian mixture model
IVP	initial value problem
LMC	linear model of coregionalization
MMD	maximum mean discrepancy
NUTS	no-U-turn sampler
PCA	principal component analysis
PE	percentile estimator
RBF	radial basis function
RL	reinforcement learning
SBQ	sequential Bayesian quadrature
SEIR	susceptible, exposed, infected, recovered
SIR	susceptible, infected, recovered
VBMC	variational Bayesian Monte Carlo
VAE	variational autoencoder
VBQ	vanilla Bayesian quadrature
WCE	worst-case error
WSABI	warped sequential active Bayesian integration
WSABI-L	WSABI with linearization
WSABI-M	WSABI with moment matching

Prologue

1 Introduction

The fundamental theorem of calculus[1] establishes the connection between integration—the computation of sums of infinitesimal quantities—to differentiation, *i.e.*, rates of change. It states that if the integral of continuous real-valued function f on a closed interval $[a, b]$ is $F(x) = \int_a^x f(x)\,\mathrm{d}x$, then F is differentiable and its derivative is $F'(x) = f(x)$. Derivative calculus follows mechanical rules so simple that computers can automatically and with little overhead differentiate through any computer program that executes an arbitrarily nested sequence of elementary operations on a set of basic functions with known derivatives [101]. Integration, alas, is more intricate and lacks a tool that is even remotely akin to *automatic differentiation*. Even for univariate integrals, the list of recipes for closed-form integration is limited [99], and an enormous body of literature is concerned with numerical methods for quadrature[2] of *intractable* integrals—integrals that have no analytic form [60]. Numerical schemes are concerned with the approximation of tasks that lack a closed-form solution by a sequence of atomic arithmetic operations on floating-point numbers that computers excel at. The difficulty of integration grounds on the fact that it is a *global* operation. In order to determine an overall mass, the density[3] must in principle be known at every location. In contrast, differentiation operates *locally*—it asks for the rate of change of a quantity at a single, given location. To make matters worse, classic numerical recipes that have proven successful in solving univariate integration problems suffer from the *curse of dimensionality*, the exponential explosion of computational cost with increasing dimension, when applied to multivariate problems [26]. The conception of adequate integration algorithms for contemporary applications is thus a matter of ongoing research, to which this thesis is merely a small contribution.

1: First proved by Isaac Barrow [48, Lecture 10] and the origin of a life-long rivalry between his student Isaac Newton and the German mathematician Gottfried Wilhelm Leibniz.

[101]: Griewank (2014), *Automatic differentiation*

[99]: Gradshteyn and Ryzhik (2014), *Table of integrals, series, and products*

2: Terminology is somewhat inconsistent. Some researchers use *quadrature* to mean numerical integration of univariate functions, and *cubature* originally for two-, later in general for multi-dimensional integration. Here we take a dimension-agnostic view and use *numerical integration* and *quadrature* synonymously, but refrain from usage of the term *cubature*.

[60]: Davis and Rabinowitz (1983), *Methods of numerical integration*

3: *i.e.*, the mass contained in an infinitesimal volume element

[26]: Bellman et al. (1957), *Dynamic programming*

Numerical integration is only one of the workhorses of computational methods that have become indispensable for modern scientific discovery. They form the cornerstone for solving complex mathematical models in virtually all disciplines of science and engineering.
Consider the example of Earth system models. These are large-scale global climate models that simulate the dynamics of the physical components of the climate system (atmosphere, ocean, land, and sea ice) including biogeochemical cycles to predict its response to external forcing [73]. Finite computational resources set a limit on the complexity of such models and mandate approximations. Simplifications are also needed to handle the lack of understanding of some processes such as cloud formation. *Uncertainty* is thus inherent to these models. The notion of uncertainty is categorized in two classes:

[73]: Flato et al. (2013), 'Evaluation of climate models'

- *Epistemic uncertainty* refers to uncertainty caused by ignorance about things that could be known in principle, such as the above examples. We can be uncertain about deterministic quantities by simply not knowing them.
- *Aleatoric uncertainty* refers to inevitable unpredictability due to randomness that corrupts the outcome of an experiment by an

> irreducible stochastic component. Stochasticity is sometimes attributed to chaotic systems such as the atmosphere in which tiny disturbances can lead to massive discrepancies in the outcomes of repeated experiments.[4]

Information is key to mitigating uncertainty. Data collection and storage has grown at a tremendous rate in recent years that is constantly challenging the ever-increasing limits of resources in compute and physical memory. The over-abundance of data has led to the emergence of powerful data-processing tools under the umbrella of *machine learning*. Machine learning models are predictive models that establish a *statistical* relationship between inputs and outputs purely from data without any *physical* assumption on the data generating process.[5]

Models—whether mechanistic or statistical—have in common that they make predictions about unknown quantities, *i.e.,* they perform *inference* on latent quantities. Inference refers to the procedure of drawing new conclusions based on given premises either through *deductive reasoning*—when logical conclusions can be drawn—or through *inductive reasoning*, the process of reducing uncertainty about unknown quantities on the grounds of incomplete information [125]. We focus on inference as a principled way for decision-making in the face of uncertainty.
In that sense, deterministic models of dynamical systems *deduce* future states of the system from the current state and knowledge about the governing dynamics. Machine learning models *infer* functional relationships between data and labels. Latent parameters of mechanistic models can be *inferred* from observations of the physical world; this type of problem is known as *inverse problem*.
Taking a rather unconventional position, numerical algorithms also perform inference: they *infer* solutions to intractable equations through computation. This perspective is taken by the young field of *probabilistic numerics*, which sees numerical algorithms as learning agents [108]. Uncertainty guides them in taking decisions about *actions* to query certain information. In probabilistic numerics, *data* are results of computation on a CPU. This thesis primarily focuses on a Bayesian view on *numerical integration*, but remains open towards the wider scope of efficient inference algorithms that enhance scientific computing.

4: Chaos is the reason why it is so difficult to predict the intensity of extreme weather events such as storms, or even just medium-term weather variability. We will touch on problems related to the prediction of rare events on a more abstract level in Chapter 7.
However, one might argue whether randomness exists in nature at all. Wouldn't tossing a die—from Latin *alea*—be perfectly predictable if we exactly knew the initial conditions of the toss, and every little detail about the friction on the surface? This would transfer said problem into the previous category of uncertainty. The same is true if the *exact* conditions of the Earth system were known could be reproduced.

5: Physics-informed machine learning models are on the rise.
[125]: Jaynes (2003), *Probability theory: The logic of science*

[108]: Hennig et al. (2015), 'Probabilistic numerics and uncertainty in computations'

1.1 Numerics for inference

Bayesian statistics is concerned with finding a posterior belief about variables of interest $\theta \in \Theta \subseteq \mathbb{R}^D$ conditioned on observed data $\mathcal{D}$. Inference requires a model about the data generating process and assumptions on values $\boldsymbol{\theta}$ that the parameters θ could take *a priori*.[6] These span an overall hypothesis space $\mathfrak{H}$ that contains all possible explanations of the data that we wish to admit.[7] The model connects parameter realizations $\boldsymbol{\theta}$ and observations $\mathcal{D}$ via the *likelihood* $p(\mathcal{D} \mid \boldsymbol{\theta}, \mathfrak{H})$. It is a function of the parameters and represents how likely it is to see the data $\mathcal{D}$ if the parameter were fixed to $\boldsymbol{\theta}$ under the hypothesis $\mathfrak{H}$. Assumptions about θ are encoded via the *prior* $p(\boldsymbol{\theta} \mid \mathfrak{H})$. *Bayes' theorem* specifies how to update

6: θ denotes the *random variable*, $\boldsymbol{\theta}$ a realization of θ, *i.e.,* one fixed set of parameters. We use the short notation $p(\boldsymbol{\theta}) = p(\theta = \boldsymbol{\theta})$ to denote the probability that θ takes the value $\boldsymbol{\theta}$; or in the continuous case that the observation falls within an infinitesimal interval around $\boldsymbol{\theta}$.

7: $\mathfrak{H}$ becomes important in testing different hypothesis against each other and is usually omitted if only one is dealt with (which we will do most of the time).

to an *a posteriori* belief about the parameters *given* the data [8]

$$\underbrace{p(\boldsymbol{\theta} \mid \mathcal{D}, \mathfrak{H})}_{\text{posterior}} = \frac{\overbrace{p(\mathcal{D} \mid \boldsymbol{\theta}, \mathfrak{H})}^{\textit{likelihood}} \overbrace{p(\boldsymbol{\theta} \mid \mathfrak{H})}^{\textit{prior}}}{\underbrace{p(\mathcal{D} \mid \mathfrak{H})}_{\textit{evidence}}}. \tag{1.1}$$

It is a straightforward corollary of the *sum* and *product rule* of probability that in turn directly follow from the axiomatic grounds that probability theory is built on.[9]

The denominator of (1.1) ensures that the posterior density is normalized. It is an integral[10] over the joint probability density $p(\mathcal{D}, \boldsymbol{\theta}) = p(\mathcal{D} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})$, dropping $\mathfrak{H}$ for readability,

$$p(\mathcal{D}) = \int_{\Theta} p(\mathcal{D} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}. \tag{1.2}$$

The integration over $\boldsymbol{\theta}$ is termed *marginalization* and can be understood as averaging over all possible states of a random variable, weighted by the probability of that particular state. $p(\mathcal{D})$ is called *evidence* or *marginal likelihood*. The term evidence originates from its significance for evaluating hypotheses against each other: $p(\mathcal{D} \mid \mathfrak{H})$ is the probability of the observed data $\mathcal{D}$ under a given hypothesis. If the probability to observe data $\mathcal{D}$ is higher under hypothesis $\mathfrak{H}_1$ than under $\mathfrak{H}_2$, we can conclude that hypothesis $\mathfrak{H}_1$ is superior over $\mathfrak{H}_2$ to explain the given data (*cf.* Figure 1.1).
Once the posterior is known, the key tasks are to compute expectations of functions of interest w.r.t. the posterior

$$\mathbb{E}_{\Theta \mid \mathcal{D}}[f(\boldsymbol{\theta})] = \int_{\Theta} f(\boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathcal{D}) \, \mathrm{d}\boldsymbol{\theta} \tag{1.3}$$

and predictions of new data $\mathcal{D}_\star$, which can be seen as posterior expectations of the likelihood

$$p(\mathcal{D}_\star \mid \mathcal{D}) = \int p(\mathcal{D}_\star \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathcal{D}) \, \mathrm{d}\boldsymbol{\theta}.$$

Unless likelihood and prior are *conjugate*, the posterior is rarely available in closed form. The immediate culprit is the evidence (1.2), and further down the line integrals of type (1.3). Boldly said, Bayesian inference problems are covered in their entirety by these two types of integrals.

If integration was easy, so would be Bayesian inference.

This statement is of course a bit of an oversimplification in the sense that inference on large datasets would still be prohibitively expensive even if the true posterior were available. Just evaluating the likelihood of a large amount of independent and identically distributed (i.i.d.) data can be a computational burden that outweighs the cost of the integration problem. Worse even, we will encounter cases in this thesis where integrals are tractable, but their evaluation requires combinatorial iteration over the dataset, which renders numerical integration superior over evaluation of the closed-form expression.

8: Named after Reverend Thomas Bayes [23], although the development of what we now call *Bayesian statistics* in fact largely goes back to Laplace [153].

9: Two different sets of axioms by Cox [54] and Kolmogorov [145] §1 have been used to arrive at the same rules for probability calculus that the reader is assumed to be familiar with. If not so, they may be referred to the excellent textbook by Jaynes [125] or MacKay [168].

10: If θ is continuous—if θ is discrete, it is a sum. We assume that all densities exist.

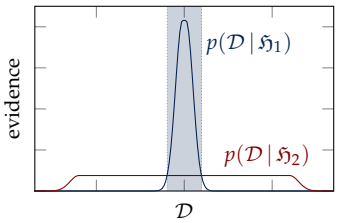


Figure 1.1: Bayesian inference penalizes complex models and thus implicitly encodes Occam's razor that states that the simplest model that is able to explain the data should be preferred. A simple model (here $\mathfrak{H}_1$) can accommodate only a limited range of predictions but assigns a larger probability—the evidence—to such data. A complex model $\mathfrak{H}_2$ can cover more data, but needs to spread the probability mass over this wider range of possible predictions. If $\mathcal{D}$ falls into the shaded area, $\mathfrak{H}_1$ is preferred over $\mathfrak{H}_2$ on the grounds of a larger evidence. This illustration is reproduced from Fig. 2 in MacKay [167].

Nevertheless, efforts in *approximate inference* in one way or the other revolve around approximating or circumventing the integrals (1.2) and (1.3) and have brought forth a zoo of algorithms (see *e.g.*, [30]). These can be categorized into *Monte Carlo* techniques for sampling [150], or approximation of the posterior by a tractable distribution, *e.g.*, *variational inference* [31], *Laplace approximation* or *expectation propagation* [176]. Sampling uses random numbers to approximate expectations as a sum

$$\mathbb{E}_{\boldsymbol{\theta}\,|\,\mathcal{D}}[f(\boldsymbol{\theta})] \simeq \hat{\mathbb{E}}[f(\boldsymbol{\theta})] = \frac{1}{S}\sum_{s=1}^{S} f(\boldsymbol{\theta}_s) \qquad \boldsymbol{\theta}_s \sim p(\boldsymbol{\theta}\,|\,\mathcal{D}).$$

[30]: Bishop (2006), 'Pattern recognition and machine learning'

[150]: Kroese et al. (2013), *Handbook of Monte Carlo methods*

[31]: Blei et al. (2017), 'Variational inference: A review for statisticians'

[176]: Minka (2013), 'Expectation propagation for approximate Bayesian inference'

Samples from the posterior usually rely on *Markov chain Monte Carlo (MCMC)* techniques that only asymptotically sample from the true posterior and have all sorts of pleasant theoretical guarantees, but are often slow in practice. Conveniently, they do not require to compute the evidence in order to estimate expectations. Inconveniently, the evidence cannot be estimated from the samples, but either require a separate procedure or one of the rare targeted algorithms that can solve both tasks, *e.g.*, through tempering Graham and Storkey [100] or nested sampling Skilling [232]. The alternative are strategies for finding distributions that are in some sense representative of the true, intractable posterior. There are multiple *rationalia* to construct such an approximation, and usually optimized towards predictive performance. A rough estimate of the evidence is merely a side product and can be of low quality [151].

[100]: Graham and Storkey (2017), 'Continuously tempered Hamiltonian Monte Carlo'

[232]: Skilling (2004), 'Nested sampling'

[151]: Kuss et al. (2005), 'Assessing approximate inference for binary Gaussian process classification.'

The unconventional, third option that we will focus on for solving inference problems is by viewing the computation of the evidence as an inference problem itself. Through appropriate choices of model, this procedure is analytically tractable and even provides a measure of uncertainty on the estimated evidence.

1.2 Inference for numerics

The loop closes when regarding numerical algorithms through the lens of probabilistic inference. A numerical method estimates a deterministic quantity that is inherently unknown on the grounds of finite computation. This process bears structural resemblance to inference: a numerical method *infers* a latent quantity that we are *epistemically uncertain* about—the intractable solution to a numerical problem—by collecting *data*, that is, outcomes of feasible auxiliary computations that bear information about the quantity of interest. The procedure of numerical computation is hence amenable to a probabilistic treatment. As opposed to classic methods that output point estimates about the numerical solution to a problem, probabilistic numerical algorithms output *probability distributions*.

The conceptual origin of probabilistic numerics dates back to the 1890s, when Henri Poincaré lectured on probability calculus at Sorbonne university in Paris. He explicitly associates numerical problems such as the computation of definite integrals, initial value problems, and inverse problems with epistemic uncertainty and recommends treating them probabilistically [205]. The second half of the following century saw scattered works[11] that largely explore theoretical relations between statistics and computational tasks, *e.g.*, [154, 4, 142]. The road towards practicable algorithms was paved by Diaconis [66], O'Hagan [191, 190] and Skilling [231] who attacked integration and differential equations from a Bayesian

11: see Cockayne et al. [51] and Hennig et al. [112] for a more complete historical context.

[205]: Poincaré (1912), *Calcul des probabilités*

[154]: Larkin (1972), 'Gaussian measure in Hilbert space and applications in numerical analysis'

[4]: Ajna and Dalenius (1960), 'Några tillämpningar av statistiska idéer på numerisk integration'

[142]: Kimeldorf and Wahba (1970), 'A correspondence between Bayesian estimation on stochastic processes and smoothing by splines'

[66]: Diaconis (1988), 'Bayesian numerical analysis'

[191]: O'Hagan (1992), 'Some Bayesian numerical analysis'

[190]: O'Hagan (1991), 'Bayes-Hermite quadrature'

[231]: Skilling (1993), 'Bayesian numerical analysis'

viewpoint, and Močkus who laid the foundations for *Bayesian optimization* [178, 179]. The field has taken off in the second decade of the 2000s under the banner of Hennig et al. [108] and Cockayne et al. [51]. By 2021, probabilistic numerical methods (PNMs) have left the stage of merely demonstrating their validity and have produced innovative algorithms that have found compelling use cases in practical and industrial applications. With a brand-new textbook [112] and `probnum`[12] , a designated toolbox purely for probabilistic numerical methods[13] the community is settling down as an established field at the intersection of numerical methods, probability theory, statistics, and machine learning.

Probabilistic numerics is still a young field that has been sparked by the emergence of increasingly challenging computational problems that arise with big data and ever more complex models in science and machine learning. Up until today, the probabilistic numerics armory has seen tremendous progress in most areas of numerical analysis. This non-exhaustive list does not do justice to the amount of recent advances that make the field evolve at unprecedented speed:

- *Bayesian quadrature (BQ)* constructs posterior measures over the value of an intractable integral $Z = \int_{\mathcal{X}} f(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x})$ from evaluations of the integrand $\boldsymbol{f}_{\boldsymbol{X}} = [f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)]^\top$ [190, 37].
- *Probabilistic linear solvers* infer the solution $\boldsymbol{x} \in \mathbb{R}^N$ to a linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ with $\boldsymbol{A} \in \mathbb{R}^{N \times N}, \boldsymbol{b} \in \mathbb{R}^N$ by assigning a prior probability to $\boldsymbol{A}$ or the unknown solution $\boldsymbol{x}$ and observing projections of $\boldsymbol{A}$ [22, 109, 260].
- Initial value problems (IVPs) are *ordinary differential equations (ODEs)* $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, t)$ with $t \in [t_0, T]$ subject to an initial condition $\boldsymbol{x}[t_0] = \boldsymbol{x}(0)$. They are structurally similar to time series and a probabilistic solution can be obtained by Gaussian filtering [49, 223, 141]. Probabilistic approaches have also been proposed for the solution of boundary value problems [147], inverse problems [140], and partial differential equations [52, 255].
- *Bayesian optimization (BO)* has been widely adopted to find the global extremum of an expensive black-box functions $\boldsymbol{x}_* = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})$ [229, 78]. Stochastic optimization methods benefit from probabilistic subroutines to select hyperparameters, *e.g.*, step sizes via line searches [169].

The probabilistic perspective on numerical methods comes with multiple advantages. First and foremost, many numerical problems are highly structured and have known properties. Knowledge such as positivity of the integrand in Bayesian quadrature [102] or symmetry of a matrix in a linear system [109, 260] can be encoded as inductive bias for the probabilistic numerical method via the prior.
Moreover, the explicit treatment of uncertainty puts a measure on the inaccuracy due to discretization and a finite amount of computational resources. The uncertainty can be used to precociously stop an algorithm once a desired accuracy is reached or to adapt parameters of the algorithm to remain within a specified confidence interval with high probability, *e.g.*, to adapt step sizes in probabilistic ODE solvers [33]. PNM further come with a sound way of incorporating noise, when exact evaluations are unavailable, e.g. because they originate from noisy measurements, or require a massive amount of data that can not be loaded in memory (*e.g.*, [62]).

[178]: Močkus (1975), 'On Bayesian methods for seeking the extremum'

[179]: Močkus (1994), 'Application of Bayesian approach to numerical methods of global and stochastic optimization'

[108]: Hennig et al. (2015), 'Probabilistic numerics and uncertainty in computations'

[51]: Cockayne et al. (2019), 'Bayesian probabilistic numerical methods'

[112]: Hennig et al. (2022), *Probabilistic Numerics: Computation as Machine Learning*

12: `probnum.readthedocs.io`

13: Individual PNM have appeared in a number of toolboxes, such as `emukit` [196], `botorch` [20], and others.

[190]: O'Hagan (1991), 'Bayes-Hermite quadrature'

[37]: Briol et al. (2019), 'Probabilistic integration: A role in statistical computation?'

[22]: Bartels et al. (2019), 'Probabilistic linear solvers: A unifying view'

[109]: Hennig (2015), 'Probabilistic interpretation of linear solvers'

[260]: Wenger and Hennig (2020), 'Probabilistic linear solvers for machine learning'

[49]: Chkrebtii et al. (2016), 'Bayesian solution uncertainty quantification for differential equations'

[223]: Schober et al. (2019), 'A probabilistic model for the numerical solution of initial value problems'

[141]: Kersting et al. (2020), 'Convergence rates of Gaussian ODE filters'

[147]: Krämer and Hennig (2021), 'Linear-time probabilistic solution of boundary value problems'

[140]: Kersting et al. (2020), 'Differentiable likelihoods for fast inversion of likelihood-free dynamical systems'

[52]: Cockayne et al. (2017), 'Probabilistic numerical methods for PDE-constrained Bayesian inverse problems'

[255]: Wang et al. (2021), 'Bayesian numerical methods for nonlinear partial differential equations'

[229]: Shahriari et al. (2016), 'Taking the human out of the loop: a review of Bayesian optimization'

[78]: Garnett (2022), *Bayesian optimization*

[169]: Mahsereci and Hennig (2015), 'Probabilistic line searches for stochastic optimization'

[102]: Gunter et al. (2014), 'Sampling for inference in probabilistic models with fast Bayesian quadrature'

[33]: Bosch et al. (2021), 'Calibrated adaptive probabilistic ODE solvers'

[62]: de Roos and Hennig (2019), 'Active probabilistic inference on matrices for preconditioning in stochastic optimization'

As a direct consequence of expressing uncertainty, the probabilistic approach unlocks Bayesian decision theory to reason about optimal actions to be taken by the algorithm based on previous data and the model. Active decision-making turns algorithms into autonomous *agents* that interact with their environment and adaptively choose actions to optimize the information gained from new observations under the current belief about the computational objective.

At the same time, PNM are becoming increasingly competitive to standard numerical algorithms and come with implementations that permit benchmarking against established classical methods, *e.g.*, [34] for ODEs. By now, there are applications in which PNM are a natural, if not the only choice. As a rule of thumb, PNM are preferred whenever information is too scarce to run a classical method up to convergence. In cases where the numerical agent interacts with an expensive source of information, there might not even be a viable alternative to PNM. This is the uncontested territory in particular for Bayesian optimization and quadrature. In brief, PNM are successful whenever *(i)* inference comes at low computational overhead, as is the case for probabilistic ODE solvers that permit linear-time solutions; or *(ii)* when uncertainty is sufficiently important to justify the computational overhead of inference. In the age of big data, algorithms play an unprecedented rôle in all facets of our everyday lives and take consequential decisions that impact humans. We better want methods that communicate their confidence and base their decisions on a principled treatment of the uncertainty that they face.

[34]: Bosch et al. (2021), 'Pick-and-mix information operators for probabilistic ODE solvers'

1.3 Contributions

The overarching theme of this thesis is to enhance numerical integration methods for probabilistic inference, where intractable integrals are ubiquitous. The first and primary part deals with Bayesian quadrature with a focus on practical aspects and use cases. It highlights practical benefits of the uncertainty quantification in BQ that enable algorithms that lack their like amongst classical numerical integration schemes. The thesis elucidates the scope of the probabilistic formulation of quadrature and aims to enhance the accessibility of the yet underemployed numerical integration scheme that BQ continues to be. This motives an intuitive introduction aimed at potential practitioners in Chapter 2 (Bayesian Quadrature).

Previous research has largely concentrated on the theoretical well-groundedness of BQ, with valuable results on consistency and convergence. Those are prerequisites to guarantee that newly conceived Bayesian numerical integration schemes produce correct outcomes. In real-world use-cases, however, BQ does not operate in the limit regime that underlies assumptions made for theoretical analysis. The theoretical grounds of BQ therefore play a minor role in this work, but the key results and connections to the well-studied theory of reproducing kernel Hilbert spaces (RKHSs) are reviewed in Section 2.3.
Large sample sizes fundamentally pose a challenge to BQ by construction due to the cubic cost of inference in the general case. It is a strong inductive bias and a careful choice of locations for evaluation that make

BQ excel when there is limited computational budget. By assigning probabilities to its action space, a PNM can quantify the expected reward of action choices and select them accordingly. Chapter 3 (Active Design for Bayesian Quadrature) studies BQ as a self-adapting agent that actively explores its search space.

Chapter 5 (Active Multi-Information Source Bayesian Quadrature) takes the decision-theoretic capabilities of BQ a step further and develops an algorithm for information transfer from secondary sources that approximate the assumed-expensive integration task of interest. This procedure requires balancing the information gained from querying less accurate approximations and the invested computational budget. It fundamentally relies on the probabilistic treatment of the numerical integration to enable information-theoretic reasoning about the utility of future observations.

Challenging integration tasks also arise in machine learning when accounting for the intrinsic geometry induced by data. Chapter 6 (Bayesian Quadrature on Riemannian Data Manifolds) ports BQ onto geometry-aware territory and exploits its core strengths—the sample efficiency of the active BQ agent equipped with a strong prior—to speed up computations for inference tasks on Riemannian manifolds.

Despite the unarguable strengths of BQ on specific integration problems, there exist settings that are notoriously hard for BQ. The computation of multivariate Gaussian probabilities is a ubiquitous integration task that epitomizes limitations of the standard BQ approach via Gaussian processes and forms the second part of this thesis. Instead of pushing BQ into the limelight purely on dogmatic grounds, certain integration tasks have properties that motivate leaving the stage to established stochastic integration methods based on (Markov Chain) Monte Carlo (MC)MC, with a concise overview in Chapter 4. Chapter 7 (Inference With Gaussians Under Linear Domain Constraints) takes a deep dive into the integration of multivariate normal distributions subject to linear domain constraints. These are ubiquitous integration problems that by design are not amenable to the standard BQ approach via Gaussian processes. We pragmatically develop a sampling and estimation scheme that relies on the interplay of tailored MCMC algorithms. This method is able to reliably estimate even tiny probabilities and to provide samples from the constrained domain needed to compute expectations.

Overall, this thesis develops conceptual guidelines for the use of BQ and traces out its scope by studying some settings that undisputably reside within and others that exceed the realms of BQ.

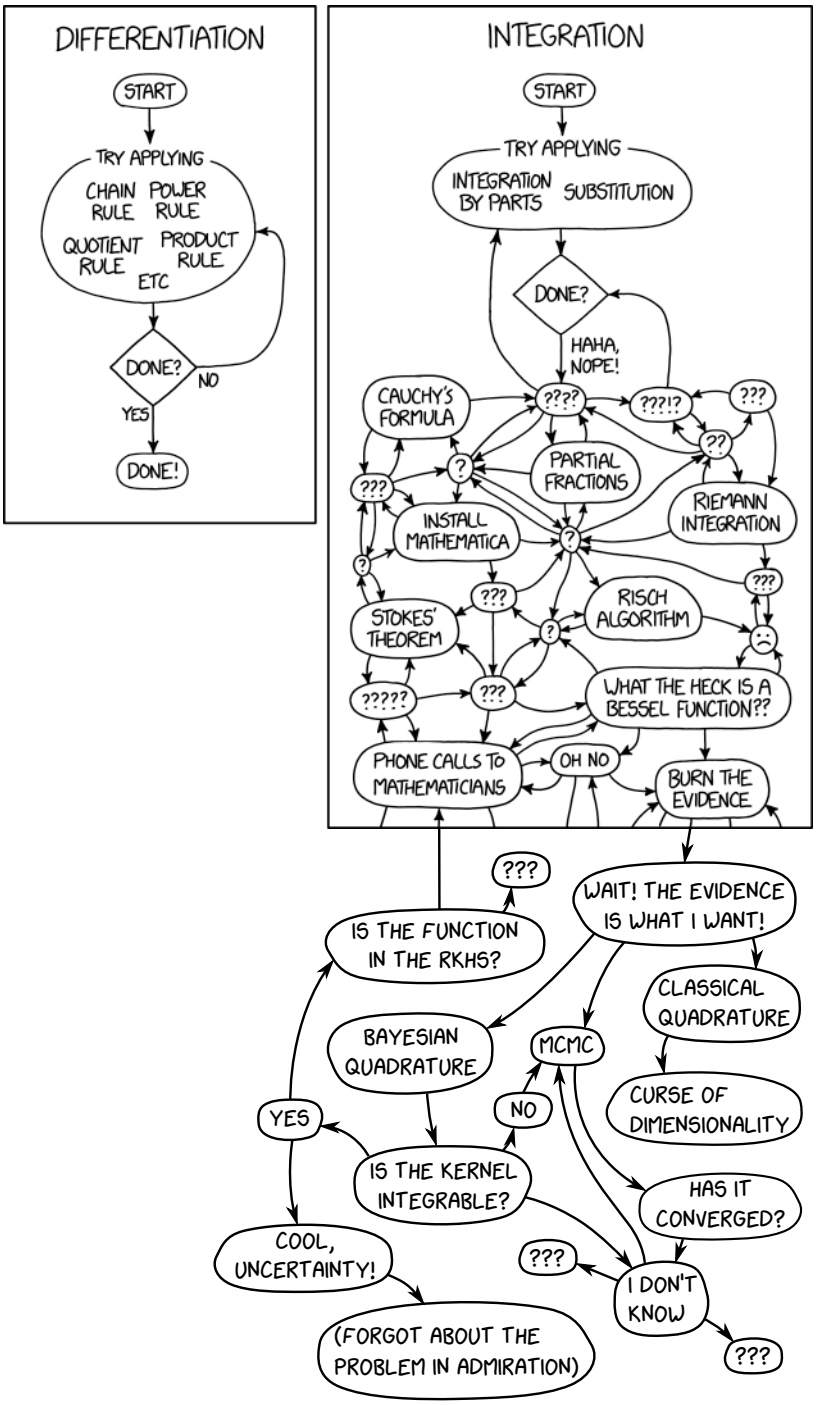


Figure 1.2: *Top:* Randall Munroe's accurate (albeit incomplete) view on integration (*Differentiation and Integration* by Randall Munroe, licensed under CC BY-NC 2.5). *Bottom:* A sparse, still incomplete (and biased) extension of the integration flowchart.

Publications

The research conducted as a part of my PhD studies has been published in the following peer-reviewed articles, of which the first three are relevant to this dissertation.

(I) A. Gessner, J. Gonzalez, and M. Mahsereci. 'Active multi-information source Bayesian quadrature'. In: *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*. Ed. by A. Globerson and R. Silva. Vol. 115. Proceedings of Machine Learning Research. AUAI Press, 2019 for Chapter 5.

(II) A. Gessner, O. Kanjilal, and P. Hennig. 'Integrals over Gaussians under linear domain constraints'. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by S. Chiappa and R. Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020 for Chapter 7.

(III) C. Fröhlich, A. Gessner, P. Hennig, B. Schölkopf, and G. Arvanitidis. 'Bayesian quadrature on Riemannian data manifolds'. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021 for Chapter 6.

(IV) F. de Roos, A. Gessner, and P. Hennig. 'High-dimensional Gaussian process inference with derivatives'. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021 is not covered.

The success of these projects heavily relied on the joint work with collaborators.

(I) Gessner et al. [89] was initiated by Maren Mahsereci and Javier Gonzalez and carried out by me as part of an internship at Amazon Research, Cambridge, UK. The original direction was proposed by Javier Gonzalez, while the project was pinned down by Maren Mahsereci, who also provided continual guidance and structure throughout the project. Derivations are due to both Maren Mahsereci and me. The implementation of the method and the bulk of the experiment have been executed by me, except for the bivariate experiment that is due to Maren Mahsereci. Chapter 3 and 5 are based on this publication.

The initial idea for (II) Gessner et al. [90] was due to Philipp Hennig, and was further developed by him and me jointly. Oindrila Kanjilal provided expertise in reliability analysis and multilevel splitting techniques, especially about the Holmes-Diaconis-Ross algorithm (HDR). I performed the majority of the work on derivations, implementation, experiments, and writing. This work is the basis of Chapter 7.

(III) Fröhlich et al. [77] was a collaborative project initiated by Georgios Arvanitidis and me. The methodological part was split thematically: Georgios Arvanitidis was responsible for everything relating to geometry; the conceptional background related to Bayesian quadrature was provided by me. Christian Fröhlich carried out the project as a part of his Master thesis, with joint supervision by Georgios and me. Much of the development of the project has happened collaboratively. Chapter 6 is in parts very close to the initial submission of this work. That said,

there might be text passages that have been primarily written by my co-authors—especially the parts on differential geometry, the locally adaptive normal distribution (LAND) model, and details on experiments—and this text might have some overlap also with the M.Sc. thesis by Christian Fröhlich. The visualizations have exclusively been prepared by Christian Fröhlich and only slightly adapted by me for this thesis.

(IV) de Roos et al. [61] will not be covered, as it is fairly unrelated to the main thread of this dissertation. The scientific idea of this work is entirely due to Filip de Roos, but has been refined in extensive discussion between us. Derivations have been carried out by him, and have been checked, complemented, and notationally aligned by me. He also took care of the implementation. My contribution was an experiment employing the methods devised by Filip de Roos, and a significant part of the paper writing.

Preliminaries

2 Bayesian Quadrature

Consider the problem of computing the integral over a function $f : \mathcal{X} \to \mathbb{R}$ over a domain $\mathcal{X} \subseteq \mathbb{R}^D$ against a measure $\nu(\boldsymbol{x})$ on $\mathcal{X}$,

$$Z = \int_{\mathcal{X}} f(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}). \tag{2.1}$$

When no closed-form solution to this integral is available, numerical methods are needed. Numerical integration schemes seek to approximate the integral (2.1) by weighted sum of function evaluations—a *quadrature rule*

$$Q(f; \boldsymbol{X}, \boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{f}_{\boldsymbol{X}} = \sum_{n=1}^{N} w_n f(\boldsymbol{x}_n), \tag{2.2}$$

with real weights $\boldsymbol{w} = [w_1 \dots w_N]^\top \in \mathbb{R}^N$ and function evaluations $\boldsymbol{f}_{\boldsymbol{X}} \coloneqq [f(\boldsymbol{x}_1) \dots f(\boldsymbol{x}_N)]^\top \in \mathbb{R}^N$ at design points $\boldsymbol{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} \subset \mathcal{X}$. Roughly put, numerical integration is concerned with finding good weights, good design points, or both, in order to approximate (2.1). Unfortunately, no one-size-fits-all solution exists that works reliably and efficiently on any problem of the type (2.1).

There are broadly two popular classes of algorithms for numerical integration, *(i)* classical quadrature schemes (Section 2.4) that work well on univariate or very low-dimensional integral problems; and *(ii) Monte Carlo methods* (Chapter 4), that use random samples $\boldsymbol{x}_n \sim \nu$ with uniform weights $w_n = \frac{1}{N}$ to obtain an estimate of the integral. This chapter deals with *probabilistic numerical integration*, a less prominent but even more promising class of numerical integration algorithms.

[190]: O'Hagan (1991), 'Bayes-Hermite quadrature'

[175]: Minka (2000), *Deriving quadrature rules from Gaussian processes*

[37]: Briol et al. (2019), 'Probabilistic integration: A role in statistical computation?'

Probabilistic integration Bayesian quadrature (BQ) [190, 175, 37] takes a different perspective on (2.1): The intractability of the integral causes Z to be unknown, even if the integrand f is 'known' in the sense that it can be evaluated anywhere by using elementary arithmetic operations.

Take the example of a univariate function[1] $f : [0, 1] \to \mathbb{R}_+$ and its integral over the domain $\mathcal{X} = [0, 1]$ against the Lebesgue measure $\mathrm{d}\nu(x) = \mathrm{d}x$,

$$f(x) = \frac{\sqrt{x}\, e^{-\sin^2(10x-2)}}{\cosh(4x-3)} \quad \text{and} \quad Z = \int_0^1 f(x) \, \mathrm{d}x, \tag{2.3}$$

depicted in Figure 2.1. The integrand can be evaluated on a computer to machine precision at any desired location $x \in [0, 1]$ by executing the computational recipe given by the function (2.3). Doing so requires arithmetic operations on the functions $e, \sin, \cosh, \sqrt{\cdot}$, which are sufficiently low-level to consider them elementary.[2] The presence of the integral sign changes the situation. The integral of said function is absent in extensive tables of closed-form integrals such as [99]. Symbolic integration libraries such as `sympy` [174] also capitulate. The integral Z is a deterministic quantity, but with the algebraic tools we are equipped with, we have no means of determining its value. The epistemic uncertainty about Z motivates thinking about the integration as an inference problem. There

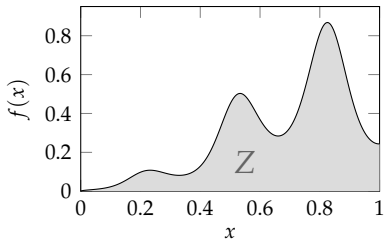


Figure 2.1: The univariate function (2.3) $\mathcal{X} = [0, 1]$. The gray area under the curve is our object of interest.

1: The motivating example heavily leans on Diaconis [66] and Hennig et al. [108].

2: The boundary between elementary operations and an algorithm is debatable. Hennig et al. [112, Chapter 3] suggest pragmatism in fixing atomic operations to those routines available in the GNU C library.

[99]: Gradshteyn and Ryzhik (2014), *Table of integrals, series, and products*

[174]: Meurer et al. (2017), 'SymPy: Symbolic computing in Python'

is nothing random about Z, but the lack of knowledge about its value lets us replace Z by a random variable Z.
Some knowledge about Z can be extracted from the functional form of the integrand and its plot in Figure 2.1. There we can see that f is positive and upper-bounded by 1. Given that the integration domain is $[0, 1]$ we know *a priori* that $0 < Z < 1$. Such knowledge should go into the construction of a prior over Z. A likelihood connects 'observations', which in quadrature usually are function evaluations[3] $f_{\boldsymbol{X}}$ to Z. All that is left is to find the posterior $\mathsf{Z} \mid f_{\boldsymbol{X}}$. As Section 2.2 will show, the expected value of this posterior takes the form of a quadrature rule (2.2) for a certain choice of prior. It further comes with a sound measure of uncertainty about the outcome of the computation. The technical details on how this is achieved is the topic of this chapter.

3: One could conceive other forms of observations, but in quadrature it is common to deal with point evaluations of the integrand.

Overview This chapter is meant to introduce the reader to numerical integration from the viewpoint of Bayesian inference. We first equip the reader with the toolset to consider basic probabilistic integration algorithms and subsequently place them in a theoretical context. The remainder of the chapter is concerned with establishing links to other numerical integration schemes, providing a literature overview, and touching on more elaborate models used to encode prior knowledge about the integral in a BQ method. At this stage, we assume a fixed set of design points $\boldsymbol{X}$ given. The following chapter, Chapter 3, will be concerned with *optimal* and *practical* schemes to decide where the integrand f should be evaluated.

2.1 Gaussian inference

2.1.1 The Gaussian distribution

Of central importance in probability theory and statistics is the *normal* or *Gaussian* distribution. The probability density function (PDF) of a univariate Gaussian random variable $\mathsf{z} \in \mathbb{R}$ with realizations $z \in \mathbb{R}$ is

$$\mathcal{N}(z; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right)$$

and is fully characterized by its mean $\mathbb{E}[\mathsf{z}] = \mu \in \mathbb{R}$ and variance $\mathbb{V}[\mathsf{z}] = \mathbb{E}[\mathsf{z}^2] - \mathbb{E}[\mathsf{z}]^2 = \sigma^2 \in \mathbb{R}_+$.

The univariate Gaussian distribution is easily generalized to N random variables $\{\mathsf{z}_i\}_{i=1}^N$, stacked into the vector $\boldsymbol{\mathsf{z}} \in \mathbb{R}^N$, that are said to be jointly Gaussian distributed with PDF

$$\mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-N/2} \det(\boldsymbol{\Sigma})^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu})\right)$$

with now a mean *vector* $\boldsymbol{\mu} \in \mathbb{R}^N$ with elements $\mathbb{E}[\mathsf{z}_i] = \mu_i$ and a symmetric, positive semidefinite covariance *matrix* $\boldsymbol{\Sigma} \in \mathbb{R}^{N \times N}$. The diagonal entries $\boldsymbol{\Sigma}_{ii} = \sigma_i^2 = \mathbb{V}[\mathsf{z}_i]$ are the variances of variables z_i, and the off-diagonal entries $\boldsymbol{\Sigma}_{ij} = \boldsymbol{\Sigma}_{ji} = \mathbb{C}[\mathsf{z}_i, \mathsf{z}_j]$, $i \neq j$ measure how z_i and z_j relate to each other. $\boldsymbol{\Sigma}_{ij} > 0$ indicates that they follow the same trend, while for $\boldsymbol{\Sigma}_{ij} < 0$ they are negatively correlated such that larger values of z_i tend to entail smaller values of z_j. $\boldsymbol{\Sigma}_{ij} = 0$ means they are

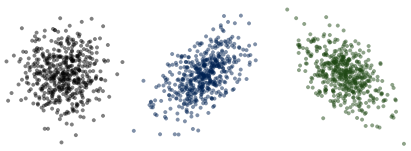


Figure 2.2: *Left to right:* Realizations of two uncorrelated ($\boldsymbol{\Sigma}_{12} = 0$), positively ($\boldsymbol{\Sigma}_{12} > 0$), and negatively ($\boldsymbol{\Sigma}_{12} < 0$) correlated Gaussian random variables.

uncorrelated and observing z_i carries no information about z_j and vice versa (*cf.* Section 3.1.1). The effect of the covariance matrix on samples in a bivariate setting is depicted in Figure 2.2.

Linear operations The distribution is called *standard normal* when $\boldsymbol{\mu} = \mathbf{0}$ and the covariance is the identity matrix $\boldsymbol{\Sigma} = \boldsymbol{I}$. Any Gaussian random variable $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be written as a linear transformation of a standard normal random variable $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

$$\mathbf{z} = \boldsymbol{L}\boldsymbol{\xi} + \boldsymbol{\mu}$$

where $\boldsymbol{L}$ is a suitable decomposition of $\boldsymbol{\Sigma}$ such that $\boldsymbol{L}\boldsymbol{L}^\top = \boldsymbol{\Sigma}$, *e.g.*, the Cholesky factorization.[4] In other words, any linear transformation of a Gaussian random variable $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\mathbf{z}_1 \in \mathbb{R}^N$ is again a Gaussian random variable $\mathbf{z}_2 \in \mathbb{R}^M$,

$$\mathbf{z}_2 = \boldsymbol{A}\mathbf{z}_1 + \boldsymbol{b} \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b}, \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^\top) \tag{2.4}$$

for $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ and $\boldsymbol{b} \in \mathbb{R}^M$.

4: Any positive semidefinite matrix $\boldsymbol{\Sigma}$ can be decomposed as $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^\top$ where $\boldsymbol{L}$ is a lower triangular matrix. This ubiquitous computational trick has been found by André-Louis Cholesky.

Marginalization and conditioning Gaussians are closed under conditioning and marginalization. That is, given the joint distribution over $\mathbf{z}_1 \in \mathbb{R}^N$ and $\mathbf{z}_2 \in \mathbb{R}^M$ as

$$p(z_1, z_2) = \mathcal{N}\left(\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}; \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix}\right)$$

both the marginals $p(z_1)$, $p(z_2)$ and the conditionals $p(z_1 \mid z_2)$, $p(z_2 \mid z_1)$ take a Gaussian form as

$$\begin{aligned} p(z_1) = \int_{\mathcal{X}} p(z_1, z_2)\,\mathrm{d}z_2 \;&=\; \mathcal{N}(z_1; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \quad \text{and} \\ p(z_1 \mid z_2) &= \mathcal{N}\big(z_1; \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1} z_2, \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}\big) \end{aligned} \tag{2.5}$$

and vice versa for $\mathbf{z}_2$. Since $p(z_1, z_2) = p(z_1 \mid z_2)p(z_2)$ is Gaussian, we can conclude that products of Gaussian densities also take Gaussian form. As all of Bayesian inference relies on the *sum* and the *product rule* of probabilities, inference with Gaussians boil down to linear algebra. The popularity of Gaussian distributions is to a large extent based on this computational convenience.

2.1.2 Gaussian process regression

The marginalization property of Gaussians states that a subset of normal random variables also follow a normal distribution. In particular, the marginalized Gaussian does not depend on the parameters of the variables that have been marginalized over. It is thus possible to consider continuous index sets $\mathcal{X}$ such that the variables indexed at any finite subset thereof follow a multivariate normal distribution. The generalization of the normal distribution from a finite to an infinite collection of Gaussian random variables is termed a *Gaussian process (GP)* [212]. A GP is thus suitable to represent a random *function* that we denote as $\mathsf{f} : \mathcal{X} \to \mathbb{R}$.

[212]: Rasmussen and Williams (2006), *Gaussian Processes for machine learning*

Realizations of the GP are real-valued functions $f : \mathcal{X} \to \mathbb{R}$ and we refer to $\boldsymbol{x} \in \mathcal{X}$ as arguments of f rather than indices.

Definition 2.1.1 *Let $\mathcal{X} \subseteq \mathbb{R}^D$ be a non-empty set. A random process $\mathsf{f} : \mathcal{X} \to \mathbb{R}$ is said to be distributed according to a Gaussian process $\mathsf{f} \sim \mathcal{GP}(m, k)$ with mean function $m : \mathcal{X} \to \mathbb{R}$ and covariance function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, if for any finite set of arguments $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subset \mathcal{X}$ and any $N \in \mathbb{N}$, the vector $\mathbf{f}_{\boldsymbol{X}} \coloneqq [\mathsf{f}(\boldsymbol{x}_1), \ldots, \mathsf{f}(\boldsymbol{x}_N)]^\top \in \mathbb{R}^N$ is a multivariate normal random variable that follows the distribution*

$$\mathbf{f}_{\boldsymbol{X}} \sim \mathcal{N}(\boldsymbol{m}_{\boldsymbol{X}}, \boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}})$$

with mean vector $[\boldsymbol{m}_{\boldsymbol{X}}]_n = m(\boldsymbol{x}_n)$ and covariance matrix $[\boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}}]_{nn'} = k(\boldsymbol{x}_n, \boldsymbol{x}_{n'})$ for $n, n' = 1, \ldots, N$.

The mean function is thus the expected value of f under the GP, $m(\boldsymbol{x}) = \mathbb{E}_{\mathsf{f}}[\mathsf{f}(\boldsymbol{x})]$ and similarly, the covariance function denotes the covariance of function values at inputs $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ as $k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{C}_{\mathsf{f}}[\mathsf{f}(\boldsymbol{x}), \mathsf{f}(\boldsymbol{x}')] = \mathbb{E}_{\mathsf{f}}[(\mathsf{f}(\boldsymbol{x}) - m(\boldsymbol{x}))\,(\mathsf{f}(\boldsymbol{x}') - m(\boldsymbol{x}'))]$.

Gaussian process regression GPs are a convenient choice of prior over functions in Bayesian inference tasks such as regression or classification. *Bayesian linear regression* deals with inferring the *latent* (but deterministic) function f from observations

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n \quad \text{where} \quad \epsilon_n \sim \mathcal{N}(0, \sigma^2)$$

that are corrupted by i.i.d. Gaussian noise. The likelihood for one datum is $p(y_n \,|\, \boldsymbol{x}_n) = \mathcal{N}(y_n; f(\boldsymbol{x}_n), \sigma^2)$. Assume we get to see N data points $\boldsymbol{y} \coloneqq y_{1:N}$ at input locations $\boldsymbol{X} \coloneqq \boldsymbol{x}_{1:N}$ summarized as *data* $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$.

The random process f plays the rôle of a *surrogate* for the *latent* function f. Assuming $\mathsf{f} \sim \mathcal{GP}(m, k)$, our prior belief—before seeing the data—about the values that f may take at these locations is[5]

$$\mathbf{f} \coloneqq \begin{bmatrix} \mathsf{f}(\boldsymbol{x}_1) \\ \vdots \\ \mathsf{f}(\boldsymbol{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{x}_1) \\ \vdots \\ m(\boldsymbol{x}_N) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \ldots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & \ldots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \right)$$

which, in short, we write as $\mathbf{f} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K})$ with the prior mean $\boldsymbol{m} \coloneqq \boldsymbol{m}_{\boldsymbol{X}} \in \mathbb{R}^N$ and the *kernel Gram matrix* $\boldsymbol{K}$ s.t. $[\boldsymbol{K}]_{nn'} = k(\boldsymbol{x}_n, \boldsymbol{x}_{n'})$ as in Definition 2.1.1.[6] The regression task is to *infer* f *after* having seen the data $\mathcal{D}$, *i.e.*, to compute the *posterior* $p(\mathsf{f} \,|\, \mathcal{D})$, and to make predictions about function values at a new input location $\boldsymbol{x}_\star$. A rigorous derivation of the posterior using Bayes' rule is non-trivial in the infinite-dimensional setting (*cf. e.g.*, [129]). Nevertheless, the posterior can be obtained through the conditioning rules for Gaussians. This requires the joint distribution of function values $\mathsf{f}_\star \coloneqq \mathsf{f}(\boldsymbol{x}_\star)$ at a new location $\boldsymbol{x}_\star \in \mathcal{X}$ and observations $\boldsymbol{y}$,

$$\begin{bmatrix} \boldsymbol{y} \\ \mathsf{f}_\star \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} \boldsymbol{m} \\ m(\boldsymbol{x}_\star) \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma^2 \boldsymbol{I} & \boldsymbol{k}(\boldsymbol{x}_\star) \\ \boldsymbol{k}(\boldsymbol{x}_\star)^\top & k(\boldsymbol{x}_\star, \boldsymbol{x}_\star) \end{bmatrix} \right),$$

where we have introduced the shorthand notation $[\boldsymbol{k}(\boldsymbol{x}_\star)]_n = k(\boldsymbol{x}_n, \boldsymbol{x}_\star)$ for the column vector of kernel evaluations at $\boldsymbol{X}$. The conditional $\mathsf{f}_\star \,|\, \boldsymbol{y}$

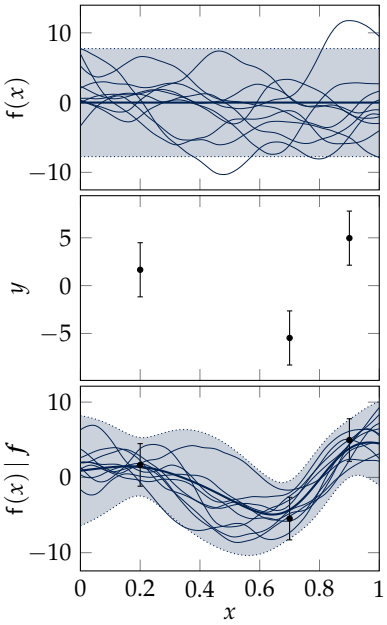


Figure 2.3: *Top:* Draws from a GP prior with Matérn-5/2 kernel (cf. Section 2.1.3), with zero prior mean. The shaded area indicate two standard deviations. *Center:* Three observations with Gaussian noise (errorbars again show two std. dev.); *bottom:* GP posterior conditioned on the data, with draws, mean and uncertainty as above.

5: Textbook linear regression departs from $\mathsf{f}(\boldsymbol{x}) = \mathbf{w}^\top \boldsymbol{\varphi}(\boldsymbol{x})$, a linear combination of basis functions $\boldsymbol{\varphi} : \mathbb{R}^D \to \mathbb{R}^M$ with Gaussian random weights $\mathbf{w} \in \mathbb{R}^M$ that are to be inferred as $p(\boldsymbol{w} \,|\, \boldsymbol{y}) \propto p(\boldsymbol{y} \,|\, \boldsymbol{w}) p(\boldsymbol{w})$. GPs are therefore known as *non-parametric models*, because the parameters $\boldsymbol{w}$ get absorbed in the covariance function [212, Chapter 2].

6: Note that we have dropped the indices $\boldsymbol{X}$ that identify the input, *i.e.*, $\mathbf{f} \coloneqq \mathbf{f}_{\boldsymbol{X}}, \boldsymbol{m} \coloneqq \boldsymbol{m}_{\boldsymbol{X}}$ and $\boldsymbol{K} \coloneqq \boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}}$ to reduce notational clutter. The indices will be used if the evaluation set deviates from the 'default' dataset $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$.

[129]: Kanagawa et al. (2018), 'Gaussian processes and kernel methods: A review on connections and equivalences' Theorem 3.1

is found by applying the familiar rules of Gaussian algebra (2.5). Since this is applicable to *any* new location $\boldsymbol{x} \in \mathcal{X}$, we can find the predictive distribution $\mathsf{f} \mid \mathcal{D} \sim \mathcal{GP}(m_{\mathcal{D}}, k_{\mathcal{D}})$ with posterior mean and covariance

$$m_{\mathcal{D}}(\boldsymbol{x}) \coloneqq \mathbb{E}_{\mathsf{f}\mid\mathcal{D}}[\mathsf{f}(\boldsymbol{x})] = m(\boldsymbol{x}) + \boldsymbol{k}(\boldsymbol{x})^\top \left(\boldsymbol{K} + \sigma^2 \boldsymbol{I}\right)^{-1} (\boldsymbol{y} - \boldsymbol{m}) \tag{2.6}$$

$$k_{\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}') \coloneqq \mathbb{C}_{\mathsf{f}\mid\mathcal{D}}[\mathsf{f}(\boldsymbol{x}), \mathsf{f}(\boldsymbol{x}')] = k(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}(\boldsymbol{x})^\top \left(\boldsymbol{K} + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}(\boldsymbol{x}). \tag{2.7}$$

A GP prior and the posterior after conditioning on three observations is shown in Figure 2.3. When *exact* observations of f are available, *i.e.*, $\sigma \to 0$, the expressions for the posterior mean and covariance require $\boldsymbol{K}$ to be invertible to be well-defined. In practice, a small noise, called *jitter*, is often added to the diagonal of $\boldsymbol{K}$ to overcome numerical instabilities.

The computational bottleneck for inference with GPs is the inversion of the kernel Gram matrix $\boldsymbol{K}$ that comes at a cost of $\mathcal{O}(N^3)$. The cubic complexity on the one hand hinders the use of GPs on large regression datasets, but on the other hand has sparked an entire sub-field of research that is concerned with GP approximations to lower the computational demand. Broadly speaking, they rely either on summarizing data in so-called *inducing points* [243], or on leveraging kernel approximations to efficiently perform the matrix inversion [136].

[243]: Titsias (2009), 'Variational learning of inducing variables in sparse Gaussian processes'

[136]: Katzfuss and Guinness (2021), 'A general framework for Vecchia approximations of Gaussian processes'

Historically, GP regression emerged from spatial modeling tasks in geostatistics, where it is known as 'kriging'. In machine learning, GPs have gained popularity in the early 2000s, following the textbook by Rasmussen and Williams [212].

Linear operations Another useful property of GPs is their closure under linear operations. This is the generalization of (2.4) for the multivariate normal distribution to GPs. Let $\mathcal{L}, \mathcal{M}$ denote linear operators and $\mathsf{f} \sim \mathcal{GP}(m, k)$. Then the joint distribution of $\mathcal{L}\mathsf{f}$ and $\mathcal{M}\mathsf{f}$ is also a GP,

$$\begin{bmatrix} \mathcal{L}\mathsf{f} \\ \mathcal{M}\mathsf{f} \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} \mathcal{L}m \\ \mathcal{M}m \end{bmatrix}, \begin{bmatrix} \mathcal{L}k\mathcal{L}' & \mathcal{L}k\mathcal{M}' \\ \mathcal{M}k\mathcal{L}' & \mathcal{M}k\mathcal{M}' \end{bmatrix} \right), \tag{2.8}$$

where $\mathcal{L}'$ and $\mathcal{M}'$ act from the right on the second argument of the covariance function. Linear functionals of f can thus be inferred from a finite set of linear observations of f. Examples of linear maps are projections, translations, differentiation, and integration, the latter being of significance in this thesis.

2.1.3 Covariance functions

Assumptions about properties such as smoothness of the latent function are encoded via the *covariance function* or *(positive definite) kernel*.[7]

Definition 2.1.2 *(Positive definite kernels) A symmetric bivariate real function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a positive definite kernel if for any finite set $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subset \mathcal{X}; N \in \mathbb{N}$, the matrix $\boldsymbol{K} \in \mathbb{R}^{N \times N}$ with elements $[\boldsymbol{K}]_{nn'} = k(\boldsymbol{x}_n, \boldsymbol{x}_{n'})$ is positive definite, i.e., for all vectors $\boldsymbol{v} \in \mathbb{R}^N, \boldsymbol{v} \neq \boldsymbol{0}$*

$$\sum_{n=1}^{N} \sum_{n'=1}^{N} v_n v_{n'}\, k(\boldsymbol{x}_n, \boldsymbol{x}_{n'}) \geq 0.$$

7: Whenever referring to *kernels*, we will mean *positive definite kernels* throughout the thesis.

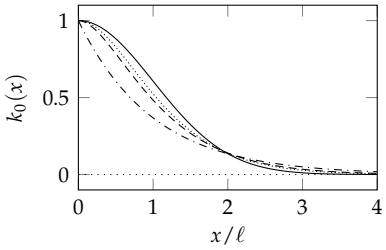


Figure 2.4: Covariance functions $k_0(x) \coloneqq k(x, 0)$ for the following kernels:
— Gaussian
····· Matérn-5/2
– – Matérn-3/2
– · – Ornstein-Uhlenbeck (Matérn-1/2)
····· Brownian motion

A *stationary kernel* can be written as $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x} - \boldsymbol{x}')$ and if $k(\boldsymbol{x}, \boldsymbol{x}') = k(\|\boldsymbol{x} - \boldsymbol{x}'\|)$, the kernel is said to be *isotropic*. Kernels can be scaled by a parameter that changes the output variance of the process, $k(\boldsymbol{x}, \boldsymbol{x}') \to \theta^2 k(\boldsymbol{x}, \boldsymbol{x}')$. A process $\mathrm{g}(\boldsymbol{x}) = a(\boldsymbol{x})\mathrm{f}(\boldsymbol{x})$ with a deterministic function $a(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}$ and $\mathrm{f} \sim \mathcal{GP}(m, k)$ has covariance $\tilde{k}(\boldsymbol{x}, \boldsymbol{x}') = a(\boldsymbol{x})k(\boldsymbol{x}, \boldsymbol{x}')a(\boldsymbol{x}')$. Also, a deterministic rescaling of the inputs $k(\boldsymbol{x}, \boldsymbol{x}') \to k(a(\boldsymbol{x}), a(\boldsymbol{x}'))$ results in a valid kernel. Sums and products of kernels are also kernels [212]. A few kernels that are relevant for BQ are introduced here and illustrated in Figure 2.4. Samples from a GP prior with these covariance functions are illustrated in Figure 2.5.

[212]: Rasmussen and Williams (2006), *Gaussian Processes for machine learning* Chapter 4.2.4

Gaussian kernel The Gaussian kernel[8] is a stationary covariance function

$$k_{\boldsymbol{\Lambda}}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}')^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x} - \boldsymbol{x}')\right) \tag{2.9}$$

with positive definite matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{D \times D}$ that defines the lengthscales on which the process varies. A diagonal $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1^2, \ldots, \lambda_D^2)$ implies different characteristic lengthscales per dimension; a scalar lengthscale $\lambda > 0$ s.t. $\boldsymbol{\Lambda} = \lambda^2 \boldsymbol{I}$ implies *isotropy*. As λ increases, the correlation between two spatially separate points rises, and typical functions from the GP vary less rapidly as a function of their input.

Matérn kernels Matérn kernels, due to [171], are also stationary and parameterized by a smoothness parameter $\gamma > 0$ and a lengthscale matrix $\boldsymbol{\Lambda}$ as in the Gaussian kernel. With the shorthand notation for the Mahalanobis distance $r_{\boldsymbol{\Lambda}} = \left((\boldsymbol{x} - \boldsymbol{x}')^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x} - \boldsymbol{x}')\right)^{1/2}$, their general form is

$$k_{\gamma,\boldsymbol{\Lambda}}(\boldsymbol{x}, \boldsymbol{x}') = \frac{2^{1-\gamma}}{\Gamma(\gamma)} \left(\sqrt{2\gamma} r_{\boldsymbol{\Lambda}}\right)^{\gamma} K_\gamma \left(\sqrt{2\gamma} r_{\boldsymbol{\Lambda}}\right)$$

where Γ is the gamma function and K_γ the modified Bessel function of second kind of order γ. In the isotropic case, $r_{\boldsymbol{\Lambda}} = \|\boldsymbol{x} - \boldsymbol{x}'\|/\lambda$. A kernel of order γ possesses $\lfloor \gamma \rfloor$ derivatives; hence, γ encodes the smoothness of the kernel. The larger γ the smoother are the functions modeled and for $\gamma \to \infty$, the Gaussian kernel is recovered [236]. For half-integer smoothness parameter that can be written as $\gamma = m + 1/2$ in terms of $m \in \mathbb{N}_0$, the expression for the Matérn kernels reduces to the product of an exponential and a polynomial in r,

$$k_{\gamma,\boldsymbol{\Lambda}}(\boldsymbol{x}, \boldsymbol{x}') = e^{-\sqrt{2\gamma} r_{\boldsymbol{\Lambda}}} \frac{\Gamma(m+1)}{\Gamma(2m+1)} \sum_{i=1}^{m} \frac{(m+i)!}{i!(m-i)!} \left(\sqrt{8\gamma} r_{\boldsymbol{\Lambda}}\right)^{m-i}.$$

The case $\gamma = 1/2$ is known as exponential or Laplace kernel and is the covariance function of the *Ornstein-Uhlenbeck process*.

[236]: Stein (2012), *Interpolation of spatial data: some theory for kriging*

Brownian motion kernel The covariance function of a *Wiener process* is known as *Brownian motion kernel* and defined for $D = 1$ on $\mathcal{X} = [0, 1]$ as

$$k^0(x, x') = \min(x, x'). \tag{2.10}$$

Draws from a GP with this kernel are one-dimensional paths of Brownian motion, therefore its name (see lowest frame of Fig-

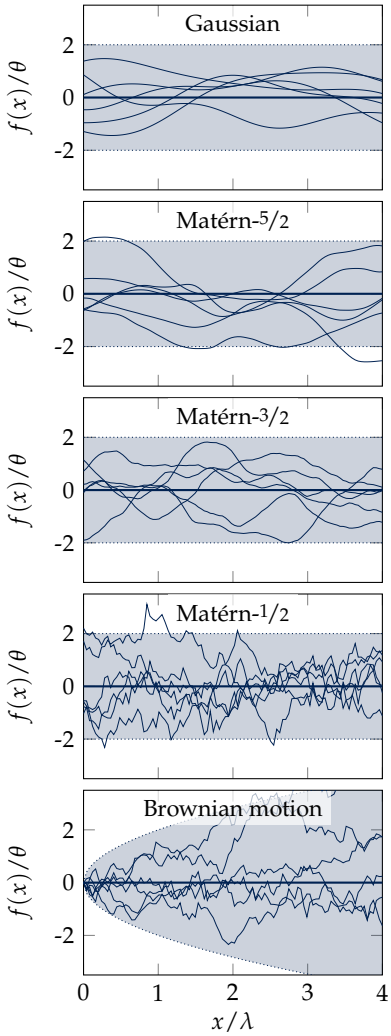


Figure 2.5: Samples from a zero-mean GP prior defined through the kernels in Figure 2.4, with the shaded area indicating the 2σ confidence interval.

8: The Gaussian kernel also goes under the names RBF (*radial basis function*) kernel, *squared exponential*, or *exponentiated quadratic* kernel.

[171]: Matérn (1960), 'Spatial variation'

ure 2.5). The M times integrated Brownian motion kernel can be obtained through the recurrence relation

$$k^M(x, x') = \int_0^x \int_0^{x'} k^{M-1}(\xi, \xi') \, \mathrm{d}\xi \mathrm{d}\xi' = (M!)^{-2} \int_0^1 (x - t)_+^M (x' - t)_+^M \, \mathrm{d}t, \tag{2.11}$$

with $M > 0$ and is M times continuously differentiable [253, 222].

[253]: Wahba (1990), *Spline models for observational data*

[222]: Schober et al. (2014), 'Probabilistic ODE solvers with Runge-Kutta means'

Wendland kernels Let $r_\lambda = \frac{\|x - x'\|}{\lambda}$ and $\lambda > 0$. Wendland [258] introduced stationary kernels given by polynomials

$$k^{D,m}(r_\lambda) = (1 - r_\lambda)_+^{\ell+m} p_{D,m}(r_\lambda),$$

where $(a)_+ := \max(0, a)$, $\ell = \lfloor D/2 \rfloor + m + 1$ determines differentiability, and $p_{D,m}(r_\lambda)$ is a polynomial of degree m. The sharp cutoff at $r_\lambda \geq 1$ means that Wendland kernels can give rise to sparse Gram matrices. However, there seems little benefit in exploiting the sparsity for the inversion of the Gram matrix in GP inference and BQ [118].

[258]: Wendland (2004), *Scattered data approximation*

[118]: Hummel (2020), *Sparse inference for Bayesian quadrature*

Integrability constraints on the kernel required by BQ (see Section 2.2 and Table 2.1) motivates the construction of multivariate kernels from tensor products of one-dimensional kernels,

$$k(\boldsymbol{x}, \boldsymbol{x}') = \prod_{d=1}^{D} k(x_d, x_d').$$

This is relevant *e.g.*, for the Matérn and Wendland kernels that can be integrated when their arguments are univariate, but not when they are multivariate.

2.1.4 On kernel parameters

In GP models, the kernel parameters are known as *hyperparameters*, as they are not direct parameters of the regression model. In the Bayesian paradigm, the hyperparameters $\boldsymbol{\theta} \in \Theta$ would be assigned a prior and marginalized over

$$p(f) = \int_\Theta p(f \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}.$$

This integral is generally intractable. In practice, the hyperparameters are therefore usually chosen s.t. they maximize the log marginal likelihood of the parameters,

$$\boldsymbol{\theta}_\star = \arg\max_{\boldsymbol{\theta} \in \Theta} \log p(\mathcal{D} \mid \boldsymbol{\theta}).$$

The log marginal likelihood for GP regression is

$$\log p(\mathcal{D} \mid \boldsymbol{\theta}) = -\frac{1}{2} \boldsymbol{y}^\top (\boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{y} - \frac{1}{2} \log |\boldsymbol{K}_{\boldsymbol{\theta}} + \sigma^2 \boldsymbol{I}| - \frac{N}{2} \log(2\pi),$$

and the hyperparameters are contained in the kernel Gram matrix, here denoted as $\boldsymbol{K}_{\boldsymbol{\theta}}$.

Instead of considering the kernel hyperparameters, a hyperprior can be placed directly on the kernel Gram matrix. An inverse Wishart process placed on $\boldsymbol{K}$ results in a Student-t process on f. Student-t processes retain

the desirable closure properties of GPs, but exhibit a heavier tail than GPs do and are thus more robust to outliers [228]. A Student-t distribution is also found when placing a Gamma prior on the inverse output variance θ^2 and then marginalizing over θ.

[228]: Shah et al. (2014), 'Student-t processes as alternatives to Gaussian processes'

2.2 Vanilla Bayesian quadrature

With a conceptual idea of probabilistic numerics in mind and the Gaussian inference machinery at hand, let us return to the intractable integral (2.1). The philosophy of BQ is to treat numerical integration as an inference task. The *deterministic* integral Z (2.1) is replaced by the *random variable* Z in order to express the *epistemic uncertainty* about its value. Its goal is to construct a posterior measure over the integral Z given evaluations of f at locations $\boldsymbol{X} = \boldsymbol{x}_{1:N}$. It is natural to think about a prior on the integrand rather than on the integral, as quadrature rules (2.2) rely on point evaluations of the integrand. As in the regression task (Section 2.1.2), the deterministic integrand f is thus modeled using a stochastic process f. The substitute integral for (2.1) is thus a random variable

$$\mathsf{Z} = \int_{\mathcal{X}} \mathsf{f}(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}). \tag{2.12}$$

The closure of GPs under linear operations plays in our favor for choosing a suitable prior: The integration is a linear functional; hence the integral of a GP takes a Gaussian form as well. The choice of a GP prior $\mathsf{f} \sim \mathcal{GP}(m, k)$ induces a univariate Gaussian prior on Z

$$\mathsf{Z} \sim \mathcal{N}(\mathfrak{m}, \mathfrak{v})$$

with moments obtained by applying Fubini's theorem to exchange the expectation over f and the integral over the input space $\mathcal{X}$,

$$\begin{aligned}
\mathfrak{m} &:= \mathbb{E}_{\mathsf{f}}[\mathsf{Z}] = \int_{\mathcal{X}} m(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}) \\
\mathfrak{v} &:= \mathbb{V}_{\mathsf{f}}[\mathsf{Z}] = \iint_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}').
\end{aligned}$$

The choice of a GP as surrogate for the integrand is the default choice in BQ, and we refer to this setup as *vanilla Bayesian quadrature (VBQ)*.
Once nodes and function evaluations $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ are available, the inference step in its essence is just GP regression, with the integral operator $\int_{\mathcal{X}} \cdot \, \mathrm{d}\nu$ applied to the posterior $\mathsf{f} \mid \mathcal{D}$. Bayesian quadrature hence offers a natural way to incorporate noisy observations $y = f(\boldsymbol{x}) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. In practice, exact evaluations are often available, and the likelihood reduces to a Dirac measure $p(y_i \mid f(\boldsymbol{x}_i)) = \delta(y_i - f(\boldsymbol{x}_i))$ in the limit $\sigma^2 \to 0$. Linearity allows us to write down the joint distribution of noisy function evaluations—that, prior to observation, are treated as random variables $\mathbf{y}$—and the integral Z from (2.8)

$$\begin{bmatrix} \mathbf{y} \\ \mathsf{Z} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{m} \\ \mathfrak{m} \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \sigma^2 \boldsymbol{I} & \int_{\mathcal{X}} \boldsymbol{k}(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}) \\ \int_{\mathcal{X}} \boldsymbol{k}(\boldsymbol{x})^\top \, \mathrm{d}\nu(\boldsymbol{x}) & \iint_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}') \end{bmatrix} \right),$$

Conditioning on the observations $\boldsymbol{y}$ yields the posterior distribution

$$\mathsf{Z} \mid \boldsymbol{y} \sim \mathcal{N}(\mathfrak{m}_{\mathcal{D}}, \mathfrak{v}_{\mathcal{D}})$$

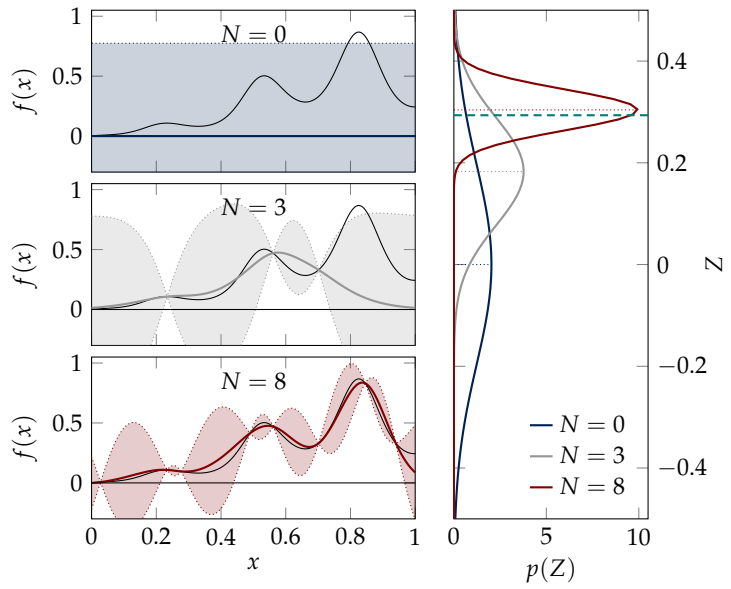


Figure 2.6: Illustration of Bayesian quadrature on our integrand (2.3). *Left column:* The GP f on the integrand with $N = 0, 3$, and 8 exact observations of f (—). Thick colored lines show the GP mean, with the shaded area marking the 95% confidence interval. *Right:* the resulting Gaussian belief over Z. With a zero prior mean function, the initial integral belief ($N = 0$) is centered around zero. With more data, the posterior contracts around the true solution (- - -). Note that in this simple VBQ setting, the known boundedness of the integral has not been encoded in the prior.

with posterior mean $\mathfrak{m}_\mathcal{D} \in \mathbb{R}$ the integral of (2.6)

$$\begin{aligned}\mathfrak{m}_\mathcal{D} &:= \mathbb{E}_{\mathsf{f}\,|\,\mathcal{D}}[\mathsf{Z}] = \int_\mathcal{X} m(\boldsymbol{x}) + \boldsymbol{k}(\boldsymbol{x})^\top (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - \boldsymbol{m})\, \mathrm{d}\nu(\boldsymbol{x}) \\ &= \int_\mathcal{X} m(\boldsymbol{x})\, \mathrm{d}\nu(\boldsymbol{x}) + \sum_{n=1}^{N} \int_\mathcal{X} k(\boldsymbol{x}, \boldsymbol{x}_n)\, \mathrm{d}\nu(\boldsymbol{x}) \left[(\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - \boldsymbol{m})\right]_n \end{aligned} \tag{2.13}$$

and variance $\mathfrak{v}_\mathcal{D} \in \mathbb{R}_+$ as the integral of (2.7) over both arguments

$$\begin{aligned}\mathfrak{v}_\mathcal{D} &:= \mathbb{V}_{\mathsf{f}\,|\,\mathcal{D}}[\mathsf{Z}] = \iint_\mathcal{X} k(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}(\boldsymbol{x})^\top (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}(\boldsymbol{x})\, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}'). \\ &= \iint_\mathcal{X} k(\boldsymbol{x}, \boldsymbol{x}')\, \mathrm{d}\nu(\boldsymbol{x}) - \sum_{nn'} \left[(\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1}\right]_{nn'} \int_\mathcal{X} k(\boldsymbol{x}, \boldsymbol{x}_n)\, \mathrm{d}\nu(\boldsymbol{x}) \int_\mathcal{X} k(\boldsymbol{x}_{n'}, \boldsymbol{x}')\, \mathrm{d}\nu(\boldsymbol{x}'). \end{aligned} \tag{2.14}$$

The second line for mean and variance single out the $\boldsymbol{x}$-dependencies. It therefore becomes clear that besides the integral over the GP prior mean—which is often assumed zero—there are two integrals to be solved, the *kernel mean*, and the *initial variance*

$$\kappa(\boldsymbol{x}) := \int_\mathcal{X} k(\boldsymbol{x}, \boldsymbol{x}')\, \mathrm{d}\nu(\boldsymbol{x}'), \tag{2.15}$$

$$\mathfrak{k} := \iint_\mathcal{X} k(\boldsymbol{x}, \boldsymbol{x}')\, \mathrm{d}\nu(\boldsymbol{x})\, \mathrm{d}\nu(\boldsymbol{x}'). \tag{2.16}$$

The integral over both arguments of the kernel (2.16) quantifies the prior variance, *i.e.*, $\mathfrak{v} = \mathfrak{k}$, and is the squared initial error estimate of the integral. Denoting the column vector $\boldsymbol{\kappa} := \kappa(\boldsymbol{X})$ as the kernel mean evaluated at nodes $\boldsymbol{X}$, we can rewrite (2.13) and (2.14) in the shorthand notation that will reappear throughout the thesis,

$$\mathfrak{m}_\mathcal{D} = \mathfrak{m} + \boldsymbol{\kappa}^\top (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1}(\boldsymbol{y} - \boldsymbol{m}), \tag{2.17}$$

$$\mathfrak{v}_\mathcal{D} = \mathfrak{k} - \boldsymbol{\kappa}^\top (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{\kappa}. \tag{2.18}$$

For a zero prior mean[9] and noise-free observations, the expected value of Z is

$$\mathfrak{m}_\mathcal{D} = \boldsymbol{\kappa}^\top \boldsymbol{K}^{-1} \boldsymbol{f} = \sum_{n=1}^{N} w_n f(\boldsymbol{x}_n)$$

9: Unless otherwise stated, we will assume w.l.o.g. the prior GP mean to be zero.

with weights $\boldsymbol{w} = \boldsymbol{K}^{-1}\boldsymbol{\kappa}$. This is the classic form of a quadrature rule (2.2) that represents the integral as a weighted sum of function evaluations. Figure 2.6 shows the GP prior and the resulting Gaussian distribution over the integral, as well as the updated posteriors on f and Z given three and eight exact evaluations of the integrand, respectively. Pseudocode for this procedure is given in Algorithm 2.1 further below.

BQ replaces the intractable integral (2.1) by integrals over the kernel. To gain a computational advantage, the kernel used for BQ is typically chosen such that the kernel mean and initial variance are available in closed form for the given integration measure ν. Otherwise, the substitute integral for (2.1) is yet another intractable integral. This can nevertheless be beneficial if the integrals over the kernel are numerically much easier to compute than the original integral, *e.g.*, when f is expensive to evaluate.
A selection of kernels that admit closed-form kernel embeddings against some integration measure is listed in Table 2.1 (adapted from [37]). A derivation for the case of a Gaussian kernel and a Gaussian integration measure is included in Section B.1.1. Few kernels can be integrated when $D > 1$, but if the univariate integral is available, the kernel can be constructed as a tensor product over dimensions as $k(\boldsymbol{x}, \boldsymbol{x}') = \prod_{d=1}^{D} k_d(x_d, x'_d)$. This is necessary for the Matérn kernels for example, to retain a closed-form integral. Tronarp et al. [247] established the connection between scale mixtures of the Gaussian kernel and the Matérn kernel. A finite scale mixture is an approximation to the Matérn kernel that permits closed-form kernel embeddings without imposing an independence assumption over dimensions.

k	ν	$\mathcal{X}$
Gaussian	Unif($\mathcal{X}$)	$[\boldsymbol{a}, \boldsymbol{b}]$
	Gaussian	$\mathbb{R}^D$
Matérn$^{\text{TP}}$	Unif($\mathcal{X}$)	$[\boldsymbol{a}, \boldsymbol{b}]$
Wendland$^{\text{TP}}$	Unif($\mathcal{X}$)	$[\boldsymbol{a}, \boldsymbol{b}]$

Table 2.1: Kernel, measure, and domain combinations for kernels from Section 2.1.3 that admit closed-form expressions for κ and $\mathfrak{k}$. Bounds $\boldsymbol{a}, \boldsymbol{b}$ need to be finite and $a_d < b_d$ for $d = 1, \ldots, D$. TP denotes 'tensor product', where only the integrals of kernels with univariate arguments are available. More integrable kernels and references for derivations are listed in [37]. Integrals for the Wendland kernel can also be found in [118].

[37]: Briol et al. (2019), 'Probabilistic integration: A role in statistical computation?' Table 1

[247]: Tronarp et al. (2018), 'Mixture representation of the Matérn class with applications in state space approximations and Bayesian quadrature'

Algorithm 2.1 Bayesian quadrature

procedure BayesQuad($f(\cdot)$, $\mathsf{f} \sim \mathcal{GP}(m, k)$, $\nu(\cdot)$, $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$, σ^2)
  $\boldsymbol{\kappa} \leftarrow \int_{\mathcal{X}} k(\boldsymbol{X}, \boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x})$ // Compute kernel mean
  $\mathfrak{k} \leftarrow \iint_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}')$ // Compute kernel variance
  $\mathfrak{m} \leftarrow \int_{\mathcal{X}} m(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x})$ // Integrate prior mean
  $\boldsymbol{m} \leftarrow m(\boldsymbol{X})$
  $\boldsymbol{w} \leftarrow (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{\kappa}$ // Compute quadrature weights
  $\mathfrak{m}_{\mathcal{D}} \leftarrow \mathfrak{m} + \boldsymbol{w}^\top (\boldsymbol{y} - \boldsymbol{m})$ // Compute BQ mean
  $\mathfrak{v}_{\mathcal{D}} \leftarrow \mathfrak{k} - \boldsymbol{w}^\top \boldsymbol{\kappa}$ // Compute BQ variance
  $p(Z \mid \boldsymbol{y}) \leftarrow \mathcal{N}(\mathfrak{m}_{\mathcal{D}}, \mathfrak{v}_{\mathcal{D}})$
  return $p(Z \mid \boldsymbol{y})$
end procedure

2.3 Connections to kernel quadrature

The notion of Gaussian processes is closely tied to frequentist kernel methods. As suggested by Figure 2.5, properties of samples from the Gaussian process depend on the choice of covariance function. In fact, each kernel uniquely identifies a function space that is called a *reproducing kernel Hilbert space (RKHS)* and defines the hypothesis space of functions that can be represented in terms of the chosen kernel. [10]

Kernel methods are—just as GPs—concerned with modeling non-linear functional relationships. In contrast to the Bayesian inference approach taken with GPs, the task of interest is phrased as an *empirical risk minimization* problem in statistical machine learning. The task is defined via a *loss function* that penalizes deviations of predictions from the true output.

10: A subtlety is that *samples* of a GP do *not* lie in the RKHS almost surely, but in a 'slightly expanded' RKHS (see Kanagawa et al. [129], Section 4)

The prediction or *estimator* arises as the minimizer of the expected loss on the available data set within a hypothesis space over functions—an RKHS.

RKHSs unite GPs and kernel methods, despite disparate modeling approaches. The equivalences and differences between the probabilistic and the frequentist view on kernels are discussed in a recent review by Kanagawa et al. [129] in a manner accessible to both communities. Here we provide a superficial overview of the rich theory behind reproducing kernel Hilbert spaces and state the main connections between kernel and Bayesian quadrature, largely following [129].

[129]: Kanagawa et al. (2018), 'Gaussian processes and kernel methods: A review on connections and equivalences'

2.3.1 Reproducing kernel Hilbert spaces

Definition 2.3.1 *Let $\mathcal{H}_k$ be a Hilbert space with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$ and norm $\| \cdot \|_{\mathcal{H}_k}$. $\mathcal{H}_k$ is said to be a reproducing kernel Hilbert space (RKHS) if there exists a symmetric, positive definite kernel that satisfies*

(i) $k(\cdot, \boldsymbol{x}) \in \mathcal{H}_k$ *and*

(ii) $f(\boldsymbol{x}) = \langle f, k(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}_k}$ $\forall \boldsymbol{x} \in \mathcal{X}$ *and* $f \in \mathcal{H}_k$ *(reproducing property).*

The *Moore-Aronszajn theorem* states that any symmetric positive definite kernel k uniquely defines an RKHS, and vice versa, every RKHS has an associated reproducing kernel [9]. The map $k(\cdot, \boldsymbol{x})$ is called *canonical feature map* of $\boldsymbol{x}$, as the reproducing property allows it to be written as $k(\boldsymbol{x}, \boldsymbol{x}') = \langle k(\cdot, \boldsymbol{x}), k(\cdot, \boldsymbol{x}') \rangle_{\mathcal{H}_k}$.
It is useful to gain an understanding about the functions that are contained in a particular RKHS. Functions that live in the RKHS induced by the kernel k are expandable as a series w.r.t. k, which in turn allows the RKHS to be written as the collection of all such functions, [11]

[9]: Aronszajn (1950), 'Theory of reproducing kernels'

$$\mathcal{H}_k = \left\{ f = \sum_{i=1}^{\infty} a_i k(\cdot, \boldsymbol{x}_i) \colon (a_1, \dots) \subset \mathbb{R}, (\boldsymbol{x}_1, \dots) \subset \mathcal{X}, \text{ s.t. } \|f\|^2_{\mathcal{H}_k} < \infty \right\}.$$

11: In fact, $\mathcal{H}_k$ is the closure of a *pre-Hilbert space* that is defined as the linear span of feature vectors, $\mathcal{H}_0 := \text{span}\,(k(\cdot, \boldsymbol{x}) : \boldsymbol{x} \in \mathcal{X})$, which allows member functions to be represented as finite feature expansions, see [129] p. 11.

From this definition, it can be seen that functions in the RKHS inherit their properties from the associated reproducing kernels. If the kernel k is bounded, every $f \in \mathcal{H}_k$ is also bounded, and if k is continuous at every $\boldsymbol{x} \in \mathcal{X}$, so is every $f \in \mathcal{H}_k$. Furthermore, if k is m times continuously differentiable, so is every $f \in \mathcal{H}_k$ [237] . The RKHS norm is an indicator for the smoothness of functions: As $\|f\|_{\mathcal{H}_k}$ decreases, the variability of f decreases and it is smoother.

[237]: Steinwart and Christmann (2008), *Support vector machines* Corollary 4.36

2.3.2 Kernel ridge regression

A regularized empirical risk minimization problem takes the form

$$\hat{f} = \underset{f \in \mathcal{H}_k}{\arg\min} \frac{1}{N} \sum_{n=1}^{N} \mathfrak{L}(\boldsymbol{x}_n, y_n, f(\boldsymbol{x}_n)) + \xi \|f\|^2_{\mathcal{H}_k} \tag{2.19}$$

with the loss function $\mathfrak{L} : \mathcal{X} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ and regularization constant $\xi \in \mathbb{R}_+$ that prevents overfitting by penalizing functions that have a large

RKHS norm. The estimator in *kernel ridge regression (KRR)* arises as unique solution to (2.19) under the square loss, $\mathfrak{L}(\boldsymbol{x}, y, f(\boldsymbol{x})) = (f(\boldsymbol{x}) - y)^2$,

$$\hat{f} = \boldsymbol{k}(\boldsymbol{x})^\top (\boldsymbol{K} + N\xi \boldsymbol{I})^{-1} \boldsymbol{y}, \tag{2.20}$$

imposing that $\boldsymbol{K} + N\xi\boldsymbol{I}$ be invertible. The estimator (2.20) is obtained via the *representer theorem* [226] that implies that the solution to (2.19) can be written as a weighted sum of feature vectors $\hat{f} = \sum_{n=1}^{N} w_n k(\cdot, \boldsymbol{x}_n)$ with $w_n \in \mathbb{R}$, $i = n, \ldots, N$. Substituting $\mathcal{N}\xi = \sigma^2$, the estimator in kernel ridge regression is equivalent to the GP posterior mean (2.6) [129, Proposition 3.6]. Furthermore, from the representation of the posterior mean as a linear combination of feature vectors $k(\cdot, \boldsymbol{x}_i)$, it can be seen that $\boldsymbol{m}_\mathcal{D} \in \mathcal{H}_k$. The equivalence of the kernel ridge regressor and the GP posterior mean also holds in the noise-free (or unregularized) case, called *kernel interpolation*, although a small regularization constant is often used in practice to ensure numerical stability [259]—the equivalent to the jitter introduced in Section 2.1.2.

[226]: Schölkopf et al. (2001), 'A generalized representer theorem'

[259]: Wendland and Rieger (2005), 'Approximate interpolation with applications to selecting smoothing parameters'

Gaussian processes come with a notion of uncertainty: the marginal posterior variance quantifies the expected squared deviation of the function from the mean under the posterior, $\mathbb{V}_{\mathsf{f}\,|\,\mathcal{D}}[\mathsf{f}] = \mathbb{E}_{\mathsf{f}\,|\,\mathcal{D}}[(\mathsf{f}(\boldsymbol{x}) - m_\mathcal{D}(\boldsymbol{x}))^2]$. The standard deviation can thus be interpreted as an *average-case error* as a function of location $\boldsymbol{x}$. In absence of a notion of uncertainty, kernel methods come with alternative measures for error. The *worst-case error (WCE)* is a prevalent error estimate and defined as

$$\varepsilon_{\text{WCE}}(\boldsymbol{x}) = \sup_{f \in \mathcal{H}_k : \|f\|_{\mathcal{H}_k} \leq 1} (f(x) - m_\mathcal{D}(\boldsymbol{x}))\,.$$

It quantifies the largest possible error as the difference of the worst adversary function with at most unit norm in the RKHS and the KRR estimator.[12] Kanagawa et al. [129] showed that the worst-case error in the RKHS is equal to the noise-corrected standard deviation of the GP, $\varepsilon_{\text{WCE}}(\boldsymbol{x}) = \sqrt{k_\mathcal{D}(\boldsymbol{x}, \boldsymbol{x}) + \sigma^2}$, where equality is reached for the noise-free case, and thus in general $\varepsilon_{\text{WCE}} \geq \sqrt{k_\mathcal{D}(\boldsymbol{x}, \boldsymbol{x})}$.

12: With the established equality to the GP posterior mean we have denoted the KRR estimator $\hat{f}$ as $m_\mathcal{D}$.

[129]: Kanagawa et al. (2018), 'Gaussian processes and kernel methods: A review on connections and equivalences' Proposition 3.8

2.3.3 Quadrature rules in the RKHS

Consider again the intractable integral (2.1). The equivalent of Bayesian quadrature in the RKHS language is *kernel quadrature*. Kernel quadrature is concerned with finding quadrature rules where the hypothesis space for the integrand f is an RKHS. The restriction of the hypothesis space to a certain RKHS is a way to impose regularity assumptions on f. We first discuss the evaluation of performance of a *given* quadrature rule with the underlying assumption that $f \in \mathcal{H}_k$ with reproducing kernel k. Having established a notion of error, we can set out to find an *optimal* rule that determines the estimator in kernel quadrature.

Worst-case error and maximum mean discrepancy Formally, we assume $(\mathcal{X}, \mathfrak{B})$ to be a measurable space with Borel σ-algebra $\mathfrak{B}$ and ν a probability measure on $\mathcal{X}$. Let further k be a bounded, measurable kernel on $\mathcal{X}$ that induces the RKHS $\mathcal{H}_k$ and satisfies $\int_\mathcal{X} \sqrt{k(\boldsymbol{x}, \boldsymbol{x})}\, \mathrm{d}\nu(\boldsymbol{x}) < \infty$. The

worst-case error possibly achieved by a quadrature rule $Q(f; \boldsymbol{X}, \boldsymbol{w})$ (2.2) with weights $w_n = [\boldsymbol{w}]_n \in \mathbb{R},\ n = 1, \ldots, N$, can be written as

$$\varepsilon_{\text{WCE}}(f; \boldsymbol{X}, \boldsymbol{w}) = \sup_{f \in \mathcal{H}_k : \|f\|_{H_k} \leq 1} \left| \int_{\mathcal{X}} f(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}) - Q(f; \boldsymbol{X}, \boldsymbol{w}) \right|. \quad (2.21)$$

Fortunately, (2.21) can be brought into a tractable form using properties of the RKHS. To this end, we revisit the *kernel mean* or *mean embedding*, (2.15),

$$\kappa_\nu := \int_{\mathcal{X}} k(\cdot, \boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}) \in \mathcal{H}_k$$

and refine the notation of the kernel mean by specifying the previously omitted integration measure ν. Using the reproducing property, the target integral (2.1) can be rewritten as the inner product of f with κ_ν,

$$\int_{\mathcal{X}} f \, \mathrm{d}\nu = \int_{\mathcal{X}} \langle f, k(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}_k} \, \mathrm{d}\nu(\boldsymbol{x}) = \left\langle f, \int_{\mathcal{X}} k(\cdot, \boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}) \right\rangle_{\mathcal{H}_k} = \langle f, \kappa_\nu \rangle_{\mathcal{H}_k} \quad (2.22)$$

for any $f \in \mathcal{H}_k$. [13] Any quadrature rule $Q(f; \boldsymbol{X}, \boldsymbol{w})$ (2.2) can be considered an integral w.r.t. the empirical measure $\tilde{\nu} = \sum_i w_i \delta_{\boldsymbol{x}_i}$ where $\delta_{\boldsymbol{x}_i}$ denotes the Dirac distribution. Therefore, we can also define an empirical kernel mean w.r.t. the approximate measure $\tilde{\nu}$ that we denote $\kappa_{\tilde{\nu}}$. Proceeding as in (2.22), the quadrature rule takes the form

$$Q(f; \boldsymbol{X}, \boldsymbol{w}) = \langle f, \kappa_{\tilde{\nu}} \rangle_{\mathcal{H}_k}$$

and, consequentially,

$$\int_{\mathcal{X}} f(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}) - Q(f; \boldsymbol{X}, \boldsymbol{w}) = \langle f, \kappa_\nu - \kappa_{\tilde{\nu}} \rangle_{\mathcal{H}_k}.$$

Applying the Cauchy-Schwartz inequality leaves us with the following expression for the worst-case error (2.21):

$$\varepsilon_{\text{WCE}}(f; \boldsymbol{X}, \boldsymbol{w}) = \|\kappa_\nu - \kappa_{\tilde{\nu}}\|_{\mathcal{H}_k}.$$

The worst achievable discrepancy between the integral and the quadrature rule is thus equivalent to the distance between the associated kernel means. This quantity is also known as *maximum mean discrepancy (MMD)* $\text{MMD}(\nu, \tilde{\nu}; \mathcal{H}_k)$. For *characteristic* kernels—kernels where the mapping $\nu \mapsto \kappa_\nu$ is injective for any ν on $\mathcal{X}$—the MMD becomes a metric on probability spaces and allows quantification of the deviation of an approximating measure $\tilde{\nu}$ (not restricted to empirical measures as for quadrature rules) from the target measure ν.

Optimal weights recover BQ The optimal quadrature rule is the one that minimizes the worst-case error. From the perspective of MMD, we try to find the *optimal empirical measure* $\tilde{\nu}$ to approximate the integration measure ν. Assume N function evaluations $\boldsymbol{f} = [f(\boldsymbol{x}_1) \ldots f(\boldsymbol{x}_N)]^\top$ to be given at fixed design points $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$. The remaining degree of freedom are the quadrature weights $\boldsymbol{w}$, which are found by minimizing the WCE—or, because the WCE is positive, we can conveniently minimize

13: The inner product and the integral commute when κ is a Bochner integral (see [37] 2.2, [237] A.5.20).

[37]: Briol et al. (2019), 'Probabilistic integration: A role in statistical computation?'

[237]: Steinwart and Christmann (2008), *Support vector machines*

its square

$$\begin{aligned}\varepsilon^2_{\text{WCE}} &= \langle \kappa_\nu - \kappa_{\tilde{\nu}}, \kappa_\nu - \kappa_{\tilde{\nu}} \rangle_{\mathcal{H}_k} \\ &= \langle \kappa_\nu, \kappa_\nu \rangle_{\mathcal{H}_k} - 2\langle \kappa_{\tilde{\nu}}, \kappa_\nu \rangle_{\mathcal{H}_k} + \langle \kappa_{\tilde{\nu}}, \kappa_{\tilde{\nu}} \rangle_{\mathcal{H}_k} \\ &= \|\kappa_\nu\|_{\mathcal{H}_k} - 2\boldsymbol{w}^\top \langle k(\cdot, \boldsymbol{X}), \kappa_\nu \rangle_{\mathcal{H}_k} + \boldsymbol{w}^\top \langle k(\cdot, \boldsymbol{X}), k(\cdot, \boldsymbol{X}) \rangle_{\mathcal{H}_k} \boldsymbol{w} \\ &= \mathfrak{k}_\nu - 2\boldsymbol{w}^\top \boldsymbol{\kappa}_\nu + \boldsymbol{w}^\top \boldsymbol{K} \boldsymbol{w}\end{aligned}$$

where $\boldsymbol{\kappa}_\nu = [\kappa_\nu(\boldsymbol{x}_1), \ldots, \kappa_\nu(\boldsymbol{x}_N)]^\top$ and $\mathfrak{k}_\nu$ the twice integrated kernel (2.16). It is easily seen that this simple quadratic form in $\boldsymbol{w}$ is uniquely minimized by the weights

$$\boldsymbol{w}_k = \boldsymbol{K}^{-1} \boldsymbol{\kappa}_\nu$$

provided $\boldsymbol{K}$ is invertible. The *kernel quadrature rule* is therefore

$$Q_k(f; \boldsymbol{X}) = \boldsymbol{\kappa}_\nu^\top \boldsymbol{K}^{-1} \boldsymbol{f}.$$

Plugging the optimal weights back into the definition of the worst-case error leaves us with

$$e^2 = \mathfrak{k}_\nu - \boldsymbol{\kappa}_\nu^\top \boldsymbol{K}^{-1} \boldsymbol{\kappa}_\nu.$$

Comparing these expressions to the ones obtained via the Bayesian inference approach taken by BQ, (2.17) and (2.18), we conclude that the kernel quadrature estimator equals the expected value for the integral found by BQ in the noise-free setting when the prior mean function of the GP is set to zero. The WCE is the standard deviation found by BQ as $\sigma^2 \to 0$. Let us halt for a moment to appreciate this remarkable equivalence. On the one hand, it implies that BQ is the unique optimal quadrature rule for a given set of design points [37]. Any other quadrature rule delivers a point estimate that is inferior over the BQ rule as it causes a larger error. On the other hand, the BQ variance, which quantifies uncertainty arising from finite computations, actually provides a meaningful notion of discretization error in the 'classical' sense. The equivalence further leverages the transfer of results from the rich RKHS literature to establish theoretical guarantees about BQ.

[37]: Briol et al. (2019), 'Probabilistic integration: A role in statistical computation?' §2.3

2.3.4 Convergence of kernel quadrature rules

In order to assess the performance of kernel (or Bayesian) quadrature rules, it is instructive to derive their *convergence rate*. The convergence rate quantifies how the integration error decays as more nodes are included. We only provide a superficial overview of the key results found in the literature and refer the reader to excellent summaries in [132, 38], which we roughly follow. A statement about the convergence rate of a quadrature rule requires assumptions on properties of the kernel as well as the nodes. The latter should be 'spread out' to cover the integration domain. Quantitatively speaking, the nodes should have a small *fill distance*

$$h_{\boldsymbol{X}} = \sup_{\boldsymbol{x} \in \mathcal{X}} \min_{n=1,\ldots,N} \|\boldsymbol{x} - \boldsymbol{x}_n\|.$$

The fill distance finds the largest possible ball within the domain $\mathcal{X}$ in which no node is contained, and $h_{\boldsymbol{X}} \to 0$ as $N \to \infty$. In a quasi-uniform grid $h_{\boldsymbol{X}}$ decays at the dimension-dependent rate of $\mathcal{O}(N^{-1/D})$.

[132]: Karvonen (2019), *Kernel-based and Bayesian methods for numerical integration; Ydinperusteiset ja bayesilaiset menetelmät numeerisessa integroinnissa*

[38]: Briol (2018), 'Statistical computation with kernels'

Convergence rates can be derived in terms of the fill distance. Choosing integration nodes in a way that the fill distance reduces quickly is therefore essential to obtaining good convergence rates. A key assumption for the derivation of upper bounds on the posterior variance is that the integrand lives in the RKHS represented by the kernel that specifies the GP prior. Not every kernel achieves the same rate on members of its RKHS: the smoothness γ affects the convergence behavior, with larger smoothness leading to better convergence rates. This motivates the consideration of Sobolev kernels, which carry an associated smoothness parameter.[14] For Sobolev kernels of smoothness γ, quasi-uniform nodes, and further assumptions on the domain $\mathcal{X}$ and measure ν (see [132, Theorem 2.18, 2.19]), the integration error contracts as

$$|Z - Q_k(f; \boldsymbol{X})| \propto \mathcal{O}(N^{-\gamma/D})$$

for sufficiently large N. Kanagawa et al. [130] studied the effect of using a quadrature rule with kernel of smoothness γ on an integrand that comes from a Sobolev space of smoothness $\gamma' \leq \gamma$ and found the convergence rate to be $\mathcal{O}(N^{-\gamma'/D})$. Hence, the convergence rate depends on the smoothness of the integrand, not on the model hypothesis. Under the assumptions needed for convergence analysis, it is also possible to show that the BQ posterior contracts around the true solution as $N \to \infty$, which tells us that the BQ estimator is consistent [38, §3.3].
Contraction rates depend on how the point set was acquired, which is the subject matter of Chapter 3, where convergence results are further discussed, as well as the implication for practical BQ algorithms.

14: Sobolev kernels are stationary kernels with smoothness γ, when their corresponding RKHS is norm-equivalent to a Sobolev space $\mathcal{H}^\gamma$ (see, *e. g.*, [38, §A.1] for a definition). For example, the RKHS induced by Matérn kernels with smoothness γ fall into this category.

[130]: Kanagawa et al. (2016), 'Convergence guarantees for kernel-based quadrature rules in misspecified settings'

Discussion Any skepticism whether a Bayesian way to think about numerical integration produces a meaningful quadrature rules should be dispersed with the established connection to kernel quadrature and its well-founded theory. At first glance, the unique link between a kernel and its RKHS makes the distinction between thinking in terms of a hypothesis class or about a prior (*ergo* a kernel) obsolete.
There is indeed no distinction for VBQ quadrature rules. The different computational approaches to finding the weights, *optimization* in kernel quadrature and *conditioning* in BQ has consequences when constructing specialized quadrature rules. In kernel quadrature, it is easy to impose constraints on the quadrature weights by including the desired constraints such as non-negativity in the optimization procedure. Such constraints on the weights cannot be imposed in BQ. The sign and magnitude of BQ weights depend on properties of the kernel as well as the locations of the quadrature nodes. At best, purely positive weights can be constructed by a careful selection of a kernel and appropriate nodes [133].
Conversely, Bayesian quadrature admits the choice of priors that deviate from the Gaussian assumption on the integrand and can therefore not easily be transferred to a kernel quadrature equivalent (see Section 2.5).

[133]: Karvonen et al. (2019), 'On the positivity and magnitudes of Bayesian quadrature weights'

[60]: Davis and Rabinowitz (1983), *Methods of numerical integration*

2.4 Connections to classical numerical integration

Classical numerical analysis has brought forth a wealth of algorithms for numerical integration of primarily univariate functions [60].[15] Most commonly used integration schemes fall under two broad classes of univariate numerical integration schemes: interpolating splines and

15: Bivariate and trivariate extensions exist and are mostly considered separately in the classical literature of numerical analysis. Higher dimensional problems (mostly) fall victim to the *curse of dimensionality*.

Gaussian quadratures based on polynomials. These methods have been studied extensively and efficient implementations are available in toolboxes such as the `Fortran` library `quadpack` [204]. A natural question arises when viewing numerical integration from the probabilistic angle:

> *Is there any correspondence between classical and probabilistic quadrature schemes?*

Both methods can be considered equivalent if the posterior mean found by BQ matches the point estimate given by the classical quadrature rule. Individual correspondences have been discovered in the early days of probabilistic numerics, e.g. the probabilistic trapezoidal rule [66]. A later overview by Minka [175] contains a probabilistic perspective on both spline and polynomial-based quadrature rules. Further research into such connections has been conducted by Karvonen, who provides an excellent review [132].

[204]: Piessens et al. (2012), *QUADPACK: A subroutine package for automatic integration*

[66]: Diaconis (1988), 'Bayesian numerical analysis'

[175]: Minka (2000), *Deriving quadrature rules from Gaussian processes*

[132]: Karvonen (2019), *Kernel-based and Bayesian methods for numerical integration; Ydinperusteiset ja bayesilaiset menetelmät numeerisessa integroinnissa*

2.4.1 Spline interpolation

Many popular classical integration schemes build on interpolating the integrand by piecewise polynomials that can be integrated in closed form. Let P_M denote all polynomials with degree up to $M \in \mathbb{N}$ and consider the domain $\mathcal{X} = [0, 1]$. A univariate, natural polynomial spline $s_N^M(x) : [a, b] \to \mathbb{R}$ on N knots $a < x_1 < x_2 < \cdots < x_N < b$ has the properties *(i)* $s_N^M \in P_{M-1}$ if $x \in [a, x_1]$ or $x \in [x_N, b]$, *(ii)* $s_N^M \in P_{2M-1}$ on the intervals $x \in [x_n, x_{n+1}]$, $n = 1, \ldots, N-1$, and *(iii)* $s_N^M \in C^{2M-2}$ for $x \in \mathbb{R}$, *i.e.*, the class of functions with $2M-2$ continuous derivatives. Its $2MN$ parameters are uniquely determined by the continuity condition (iii) and the interpolation condition $s_N^M(\boldsymbol{X}) = \boldsymbol{f}$. Spline interpolation was first introduced by Schoenberg [224, 225]. The relations to kernel methods are extensively treated by Wahba [253]. In particular, odd spline interpolants of degree $2M+1$ are recovered—up to technicalities (see *e.g.*, Karvonen [132, §5.5])—by the kernel interpolant using the M times integrated Brownian motion kernel (2.11) [253, §1.3]. Knowing that the kernel interpolant and *ergo* the GP posterior mean equal a spline interpolator, it is not surprising that spline-based quadrature rules of odd order are recovered by BQ rules with said kernels.[16] Let us consider the (0^{th} order) Brownian motion kernel (2.10) as simplest and instructive example [66, 51, 108, 112].

[224]: Schoenberg (1946), 'Contributions to the problem of approximation of equidistant data by analytic functions: Part A. On the problem of smoothing or graduation. A first class of analytic approximation formulae'

[225]: Schoenberg (1946), 'Contributions to the problem of approximation of equidistant data by analytic functions. Part B. On the problem of osculatory interpolation. A second class of analytic approximation formulae'

[253]: Wahba (1990), *Spline models for observational data*

16: Even-order splines, however, lack a natural equivalent in the kernel world. Simpson's rule, relying on quadratic splines, does therefore not have a probabilistic counterpart [66].

[66]: Diaconis (1988), 'Bayesian numerical analysis'

[51]: Cockayne et al. (2019), 'Bayesian probabilistic numerical methods'

[108]: Hennig et al. (2015), 'Probabilistic numerics and uncertainty in computations'

[112]: Hennig et al. (2022), *Probabilistic Numerics: Computation as Machine Learning*

Example 2.4.1 *The probabilistic trapezoidal rule* Let $\mathsf{f} \sim \mathcal{GP}(0, k)$ with $k(x, x') = \min(x, x')$ and $x, x' \in [0, 1]$.[17] , Using that $0 \leq x \leq 1$ and $\int_0^1 \min(x, x')\,\mathrm{d}x' = \int_0^x x'\,\mathrm{d}x' + \int_x^1 x\,\mathrm{d}x'$, the kernel embeddings are

$$\kappa(x) = \int_0^1 k(x, x')\,\mathrm{d}x' = x - \frac{1}{2}x^2 \quad \text{and} \quad \mathfrak{k} = \frac{1}{3}.$$

Conditioning on $\boldsymbol{f}$ at N knots where $x_1 = 0$ and $x_N = 1$, the posterior mean is a linear function between the knots. To see this, assume that

17: Generalization to other integration domains $[a, b]$ is possible by adding a shift to the kernel. See [112] for a detailed treatment.

$x \in [x_j, x_{j+1}]$ and define kernel weights $\boldsymbol{w} = \boldsymbol{K}^{-1}\boldsymbol{f}$. Then (2.6) is

$$m_{\mathcal{D}}(x) = \sum_{i=1}^{j} w_i x + \sum_{i=j+1}^{N} w_i x_i = f(x_j) + \frac{x - x_j}{x_{j+1} - x_j}\left(f(x_{j+1}) - f(x_j)\right),$$

which is the unique linear spline interpolator. Integration of the posterior mean yields

$$\mathfrak{m}_{\mathcal{D}} = \sum_{i=1}^{N-1} \frac{1}{2}(x_{j+1} - x_j)\left(f(x_{j+1}) - f(x_j)\right).$$

This is the *trapezoidal rule* [60], illustrated in Figure 2.7. The posterior variance of the integral is $\mathfrak{v}_{\mathcal{D}} = \sum_{i=1}^{N-1} \frac{(x_{i+1}-x_i)^3}{12}$. Placing the nodes such that the variance is minimized results in an equidistant grid with $x_i = \frac{2i}{2N+1}$. The standard deviation $\sqrt{\mathfrak{v}_{\mathcal{D}}}$ contracts as the classical rate in $\mathcal{O}(N^{-1})$.

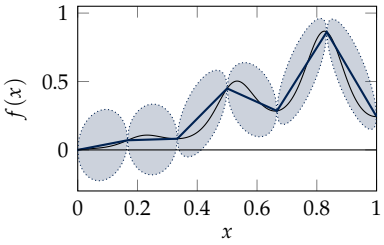


Figure 2.7: The interpolant of a Wiener process is piecewise linear and its integral is equivalent to the trapezoidal rule.

[60]: Davis and Rabinowitz (1983), *Methods of numerical integration*

GP models with Brownian motion kernels are *Gauss-Markov processes* and can be phrased as solutions of linear stochastic differential equations with Gaussian initial conditions. Inference in this representation is linear (instead of cubic) in time and relies on Kalman filtering [219]. Bayesian spline quadrature rules are thus available at the same computational complexity as their classic counterparts.

[219]: Särkkä (2013), *Bayesian filtering and smoothing*

2.4.2 Polynomial and Gaussian quadrature

Polynomial interpolation models a function f that has been evaluated at N distinct, but arbitrary locations $\boldsymbol{X}$ by a single polynomial of degree $N-1$ $\varphi_{N-1}(x) = \sum_{n=1}^{N} a_n x^n$. The unique interpolating polynomial is found by solving the linear system $\varphi_{N-1}(x_n) = f(x_n)$ for each $n = 1, \ldots, N$. Such polynomial can be integrated in closed form and thus forms the basis for a *polynomial quadrature rule*. When an equidistant grid is given, *Newton-Cotes rules* apply. They employ Lagrange polynomials as basis functions, but are susceptible to catastrophic oscillations as the number of equally spaced points increase, known as Runge's phenomenon [217]. An example for a popular polynomial quadrature rule is *Clenshaw-Curtis quadrature*, which integrates functions on $\mathcal{X} = [-1, 1]$ against the Lebesgue measure using Chebyshev polynomials [50].

[217]: Runge (1901), 'Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten'

[50]: Clenshaw and Curtis (1960), 'A method for numerical integration on an automatic computer'

Definition 2.4.1 *A quadrature rule $Q(f; \boldsymbol{X}, \boldsymbol{w})$ is said to be of degree M if it exactly integrates every polynomial φ_M of degree M, but is inexact on some polynomial of degree $M+1$.*

The Clenshaw-Curtis rule is of degree $N-1$. A wise choice of these nodes, however, can improve a polynomial quadrature rule to achieve degree $2N-1$ on these N evaluations. Such rules are termed *Gaussian quadrature rules* and are unique for a given integration measure ν. The nodes are given as the roots of the N^{th} polynomial ψ_N that is orthogonal under ν, *i.e.*, $\int_{\mathcal{X}} \psi_i(x)\psi_j(x)\,\mathrm{d}\nu(x) = \delta_{ij}$. Its weights are positive. Examples for Gaussian quadrature rules are listed in Table 2.2.

Bayes-Hermite quadrature, coined by O'Hagan [190], is not in fact the probabilistic twin of Gauss-Hermite quadrature, despite its suggestive name. A Bayesian interpretation of Gaussian quadrature rules is available

[190]: O'Hagan (1991), 'Bayes-Hermite quadrature'

Quadrature rule	$\mathcal{X}$	ν	$\{\psi_i\}$
Gauss-Legendre	$[-1, 1]$	$-$	Legendre
Chebyshev-Gauss	$(-1, 1)$	$1/\sqrt{1-x^2}$	Chebyshev
Gauss-Laguerre	$[0, \infty)$	e^{-x}	Laguerre
Gauss-Hermite	$\mathbb{R}$	e^{-x^2}	Hermite

Table 2.2: A selection of classical Gaussian quadrature rules.

through polynomial kernels [135]. The polynomial kernel of degree Q is $k(x, x') = \sum_{q=0}^{Q-1} \varphi_q(x)\varphi_q(x')$ with polynomials φ_q defined as above. The Bayesian quadrature rule with this kernel coincides with a classical quadrature rule of degree $M-1$ if and only if $N \leq Q \leq M$ [135, Theorem 4]. However, because of the orthogonality property of the polynomials, the posterior variance of the Bayesian quadrature rule vanishes, and with it the added value provided by BQ.

[135]: Karvonen and Särkkä (2017), 'Classical quadrature rules via Gaussian processes'

Another novel interpretation of a classical polynomial quadrature rule is Bayes-Sard cubature [134]. It is a Bayesian replication of a classical method by Sard [218] and relies on a parametric polynomial prior mean function.

[134]: Karvonen et al. (2018), 'A Bayes-Sard cubature method'

[218]: Sard (1949), 'Best approximate integration formulas; best approximation formulas'

2.4.3 Locally adaptive quadrature

Off-the-shelf univariate numerical integration algorithms rely on *adaptive quadrature*. For example, the go-to method for univariate integration in `Python`, `scipy.integrate.quad` defaults to `quadpack`'s adaptive methods [204, 251]. Classical adaptive quadrature rules recursively refine the grid into sub-intervals on which the quadrature rule is applied while the error exceeds a user-defined threshold. In this way, the integration rule adapts to the integrand by affording more evaluations where the integrand is more variable. Adaptive quadrature requires two ingredients, *(i)* a quadrature rule and *(ii)* a strategy to quantify the error.

[204]: Piessens et al. (2012), *QUADPACK: A subroutine package for automatic integration*

[251]: Virtanen et al. (2020), 'SciPy 1.0: Fundamental algorithms for scientific computing in Python'

Algorithm 2.2 Locally adaptive quadrature.

procedure AdapQuad(f, a, b, τ)
  $Q \leftarrow \boldsymbol{w}^\top \boldsymbol{f}$ // quadrature rule
  $\epsilon \leftarrow \left| Q - \int_a^b f(x)\,\mathrm{d}x \right|$ // error estimate
  while $\epsilon > \tau$ **do**
    $Q \leftarrow$ AdapQuad($f, a, \frac{a+b}{2}, \tau/2$) + AdapQuad($f, \frac{a+b}{2}, b, \tau/2$)
  end while
end procedure

A Bayesian version of such an algorithm has been introduced by Fisher et al. [71]. They use a non-stationary model with a location-dependent lengthscale and recursively add evaluations to the dataset by minimizing the expected posterior variance of the integral and then updating the model's hyperparameters until the variance has shrunk to the desired tolerance. A class of *globally* adaptive Bayesian algorithms will be discussed in Chapter 3.

[71]: Fisher et al. (2020), 'A locally adaptive Bayesian cubature method'

2.5 Warped Bayesian quadrature

The Bayesian approach to numerical integration allows prior knowledge to be encoded through the choice of prior. We have seen that some char-

acteristics of the target function are captured by the choice of covariance function in Section 2.1.3. Yet, the Gaussian distribution is defined on the real axis and hence the tacit assumption when using a Gaussian process prior is that f maps to $\mathbb{R}$.
If we pursue the goal of computing a normalization constant of an unnormalized density, we know *a priori* that we are dealing with a strictly positive function and clearly, this is prior knowledge that we want to include. Stochastic processes other than GPs are not a viable solution, due to the lack of favorable properties such as closed-form conditioning, marginalization, and integration.

A workaround to encode such constraints on functions is to employ *warped Gaussian processes* [234]. The task is still the integration of a function $f : \mathcal{X} \to \mathcal{Y} \subset \mathbb{R}$ as in (2.1), but now we consider the range of f to be constrained. Instead of modeling f directly with a GP, we take a detour by defining the random function $\mathsf{g} \sim \mathcal{GP}(m_\mathsf{g}, k_\mathsf{g})$ and a surjective (but not necessarily injective) warping $\mathcal{T} : \mathbb{R} \to \mathcal{Y}$ and let

[234]: Snelson et al. (2004), 'Warped Gaussian processes'

$$\mathsf{f} = \mathcal{T}[\mathsf{g}],$$

which will serve as a surrogate for f. f is also a random process, but not a GP if $\mathcal{T}$ is non-linear. In particular, for any non-trivial warping, the distribution over the integral Z (2.12) cannot be computed analytically, but needs to be approximated. Gunter et al. [102] worked out an algorithm for inference on integrals of non-negative functions with the warping $\mathcal{T} : \mathsf{g} \mapsto \mathsf{g}^2 + c, c \geq 0$. A workflow on how to do inference on affine transformations of warped GPs $\mathcal{L}\mathsf{f}$ has been generalized by Chai and Garnett [44] for various choices of $\mathcal{T}$.

[102]: Gunter et al. (2014), 'Sampling for inference in probabilistic models with fast Bayesian quadrature'

[44]: Chai and Garnett (2019), 'Improving quadrature for constrained integrands'

Function evaluations $f(\boldsymbol{x})$ translate to pseudo-observations of an unconstrained function $g : \mathcal{X} \to \mathbb{R}$ as $g(\boldsymbol{x}) = \mathcal{T}^{-1}[f](\boldsymbol{x})$. The GP g is then conditioned on the pseudo-data $\mathcal{D} = \{\boldsymbol{X}, g_{\boldsymbol{X}}\}$ to obtain the Gaussian posterior $p(\mathsf{g} \mid \mathcal{D})$. The warping induces a posterior belief on f. The inference step therefore takes the detour over the tractable process g. Approximation are needed once further operations on f are of interest, as is the case for integration. Then its density can be approximated by a GP via moment matching, such that $p(\mathsf{f} \mid \mathcal{D}) \approx \mathcal{GP}(m_{\mathsf{f}\mid\mathcal{D}}, k_{\mathsf{f}\mid\mathcal{D}})$ with the first two moments

$$m_{\mathsf{f}\mid\mathcal{D}}(\boldsymbol{x}) = \mathbb{E}_{\mathsf{g}\mid\mathcal{D}}\left[\mathcal{T}[\mathsf{g}](\boldsymbol{x})\right], \tag{2.23}$$

$$k_{\mathsf{f}\mid\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{C}_{\mathsf{g}\mid\mathcal{D}}\left[\mathcal{T}[\mathsf{g}](\boldsymbol{x}), \mathcal{T}[\mathsf{g}](\boldsymbol{x}')\right] \tag{2.24}$$

that are functions of the posterior mean and covariance of g, $m_{\mathsf{g}\mid\mathcal{D}}$ and $k_{\mathsf{g}\mid\mathcal{D}}$. Gunter et al. [102] alternatively suggested to locally linearize the warping function $\mathcal{T}$ by Taylor-expanding around $m_{\mathsf{g}\mid\mathcal{D}}$,

$$\mathcal{T}[\mathsf{g}]\Big|_{\mathsf{g}=m_{\mathsf{g}\mid\mathcal{D}}} \approx \mathcal{T}[m_{\mathsf{g}\mid\mathcal{D}}] + (\mathsf{g} - m_{\mathsf{g}\mid\mathcal{D}})\mathcal{T}'[m_{\mathsf{g}\mid\mathcal{D}}].$$

Doing so turns f into an affine transformation of g and thus also a GP with mean and covariance

$$m^{\text{lin}}_{\mathsf{f}\mid\mathcal{D}}(\boldsymbol{x}) = \mathcal{T}[m_{\mathsf{g}\mid\mathcal{D}}](\boldsymbol{x}), \tag{2.25}$$

$$k^{\text{lin}}_{\mathsf{f}\mid\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}') = \mathcal{T}'[m_{\mathsf{g}\mid\mathcal{D}}](\boldsymbol{x})\, k_{\mathsf{g}\mid\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}')\, \mathcal{T}'[m_{\mathsf{g}\mid\mathcal{D}}](\boldsymbol{x}'). \tag{2.26}$$

Linearization is a viable alternative if the moments of the warping are intractable or difficult to compute. What is evident from the linearization approximation is that the *covariance* of the approximate GP depends on the *mean* of the unwarped GP and, hence, the posterior covariance depends on the data. This dependence on the mean function also arises for moment-matched warpings. This is remarkable because in GP models, the covariance only depends on *locations* where data is observed (*i.e.*, where the function is evaluated), but *not* on the function values themselves. The additional dependence of the covariance on the function evaluations is intuitively desirable; our degree of certainty about a quantity of interest should very much depend on all data, not merely on the input locations. This feature is an additional asset of warped models, besides the possibility to better encode prior knowledge. Its implications for Bayesian quadrature will be discussed in Section 3.3.

Some examples for warpings and their moments are given below. [18]

The square transform $\mathcal{T} : \mathbb{R} \to \mathbb{R}_+, \mathrm{g} \mapsto \mathrm{g}^2 + c$ with a small constant $c \geq 0$ has been studied by Gunter et al. [102] for BQ. The square transform encodes positivity of the target function. The true marginal of $\mathrm{f}(\boldsymbol{x})$ is a non-central χ^2 distribution, which does not permit closed-form integration. This warping is of importance to Chapter 6; hence we state both moment matching and linearization approximations. Moment matching results in

$$m_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x}) = m_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x})^2 + k_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}) + c, \tag{2.27}$$

$$k_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}') = 2k_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}')^2 + 4m_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x})k_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}')m_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x}'), \tag{2.28}$$

and will be referred to as WSABI-M in Chapter 6.[19] Local linearization for WSABI-L yields slightly simpler moments

$$m^{\mathrm{lin}}_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x}) = m_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x})^2 + c,$$

$$k^{\mathrm{lin}}_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}') = 4m_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x})k_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}')m_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x}'). \tag{2.29}$$

The base GP and the linearized and moment-matched approximation of its square are shown in Figure 2.8. Notice the increased variance where the predictive mean is high and the converse when the mean is close to zero. The corresponding covariance matrices are shown in Figure 2.9.
Other polynomial warpings $\mathcal{T}[\mathrm{g}] = \sum_{m=0}^{M} a_m \mathrm{g}^m$ are possible, although odd powers induce an unbounded warping. Moments are available through Isserlis' theorem [122] and are related to finding the appropriate multivariate Hermite polynomial [262].

The exponential transform $\mathcal{T} : \mathbb{R} \to \mathbb{R}_+, \mathrm{g} \mapsto e^{\mathrm{g}}$ is an alternative option to encode positivity of f and has moments

$$m_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x}) = \exp\left(m_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x}) + {}^1\!/\!_2 k_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x})\right)$$

$$k_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}') = m_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x}) \exp\left(k_{\mathrm{g}\,|\,\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}')\right) m_{\mathrm{f}\,|\,\mathcal{D}}(\boldsymbol{x}').$$

Osborne et al. [193] used the exponential transform to model likelihoods, and, as Chai and Garnett [44], advocate the exponential transform to deal with the high dynamic range often encountered in

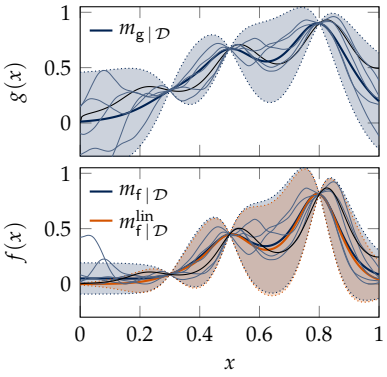


Figure 2.8: *Top:* The base process g with samples after conditioning on three observations of $g = \sqrt{f}$. *Bottom:* The moment-matched and linearized GP of the square of g, yielding a model for f with a mean guaranteed to be ≥ 0. The difference between the approximations stands out where the mean is close to zero. Then the extra covariance term in (2.28) is dominant.

18: In Gunter et al. [102] and Chai and Garnett [44] the names refer to the *inverse* warping, which is more natural when starting from the target function f and searching for a transformation $\mathcal{T}^{-1} : \mathcal{Y} \to \mathbb{R}$ that maps the given output range to the real numbers in order to apply a GP. In order to avoid confusion about mappings and inverse mappings, we term the warping according to their 'forward' definition.

19: Gunter et al. [102] termed their algorithm *warped sequential active Bayesian integration (WSABI)* WSABI-M and WSABI-L specify the approximation used, moment matching, or linearization, respectively. The 'active' aspect is discussed in Section 3.3.

[102]: Gunter et al. (2014), 'Sampling for inference in probabilistic models with fast Bayesian quadrature'
[122]: Isserlis (1918), 'On a formula for the product moment coefficient of any order of a normal frequency distribution in any number of variables'
[262]: Withers (1985), 'The moments of the multivariate normal'
[193]: Osborne et al. (2012), 'Active learning of model evidence using Bayesian quadrature'
[44]: Chai and Garnett (2019), 'Improving quadrature for constrained integrands'

likelihood functions. Especially in the presence of a large amount of i.i.d. data, the likelihood for each datum multiplies and the output can span large magnitudes.

Sigmoidal transforms $\mathcal{T} : \mathbb{R} \to [0,1], \mathsf{g} \mapsto \sigma(\mathsf{g})$ where $\sigma(\cdot)$ is a sigmoidal function such as the logistic function $\sigma(x) = \frac{1}{1+e^{-x}}$, the cumulative Gaussian $\Phi(x)$, or the hyperbolic tangent $\sigma(x) = 1/2\,(\tanh(x)+1)$. These warpings are useful if we know that f is constrained from below *and* above; they can be scaled and shifted to account for more general ranges $\mathcal{Y} = [a,b]$. The cumulative Gaussian offers a closed-form solution for the moments (see [44, Table 1], and [212, §3.9]). Moment computation for sigmoidal warpings is related to the problem of Gaussian process classification that relies on different approximation schemes to overcome intractability of the posterior [212, §3]. Linearization can offer an alternative approach to finding an approximate Gaussian process for f.

[44]: Chai and Garnett (2019), 'Improving quadrature for constrained integrands'
[212]: Rasmussen and Williams (2006), *Gaussian Processes for machine learning*

The BQ estimate for the integral over f is obtained by integrating the approximate Gaussian belief about f as in Section 2.2. BQ demands integration of the approximate mean (2.23) or (2.25) and covariance (2.24) or (2.26) for moment matching or linearization, respectively. Recall that vanilla BQ[20] relies on two integrated quantities only, the kernel mean (2.15) and the kernel integrated over both arguments (2.16). A warping introduces new integrals to be solved. The desideratum that the posterior over the integral be tractable restricts the choice of kernel in a warped GP model even further than in VBQ. For example, the *square transform* additionally requires the following integrals over expressions of the kernel for tractable BQ moments:

$$\begin{aligned}
\int_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}_i) k(\boldsymbol{x}, \boldsymbol{x}_j)\, \mathrm{d}\nu(\boldsymbol{x}) \\
\int_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x})\, \mathrm{d}\nu(\boldsymbol{x}) \\
\iint_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}')^2\, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}') \\
\iint_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}_i) k(\boldsymbol{x}, \boldsymbol{x}') k(\boldsymbol{x}', \boldsymbol{x}_j)\, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}').
\end{aligned} \tag{2.30}$$

The second term is an integral over a constant in stationary kernels and, as well as the third integral, only needed for the moment-matching approximation. These expressions are still manageable for most kernels that fulfil the integrability requirement for BQ, *e.g.*, the Gaussian kernel (2.9).[21] Worse is the *exponential transform* that requires integration of exponentiated expressions of the kernel. Not even product spline kernels are feasible, because the product moves into the exponential. Chai and Garnett [44] propose to Taylor-expand the approximate GP moments and report reasonable empirical results when the warping is dominated by the linear regime of the exponential function.
Polynomial expressions of mean and covariance, arising in the Taylor series, give rise to yet another issue: even if the integral is available in closed form, which is the case for the Gaussian kernel, evaluation of the term of m^{th} order will take $\mathcal{O}(N^{2m})$ floating point operations after N function evaluations have been made. This renders polynomial or Taylor-expanded warpings unfavorable for $m \geq 2$, when the cost exceeds the cubic complexity of GP inference.
As an alternative to approximating the *integrands* to obtain tractable kernel

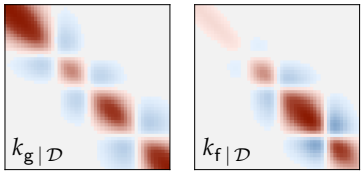


Figure 2.9: The covariance of the original GP *(left)* and of the moment-matched GP *(right)* for the square transform. The diagonal represents the variance that is plotted in Figure 2.8.

20: Vanilla BQ refers to the basic form of BQ introduced in Section 2.2. We refer to VBQ if the clear distinction to warped BQ or any other form of modified probabilistic integration procedure is needed. 'BQ' generally refers to the entity of Bayesian integration schemes.

21: *cf.* Section B.1.2 for a derivation of these integrals for a Gaussian kernel in combination with a Gaussian integration measure.

integrals, (Markov chain) Monte Carlo ((MC)MC) is a viable option for integrating the approximate GP moments. This is beneficial when queries of the surrogate are considerably less time-consuming than evaluating the integrand.

In contrast to VBQ, the integral variance of warped BQ depends on the function evaluations. While moment matching is a sensible representation of the non-Gaussian marginal, the skewness of the distribution is discarded. The interpretation of the variance as worst-case error in the RKHS is sacrificed through the approximation. With the dependence on the data, the variance is not guaranteed to shrink as new data is added. Also the calibration of the variance is less straight-forward due to the scaling with a transformation of the auxiliary GP mean. To obtain a reasonable scaling, hyperparameter optimization should be performed in f-space, and not with the actual GP [44].
The data dependence of the variance motivates BQ algorithms that adaptively select nodes and lack a classical interpretation [102, 128]. We will delve into smart node selection in Chapter 3 and then develop practicable BQ algorithms that rely on warped GPs.

2.6 Related work

Besides the works that are explicitly discussed earlier in this chapter, there has been a considerable amount of work either in developing Bayesian quadrature methods, or applying them to real-world problems. This is a brief, by no means exhaustive summary that highlights some directions that should not go unmentioned.

2.6.1 Specialized BQ methods

BQ commonly assumes a GP prior on the integrand. Other probabilistic models for BQ have been explored, such as trees [270] and Student-t processes [210]. The latter are closely related to GPs and retain the favorable closure properties under conditioning and marginalization [228] (*cf.* Section 2.1.4).
BQ is related to other concepts in machine learning, in particular to kernel herding [121], random feature expansion [18], and MMD (*cf.* Section 2.3.3 and *e.g.*, [206]).
Given the closure of GPs under linear operations, the inclusion of gradient and Hessian information about the integrand is possible with BQ [264, 207]. Other tailored BQ methods have been developed for estimating ratios of related probabilistic integrals with BQ [194]; and for solving intractable integrals in variational mixture models [1, 2]. Nguyen et al. [187] and Iwazaki et al. [123] used a combination of Bayesian optimization and quadrature to optimize expectations of expensive functions to find the $\arg\max_{x \in \mathcal{X}} \int_\Omega f(x, \omega)\, \mathrm{d}\nu(\omega)$.

BQ has also received considerable interest in the signal processing community. Quadrature rules applied to integrals arising in nonlinear Kalman filtering are known as 'sigma-point rules', which are related to BQ for certain covariance functions [220]. BQ has been extensively applied in the context of filtering, *e.g.*, [63, 64, 208, 209]. Gaussian filtering is relevant to probabilistic numerics for the construction of probabilistic ODE solvers, where Kersting and Hennig [139] used BQ for uncertainty calibration.

[44]: Chai and Garnett (2019), 'Improving quadrature for constrained integrands'
[102]: Gunter et al. (2014), 'Sampling for inference in probabilistic models with fast Bayesian quadrature'
[128]: Kanagawa and Hennig (2019), 'Convergence guarantees for adaptive Bayesian quadrature methods'
[270]: Zhu et al. (2020), 'Bayesian probabilistic numerical integration with tree-based models'
[210]: Prüher et al. (2017), 'Student-t process quadratures for filtering of non-linear systems with heavy-tailed noise'
[228]: Shah et al. (2014), 'Student-t processes as alternatives to Gaussian processes'
[121]: Huszár and Duvenaud (2012), 'Optimally-weighted herding is Bayesian quadrature'
[18]: Bach (2017), 'On the equivalence between kernel quadrature rules and random feature expansions'
[206]: Pronzato (2021), 'Performance analysis of greedy algorithms for minimising a Maximum Mean Discrepancy'
[264]: Wu et al. (2017), 'Exploiting gradients and Hessians in Bayesian optimization and Bayesian quadrature'
[207]: Prüher and Särkkä (2016), 'On the use of gradient information in Gaussian process quadratures'
[194]: Osborne et al. (2012), 'Bayesian quadrature for ratios'
[1]: Acerbi (2018), 'Variational Bayesian Monte Carlo'
[2]: Acerbi (2020), 'Variational Bayesian Monte Carlo with noisy likelihoods'
[187]: Nguyen et al. (2020), 'Distributionally robust Bayesian quadrature optimization'
[123]: Iwazaki et al. (2020), 'Bayesian quadrature optimization for probability threshold robustness measure'
[220]: Särkkä et al. (2016), 'On the relation between Gaussian process quadratures and sigma-point methods'
[63]: Deisenroth et al. (2009), 'Analytic moment-based Gaussian process filtering'
[64]: Deisenroth et al. (2011), 'Robust filtering and smoothing with Gaussian processes'
[208]: Prüher and Šimandl (2015), 'Bayesian quadrature in nonlinear filtering'
[209]: Prüher and Šimandl (2016), 'Bayesian quadrature variance in sigma-point filtering'
[139]: Kersting and Hennig (2016), 'Active uncertainty calibration in Bayesian ODE Solvers'

2.6.2 Applications of BQ

BQ has been employed in various settings that benefit from its sample efficiency as well as uncertainty quantification. It has been applied to probabilistic models themselves, *e.g.*, for marginalization over stationary kernels in GP models [103] or for Bayesian model selection [45].

Another application area is reinforcement learning and control [198, 5], where the expected return estimated via BQ improves the robustness of policies. [165] used BQ to actively identify for extended regions that are of interest for evaluation in experimental design. BQ has further been used in fairness [72], human cognition [104], and reliability analysis [269].
A related concept in the GP literature is when data is binned, *i.e.*, it is only possible to observe spatial or temporal averages. These averages manifest as expectations over the kernel [233, 266].

[103]: Hamid et al. (2021), 'Marginalising over stationary kernels with Bayesian quadrature'

[45]: Chai et al. (2019), 'Automated model selection with Bayesian quadrature'

[198]: Paul et al. (2020), 'Robust reinforcement learning with Bayesian optimisation and quadrature'

[5]: Akella et al. (2021), 'Deep Bayesian quadrature policy optimization'

[165]: Ma et al. (2014), 'Active area search via Bayesian quadrature'

[72]: Fitzsimons et al. (2019), 'A general framework for fair regression'

[104]: Hamrick and Griffiths (2013), 'Mental rotation as Bayesian quadrature'

[269]: Zhou and Peng (2020), 'Adaptive Bayesian quadrature based statistical moments estimation for structural reliability analysis'

[233]: Smith et al. (2018), 'Gaussian process regression for binned data'

[266]: Yousefi et al. (2019), 'Multi-task learning for aggregated data using Gaussian processes'

2.6.3 Toolboxes

Bayesian quadrature implementations have not yet reached the stage of being usable as off-the-shelf methods. Besides specific implementations linked to individual publications, there are a few efforts that are worth highlighting:

- github.com/OxfordML/bayesquad [252] contains an implementation of WSABI with various batch selection schemes (*cf.* Chapter 3). However, it natively only supports the Gaussian kernel with a Gaussian measure and therefore leaves little freedom to choose a model.
- A model-agnostic implementation is available at emukit.github.io/ [196]. `emukit` is designed in a modular way such that functionality can easily be added for a new probabilistic model. VBQ and WSABI are, however, only available with the Gaussian kernel under a Gaussian or Lebesgue measure.
- A recent collaborative effort to build a joint library for probabilistic numerical methods, termed `probnum`, is available at probabilistic-numerics.org [261]. `probnum` has a rudimentary version of BQ implemented at the time of writing. The `probnum.quad` package is under active development and envisaged to contain a larger variety of kernel embeddings, BQ methods, and point selection schemes (*cf.* Chapter 3) than previous libraries.

[252]: Wagstaff et al. (2018), 'Batch selection for parallelisation of Bayesian quadrature'

[196]: Paleyes et al. (2019), 'Emulation of physical processes with emukit'

[261]: Wenger et al. (2021), *ProbNum: Probabilistic numerics in Python*

Outlook

This chapter motivated an inferential approach to numerical integration and placed Bayesian quadrature into context with established classical methods for integration. The connection to kernel quadrature validates the Bayesian approach and permits the transfer of theoretical results from the rich theoretical work on kernels in statistics and machine learning.
A key desideratum in BQ is that the kernel be integrable in closed form, because the intractable integral (2.1) is replaced by integrals over the kernel. This requisite is relaxed when the target integral involves an integrand that is expensive to evaluate and there is a computational benefit even if the surrogate needs to be integrated numerically.

BQ comes at the computational cost of $\mathcal{O}(N^3)$ in the number of quadrature nodes N (except for univariate problems with Markovian kernels that allow inference in linear time). This computational bottleneck limits the number of nodes and function evaluations to keep the inference computationally feasible. The nodes should therefore be placed in a space-filling manner. Optimal and other practical ways to select nodes for evaluating the integrand is addressed in Chapter 3.
On a different note, BQ suffers from the curse of dimensionality and is suitable for integrations of low to moderate dimensionality. Empirically, the more regular the integrand and the more prior knowledge can be accounted for, the larger the dimension in which integration problems remain feasible (but not necessarily competitive).

Active Design for Bayesian Quadrature

3

A frequent assumption in machine learning is that the data source is beyond our control and that training data are readily available. Yet there are scenarios in which this is not true and we can actively choose inputs to query for data. Data selection strategies might also be desired to select a subset if there is a massive amount of data [166]. Probabilistic numerical methods fall into the former category: data are the result of computations, and we can usually choose which computations to carry out. Decision theory provides the framework to automate the process of taking smart decisions under uncertainty. Probabilistic numerical methods (PNMS) can be turned into autonomous agents that interact with the data source (here: the CPU) to perform queries whose results are expected to yield promising outcomes for inferring the quantity of interest. In Bayesian quadrature this would correspond to an algorithm that decides on locations for evaluating the integrand without interference by the user.

[166]: MacKay (1992), 'Information-based objective functions for active data selection'

The automation of deciding about prospective actions to be taken by an algorithm in order to achieve a certain learning objective is called *active learning*. Much of active learning originates in the *experimental design* literature. A finite budget of potentially expensive[1] physical or computational experiments motivates the care taken in designing them in a meaningful manner (see *e.g.*, [46]).
In active learning, the reward of performing an action a from an action space $\mathcal{A}$ is measured through a *utility function* $u(a, \boldsymbol{\theta}, \mathcal{D}, \mathcal{D}_\star)$.[2] The utility is a function of previously observed data $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$ and prospective, yet unobserved data $\mathcal{D}_\star = \{\boldsymbol{X}_\star, \boldsymbol{y}_\star\}$ in a model that has unknown parameters $\boldsymbol{\theta} \in \Theta$. Oftentimes, the action space collapses onto the selection of new inputs, and we can absorb the action into the new design $\boldsymbol{X}_\star$.[3] The goal is to identify candidate measurements that are expected to yield a high utility for the given learning objective. The *expected utility* of choosing any design $\boldsymbol{X}_\star$ is given by the *acquisition function*[4]

$$\alpha_{\mathcal{D}}(\boldsymbol{X}_\star) \coloneqq \mathbb{E}_{\boldsymbol{y}_\star \mid \mathcal{D}} \mathbb{E}_{\boldsymbol{\theta} \mid \mathcal{D} \cup \mathcal{D}_\star} \left[u(\boldsymbol{\theta}, \mathcal{D}, \mathcal{D}_\star) \right].$$

Optimizing this acquisition function for the design $\boldsymbol{X}_\star$ induces an *acquisition policy* that pins down the action to take next. The selection of a utility function for a given task is important to finding a suitable design. Even for a given task, there exist a plethora of criteria to formulate policies that can lead to different designs.

In practice, an active learning procedure produces a sequence of actions by iteratively performing the optimal action and updating the model with the $N_\star$ new observations in the general multi-step look-ahead (*non-myopic*) approach. A *myopic* approximation is to instead optimize for a single new observation pair $\{\boldsymbol{x}_\star, y_\star\}$ at a time. Besides feasibility, the lack of exact model knowledge motivates a loop in which the model is repeatedly updated with new observations. The iterative procedure of consecutive automated updates is known as *sequential design*.

1: *Expensive* refers to resource-demanding, be it costly in the monetary sense, time- or energy-consuming.
[46]: Chaloner and Verdinelli (1995), 'Bayesian experimental design: A review'

2: The pessimistic perspective is to call the harm incurred by performing an experiment a *loss* or *regret*.

3: The notion of 'actions' is a central tenet in *reinforcement learning (RL)* [160, 240]. RL formalizes the interaction of an agent with its environment in order to maximize cumulative reward. This area is orthogonal to optimal experimental design in that the agent alters the state of the environment it is interacting with.

[160]: Levine (2018), 'Reinforcement learning and control as probabilistic inference: Tutorial and review'
[240]: Sutton and Barto (2018), *Reinforcement learning: An introduction*

4: Terminology depends on the community; *expected utility* being common in experimental design, and *acquisition function* in machine learning, *e.g.*, in Bayesian optimization.

Throughout Chapter 2, we have assumed nodes $\boldsymbol{X} = \boldsymbol{x}_{1:N}$ to be given for Bayesian quadrature (BQ). This chapter takes a decision-theoretic perspective on BQ. To this end, we review standard approaches to optimal design and then apply these concepts to BQ. Policies obtained through optimization of resulting acquisition functions usually depend on the definition of 'improvement', but in Gaussian models, many of the standard concepts collapse onto the same policy. Yet, Chapter 5 deals with a setting that lifts the degeneracy of the policy, and thus it is instructional to understand the differences between the active learning schemes. BQ has also seen a considerable amount of non-optimal strategies for the selection of quadrature nodes which we review at the end of the chapter.

3.1 A brief survey of information-theoretic decision criteria

Information theory provides a principled framework for decision-making under uncertainty [168]. A utility function should therefore represent the informativeness of actions towards the learning objective. The new data should be chosen where the *expected information gain* is maximized. The design of a utility depends on the task we are trying to solve and the way in which it is solved. Learning objectives in probabilistic models could be to find evaluations that *(i)* are informative about model parameters $\boldsymbol{\theta}$, *(ii)* improve the accuracy of predictions in a pre-specified domain $\tilde{\mathcal{X}} \subset \mathcal{X}$ or on correlated variables, or *(iii)* that are informative for discerning between competing models [166]. The goal in Bayesian quadrature to increase the accuracy of the integral estimate is an instance of *(ii)*.

[168]: MacKay (2003), *Information theory, inference and learning algorithms*

[166]: MacKay (1992), 'Information-based objective functions for active data selection'

3.1.1 Information and entropy

To define the informativeness of actions in a probabilistic model, we briefly review the very basics of information theory.[5] Consider a continuous random variable $\mathbf{x}$ with density p. Denote realizations of $\mathbf{x}$ as $\boldsymbol{x}$. Shannon defined information content as [230]

$$h(\boldsymbol{x}) = -\log_b(p(\boldsymbol{x}))$$

and it is measured in units of `bits` or `nats`, depending on whether the binary ($b = 2$) or the natural logarithm ($b = e$) is used. We stick to the natural logarithm in our setting. Information content measures the amount of 'surprise' when observing an event $\boldsymbol{x}$. The less probable it is to observe $\boldsymbol{x}$, the larger is its information content, *i.e.*, the more surprising it is to see $\boldsymbol{x}$.

The differential entropy of an ensemble refers to the expected information content of the random variable $\mathbf{x}$,

$$H[\mathbf{x}] = -\int_{\mathcal{X}} p(\boldsymbol{x}) \log p(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}.$$

In the discrete case, the entropy gets largest when all events are equiprobable. If the support of p is restricted to a box with finite bounds, the entropy is maximized by the uniform distribution. As the posterior contracts when updated with data, the entropy shrinks monotonically. The entropy of a Gaussian random variable $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu} \in \mathbb{R}^D$

5: We refer the reader to MacKay [168] for an excellent introduction to the topic.

[230]: Shannon (1948), 'A mathematical theory of communication'

and $\mathbf{\Sigma} \in \mathbb{R}^{D\times D}$ is

$$H[\mathbf{x}] = \frac{D}{2}\log(2\pi e) + \frac{1}{2}\log\det\mathbf{\Sigma}. \tag{3.1}$$

Consider now correlated random variables $\mathbf{x}$ and $\mathbf{y}$ with joint density $p(\boldsymbol{x}, \boldsymbol{y})$. The following identities are useful to quantify the information they carry about each other [168, §8]:

$$\begin{aligned} H[\mathbf{x},\mathbf{y}] &= -\int_{\mathcal{X}}\int_{\mathcal{Y}} p(\boldsymbol{x},\boldsymbol{y})\log p(\boldsymbol{x},\boldsymbol{y})\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{y} && \textit{joint entropy}\\ H[\mathbf{x}\,|\,\mathbf{y}] &= -\int_{\mathcal{X}}\int_{\mathcal{Y}} p(\boldsymbol{x},\boldsymbol{y})\log p(\boldsymbol{x}\,|\,\boldsymbol{y})\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{y} && \textit{conditional entropy}\\ H[\mathbf{x},\mathbf{y}] &= H[\mathbf{y}] + H[\mathbf{x}\,|\,\mathbf{y}] = H[\mathbf{x}] + H[\mathbf{y}\,|\,\mathbf{x}] \end{aligned}$$

where $H[\mathbf{x}]$ and $H[\mathbf{y}]$ are the *marginal entropies* of $\mathbf{x}$ and $\mathbf{y}$. The conditional entropy $H[\mathbf{x}\,|\,\mathbf{y}]$ quantifies the information needed to describe $\mathbf{x}$ once $\mathbf{y}$ is known and the converse for $H[\mathbf{y}\,|\,\mathbf{x}]$. If $H[\mathbf{x}\,|\,\mathbf{y}] = 0$, $\mathbf{x}$ is fully determined through $\mathbf{y}$. We can then define the *mutual information* as the amount of information that the observation of $\mathbf{y}$ conveys about $\mathbf{x}$ and vice versa,

$$I[\mathbf{x};\mathbf{y}] = H[\mathbf{x}] - H[\mathbf{x}\,|\,\mathbf{y}]. \tag{3.2}$$

The relationship between the measures of entropy in joint ensembles is illustrated in Figure 3.1.

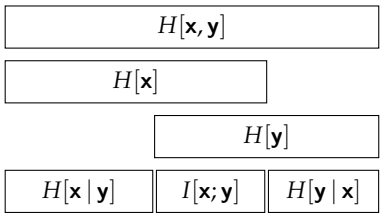


Figure 3.1: The connection between joint, marginal, and conditional entropy in correlated random variables, reproduced from [168, Figure 8.1].

3.1.2 Information measures

Consider a quantity of interest[6] $\boldsymbol{\xi} \in \Xi$, data $\mathcal{D}$ given and $\mathcal{D}_\star = \{\boldsymbol{X}_\star, \boldsymbol{y}_\star\}$ to be observed. The prospective data consists of $N_\star$ yet unobserved data pairs $\{\boldsymbol{x}_{n_\star}, y_{n_\star}\}_{n_\star=1}^{N_\star}$. There are two widely accepted ways to define measures of information gain [166, 46, 162, 149]:

Expected Shannon information gain The utility that quantifies the expected gain of Shannon information can be written as [46]

$$u^{\mathrm{SI}}(\mathcal{D}_\star;\mathcal{D}) = \mathbb{E}_{\boldsymbol{\xi}\,|\,\mathcal{D}_\star\cup\mathcal{D}}\left[\log\frac{p(\boldsymbol{\xi}\,|\,\mathcal{D}\cup\mathcal{D}_\star)}{p(\boldsymbol{\xi}\,|\,\mathcal{D})}\right].$$

The expectation w.r.t. the posterior $\boldsymbol{\xi}\,|\,\mathcal{D}\cup\mathcal{D}_\star$ is the negative *cross entropy* between two consecutive updates on $\boldsymbol{\xi}$ [166].

Mutual information The mutual information $I[\boldsymbol{\xi};\mathcal{D}_\star\,|\,\mathcal{D}]$ (3.2) can be written as the reduction of entropy achieved by adding new data $\mathcal{D}_\star$ to the present dataset $\mathcal{D}$,

$$u^{\mathrm{MI}}(\mathcal{D}_\star;\mathcal{D}) = I[\boldsymbol{\xi};\mathcal{D}_\star\,|\,\mathcal{D}] = H[\boldsymbol{\xi}\,|\,\mathcal{D}] - H[\boldsymbol{\xi}\,|\,\mathcal{D}\cup\mathcal{D}_\star].$$

Marginalization over prospective observations $\mathbf{y}_\star$ yields the acquisition function

$$\alpha_{\mathcal{D}}(\boldsymbol{X}_\star) = \mathbb{E}_{\mathbf{y}_\star\,|\,\boldsymbol{X}_\star,\mathcal{D}}\left[u(\boldsymbol{X}_\star,\mathbf{y}_\star;\mathcal{D})\right]. \tag{3.3}$$

6: These could be parameters $\boldsymbol{\theta}$, predictions at selected locations $\tilde{\boldsymbol{X}}$, or a projection such as integration.

[166]: MacKay (1992), 'Information-based objective functions for active data selection'

[46]: Chaloner and Verdinelli (1995), 'Bayesian experimental design: A review'

[162]: Lindley (1956), 'On a measure of the information provided by an experiment'

[149]: Krause et al. (2008), 'Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies'

that is to be maximized to find the most informative proposals,

$$\boldsymbol{X}_{\text{new}} = \underset{\boldsymbol{X}_\star \in \mathcal{X}^{N_\star}}{\arg\max}\, \alpha_{\mathcal{D}}(\boldsymbol{X}_\star). \tag{3.4}$$

Despite a discrepancy of the utility functions, the expected mutual information between the quantity of interest $\boldsymbol{\xi}$ and the new data $\mathcal{D}_\star$ is identical to the expected information gain (see [166] §2 for a proof). Since the initial entropy is an additive constant, it bears no relevance for the optimal value of $\boldsymbol{X}_\star$.

Linear models often admit a closed form representation of the acquisition function (3.3). This is the case for example when the goal is to infer the parameters in a linear regression model. Finding a design that maximizes expected Shannon information is then equivalent to optimizing the determinant of the Fisher information matrix. [7] In the frequentist experimental design literature this is known as D-optimality.[8] In particular, the optimal design is then independent of the previous observations $\boldsymbol{y}$ and can be computed offline for a given model. In nonlinear models, the expected utility is intractable; hence they require approximations. A PNM that falls into this category is Bayesian optimization, where the quantity to be inferred is the location of the global minimum of a function [113, 229].

Even when the functional form of (3.3) is known, finding its global maximizer is challenging. Solving the multivariate optimization problem (3.4) of size $DN_\star$ for new nodes $\boldsymbol{X}_\star \subset \mathcal{X}$ comes at high computational complexity [144, 149]. Oftentimes, a greedy approach is hence preferential to alleviate prohibitive computational cost. This leads to the above-mentioned myopic approximations in which one location is optimized for at a time. In linear regression, a myopic selection of the most informative point is equivalent to selecting the point of largest variance on $\boldsymbol{\xi}$.

3.2 Optimal design for BQ

Combined with an active design rule, BQ becomes an autonomous learning agent. The quantity of interest $\boldsymbol{\xi}$ is now the integral Z and implies that the action rule should select locations that are informative about the value of the integral over f. For now, we consider the vanilla BQ setting, in which $\boldsymbol{f}_X$ and Z are jointly Gaussian distributed. Before we set out to explicitly state utility functions that are commonly used in BQ,[9] we define the *canonical squared correlation* for vanilla Bayesian quadrature (VBQ) as

$$\rho^2_{\mathcal{D}}(\boldsymbol{X}_\star) = \mathfrak{v}_{\mathcal{D}}^{-1}\, \boldsymbol{\kappa}_{\star\,|\,\mathcal{D}}^\top\, \boldsymbol{C}_{\star\,|\,\mathcal{D}}^{-1}\, \boldsymbol{\kappa}_{\star\,|\,\mathcal{D}} \tag{3.5}$$

with the posterior variance of the integral $\mathbb{V}[\mathrm{Z}\,|\,\mathcal{D}] = \mathfrak{v}_{\mathcal{D}}$ and the posterior kernel mean evaluated at the new inputs

$$\boldsymbol{\kappa}_{\star\,|\,\mathcal{D}} = \kappa(\boldsymbol{X}_\star) - \kappa(\boldsymbol{X})(\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star),$$

as well as the noise-corrected posterior covariance matrix of the new data points

$$\boldsymbol{C}_{\star\,|\,\mathcal{D}} = k(\boldsymbol{X}_\star, \boldsymbol{X}_\star) - k(\boldsymbol{X}_\star, \boldsymbol{X}) \left(\boldsymbol{K} + \sigma^2 \boldsymbol{I}_N\right)^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) + \sigma^2 \boldsymbol{I}_{N_\star},$$

with the $N \times N$ identity matrix $\boldsymbol{I}_N$ and the $N_\star \times N_\star$ version $\boldsymbol{I}_{N_\star}$. $\rho^2(\boldsymbol{X}_\star) \in [0, 1]$ is a scalar quantity that measures the correlation be-

7: The Fisher information is defined as

$$\mathcal{I}_\xi = \mathbb{E}_\xi \left[\nabla_\xi \nabla_\xi^\top \log p(\boldsymbol{y}\,|\,\xi)\right].$$

8: Classic experimental design has developed an alphabetic naming scheme to term optimality according to different criteria. The letters refer to operations performed on the information matrix (in linear models). Bayesian interpretations of these criteria are reviewed in [46].

[113]: Hennig and Schuler (2012), 'Entropy search for information-efficient global optimization.'

[229]: Shahriari et al. (2016), 'Taking the human out of the loop: a review of Bayesian optimization'

[144]: Ko et al. (1995), 'An exact algorithm for maximum entropy sampling'

[149]: Krause et al. (2008), 'Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies'

9: Active design rules for BQ have seen a limited amount of discussion in the literature. The reason for this is that in Gaussian models such as vanilla BQ, different utilities yield the same active learning policy.

tween the integral Z and new observations $\boldsymbol{y}_\star$. In the one-step look-ahead case, this reduces to the correlation between the value of the integral and the new observation

$$\rho^2_{\mathcal{D}}(\boldsymbol{x}_\star) = \frac{|\kappa_{\star\,|\,\mathcal{D}}|^2}{(v_{\mathcal{D}}(\boldsymbol{x}_\star) + \sigma^2)\, \mathfrak{v}_{\mathcal{D}}}.$$

Here, $v_{\mathcal{D}}(\boldsymbol{x}_\star)$ denotes the Gaussian process (GP) posterior variance evaluated at $\boldsymbol{x}_\star$. In experimental design, this quantity can be found in [42].

[42]: Caselton and Zidek (1984), 'Optimal monitoring network designs'

3.2.1 Information-based design

Mutual information The mutual information between the Gaussian integral Z and new observations $\mathbf{y}_\star$ at locations $\boldsymbol{X}_\star$ in VBQ can be derived using the entropy of Gaussian random variables (3.1). In terms of the squared correlation (3.5) [42, 149][10]

$$\begin{aligned} \alpha^{\mathrm{MI}}_{\mathcal{D}}(\boldsymbol{X}_\star) &= I[\mathrm{Z}; \mathbf{y}_\star \,|\, \mathcal{D}] = H[Z \,|\, \mathcal{D}] - \mathbb{E}_{\mathbf{y}_\star\,|\,\mathcal{D}}[H[Z \,|\, \mathbf{y}_\star, \mathcal{D}]] \\ &= -\frac{1}{2} \log\left(1 - \rho^2_{\mathcal{D}}(\boldsymbol{X}_\star)\right) \geq 0. \end{aligned} \tag{3.6}$$

10: *cf.* Section B.2.2 for details

It measures the expected amount of information gained about Z by observing the yet unseen $\mathbf{y}_\star$, and vice versa. New locations that are uncorrelated with the target quantity, *i.e.*, $\rho = 0$, yield a minimum ($\alpha^{\mathrm{MI}}_{\mathcal{D}} = 0$) in the acquisition function and thus, such points are never selected. Conversely, as $\rho \to 1$, $\alpha^{\mathrm{MI}}_{\mathcal{D}} \to +\infty$ and perfectly correlated points will always be selected since they maximize the acquisition function.

3.2.2 Variance-based design

It is widely known that in Gaussian models, information criteria yield a policy that maximally reduces the variance on the quantity of interest. There are multiple equivalent ways to phrase the reduction of integral variance that results in different functional forms of the acquisition function. Their functional dependence on the squared correlation (3.5) is illustrated in Figure 3.2, alongside with the mutual information acquisition.

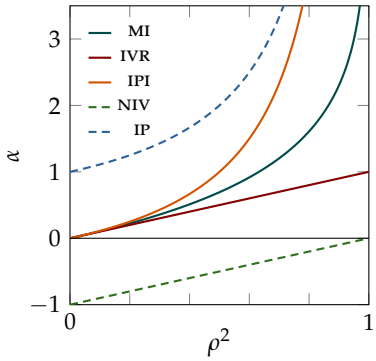


Figure 3.2: Common BQ acquisitions as a function of the canonical correlation ρ (3.5) (ignoring arbitrary scaling constants). They are all monotonic transformations of each other and give rise to a degenerate policy. But not all of them are suitable for trade-off with other criteria that might play a rôle in the selection (*cf.* Chapter 5). Solid acquisition functions satisfy the properties required for extension, dashed ones do not. The naming convention is

MI Mutual information (3.6)

IVR Integral variance reduction (3.7)

IPI Integral precision increase (3.9)

NIV Negative integral variance (3.8)

IP Integral precision (3.10)

See Section 5.4.2 for consequences.

Integral variance reduction The change of variance by evaluating at $\boldsymbol{X}_\star$ and observing $\boldsymbol{y}_\star$ relative to the variance at the current step is

$$\begin{aligned} \alpha^{\mathrm{IVR}}_{\mathcal{D}}(\boldsymbol{X}_\star) &= \frac{\mathbb{V}[\mathrm{Z} \,|\, \mathcal{D}] - \mathbb{V}[\mathrm{Z} \,|\, \mathcal{D} \cup \{\boldsymbol{X}_\star, \boldsymbol{y}_\star\}]}{\mathbb{V}[\mathrm{Z} \,|\, \mathcal{D}]} \\ &= \rho^2_{\mathcal{D}}(\boldsymbol{X}_\star), \end{aligned} \tag{3.7}$$

which is independent of upcoming function evaluations $\boldsymbol{y}_\star$. The derivation can be found in Section B.2.1. Interestingly, this utility takes a different form than $\alpha^{\mathrm{MI}}_{\mathcal{D}}$, but it is a monotonic transformation of (3.6), so their global maximizer is identical.

Posterior integral variance Since $\mathbb{V}[\mathrm{Z} \,|\, \mathcal{D}]$ is independent of the new design, it is equivalent to select $\boldsymbol{X}_\star$ such as to *minimize* the posterior variance of Z, or, to phrase it as a maximization problem, to consider

the negative integral variance (NIV) an acquisition function: The change of variance by observing $\boldsymbol{y}_\star$ at $\boldsymbol{X}_\star$ relative to the variance at the current step is

$$\alpha^{\text{NIV}}_{\mathcal{D}}(\boldsymbol{X}_\star) = -\mathbb{V}[\mathsf{Z} \mid \mathcal{D} \cup \{\boldsymbol{X}_\star, \boldsymbol{y}_\star\}] \propto \rho^2_{\mathcal{D}}(\boldsymbol{X}_\star) - 1 \leq 0. \tag{3.8}$$

Integral precision increase Instead of reducing the variance, it is equivalent to *increase* the precision[11] of the Gaussian distribution over the integral Z,

$$\begin{aligned}\alpha^{\text{IPI}}_{\mathcal{D}}(\boldsymbol{X}_\star) &= \frac{\mathbb{V}^{-1}[\mathsf{Z} \mid \mathcal{D} \cup \{\boldsymbol{X}_\star, \boldsymbol{y}_\star\}] - \mathbb{V}^{-1}[\mathsf{Z} \mid \mathcal{D}]}{\mathbb{V}^{-1}[\mathsf{Z} \mid \mathcal{D}]} \\ &= \frac{\rho^2_{\mathcal{D}}(\boldsymbol{X}_\star)}{1 - \rho^2_{\mathcal{D}}(\boldsymbol{X}_\star)},\end{aligned} \tag{3.9}$$

again taken in relative terms to the starting precision $\mathbb{V}^{-1}[\mathsf{Z} \mid \mathcal{D}]$.

11: The precision is reciprocal to the variance and the precision matrix is the inverse of the covariance matrix.

Posterior integral precision As for the variance, the posterior precision technically does not need to be put in relation to the current precision that is independent of the new inputs. The corresponding acquisition function is

$$\alpha^{\text{IP}}_{\mathcal{D}}(\boldsymbol{X}_\star) = \mathbb{V}^{-1}[Z \mid \mathcal{D} \cup \mathcal{D}_\star] \propto \frac{1}{1 - \rho^2_{\mathcal{D}}(\boldsymbol{X}_\star)} \tag{3.10}$$

Long story short, all of these acquisition functions are monotonic transformations of the canonical correlation (3.5) (*cf.* Figure 3.2 and Table B.1) and therefore, of the information criterion based on the mutual information (3.6). As a result, they all induce the same active learning policy.[12] A monotonic transformation does not change the *location* of a global maximum of a function—hence the *acquisition policy* remains unaltered. This is because the policy only depends on the *locations*, but not the *value* of the utility function's global maximum. As a consequence, any monotonic transformation of (3.5), whether interpretable or not, gives rise to the same policy. As for the variance-based acquisitions, the *relative* change was stated, hence they are unit-less. The mutual information technically carries the units of `bits` or `nats` and is thus a meaningful measure of improvement.

12: This is the reason why these options are not greatly discerned in the literature.

Now why is it worth stating them all? Once information gain is not the only criterion for selecting new inputs, but needs to be traded off against another criterion, the discrepancy between the acquisition functions become important. Chapter 5 considers active Bayesian quadrature when there is a location-dependent cost associated with queries. The new objective is information gain *per cost* and lifts the redundancy of the above acquisition functions. For now, the statement of degenerate function shall suffice—properties of an acquisition function to be robust to extension in concurrence with other selection criteria are discussed in Chapter 5.

Since the variance of a GP is independent of previous function evaluations $\boldsymbol{y}$, so is the variance of Z. As a consequence, only the *locations* of the future evaluations matter on these criteria and therefore, policies are fundamentally *non-adaptive*. Design rules for BQ can thus be precomputed, without interacting with the integrand [175].

[175]: Minka (2000), *Deriving quadrature rules from Gaussian processes*

It might sound unintuitive that the optimal active learning strategy should not depend on previous data. Adaptivity does not help when the hypothesis class is symmetric and convex, which is the case for the Gaussian (see [128] for a discussion in the context of BQ).

[128]: Kanagawa and Hennig (2019), 'Convergence guarantees for adaptive Bayesian quadrature methods'

The above policies acquire point sets that are *optimal* in the sense that they maximally reduce the integral variance and thus, the worst-case error given a fixed number of nodes. Therefore, the convergence rate of optimal BQ for a given kernel k and under the assumption that the integrand belongs to the associated RKHS $\mathcal{H}_k$ is at least as fast as the rates found for non-optimal 'space-filling' design, as discussed in Section 2.3.4.

3.2.3 Sequential design

Given a model, the joint optimization over $N_\star$ new nodes $\boldsymbol{X}_\star$ is in essence a combinatorial problem that quickly becomes infeasible as the number of nodes that are jointly optimized increases [144]. A more practicable procedure is to greedily select individual nodes from the given acquisition functions in a sequential manner [149]. Bayesian quadrature with sequential updates has been referred to as sequential Bayesian quadrature (SBQ) in the literature [36]. Early work on SBQ goes back to [53]. Pseudocode for SBQ is given in Algorithm 3.1. There are weak guarantees on the goodness of the myopic approximation as compared to the optimal strategy as the BQ objective functions are only *approximately* submodular. The greedy strategy on a submodular objective has been shown to deviate no more than a factor of $(1 - 1/e)$ from the optimal one. For BQ, the lower bound on the objective is reduced by a constant [149, 121]. [13] Convergence rates for SBQ are discussed in [206]. They show that the variance contracts at least as $\mathcal{O}(N^{-1})$, but argue that this bound is pessimistic for BQ.

[144]: Ko et al. (1995), 'An exact algorithm for maximum entropy sampling'

[149]: Krause et al. (2008), 'Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies'

[36]: Briol et al. (2015), 'Frank-Wolfe Bayesian quadrature: probabilistic integration with theoretical guarantees'

[53]: Cook (1993), 'Sequential Bayesian quadrature'

[121]: Huszár and Duvenaud (2012), 'Optimally-weighted herding is Bayesian quadrature'

13: Submodularity of a set function $s : 2^{\mathcal{X}} \to \mathbb{R}$ signifies that for all $A \subseteq B \subseteq \mathcal{X}$ and $\boldsymbol{x} \in \mathcal{X}$

$$s(A \cup \{\boldsymbol{x}\}) - s(A) \geq s(B \cup \{\boldsymbol{x}\}) - s(B).$$

Submodularity thus encodes the property of diminishing returns: the relative effect of adding an element is larger when the present set is smaller. The BQ objectives still retain the property of decreasing monotonically as data is added, yet they do not exactly satisfy the diminishing returns property and are thus only approximately submodular. Approximate submodularity only satisfies this inequality up to a constant.

Algorithm 3.1 Sequential Bayesian quadrature

procedure SEQUENTIALBQ($f(\cdot)$, $\mathsf{f}_{\boldsymbol{\theta}}$, $\nu(\cdot)$, $\alpha(\cdot)$, $N_{\mathtt{max}}$)
  $\mathcal{D} = \{\,\}$ // initialize data
  for $n = 1 : N_{\mathtt{max}}$ **do**
    $\boldsymbol{x} \leftarrow \arg\max_{\boldsymbol{x} \in \mathcal{X}} \alpha(\boldsymbol{x})$ // optimize acquisition function
    $y \leftarrow f(\boldsymbol{x}) + \epsilon$ // evaluate integrand
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{\boldsymbol{x}, y\}$ // add new data
    $\mathsf{f}_{\boldsymbol{\theta}} \leftarrow \mathsf{f}_{\boldsymbol{\theta}} \mid \mathcal{D}$ // condition GP on data
    $\boldsymbol{\theta} \leftarrow \arg\max_{\boldsymbol{\theta} \in \Theta} \log p(\boldsymbol{y} \mid \boldsymbol{\theta})$ // update GP hyperparameters
    $\alpha \leftarrow \alpha_{\mathcal{D}}$ // update acquisition function
    $p(Z \mid \boldsymbol{y}) \leftarrow$ BAYESQUAD($f, \mathsf{f}_{\boldsymbol{\theta}}, \nu, \mathcal{D}$) // call vanilla BQ (Algorithm 2.1)
  end for
  return $p(Z \mid \boldsymbol{y})$
end procedure

[206]: Pronzato (2021), 'Performance analysis of greedy algorithms for minimising a Maximum Mean Discrepancy'

The above exposition assumes the model is known and discards uncertainty about model hyperparameters. Instead of marginalizing them, tractability usually motivates optimizing the marginal likelihood of the GP w.r.t. hyperparameters. In that sense, the model adapts to the data, and *implicitly*, through the kernel hyperparameters, the Gaussian model, and hence the active learning policy, depends on the data. In practice, both the cost of batch selection and the lack of model knowledge motivates placing vanilla BQ in a loop and iteratively select new input locations, while regularly adapting the hyperparameters as new observations come

in. The empirical nature of this procedure makes it less amenable to a rigorous theoretical analysis.

Other sequential active selection schemes have been proposed for BQ, such as Frank-Wolfe quadrature [36], where quadrature nodes are selected by framing quadrature as a Frank-Wolfe optimization problem [19, 75]. This scheme is known to be suboptimal and is empirically outperformed by the sequential approximation to the optimal strategies discussed above. A different greedy point selecting scheme for quadrature has been proposed by J. Oettershagen [124], who suggests to sequentially find the points that maximize the posterior kernel mean.

[19]: Bach et al. (2012), 'On the equivalence between herding and conditional gradient algorithms'

[75]: Frank, Wolfe, et al. (1956), 'An algorithm for quadratic programming'

[124]: J. Oettershagen (2017), 'Construction of optimal cubature algorithms with applications to econometrics and uncertainty quantification'

While alleviating the issue of a high-dimensional optimization problem, a myopic strategy fundamentally entails serial computation that is not amenable to parallelization. Even when the model is learned on the fly, one might not want to update hyperparameters in every iteration. Batch selection methods that choose multiple promising nodes simultaneously, are a viable solution. In vanilla BQ, batch methods boil down to solving (3.4) for a feasible number of new inputs.

Other design rules have been proposed for BQ to overcome the intractability of the joint optimization of many nodes. Recently, Tanaka [241] took a different perspective on sequential updates to BQ: In their paper, they fix a population of nodes that are initialized at random. The nodes are, one by one, iteratively updated via gradient descent on an upper bound on the worst-case error that they derive.

[241]: Tanaka (2021), *Kernel quadrature by applying a point-wise gradient descent method to discrete energies*

3.3 Adaptive Bayesian quadrature

Adaptive BQ refers to active learning strategies that *explicitly* depend on previous function observations. Adaptive methods are suboptimal in vanilla BQ, but they may be helpful when additional constraints on the integrand motivate a model other than a GP. This is the case for *warped* BQ (*cf.* Section 2.5), in which the approximate posterior variance of the warped process depends on function evaluations $\boldsymbol{y}$.
The posterior variance $\mathbb{V}[\mathrm{Z} \mid \mathcal{D}]$ is in general intractable for a nonlinear transformation and hence, the information gain cannot be evaluated in closed form. A simple remedy is to disregard the objective of maximally reducing uncertainty about the integral and to instead evaluate the approximate GP at locations where its marginal variance is maximal. This sequential strategy is commonly known as *uncertainty sampling* and has been widely adopted for active learning in warped BQ [102, 44]. In warped BQ, the objective function for sequential design is the marginal posterior variance of f scaled with the integration measure ν

[102]: Gunter et al. (2014), 'Sampling for inference in probabilistic models with fast Bayesian quadrature'

[44]: Chai and Garnett (2019), 'Improving quadrature for constrained integrands'

$$\alpha^{\mathrm{US}}_{\mathcal{D}}(\boldsymbol{x}_\star) = k_{\mathrm{f} \mid \mathcal{D}}(\boldsymbol{x}_\star, \boldsymbol{x}_\star)\nu(\boldsymbol{x}_\star)^2. \tag{3.11}$$

The appearance of the squared integration measure can be understood from the natural scaling of the covariance with ν for each input. This corresponds to favoring points with large standard deviation in the support of the integration measure. With the approximate variance in (3.11) typically depending on the posterior mean of the auxiliary GP, uncertainty sampling explicitly depends on previous function evaluations. For example, the square and exponential transforms give rise to acquisition functions that take larger values at locations where the predictive mean

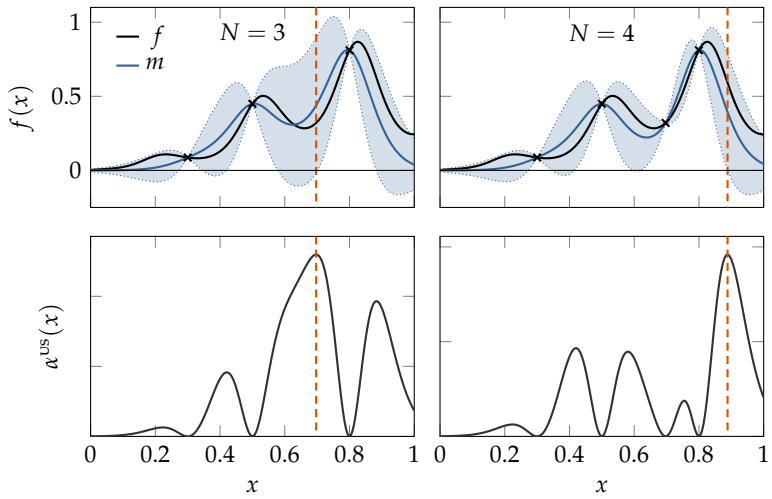


Figure 3.3: *Rows:* Two consecutive steps of WSABI-L (warped BQ with linearized moments). *Top row:* The warped GP and the integrand f conditioned on $N = 3$ and 4 points; *bottom row:* The acquisition function in the given iteration. Since the integration measure here is uniform over $[0, 1]$, it is simply the approximated posterior variance of the warped GP. The next proposal is the maximizer of α and indicated as - - -. The preference of locations with a large predictive mean leads to higher values of the acquisition function towards the right.

is large. Algorithmically, this can signify that the posterior variance of the integral does not need to shrink monotonically and might grow if a region with surprisingly high function values is discovered. Uncertainty sampling in warped BQ is illustrated in Figure 3.3. The behavior of evaluating the integrand where large function values are expected is intuitively desirable when positivity of the integrand is known. It is not the optimal strategy in warped BQ models—this is intractable—but since it guarantees minima in the acquisition functions at previously evaluated locations, it is guaranteed not to 'get stuck' and re-evaluate at existing locations.

Theoretical guarantees on the convergence of adaptive warped BQ methods that obtain their data through uncertainty sampling have been given by Kanagawa and Hennig [128]. They establish exponential asymptotic convergence rates $\mathcal{O}(e^{-CN^{1/D}})$ with a constant $C > 0$ for infinitely smooth kernels (*cf.* [128, §4.1]). Adaptivity can thus be advantageous when the hypothesis class is asymmetric, which is achieved through the GP warping.

[128]: Kanagawa and Hennig (2019), 'Convergence guarantees for adaptive Bayesian quadrature methods'

Uncertainty sampling—the sequential optimization of (3.11)—is an inherently serial procedure. Batch selection techniques that overcome the serial nature of uncertainty sampling have been transferred to warped BQ by Wagstaff et al. [252], which build on an extensive literature on batch Bayesian optimization (*e.g.*, [94, 97]).

[252]: Wagstaff et al. (2018), 'Batch selection for parallelisation of Bayesian quadrature'

[94]: Ginsbourger et al. (2010), 'Kriging is well-suited to parallelize optimization'

[97]: González et al. (2016), 'Batch Bayesian optimization via local penalization'

3.3.1 Empirical adaptive schemes

An empirical adaptive strategy has been proposed for BQ applied to variational inference. *Variational Bayesian Monte Carlo (VBMC)*[14] estimates the expectation of the log joint of the intractable model, $f = p(\mathcal{D} \mid \boldsymbol{x})p(\boldsymbol{x})$ w.r.t. the variational distribution $q = q_{\boldsymbol{\theta}}(\boldsymbol{x})$ by applying VBQ to the integral $\mathbb{E}_q[f]$ [1, 2]. Alteration of the variational parameters $\boldsymbol{\theta}$ motivates an explorative strategy that takes into account the predictive mean. The strategy is termed *prospective uncertainty sampling*, and has been proposed in the form

14: The naming scheme is inherited from [211], but VBMC does not employ Monte Carlo techniques.

[1]: Acerbi (2018), 'Variational Bayesian Monte Carlo'

[2]: Acerbi (2020), 'Variational Bayesian Monte Carlo with noisy likelihoods'

$$\alpha^{\text{PUS}}_{\mathcal{D}}(\boldsymbol{x}_\star) = k_{\mathsf{f} \mid \mathcal{D}}(\boldsymbol{x}_\star, \boldsymbol{x}_\star) q_{\boldsymbol{\theta}}(\boldsymbol{x}_\star) \exp(m_{\mathcal{D}}(\boldsymbol{x}_\star)). \tag{3.12}$$

The variational distribution q takes the rôle of the integration measure. The lack of the square as compared to (3.11) reduces the dampening of α in the tails of the variational distribution and thus encourages exploration as well. VBMC is a demonstration that empirical strategies can in practice yield superior performance over theoretically optimal strategies. Its convergence properties under the use of (3.12) has been shown by Kanagawa and Hennig [128] under boundedness conditions on the variational distribution q.

[128]: Kanagawa and Hennig (2019), 'Convergence guarantees for adaptive Bayesian quadrature methods'

Furthermore, Acerbi [1] proposed to regularize acquisition functions by exponentially dampening their value when the variance drops below a threshold value. Doing so impedes evaluations in the near vicinity of previously queried locations. Close input points can cause numerical instabilities in infinitely smooth models such as the widely used Gaussian kernel.

[1]: Acerbi (2018), 'Variational Bayesian Monte Carlo'

3.4 Utility-free design rules

The exposition on design in BQ would be incomplete without accounting for sampling rules that are not phrased in terms of a utility. On the one hand, these comprise model-free strategies that select new design points without taking the probabilistic model of Z into account. On the other hand, there are both random and deterministic rules for node placement that do account for the model, but without reasoning about informative locations.

Monte Carlo The simplest conceivable and model-free strategy to obtain new locations where to evaluate the function to be integrated is by random sampling[15] from the integration measure [211]. Rasmussen and Ghahramani dubbed the approach of conditioning the GP on random nodes *Bayesian Monte Carlo*. From the perspective of a quadrature rule (2.2), BQ with Monte Carlo states $x_n \sim \nu$ replaces the uniform weights $w_n^{\mathrm{MC}} = 1/N$ by the BQ weights $w_n^{\mathrm{BQ}} = [\boldsymbol{K}^{-1}\boldsymbol{\kappa}]_n$. This replacement compromises unbiasedness of the integral estimator.[16] It is to note that duplicate states can occur in Monte Carlo (MC) methods but have to be discarded in order to keep the Gram matrix invertible.

15: *cf.* Chapter 4 for an introduction to Monte Carlo methods

[211]: Rasmussen and Ghahramani (2003), 'Bayesian Monte Carlo'

16: Unbiasedness is a desirable requirement on estimators in statistics that states that in expectation, they take the value of the quantity estimated (*cf.* Chapter 4 for details). It is therefore a statement about the distribution of the estimator and meaningless if the corresponding algorithm is not either run many times, or until convergence. More important for convergence analysis is *consistency*, which ensures that an estimator converges to the true value as $N \to \infty$. BQ is consistent, but not unbiased.

When the integration measure is unnormalized, importance sampling or Markov chain Monte Carlo (MCMC) may be considered [37, 192]. BQ with nodes acquired via Monte Carlo sampling, MCMC, or importance sampling is consistent and converges at a rate that improves with the smoothness of the kernel k, again under the assumption that $f \in H_k$. At the same time, the rate deteriorates with increasing dimension [38, §3.3.2]. Sequential Monte Carlo [65] has also been used for obtaining quadrature nodes [39]. However, an intractable integration measure adds further complication to BQ, as it inhibits tractability of the kernel integrals. Therefore, these well-analyzed methods are short of practical use-cases.

[37]: Briol et al. (2019), 'Probabilistic integration: A role in statistical computation?'

[192]: Oates et al. (2019), 'Convergence rates for a class of estimators based on Steins method'

[38]: Briol (2018), 'Statistical computation with kernels'

[65]: Del Moral et al. (2006), 'Sequential Monte Carlo samplers'

[39]: Briol et al. (2017), 'On the sampling problem for kernel quadrature'

Quasi Monte Carlo Quasi-Monte Carlo (QMC) methods construct space filling points sets in a deterministic manner. As (MC)MC, QMC methods are used for numerical integration with uniform weights $1/N$ [68]. The grid-like point sets are then usually chosen in order to reduce the worst-case error in the RKHS that forms the hypothesis space of the

[68]: Dick et al. (2013), 'High-dimensional integration: the quasi-Monte Carlo way'

integrand. The design can thus be seen as aware of the prior, but not of the probabilistic model. QMC-based quadrature rules usually converge faster than MC methods due to the space-filling property of the design points. They are, however, restricted to integration problems on the unit cube $[0,1]^D$ and against a uniform integration measure.
BQ can be conditioned on the point set found by QMC. This corresponds to a quadrature rule that uses QMC point sets in concurrence with the BQ—instead of uniform—weights [37]. Under restrictive smoothness assumptions, the contraction of the BQ posterior attains a higher rate when using specific QMC point sets rather than randomly sampled points. In particular, the rates achieved with QMC are independent of the input dimension.[17] When constructing QMC point sets, the kernel needs to be stated upfront. Conversely, in MC one could choose a model after having evaluated the integrand.

17: see [37, Theorem 2]; and [38, §3.3.3]

Determinantal point processes Closer in spirit to uncertainty sampling is the use of draws from a *determinantal point process (DPP)* as quadrature nodes. A N-DPP is a point process associated with a kernel k, such that the joint probability of observing a point set $\boldsymbol{x}_{1:N}$ is proportional to the determinant of the associated kernel Gram matrix[18]

$$p(\boldsymbol{x}_{1:N}) \propto \det\left(k(\boldsymbol{x}_n, \boldsymbol{x}_{n'})\right)_{1\leq n,n'\leq N}.$$

18: The kernel k has to have a rank $\geq N$.

The determinant becomes small as points move closer to each other and zero if points overlap. As the nodes move away from each other, the volume of their covariance increases. DPPs hence encode repulsiveness and have been used in physics to describe non-interacting fermions [117].

[117]: Hough et al. (2006), 'Determinantal processes and independence'

DPPs are closely connected to GPs: Samples from a DPP can be drawn in an iterative manner by decomposing the joint probability density over the point set $\boldsymbol{X}$ into the product of their conditional probabilities [21]

[21]: Bardenet, Hardy, et al. (2020), 'Monte Carlo with determinantal point processes'

$$p(\boldsymbol{x}_{1:N}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n \mid \boldsymbol{x}_{1:n-1}) \propto \prod_{n=1}^{N} v_{1:n}(\boldsymbol{x}_n).$$

Proportionality constants take care of normalization and the exchangeability of the 'particles'. In other words, the order of the nodes is irrelevant. The quantity $v_{1:n}(\boldsymbol{x}_n)$ can be obtained via repeated application of the matrix inversion lemma (A.2) s.t.

$$v_{1:n}(\boldsymbol{x}) = \begin{cases} k(\boldsymbol{x},\boldsymbol{x}) & \text{if } n = 1, \\ k(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}_{1:n}^\top(\boldsymbol{x})\boldsymbol{K}_{1:n}^{-1}\boldsymbol{k}_{1:n}(\boldsymbol{x}) & \text{for } n > 1. \end{cases}$$

We use the notation $\boldsymbol{K}_{1:n} = k(\boldsymbol{x}_{1:n}, \boldsymbol{x}_{1:n})$ here to denote the kernel Gram matrix of all points up to n. $v_{1:n}(\boldsymbol{x})$ is the posterior predictive variance of a GP in the noise-free setting. DPPs are therefore a *model-aware* sampling scheme.

Under a few technical assumptions, samples from a DPP arise when viewing the posterior variance of a GP as an unnormalized probability density and iteratively drawing one sample thereof, with which the GP is updated. The rank-1 update ensures that the kernel Gram matrix does not need to be inverted from scratch in every iteration (*cf.* (A.1)). The similarity

to uncertainty sampling—as well as the misnomer—is apparent: Instead of *optimizing* the posterior variance iteratively to select new nodes, a DPP provides random samples from it. Combined with warped BQ, it constitutes a randomized adaptive strategy that preferentially selects points of large variance and is guaranteed to avoid resampling identical points [110]. Bayesian quadrature with point sets obtained from a DPP has been analyzed by Belhadji et al. [25].

[110]: Hennig and Garnett (2016), 'Exact sampling from determinantal point processes'

[25]: Belhadji et al. (2019), 'Kernel quadrature with DPPs'

3.5 Summary and BQ in practice

Bayesian quadrature not only enables encoding prior knowledge about the integrand into the quadrature model (*cf.* Chapter 2). The probabilistic approach also comes with a principled way of selecting new nodes for evaluating the integrand. In essence, the *optimal* BQ nodes are the ones that minimize the integral variance given the probabilistic model. Despite defining an *active learning* rule, the reduction of the integral variance is inherently independent of previous function evaluations. This renders such a rule *non-adaptive* and nodes pre-computable, although sequential schemes are usually preferred to overcome the prohibitive cost of jointly optimizing the nodes. Implicit dependence on data is achieved by repeatedly adjusting the model hyperparameters during alternating updates of the node set and the integral estimate. Adaptation of the hyperparameters also serves to calibrate the uncertainty of the integral estimate, which would otherwise merely depend on the choice of prior.

We therefore propose a pragmatic classification of adaptivity in BQ: *(i) implicitly* adaptive BQ, where the active learning scheme respects scales of the model that are iteratively improved; *(ii) explicitly* adaptive BQ, where the objective depends explicitly on function evaluations (which does not exclude an adaptation of the model over iterations), and *(iii)* fundamentally non-adaptive schemes in which the sampling or optimization objective is independent of the model.

Model updates during the procedure of node acquisition disrupt assumptions that underlie the theoretical guarantees given for various forms of BQ. These require the RKHS to be fixed, and the integrand to be a member thereof.[19] From a practitioner's perspective, it is rarely known if the integrand is a member of a particular RKHS. Integrability constraints on the kernel further usually limit the choices of a prior for BQ and make the model a convenience choice to some extent. Practical implementations of BQ algorithms resembling Algorithm 3.1 thus still rely on empirical performance evaluation.

19: except for Kanagawa et al. [130]

Monte Carlo Methods in a Nutshell

4

Monte Carlo (MC) methods are a tool for approximate inference and play a central rôle in the automatization of inference, *e.g.*, in probabilistic programming [41, 80]. Monte Carlo methods address two problems,

1. to generate samples from a target density p and
2. to use these samples to compute integrals w.r.t. this density.

This chapter is by no means a complete introduction to Monte Carlo methods, and the reader is referred to a number of excellent books and reviews on this topic. A comprehensive and intuitive introduction can be found in [168, §29 & 30]; the textbook by Kroese et al. [150] is an excellent reference for advanced MC methods.
In the context of Bayesian quadrature, MC methods may be used to generate design points, commonly referred to as *Bayesian Monte Carlo* (*cf.* [211] and Section 3.4). Above all, MC methods are a competitor to BQ for computing integrals and, despite suffering from slow convergence, they serve a large palette of integration problems and are relevant to the setting that is considered in Chapter 7.

[41]: Carpenter et al. (2017), 'Stan: A probabilistic programming language'
[80]: Ge et al. (2018), 'Turing: A language for flexible probabilistic inference'
[168]: MacKay (2003), *Information theory, inference and learning algorithms*
[150]: Kroese et al. (2013), *Handbook of Monte Carlo methods*
[211]: Rasmussen and Ghahramani (2003), 'Bayesian Monte Carlo'

4.1 Simple Monte Carlo

Monte Carlo methods use random draws $\boldsymbol{x}_s$ from a probability density p to estimate expected values of functions w.r.t. to p. The quadrature rule for such an integral has uniform weights $w = \frac{1}{S}$ for a sample size of S and approximates integrals of type (2.1) as

$$\begin{aligned} Z &= \int_{\mathcal{X}} f(\boldsymbol{x}) p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ &\simeq \hat{Z} = Q_{\mathrm{MC}}(f; \boldsymbol{X}) = \frac{1}{S} \sum_{s=1}^{S} f(\boldsymbol{x}_s) \quad \text{where} \quad \boldsymbol{x}_s \sim p. \end{aligned}$$

The expected value of $\hat{Z}$ is

$$\mathbb{E}_p[\hat{Z}] = \frac{1}{S} \sum_{s=1}^{S} \mathbb{E}_p\left[f(\boldsymbol{x}_s)\right] = Z.$$

The correctness in expectation is known as *unbiasedness* and widely celebrated in statistics. The variance of the Monte Carlo estimator is

$$\mathbb{V}_p[\hat{Z}] = \frac{\mathbb{V}_p[f]}{S}.$$

Hence, the error (*i.e.*, the standard deviation) of the estimator drops as $\mathcal{O}(S^{-1/2})$. Crucially, this rate is independent of the dimensionality of the integration problem, making Monte Carlo virtually immune against the curse of dimensionality. Nevertheless, high-dimensional spaces exhibit issues that also cause challenges for Monte Carlo methods [168]. Furthermore, since the variance of the integrand itself is usually unknown, it is typically difficult to estimate this error in practice.

Monte Carlo methods require independent samples $\{\boldsymbol{x}_s\}_{s=1}^S$ from the target density p. These are not easy to obtain in the general case. However, there are satisfactory pseudo-random number generators that enable drawing samples from a uniform distribution on $[0, 1]$ Kroese et al. [150, Chapter 1]. Uniform samples can be transformed to samples from other densities using the transformation rule for probability densities. When such a transformation is unavailable, other tricks have to be used to generate samples from the target density.

[150]: Kroese et al. (2013), *Handbook of Monte Carlo methods*

Rejection sampling Rejection sampling introduces a density q that we know how to sample from and scales it by a constant c such that $cq(\boldsymbol{x}) > p(\boldsymbol{x})\ \forall \boldsymbol{x} \in \mathcal{X}$. For each sample $\boldsymbol{x}_s \sim q$, draw a random uniform number $u \sim \text{Uniform}(0, 1)$ and accept the sample if $ucq(\boldsymbol{x}_s) < p(\boldsymbol{x}_s)$, otherwise reject the proposal. The larger the constant c (assuming both q and p are normalized), the more samples will be rejected. This renders rejection sampling unfeasible in most practical applications.

Importance sampling A more widely employed alternative is importance sampling: Again, introduce a density q that can be sampled from and rewrite the integral (2.1) as

$$\begin{aligned} Z &= \int_{\mathcal{X}} f(\boldsymbol{x}) \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} q(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ &\approx \frac{1}{S} \sum_{s=1}^{S} w_q(\boldsymbol{x}_s) f(\boldsymbol{x}_s) \quad \text{where} \quad \boldsymbol{x}_s \sim q \end{aligned}$$

with importance weights $w_q(\boldsymbol{x}_s) = \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}$. To avoid a large variance caused by astronomical importance weights, the proposal distribution q should be heavy-tailed.

In order to be effective, both rejection and importance sampling hinge on finding proposal densities that are similar to the target density. This gets increasingly difficult as dimensions grow and other methods are needed.

4.2 Markov chain Monte Carlo

Instead of trying to sample directly from the target density, Markov chain Monte Carlo (MCMC) methods propose a new state $\boldsymbol{x}'$ based on the current state $\boldsymbol{x}$ [214, 150]. To quantify the probability for transitioning into the new state $\boldsymbol{x}'$ a *Markov transition probability density* is defined as $\tau : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+ : (\boldsymbol{x}, \boldsymbol{x}') \mapsto \tau(\boldsymbol{x}' \mid \boldsymbol{x})$, also sometimes called the *transition kernel*. The transition density is chosen such that it can be sampled from, and the new state is determined as a random draw thereof, given the current state. Sequentially applying Markov transitions to an initial state $\boldsymbol{x}_0$

$$\begin{aligned} \boldsymbol{x}_1 &\sim \tau(\boldsymbol{x}_1 \mid \boldsymbol{x}_0) \\ \boldsymbol{x}_2 &\sim \tau(\boldsymbol{x}_2 \mid \boldsymbol{x}_1) \\ &\vdots \\ \boldsymbol{x}_S &\sim \tau(\boldsymbol{x}_S \mid \boldsymbol{x}_{S-1}) \end{aligned}$$

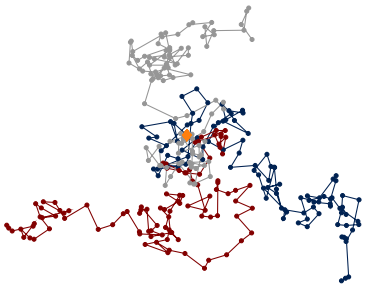


Figure 4.1: Three Markov chains starting out from ♦ with a Gaussian transition kernel centered around the current state.

[214]: Robert et al. (2004), *Monte Carlo statistical methods*
[150]: Kroese et al. (2013), *Handbook of Monte Carlo methods*

yields a *Markov chain* in which each state only depends on its predecessor and not on earlier states, thus fulfilling the *Markov property*. If the starting point is a sample from a density $p^{(0)}$, then the probability for the s^{th} element in the chain is $p^{(s)}(\boldsymbol{x}_s) = \int p^{(s-1)}(\boldsymbol{x}_{s-1})\tau(\boldsymbol{x}_s \,|\, \boldsymbol{x}_{s-1})\mathrm{d}\boldsymbol{x}_{s-1}$. Due to this dependence, a Markov chain does not create independent samples, but returns a *correlated* ensemble of draws, as illustrated in Figure 4.1.

4.2.1 Validity of MCMC methods

We wish to construct Markov transition processes in such a way that we create samples from the target density p. To achieve this, a MCMC method needs to fulfil two requirements:

Invariance The target distribution p needs to be an *invariant* or *stationary* distribution under the transition operator, *i.e.*,

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}')\tau(\boldsymbol{x}' \,|\, \boldsymbol{x})\mathrm{d}\boldsymbol{x}'.$$

In other words, p must be an eigenfunction of the transition kernel with eigenvalue 1.

Ergodicity The Markov chain is *ergodic* if for any $p^{(0)}$ and $\boldsymbol{x}_0 \sim p^{(0)}$, it converges to the target density

$$\lim_{s\to\infty} p^{(s)}(\boldsymbol{x}) = p(\boldsymbol{x}).$$

Ergodicity ensures that Markov chains do not get trapped in a corner of the state space $\mathcal{X}$.

The Markov chain is further termed *reversible* if the probability to transition from $\boldsymbol{x} \to \boldsymbol{x}'$ is the same as the reverse way

$$p(\boldsymbol{x})\tau(\boldsymbol{x}' \,|\, \boldsymbol{x}) = p(\boldsymbol{x}')\tau(\boldsymbol{x} \,|\, \boldsymbol{x}').$$

This property is known as *detailed balance* and entails invariance of p under τ.

4.2.2 The Metropolis-Hastings method

The most widely known MCMC method is the Metropolis-Hastings algorithm [173, 105], sketched in Figure 4.2. It proposes a new state $\boldsymbol{x}'$ from a proposal density $q(\boldsymbol{x}'; \boldsymbol{x})$ which is accepted with a probability of

$$u(\boldsymbol{x}'; \boldsymbol{x}) = \min\left(\frac{p(\boldsymbol{x}')q(\boldsymbol{x}; \boldsymbol{x}')}{p(\boldsymbol{x})q(\boldsymbol{x}'; \boldsymbol{x})}, 1\right).$$

This corresponds to a transition kernel $\tau(\boldsymbol{x}' \,|\, \boldsymbol{x}) = q(\boldsymbol{x}'; \boldsymbol{x})u(\boldsymbol{x}'; \boldsymbol{x}) + \delta(\boldsymbol{x}' - \boldsymbol{x}) \int q(\boldsymbol{x}'' \,|\, \boldsymbol{x})(1 - u(\boldsymbol{x}'' \,|\, \boldsymbol{x}))\mathrm{d}\boldsymbol{x}''$, which meets the detailed balance condition. The acceptance criterion ensures that p is invariant under this transition. The choice of proposal density q greatly impacts the performance of the sampler. The random walk that is characteristic for the Metropolis-Hastings method is a slow way to explore the state space especially in high dimensions.

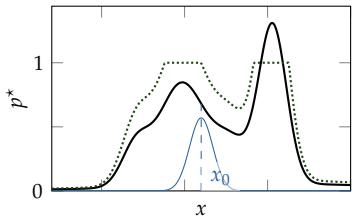


Figure 4.2: The Metropolis-Hastings algorithm: A proposal is drawn from $q(x, x_0)$ (—) and accepted with probability $u(x; x_0)$ (·····). The target density (**—**) need not be normalized, indicated by the $\star$ superscript. A symmetric proposal density reduces the acceptance criterion to the density ratio; this special setting is the one originally proposed by Metropolis et al.

[173]: Metropolis et al. (1953), 'Equation of state calculations by fast computing machines'

[105]: Hastings (1970), 'Monte Carlo sampling methods using Markov chains and their applications'

4.2.3 Gibbs sampling

Gibbs sampling grounds on the idea that sampling from a univariate distribution is easier than from a multivariate one. It has been initially devised by Geman and Geman [83], but popularized by Gelfand and Smith [81]. In Gibbs sampling, the target density is written as a product of conditional probabilities for each variable,

$$p(\boldsymbol{x}) = p(x_1)p(x_2 \,|\, x_1) \dots p(x_D \,|\, \boldsymbol{x}_{1:D-1}).$$

A new state $\boldsymbol{x}_{s+1}$ is found by iteratively sampling the x_d, $d = 1 \dots D$ conditioned on the already drawn variables that are part of the new state $[\boldsymbol{x}_{s+1}]_{1:d-1}$ and the remaining ones from the previous state $[\boldsymbol{x}_s]_{d+1:D}$. Figure 4.3 illustrates this procedure. Gibbs sampling is parameter-free and therefore easy to get running; however it can get very inefficient if variables are strongly correlated.

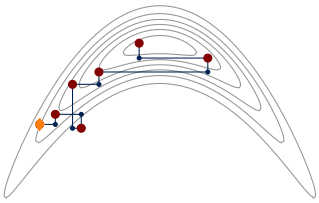


Figure 4.3: A few steps taken by a Gibbs sampler starting from ♦. A new sample is accepted (●) once all dimensions of the state vector have been updated by drawing from conditionals.

[83]: Geman and Geman (1984), 'Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images'

[81]: Gelfand and Smith (1990), 'Sampling-based approaches to calculating marginal densities'

4.2.4 Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC) was introduced as hybrid Monte Carlo [69] but has been renamed to do justice to its origin in Hamiltonian dynamics from physics. HMC uses gradients of the target density to overcome random walk behavior and ensure an improved exploration behavior [186, 28]. The key concept is to simulate the dynamics of a fictitious particle of mass m in a potential well $U(\boldsymbol{x}) = -\log p(\boldsymbol{x})$. To this end, the state space is augmented with a momentum variable $\boldsymbol{p}$. The total energy of the particle is the sum of potential and kinetic energy and given through the Hamiltonian

$$H(\boldsymbol{x}, \boldsymbol{p}) = U(\boldsymbol{x}) + \frac{\boldsymbol{p}^\top \boldsymbol{p}}{2m}.$$

It is a conserved quanity. Hamiltonian dynamics are the solution to the Hamiltonian equations of motion

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \nabla_{\boldsymbol{p}} H = \frac{\boldsymbol{p}}{m} \quad \text{and} \quad \frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\nabla_{\boldsymbol{x}} H = -\nabla_{\boldsymbol{x}} U. \tag{4.1}$$

The joint probability density of a state $(\boldsymbol{x}, \boldsymbol{p})$ is $\propto e^{-H(\boldsymbol{x},\boldsymbol{p})}$. The momenta are normally distributed and independent of $\boldsymbol{x}$. New states are proposed by simulating trajectories of the particle according to (4.1) with a randomly drawn momentum $\boldsymbol{p} \sim \mathcal{N}(0, m\boldsymbol{I})$. This is done numerically by a leapfrog integrator[1] that alternates updates on $\boldsymbol{p}$ and $\boldsymbol{x}$. The algorithm requires two parameters for the solver, the step-size ϵ and the number of steps T. Its performance is quite sensitive to the choice of these values. An exemplary trajectory is shown in Figure 4.4. The no-U-turn sampler (NUTS) [116] is a modification of HMC that overcomes the need for manual tuning. NUTS auto-tunes on the given problem by simulating trajectories both forward and reverse in time and terminates the chains once the momenta projected onto the connecting line are anti-aligned. This is expected to happen when further simulation would reduce the distance between the states and likely bring them into already explored realms. NUTS is nowadays the default MCMC method in a number of probabilistic programming languages such as `Stan` and `Turing.jl` [41, 80].

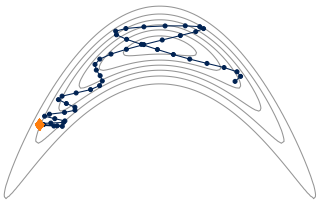


Figure 4.4: One trajectory simulated with Hamiltonian Monte Carlo starting from ♦.

1: The leapfrog method is a symplectic integrator designed for systems with a conserved quantity, in this case energy.

[69]: Duane et al. (1987), 'Hybrid Monte Carlo'

[186]: Neal (2012), 'MCMC using Hamiltonian dynamics'

[28]: Betancourt (2017), 'A conceptual introduction to Hamiltonian Monte Carlo'

[116]: Hoffman and Gelman (2011), 'The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo'

[41]: Carpenter et al. (2017), 'Stan: A probabilistic programming language'

[80]: Ge et al. (2018), 'Turing: A language for flexible probabilistic inference'

4.2.5 Elliptical slice sampling

The previous MCMC algorithms are in principle suitable for sampling from any density p. Elliptical slice sampling (ESS) by Murray et al. [182] is a specialized MCMC method to draw samples from a posterior distribution with likelihood $\ell(\boldsymbol{x})$ when the prior is multivariate normal $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Conceptually, ESS reduces the space for transitions to a one-dimensional ellipse at each iteration of the Markov chain. Given an initial location $\boldsymbol{x}_s \in \mathbb{R}^D$, ESS draws an auxiliary vector from the Gaussian prior $\boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to construct an ellipse

$$\boldsymbol{x}(\vartheta) = \boldsymbol{x}_s \cos\vartheta + \boldsymbol{\xi} \sin\vartheta. \tag{4.2}$$

parameterized by an angle $\vartheta \in [0, 2\pi]$.[2] The new proposal $\boldsymbol{x}_{s+1}(\vartheta)$ is found by applying slice sampling [185] to the angular domain: Given the current state $\boldsymbol{x}_s$ with likelihood value $\ell(\boldsymbol{x}_s)$, a likelihood threshold is drawn uniformly, $u \sim \text{Uniform}(0, \ell(\boldsymbol{x}_s))$. This threshold determines the stretches—or *slice*—on the ellipse that produce admissible proposals (top panel of Figure 4.5). Technically, the next location $\boldsymbol{x}_{s+1}$ is a uniform sample from these admissible sections. However, for general likelihoods it is infeasible to directly produce a draw from the slice. The workaround taken by ESS is to iteratively adjust a bracket $[\vartheta_{\text{min}}, \vartheta_{\text{max}}]$. Initially, the bracket is $[0, 2\pi]$, from which a proposal is drawn uniformly, which is rejected if not on the slice. The limits of the bracket are shrunk to rejected angles such that $\boldsymbol{x}_s$ is always contained (center panel of Figure 4.5), until an angle satisfying $\ell(\boldsymbol{x}(\vartheta)) \geq u$ is found and accepted (bottom panel of Figure 4.5). The new state $\boldsymbol{x}_{s+1}$ is constructed with this angle θ from (4.2). The angle-finding procedure is detailed in Algorithm 4.1 and illustrated in Figure 4.5.

The key asset of the ESS algorithm is its simplicity and that it does not require parameter tuning. Since proposals are made in a univariate domain, they are cheap to generate. Yet, for complicated likelihood functions, the acceptance rate may become very low and other MCMC schemes might be more appropriate.

4.2.6 Practical notes on MCMC

Bayesian inference with MCMC MCMC methods usually do not require the target density to be normalized. For this reason, they are a popular tool for posterior inference since they can be used to generate samples from the posterior without needing to compute the evidence. The posterior samples are used to compute expectations as (2.1), where ν takes the rôle of the unnormalized posterior density, and for predictions, which are expectations of the likelihood under the posterior.

The marginal likelihood While posterior inference with MCMC is possible without knowing the evidence, it is generally impossible to estimate the normalization constant from posterior samples. The marginal likelihood is a desired quantity in Bayesian model comparison. There exist hence MCMC methods targeted purely at estimating the evidence, *e.g.*, annealed importance sampling, which used a sequential *tempering* scheme to estimate ratios of intermediary normalization constants of which the

[182]: Murray et al. (2010), 'Elliptical slice sampling'

2: Note that no matter the dimension of $\boldsymbol{x}$ the ellipse remains a one-dimensional object to sample from, and the new state is fully defined through the angle ϑ_*. In high dimensions, the eccentricity of the ellipse is likely to be small and ellipses tend to be almost circular.

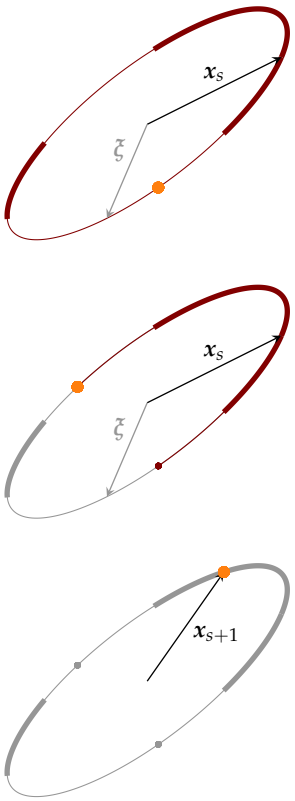


Figure 4.5: Sketch of ESS. The ellipse is fully defined by the current state $\boldsymbol{x}_s$ and the auxiliary vector $\boldsymbol{\xi}$ drawn from the Gaussian prior. The *slice* on the ellipse where the likelihood $\ell(\boldsymbol{x}(\vartheta))$ exceeds the threshold level u is indicated by thick lines (▬). The initial proposal (●), which lies off the slice here, determines the initial bounds of the bracket $[\vartheta_{\text{min}}, \vartheta_{\text{max}}]$ *(top)*. The second draw *(center)* also comes from the entire ellipse, and, because it is not on the slice, the two draws define the new bracket s.t. $\boldsymbol{x}_s$ lies within the bracket (▬). The prohibited part of the ellipse is grayed out. This procedure is repeated until a sample falls onto the slice and is accepted as new state $\boldsymbol{x}_{s+1}$ *(bottom)*.

Algorithm 4.1 One iteration of elliptical slice sampling

procedure EllipticalSliceSampling(Ellipse base vectors $\boldsymbol{x}_s$ and $\boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, likelihood function $\ell(\cdot)$)
  $u \sim \text{Uniform}(0, \ell(\boldsymbol{x}_s))$ // draw likelihood threshold
  $\vartheta \sim \text{Uniform}(0, 2\pi)$ // draw initial angle
  $[\vartheta_{\min}, \vartheta_{\max}] \leftarrow [\vartheta - 2\pi, \vartheta]$ // set bounds for angle bracket
  while $\ell(\boldsymbol{x}_s \cos\vartheta + \boldsymbol{\xi} \sin\vartheta) < u$ **do** // accept sample that exceeds the likelihood threshold
    $\vartheta \sim \text{Uniform}(\vartheta_{\min}, \vartheta_{\max})$ // draw uniform sample from angle bracket
    if $\vartheta < 0$ **then** $\vartheta_{\min} \leftarrow \vartheta$ **else** $\vartheta_{\max} \leftarrow \vartheta$ // shrink bracket
  end while
  return ϑ // return sample
end procedure

evidence can be derived [184]. A few methods have been devised to estimate the evidence concurrently with producing posterior samples, such as *nested sampling* [232].

[184]: Neal (2001), 'Annealed importance sampling'

[232]: Skilling (2004), 'Nested sampling'

Diagnostics In order to achieve reliable estimates from samples that have been obtained with MCMC, the Markov chain should have converged to the underlying target distribution. If not simulated for long enough, the samples produced by MCMC may be unrepresentative of the true target. This is especially true for multimodal models, in which it can take a long time for a chain to discover spatially separated modes. Assessing convergence of MCMC methods is a difficult problem in its own right and there is no way to prove whether the sampler has reached equilibrium [82]. Another issue is that samples in the chain are auto-correlated and thus violate the i.i.d. assumption of the Monte Carlo estimator. This is not an issue at convergence, but very much within finite runtime. Correlations between samples deteriorate estimates computed from the samples.

[82]: Gelman et al. (1995), *Bayesian data analysis*

When employing MCMC, there are a few practical steps that can be taken to inspect and monitor the performance of the sampler. As a measure to counteract bias from initialization that might confine chains in a low-probability corner of the state space, the first part of the Markov chain is typically discarded. This procedure is called *burn-in*.
The dependence of samples within the chain is addressed by estimating an *effective sample size* from auto-correlations of samples or by *thinning* the chain by only recording every say 10^{th} sample in the chain. It is further common practice to simulate multiple sequences that have been initialized at a diverse set of points. Doing so comes with the advantage that sampling can be parallelized. Elaborate algorithms have been devised for diverse initialization, *e.g.*, [267]. Multiple chains further allow tracking convergence by inter-chain comparison. The $\hat{R}$ statistics is another widely used diagnostic tool that uses variances across multiple chains as well as intra-chain variances to assess convergence of the ensemble of Markov chains over simulation time [249]. It is thus to be said that while being an essential component of automated inference, MCMC methods do require expert knowledge and critical assessment of predictions made.

[267]: Zhang et al. (2021), 'Pathfinder: Parallel quasi-Newton variational inference'

[249]: Vehtari et al. (2021), 'Rank-normalization, folding, and localization: An improved $\hat{R}$ for assessing convergence of MCMC (with discussion)'

Bayesian Quadrature Case Studies

Active Multi-Information Source Bayesian Quadrature

5

Due to its data efficiency, Bayesian quadrature (BQ) is suitable to solve integrals of expensive-to-evaluate black-box functions. Previous *active* BQ learning schemes have focused merely on the integrand itself as information source (*cf.* Chapter 3), and do not consider information transfer from cheaper, related functions. Approximations to the costly integrand are, however, frequently available in practice, for example when evaluating the integrand requires a complex simulation to be run that can be approximated by simulating at lower levels of sophistication and at lesser expense.
This chapter introduces a version of BQ that enables transferring information from related information sources to the main integration problem. In particular, it develops an active learning scheme that decides which source the information should be retrieved from, based on potentially variable cost of the sources. To this end, we construct meaningful cost-sensitive multi-source acquisition *rates* as an extension to common utility functions from vanilla Bayesian quadrature (VBQ), and discuss pitfalls that arise from blindly generalizing. In proof-of-concept experiments we scrutinize the behavior of our generalized acquisition functions and demonstrate that active multi-information source Bayesian quadrature (AMSBQ) allocates budget more efficiently than VBQ for learning the integral to a good accuracy. The content of this chapter has been published as

> A. Gessner, J. Gonzalez, and M. Mahsereci. 'Active multi-information source Bayesian quadrature'. In: *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*. Ed. by A. Globerson and R. Silva. Vol. 115. Proceedings of Machine Learning Research. AUAI Press, 2019.

5.1 Setting

The goal is to estimate the integral over the information source of interest (the *primary* source), w.l.o.g. indexed by 1, $f_1 : \mathcal{X} \mapsto \mathbb{R}$, $\boldsymbol{x} \mapsto f_1(\boldsymbol{x})$ and integrated against the probability measure ν on $\mathcal{X} \subseteq \mathbb{R}^D$,

$$Z_1 =: \int_{\mathcal{X}} f_1(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}) \tag{5.1}$$

in presence of $L-1$ not necessarily ordered or orderable *secondary* information sources $f_2, \ldots, f_L$, with $f_l : \mathcal{X} \mapsto \mathbb{R}$. Each source $l \in \mathcal{L} = \{1, \ldots, L\}$ comes with an input-dependent cost $c_l(\boldsymbol{x})$ which must be invested to query f_l at location $\boldsymbol{x}$. For ease of interpretation and numerical stability we set $c : \mathcal{L} \times \mathcal{X} \mapsto [\delta, 1]$ and $0 < \delta \leq 1$. This is equivalent to assuming there exists a $c_{\min} > 0$ and a $c_{\max} < \infty$ s.t. $c_{\min} \leq c_l(\boldsymbol{x}) \leq c_{\max}$ and then normalizing w.r.t. $c_{\max}$, *i.e.*, $\delta = \frac{c_{\min}}{c_{\max}}$. In other words, no query takes an infinite amount of resources, nor does any evaluation come for free. Normalization is not required and in practice, neither $c_{\max}$ nor $c_{\min}$ need to be known.

5.2 Motivation

Integrals of expensive-to-evaluate functions arise in many scientific and industrial applications, for example when expected values need to be computed and each evaluation of the integrand requires the run of a complex computer simulation where an input is only known by its distribution e.g., in meteorology, astrophysics, fluid dynamics, biology, operations research, *et cetera*. The complex simulation could be a Monte Carlo simulation, a finite-element or finite-volume simulation, or a stochastic model. Complex models can often be studied at various levels of sophistication. We denote models that are meant to solve the same task *information sources*, and term the *primary source* the highest-quality source. A primary source could be an elaborate Earth system model to simulate anthropogenic climate change. Secondary sources might parameterize important effects like albedo or neglect detailed land surface processes or ocean biogeochemistry [73]. These approximations come at a reduced cost at the expense of quality. They could be numerical models that are run at a lower resolution (*e.g.*, a coarser grid in a fluid dynamics application), model simplifications by neglecting details or by using an approximate model that is easier to solve numerically, but also analytic approximations. The task of finding approximations to computationally demanding numerical models is an area of active research all on its own (see *e.g.*, [27]).

[73]: Flato et al. (2013), 'Evaluation of climate models'

[27]: Benner et al. (2017), *Model reduction and approximation: theory and algorithms*

Depending on the computational budget and requirements for accuracy, it may be sufficient to employ only the secondary sources for certain tasks. This is not the case for integration: errors introduced by downscaling the original problem add up and introduce an arbitrary bias. Nevertheless, it would be desirable to include information from these cheaper approximations to shrink the variance of the integral over the primary source.

This is what multi-source models do. Multi-source modeling is a statistical technique for harvesting information from related functions by constructing correlated surrogates over multiple sources. When the information sources are hierarchical in that they are ordered from most to least informative, this concept is known as multi-fidelity modeling [137, 199, 74, 156]. The notion of *multi-source* models is more overarching and includes settings in which sources do not exhibit an easily identifiable order, if any. Each of the sources has its own *cost* function that quantifies the cost of evaluating the source at a certain input, similar to the setup considered by Krause et al. [148]. An input-dependent cost might arise when the simulation run to query the integrand needs to be refined for certain values of the input to ensure numerical stability. A linear instance of a multi-source model is a multi-output Gaussian process (GP) aka. co-kriging [6]. BQ with multi-output GPs to integrate several related functions has been studied by Xi et al. [265], who impose properties on data—which they assume given—to prove theoretical guarantees and consistency of the Bayes estimator.

[137]: Kennedy and O'Hagan (2000), 'Predicting the output from a complex computer code when fast approximations are available'

[199]: Peherstorfer et al. (2018), 'Survey of multifidelity methods in uncertainty propagation, inference, and optimization'

[74]: Forrester et al. (2007), 'Multi-fidelity optimization via surrogate modelling'

[156]: Le Gratiet and Garnier (2014), 'Recursive co-kriging model for design of computer experiments with multiple levels of fidelity'

[148]: Krause et al. (2006), 'Near-optimal sensor placements: Maximizing information while minimizing communication cost'

[6]: Alvarez et al. (2012), 'Kernels for vector-valued functions: A review'

[265]: Xi et al. (2018), 'Bayesian quadrature for multiple related integrals'

Yet when practically faced with integrating a function with the help of secondary sources, *active learning* comes in handy. We wish the algorithm to take care of the choice where and from which source to gather information to improve most on the learning target while keeping costs manageable.

This chapter lays the foundations for *active multi-source Bayesian quadrature* (AMSBQ) for the task of integrating an expensive function that comes with cheaper approximations. We generalize the BQ acquisition functions discussed in Chapter 3 to acquisition *rates* that trade off improvement on the integral against cost. We find that some rates induce sane, others pathological acquisition policies. Pathologies were not present in the common VBQ acquisition schemes that all give rise to the same degenerate policy, regardless of the acquisition's *value*. Cost-adapted rate policies do depend on these values and are thus intricately tied to the meaning of the acquisition function that encodes progress on the quadrature task. Simply put, *all* considered (even pathological) multi-source acquisition policies collapse onto a single policy for VBQ, as a corner case of AMSBQ.

5.3 A linear multi-source model for Bayesian quadrature

For now, we disregard the cost and first establish the required ingredients for jointly modeling and integrating multiple related functions.

Linear multi-source models can be phrased as multi-output Gaussian processes [6] over a vector-valued function $\boldsymbol{f} = [f_1, \ldots, f_L]$, $\boldsymbol{f} : \mathcal{X} \mapsto \mathbb{R}^L$. Non-linear models for multi-source modeling exist as well [201]. They do however come with the additional technical difficulty that the model may not be integrable analytically—a sensible prerequisite for BQ—and are thus another beast altogether.

[6]: Alvarez et al. (2012), 'Kernels for vector-valued functions: A review'

[201]: Perdikaris et al. (2017), 'Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling'

5.3.1 Multi-source models via multi-output Gaussian processes

The notation of multi-output GPs mimics the single-output case, that is,

$$\mathbf{f} \sim \mathcal{GP}(\boldsymbol{m}, \boldsymbol{K})$$

where $\boldsymbol{K}$ is an $L \times L$ matrix-valued covariance function. More precisely, the covariance between two sources f_l and $\mathrm{f}_{l'}$ at inputs $\boldsymbol{x}$ and $\boldsymbol{x}'$ is $\mathbb{C}[\mathrm{f}_l(\boldsymbol{x}), \mathrm{f}_{l'}(\boldsymbol{x}')] = k_{ll'}(\boldsymbol{x}, \boldsymbol{x}')$. The kernel $k_{ll'}(\boldsymbol{x}, \boldsymbol{x}')$ encodes not only characteristics of the individual sources (*e.g.*, smoothness), but also the correlation between them.

Element-wise observations In the multi-source setting, observations come in source-location-evaluation triplets $(l, \boldsymbol{x}, y_l)$ with $y_l = f_l(\boldsymbol{x}) + \epsilon_l$ and source-dependent observation noise $\epsilon_l \sim \mathcal{N}(0, \sigma_l^2)$ as usually only one element of $\boldsymbol{f}$ is being observed at a time. This corresponds to observing elements of full vectorial observations $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{\epsilon}$, where $\boldsymbol{y} \in \mathbb{R}^L$ and can be written as projections thereof

$$y_l = \boldsymbol{h}_l^\top \boldsymbol{y}$$

where $\boldsymbol{h}_l$ denotes a vector with a 1 in the l^{th} coordinate and zero elsewhere. Let $\boldsymbol{Y} \in \mathbb{R}^{NL}$ denote the vector of N stacked vector-valued noisy observations $[\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N]$. The elements (or sources) that have been observed at any of the N observations is written as $\boldsymbol{\ell} = [l_1 \ \ldots \ l_N]^\top$. The observations

at only these sources are projections of $\boldsymbol{Y}$,

$$\boldsymbol{y}_\ell = \begin{bmatrix} \boldsymbol{h}_{l_1}^\top & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{h}_{l_N}^\top \end{bmatrix} \boldsymbol{Y} =: \boldsymbol{H}^\top \boldsymbol{Y},$$

where $\boldsymbol{H}$ is a sparse $NL \times N$ matrix. Note the delicate notational difference between the N observations of single elements of $\boldsymbol{f}$, $\boldsymbol{y}_\ell \in \mathbb{R}^N$, and a single evaluation of the vector-valued function $\boldsymbol{y} \in \mathbb{R}^L$.

The covariance matrix between all the observations is

$$\mathbb{C}[\boldsymbol{y}_\ell, \boldsymbol{y}_\ell] = \boldsymbol{H}^\top \left(\underbrace{\begin{bmatrix} \boldsymbol{K}(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & \boldsymbol{K}(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ \boldsymbol{K}(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & \boldsymbol{K}(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix}}_{\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X})} + \boldsymbol{\Sigma} \otimes \boldsymbol{I}_N \right) \boldsymbol{H}$$

where $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1^2, \ldots, \sigma_L^2) \in \mathbb{R}^{L \times L}$, $\boldsymbol{I}_N$ is an $N \times N$ identity matrix, and $\otimes$ is the Kronecker product.[1] Also, $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) \in \mathbb{R}^{NL \times NL}$. We introduce the following shorthand notation for element-wise observations

$$\begin{aligned} \boldsymbol{K}_{\ell\ell}(\boldsymbol{X}, \boldsymbol{X}) &= \boldsymbol{H}^\top \boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}} \boldsymbol{H}, \\ \boldsymbol{\Sigma}_\ell &= \boldsymbol{H}^\top (\boldsymbol{\Sigma} \otimes \boldsymbol{I}_N) \boldsymbol{H}. \\ \boldsymbol{y}_\ell &= \boldsymbol{H} \boldsymbol{Y}. \end{aligned}$$

Evaluations of individual sources can be incorporated easily in the multi-source model without needing to relate them to the vector-valued observations $\boldsymbol{Y}$.

1: Let $\boldsymbol{A}$ be a $N \times M$ and $\boldsymbol{B}$ a $P \times Q$ matrix. Then $\boldsymbol{A} \otimes \boldsymbol{B}$ is a $NP \times MQ$ matrix in which the n, m^{th} block is $a_{nm}\boldsymbol{B}$, with $n = 1, \ldots, N$ and $m = 1, \ldots, M$. See [248] for details.
[248]: Van Loan (2000), 'The ubiquitous Kronecker product'

GP posterior The dataset $\mathcal{D} = \{\boldsymbol{\ell}, \boldsymbol{X}, \boldsymbol{y}_\ell\}$ contains the N data triplets from evaluating elements $\boldsymbol{\ell}$ of $\boldsymbol{f}$ at locations $\boldsymbol{X}$,

$$\begin{aligned} \boldsymbol{X} &= [\boldsymbol{x}_1 \ \ldots \ \boldsymbol{x}_N]^\top, \\ \boldsymbol{\ell} &= [l_1 \ \ldots \ l_N]^\top, \\ \boldsymbol{y}_\ell &= [f_{l_1}(\boldsymbol{x}_1) + \epsilon_{l_1} \ \ldots \ f_{l_N}(\boldsymbol{x}_N) + \epsilon_{l_N}]^\top. \end{aligned}$$

The l^{th} component of the posterior mean and covariance are

$$\begin{aligned} m_{l|\mathcal{D}}(\boldsymbol{x}) &= m_l(\boldsymbol{x}) + \boldsymbol{k}_{l\ell}(\boldsymbol{x}, \boldsymbol{X}) \boldsymbol{G}_\ell(\boldsymbol{X})^{-1} (\boldsymbol{y}_\ell - \boldsymbol{m}_\ell(\boldsymbol{X})), \\ k_{ll'|\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}') &= k_{ll'}(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}_{l\ell}(\boldsymbol{x}, \boldsymbol{X}) \boldsymbol{G}_\ell(\boldsymbol{X})^{-1} \boldsymbol{k}_{\ell l}(\boldsymbol{X}, \boldsymbol{x}'), \end{aligned}$$

where $\boldsymbol{k}_{l\ell}(\boldsymbol{x}, \boldsymbol{X}) = \boldsymbol{k}_{l\ell}(\boldsymbol{X}, \boldsymbol{x})^\top$ is a row vector with elements $k_{ll_N}(\boldsymbol{x}, \boldsymbol{x}_n)$, *i.e.*, the covariance between the source f_l at location $\boldsymbol{x}$ and f_{l_n} at $\boldsymbol{x}_n$. We also introduce a short-hand notation of the noise-corrected kernel Gram matrix $\boldsymbol{G}_\ell(\boldsymbol{X}) = \boldsymbol{K}_{\ell\ell}(\boldsymbol{X}, \boldsymbol{X}) + \boldsymbol{\Sigma}_\ell \in \mathbb{R}^{N \times N}$ and $\boldsymbol{\Sigma}_\ell = \operatorname{diag}(\sigma_{l_1}^2, \ldots, \sigma_{l_N}^2)$.

5.3.2 Multi-source Bayesian quadrature

The multi-output GP can be integrated and gives rise to a quadrature rule similar to VBQ (*cf.* Section 2.2). Let Z_1 denote the random variable

representing the integral of interest Z_1 of (5.1). The integral over the l^{th} source is *a priori*—just as in the single-source case—$\mathsf{Z}_l \sim \mathcal{N}(\mathfrak{m}_l, \mathfrak{v}_l)$ with

$$\mathfrak{m}_l = \int_{\mathcal{X}} m_l(\boldsymbol{x}) \, \mathrm{d}\nu(\boldsymbol{x}),$$
$$\mathfrak{v}_l = \mathfrak{k}_l = \iint_{\mathcal{X}} k_{ll}(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}'),$$

although we only care about Z_1.
The posterior over Z_1 given the data triplets $\mathcal{D}$ is a univariate Gaussian with mean and variance

$$\mathfrak{m}_{1\,|\,\mathcal{D}} \coloneqq \mathbb{E}[\mathsf{Z}_1 \,|\, \mathcal{D}] = \mathfrak{m}_1 + \boldsymbol{\kappa}_{1\ell}(\boldsymbol{X})^\top \boldsymbol{G}_\ell(\boldsymbol{X})^{-1}(\boldsymbol{y}_\ell - \boldsymbol{m}_\ell(\boldsymbol{X})),$$
$$\mathfrak{v}_{1\,|\,\mathcal{D}} \coloneqq \mathbb{V}[\mathsf{Z}_1 \,|\, \mathcal{D}] = \mathfrak{k}_{11} - \boldsymbol{\kappa}_{1\ell}(\boldsymbol{X})^\top \boldsymbol{G}_\ell(\boldsymbol{X})^{-1}\boldsymbol{\kappa}_{\ell 1}(\boldsymbol{X}),$$

where the kernel mean of source 1 follows the notation scheme from Section 2.2,

$$\boldsymbol{\kappa}_{\ell 1}(\boldsymbol{X}) = \int_{\mathcal{X}} \boldsymbol{k}_{1\ell}(\boldsymbol{x}, \boldsymbol{X}) \, \mathrm{d}\nu(\boldsymbol{x}).$$

Just as in VBQ, it is desired that the kernel be integrable, although numerical integration of the kernel can be advantageous over directly solving the costly integral in some cases.

We choose an intrinsic coregionalization model (ICM) [6, § 4.2.2] with kernel

$$k_{ll'}(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{B}_{ll'}\hat{k}(\boldsymbol{x}, \boldsymbol{x}'), \tag{5.2}$$

where $\boldsymbol{B} \in \mathbb{R}^{L\times L}$ is a positive definite matrix. (5.2) is the simplest extension of a single-source kernel $\hat{k}(\boldsymbol{x}, \boldsymbol{x}')$ to the multi-source case which the correlation between the sources and input locations are assumed to factorize. In other words, the assumption in the ICM is that the sources can be written as a linear combination of Q basis functions u_q that are each distributed according to a zero-mean GP but that all share the same kernel $\hat{k}$,

$$f_l(\boldsymbol{x}) = \sum_{r=1}^{Q} a_{l,q} u_q(\boldsymbol{x}).$$

R determines the rank of $\boldsymbol{B}$. This model is a special case of the linear model of coregionalization (LMC) that considers the more general case that each u_q has its own kernel $\hat{k}_q$, which allows to incorporate different lengthscales of the sources.
If $\hat{k}(\boldsymbol{x}, \boldsymbol{x}')$ is integrable analytically, $k_{ll'}(\boldsymbol{x}, \boldsymbol{x}')$ will be, too, and thus retains the favorable property of a BQ-kernel. A typical choice for $\hat{k}$ is the squared-exponential, aka. RBF kernel $\hat{k}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\|x-x'\|_2^2/2\lambda^2)$ with no dependence on the sources l and l'. While considering the ICM here for proof-of-concept, this model can easily be extended *e.g.*, to a LMC without challenging integrability of k. Doing so would untie the lengthscales between sources, but would also introduce $L-1$ additional generally unknown kernel parameters. The simpler ICM is also used by Xi et al. [265] to establish convergence rates for a multi-output BQ rule. In particular, they find the convergence rate of multi-output BQ to be the same as for VBQ.

[6]: Alvarez et al. (2012), 'Kernels for vector-valued functions: A review'

[265]: Xi et al. (2018), 'Bayesian quadrature for multiple related integrals'

5.4 Active design for multi-source BQ

Chapter 3 established multiple approaches to active learning in BQ. Due to linearity of the integral operator, the optimal utility functions are analytically tractable. This result is not compromised in multi-source BQ. However, if the goal was to learn multiple integrals simultaneously, it would be challenging to define an objective that quantifies the overall improvement without sacrificing the accuracy of individual sources. In our setting, the design of an active learning scheme is unambiguous, where information or variance criteria can be formulated w.r.t. the primary source.

The goal here is to obtain a sequence of actions to select new source-location-evaluation triplets that are informative about the primary source. We adopt a sequential approach to acquiring new data in order to overcome the combinatorial explosion of the joint optimization over N nodes (*cf.* Section 3.2.3). Besides feasibility, the lack of exact model knowledge motivates a loop in which the model is repeatedly updated with new observations, leading to an implicit adaptivity of the active learning policy. To this end, BQ is placed in a loop where it is iteratively fed with $N_\star$ new observation triplets $(\boldsymbol{\ell}_\star, \boldsymbol{X}_\star, \boldsymbol{y}_{\boldsymbol{\ell}_\star})$ in the general multi-step look-ahead (*non-myopic*) approach. In practice we resort to a *myopic* approximation and optimize for a single new observation triplet $(l_\star, \boldsymbol{x}_\star, y_{l_\star})$ at a time.

Section 5.4.1 treats the generalization of the standard VBQ acquisitions introduced in Section 3.2.1 and 3.2.2 to the multi-source setting. All these utilities give rise to the same acquisition policy in the absence of cost and are thus not greatly differentiated between in the literature. Intriguingly, the policies do not coincide for AMSBQ if cost is accounted for in the acquisition functions, as will be shown and discussed in Section 5.4.2.

5.4.1 Policies for multi-source Bayesian quadrature

In the absence of any notion of evaluation cost (or if all sources come at the same cost), the utility functions from VBQ generalize straightforwardly to the multi-source case. The VBQ case can be recovered by setting the number of sources to one.

Mutual Information From an information theoretic perspective, new source-location pairs $(\boldsymbol{\ell}_\star, \boldsymbol{X}_\star)$ can be chosen such that they jointly maximize the mutual information (MI) $I[\mathsf{Z}_1; \mathbf{y}_{\boldsymbol{\ell}_\star}]$ between the integral of the primary source Z_1 and a set of new but yet unobserved evaluations $\mathbf{y}_{\boldsymbol{\ell}_\star}$. In terms of the individual and joint differential entropies over Z_1 and $\mathbf{y}_{\boldsymbol{\ell}_\star}$, $I[\mathsf{Z}_1; \mathbf{y}_{\boldsymbol{\ell}_\star}] = H[\mathsf{Z}_1] + H[\mathbf{y}_{\boldsymbol{\ell}_\star}] - H[\mathsf{Z}_1, \mathbf{y}_{\boldsymbol{\ell}_\star}]$. Section 5.3 and Section 5.3.2 imply that both Z_1 and $\mathbf{y}_{\boldsymbol{\ell}_\star}$ are normally distributed and so is their joint. Thus, the mutual information criterion straightforwardly generalizes from (3.6). Since there is no explicit dependence on the value taken by $\mathbf{y}_{\boldsymbol{\ell}_\star}$, we (sloppily) express the mutual information as a function of the new source-location pairs $(\boldsymbol{\ell}_\star, \boldsymbol{X}_\star)$,

$$I[\mathsf{Z}_1; \boldsymbol{\ell}_\star, \boldsymbol{X}_\star] = -\frac{1}{2} \log\left(1 - \rho^2_{1\boldsymbol{\ell}_\star|\mathcal{D}}(\boldsymbol{X}_\star)\right) \tag{5.3}$$

with the scalar correlation (3.5) adapted to the multi-source setting

$$\rho^2_{1\boldsymbol{\ell}_\star|\mathcal{D}}(\boldsymbol{X}_\star) = \frac{\boldsymbol{\kappa}_{1\boldsymbol{\ell}_\star|\mathcal{D}}(\boldsymbol{X}_\star)^\top \boldsymbol{C}^{-1}_{\boldsymbol{\ell}_\star|\mathcal{D}}(\boldsymbol{X}_\star)\, \boldsymbol{\kappa}_{\boldsymbol{\ell}_\star 1|\mathcal{D}}(\boldsymbol{X}_\star)}{\mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D}]} \in [0,1]. \tag{5.4}$$

$\boldsymbol{C}_{\boldsymbol{\ell}_\star|\mathcal{D}}(\boldsymbol{X}_\star) = \boldsymbol{K}_{\boldsymbol{\ell}_\star\boldsymbol{\ell}_\star|\mathcal{D}}(\boldsymbol{X}_\star, \boldsymbol{X}_\star) + \boldsymbol{\Sigma}_{\boldsymbol{\ell}_\star} \in \mathbb{R}^{N_\star \times N_\star}$ denotes the noise-corrected posterior covariance matrix. The derivation is identical to the single-source case (*cf.* Section 3.2) and expanded on in Section B.2. In the one-step look-ahead case ($N_\star = 1$),

$$\rho^2_{1l_\star|\mathcal{D}}(\boldsymbol{x}_\star) = \frac{\kappa^2_{1l_\star|\mathcal{D}}(\boldsymbol{x}_\star)}{v_{l_\star|\mathcal{D}}(\boldsymbol{x}_\star)\, \mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D}]}$$

is the bivariate squared correlation between Z_1 and $y_{l_\star}$, where $v_{l_\star|\mathcal{D}}$ is the scalar version of $\boldsymbol{C}_{\boldsymbol{\ell}_\star|\mathcal{D}}$.

Variance-Based Acquisitions The generalized form of variance reduction on the integral also accounts for variance reduction achieved by evaluating other information sources than the primary. The policy selects $(\boldsymbol{\ell}_\star, \boldsymbol{X}_\star)$ such that the variance on Z_1 shrinks maximally. As MI, the integral variance reduction (IVR) normalized by the current integral variance $\mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D}]$ generalizes from the VBQ expression (3.7)

$$\frac{\Delta\mathbb{V}[\mathsf{Z}_1; \boldsymbol{\ell}_\star, \boldsymbol{X}_\star]}{\mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D}]} = \frac{\mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D}] - \mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D} \cup (\boldsymbol{\ell}_\star, \boldsymbol{X}_\star, \boldsymbol{y}_{\boldsymbol{\ell}_\star})]}{\mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D}]} = \rho^2_{\boldsymbol{\ell}_\star|\mathcal{D}}(\boldsymbol{X}_\star) \tag{5.5}$$

Other acquisition functions introduced in Section 3.2.2 can be extended to the multi-source setting in an analogous manner. The variance-based acquisitions are monotonic transformations of (5.3) and therefore, they share the same global maximizer $\boldsymbol{X}_\star$ also in the case of multiple information sources. The same is true for other monotonic transformations of the squared correlation, and acquisitions like the maximization of integral precision (IP) can be considered equivalently.

5.4.2 Cost-sensitive acquisition functions

When there is a location and/or source-dependent cost associated to evaluating the information sources (*cf.* Section 5.1), the utility function should trade off the improvement made on the integral against the overall budget spent for prospective function evaluations,

$$c_{\boldsymbol{\ell}_\star}(\boldsymbol{X}_\star) = \sum_{i=1}^{N_\star} c_{l_i}(\boldsymbol{x}_i). \tag{5.6}$$

This is achieved by considering the ratio of a cost-agnostic BQ utility and the corresponding cost function.

Definition 5.4.1 *(Acquisition rate) In presence of cost* (5.6)*, a* BQ *acquisition rate is the ratio between a cost-agnostic* BQ *acquisition function and the cost function,*

$$\hat{\alpha}_{\boldsymbol{\ell}_\star}(\boldsymbol{X}_\star) = \frac{\alpha_{\boldsymbol{\ell}_\star}(\boldsymbol{X}_\star)}{c_{\boldsymbol{\ell}_\star}(\boldsymbol{X}_\star)}.$$

$\alpha_{\ell_\star}$ is any of the above multi-source acquisition functions.[2] This ratio can be interpreted as a *rate* as it bears the units of the utility function divided by units of cost. The notion of a rate becomes clearer when considering for example the mutual information utility (5.3) with cost measured in terms of evaluation time: the unit is $\frac{\text{bits}}{\text{second}}$, *i.e.*, a rate of information gain. This construction has an important consequence: Modification of the VBQ utility function (*i.e.*, the numerator), even by a monotonic transformation, changes the maximizer of the cost-adapted acquisition rate and hence, also the acquisition *policy*. In other words, the degeneracy of BQ acquisition functions in terms of the policy they induce in the absence of cost is lifted when evaluation cost is included, firstly, because the argmax of each acquisition is shifted differently with cost, and, secondly, because acquisition *values* from different sources are discriminated against each other now. As will be discussed below, not all monotonic transformations yield a sensible acquisition policy; indeed, some display pathological behavior.

2: The reference to the primary source is not explicitly taken care of in this notation. All acquisitions are still defined w.r.t. the integral Z_1.

The adapted non-myopic acquisition rates for the BQ utilities mutual information (MI) (5.3) and integral variance reduction (IVR) (5.5) are

$$\hat{\alpha}^{\text{MI}}_{\ell_\star}(\boldsymbol{X}_\star) \coloneqq \frac{-\log\left(1-\rho^2_{1\ell_\star|\mathcal{D}}(\boldsymbol{X}_\star)\right)}{c_{\ell_\star}(\boldsymbol{X}_\star)}$$
$$\hat{\alpha}^{\text{IVR}}_{\ell_\star}(\boldsymbol{X}_\star) \coloneqq \frac{\rho^2_{1\ell_\star|\mathcal{D}}(\boldsymbol{X}_\star)}{c_{\ell_\star}(\boldsymbol{X}_\star)},$$

where we have dropped the factor $1/2$ in MI as an arbitrary scaling factor. It is evident that these acquisition rates do no longer share their maximizer; yet they still induce a meaningful acquisition scheme.

To gain an intuition about the behavior of the acquisition functions, it is instructive to consider their functional dependence on the squared correlation[3] *i.e.*, $\alpha(\rho)$ prior to cost-adjustment. Both MI and IVR have the property to be zero at $\rho^2 = 0$ and thus never select points $\boldsymbol{X}_\star$ that are uncorrelated with the integral Z, no matter the cost, *e.g.*, locations that have already been observed exactly (with $\sigma^2 = 0$). Such points do not update the posterior of the integral Z when conditioned on. In VBQ these locations are the minimizers of all acquisition functions and thus excluded no matter their value. This is not ensured for the cost-adapted acquisition rates and therefore, they additionally require the numerator to be zero at $\rho^2 = 0$. Hence, not every monotonic transformation of the BQ utility produces a sane acquisition policy in the presence of cost.

3: To avoid notational clutter, we refer to $\rho^2_{1\ell_\star|\mathcal{D}}(\boldsymbol{X}_\star)$ as ρ^2 in the following discussion, but still mean the multi-source posterior scalar correlation. Similarly, we sloppily denote multi-source acquisitions as α and their cost-adjusted counterparts as $\hat{\alpha}$.

Consider for example the valid transformation $\rho^2 \mapsto \rho^2 - 1$, which is -1 at $\rho = 0$. Maximizing this utility function corresponds to maximizing the negative integral variance (3.8), *i.e.*, minimizing the integral variance, which is very commonly done in VBQ. Since $\rho^2 \in [0,1]$, $\rho^2 - 1$ is negative everywhere and gets larger (takes a smaller negative value) with larger cost. Hence, when maximized, this acquisition would favor expensive evaluations.

More subtle is the misbehavior of the integral precision (IP) which is another valid way to rephrase the reduction of variance by saying that we want to maximize the *precision* (the inverse variance) of the integral,

$$\mathbb{V}[\mathsf{Z}_1 \mid \mathcal{D} \cup (\boldsymbol{\ell}_\star, \boldsymbol{X}_\star, \boldsymbol{y}_{\boldsymbol{\ell}_\star})]^{-1} \propto (1-\rho^2)^{-1} =: \alpha^{\text{IP}}_{\boldsymbol{\ell}_\star}(\boldsymbol{X}_\star).$$

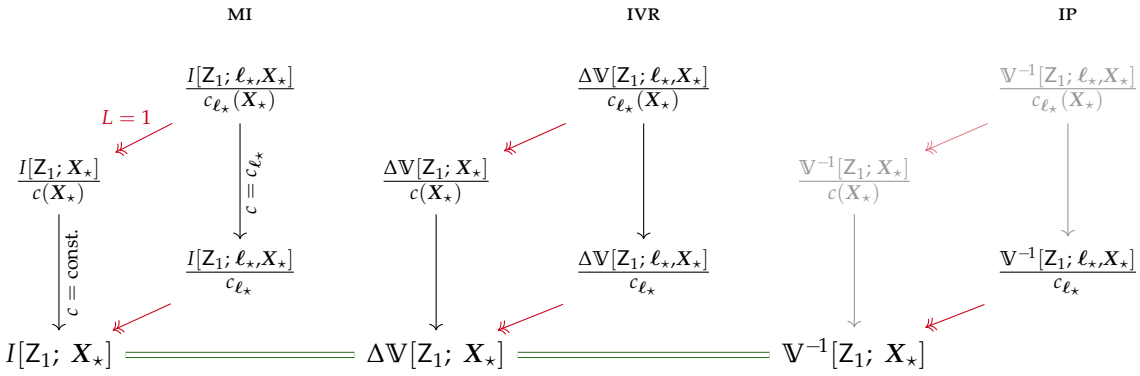


Figure 5.1: The multi-source acquisition cube for a few of the possible acquisition functions. MI, IVR, and IP stand for 'mutual information', 'integral variance reduction', and 'integral precision', respectively. The forward arrows (↙) denote the special case of one source only ($L = 1$) as in the case of VBQ. The downward facing arrows (↓) denote the special case where the cost c is not dependent on the locations $\boldsymbol{X}_\star$. The double-lines (═) between nodes denote that these acquisition functions are equivalent in the sense that they yield the same optimal $\boldsymbol{X}_\star$. The two grayed-out acquisitions for IP highlight that they exhibit non-favorable behavior (Proposition 5.4.1). The bottom front row in the cube denotes the special case of VBQ ($L = 1$ and $c(\boldsymbol{X}_\star) = \text{const.}$) where all three acquisition policies (MI, IVR, IP) coincide.

This function is positive everywhere and its cost-adjusted version $\hat{\alpha}^{\text{IP}}$ has the desired behavior of favoring low-cost evaluations. However, α^{IP} is non-zero at $\rho^2 = 0$ and therefore, it does not exclude points of zero correlation when they come at sufficiently low cost, and in experiments we observe it getting stuck re-evaluating at the location of minimum cost over and over again. We conjecture that this is because IP only encodes an absolute scale of the integral variance but does not quantify any 'improvement' on the integral value.
Improvement can be encoded by maximizing the difference between the prospective and the current integral precision,

$$\mathbb{V}[Z_1 \mid \mathcal{D} \cup (\ell_\star, \boldsymbol{X}_\star, \boldsymbol{y}_{\ell_\star})]^{-1} \; - \; \mathbb{V}[Z_1 \mid \mathcal{D}]^{-1} \propto \frac{\rho^2}{1-\rho^2} =: \alpha^{\text{IPI}}_{\ell_\star}(\boldsymbol{X}_\star),$$

labeled as *integral precision increase (IPI)*. The relative way to phrase the change of integral precision again retains the favorable properties of a *cost-adjustable* acquisition function. We summarize the necessary conditions of a BQ acquisition function to be amenable to usage in the cost-aware setting.

Proposition 5.4.1 *(Cost-adjustable BQ acquisition function) An acquisition function α in Bayesian quadrature is called cost-adjustable if it satisfies the following requirements as a function of scalar correlation ρ (5.4),*

(i) $\alpha(\rho) > 0$ *for all* $\rho \in [0, 1]$, *(positivity)*
(ii) $\alpha(\rho = 0) = 0$, *(zero point)*
(iii) $\alpha(\rho_2) > \alpha(\rho_1) \;\Leftrightarrow\; \rho_2 > \rho_1$. *(monotonicity)*

An acquisition function that fulfills these properties gives rise to a desirable policy when divided by a cost function.

Figure 5.1 illustrates the augmentation of utility functions from VBQ with multiple information sources and cost. The dependence of these acquisition functions on the squared correlation $\rho^2 \in [0, 1]$ in the absence of cost is shown in Figure 3.2. All acquisitions are strictly monotonically increasing functions of ρ^2. Among the sane acquisition rates that are

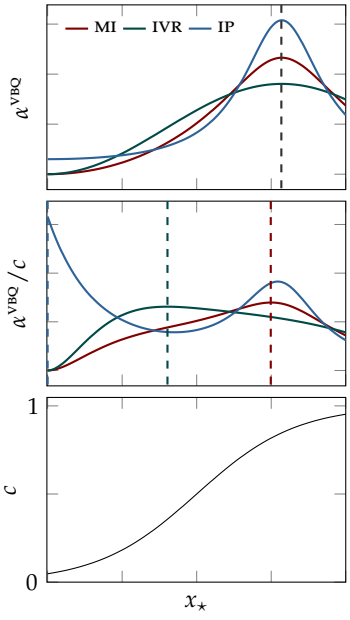


Figure 5.2: A selection of VBQ acquisitions α^{VBQ} as a function of univariate $x_\star$ and myopic step ($N_\star = 1$) for a synthetic $\rho^2(x_\star)$. Without cost, their maximizers coincide (*top*), but when divided by an input-dependent cost $c(x_\star)$ (*bottom*), the maximizers disperse (indicated by the dashed vertical lines) (*middle*). For implications, *cf.* Section 5.4.2.

zero at $\rho^2 = 0$, the differences in the corresponding policy can also be understood from the functional dependence on ρ. MI diverges at perfect correlation $\rho^2 \to 1$. Therefore, and since the cost c lies in $[\delta, 1]$, MI will always take a 'perfect step' to learn the integral exactly, *i.e.*, it will always select the points $\boldsymbol{X}_\star$ with correlation $\rho^2(\boldsymbol{X}_\star) = 1$, if the step is available and no matter the cost. IVR, however, is finite at $\rho^2 = 1$ and trades off cost against correlation even if the perfect $\boldsymbol{X}_\star$ with $\rho^2(\boldsymbol{X}_\star) = 1$ exists. In Figure 5.2 we plot MI, IVR, and IP versus a univariate $x_\star$ for the synthetic choice $\rho^2(x_\star) = 0.95 \sin^2(10x_\star)$, $x_\star \in [0, 0.2]$ and a myopic step ($N_\star = 1$). In the pure VBQ situation, the locations of all their maxima coincide, but as soon as a non-constant cost $c(x_\star)$ is applied, the shapes of the acquisition functions become relevant which discriminates their $\boldsymbol{X}_\star$ and lifts the degeneracy in policies. MI tends more towards higher correlation than IVR, the maximizer of which moves further towards locations of lower cost. While MI and IVR act differently, they are both sensible choices for acquisition functions in AMSBQ. In fact for low to mid-ranged values of $\rho^2 \lesssim 0.5$ where MI is approximately a linear function of ρ^2 they roughly coincide.
The choice of acquisition ultimately depends on the application and the user, who may choose which measures of improvement on the integral and cost to trade off.

5.5 Experiments

The key practical applications for AMSBQ is to solve integrals of expensive-to-evaluate black-box functions that are accompanied by cheaper approximations, potentially in a setting where a finite budget is available. Typical applications are models of complex nonlinear systems that need to be tackled computationally. With evaluations being precious, the goal is to get a decent estimate of the integral with as little budget as possible, rather than caring about floating-point precision. In the experiments, we focus on the rear vertices of the acquisition cube Figure 5.1, *i.e.*, multiple sources with source and input-dependent or only source-dependent cost, and separate them into two main experiments:

1. A synthetic multi-source setting with cost that varies in source and location for the purpose of exploring and demonstrating the behavior of the acquisition functions derived in Section 5.4.2.
2. An epidemiological model of the spread of a disease with uncertain input, in which two sources correspond to simulations that differ in cost as well as quality of the quantity of interest.

To demonstrate validity in a multi-dimensional setting, we present a bivariate experiment with three sources in Section 5.5.3. We take a myopic approach to all scenarios in that we optimize the acquisition for a single source-location pair a time. The implementation of the GP-model uses `GPy` [98] in `Python` 3.7.

[98]: `GPy` (since 2012), *GPy: A Gaussian process framework in Python*

5.5.1 Multi-source, variable cost

For demonstration purposes, we initially consider a synthetic two-source setting with univariate input. The cost functions depend on both source and location. The experiment's purpose is to demonstrate our findings from Section 5.4.2 and convey intuition about the behavior of the novel

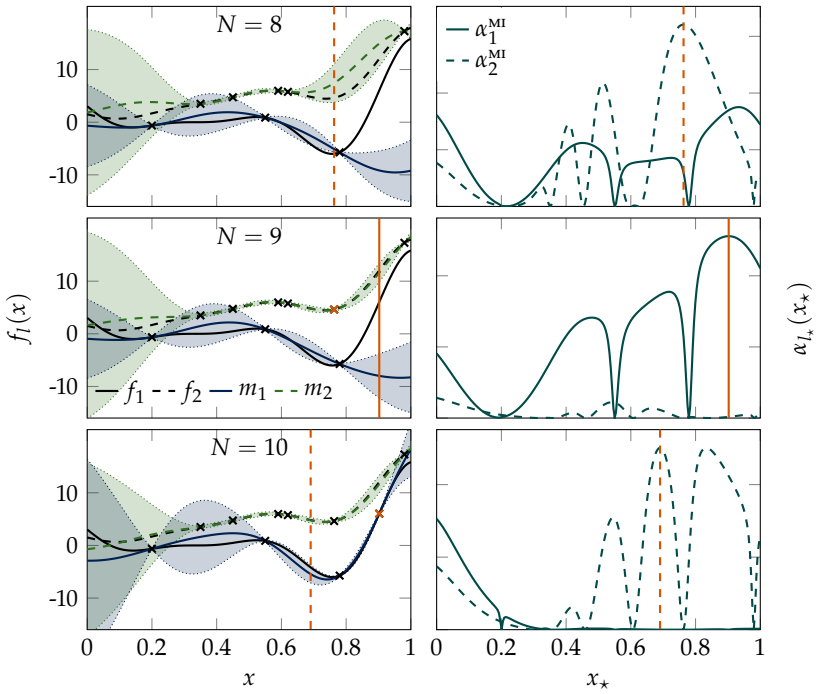


Figure 5.3: Demonstration of the sequential selection of new source-location pairs to query f using the MI acquisition in a two-source setting with a cost function that depends on both source and location (Figure 5.4). *Left column:* The multi-output GP; *right column:* the acquisition function for the primary (solid) and secondary source (dashed) for three consecutive iterations. Vertical orange lines indicate the location and source of the prospective query.

acquisition functions. The sources we consider have been suggested by [74] with the primary and secondary source

$$f_1(x) = (6x-2)^2 \sin(12x-4)$$
$$f_2(x) = \frac{1}{2} f_1(x) + 10x$$

for $x \in [0,1]$, respectively. The cost functions both take the form of a scaled and shifted logistic function in a way that the cost lies in $(0,1]$ (*cf.* Figure 5.4). The costs of both sources converge to the same value close to $x_\star = 0$; for larger $x_\star$, f_2 is two orders of magnitude cheaper than f_1.

Figure 5.3 shows snapshots of three consecutive query decisions taken by the MI multi-source acquisition. The GP model (depicted in the left column) has been initialized with 3 data points in the primary and 5 in the secondary source and merely the noise variance was constrained to 10^{-2}. The MI acquisition given the current state of the GP is shown on the right—the top left frame is shown for MI, IVR, and IP in Figure 5.6 to emphasize the pathology of IP and to highlight the subtle difference between MI and IVR in practice. The acquisition function is optimized using the L-BFGS-B optimizer in `scipy`. A later state of the acquisition loop shown in Figure 5.5 demonstrates that AMSBQ does not query f_2 where the source costs are almost identical for $x_\star \lesssim 0.2$. This is because the two sources are not perfectly correlated and evaluating f_1 always conveys more information about Z_1 than f_2. The fact that c_2 decreases with increasing $x_\star$ is nicely represented in the increasing height of the maxima of the dashed acquisition function for the secondary source in the top right frame of Figure 5.3.

For assessing the performance of AMSBQ, we compare against VBQ and a percentile estimator (PE) that both operate on the primary source. The latter is obtained by separating the domain into intervals that contain

[74]: Forrester et al. (2007), 'Multi-fidelity optimization via surrogate modelling'

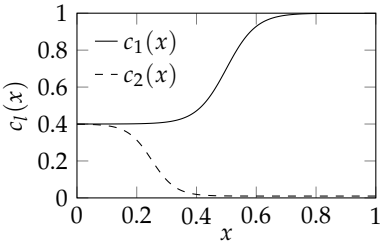


Figure 5.4: The artificial cost assigned to the Forrester functions and used for demonstrating the cost-sensitive acquisition scheme in Figure 5.3.

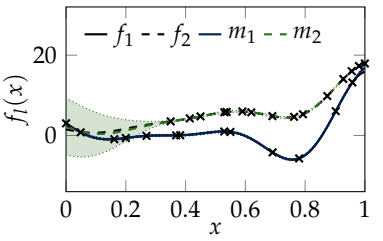


Figure 5.5: A later state for the experiment shown in Figure 5.3. Note the absence of f_2 evaluations for small x where c_1 and c_2 are similar.

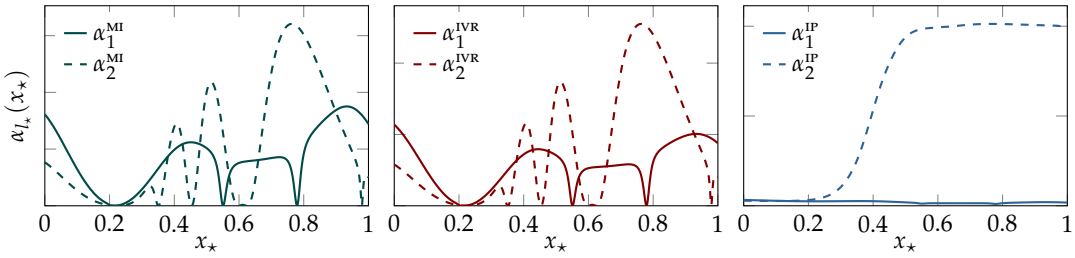


Figure 5.6: MI, IVR, and IP acquisitions for the top row of Figure 5.3. MI and IVR do not differ a lot, *i.e.*, the correlation ρ is rarely large enough for MI to leave the linear regime. MI puts slightly more emphasis on the primary source where $x_\star$ is close to 1. This indicates that the correlation between Z and $y_\star$ quite large there. The bottom plot displays the pathology of IP, where the acquisition for the secondary source essentially follows the inverse cost c_2. Note that the vertical scale is irrelevant and does not even have the same unit across plots.

the same probability mass of the probability measure ν and summing up the function values at these nodes. For the uniform integration measure used here, this is equivalent to a right Riemann sum. We assume that GP inference comes at negligible cost as compared to the function evaluations and thus consider cost to be incurred purely by querying the information sources.

To render the integration problem slightly more difficult, we modify the Forrester functions to vary on a smaller length scale by adding a sinusoidal term and adapting some parameters, s.t.

$$f_1(x) = (6x-2)^2 \sin(12x-4) - (2-x)^2 \sin(36x)$$
$$f_2(x) = \frac{3}{4} f_1(x) + 16\left(x - \frac{1}{2}\right) + 10$$

which we integrate from 0 to 1 against a uniform measure (*cf.* Figure 5.8, top left). To avoid over- or underfitting, we set a conservative gamma prior on the lengthscale with a mode at a small fraction of the domain $[0, 1]$ for both VBQ and AMSBQ, and assume zero observation noise. Due to the construction of the coregionalization matrix $\boldsymbol{B} = \boldsymbol{W}\boldsymbol{W}^\top + \operatorname{diag}(\boldsymbol{\eta})$ AMSBQ has six more hyperparameters than VBQ. Therefore, AMSBQ is more prone to over-/underfitting, and we further set a prior on $\boldsymbol{B}$ (*cf.* Section 5.3.2) with parameters estimated from the initial three data points using empirical Bayes. This is to avoid initial over- or underestimation of the correlation between sources, which would either cause the active scheme to select only f_2 or only f_1, respectively. Compared to the previous experiment, the cost is changed to have a minimum, but still composed of a sum of logistic functions and normalized to be in $(0, 1]$ (Figure 5.8, top right). The effect of these cost functions on the final state is depicted in Figure 5.7. Furthermore, this setting reveals the pathology of the IP acquisition (*cf.* Section 5.4.2) that everlastingly re-evaluates the secondary source at the location of minimal cost. The convergence behavior of the well-behaved acquisition functions MI and IVR are displayed in Figure 5.8 (bottom) in comparison to VBQ and PE. The hyperparameters of the GP are optimized after every newly acquired node, both for VBQ and AMSBQ. Figure 5.8 shows the superior performance of both AMSBQ methods in arriving close the true integral with little budget. The vertical jumps in the AMSBQ methods occur when f_2 is evaluated at cheaper cost.

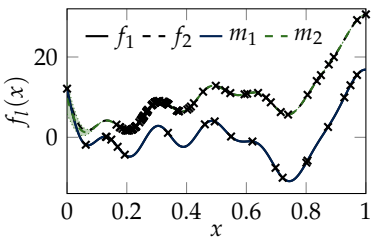


Figure 5.7: Final state of the GP for the second experiment explained in Section 5.5.1 and shown in Figure 5.8 ('wigglified Forrester'). Note the increasing density of evaluations of the secondary source where the cost is minimal, and the lack of f_2 queries where $c_1(x) \simeq c_2(x)$. The leftmost evaluation is at the primary source. See Figure 5.8 for the cost functions. In this experiment, the IP acquisition exclusively evaluates at the location of the minimum of the secondary cost function and is thus stuck.

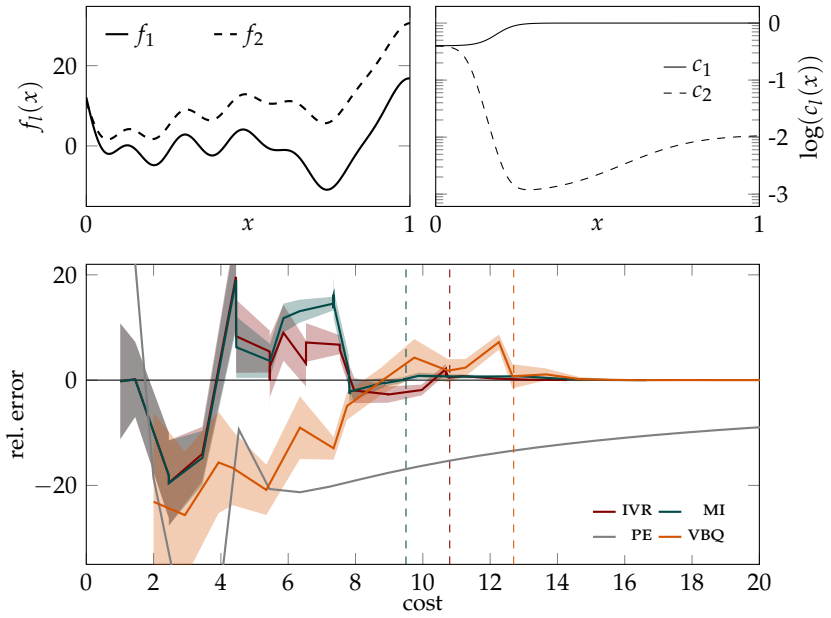


Figure 5.8: *Top left:* the wigglified Forrester functions with f_1 and f_2 primary/secondary source, respectively; *top right:* the cost functions used; *bottom:* relative error $\mathbb{E}[Z]-Z/Z$ with two std. deviations (shaded) as a function of normalized cost for the AMSBQ acquisitions MI and IVR compared to VBQ and a PE. Vertical dashed lines are a visual help to indicate the cost spent to achieve acceptable accuracy.

5.5.2 A simulation of infections

We now consider multi-source models in which sources come at input-independent cost, a.k.a. multi-fidelity models (bottom rear MI vertex in Figure 5.1). We choose an epidemiological model in which evaluating the primary source requires running numerous stochastic simulations and the secondary source solves a system of ordinary differential equations. Epidemiological models deal with simulating the propagation of an infectious disease through a population. The SIR model forms the base for many compartmental models and assumes a population of fixed size N where at any point in time, each individual is in one of three states—susceptible, infected, and recovered (SIR)—with sizes N_S, N_I, and N_R [138].[4] The dynamics are determined by stochastic discrete-time events of individuals changing infection state, for which Poisson processes (*i.e.*, exponentially distributed inter-event times) are commonly assumed [58, *e.g.*,]. In the thermodynamic limit where N is large, the average dynamics is governed by a system of ordinary differential equations (ODEs) that does not admit a generic analytic solution. The thermodynamic limit can safely be assumed for any realistic population size. When the population size is large, the SIR model can be described by the following system of ordinary differential equations,

$$\begin{aligned} \frac{\mathrm{d}\, N_S}{\mathrm{d}\, t} &= -a\frac{N_S N_I}{N}, \\ \frac{\mathrm{d}\, N_I}{\mathrm{d}\, t} &= a\frac{N_S N_I}{N} - bN_I, \\ \frac{\mathrm{d}\, N_R}{\mathrm{d}\, t} &= bN_I, \end{aligned} \tag{5.7}$$

in which a is the rate of infection and b the rate of recovery. The SIR model is easily extended to more elaborate epidemiological models by accounting for more compartments. Extensions accommodate additional effects *e.g.*, vital dynamics, immunity, incubation time [115]. Some of these extensions serve as a general model refinement, others are relevant

[138]: Kermack and McKendrick (1927), 'A contribution to the mathematical theory of epidemics'

4: Compartmental models for epidemiological modeling have experienced an extraordinary revival following the outbreak of the COVID-19 pandemic [*e.g.*, 95]. They have been studied from a probabilistic perspective [221] to predict infection numbers with uncertainty. Even prior to COVID-19, these models have been foundational to developing policies for combatting epidemics (*e.g.*, tuberculosis [180]). The simplistic study presented here has been carried out prior to COVID-19.

[95]: Giordano et al. (2020), 'Modelling the COVID-19 epidemic and implementation of population-wide interventions in Italy'

[221]: Schmidt et al. (2021), *A probabilistic state space model for joint inference from differential equations and data*

[180]: Moualeu et al. (2015), 'Optimal control for a tuberculosis model with undetected cases in Cameroon'

[58]: Daley and Gani (1999), *Epidemic modelling: An introduction*

[115]: Hethcote (2000), 'The mathematics of infectious diseases'

to specific diseases. More compartments inevitably introduce new free parameters to describe the transition rates between them. The dynamics of the SIR model and an extension that accounts for incubation time is illustrated in Figure 5.9.

Statistical properties, however, are not captured by the description through ODEs and call for a stochastic model. The Gillespie algorithm [92, 93] enables discrete and stochastic simulations in which every trajectory is an exact sample of the solution of the 'master equation' that defines a probability distribution over solutions to a stochastic equation. In the SIR model, the rate constants are time-independent and thus, the underlying process is Markovian in which the event times are Poisson distributed. Here, an event denotes the transition of one individual from one compartment to another (*e.g.*, $N_I \to N_R$).

[92]: Gillespie (1976), 'A general method for numerically simulating the stochastic time evolution of coupled chemical reactions'

[93]: Gillespie (1977), 'Exact stochastic simulation of coupled chemical reactions'

Experimental details For the AMSBQ experiment, we assume that we know the recovery rate b, but we are uncertain about the infection rate a. Therefore, we rescale the ODEs and place a shifted gamma prior on a/b that starts at $a/b = 1$ and has shape and scale parameters 5 and 4 respectively. With this prior we encode our belief that the infection rate is significantly larger than the recovery rate so an offset of the epidemic is very likely. Also, we set the population size to $N = 100$ to be well below the thermodynamic limit and set one individual to be infected initially.[5] We are interested in the expected maximum number $\mathbb{E}_a[\max_t N_I(t)]$ of simultaneously infected individuals and the time this maximum occurs $\mathbb{E}_a[\arg\max_t N_I(t)]$, which might be relevant for predicting hospitalization numbers. Querying the primary source f_1 for the quantities of interest as a function of a requires numerous realizations—we average over 1000 trajectories—of a stochastic four-compartments epidemic model (an extension to the SIR model) using the Gillespie algorithm [92, 93]. In addition to the base model (SIR), we include the state 'exposed' (E), in which individuals are infected but not yet infectious. The modified system of ODEs with assumed known

5: For significantly higher population sizes, the system of ODEs yield results that are almost indistinguishable from the average over many stochastic realizations. At the same time, the latter become prohibitively expensive. The experimental setup has been chosen here for demonstrative purposes, not for practical relevance.

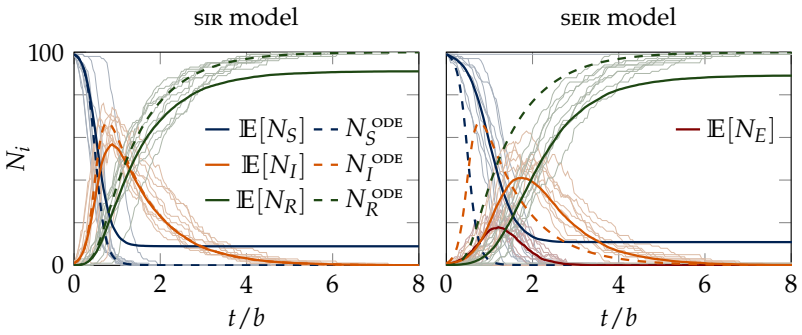


Figure 5.9: Ten trajectories (thin lines) of the stochastic SIR *(left)* and SEIR model *(right)* for $a/b = 10$. Solid lines display the mean over 100 of these stochastic realizations; dashed lines show the solution of the ODEs. The non-zero mean of the stochastic simulations in susceptible individuals after decline of the infection is caused by trajectories in which the infection dies out before an outbreak.

incubation time γ^{-1} is

$$\begin{aligned}\frac{\mathrm{d}\,N_S}{\mathrm{d}\,t} &= -a\frac{N_S N_I}{N},\\ \frac{\mathrm{d}\,N_E}{\mathrm{d}\,t} &= a\frac{N_S N_I}{N} - \gamma N_E,\\ \frac{\mathrm{d}\,N_I}{\mathrm{d}\,t} &= \gamma N_E - bN_I,\\ \frac{\mathrm{d}\,N_R}{\mathrm{d}\,t} &= bN_I.\end{aligned} \tag{5.8}$$

In the stochastic model, the maximum value and time are computed for each trajectory and subsequently averaged over. In our implementation, each query of f_1 takes $\sim 16\,\mathrm{s}$ on a laptop's CPU. The secondary source f_2 solves the system of ODEs (5.8) for given a and computes the maximum value and time for the resulting function $N_I(t)$, which takes about $8 \cdot 10^{-3}\mathrm{s}$ to evaluate. For the integration task, we absorb the prior on a/b in the black-box function for all methods and integrate against a uniform measure.

As in previous experiments, we set a gamma prior on the kernel lengthscale λ, a prior on the coregionalization matrix $\boldsymbol{B}$, and the noise variance to zero as in Section 5.5.1. Both VBQ and AMSBQ are given the same initial value of f_1, and AMSBQ additionally gets the value of f_2 at the same location, as well as one more random datum from f_2. This is justified since AMSBQ needs to learn more hyperparameters than VBQ and secondary source evaluations are very cheap. Otherwise, if the initial evaluations of f_2 were further apart than the prior lengthscale from the locations of the initial primary datum, virtually zero correlation would be inferred between the sources, and the primary source would be evaluated until a sampled location roughly coincides with locations where the secondary sources have been evaluated.

Figure 5.9 shows the SIR and SEIR models ((5.7) and (5.8), respectively). We also use the SIR model for solving the ODEs even though the stochastic model simulates the SEIR model. The purpose of doing so is to mimic secondary sources that simplify the primary model by disregarding minor contributions to the overall dynamics. In the stochastic case, there is not always an outbreak of the disease, *i.e.*, the initially infected individual recovers before infecting someone else. This causes the average N_R to level off significantly below 1. For the integrals, only outbreaks are taken into account. The corresponding integrands for the quantities of interest are shown in Figure 5.10.

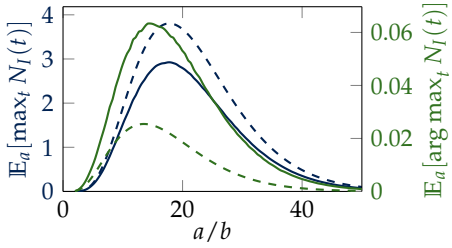


Figure 5.10: Integrands used for the epidemiological model. Solid lines denote the primary source (*i.e.*, stochastic simulations), dashed lines indicate the secondary source (solving the system of ODEs). It is apparent from the function that simply integrating the cheap source introduces a significant bias.

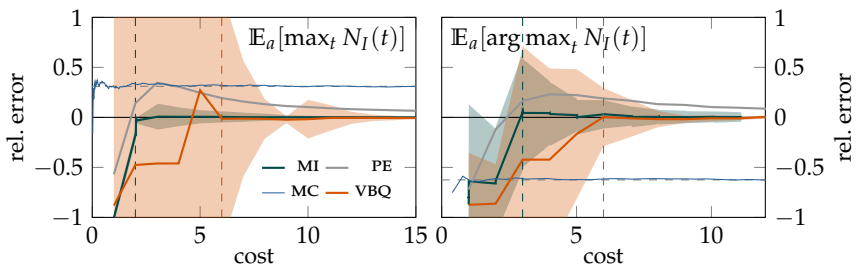


Figure 5.11: Relative error vs. budget spent for the SIR model for the max number of simultaneously infected individuals (left) and for the time after which the maximum occurs (right). Primary source has cost 1.

Figure 5.11 shows the relative error of the AMSBQ estimator against normalized cost as compared to VBQ and PE for $\mathbb{E}_a[\max_t N_I(t)]$ (left) and $\mathbb{E}_a[\arg\max_t N_I(t)]$ (right). The horizontal dashed line shows $Z_2 = \int_{\mathcal{X}} f_2(a)\,\mathrm{d}a$, *i.e.,* the integral of the secondary source with one evolution of a Monte Carlo estimator of f_2. This illustrates that simply using the secondary source for the integral estimate might be computationally cheap, but results in an unknown bias. In the left plot, AMSBQ achieves a good estimate with one additional evaluation of f_1 only, while VBQ takes another six evaluations. Again, the vertical jumps for AMSBQ are caused by evaluations of f_2. The initial high confidence on the integral is caused by the choice of prior on the output scale from the initial data, which is located in the tail of the gamma prior on a. Figure 5.12 displays the order in which AMSBQ evaluates primary and secondary source. It demonstrates that the secondary source is queried considerably more often than the primary source.

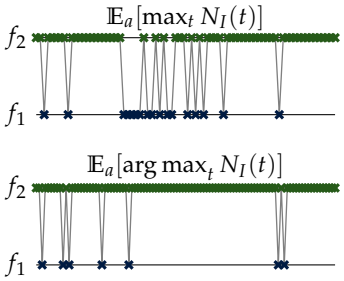


Figure 5.12: Evaluation sequences of primary and secondary source in the SIR experiment.

5.5.3 Bivariate linear combinations of Gaussians

We construct a bivariate integrand f_1 on the domain $[-3,3]^2$ as a linear combination of $J = 20$ normalized Gaussian basis functions

$$f_1(\boldsymbol{x}) = \sum_{k=1}^{K} z_j^1 \varphi_j^1(\boldsymbol{x}),$$
$$\varphi_j^l(\boldsymbol{x}) = (2\pi|\boldsymbol{A}_j^l|)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m}_j^1)^\top(\boldsymbol{A}_j^l)^{-1}(\boldsymbol{x}-\boldsymbol{m}_j^l)\right).$$

The $J = 20$ means are sampled uniformly $\boldsymbol{m}_j^1 \sim \mathrm{Uniform}[-3,3]^2$. Covariance matices $\boldsymbol{A}_j^1$ are constructed according to $\boldsymbol{v}_j^1 \sim \mathcal{N}(0, \boldsymbol{I})$, $\boldsymbol{\eta}_j^1 \sim \mathrm{Uniform}[0,1]^2$, and $\boldsymbol{A}_j^1 := \mathrm{diag}(\boldsymbol{\eta}_j^1) + \boldsymbol{v}_j^1(\boldsymbol{v}_j^1)^\top$. The scalar weights z_j^1 are sampled from a standard Gaussian $z_j^1 \sim \mathcal{N}(0,1)$ and can be negative. Thus, f_1 is not a probability density function but rather a linear combination of Gaussians with varying location, shape, and weight.
We then construct secondary sources f_2 and f_3 consecutively by adding uniform noise to the means, and additive uniform noise to the diagonal of the covariance matrices. Thus, with each additional source, each of the J means get randomly but consecutively shifted up and right, and the basis functions $\varphi_j^l(\boldsymbol{x})$, $l = 2,3$ randomly become wider and flatter. Additionally, we consecutively add Gaussian random noise to the weights z_j which ensures that the true integrals of the secondary sources differ from the integral of the primary source. All sources are depicted in Figure 5.13; the primary source f_1 on the left, and secondary sources f_2 and f_3 in the middle and right respectively. The cost for evaluating the primary source is 1 everywhere, the cost of evaluating f_2 and f_3 are 5% of the primary cost each.

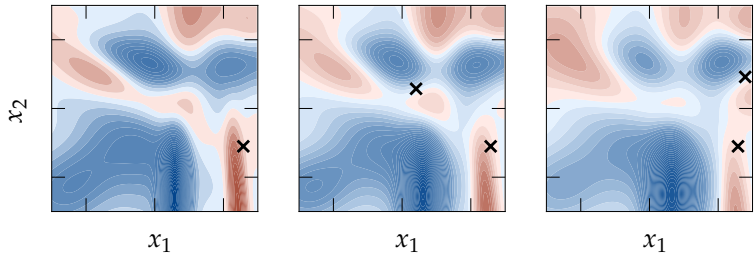


Figure 5.13: Integrands used for the bivariate linear combination of Gaussians. From left to right: primary source f_1 and secondary sources f_2 and f_3. Initial evaluations marked as black crosses.

The priors on the kernel lengthscale and coregionalization matrix $\boldsymbol{B}$ are set analogously to the other experiments already described in Section 5.5.1. AMSBQ is initialized with one evaluation of the primary source and two evaluations each of the secondary sources which amounts to a total initial cost of 1.2 (initial evaluations shown as red dots in Figure 5.13). VBQ is initialized with three evaluations which are needed to get an initial guess for its hyperparameters (initial cost=3). The result is shown in Figure 5.14 which plots relative error of the integral estimate versus the budget spent as well as two standard deviations of the relative error as returned by the model. It is apparent that AMSBQ finds a good solution faster than vanilla-BQ.

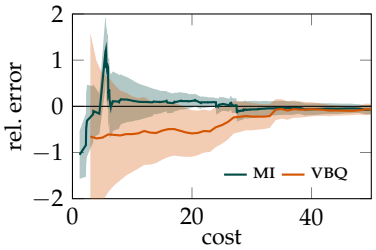


Figure 5.14: Relative error vs. budget spent for vanilla-BQ and AMSBQ.

Figure 5.15 illustrates the sequence of sources chosen by AMSBQ. Secondary source f_2 is chosen more often than secondary source f_3 at equal evolution cost of 0.05. This is intuitive since f_2, by construction, provides more information about f_1 than f_3, but both secondary sources shrink the budget equally when queried. The percentage of number of evaluations for each source after spending a total budget of 50 is 15%, 57%, 28% for sources f_1, f_2, f_3 respectively.

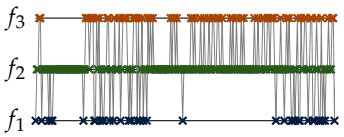


Figure 5.15: Evaluation sequence of the three sources in the 2D experiment over 250 evaluations.

5.6 Discussion

The purpose of the project was to study active learning schemes for multi-source BQ to infer the integral of a primary source while including information from cheaper secondary sources. We discovered that utilities that yield redundant acquisition policies in VBQ give rise to various policies, some desirable and others pathological, when evaluation cost is accounted for and phrased desiderata for sane acquisition functions in BQ. Our experiments illustrate that with the sensible acquisition functions, the AMSBQ algorithm allocates budget to information retrieval more efficiently than traditional methods do for solving expensive integrals.

The multi-source model presented in Section 5.3 can be extended in various ways to increase its expressiveness by using a more elaborate kernel (*e.g.*, one lengthscale per source), or by encoding knowledge about the functions to be integrated, *e.g.*, a probabilistic integrand. For example, it could be used concurrently with a warped BQ model (*cf.* Section 2.5). Generally, applications might come with the complication that the cost function c is unknown a priori and needs to be learned during the active BQ-loop from measurements of the amount of resource required during the queries. A simple example was presented in Section 5.5.2 where the cost was parameterized by a constant, estimated during the initial observations. A probabilistic model of the cost would induce an acquisition function which is not only conditioned on the uncertain model predictions but also on the uncertain cost predictions. Furthermore, as in

other active learning schemes, non-myopic steps for acquiring multiple observations $\boldsymbol{y}_{\ell_\star}$ at once might be beneficial especially when the multi-source model is already known, and does not benefit from being re-fitted to new data. Settings in which multiple evaluations of sources come at lower cost than evaluating sequentially are also conceivable, and would also benefit from a non-myopic treatment. The presented experiments have merely a proof-of-concept character, and the practical benefit of using AMSBQ in real-world applications should be empirically studied.

Bayesian Quadrature on Riemannian Data Manifolds

6

Riemannian manifolds provide a principled way to model nonlinear geometric structure inherent in data. A Riemannian metric on said manifolds determines geometry-aware shortest paths and provides the means to define statistical models accordingly. However, these operations are typically computationally demanding. To ease this computational burden, we advocate probabilistic numerical methods for Riemannian statistics. Following the main thread in this thesis, we focus on Bayesian quadrature (BQ) to numerically compute integrals over normal laws on Riemannian manifolds learned from data. In this task, each function evaluation relies on the solution of an expensive initial value problem. We show that by leveraging both prior knowledge and an active exploration scheme, BQ significantly reduces the number of required evaluations and thus outperforms Monte Carlo methods on a wide range of integration problems. As a concrete application, we highlight the merits of adopting Riemannian geometry with our proposed framework on a nonlinear dataset from molecular dynamics.
The content of this chapter has been published as

> C. Fröhlich, A. Gessner, P. Hennig, B. Schölkopf, and G. Arvanitidis. 'Bayesian quadrature on Riemannian data manifolds'. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021.

The code is publicly available at github.com/froec/BQonRDM.

6.1 Context

The tacit assumption of a Euclidean geometry, implying that distances can be measured along straight lines, is inadequate when data follows a nonlinear trend. This is known as the *manifold hypothesis* (*cf.* Figure 6.1). As a result, probability distributions based on a flat geometry may poorly model the data and fail to capture its underlying structure. Generalized distributions that account for curvature of the data space have been put forward to alleviate this issue. In particular, Pennec [200] proposed an extension of the normal distribution on Riemannian manifolds such as the sphere.

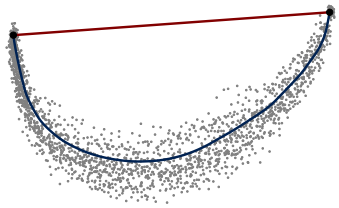


Figure 6.1: Shortest paths according to the Euclidean (—) and the Riemannian (—) view.

The key strategy to use such distributions on general data manifolds is by replacing straight lines with continuous shortest paths, known as *geodesics*, which respect the nonlinear structure of the data. This is achieved by introducing a Riemannian metric in the data space that specifies how data distorts distance and volume locally.
Arvanitidis et al. [10] proposed a maximum likelihood estimation scheme to learn the parameters of a *locally adaptive normal distribution (LAND)* on a data-induced metric space, illustrated in Figure 6.2. A practical limitation of this method is the computationally expensive optimization task that demands repeated numerical integration of the unnormalized density on the manifold, for which no closed-form solution exists. Hence, we

[200]: Pennec (2006), 'Intrinsic statistics on Riemannian manifolds: Basic tools for geometric measurements'

[10]: Arvanitidis et al. (2016), 'A locally adaptive normal distribution'

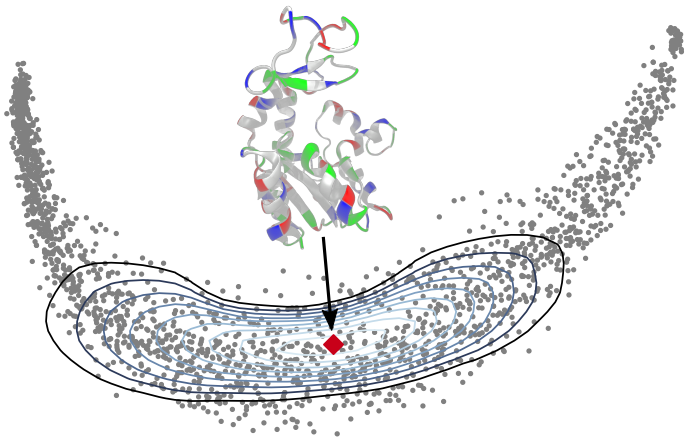


Figure 6.2: A LAND on a protein trajectory manifold of the enzyme adenylate kinase (ADK) reduced to two dimensions using linear principal component analysis (PCA). Each datum represents a conformation of the protein, i.e., a spatial arrangement of its atoms. The conformation corresponding to the LAND mean (◆) is visualized. The ADK dataset will be used throughout this chapter to illustrate particular aspects of the suggested manifold approach.

are interested in techniques to improve the efficiency of the numerical integration scheme on manifolds. Integrals on Riemannian manifolds, and in particular as they appear in the LAND model, satisfy properties that make Bayesian quadrature an excellent choice for the integration task(s):

- Knowledge about the structure of the integrals in curved spaces allows to specify an appropriate prior. In the case of a normalization constant, a warped BQ model is appropriate to encode positivity of the integrand in the model (*cf.* Section 2.5).
- The LAND optimization loop requires repeated evaluation of the normalization constant, and a small speed-up in the computation of a single normalization constant induces tremendous savings in the overall LAND procedure.
- Each individual integration problem on the manifold is itself computationally challenging, as it requires solving the geodesic equations, a system of second-order ordinary differential equations. In other words, evaluation of the integrand is costly and benefits from the sample-efficiency of BQ.
- The manifold structure calls for a tailored acquisition function that aggregates evaluations along geodesics by considering informative *directions* instead of *locations* on the manifold.
- The repeated evaluation of similar integrals in subsequent iterations of the LAND optimization scheme enables information transfer from previous iterations.

The uptake of Riemannian methods in machine learning is principally hindered by prohibitive computational costs. We here address a key aspect of this bottleneck by improving the efficiency of integration on data manifolds. Although integrals on Riemannian manifolds are generally amenable to treatment by BQ, the focus of this project are manifolds that are constructed from data. For studying integration on manifolds, the LAND serves as prime example to demonstrate the effectiveness of BQ on said manifolds. This chapter first establishes the relevant background on Riemannian manifolds needed to construct a custom BQ method. Section 6.2 provides a more illustrative than rigorous introduction into differential geometry, Section 6.2.1 details how to construct manifolds from data, and the LAND model is introduced in Section 6.3. The remainder

of the chapter deals with the development of the custom BQ method and its evaluation. The proposed BQ method achieves speedups by factors of up to 20 on synthetic and real-world data manifolds.

6.2 Riemannian geometry

A *manifold* $\mathcal{M}$ of dimension D is a topological space which does not carry a global vector space structure. As opposed to the familiar $\mathbb{R}^D$, a manifold lacks the possibility of adding or scaling vectors globally. Instead, an *atlas* is used to cover the manifold in *charts*, which only locally give a Euclidean view of the manifold. If transition maps between overlapping charts are smooth, we call $\mathcal{M}$ a *smooth manifold*, which provides the means for doing calculus.

In our applied setting, we view $\mathbb{R}^D$ as a *smooth manifold* $\mathcal{M}$ with a changed notion of distance and volume measurement as compared to the Euclidean case. This view arises from the assumption that data that lives in (the set) $\mathbb{R}^D$ have a general underlying nonlinear structure within $\mathbb{R}^D$, which is captured by the manifold $\mathcal{M}$ (see Figure 6.3). Here we are *not* concerned with lower-dimensional embedded manifolds that data could lie on, or manifolds with structure known a priori, e.g., spheres and tori, which are thus excluded from the discussion.

When $\mathcal{M}$ imposes geometric structure on $\mathbb{R}^D$, the corresponding *tangent space* $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ at a point $\boldsymbol{\mu} \in \mathcal{M}$ is again $\mathbb{R}^D$, but centered at $\boldsymbol{\mu}$. The tangent space is a vector space that allows to represent points on the manifold as tangent vectors $\boldsymbol{v} \in \mathbb{R}^D$. Pictorially, a vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ is tangential to some curve passing through $\boldsymbol{\mu}$. As a bundle, these vectors give a linearized view on the manifold with respect to a base point $\boldsymbol{\mu}$.

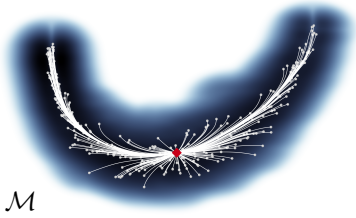


Figure 6.3: A protein trajectory manifold. A subset of the geodesics is shown with respect to a fixed point $\boldsymbol{\mu}$ (♦). The background is colored according to the volume element $\sqrt{|\boldsymbol{M}|}$ (Section 6.2.1) on a log scale. We omit a colorbar, since the values are not easily interpreted. Darker color indicates regions with small metric, to which shortest paths are attracted; white areas correspond to a large volume element that would drastically increase the lengths of any passing path.

A *Riemannian metric* is a positive definite matrix $\boldsymbol{M} : \mathbb{R}^D \to \mathbb{R}_+^{D\times D}$ that varies smoothly across the manifold. Therefore, we can define a local inner product between tangent vectors $\boldsymbol{v}, \boldsymbol{w} \in \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ as $\langle \boldsymbol{v}, \boldsymbol{w} \rangle_{\boldsymbol{\mu}} = \langle \boldsymbol{v}, \boldsymbol{M}(\boldsymbol{\mu})\boldsymbol{w} \rangle$, where $\langle \cdot, \cdot \rangle$ is the Euclidean inner product. This inner product makes the smooth manifold a *Riemannian manifold* [40, 158].

A Riemannian metric locally scales the infinitesimal distances and volume. Consider a curve $\gamma : [0, 1] \to \mathcal{M}$ with $\gamma(0) = \boldsymbol{\mu}$ and $\gamma(1) = \boldsymbol{x}$. The length of this curve on the Riemannian manifold $\mathcal{M}$ is computed as

$$L(\gamma) = \int_0^1 \sqrt{\langle \dot{\gamma}(t), \boldsymbol{M}(\gamma(t))\dot{\gamma}(t) \rangle}\, \mathrm{d}t,$$

where $\dot{\gamma}(t) = \frac{\mathrm{d}}{\mathrm{d}t}\gamma(t) \in \mathcal{T}_{\gamma(t)}\mathcal{M}$ is the velocity of the curve. The $\gamma_\star$ that minimizes this functional is the shortest path between the points. To overcome the invariance of L under reparameterization of γ, shortest paths can equivalently be defined as minimizers of the energy functional

$$E(\gamma) = \frac{1}{2}\int_0^1 \langle \dot{\gamma}(t), \boldsymbol{M}(\gamma(t))\dot{\gamma}(t) \rangle\, \mathrm{d}t.$$

In physics, the argument $\mathscr{L} := \langle \dot{\gamma}(t), \boldsymbol{M}(\gamma(t))\dot{\gamma}(t) \rangle$ of the integral is known as *Lagrangian*. The shortest path between two points $\boldsymbol{\mu} \in \mathcal{M}$ and $\boldsymbol{x} \in \mathcal{M}$ is obtained by solving the Euler-Lagrange equations, which are a system of 2nd order nonlinear ODEs. Given the boundary conditions $\gamma(0) = \boldsymbol{\mu}$ and $\gamma(1) = \boldsymbol{x}$, this set of ODEs known as *geodesic equations* pose a boundary value problem (BVP).[1] The resulting length-minimizing curve is known as *geodesic*.

[40]: Carmo (1992), *Riemannian geometry*
[158]: Lee (2018), *Introduction to Riemannian manifolds*

1: The geodesic equations are provided in Section C.1.1.

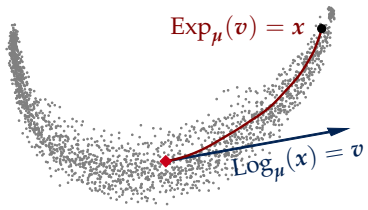


Figure 6.4: Illustration of the logarithmic and exponential map on the protein trajectory dataset.

To perform computations on $\mathcal{M}$ we introduce two operators. The first is the *logarithmic map* $\mathrm{Log}_{\boldsymbol{\mu}}(\cdot) : \mathcal{M} \to \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$

$$\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}) = \boldsymbol{v}$$

which represents a point $\boldsymbol{x} \in \mathcal{M}$ as a tangent vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$. The logarithmic map finds the initial velocity of the geodesic that reaches $\boldsymbol{x}$ at $t = 1$ with starting point $\boldsymbol{\mu}$. The inverse operator is the *exponential map* $\mathrm{Exp}_{\boldsymbol{\mu}}(\cdot) : \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M} \to \mathcal{M}$

$$\mathrm{Exp}_{\boldsymbol{\mu}}(t \cdot \boldsymbol{v}) = \boldsymbol{\gamma}(t)$$

that generates a unique geodesic starting out at $\boldsymbol{\gamma}(0) = \boldsymbol{\mu}$ with a given initial velocity $\dot{\boldsymbol{\gamma}}(0) = \boldsymbol{v} \in \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$. This geodesic terminates at $\boldsymbol{\gamma}(1) = \boldsymbol{x}$. The exponential map therefore takes tangent vectors $\boldsymbol{v}$ and maps them to points on the manifold $\boldsymbol{x} \in \mathcal{M}$. The two operations are illustrated on the ADK dataset in Figure 6.4. Note that $\mathrm{Log}_{\boldsymbol{\mu}}(\mathrm{Exp}_{\boldsymbol{\mu}}(\boldsymbol{v})) = \boldsymbol{v}$, and also, $\|\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\|_2 = \|\boldsymbol{v}\|_2 = L(\boldsymbol{\gamma})$. Computationally, the logarithmic map amounts to solving a BVP, whereas the exponential map corresponds to an initial value problem (IVP). For general data manifolds, analytic solutions of the geodesic equations do not exist, so we rely on specialized approximate numerical solvers for the BVPs; however, finding shortest paths still remains a computationally expensive problem [111, 12]. In contrast, the exponential map as an IVP is an easier problem and solutions are significantly more efficient. We illustrate our applied manifold setting in Figure 6.3, where we show geodesics between $\boldsymbol{\mu}$ and other data points.

[111]: Hennig and Hauberg (2014), 'Probabilistic solutions to differential equations and their application to Riemannian statistics'

[12]: Arvanitidis et al. (2019), 'Fast and robust shortest paths on manifolds learned from data'

6.2.1 Constructing Riemannian manifolds from data

The Riemannian volume element or measure $\mathrm{d}\mathcal{M}(\boldsymbol{x}) = \sqrt{|\boldsymbol{M}(\boldsymbol{x})|}\,\mathrm{d}\boldsymbol{x}$ represents the distorted infinitesimal standard Lebesgue measure $\mathrm{d}\boldsymbol{x}$. To capture the geometry inherent to the data, a meaningful metric should be small near the data and increases as we move away from them. Intuitively, such a metric behavior pulls shortest paths near the data, because this is where the volume element is smaller (*cf.* Figure 6.3).

There are broadly two unsupervised approaches to learn such an adaptive metric from data. Given a dataset $\boldsymbol{x}_{1:N}$ of N points in $\mathbb{R}^D$, Arvanitidis et al. [10] proposed a nonparametric metric to model nonlinear data trends as the inverse of a local diagonal covariance matrix with entries

[10]: Arvanitidis et al. (2016), 'A locally adaptive normal distribution'

$$M_{dd}(\boldsymbol{x}) = \left(\sum_{n=1}^{N} w_n(\boldsymbol{x})(x_{nd} - x_d)^2 + \rho \right)^{-1}, \tag{6.1}$$

where the weights w_n are obtained from an isotropic Gaussian kernel $w_n(\boldsymbol{x}) = \exp\left(-\frac{||\boldsymbol{x}_n - \boldsymbol{x}||^2}{2\sigma^2}\right)$. The lengthscale σ determines the curvature of the manifold, i.e., how fast the metric changes. The hyperparameter $\rho > 0$ controls the value of the metric components that is reached far from the data, so the measure there is $\sqrt{|\boldsymbol{M}|} = \rho^{-\frac{D}{2}}$. Typically, ρ is set to a small scalar to encourage geodesics to follow the data trend. However, this metric does not scale to higher dimensions due to the curse of dimensionality [30, Chapter 1.4].

[30]: Bishop (2006), 'Pattern recognition and machine learning'

Another approach relies on generative models to capture the geometry of high-dimensional data in a low-dimensional latent space [244, 11]. Let a dataset $\boldsymbol{y}_{1:N} \in \mathbb{R}^{D'}$ with latent representation $\boldsymbol{x}_{1:N} \in \mathbb{R}^{D}$ and $D' > D$, such that $\boldsymbol{y}_n \approx \boldsymbol{\eta}(\boldsymbol{x}_n)$ where $\boldsymbol{\eta}$ is a stochastic function with Jacobian $\boldsymbol{J}_{\boldsymbol{\eta}}(\boldsymbol{x}) \in \mathbb{R}^{D' \times D}$. Then, the *pullback metric* $\boldsymbol{M}(\boldsymbol{x}) = \mathbb{E}[\boldsymbol{J}_{\boldsymbol{\eta}}^\top(\boldsymbol{x})\boldsymbol{J}_{\boldsymbol{\eta}}(\boldsymbol{x})]$ is naturally induced in $\mathbb{R}^D$, which enables the computation of lengths that respect the geometry of the data manifold in $\mathbb{R}^{D'}$. Even though this metric reduces the dimensionality of the problem and can be learned directly from the data by learning $\boldsymbol{\eta}$, it is computationally expensive to use due to the Jacobian.

[244]: Tosi et al. (2014), 'Metrics for probabilistic geometries'

[11]: Arvanitidis et al. (2018), 'Latent space oddity: On the curvature of deep generative models'

To mitigate this shortcoming, we propose a surrogate Riemannian metric. Consider a variational autoencoder (VAE) [143, 213] with

$$q_\phi(\boldsymbol{x}|\boldsymbol{y}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_\phi(\boldsymbol{y}), \mathrm{diag}(\boldsymbol{\sigma}_\phi^2(\boldsymbol{y}))) \qquad \text{(encoder)}$$
$$p_\theta(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_\theta(\boldsymbol{x}), \mathrm{diag}(\boldsymbol{\sigma}_\theta^2(\boldsymbol{x}))) \qquad \text{(decoder)}$$

[143]: Kingma and Welling (2014), 'Auto-encoding variational Bayes'

[213]: Rezende et al. (2014), 'Stochastic backpropagation and approximate inference in deep generative models'

and prior $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{0}, \boldsymbol{I}_D)$, with deep neural networks as the functions that parametrize the distributions. Then, the aggregated posterior is

$$q_\phi(\boldsymbol{x}) = \int_{\mathbb{R}^{D'}} q_\phi(\boldsymbol{x} \mid \boldsymbol{y}) p(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \approx \frac{1}{N} \sum_{n=1}^{N} q_\phi(\boldsymbol{x} \mid \boldsymbol{y}_n),$$

where the integral is approximated from the training data. This is a Gaussian mixture model that assigns non-zero density only near the latent codes of the data. Thus, motivated by Arvanitidis et al. [13] we define a diagonal Riemannian metric in the latent space as

[13]: Arvanitidis et al. (2021), 'Geometrically enriched latent spaces'

$$\boldsymbol{M}(\boldsymbol{x}) = (q_\phi(\boldsymbol{x}) + \rho)^{-\frac{2}{D}} \cdot \boldsymbol{I}_D.$$

This metric fulfills the desideratum of modeling the local behavior of the data in the latent space, and it is more efficient than the pullback metric. The variance $\boldsymbol{\sigma}_\phi^2(\cdot)$ of the components is typically small, so the metric adapts well to the data, which, however, may result in high curvature.

6.3 Gaussians on Riemannian manifolds

The *Riemannian normal distribution* has been derived by Pennec [200] as the maximum entropy distribution on a manifold $\mathcal{M}$, given a mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The density[2] on $\mathcal{M}$ is

[200]: Pennec (2006), 'Intrinsic statistics on Riemannian manifolds: Basic tools for geometric measurements'

$$p(\boldsymbol{x}) = \frac{1}{\mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\Sigma})} \exp\left(-\frac{1}{2}\left\langle \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}), \boldsymbol{\Sigma}^{-1}\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\right\rangle\right).$$

2: The covariance $\boldsymbol{\Sigma}_{\mathcal{M}}$ on $\mathcal{M}$ is not equal to $\boldsymbol{\Gamma}_{\mathcal{M}}^{-1}$, but the covariance on $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ is $\boldsymbol{\Sigma} = \boldsymbol{\Gamma}^{-1}$. Throughout the paper we take the second perspective. For additional technical details see Section C.1.2.

It is reminiscent of the familiar Euclidean density, but with a Mahalanobis distance based on the nonlinear logarithmic maps. Analytic solutions for the normalization constant $\mathcal{C}$ can be given only for certain manifolds that are known a priori, like the sphere, since this requires analytic solutions for the logarithmic and exponential maps. Hence, numerical integration is indispensable when defining densities on arbitrary Riemannian manifolds, such as the manifolds constructed from data in Section 6.2.1.

In our data-driven setting, once the metric is constructed, the parameters of such a geometry-aware normal distribution can also be learned from data. This adaptive generalized Gaussian model is termed *locally adaptive*

normal distribution (LAND) [10]. For the LAND, we consider settings in which $\mathcal{M} = \mathbb{R}^D$ and $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M} = \mathbb{R}^D$, so $\boldsymbol{\Sigma} \in \mathbb{R}_+^{D\times D}$.
The parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ of the distribution are founds using block coordinate descent scheme to maximize the likelihood of the data. We use gradient descent for this non-convex optimization problem and update the parameters in an alternating fashion. That is, we keep $\boldsymbol{\mu}$ fixed while optimizing $\boldsymbol{\Sigma}$ and vice versa, as detailed in Section C.2.

[10]: Arvanitidis et al. (2016), 'A locally adaptive normal distribution'

In analogy to a standard Gaussian mixture model, it is possible to construct a mixture of LANDS. Given a dataset $\boldsymbol{x}_{1:N}$ assumed to be i.i.d., the log-likelihood of the LAND mixture can be stated as [10]

$$\mathcal{L}\left(\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{1:K}\right) = \sum_{k=1}^{K}\sum_{n=1}^{N} r_{nk}\left[\frac{1}{2}\langle \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n), \boldsymbol{\Sigma}_k^{-1}\mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)\rangle + \log\left(\mathcal{C}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right) - \log(\pi_k)\right] \tag{6.2}$$

where π_k is the weight of the k^{th} component, $\sum_{k=1}^{K}\pi_k = 1$ and $r_{nk} = \frac{\pi_k p(x_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{l=1}^{K}\pi_l p(x_n|\boldsymbol{\mu}_l,\boldsymbol{\Sigma}_l)}$ is the responsibility of the k^{th} component for the n^{th} datum. As in the single component case, the maximum likelihood solution is obtained by alternating between gradient descent updates of $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$, and further cycling through the components k, as described in Algorithm C.1.
The maximum likelihood parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ require estimating the normalization constant $\mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. It acts as a regularizer that keeps $\boldsymbol{\mu}$ near the data manifold and penalizes an overestimated $\boldsymbol{\Sigma}$. In the LAND mixture model, $\mathcal{C}$ is essential to deriving correct weights.

The intractable normalization constant—the quantity of interest in this work—is an integral over the manifold

$$\mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \;=\; \int_{\mathcal{M}} \exp\left(-\frac{1}{2}\langle \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}), \boldsymbol{\Sigma}^{-1}\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\rangle\right) \mathrm{d}\mathcal{M}(\boldsymbol{x}), \tag{6.3}$$

but can be lifted to an easier integration problem on the tangent space

$$\mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \;=\; \sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}\int_{\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}} f_{\boldsymbol{\mu}}(\boldsymbol{v})\,\mathcal{N}(\boldsymbol{v}; \mathbf{0}, \boldsymbol{\Sigma})\,\mathrm{d}\boldsymbol{v}, \tag{6.4}$$

by introducing the tangent space view on the volume element

$$f_{\boldsymbol{\mu}} : \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M} \to \mathbb{R}_+ : \boldsymbol{v} \mapsto f_{\boldsymbol{\mu}}(\boldsymbol{v}) = \sqrt{|\boldsymbol{M}(\mathrm{Exp}_{\boldsymbol{\mu}}(\boldsymbol{v}))|}. \tag{6.5}$$

The $f_{\boldsymbol{\mu}}$ that corresponds to the metric constructed on the protein trajectory data Figure 6.3 is depicted in Figure 6.5. Instead of having to solve BVPS for the logarithmic maps (6.3), it is possible to integrate on the Euclidean tangent space (6.4). On the tangent space, evaluation of the integrand (6.5) requires solving considerably faster exponential maps, *i.e.*, IVPS instead of BVPS.

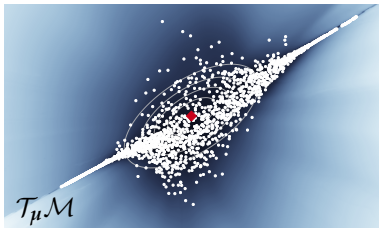


Figure 6.5: The function $f_{\boldsymbol{\mu}}$ on the tangent space of the protein trajectory manifold. The origin (♦) corresponds to the point $\boldsymbol{\mu}$ on the manifold from which the exponential maps are computed. Contours of the integration measure $\mathcal{N}(\boldsymbol{v}; \mathbf{0}, \boldsymbol{\Sigma})$ are in light gray. Logarithmic maps $\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}_n)$ of the data are scattered in white. The background is colored according to the volume element on a log scale. The color scale is quantitatively unrelated to Figure 6.3.

In the original algorithm presented by Arvanitidis et al. [10], the normalization constant is computed using a naïve Monte Carlo (MC) scheme as

$$\mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \simeq \frac{1}{S}\sum_{s=1}^{S} f_{\boldsymbol{\mu}}(\boldsymbol{v}_s), \qquad \boldsymbol{v}_s \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}).$$

Despite the relatively cheaper version of computing the integral via (6.4) instead of (6.3), the computation of S exponential maps for the

MC estimator is still a significant overhead. The MC estimator is ignorant about known structure of the integrand and requires a large number of samples to reach a good accuracy. We replace MC by BQ to drastically reduce the number of these costly evaluations needed to retain accuracy. Our foremost goal is to speed up numerical integration on data manifolds since exponential maps are, albeit faster than the BVPs, still relatively slow. The runtime of exponential maps depends on the employed metric (see Section 6.2.1) and on other factors such as curvature or curve length.

6.4 Bayesian quadrature on manifolds

The computational cost incurred by evaluating exponential maps motivates the use of BQ for computing the integral (6.4).[3] Consider the Gaussian on the tangent space the integration measure, *i.e.*,

$$\nu(\boldsymbol{v}) = \mathcal{N}(\boldsymbol{v}; \mathbf{0}, \boldsymbol{\Sigma}).$$

The integration is carried out on the tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ (which is isomorphic to $\mathbb{R}^D$) over tangent vectors $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$ and we can write (6.4) as

$$\mathcal{C} \propto \int_{\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}} f_{\boldsymbol{\mu}}(\boldsymbol{v}) \, \mathrm{d}\nu(\boldsymbol{v}).$$

with known proportionality constant. In the tangent space view, the integral is over a Euclidean space and thus, all the geometric effects are conveniently absorbed in the integrand $f_{\boldsymbol{\mu}}$ itself. A modification of BQ is therefore *not* necessary. Yet, the manifold setting offers advantages that we can tailor the BQ method to.

3: Other integrals arising on the manifold, *e.g.*, expectations, can similarly be solved with BQ. Due to the abundance of integrals for the normalization constant in the LAND procedure, we only focus on those.

6.4.1 Encoding positivity

The most evident adaptation is to employ warped BQ (*cf.* Section 2.5) to account for the positivity of the integrand. To this end, we introduce a random function $\mathsf{g} \sim \mathcal{GP}(m_{\mathsf{g}}, k_{\mathsf{g}})$ and consider the random process induced by the warping $\mathsf{f} = \frac{1}{2}\mathsf{g}^2 + \delta$ a surrogate for the integrand $f_{\boldsymbol{\mu}}$. $\delta > 0$ is a small additive constant here. Notice the slight difference to the square transform given in Section 2.5 as we employ the transformation as defined by Gunter et al. [102]. Approximation of f by a Gaussian process (GP) yields the moments stated in (2.27) to (2.29) that can be summarized as

$$\begin{aligned} m_{\mathsf{f}\,|\,\mathcal{D}}(\boldsymbol{v}) &= \delta + \frac{1}{2} m_{\mathsf{g}\,|\,\mathcal{D}}(\boldsymbol{v})^2 + \frac{\eta}{2} k_{\mathsf{g}\,|\,\mathcal{D}}(\boldsymbol{v}, \boldsymbol{v}), \\ k_{\mathsf{f}\,|\,\mathcal{D}}(\boldsymbol{v}, \boldsymbol{v}') &= \frac{\eta}{2} k_{\mathcal{D}}(\boldsymbol{v}, \boldsymbol{v}') + m_{\mathsf{g}\,|\,\mathcal{D}}(\boldsymbol{v}) k_{\mathsf{g}\,|\,\mathcal{D}}(\boldsymbol{v}, \boldsymbol{v}') m_{\mathsf{g}\,|\,\mathcal{D}}(\boldsymbol{v}'). \end{aligned}$$

where $\eta = 0$ for WSABI-L and $\eta = 1$ for WSABI-M. The 'data'—not to be confused with the dataset that the LAND is defined on—now contain the pairs of tangent vectors and integrand evaluations $\mathcal{D} = \{\boldsymbol{v}_n, f_{\boldsymbol{\mu}}(\boldsymbol{v}_n)\}_{n=1}^{N_{\text{BQ}}}$ where we define N_{BQ} here as the number of evaluations used for BQ.

In the present setting, WSABI offers three main advantages over vanilla BQ and MC: First, it encodes the prior knowledge that the integrand is positive everywhere. Second, for metrics learned from data, the volume element typically grows fast and takes on large values away from the data. This makes modeling g directly by a GP impractical, especially when the kernel encourages smoothness. The square transform alleviates this

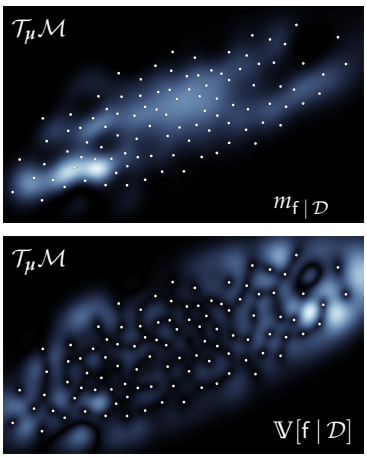


Figure 6.6: Posterior mean (*top*) over $f_{\boldsymbol{\mu}}(\boldsymbol{v})$ multiplied with the integration measure $\mathcal{N}(\boldsymbol{v}; \mathbf{0}, \boldsymbol{\Sigma})$ according to WSABI-L with points found sequentially using uncertainty sampling. High values have a brighter color.

[102]: Gunter et al. (2014), 'Sampling for inference in probabilistic models with fast Bayesian quadrature'

problem by reducing the dynamic range of f compared to that of g. The exponential transform, which would be an alternative warping to encode positivity, proved too extreme in reducing the metric's range. Finally, WSABI comes with a simple active learning scheme to select tangent vectors (*cf.* Section 3.3). The traditional WSABI objective is the posterior variance of the unwarped GP scaled with the squared integration measure (3.11)

$$\alpha^{\text{US}}(\boldsymbol{v}) = k_{\mathsf{f}\,|\,\mathcal{D}}(\boldsymbol{v}, \boldsymbol{v})\, \nu(\boldsymbol{v})^2,$$

that, when optimized sequentially, finds the modes of high posterior variance $\mathbb{V}[\mathsf{f}\,|\,\mathcal{D}] = k_{\mathsf{f}\,|\,\mathcal{D}}(\boldsymbol{v}, \boldsymbol{v})$ of the approximate GP under the integration measure. Figure 6.6 shows the posterior mean and variance found with uncertainty sampling using the WSABI-L approximation to the warped GP.

6.4.2 Active learning on the tangent space

The known structure of the computations performed to evaluate the integrand on the tangent space—the solution of exponential maps—is additional information that can be exploited towards a custom, although heuristic, active learning policy. The numerical solution of exponential maps yields intermediate steps along straight lines in the tangent space, or, equivalently points along a geodesic on the manifold. The magnitude of the initial velocity does not change the path of the geodesic, only the distance traveled. Define the unit vector $\hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{v}$ with norm $v = \|\boldsymbol{v}\|$ as the *direction* of an initial velocity of a curve. When solving $\mathrm{Exp}_{\boldsymbol{\mu}}(\boldsymbol{v}) = \mathrm{Exp}_{\boldsymbol{\mu}}(v \cdot \hat{\boldsymbol{v}})$ given $\hat{\boldsymbol{v}}$, solutions to the exponential map for initial velocities $\beta\hat{\boldsymbol{v}}$, $0 < \beta < v$ are also available from intermediate steps of the numerical solution of the IVP. Once the exponential maps are computed, evaluation of the integrand $g_{\boldsymbol{\mu}}$ is cheap. Hence, integrand evaluations are cheaply available along straight lines in the tangent space.

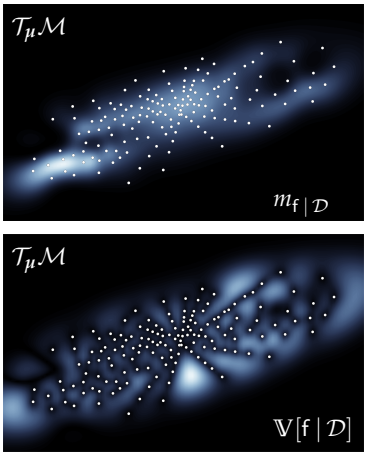


Figure 6.7: The posterior mean and variance with WSABI-L as in Figure 6.6, but here the points here have been found using the DCV acquisition function. Because they lie along straight lines in the tangent space, they lie along rays that originate from the current mean $\boldsymbol{\mu}$ (*cf.* Figure 6.9). Compared to WSABI-L with uncertainty sampling, there is more unexplored space, but this setting also required solving less exponential maps.

This observation motivates rethinking the scheme for sequential design to select good initial directions instead of fixed velocities for the exponential map. We propose to select these initial directions such that the cumulative variance along the direction on the tangent space is maximized. Along this line with large marginal variance, multiple points are then collected in a way that they fall on modes of the angularly constrained variance. The modified acquisition function that expresses the cumulative variance along a straight line, can be written as

$$\bar{\alpha}(\hat{\boldsymbol{v}}) = \int_0^\infty \alpha(\beta\hat{\boldsymbol{v}})\, \mathrm{d}\beta. \tag{6.6}$$

The new acquisition policy arises from optimizing $\bar{\alpha}$ for unit tangent vectors $\hat{\boldsymbol{v}}$. We call the acquisition function from (6.6) *directional cumulative variance (DCV)*. While it does have a closed-form solution (derived in Section C.3.1), that solution costs $\mathcal{O}(N_{\text{BQ}}^4)$ to evaluate in the number N_{BQ} of evaluations chosen for BQ.[4] We resort to numerical integration to compute the objective and its gradient (Section C.3). This is feasible because these are multiple univariate integrals that can efficiently be estimated from the same evaluations. Since $\hat{\boldsymbol{v}}$ is constrained to lie on the unit hypersphere, we employ a manifold optimization algorithm. Once an exponential map is computed, we use the standard WSABI objective

4: This is one of the remarkable integrals that do admit a closed-form solution. However, such a solution causes combinatorial entanglement of the data, making it more expensive to evaluate than to solve the integral numerically.

to sample multiple informative points along the straight line $v \cdot \hat{v}$. For simplicity, we use DCV only in conjunction with WSABI-L.

Optimizing this acquisition function is costly as it requires posterior mean predictions and predictive gradients of the GP inside the integration. Furthermore, confining observations to lie along straight lines implies that BQ may cover less space given a fixed number of function evaluations. Therefore, DCV will be useful in settings where exponential maps come at a high computational cost. Figure 6.7 illustrates the posterior mean and variance arising when nodes are acquired using the DCV acquisition. In contrast to Figure 6.6, DCV result in a less space-filling design, but also requires a lower computational budget. Figure 6.8 illustrates the geodesics chosen by DCV, and Figure 6.9 illustrates the same setting on the tangent space.

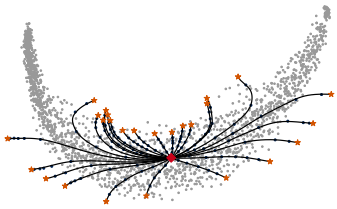


Figure 6.8: The geodesics chosen by the DCV design scheme on the manifold. The end positions ⋆ fall well off the data support. Points collected along the trajectories are indicated by •.

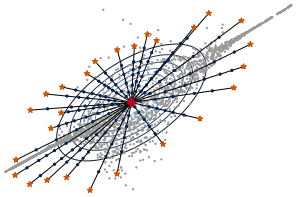


Figure 6.9: Figure 6.8 seen from the tangent space, showing the tangent vectors selected by DCV. Additionally, the contours of the Gaussian integration measure are plotted.

6.4.3 Transfer learning in the LAND loop

An additional benefit of BQ crystallizes in the setting in which integrals are estimated in the LAND model. The LAND optimization process requires the normalization constant (6.4) to be computed once per iteration. Due to its dependency on the LAND parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, this integral changes smoothly as the parameters are altered over iterations. Integrals in consecutive iterations are hence correlated and suggest the possibility of transferring information rather than recomputing normalization constants from scratch. Elaborate schemes as in [265] and Chapter 5 that explicitly model the covariance between integrals immediately come to mind.
However, the block coordinate descent algorithms that updates $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ in an alternating manner enables a simpler, more elegant way to recycle information. Reconsidering the integral (6.4), we make use of the fact that the integrand $f_{\boldsymbol{\mu}}$ (6.5) only depends on the LAND mean $\boldsymbol{\mu}$, but is independent on the covariance $\boldsymbol{\Sigma}$. The covariance merely enters the integration measure ν. Since the GP surrogate is only defined over $f_{\boldsymbol{\mu}}$ (and, importantly, *not* over ν), the integrand only changes in an iteration that updates $\boldsymbol{\mu}$. When only the covariance $\boldsymbol{\Sigma}$ is updated from one iteration to the next while the mean $\boldsymbol{\mu}$ remains fixed, $f_{\boldsymbol{\mu}}$ remains unaltered. This also means that the expensive bit, namely the exponential maps, do not change. BQ can thus reuse the observations from the previous iteration and only needs to collect a reduced number of new samples to account for the changed integration measure. Only updates of the covariance matrix call for a re-computation of exponential maps. This node reuse enables tremendous runtime savings and motivates an adaptation of the optimization scheme that runs multiple updates of the mean before the covariance is updated again.

[265]: Xi et al. (2018), 'Bayesian quadrature for multiple related integrals'

6.4.4 Choice of model

The known smoothness of the metric tensor makes the square exponential kernel (RBF) a suitable choice in most cases. However, for high-curvature manifolds, in particular in two dimensions, we found the Matérn-5/2 kernel to be numerically slightly more stable, so we use it throughout instead of the RBF. Even if chosen isotropic, the Matérn kernel does not

provide an analytic form for the kernel mean. We therefore use f as an emulator and compute the integrals (2.15) and (2.16) with exhaustive MC sampling. To compute the integral without loss of precision, we use $S = 30{,}000$ samples to estimate the integrals. The time overhead and the approximation error of this procedure are negligible in practice, since the GP emulator does not rely on expensive exponential maps.
We optimize the marginal likelihood of the GP with respect to the hyperparameters and use their final values to initialize the next iteration, since during the optimization the function changes smoothly from each step to the next. This information is not shared across the K components, but kept separately.
Depending on the employed Riemannian metric, we set the constant prior mean of f to the known volume element far from the data (Section 6.2.1). This amounts to the prior assumption that wherever there are no observations yet, the distance to the data is likely high.
Our implementation of BQ builds upon the `bayesquad` `Python` library [252] available at `https://github.com/OxfordML/bayesquad`.

[252]: Wagstaff et al. (2018), 'Batch selection for parallelisation of Bayesian quadrature'

6.5 Experiments

We test the methods (WSABI-L, WSABI-M, DCV) on both synthetic and real-world data manifolds. Our aim is to show that Bayesian quadrature is faster compared to the Monte Carlo baseline, yet retains high accuracy. The experiments focus on the LAND model to illustrate practical use cases of Riemannian statistics. Furthermore, the iterative optimization process yields a wide range of integration problems of varying difficulty. In total, our experiments comprise 43,920 BQ integrations. For different manifolds, we conduct two kinds of experiments:

Boxplot Experiments First, we fit the LAND model and record all integration problems arising during the optimization procedure. This allows us to compare the competitors on the whole problem, where BQ can benefit from node reuse. As the ground truth, we use extensive MC sampling ($S = 40{,}000$).[5] We fix the number of acquired samples for BQ and generate boxplots from the mean errors on the whole LAND fit for 16 independent runs (Figure 6.10). Due to the alternating update of LAND parameters during optimization, *either* the integrand *or* the integration measure changes over consecutive iterations. We let WSABI-L and WSABI-M actively collect 80 in the former and 10 samples additionally to the reused ones in the latter case; for DCV, we fix 18 and 2 exponential maps, respectively, and acquire 6 points on each straight line. Integration cost for BQ is thus highly variable over iterations. Allocating a fixed runtime would not be sensible as BQ benefits from collecting more information after updates to the mean, a time investment that is over-compensated in the more abundant and—due to node reuse—cheap covariance updates. We choose sample numbers so as to allow for sufficient exploration of the space with practical runtime. For MC, we allocate the runtime budget of the mean slowest BQ method on that particular problem in order to compare accuracy over runtime. Mean runtimes for single integrations, averaged on entire LAND fits, are shown in Figure 6.11 and mean exponential map runtimes, as computed by MC, are reported in Table 6.1.

Table 6.1: Mean exponential map runtime in milliseconds for the manifolds described further on, obtained by averaging over MC runtimes on the entire LAND fit.

CIRCLE	60
CIRCLE 5D	50
MNIST	238
ADK	68
CIRCLE 3D	32
CIRCLE 4D	45
CURLY	62
2-CIRCLES	36

5: Since obtaining a large number of exponential maps is computationally extremely expensive, we subsampled from this pool of ground truth samples when MC samples were required in the experiments, instead of running MC again. For example, for Figure 6.12, we calculated the mean MC runtime per sample from the ground truth pool of this particular problem and then subsampled as many samples as the given runtime limit affords. For the boxplot experiments, we averaged the MC runtimes over the whole LAND fit and always obtained the same number of samples per integration. Note that the MC runtime practically corresponds to the runtime of the exponential maps, since the overhead is minimal.

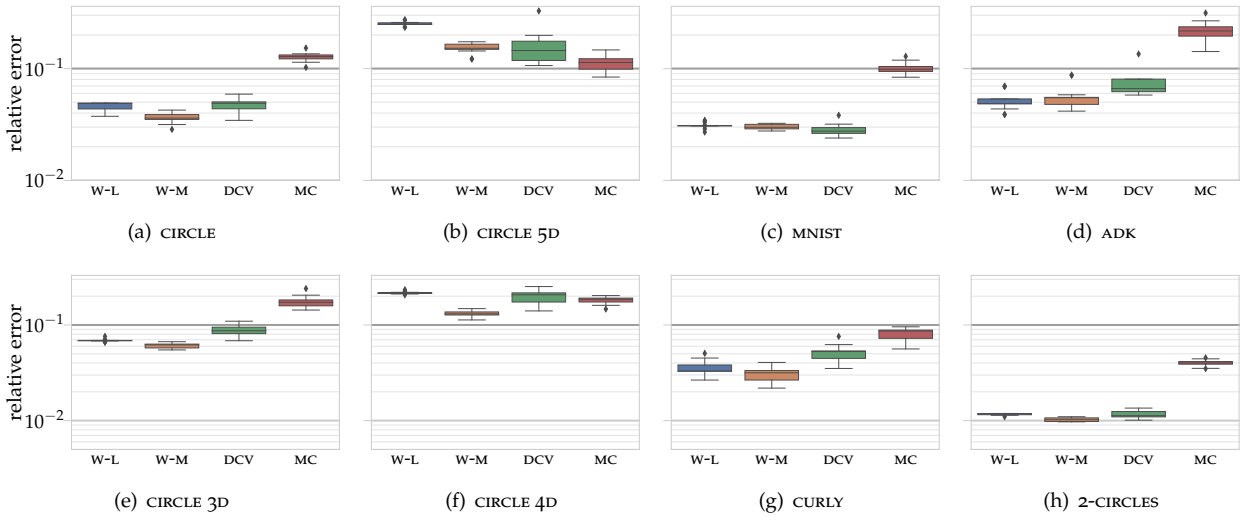


Figure 6.10: Boxplot error comparison (log scale, shared y-axis) of BQ and MC on whole LAND fit for different manifolds. For MC, we allocate the runtime of the mean slowest BQ method. Each box contains 16 independent runs.

These experiment were conducted on whole LAND fits, with 16 independent runs for each of the 3 BQ methods. From Table 6.3, we can easily calculate the total number of runs as $48 \cdot (67 + 39 + 40 + 34 + 105 + 36 + 33 + 111) = 22{,}320$.

Error vs. Runtime Experiments Secondly, we focus on the first integration problem of each LAND fit in detail and compare the convergence behavior of the different BQ methods and MC over wall clock runtime (Figure 6.12). We use the kernel metric (Section 6.2.1) when not otherwise mentioned. In the plot legends, we abbreviate WSABI-L/WSABI-M with W-L/W-M, respectively.

We evenly space 30 runtime limits between 5 and 65 seconds using `np.linspace(5., 65., 30)`. For each of these runtime limits, we let each BQ method run 30 times. BQ will stop collecting more samples as soon as the runtime limit is reached. After this, however, it will take some more time to finalize, as an ongoing computation is not interrupted. We then record the actually resulting runtimes and average over the 30 runs. These averages are then used for the x-axes of the plots, whereas the mean relative error is on the y-axes. In total, each BQ method thus has 900 runs on each problem. The 8 plots contain $3 \cdot 900 \cdot 8 = 21{,}600$ runs. Together with the boxplot experiments, we obtain $21{,}600 + 22, = 43{,}920$ BQ runs, that is, 14,640 for each of the 3 methods.

In Figure 6.12(c), we removed 4 extreme DCV outliers, where seemingly the GP failed. This amounts to $\frac{4}{21{,}600} = 0.01852\%$ of the BQ runs in the 8 plots.

All experiments were run in a cloud setting on 8 virtual CPUs. We restricted the core usage of BLAS linear algebra subroutines to a single core, so as not to create interference between multiple processes.

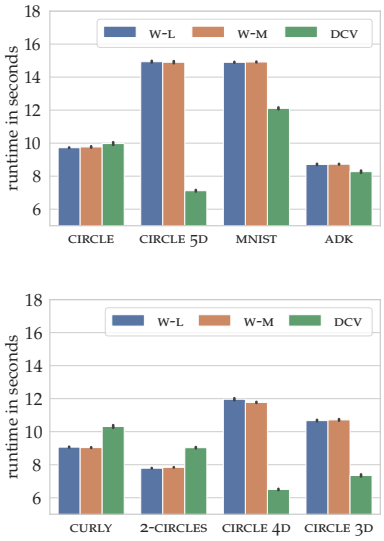


Figure 6.11: Mean runtime required by the BQ methods for the integration task of Figure 6.10, averaged over 16 runs for the LAND fit on the respective manifold. Errorbars indicate 95% confidence intervals over these runs.

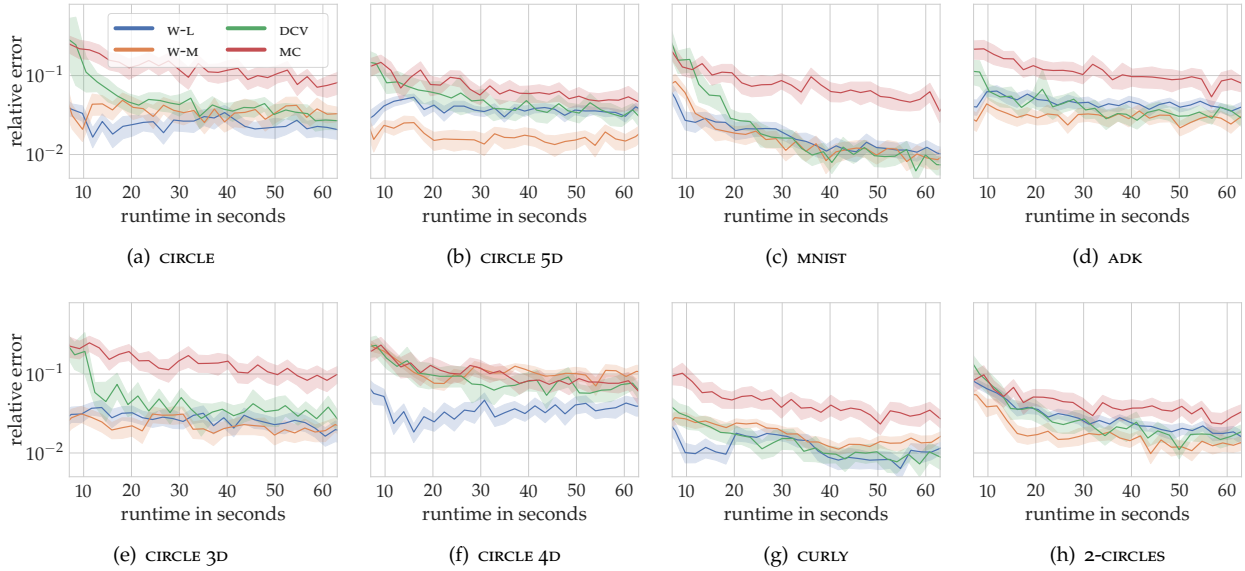


Figure 6.12: Comparison of BQ and MC errors against runtime (vertical log scale, shared legend and axes) for different manifolds, on the first integration problem of the respective LAND fit. Shaded regions indicate 95% confidence intervals over 30 independent runs.

6.5.1 Synthetic experiments

Toy Data We generated three toy data sets (CIRCLE, CURLY, and 2-CIRCLES) and fitted the LAND model with pre-determined component numbers. Figure 6.13 displays the circle manifold with 1000 data points, for which we compare the resulting LAND fit to the Euclidean Gaussian mixture model (GMM) in Figure 6.13. The other two synthetic manifolds are displayed in Figure 6.14.

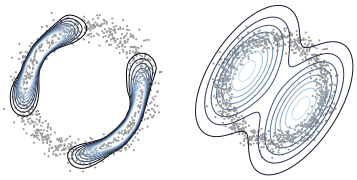


Figure 6.13: Comparison of a two-component LAND vs. a Gaussian mixture model on the synthetic CIRCLE data.

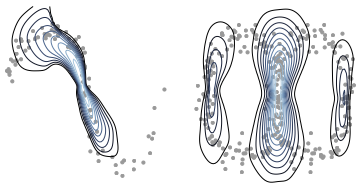


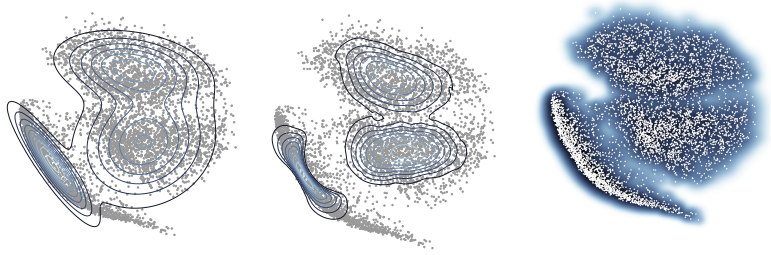
Figure 6.14: (a) A LAND fit on the CURLY manifold; (b) a 3-component LAND mixture fit on the 2-CIRCLES manifold.

Higher-dimensional Toy Data With increasing number of dimensions, new challenges for metric learning and geodesic solvers appear. With the simple kernel metric, almost all of the volume will be far from the data as the dimension increases, a phenomenon which we observe already in relatively low dimensions. Such metric behavior can lead to pathological integration problems, as the integrand may then become almost constant. In this experiment, we embed the circle toy data in higher dimensions by sampling random orthonormal matrices. After projecting the data, we add Gaussian noise $\epsilon_i \sim \mathcal{N}(0, 0.01)$ and standardize. The CIRCLE dataset has been considered in $d = \{3, 4, 5\}$.

6.5.2 Real-world experiments

MNIST We sampled 5,504 random data points from the first three digits of MNIST [157], which were preprocessed by normalizing them feature-wise to $[-1, +1]$ using `sklearn.preprocessing.MinMaxScaler`. We trained a simple variational autoencoder (VAE) to embed the 784 dimensional input in a latent space of dimension 2. The architecture uses separate encoders $\boldsymbol{\mu}_\phi, \boldsymbol{\sigma}_\phi$ and decoders $\boldsymbol{\mu}_\theta, \boldsymbol{\sigma}_\theta$. The VAE setup we used summarized in Table 6.2.

[157]: LeCun et al. (1998), 'Gradient-based learning applied to document recognition'



(a) GMM (b) LAND (c) metric

Figure 6.15: Comparison of a two-component LAND vs. a GMM on three-digit MNIST. (c) shows the surrogate metric on a logarithmic scale.

Encoder/Decoder	Layer 1	Layer 2	Layer 3
μ_ϕ	128 (tanh)	64 (tanh)	2 (linear)
σ_ϕ	128 (tanh)	64 (tanh)	2 (softplus)
μ_θ	64 (linear)	128 (linear)	784 (linear)
σ_θ	64 (linear)	128 (linear)	784 (softplus)

Table 6.2: VAE settings for dimension reduction in MNIST.

We trained the network for 200 epochs using ADAM with a learning rate of 10^{-3}. The resulting latent codes were used to construct the *Aggregated Posterior Metric*, with $\rho = 0.001$, such that the measure far from the data is 1000. The small variances cause high curvature, which makes the integration tasks challenging and geodesic computations slow. To fit the LAND, we used 250 subsampled points to lower the amount of time spent on BVPs. In contrast, the GMM was fitted on the whole 5,504 points. Figure 6.15 shows this training data.

The LAND is able to distinguish the three clusters more clearly than a Euclidean Gaussian mixture model (GMM), see Figure 6.15. The LAND favors regions of higher density, where the VAE has more training data. In this experiment, the gain in speed of BQ is even more pronounced, since exponential maps are slow due to high curvature. MC with 1000 samples achieves 2.78% mean error on the whole LAND fit with a total runtime of 6 hours and 56 minutes, whereas DCV (18/2 exponential maps) achieves 2.84% error within 21 minutes; a speedup by a factor of ≈ 20.

Molecular Dynamics In molecular dynamics, biophysical systems are simulated on the atomic level. This approach is useful to understand the conformational changes of a protein, i.e., the structural changes it undergoes. A Riemannian model is appropriate in this setting, because not all atom coordinates represent physically realistic conformations. For instance, a protein clearly does not self-intersect. Adapting locally to the data by space distortion is thus critical for modeling. More specifically, the LAND model is relevant because clustering conformations and finding representative states are of scientific interest (see *e.g.*, [197, 263, 235, 246]). The LAND can visualize the conformational landscape and generate realistic samples. Plausible transitions between conformations may be conceived of as geodesics under the Riemannian metric.

We obtained multiple trajectories of the closed to open transition of the enzyme adenylate kinase (ADK) [227].[6] Each observation consists of the Cartesian (x, y, z) coordinates for each of the 3,341 atoms, yielding a 10,023 dimensional vector. As is common in the field, we used PCA to extract the *essential dynamics* [7], which clearly exhibit manifold structure

[197]: Papaleo et al. (2009), 'Free-energy landscape, principal component analysis, and structural clustering to identify representative conformations from molecular dynamics simulations: The Myoglobin case'

[263]: Wolf and Kirschner (2013), 'Principal component and clustering analysis on molecular dynamics data of the ribosomal L11·23S subdomain'

[235]: Spellmon et al. (2015), 'Molecular dynamics simulation reveals correlated inter-lobe motion in protein Lysine Methyltransferase SMYD2'

[246]: Tribello and Gasparotto (2019), 'Using dimensionality reduction to analyze protein trajectories'

[227]: Seyler et al. (2015), 'Path similarity analysis: A method for quantifying macromolecular pathways'

6: We obtained protein trajectory data of adenylate kinase from `https://www.mdanalysis.org/MDAnalysisData/adk_transitions.html#adk-dims-transitions-ensemble-dataset`. We use the DIMS variant, a dataset which comprises 200 trajectories and select a subset consisting of the trajectories $160 - 200$, which contain in total 2,038 data points.

[7]: Amadei et al. (1993), 'Essential dynamics of proteins'

(Figure 6.2). The first two eigenvectors already explain 65% of the total variance and suffice to capture the transition motion.

We model the ADK manifold with the kernel metric (6.1) with $\sigma = 0.035$ and large measure far from the data ($\rho = 10^{-5}$) to account for high curvature and the knowledge that realistic trajectories lie closely together. This makes for a challenging integration problem, since most mass is near the data boundary due to extreme metric values.
A single-component LAND yields a representative state for the transition between the closed and open conformation. Whereas the Euclidean mean falls outside the data manifold, the LAND mean is reasonably situated. The eigenvectors of the covariance matrix demonstrate that the LAND captures the intrinsic dimensions of the data manifold (Figure 6.17) and that the mean interpolates between the closed and open state (Figure 6.16).
The purpose of this experiment is to highlight the applicability of Riemannian statistics to molecular dynamics and sketch potential experiments, which are then for domain experts to design.

6.5.3 Details on experiments

In Table 6.3, we report the relevant hyperparameters for the metrics (σ, ρ), which were used to construct the manifolds, and those optimization parameters which are not equal across all problems.

6.5.4 Interpretation

We find that BQ consistently outperforms MC in terms of speed. Even on high-curvature manifolds with volume elements spanning multiple orders of magnitudes, such as MNIST and ADK, the GP succeeds in approximating the integrand well. Among the different BQ candidates, we cannot discern a clear winner, since their performance depends on the specific problem geometry and exponential map runtimes. DCV performs especially well when geodesic computations are costly, such as for MNIST. We note that geodesic solvers and metric learning are subject to new challenges in higher dimensions, which merit further research effort.

[119]: Humphrey et al. (1996), 'VMD – Visual Molecular Dynamics'

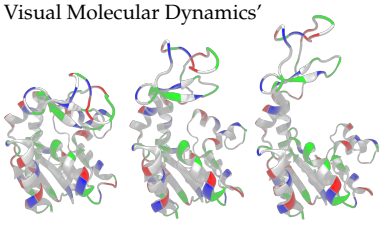


Figure 6.16: The protein ADK in closed, LAND mean and open state. To visualize spatial protein structure, we used the software VMD [119] with the 'new cartoon' representation, colored according to 'residue type'.

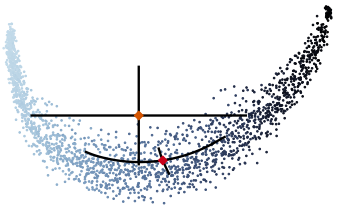


Figure 6.17: Comparison of the Euclidean Gaussian vs. LAND mean and eigenvectors on ADK data. Data is colored according to the *radius of gyration*, a measure indicating how 'open' the protein is, providing a visual argument for the manifold hypothesis.

Table 6.3: Manifold and LAND optimization hyperparameters and resulting number of integrations.

Parameter	CIRCLE	CIRCLE 3D	CIRCLE 4D	CIRCLE 5D	MNIST	ADK	CURLY	2-CIRCLES
σ	0.1	0.25	0.25	0.25	-	0.035	0.2	0.15
ρ	0.001	0.01	0.0316	0.063	0.001	0.00001	0.01	0.01
K	2	2	2	2	3	1	1	3
$t_{\max}$	7	4	4	4	7	7	7	7
α_μ^1	0.3	0.3	0.3	0.3	0.3	0.2	0.3	0.3
ϵ_{∇_μ}	0.01	0.01	0.01	0.01	0.015	0.01	0.01	0.01
# integrations	67	39	40	34	105	36	33	111

6.6 Conclusion and discussion

Riemannian statistics is the appropriate framework to model real data with nonlinear geometry. Yet, its wide adoption is hampered by the prohibitive cost of numerical computations required to learn geometry from data and operate on manifolds. In this project, we have demonstrated on the example of numerical integration the great potential of probabilistic numerical methods (PNMs) to reduce this computational burden. The deliberate choice of informative computations in a Bayesian framework saves unnecessary operations on the manifold. Bayesian quadrature outperforms Monte Carlo on Riemannian manifolds over a large number of integration problems owing to its increased sample efficiency. Information transfer and a novel acquisition scheme both help to further reduce the number of expensive geodesic evaluations needed to estimate the integral.

Outlook Numerical integration is just one of multiple numerical tasks in the context of statistics on Riemannian manifolds where PNM suggest promising improvements. The key operations on data manifolds are geodesic computations, i.e., solutions of ordinary differential equations. Geodesics have been viewed through the PN lens, e.g., by Hennig and Hauberg [111], but still offer a margin for increasing the performance of statistical models such as the considered LAND.

[111]: Hennig and Hauberg (2014), 'Probabilistic solutions to differential equations and their application to Riemannian statistics'

Once multiple PNM are established for Riemannian statistics, the future avenue directs towards having them operate in a concerted fashion. As data-driven Riemannian models rely on complex computation pipelines with multiple sources of epistemic and aleatory uncertainty, their robustness and efficiency can benefit from modeling and propagating uncertainty through the computations.

All in all, we believe the coalition of *geometry*- and *uncertainty*-aware methods to be a fruitful endeavor, as these approaches are united by their common intention to respect structure in data and computation that is otherwise often neglected.

Truncated Normal Distributions

Inference With Gaussians Under Linear Domain Constraints

7

Multivariate Gaussian *densities* are omnipresent in statistics and machine learning. Yet, Gaussian *probabilities* are hard to compute—they require solving an integral over a constrained Gaussian volume—owing to the intractability of the multivariate version of the Gaussian cumulative distribution function (CDF). Inference in models that contain truncated multivariate normal distributions thus inevitably relies on approximations.

This chapter addresses two aspects of approximate inference in such a setting: *sampling* from multivariate normal distributions, and *estimating* the mass of said domain under the Gaussian measure. This encourages a natural split into the following two parts,

Sampling Simulation from a truncated Gaussian can be achieved through an adapted version of elliptical slice sampling (ESS) which we call LIN-ESS. It allows for *rejection-free* sampling from the linearly constrained domain. Its effectiveness is not compromised even if the probability mass of the domain is very small.

Integration Based on the LIN-ESS algorithm, we introduce an efficient integration scheme for truncated Gaussians. It relies on a sequence of nested domains to decompose the integral into multiple, easier-to-solve, conditional probabilities. The method embeds the Holmes-Diaconis-Ross algorithm [67, 216, 150] into a pipeline adapted for computing the mass of linearly restricted Gaussians.

[67]: Diaconis and Holmes (1995), 'Three examples of Monte-Carlo Markov chains: At the interface between statistical computing, computer science, and statistical mechanics'

[216]: Ross (2012), *Simulation*

[150]: Kroese et al. (2013), *Handbook of Monte Carlo methods*

The contents of this chapter have been published as

> A. Gessner, O. Kanjilal, and P. Hennig. 'Integrals over Gaussians under linear domain constraints'. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by S. Chiappa and R. Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020

and coincide with the article in large parts. A `Python` implementation of the introduced methods are publicly available at `https://github.com/alpiges/LinConGauss`.

7.1 Problem setting

Consider a standard normal measure on $\mathbb{R}^D$ and M linear functions $f_m : \mathbb{R}^D \to \mathbb{R} : \boldsymbol{x} \mapsto \boldsymbol{a}_m^\top \boldsymbol{x} + b_m$, where $\boldsymbol{a}_m \in \mathbb{R}^D$ and $b_m \in \mathbb{R}$. Let $\mathcal{L} \subset \mathbb{R}^D$ denote the linearly restricted domain of interest as the intersection where all the M constraints exceed zero,

$$\mathcal{L} = \left\{ \boldsymbol{x} : \boldsymbol{x} \in \bigcap_{m=1}^{M} \left\{ \boldsymbol{a}_m^\top \boldsymbol{x} + b_m > 0 \right\} \right\}. \tag{7.1}$$

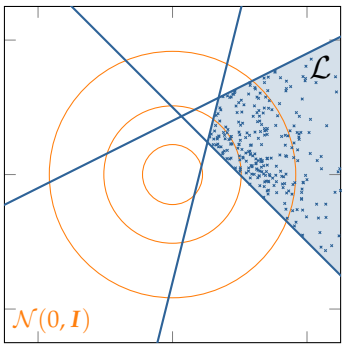


Figure 7.1: Qualitative illustration of the setup: The goal is to draw samples from the shaded domain $\mathcal{L}$ and estimate its mass under a standard normal distribution.

The probability mass that lies within this domain $\mathcal{L}$ can be written as

$$Z = P(\boldsymbol{x} \in \mathcal{L}) = \int_{\mathbb{R}^D} \prod_{m=1}^{M} \mathbb{1}\left[\boldsymbol{a}_m^\top \boldsymbol{x} + b_m > 0\right] \mathcal{N}(\boldsymbol{x}; 0, \boldsymbol{I})\, \mathrm{d}\boldsymbol{x}, \tag{7.2}$$

in terms of the indicator function

$$\mathbb{1}[X] := \begin{cases} 1 & \text{if } X \text{ evaluates to } \texttt{true} \\ 0 & \text{if } X \text{ evaluates to } \texttt{false}. \end{cases}$$

which simply acts as a selector regardless of whether $\boldsymbol{x}$ falls into $\mathcal{L}$ or not. The setup is illustrated in Figure 7.1. We use the shorthand notation

$$\mathbb{1}_{\mathcal{L}} = \prod_{m=1}^{M} \mathbb{1}\left[\boldsymbol{a}_m^\top \boldsymbol{x} + b_m > 0\right] = \mathbb{1}[\boldsymbol{x} \in \mathcal{L}]. \tag{7.3}$$

It is also convenient to write the linear constraints of (7.2) in vectorial form,

$$\boldsymbol{A}^\top \boldsymbol{x} + \boldsymbol{b}, \tag{7.4}$$

where $\boldsymbol{A} \in \mathbb{R}^{D \times M}$, $\boldsymbol{x} \in \mathbb{R}^D$, and $\boldsymbol{b} \in \mathbb{R}^M$. We take the integration measure to be a standard normal without loss of generality, because any correlated multivariate Gaussian can be whitened by linearly transforming the integration variable.
For example, orthant probabilities of a correlated Gaussian $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be written in the form of (7.2) by using the transformation $\boldsymbol{x} = \boldsymbol{L}\boldsymbol{z} + \boldsymbol{\mu}$, where $\boldsymbol{L}$ is the Cholesky factor of $\boldsymbol{\Sigma}$. Typically, we expect $M \geq D$, *i.e.*, there are at least as many linear constraints as dimensions. This is because if $M < D$, there exist $D - M$ unconstrained directions that can be integrated out in closed form after an appropriate linear transformation, and an M-dimensional integral with M constraints remains. However, there are situations in which integrating out dimensions might be undesired, for instance in probit regression, where samples from the untransformed integrand are required for posterior predictions.

7.2 Motivation

Gaussian models with linear domain constraints occur in a myriad of applications that span all disciplines of applied statistics. They arise when a joint Gaussian assumption is made on a set of random variables, but there are known inequality constraints on linear combinations of these variables, which is usually captured by the likelihood. Instances of truncated Gaussians appear in biostatistics [242], medicine [47], physics [268], environmental sciences [256], robotics and control [70], machine learning [239] and many more.

A common occurrence of *integrals* over linearly restricted Gaussians is in spatial statistics, such as Markov random fields [32], the statistical modeling of spatial extreme events called max-stable processes [120, 85], or in modeling uncertainty regions for latent Gaussian models. An example for the latter is to find regions that are likely to exceed a given reference level, *e.g.*, pollution levels in geostatistics and environmental monitoring [32], or in climatology [76]. Another area where such integrals are often encountered is in reliability analysis [15, 172, 8, 238]. A key

[242]: Thiébaut and Jacqmin-Gadda (2004), 'Mixed models for longitudinal left-censored repeated measures'

[47]: Chen and Chang (2007), 'Identification of the minimum effective dose for right-censored survival data'

[268]: Zhou et al. (2019), 'Reexamining the proton-radius problem using constrained Gaussian processes'

[256]: Wani et al. (2017), 'Parameter estimation of hydrologic models using a likelihood function for censored and binary observations'

[70]: Fisac et al. (2018), 'A general safety framework for learning-based control in uncertain robotic systems'

[239]: Su et al. (2016), 'Nonlinear statistical learning with truncated Gaussian graphical models'

[32]: Bolin and Lindgren (2015), 'Excursion and contour uncertainty regions for latent Gaussian models'

[120]: Huser and Davison (2013), 'Composite likelihood estimation for the BrownResnick process'

[85]: Genton et al. (2011), 'On the likelihood function of Gaussian max-stable processes'

[76]: French and Sain (2013), 'Spatio-temporal exceedance locations and confidence regions'

[15]: Au and Beck (2001), 'Estimation of small failure probabilities in high dimensions by subset simulation'

[172]: Melchers and Beck (2018), *Structural reliability analysis and prediction*

[8]: Andersen et al. (2018), 'Efficient simulation for dependent rare events with applications to extremes'

[238]: Straub et al. (2020), 'Reliability analysis of deteriorating structural systems'

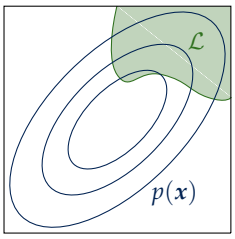


Figure 7.2: Sketch of the estimation of the reliability of a system by computing its complement, the probability of failure, *i.e.*, the probability mass of the domain in parameter space that causes a failure mode.

problem there is to estimate the probability for a rare event (*e.g.*, a flood) to occur or for a mechanical system to enter a failure mode. Commonly, such integrals even deal with considerably more complicated domain boundaries than the linear ones considered here (see Figure 7.2 for a sketch).

In machine learning, there are many Bayesian models in which linearly constrained multivariate normal distributions play a role, such as Gaussian processes under linear constraints [163, 164, 3, 57], inference in graphical models [181], multi-class Gaussian process classification [212], ordinal and probit regression [155, 14], incomplete data classification [161], and Bayesian optimization [113, 254], to name a few.

7.2.1 Related work

This practical relevance has fed a slow-burn research effort in the integration of truncated Gaussians over decades [91, 86, 126, 250, 188]. Gassmann et al. [79] and Genz and Bretz [88] provide comparisons and attest the best accuracy across a wide range of test problems to the algorithm by Genz [86], which has made it a default choice in the literature. Genz's method, detailed in Section 7.4.4, applies a sequence of transformations to transform the integration region to the unit cube $[0, 1]^D$ and then solves the integral numerically using quasi-random integration points. Other methods focus on specialized settings such as bivariate or trivariate Gaussian probabilities [87, 107], or on orthant probabilities [177, 55, 189, 106]. Yet, these methods are only feasible for at most a few tens of variables. Only recent advances have targeted higher-dimensional integrals: Azzimonti and Ginsbourger [17] study high-dimensional orthant probabilities and Genton et al. [84] consider the special case where the structure of the covariance matrix allows for hierarchical decomposition to reduce computational complexity. Phinikettos and Gandy [203] employ a combination of four variance reduction techniques to solve such integrals with Monte Carlo methods. Botev [35] constructs an exponential tilting of an importance sampling measure that builds on the method by Genz [86] and reports effectiveness for dimensions $D \lesssim 100$. A different approach has been suggested by Cunningham et al. [56]: They use expectation propagation (EP) to approximate the constrained normal integrand by a moment-matched multivariate normal density (*cf.* Section 7.4.4). This allows for fast integration, at the detriment of guarantees. Indeed, the authors report cases in which EP is far off the ground truth integral.

Closely related to integration is *simulation* from linearly constrained Gaussians, yet these tasks have rarely been considered concurrently, except for Botev [35] who proposes an accept-reject sampler alongside the integration scheme. Earlier attempts employ Markov chain Monte Carlo techniques (*cf.* Chapter 4) such as Gibbs sampling [91], or Hamiltonian Monte Carlo [195]. Previous approaches to sampling are discussed in more detail in Section 7.3.2.

[163]: López-Lopera et al. (2018), 'Finite-dimensional Gaussian approximation with linear inequality constraints'

[164]: López-Lopera et al. (2019), 'Gaussian process modulated cox processes under linear inequality constraints'

[3]: Agrell (2019), 'Gaussian Processes with linear operator inequality constraints'

[57]: Da Veiga and Marrel (2012), 'Gaussian process modeling with inequality constraints'

[181]: Mulgrave and Ghosal (2018), 'Bayesian inference in nonparanormal graphical models'

[212]: Rasmussen and Williams (2006), *Gaussian Processes for machine learning*

[155]: Lawrence et al. (2008), 'Bayesian inference for multivariate ordinal data using parameter expansion'

[14]: Ashford and Sowden (1970), 'Multivariate probit analysis'

[161]: Liao et al. (2007), 'Quadratically gated mixture of experts for incomplete data classification'

[113]: Hennig and Schuler (2012), 'Entropy search for information-efficient global optimization.'

[254]: Wang et al. (2020), 'Parallel Bayesian global optimization of expensive functions'

[91]: Geweke (1991), 'Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint probabilities'

[86]: Genz (1992), 'Numerical computation of multivariate normal probabilities'

[126]: Joe (1995), 'Approximations to multivariate normal rectangle probabilities based on conditional expectations'

[250]: Vijverberg (1997), 'Monte Carlo evaluation of multivariate normal probabilities'

[188]: Nomura (2014), 'Computation of multivariate normal probabilities with polar coordinate systems'

[79]: Gassmann et al. (2002), 'Computing multivariate normal probabilities: A new look'

[88]: Genz and Bretz (2009), *Computation of multivariate normal and t probabilities*

[87]: Genz (2004), 'Numerical computation of rectangular bivariate and trivariate normal and t probabilities'

[107]: Hayter and Lin (2013), 'The evaluation of trivariate normal probabilities defined by linear inequalities'

[177]: Miwa et al. (2003), 'The evaluation of general non-centred orthant probabilities'

[55]: Craig (2008), 'A new reconstruction of multivariate normal orthant probabilities'

[189]: Nomura (2016), 'Evaluation of Gaussian orthant probabilities based on orthogonal projections to subspaces'

7.2.2 Why not Bayesian quadrature?

Failure probabilities, or in the simpler case, linearly constrained Gaussian densities are inherently difficult settings for Bayesian quadrature. The involved step function that delimits the region of interest is the cause of trouble for Bayesian quadrature (BQ). Under such circumstances, the need to choose a suitable prior clashes with the kernel integrability requirements (2.15) and (2.16). Selecting a covariance function according to this constraint entails a misspecified model that is unable to capture the geometry of the step function. Non-stationary covariance functions that fit the needs of capturing the step come at the detriment of not being integrable [212] . The alternative approach of keeping the indicator function in the integral, *i.e.*, to adapt the integration domain according to the constraints, is not a viable option because they equally break the integrability of the kernel mean and variance. In reliability analysis with expensive simulators, Gaussian processes are nevertheless used as a surrogate that is then integrated using Monte Carlo. This does not apply to Gaussian probabilities, where the step function, the adversary of BQ, turns out to play exceptionally well with Monte Carlo methods.

[106]: Hayter and Lin (2012), 'The evaluation of two-sided orthant probabilities for a quadrivariate normal distribution'

[17]: Azzimonti and Ginsbourger (2017), 'Estimating orthant probabilities of high-dimensional Gaussian vectors with an application to set estimation'

[84]: Genton et al. (2018), 'Hierarchical decompositions for the computation of high-dimensional multivariate normal probabilities'

[203]: Phinikettos and Gandy (2011), 'Fast computation of high-dimensional multivariate normal probabilities'

[35]: Botev (2016), 'The normal law under linear restrictions: Simulation and estimation via minimax tilting'

[56]: Cunningham et al. (2011), 'Gaussian probabilities and expectation propagation'

[195]: Pakman and Paninski (2014), 'Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians'

7.3 Sampling from truncated Gaussians

We first focus on the problem of *simulating* from the domain $\mathcal{L}$. Naïve rejection sampling, *i.e.*, drawing from the Gaussian and rejecting samples that do not satisfy the constraints, quickly becomes impracticable as dimension increases and the probability of a sample to fall into the domain plummets. It is illustrative to remember that orthant probabilities under a standard normal distribution decrease exponentially with the dimension as 2^{-D}, and we can expect a similar scaling for tilted constraints. We therefore resort to a Markov chain Monte Carlo method that we dub LIN-ESS. This routine is a special case of elliptical slice sampling [182] (ESS, *cf.* Section 4.2.5) that leverages the analytic tractability of intersections of ellipses and hyperplanes to speed up the ESS loop. LIN-ESS permits rejection-free sampling from a linearly constrained Gaussian domain of arbitrarily small mass once an initial sample within the domain is known. LIN-ESS acts at the back-end of the integration method, which is introduced in Section 7.4.

[212]: Rasmussen and Williams (2006), *Gaussian Processes for machine learning* Chapter 5.4, see Figure 5.9 and 5.10

[182]: Murray et al. (2010), 'Elliptical slice sampling'

7.3.1 Elliptical slice sampling on linearly constrained domains

elliptical slice sampling (ESS) is designed for generic likelihood functions under a multivariate normal prior. In a truncated Gaussian model, the selector function $\mathbb{1}_{\mathcal{L}}$ in (7.2) and (7.3) takes the rôle of the likelihood function. The particular form of this likelihood can be leveraged to significantly simplify the ESS algorithm:

1. The 'selector likelihood' $\ell(\boldsymbol{x}) := \mathbb{1}_{\mathcal{L}}$ can take only the values 0 and 1. Hence, there is no need for a likelihood threshold, the slice is always defined by $\ell(\boldsymbol{x}) = 1$ for $\boldsymbol{x}(\vartheta)$ on the ellipse.
2. The intersections between the ellipse and the linear constraints have closed-form solutions. The angular domain(s) to sample from can be constructed analytically, and LIN-ESS is thus rejection-free.

The typical bisection search of slice sampling becomes a simple analytic expression.

With these simplifications to ESS, each new sample from $\mathcal{L}$ requires exactly *one* auxiliary normal sample $\boldsymbol{\xi} \sim \mathcal{N}(0, \boldsymbol{I}) \in \mathbb{R}^D$ and a scalar uniform sample $u \sim \text{Uniform}[0,1]$ to sample from the angular bracket. Figure 7.3 illustrates the process of drawing a sample from the domain of interest (blue shaded area) using our version of ESS. Given the current state $\boldsymbol{x}_n \in \mathcal{L}$ and an auxiliary vector $\boldsymbol{\xi}$, the ellipse is parameterized by its angle $\vartheta \in [0, 2\pi]$ as

$$\boldsymbol{x}(\vartheta) = \boldsymbol{x}_n \cos\vartheta + \boldsymbol{\xi} \sin\vartheta.$$

The intersections between the ellipse and the domain boundaries given by $\boldsymbol{A}^\top \boldsymbol{x} + \boldsymbol{b} = \boldsymbol{0}$ can be found by solving the set of equations $\boldsymbol{A}^\top \boldsymbol{x}(\vartheta) + \boldsymbol{b} = \boldsymbol{0}$ for ϑ. For the m^{th} constraint, this equation has between zero and two closed-form solutions, [1]

$$\vartheta_m^{1/2} = \pm \arccos\left(-\frac{b_m}{r}\right) + \arctan\left(\frac{\boldsymbol{a}_m^\top \boldsymbol{\xi}}{r + \boldsymbol{a}_m^\top \boldsymbol{x}_n}\right) \tag{7.5}$$

with $r = \sqrt{(\boldsymbol{a}_m^\top \boldsymbol{x}_n)^2 + (\boldsymbol{a}_m^\top \boldsymbol{\xi})^2}$. Not all intersection angles lie on the domain boundary, and we need to identify those *active* intersections where $\ell(\boldsymbol{x}(\vartheta))$ switches on or off. To identify potentially multiple brackets, we sort the angles in increasing order and check for each of them if adding or subtracting a small $\delta\vartheta$ causes a likelihood jump. If there is no jump, the angle is discarded, otherwise the sign of the jump is stored (whether from 0 to 1 or the reverse), in order to know the direction of the relevant domain on the slice. The procedure for LIN-ESS can be found in Algorithm Algorithm 7.1 below. The computational cost of drawing one sample on the ellipse is dominated by the M inner products that need to be computed for the intersections, hence the complexity is $\mathcal{O}(MD)$. At first glance, this seems comparable with standard ESS for which drawing from a multivariate normal distribution is $\mathcal{O}(D^2)$, but this scaling does not include the evaluation of the likelihood function yet.

LIN-ESS is a rejection-free sampling method to sample from a truncated Gaussian of arbitrarily small mass. Yet it requires an initial point within the domain from where to launch the Markov chain. How to obtain such a sample without falling into the rejection trap will be discussed in Section 7.4.2.

In principle, LIN-ESS supports sampling from non-convex domains. Valid slices on the ellipse may be disconnected even when the domain is defined through a convex polytope. Since the bracket-defining angles are available in closed form, the uniform sample can be mapped to a valid section on the ellipse by respecting the total arclength of the valid sections. This procedure applies to both convex and non-convex domains; from the sampling perspective, there is hence no need to discriminate between those. The challenging aspect of non-convex domains is an appropriate extension of the definition of the linear constraints. This is easily done if we considered unions instead of intersections, *i.e.*, $\mathcal{L} = \left\{\boldsymbol{x} : \boldsymbol{x} \in \bigcup_{m=1}^{M} \left\{\boldsymbol{a}_m^\top \boldsymbol{x} + b_m > 0\right\}\right\}$ instead of (7.1). More general cases would require the integral to be split up into sums of domains $\mathbb{1}_{\mathcal{L}_k}$ which amounts to sampling from multiple convex domains.

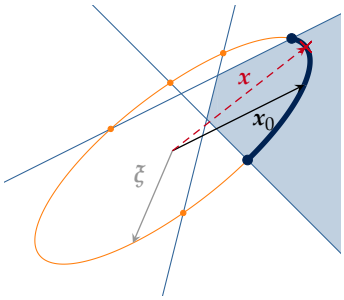


Figure 7.3: Sampling from a constrained normal space using ESS. $\boldsymbol{x}_0$ is a current state from the domain $\mathcal{L}$ and, together with the auxiliary $\boldsymbol{\xi}$, defines the ellipse. From all intersections of the ellipse and zero lines (or hyperplanes in higher dimensions), the *active* intersections at the domain boundary are identified (•). These define the slice from which a uniform sample is drawn ($\boldsymbol{x}$).

1: The m^{th} hyperplane and the ellipse either intersect at two points, namely when $|b_m| < r$, or they do not, when $|b_m| > r$, since $\arccos : [-1, 1] \to \mathbb{R}$. A tangential intersection, leading to a single solution at $|b_m| = r$, is unlikely to occur.

Algorithm 7.1 Elliptical slice sampling for a linearly constrained standard normal distribution

procedure LinESS($\boldsymbol{A}, \boldsymbol{b}, N, \boldsymbol{x}_0$)
  ensure all $(\boldsymbol{a}_m^\top \boldsymbol{x}_0 + b_m > 0 \; \forall m)$ // initial vector needs to be in domain
  $\boldsymbol{X} = \varnothing$ // initialize sample array
  for $n = 1, \dots, N$ **do**
    $\boldsymbol{\xi} \sim \mathcal{N}(0, \boldsymbol{I})$
    $\boldsymbol{x}(\vartheta) = \boldsymbol{x}_0 \cos\vartheta + \boldsymbol{\xi} \sin\vartheta$ // construct ellipse
    $\boldsymbol{\vartheta} \leftarrow \mathbf{sort}(\{\vartheta_j^{1/2}\}_{j=1}^{M})$ s.t. $\boldsymbol{a}_j^\top (\boldsymbol{x}_0 \cos\vartheta_j^{1/2} + \boldsymbol{\xi} \sin\vartheta_j^{1/2}) = 0$ // $2M$ intersections, Eq. (7.5)
    $\boldsymbol{\vartheta}_{\text{act}} \leftarrow \{[\vartheta_l^{\min}, \vartheta_l^{\max}]\}_{l=1}^{L}$ s.t. $\ell(x(\vartheta_l^{\min/\max} + \delta\vartheta)) - \ell(x(\vartheta_l^{\min/\max} - \delta\vartheta)) = \pm 1$ // Set brackets
    $u \sim [0,1] \cdot \sum_l^L (\vartheta_l^{\max} - \vartheta_l^{\min})$
    $\vartheta_u \leftarrow$ transform u to angle in bracket
    $\boldsymbol{X}[n] \leftarrow \boldsymbol{x}(\vartheta_u)$ // update sample array
    $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}(\vartheta_u)$ // set new initial vector
  end for
  return X
end procedure

7.3.2 Related sampling schemes

Markov chain Monte Carlo (MCMC) sampling Other common MCMC samplers also simplify in the considered setting. A well-known approach to sampling from truncated Gaussians is the Gibbs sampler [91, 146]. Dimension-wise sampling reduces each conditional to a one-dimensional truncated Gaussian which can be sampled from efficiently [215, 59]. The difficulty with Gibbs sampling arises when variables are strongly correlated, which can deteriorate the mixing process.

[91]: Geweke (1991), 'Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint probabilities'

[146]: Kotecha and Djuric (1999), 'Gibbs sampling approach for generation of truncated multivariate Gaussian random variables'

[215]: Robert (1995), 'Simulation of truncated normal variables'

[59]: Damien and Walker (2001), 'Sampling truncated normal, beta, and gamma densities'

Pakman and Paninski [195] found that Hamiltonian Monte Carlo, just as ESS, becomes very efficient in the specific case of linear or quadratic constraints. With the base measure a standard normal distribution, the Hamiltonian is

$$H(\boldsymbol{x}, \boldsymbol{p}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{x} + \frac{1}{2}\boldsymbol{p}^\top \boldsymbol{p}$$

with momentum $\boldsymbol{p} \in \mathbb{R}^D$. The Hamiltonian equations of motion are

$$\dot{\boldsymbol{x}} = \nabla_{\boldsymbol{p}} H = \boldsymbol{p} \qquad \text{and} \qquad \dot{\boldsymbol{p}} = -\nabla_{\boldsymbol{x}} H = -\boldsymbol{x},$$

which are the equations of the simple harmonic oscillator, $\ddot{\boldsymbol{x}} = -\boldsymbol{x}$. Trajectories have the closed-form solution $\boldsymbol{x}(t) = \boldsymbol{c}_0 \sin(t) + \boldsymbol{c}_1 \cos(t)$. The amplitudes $\boldsymbol{c}_0$ and $\boldsymbol{c}_1$ are determined through the initial conditions $\boldsymbol{x}(0)$ and $\boldsymbol{p}(0)$. For each trajectory, the time of collision with a linear constraint can be computed, at which the fictitious particle bounces off elastically. As most compute goes into finding wall bounces, the efficiency of the algorithm depends on the geometry of the domain. Non-convex domains might entail very low probability transitions, which is an issue ESS does not suffer from.

[195]: Pakman and Paninski (2014), 'Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians'

Exact sampling Botev [35] achieves exact sampling using minimax tilting. This method builds on the popular integration method by Genz [86] by further transforming the integrand using exponential tilting, which associates a density $p(\boldsymbol{x})$ with its exponentially tilted counterpart

[35]: Botev (2016), 'The normal law under linear restrictions: Simulation and estimation via minimax tilting'

[86]: Genz (1992), 'Numerical computation of multivariate normal probabilities'

$e^{\boldsymbol{\mu}^\top \boldsymbol{x} - K(\boldsymbol{\mu})} p(\boldsymbol{x})$ with tilting parameter $\boldsymbol{\mu}$ and the cumulant generating function $K(\boldsymbol{\mu})$. The tilting parameter can be optimized to achieve the best possible asymptotic efficiency. Estimation itself is relies on importance sampling from a density that becomes asymptotically indistinguishable from the true truncated Gaussian. Acceptance rates are clearly superior after exponential tilting of the proposal density, but drop with dimensionality. The author therefore proposes to speed up Gibbs sampling in high dimensions by sampling manageable chunks of variables using minimax tilting.

7.3.3 Discussion

In the setting of linear constraints, ESS simplifies to a rejection-free MCMC method. Its key advantage over the methods discussed in Section 7.3.2, LIN-ESS is able to deal with non-convex or even disconnected linearly restricted domains. The former case arises for example in minimax optimization problems, when a function is maximized w.r.t. one variable, and minimized w.r.t. another [257]. In the case of disconnected domains, the algorithm detects all active intersections with a likelihood jump and samples from the entire slice, no matter if connected or not.[2] The corresponding version of Hamiltonian Monte Carlo is confined to a single domain and would exhibit difficulties in non-convex domains. Given there are a number of samplers available at this point, an empirical comparison of their performance in various settings would be desirable, but would require a considerable amount of labor to ensure comparable implementations.

[257]: Weichert and Kister (2021), 'Bayesian optimization for min max optimization'

2: The current implementation of the algorithm does not allow the definition of multiple domains. However, this is a simple extension, since the sampler is already sufficiently general to incorporate disconnected slices.

7.4 From rare event estimation to Gaussian probabilities

Reliability analysis has brought forth a zoo of algorithms to estimate probabilities of rare events.[3] Sampling techniques for rare event estimation are conceptually similar in that they direct the simulation towards the regions in parameter space that initiate a rare event. The integral is decomposed into easier-to-estimate probabilities that can be estimated as samples draw closer to the domain of interest.

Popular contemporary algorithms go under the keywords *multilevel splitting* [96, 152], *sequential Monte Carlo* [43] and *subset simulation* [15], to name a few. We will here consider *subset simulation* and a closely related method named after Diaconis and Holmes [67] and Ross [216], which will serve as integrators for the linearly restricted multivariate normal distributions. The rejection-free LIN-ESS takes enters the game as an MCMC sampler. It renders the need for tricks to drag the sampler towards the domain of interest obsolete.

3: The initial idea dates back to at least 1951, referred to as *multilevel technique* by Kahn and Harris [127], which they attribute to von Neumann. They were concerned with estimating particle transmission probabilities in nuclear physics for which they define *regions of importance* and,

> *[. . .] when the sampled particle goes from a less important to a more important region, it is split into two independent particles, each one-half of the weight of the original. [. . .] The purpose of this is to spend most of the time studying the important rather than the typical particles [. . .].*

[96]: Glasserman et al. (1999), 'Multilevel splitting for estimating rare event probabilities'

[152]: Lagnoux and Lezaud (2017), 'Multilevel branching and splitting algorithm for estimating rare event probabilities'

[43]: Cérou et al. (2012), 'Sequential Monte Carlo for rare event estimation'

[15]: Au and Beck (2001), 'Estimation of small failure probabilities in high dimensions by subset simulation'

[67]: Diaconis and Holmes (1995), 'Three examples of Monte-Carlo Markov chains: At the interface between statistical computing, computer science, and statistical mechanics'

[216]: Ross (2012), *Simulation*

[150]: Kroese et al. (2013), *Handbook of Monte Carlo methods*

7.4.1 The Holmes-Diaconis-Ross algorithm

The Holmes-Diaconis-Ross algorithm (HDR) [67, 216, 150] is a specialized method for constructing an unbiased estimator for probabilities of the form $P(\boldsymbol{x} \in \mathcal{L})$ under an arbitrary prior measure $\boldsymbol{x} \sim p_0(\boldsymbol{x})$ and a domain $\mathcal{L} = \{\boldsymbol{x} \text{ s.t. } f(\boldsymbol{x}) \geq 0\}$ with a deterministic function $f : \mathbb{R}^D \mapsto \mathbb{R}$. If this domain has very low probability mass, $P(\mathcal{L})$ is expensive to compute with simple Monte Carlo because most samples are rejected. HDR mitigates

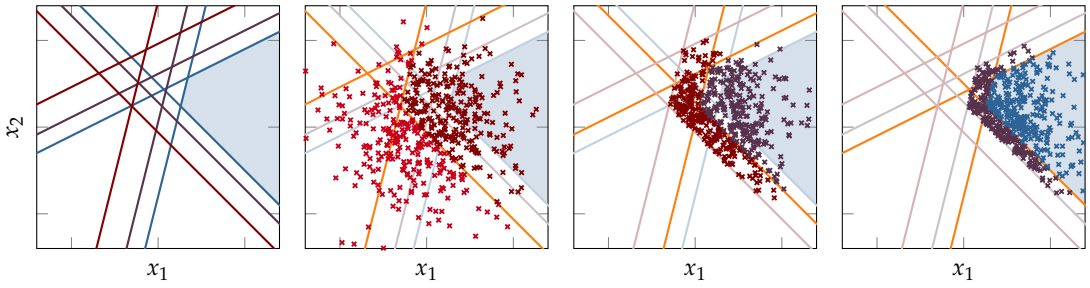


Figure 7.4: Sketch of the HDR algorithm in a bivariate setting. The nested domains are fixed *a priori (left)*. Conditional probabilities are estimated from the number of samples drawn from the current domain that also fall into the subsequent nesting, until all the nestings have been traversed and the domain of interest is reached *(left to right)*.

this by using a sequence of T nested domains $\mathbb{R}^D = \mathcal{L}_0 \supset \mathcal{L}_1 \supset \mathcal{L}_2 \supset \ldots \supset \mathcal{L}_T = \mathcal{L}$, s.t. $\mathcal{L}_t = \bigcap_{i=1}^{t} \mathcal{L}_i$. The probability mass of the domain of interest can be decomposed into a product of conditional probabilities,

$$Z = P(\mathcal{L}) = P(\mathcal{L}_0) \prod_{t=1}^{T} P(\mathcal{L}_t | \mathcal{L}_{t-1}). \tag{7.6}$$

The conditional probabilities $P(\mathcal{L}_t|\mathcal{L}_{t-1})$ are estimated from samples as the fraction of samples in $\mathcal{L}_{t-1}$ that also fall into $\mathcal{L}_t$. If each of the conditional probabilities $P(\mathcal{L}_t|\mathcal{L}_{t-1})$ is closer to $1/2$, they all require quadratically fewer samples, reducing the overall cost despite the linear increase in individual sampling problems. Noting that $P(\mathcal{L}_0) = 1$ and introducing the shorthand $\rho_t = P(\mathcal{L}_t|\mathcal{L}_{t-1})$, (7.6) can be written in logarithmic form as

$$\log Z = \sum_{t=1}^{T} \log \rho_t.$$

HDR does not deal with the construction of the nested domains—a method to obtain them is discussed in Section 7.4.2. For now, they are assumed to be given in terms of a decreasing sequence of positive scalar values $\{\gamma_1, \ldots, \gamma_T\}$, where $\gamma_T = 0$. Each shifted domain $\mathcal{L}_t$ can then be defined through its corresponding shift value γ_t. In the general setting, this is $\mathcal{L}_t = \{\boldsymbol{x} \text{ s.t. } f(\boldsymbol{x}) + \gamma_t \geq 0\}$; in our specific problem of linear constraints, $\boldsymbol{x} \in \mathcal{L}_t$ if $\ell_t(\boldsymbol{x}) = \prod_{m=1}^{M} \mathbb{1}[\boldsymbol{a}_m^\top \boldsymbol{x} + b_m + \gamma_t > 0] = 1$. Any positive shift γ_t thus induces a domain $\mathcal{L}_t$ that contains all domains $\mathcal{L}_{t'}$ with $\gamma_{t'} < \gamma_t$, and that engulfs a larger volume than $\mathcal{L}_{t'}$. The T^{th} shift $\gamma_T = 0$ identifies $\mathcal{L}$ itself.

Given the shift sequence $\{\gamma_1, \ldots, \gamma_T\}$, the HDR algorithm proceeds as follows: Initially, N samples are drawn from $\mathcal{L}_0$, the integration measure, in our case a standard normal. $\mathcal{L}_0$ corresponds to $\gamma_0 = \infty$ which is ignored in the sequence. The conditional probability $\rho_1 = P(\mathcal{L}_1 \,|\, \mathcal{L}_0)$ is estimated as the fraction of samples from $\mathcal{L}_0$ that also fall into $\mathcal{L}_1$. To estimate the subsequent conditional probabilities ρ_t for $t > 1$ as the fraction of samples from $\mathcal{L}_{t-1}$ falling into $\mathcal{L}_t$, standard HDR uses an MCMC sampler to simulate from $\mathcal{L}_{t-1}$. If the sequence of nestings is chosen well and initial seeds in the domain $\mathcal{L}_{t-1}$ are known, these samplers achieve a high acceptance rate. This procedure is repeated until $t = T$. The procedure is illustrated in Figure 7.4 for our setting with a domain constrained by linear functions. With the estimated conditional

Algorithm 7.2 The Holmes-Diaconis-Ross algorithm applied to linearly constrained Gaussians

```
1  procedure HDR(A, b, {γ_1, ..., γ_T}, N)
2      X ~ N(0, I)                                   // N samples
3      log Ẑ = 0                                     // initialize log integral value
4      for t = 1 ... T do
5          L_t = {x : min_m(a_m^⊤ x_n + b_m) + γ_t > 0}_{n=1}^N   // find samples inside current nesting
6          log Ẑ ← log Ẑ + log(#(X ∈ L_t)) − log N
7          choose x_0 ∈ L_t
8          X ← LinESS(A, b + γ_t, N, x_0)            // draw new samples from constrained domain
9      end for
10     return log Ẑ
11 end procedure
```

probabilities $\hat{\rho}_t$, the estimator for the probability mass is then

$$\log \hat{Z} = \sum_{t=1}^{T} \log \hat{\rho}_t.$$

In our adapted version of HDR, the LIN-ESS algorithm (*cf.* Section 7.3) comes into play, which achieves a 100% acceptance rate for simulating from the nested domains. In order to simulate rejection-free from $\mathcal{L}_t$, LIN-ESS requires an initial sample from the domain $\mathcal{L}_t$, which is obtained from the previous iteration of the algorithm. Every location sampled requires evaluating the linear constraints, hence the cost for each subset in HDR is $\mathcal{O}(NMD)$. Pseudocode for this algorithm is shown in Algorithm 7.2, where LinESS is a call to the LIN-ESS sampler (*cf.* Section 7.3 and Algorithm 7.1) that simulates from the linearly constrained domain.

7.4.2 Obtaining nested domains

As the final missing ingredient, the HDR algorithm requires a sequence of nested domains or level sets defined by positive shifts γ_t, $t = 1, \ldots, T$. In theory, the nested domains should ideally have conditional probabilities of $\rho_t = 1/2 \ \forall t$ (then each nesting improves the precision by one bit). Yet, in a more practical consideration, the computational overhead for constructing the nested domains should also be small. In practice, the shift sequence is often chosen in an ad hoc way, hoping that conditional probabilities are large enough to enable a decently accurate estimation via HDR [131]. This is not straightforward and requires problem-specific knowledge.

[131]: Kanjilal and Manohar (2015), 'Markov chain splitting methods in structural reliability integral estimation'

We suggest constructing the nestings via subset simulation [15] which is very similar to HDR. It only differs in that the conditional probabilities ρ_t are fixed a priori to a value ρ, and then the shift values γ_t are computed such that a fraction ρ of the N samples drawn from $\mathcal{L}_{t-1}$ falls into the subsequent domain $\mathcal{L}_t$.

[15]: Au and Beck (2001), 'Estimation of small failure probabilities in high dimensions by subset simulation'

The construction of the nested domains is depicted in Figure 7.5. To find the shifts, N samples are drawn from the integration measure initially (left-most scene in Figure 7.5). Then the first (and largest) shift γ_1 is determined such that a fraction ρ of the samples fall into the domain $\mathcal{L}_1$. This is achieved by computing for each sample by how much the linear

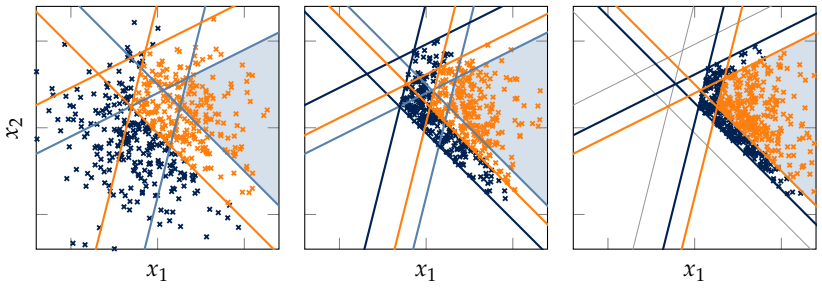


Figure 7.5: Finding the level sets in subset simulation for linear constraints. *Left:* Draw standard normal samples and find the shift γ_1 for which a fraction ρ of the samples lie inside the new domain (orange lines); *center:* Use LIN-ESS to draw samples from the subsequent domain defined by γ_1 (now in dark blue) and find γ_2 (orange lines) similarly; *right:* Proceed until the domain of interest (shaded area) is reached. Details in text.

constraints would need to be shifted to encompass the sample. For the subsequent shifts, N samples are simulated from the current domain $\mathcal{L}_{t-1}$, and the next shift γ_t is again set s.t. $\lfloor N\rho \rfloor$ samples fall into the next domain $\mathcal{L}_t$ (Figure 7.5, center). This requires an initial sample from $\mathcal{L}_{t-1}$ to launch the LIN-ESS sampler, which is obtained from the samples gathered in the previous nesting $\mathcal{L}_{t-1}$ that also lie in $\mathcal{L}_t$, while all other samples are discarded to reduce dependencies. This nesting procedure is repeated until more than $\lfloor N\rho \rfloor$ samples fall into the domain of interest $\mathcal{L}$ (Figure 7.5, right). We set $\rho = 1/2$ to maximize the entropy of the binary distribution over whether samples fall in- or outside the next nested domain, yet in reliability analysis a common choice is $\rho = 0.1$ [16], which has the advantage of requiring less nestings (to the detriment of more samples). Pseudocode can be found in Algorithm 7.3.

[16]: Au and Beck (2001), 'First excursion probabilities for linear systems by very efficient importance sampling'

In fact, subset simulation itself also permits the estimation of the integral Z, without appealing to HDR: Since the subsets are constructed such that the conditional probabilities take a predefined value, the estimator for the integral is $\hat{Z}_{\text{ss}} = \rho^{T-1}\rho_T$ where $\rho_T = P(\mathcal{L}_T|\mathcal{L}_{T-1}) \in [\rho, 1]$ is the conditional probability for the last domain. For $\rho = 1/2$ the number of required nestings is roughly the negative binary logarithm of the integral estimator $T \approx -\log_2 \hat{Z}_{\text{ss}}$ (*cf.* Figure 7.6). The main reason not to rely on subset simulation alone is that its estimator $\hat{Z}_{\text{ss}}$ is biased, because the samples are both used to construct the domains and to estimate Z. We thus use HDR to get an unbiased estimate of the integral from level sets constructed with subset simulation.

Both subset simulation and HDR are instances of a wider class of so-called *multilevel splitting* methods which are related to *sequential Monte Carlo (SMC)* in that they are concerned with simulating from a sequence of probability distributions. SMC methods (aka. *particle filters*) were conceived for online inference in state space models, but can be extended to non-Markovian latent variable models [183]. In this form, SMC methods have gained popularity for the estimation of rare events [65, 24, 43].

[183]: Naesseth et al. (2019), 'Elements of sequential Monte Carlo'

[65]: Del Moral et al. (2006), 'Sequential Monte Carlo samplers'

[24]: Bect et al. (2017), 'Bayesian subset simulation'

[43]: Cérou et al. (2012), 'Sequential Monte Carlo for rare event estimation'

7.4.3 Derivatives of Gaussian probabilities

Many applications (e.g. Bayesian optimization, see below) additionally require *derivatives* of the Gaussian probability w.r.t. to parameters λ of the integration measure or the linear constraints. The absence of such derivatives in classic quadrature sub-routines has thus sometimes been mentioned as an argument against them (*e.g.*, [56]). Our integration method allows to efficiently compute such derivatives, because it produces samples. This leverages the classic result that derivatives of

[56]: Cunningham et al. (2011), 'Gaussian probabilities and expectation propagation'

Algorithm 7.3 Subset simulation for linear constraints

procedure SUBSETSIM($\boldsymbol{A}, \boldsymbol{b}, N, \rho = \frac{1}{2}$)
  $\gamma \leftarrow \varnothing$ // initialize shift sequences
  $\boldsymbol{X} \sim \mathcal{N}(0, \boldsymbol{I})$ // N initial samples
  $\gamma_{\text{new}}, \hat{\rho} = \text{FINDSHIFT}(\rho, \boldsymbol{X}, \boldsymbol{A}, \boldsymbol{b})$ // find new shift value
  $\log \hat{Z} = \log \hat{\rho}$ // record the integral
  while $\gamma_{\text{new}} > 0$ **do**
    $\boldsymbol{X} \leftarrow \text{LINESS}(\boldsymbol{A}, \boldsymbol{b} + \gamma_{\text{new}}, N, \boldsymbol{x}_0)$ // draw new samples from new constrained domain using Algorithm 7.1
    $\gamma_{\text{new}}, \hat{\rho} \leftarrow \text{FINDSHIFT}(\rho, \boldsymbol{X}, \boldsymbol{A}, \boldsymbol{b})$ // find new shift value
    $\gamma \leftarrow \gamma \cup \{\gamma_{\text{new}}\}$
    $\log \hat{Z} \leftarrow \log \hat{Z} + \log \hat{\rho}$ // update integral with new conditional probability
  end while
  return $\log \hat{Z}, \gamma$
end procedure

function FINDSHIFT($\rho, \boldsymbol{X}, \boldsymbol{A}, \boldsymbol{b}$) // find shift s.t. a fraction ρ of $\boldsymbol{X}$ fall into the resulting domain.
  $\boldsymbol{\gamma} \leftarrow \text{SORT}(-\min_m(\boldsymbol{a}_m^\top \boldsymbol{x}_n + b_m)_{n=1}^N)$ // sort shifts in ascending order
  $\gamma_{\text{new}} \leftarrow (\boldsymbol{\gamma}[\lfloor \rho N \rfloor] + \boldsymbol{\gamma}[\lfloor \rho N \rfloor + 1])/2$ // find shift s.t. ρN samples lie in the domain
  $\hat{\rho} \leftarrow \text{COUNT}(\boldsymbol{\gamma} < \gamma_{\text{new}})/N$ // true fraction could deviate from ρ
  return $\gamma_{\text{new}}, \hat{\rho}$
end function

exponential families with respect to their parameters can be computed from expectations of the sufficient statistics. To do so, it is advantageous to rephrase (7.2) as the integral over a *correlated* Gaussian with mean $\boldsymbol{\mu}_\lambda$ and covariance matrix $\boldsymbol{\Sigma}_\lambda$ with axis-aligned constraints (or constraints that are independent of λ). The derivatives w.r.t. a parameter λ can then be expressed as an expected value,

$$\frac{\mathrm{d}Z}{\mathrm{d}\lambda} = \mathbb{E}\left[\frac{\mathrm{d}\log \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_\lambda, \boldsymbol{\Sigma}_\lambda)}{\mathrm{d}\lambda}\right], \tag{7.7}$$

where the expectation is taken with respect to the transformed integrand (2.1). Since LIN-ESS permits us to simulate from the integrand of (2.1), derivatives can be estimated via expectations. We demonstrate in Section 7.5.2 that this is a lot more efficient than finite differences, which requires Z to be estimated twice and at considerably higher accuracy.

7.4.4 Related integration methods

[86]: Genz (1992), 'Numerical computation of multivariate normal probabilities'

Genz's Method The most widely employed algorithm for estimating multivariate normal probabilities is the highly optimized method by Genz [86] .[4] The procedure relies on a sequence of transformations that map the integration domain to the unit hypercube. It requires the linear constraints to take the form of a potentially half-open box for some covariance of the Gaussian, *i.e.*,

4: Parts of its success can be attributed to the efficient `Fortran` implementation, which is available (though somewhat hidden) in `Python`'s `scipy` as `scipy.stats.mvn.mvndst`.

$$Z = \int_{l_1}^{u_1} \cdots \int_{l_D}^{u_D} \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})\, \mathrm{d}x_1 \ldots \mathrm{d}x_D$$

for lower and upper bounds $l_d, u_d \in \mathbb{R}$; $d = 1, \ldots, D$, respectively. Bringing this integral into the form of (7.2) yields bounds for the d^{th}

integral that are successively dependent on those for $d' < d$ due to the triangular form of the Cholesky decomposition $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^\top$. They can further be written as cumulative Gaussians $\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\xi^2/2} \, \mathrm{d}\xi$, which is used to map the integral to the unit hypercube as

$$Z = (u'_1 - l'_1) \int_0^1 (u'_2 - l'_2) \int_0^1 (u'_3 - l'_3) \cdots \int_0^1 (u'_D - l'_D) \, \mathrm{d}\boldsymbol{\xi}$$

with $l'_d(z_1, \ldots, z_d) = \Phi\left((l_d - \sum_{i=1}^{d-1} L_{di} \Phi^{-1}(\boldsymbol{z}_i)) / L_{dd} \right)$, $u'_d(z_1, \ldots, z_d) = \Phi\left((u_d - \sum_{i=1}^{d-1} L_{di} \Phi^{-1}(\boldsymbol{z}_i)) / L_{dd} \right)$ and the additional transformation $z_d = l'_d + \xi_d (u'_d - l'_d)$. This integral can then be estimated for instance from random uniform samples, or a quasi-Monte Carlo method that operates on the hypercube.

Integration via Expectation Propagation Cunningham et al. [56] approach the same problem with [176]. Expectation propagation (EP) is an approximate inference algorithm that finds a Gaussian approximation to the non-Gaussian integrand, in this case the truncated normal distribution, by iterative moment matching. This is achieved by updating the approximate density dimension-wise— thereby reducing the integration problem to multiple univariate ones—to approximately minimize the Kullback-Leibler divergence between the Gaussian approximation and the unnormalized truncated Gaussian.[5] This scheme has proven successful *e.g.,* in Gaussian process classification [151], which is similar in that the likelihood is a softmax instead of a hard step function. EP may be orders of magnitude faster than MCMC procedures for the same task, but it has two fundamental issues: *(i)* EP works well empirically, but lacks guarantees for convergence, and *(ii)* there is no means to estimate the error committed on the integral itself. In Section 7.5 we observe that EP fails in low-probability settings.

[56]: Cunningham et al. (2011), 'Gaussian probabilities and expectation propagation'

[176]: Minka (2013), 'Expectation propagation for approximate Bayesian inference'

5: For a detailed description of the algorithm, the reader may consult *e.g.,* [212, 176, 56].

[212]: Rasmussen and Williams (2006), *Gaussian Processes for machine learning*

[151]: Kuss et al. (2005), 'Assessing approximate inference for binary Gaussian process classification.'

7.5 Applications and experiments

To shed light on the interplay of subset simulation, HDR, and LIN-ESS, we dissect a single 500-dimensional synthetic integration problem with a closed-form solution. Besides more synthetic experiments, we consider integration problems arising bin Bayesian optimization where we demonstrate our algorithm's ability to estimate derivatives.

7.5.1 Synthetic experiments

As an initial integration problem we consider axis-aligned constraints in a 500-dimensional space. Since this task amounts to computing the mass of a shifted orthant under a standard normal distribution, it allows comparison to an exact analytic answer. The goal of this setup is twofold: 1) to demonstrate that our method can compute small Gaussian probabilities to high accuracy, and 2) to explore configurations for the construction of nested domains using subset simulation. The domain is defined by $\ell(\boldsymbol{x}) = \prod_{d=1}^{D} \mathbb{1}[x_d + 1 > 0]$. The true mass of this domain is $Z = 3.07 \cdot 10^{-38} = 2^{-124.6}$. Estimating this integral naïvely by sampling from the Gaussian would require of the order of 10^{38} samples for

one to fall into the domain of interest. With a standard library like `numpy.random.randn`, this would take about 10^{15} ages of the universe.

Subset simulation First, we compute the shift sequence $\{\gamma_1, \dots, \gamma_T\}$ using subset simulation for various numbers of samples N per subset and a fixed conditional probability of $\rho = 1/2$. Since the contributing factor of each nesting is $\rho = 1/2$, the integral estimate is roughly 2^{-T} for our choice of ρ (*cf.* Section 7.4.2). In other words, subset simulation should ideally find $T = \lceil Z \rceil = 125$ nestings for the given example. The relation between the number of subsets T and the estimated integral value $\hat{Z}_{\text{ss}}$ is visualized in Figure 7.6. It shows the sequences of shift values for increasing sample sizes and the resulting integral estimate $\log_2 \hat{Z}_{\text{ss}}$. The T^{th} nesting has shift value $\gamma = 0$ and is the only subset with a conditional probability that deviates from the chosen value of ρ. Yet, T is a good indicator for the value of the negative binary logarithm of the estimated integral. Hence, we use the same axis to display the number of subsets and $-\log_2 \hat{Z}_{\text{ss}}$. The plot highlights the bias of subset simulation: For small sample sizes, e.g. $N = 2, 4, 8$, the integral is severely underestimated. This bias is caused by the dependency of the subset construction method on the samples themselves: Since we are using a MCMC method for simulating from the current domain, samples are correlated and do not fall into the *true* next subset with probability exactly ρ. This is why we only accept every 10^{th} sample to diminish this effect when constructing the subsets. For the subsequent HDR simulation, we accepted every second sample from the ESS procedure.
We choose powers of 2 for the number of samples per subset and observe that as of 16 samples per subset, the subset sequence is good enough to be handed to HDR for more accurate and unbiased estimation. This low requirement of 16 samples per nesting also means that subset simulation is a low-cost preparation for HDR, and causes only minor computational overhead.

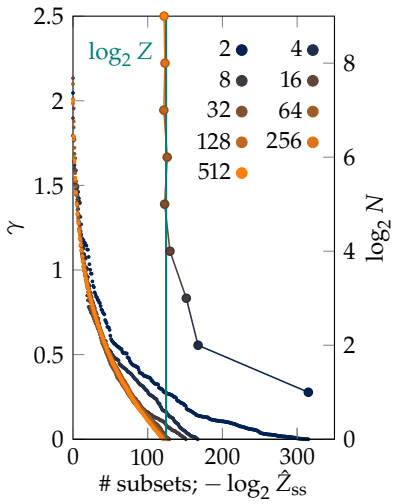


Figure 7.6: Shift values γ against number of subsets T for different sample size per nesting N (small dots). The connected dots show $-\log_2 \hat{Z}_{\text{ss}}$ vs. $\log_2 N$. The ground truth is indicated by the vertical line. This plot emphasizes the connection between T and $-\log_2 Z$ for $\rho = 1/2$ (see text for details).

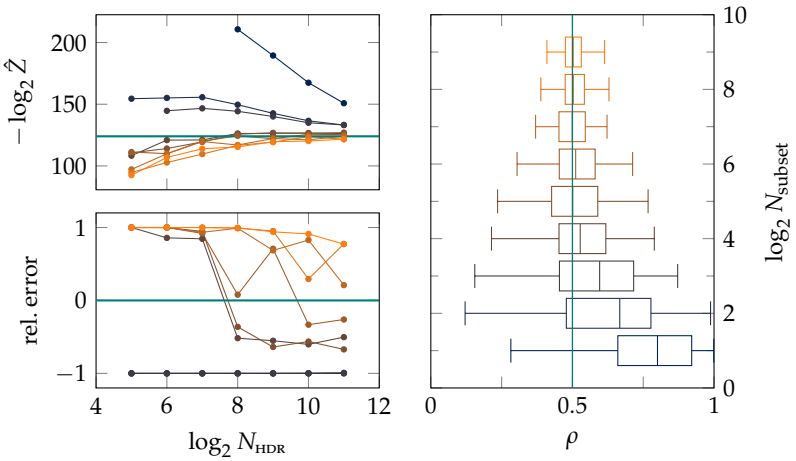


Figure 7.7: *Left:* HDR integral estimates for subset sequences that have been obtained with a varying number of samples between 2^5 to 2^{11} (same color coding as in Figure 7.6). *Left column:* Integral estimate from using different sample sizes for HDR on the different nesting sequences, *top:* evaluated in form of the binary logarithm of the ground truth (horizontal line), and *bottom:* the relative error. *Right:* Conditional probabilities found by HDR with $N_{\text{HDR}} = 2^{11}$ for the subset sequences created from different sample sizes and $\rho = 1/2$ (vertical line) with subset simulation.

Holmes-Diaconis-Ross Figure 7.7 shows the results achieved by HDR for the nine subset sequences obtained with 2^1 to 2^9 samples per subset and for different numbers of samples per nesting for HDR. The top left panel of Figure 7.7 shows the binary logarithm of the HDR integral estimator. The bad performance for the subsets created with 2, 4, or 8 samples

per nesting indicates that a good nesting sequence is essential for the effectiveness of HDR, but also that such a sequence can be found using only about 16 samples per subset (this is thus the number used for all subsequent experiments). The bottom left panel displays the relative error of the HDR estimator. Bear in mind that the relative error is $9/11$ if the estimator is one order of magnitude off, indicating that HDR achieves the right order of magnitude with a relatively low sample demand. The right panel of Figure 7.7 shows the values for the conditional probabilities found by HDR, using 2^{11} samples per subdomain. If subset simulation were perfectly reliable, these should ideally be $\rho = 1/2$. The plot confirms that, with $N \geq 16$, all conditional probabilities found by HDR are far from 0 and 1, warranting the efficiency of HDR. The lowermost box also indicates that HDR corrects for the terrible sequence constructed with only 2 samples per nesting. HDR correctly finds conditional probabilities to be larger than $1/2$, and the integral estimate found by HDR is more accurate than the corresponding estimator by subset simulation.

1000D integrals We further consider three similar synthetic integrals over orthants of 1000D correlated Gaussians with a fixed mean and a randomly drawn covariance matrix. Table 7.1 shows the mean and std. dev. of the binary logarithm of the integral estimator averaged over five runs of HDR using 2^8 samples per nesting for integration, as well as the average CPU time.[6]

Table 7.1: Three linearly constrained integration problems in 1000D

$\langle \log_2 \hat{Z} \rangle$	std. dev.	$t_{\text{CPU}}[10^3\text{s}]$
-162.35	4.27	8.86
-160.54	2.09	7.40
-157.62	3.19	7.64

6: The computations where carried out on 6 CPUs, leading to a wall clock time of about 20 min per run.

7.5.2 Bayesian optimization

Bayesian optimization is a sample-efficient approach to global optimization of expensive-to-evaluate black-box functions [78]. A surrogate over the objective function $f(\boldsymbol{x})$ serves to build a utility function and ultimately derive a policy to determine the next query point. Information-based utilities are directly concerned with the posterior distribution over the minimizer, $p_{\min}(\boldsymbol{x} \,|\, \mathcal{D})$, where $\mathcal{D} = \{\boldsymbol{x}_n, f(\boldsymbol{x}_n)\}_{n=1}^N$ summarizes previous evaluations of f. Entropy search [113] seeks to evaluate the objective function at the location that bears the most information about the minimizer. The expression $p_{\min}(\boldsymbol{x} \,|\, \mathcal{D})$ is an infinite-dimensional integral itself, but for practical purposes, it can be discretized considering the distribution over so-called *representer points*. The probability of the i^{th} representer point to be the minimum can be approximated as

[78]: Garnett (2022), *Bayesian optimization*

[113]: Hennig and Schuler (2012), 'Entropy search for information-efficient global optimization.'

$$\hat{p}_{\min}(\boldsymbol{x}_i) = \int \mathrm{d}\boldsymbol{f}\, \mathcal{N}(\boldsymbol{f}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \prod_{\substack{j=1 \\ j \neq i}}^{N_R} \mathbb{1}[f(\boldsymbol{x}_j) - f(\boldsymbol{x}_i) > 0], \tag{7.8}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the posterior mean and covariance of the Gaussian process over f, respectively. Clearly, this is a linearly constrained Gaussian integral in the form of (2.1) which has to be solved for all N_R representer points. The linear constraints can be rewritten in vectorial form as in (7.4) by introducing the $(N_R - 1) \times N_R$ matrix

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{I}_{(i-1)\times(i-1)} & -\mathbf{1}_{i-1} & \mathbf{0}_{(i-1)\times(N_R-i)} \\ \mathbf{0}_{(N_R-i)\times(i-1)} & -\mathbf{1}_{N_R-i} & \boldsymbol{I}_{(N_R-i)\times(N_R-i)} \end{bmatrix},$$

a $(N_R - 1) \times (N_R - 1)$ identity matrix with a vector of -1s added in the i^{th} column. This allows transforming the linear constraints in (7.8) to a standard normal space

$$\begin{aligned}\hat{p}_{\min}(\boldsymbol{x}_i) &= \int \mathcal{N}(\boldsymbol{f}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \prod_{j \neq i}^{N_R} \mathbb{1}\left[[\boldsymbol{M}\boldsymbol{f}]_j > 0\right] \mathrm{d}\boldsymbol{f} \\ &= \int \mathcal{N}(\boldsymbol{u}, \boldsymbol{0}, \boldsymbol{I}) \prod_{j \neq i}^{N_R} \mathbb{1}\left[\left[\boldsymbol{M}\left(\boldsymbol{\Sigma}^{1/2}\boldsymbol{u} + \boldsymbol{\mu}\right)\right]_j > 0\right] \mathrm{d}\boldsymbol{u},\end{aligned}$$

where we have done the substitution $\boldsymbol{u} = \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{f} - \boldsymbol{\mu})$, and hence $\boldsymbol{f} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{u} + \boldsymbol{\mu}$. The original paper and implementation uses expectation propagation (EP) to approximate this integral.

Probability of minimum For our experiment, we consider the one-dimensional Forrester function [74] with three initial evaluations. The top plot in Figure 7.8 shows the ground truth distribution over the minimum obtained by Thompson sampling, *i.e.*, drawing samples from the discretized posterior Gaussian process (GP) and recording their respective minimum, and the approximation over this distribution obtained by EP. It is apparent that EP fails to accurately represent $\hat{p}_{\min}$. For HDR, we consider four locations (indicated by the vertical lines) and show that while it takes longer to compute, the estimate obtained by HDR converges to the true solution (see bottom plot of Figure 7.8). In the experiment we use 200 representer points—which is an unusually high number for a 1D problem—to show that our method can deal with integrals of that dimension. Also note that we are reporting CPU time, which means that due to automatic parallelization in NUMPY the wall clock time is considerably lower.

[74]: Forrester et al. (2007), 'Multi-fidelity optimization via surrogate modelling'

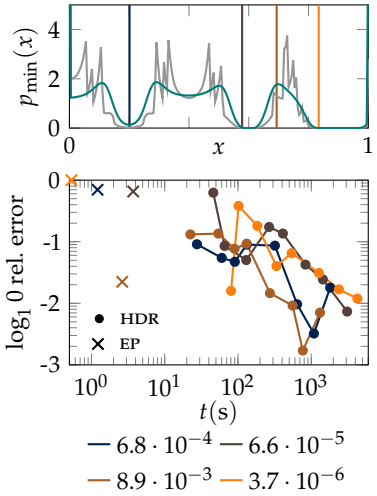


Figure 7.8: *Top:* Probability for x to be the minimum, estimated via Thompson sampling (blue), and EP (gray). Vertical lines indicate locations at which we run HDR. *Bottom:* Absolute relative error by EP and HDR against CPU time at the locations indicated above. Each HDR sequence shown uses 2^6 to 2^{13} samples per nesting. The smaller $\hat{p}_{\min}$, the longer takes the HDR run, since there are more subsets to traverse.

Derivatives Entropy search requires derivatives of (7.8) to construct a first-order approximation of the predictive information gain from evaluating at a new location $\boldsymbol{x}_\star$. We can estimate derivatives using expectations (*cf.* Section 7.4.3 and Appendix D). Initially we choose 5 representer points to validate the approach of computing derivatives via moments against finite differences. The latter requires estimating $\hat{p}_{\min}$ at very high accuracy and has thus a high sample demand even in this low-dimensional setting, for which we employ both rejection sampling and HDR. The derivatives computed via moments from rejection sampling and LIN-ESS take 0.7% of the time required to get a similar accuracy with finite differences. Unsurprisingly, rejection sampling is faster in this case, with $\hat{p}_{\min}(\boldsymbol{x}_i) \approx 1/4$, i.e. only $\sim 3/4$ of the samples from the posterior over f need to be discarded to obtain independent draws that have their minimum at $\boldsymbol{x}_i$. LIN-ESS only outperforms rejection sampling at higher rejection rates common to higher-dimensional problems.
Therefore, we also consider 20 representer points, which corresponds to a 20D linearly constrained space to sample from. In this setting, we consider a location of low probability, with $\hat{p}_{\min} = 1.6 \cdot 10^{-4}$, which renders an estimation via finite differences impossible and highly disfavors rejection sampling even for computing the moments. LIN-ESS, however, enables us to estimate the gradient of the normal distribution w.r.t. its mean and covariance matrix with a relative standard deviation on the 2-norm

of the order of 10^{-2} using $5 \cdot 10^5$ samples and an average CPU time of 325 s for a problem that was previously unfeasible. An ill-conditioned covariance matrix in (7.7) deteriorates runtime (which is already apparent in the considered case) since it requires estimating moments at very high accuracy to compensate for numerical errors.

7.5.3 Constrained samples

We emphasize that LIN-ESS allows to draw samples from linearly constrained Gaussians *without rejection*. In the Gaussian process setting, this permits to efficiently draw samples that are subject to linear restrictions [3, 163, 57]. In particular, the time required for sampling is essentially independent of the probability mass of the domain of interest. This probability mass only affects the pre-computation required to find an initial sample in the domain for LIN-ESS (*cf.* Section 7.4.2). Since this can be achieved with $\sim$16 samples per subset (*cf.* Section 7.5.1), this initial runtime is typically negligible compared to the actual sampling. Figure 7.9 displays the posterior distribution of a GP conditioned on the location of the minimum from the Bayesian optimization context, estimated from LIN-ESS samples. This distribution is required in predictive entropy search [114]—a reformulation of the original entropy search—where it is approximated by imposing several related constraints (*e.g.*, on the derivatives at the minimizer $\boldsymbol{x}_{\min}$). The probability for the given location to be the minimizer is $\lesssim 10^{-6}$, which renders direct sampling virtually impossible. The unaltered ESS algorithm fails on this problem due to the domain selector—a binary likelihood.

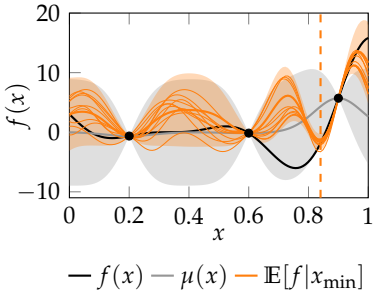


Figure 7.9: The Forrester function (black), the posterior GP given three evaluations (gray), and the posterior distribution over f conditioned on the minimum being located at where the vertical line indicates (- - -), each with the 2σ confidence interval shaded. The latter has been obtained from drawing 10^5 samples using LIN-ESS, 10 of which are shown (——).

7.6 Conclusions

This chapter focused on a ubiquitous and notoriously hard integration problem: the estimation of Gaussian probabilities. The step functions implied by linear constraints render this problem unsuitable for Bayesian quadrature. In this case, Monte Carlo methods come to the rescue: We have introduced a black-box algorithm that computes these integrals over linearly constrained Gaussian densities with high numerical precision, even if the integration domain is of high dimensionality and the probability to be computed is very small. This was achieved by adapting two separate pieces of existing prior art and carefully matching them to the problem domain: We designed a special version of elliptical slice sampling that takes explicit advantage of the linearly-constrained Gaussian setting, and used it as an internal step of the HDR algorithm. We showed that, because this algorithm can not just compute integrals but also produces samples from the nestings alongside, it also permits the evaluation of derivatives of the integral with respect to the parameters of the measure. One current limitation is that, because our algorithm was designed to be unbiased, it has comparably high computational cost (but also superior numerical precision) over alternatives like expectation propagation. This problem could be mitigated if one is willing to accept unbiasedness and thus reuse samples. Furthermore, both HDR and LIN-ESS are highly parallelizable (as opposed to EP) and thus offer a straightforward implementational improvement for application on modern hardware.

[3]: Agrell (2019), 'Gaussian Processes with linear operator inequality constraints'

[163]: López-Lopera et al. (2018), 'Finite-dimensional Gaussian approximation with linear inequality constraints'

[57]: Da Veiga and Marrel (2012), 'Gaussian process modeling with inequality constraints'

[114]: Hernández-Lobato et al. (2014), 'Predictive entropy search for efficient global optimization of black-box functions'

Epilogue

Conclusion 8

After a brief summary of the main contributions of this thesis, we outline some of the challenges that probabilistic integration methods hinder from more widespread adoption. We then provide an outlook on future research topics that might extend the practical scope probabilistic numerical methods for integration.

8.1 Summary and discussion

The main topic of this thesis is numerical integration as a tool in approximate Bayesian inference. Motivated by the probabilistic formulation of numerical methods, the research conducted as a part of this work was centered around Bayesian quadrature (BQ). A main aspect was the identification and demonstration of the scope of probabilistic integration methods.

8.1.1 The scope of Bayesian quadrature

Active Bayesian quadrature Chapter 3 and 5 highlight active learning as a benefit that the Bayesian formulation of a numerical method naturally entails. This not only permits the construction of integration schemes that adapt to the integrand, but also unlock application areas that are out of reach for classical integration methods. One such domain is transfer learning, in which information retrieved from a correlated function can be carried over to the main problem of interest. This setting is scrutinized with regard to active learning schemes in Chapter 5 when information is not the only quantity optimized for, but when there is also a trade-off with evaluation cost of multiple present information sources. We found that this extension lifts the degeneracy between commonly used active learning policies and observed properties of acquisition functions that prevent such a policy from producing pathological strategies.

BQ for expensive integrands Chapter 5 and 6 take up the theme of employing BQ to computational settings in which integrals of expensive-to-evaluate functions appear. While Chapter 5 deals with retrieving information from cheaper sources to save cost, the considered application in Chapter 6 are integrals on Riemannian manifolds that occur when learning normal distributions that respect the intrinsic geometry of given data. Integrand evaluations require solving the geodesic equations and are thus moderately expensive; a setting where BQ demonstrates clear superiority over simple Monte Carlo. In particular, the need for repeated computation of similar integrals entails tremendous savings over the entire procedure of learning a locally adaptive normal distribution.

8.1.2 Challenges

Not every integration problem is suitable to be solved with Bayesian quadrature and theoretical as well as practical challenges of the method remain.

Priors for BQ The main restriction when choosing a GP prior (*i.e.*, a kernel) is its integrability w.r.t. the integration measure. The kernel mean (2.15) and initial variance (2.16)

$$\kappa(\boldsymbol{x}) := \int_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\nu(\boldsymbol{x}'),$$
$$\mathfrak{k} := \iint_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}') \, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}')$$

are integrals that only have closed-form expressions for a small amount of combinations of (mostly stationary) kernels and measures (*cf.* Table 2.1). Surrogate models over integrands are thus severely limited by the availability of integrable kernels to avoid having to use another level of numerical integration.[1]
Besides practical constraints imposed by integrability of the kernel, the choice of a suitable prior for a given integrand is not usually theoretically founded. Even if properties of the integrand are known, it is rarely possible to inspect whether it is a member of a certain RKHS. As a result, prior choice ends up being an empirical procedure in practice, and recommendations are to use kernels that 'behave well' empirically on a wide range of problems.

1: Numerical integration of the surrogate model can be a desirable option when an emulator is already in use for other purposes and/or cheaper to integrate numerically than its underlying ground truth. However, this approach introduces a second layer of numerical approximation and will introduce error on the BQ posterior distribution.

Uncertainty calibration The potential danger of model misspecifiction raises the question whether the uncertainty predicted by the BQ methods is a well-calibrated representation of numerical error. A key ingredient to calibration is hyperparameter optimization that adapts the model to the typical scales of the problem at hand.

Computational cost A widely criticized limitation of Gaussian processes and consequentially BQ is the cubic computational cost of inference in the number of quadrature nodes. This scaling compares unfavorably with the low overhead of competitors such as Monte Carlo methods. Additional cost arises from intermediate optimization during model adaptation and active learning. This limitation confines BQ to problems where benefits of the probabilistic approach outweigh the computational investment (*cf.* Section 8.1.3).

Curse of dimensionality A strong prior can permit BQ to extend beyond the tight dimensionality constraints of classical quadrature methods. Still, Gaussian processes do not scale well to high-dimensional input and dimensionality compromises convergence rates of BQ methods (*cf.* Section 2.3.4).

Globality of quadrature On top of the issue of dimensionality, the fact that integration is a global operation exacerbates the quadrature task. A surrogate needs to be learned well everywhere as deviations enter additively into the integral estimate. This property of integration renders BQ a lot more intricate than for instance Bayesian optimization that only requires a good surrogate locally.

The identification of key challenges for probabilistic integration motivated Chapter 7 as a tangential project that standard BQ is utterly unsuitable for: The computation of multivariate normal probabilities. These entail integrating correlated Gaussians over a box or half-bounded domain bounded through linear constraints. The linear constraints compromise the tractability of the kernel integrals required in BQ, which is therefore stuck itself with precisely the kind of problem that one is trying to model. Instead, we designed a custom MCMC method for estimating the mass and sampling from such a domain. The particular setting permits rejection-free sampling that at the backend of multilevel splitting methods enables efficient estimation even of extremely small probability masses.

8.1.3 A checklist for BQ

To guide a potential user, we have compiled a checklist that helps decide whether BQ might be a viable method for a given problem. As a rule of thumb, integration problems with any of the following properties tend to be auspicious for BQ:

- The function to be integrated is at least moderately expensive to evaluate,[2] such that function queries amortize the cost of inference.
- The integrand has well-known properties such as smoothness or a known number of available derivatives that can naturally be encoded in a Gaussian process prior through the choice of kernel.
- The integrand has properties such as positivity that can be encoded in the prior via a transformation.
- Similar integration problems co-occur, *e.g.*, when an integral has to be computed repeatedly in a loop.
- Uncertainty over the outcome of the computation is essential for consequential decision-making down the line.

2: To put numbers on 'moderately expensive', Chapter 6 already saw considerable gains over simple Monte Carlo for evaluations that take of the order of tens of milliseconds (*cf.* Table 6.1) in low-dimensional settings with $D \leq 5$.

If the integration is further over a low to moderate number of correlated variables, the considered problem falls well into the scope of BQ.

8.2 Outlook and future work

There are numerous conceivable directions to increase the scope of probabilistic numerical integration methods. The following ideas may serve as starting point for future work and inspiration for research projects.

Implementation To date, BQ has reached the maturity that it could be an off-the-shelf integration method whenever uncertainty over the output is essential. What has restrained BQ from reaching this point are to a certain extent the above-named challenges as well as limited dissemination. Yet first and foremost, it is the lack of a comprehensive implementation that facilitate adoption by non-expert users that has prevented BQ from overcoming its niche rôle amongst methods for numerical integration. The recent collaborative effort to join implementations of probabilistic numerical methods in one toolbox, `probnum`, is a step in the right direction. The `quad` package within `probnum` is yet in its infancy and awaits further contribution by the BQ community.

Benchmarks Beyond pure implementation, an extensive empirical study will provide empirical clarity about the practical scope of BQ on real-world integration problems. Most surveys, such as [37] have been studying the performance of BQ from a theoretical perspective. A large scale benchmarking study would permit framing the guidelines introduced above in a more quantitative manner and sharpen the limitations of BQ.

[37]: Briol et al. (2019), 'Probabilistic integration: A role in statistical computation?'

Dimensionality reduction Contemporary inference problems often deal with a large quantity of correlated random variables. High-dimensional numerical integration is challenging for BQ due to the vast volume that needs to be explored. A required assumption to deal with high-dimensional problems is that they are not truly high-dimensional, but typically live on a lower-dimensional manifold that is embedded into the high-dimensional space [159]. Other domains of machine learning employ *dimensionality reduction* techniques to find lower-dinmensional representations with small loss of information. Such techniques have not yet been applied in the context of BQ. In the simplest, linear case, this would permit decoupling directions of low variation to be left with a product of lower-dimensional integrals that BQ can solve. An evident challenge is quantifying the uncertainty introduced by discarding correlations of neglected variables.

[159]: Levina and Bickel (2005), 'Maximum likelihood estimation of intrinsic dimension'

Uncertainty propagation Ideally, Bayesian quadrature is only one element in a computational pipeline in which every step introduces new numerical uncertainty. An example is the LAND,[3] where integration hinges on the solution of initial or boundary value problems. In such settings, chaining probabilistic numerical methods will be desirable to achieve a meaningful uncertainty estimate of a sequence of numerical problems. Most methods to date, however, deal with *outputting* probability distributions, and there is little work on how these methods should process probability distributions as *inputs*. Non-linearity renders the propagation of uncertainty a challenging problem across sub-disciplines of probabilistic numerics.

3: Local adaptive normal distribution, introduced in Chapter 6

A vision for quadrature

The holy grail of numerical integration is the full automation of quadrature. Neither should the user have to invest manual effort and time to select a specialized integration scheme that is suitable for their particular problem at hand, nor should they need to resort to overly generic algorithms that are applicable across a wide range of integration tasks, but at the detriment of performance. To date, the function passed by the user is usually treated as a black-box by the numerical integration method and adaptation is achieved merely—if at all—through interaction with the black-box. Automizing the selection of an integration scheme demands opening the black-box and parsing its properties to attack the meta-decision problem of choosing an appropriate method. Within probabilistic integration schemes, this would amount to automizing the selection of a prior and could be phrased as a Bayesian model selection problem [170, 45]. The extension to non-probabilistic or stochastic methods is a tougher nut to crack but of pragmatic relevance in the endeavor of automizing quadrature.

[170]: Malkomes et al. (2016), 'Bayesian optimization for automated model selection'

[45]: Chai et al. (2019), 'Automated model selection with Bayesian quadrature'

Appendix

Useful Identities

A

A.1 Relevant matrix identities

We summarize here matrix identities used in the main text. A more complete exposition is found in [202].

[202]: Petersen, Pedersen, et al. (2008), 'The matrix cookbook'

Define

$$\begin{aligned} \boldsymbol{C}_1 &= \boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21} \\ \boldsymbol{C}_2 &= \boldsymbol{A}_{22} - \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12} \end{aligned}$$

Block matrix inversion

$$\begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{pmatrix}^{-1} = \begin{pmatrix} \boldsymbol{C}_1^{-1} & -\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\boldsymbol{C}_2^{-1} \\ -\boldsymbol{C}_2^{-1}\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1} & \boldsymbol{C}_2^{-1} \end{pmatrix}^{-1} \tag{A.1}$$

Matrices that can be written as $(\boldsymbol{A} - \boldsymbol{U}\boldsymbol{B}\boldsymbol{V}^\top)$ with $\boldsymbol{A}, \boldsymbol{B}$ square matrices of size $N \times N$ and $M \times M$, respectively, and $\boldsymbol{U}, \boldsymbol{V}$ of size $N \times M$ can be inverted using the Woodbury inversion lemma

$$(\boldsymbol{A} - \boldsymbol{U}\boldsymbol{B}\boldsymbol{V}^\top)^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}(\boldsymbol{B}^{-1} + \boldsymbol{V}^\top\boldsymbol{A}^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}^\top\boldsymbol{A}^{-1}.$$

The determinant of such a matrix can be written as

$$\det(\boldsymbol{A} - \boldsymbol{U}\boldsymbol{B}\boldsymbol{V}^\top) = \det(\boldsymbol{A})\det(\boldsymbol{B})\det(\boldsymbol{B}^{-1} + \boldsymbol{V}^\top\boldsymbol{A}^{-1}\boldsymbol{U}). \tag{A.2}$$

The determinant of a matrix written in terms of submatrices can be written in terms of determinants of these blocks,

$$\begin{aligned} \det\begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{pmatrix} &= \det \boldsymbol{A}_{22} \det \boldsymbol{C}_1 \\ &= \det \boldsymbol{A}_{11} \det \boldsymbol{C}_2. \end{aligned} \tag{A.3}$$

A.2 Gaussian identities

Consider a D-dimensional Gaussian distribution,

$$\mathcal{N}(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{A}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{A}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{a})^\top \boldsymbol{A}^{-1}(\boldsymbol{x} - \boldsymbol{a})\right).$$

A product of N Gaussian densities can be written as one term that depends on $\boldsymbol{x}$ and $N - 1$ Gaussian terms that do not depend on $\boldsymbol{x}$,

$$\begin{aligned} \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}, \boldsymbol{a}_i, \boldsymbol{A}_i) =& \mathcal{N}\left(\boldsymbol{x}, \left(\sum_{i=1}^{N} \boldsymbol{A}_i^{-1}\right)^{-1} \left(\sum_{i=1}^{N} \boldsymbol{A}_i^{-1}\boldsymbol{a}_i\right), \left(\sum_{i=1}^{N} \boldsymbol{A}_i^{-1}\right)^{-1}\right) \\ &\times \prod_{j=1}^{N} \mathcal{N}\left(\boldsymbol{a}_j, \left(\sum_{i=1}^{j-1} \boldsymbol{A}_i^{-1}\right)^{-1} \left(\sum_{i=1}^{j-1} \boldsymbol{A}_i^{-1}\boldsymbol{a}_i\right), \boldsymbol{A}_j + \left(\sum_{i=1}^{j-1} \boldsymbol{A}_i^{-1}\right)^{-1}\right) \end{aligned} \tag{A.4}$$

From this expression follows for example that

$$\mathcal{N}(\boldsymbol{x};\boldsymbol{a},\boldsymbol{A})^2 = \pi^{-D/2}|\boldsymbol{A}|^{-1/2}\mathcal{N}\left(\boldsymbol{x};\boldsymbol{a};\frac{\boldsymbol{A}}{2}\right). \tag{A.5}$$

Derivations Related to Bayesian Quadrature

B

B.1 Kernel embeddings for a Gaussian kernel and Gaussian measure

We include an exemplary derivation of analytic expressions for integrals that arise in BQ. A compilation of kernel embeddings, *i.e.*, solutions to BQ integrals for known pairs of kernels and integration measures is in preparation.[1] Let $\mathcal{X} = \mathbb{R}^d$ and consider the case where the kernel is Gaussian (aka. exponentiated quadratic) (2.9) with p.s.d. lengthscale matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{D\times D}$ and the integration measure is Gaussian with mean $\boldsymbol{\mu} \in \mathbb{R}^D$ and covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{D\times D}$. We restate the expressions for reference and rewrite the Gaussian kernel in terms of the normal distribution as

1: as joint work with T. Karvonen, M. Mahsereci, and F.-X. Briol

$$k_{\boldsymbol{\Lambda}}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^\top \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}-\boldsymbol{x}')\right) = (2\pi)^{D/2}|\boldsymbol{\Lambda}|^{1/2}\mathcal{N}(\boldsymbol{x}; \boldsymbol{x}', \boldsymbol{\Lambda}),$$

$$\nu(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

The integrals that arise can be solved using the identity for the product of Gaussian density functions (A.4) to be left with one Gaussian to be integrated over, and then exploiting that the Gaussian density integrates to 1.

B.1.1 Integrals for VBQ

In VBQ, the two integrals to be solved are the kernel mean (2.15) and the initial variance (2.16). The kernel mean is

$$\begin{aligned}\kappa(\boldsymbol{x}) &:= \int_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}')\,\mathrm{d}\nu(\boldsymbol{x}') \\ &= (2\pi)^{D/2}|\boldsymbol{\Lambda}|^{1/2}\int_{\mathcal{X}} \mathcal{N}(\boldsymbol{x}'; \boldsymbol{x}, \boldsymbol{\Lambda})\,\mathcal{N}(\boldsymbol{x}'; \boldsymbol{\mu}, \boldsymbol{\Sigma})\,\mathrm{d}\boldsymbol{x}' \\ &\overset{(A.4)}{=} (2\pi)^{D/2}|\boldsymbol{\Lambda}|^{1/2}\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}+\boldsymbol{\Lambda}),\end{aligned}$$

and the initial variance

$$\begin{aligned}\mathfrak{k} &:= \iint_{\mathcal{X}} k(\boldsymbol{x}, \boldsymbol{x}')\,\mathrm{d}\nu(\boldsymbol{x})\,\mathrm{d}\nu(\boldsymbol{x}') \\ &= \int_{\mathcal{X}} \kappa(\boldsymbol{x})\,\mathrm{d}\nu(\boldsymbol{x}) \\ &= (2\pi)^{D/2}|\boldsymbol{\Lambda}|^{1/2}\int_{\mathcal{X}} \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}+\boldsymbol{\Lambda})\,\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})\,\mathrm{d}\boldsymbol{x} \\ &\overset{(A.4)}{=} \left(\frac{|\boldsymbol{\Lambda}|}{|2\boldsymbol{\Sigma}+\boldsymbol{\Lambda}|}\right)^{1/2} = |\boldsymbol{I}+\boldsymbol{\Sigma}\boldsymbol{\Lambda}^{-1}|^{-1/2}\end{aligned}$$

B.1.2 Integrals for warped BQ (square transform)

The derivations rely on the same principles when considering the integrals (2.30) that appear when approximating the square transform in warped BQ, but they get more tedious. The required integrals are

$$\begin{aligned}\int_{\mathcal{X}} k(\boldsymbol{x},\boldsymbol{x}_i)k(\boldsymbol{x},\boldsymbol{x}_j)\,\mathrm{d}\nu(\boldsymbol{x}) &= (2\pi)^D|\boldsymbol{\Lambda}| \int_{\mathcal{X}} \mathcal{N}(\boldsymbol{x};\boldsymbol{x}_i,\boldsymbol{\Lambda})\,\mathcal{N}(\boldsymbol{x};\boldsymbol{x}_j,\boldsymbol{\Lambda})\,\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma})\,\mathrm{d}\boldsymbol{x}\\ &\overset{(A.4)}{=} (2\pi)^D|\boldsymbol{\Lambda}|\,\mathcal{N}\big(\boldsymbol{\mu};\frac{\boldsymbol{x}_i+\boldsymbol{x}_j}{2},\boldsymbol{\Sigma}+\frac{\boldsymbol{\Lambda}}{2}\big)\,\mathcal{N}(\boldsymbol{x}_i;\boldsymbol{x}_j,2\boldsymbol{\Lambda}),\end{aligned}$$

the double integral over the squared kernel

$$\begin{aligned}\iint_{\mathcal{X}} k(\boldsymbol{x},\boldsymbol{x}')^2\,\mathrm{d}\nu(\boldsymbol{x})\mathrm{d}\nu(\boldsymbol{x}') &\overset{(A.5)}{=} \pi^{D/2}|\boldsymbol{\Lambda}|^{1/2} \iint_{\mathcal{X}} \mathcal{N}\big(\boldsymbol{x};\boldsymbol{x}',\frac{\boldsymbol{\Lambda}}{2}\big)\,\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma})\,\mathcal{N}(\boldsymbol{x}';\boldsymbol{\mu},\boldsymbol{\Sigma})\,\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}',\\ &\overset{(A.4)}{=} \pi^{D/2}|\boldsymbol{\Lambda}|^{1/2} \int_{\mathcal{X}} \mathcal{N}\big(\boldsymbol{x}';\boldsymbol{\mu},\boldsymbol{\Sigma}+\frac{\boldsymbol{\Lambda}}{2}\big)\,\mathcal{N}(\boldsymbol{x}';\boldsymbol{\mu},\boldsymbol{\Sigma})\,\mathrm{d}\boldsymbol{x}'\\ &\overset{(A.4)}{=} \pi^{D/2}|\boldsymbol{\Lambda}|^{1/2}\,(2\pi)^{-D/2}\left|2\boldsymbol{\Sigma}+\frac{\boldsymbol{\Lambda}}{2}\right|\\ &= |4\boldsymbol{\Lambda}^{-1}\boldsymbol{\Sigma}+\boldsymbol{I}|^{-1/2},\end{aligned}$$

and lastly, the double integral

$$\begin{aligned}&\iint_{\mathcal{X}} k(\boldsymbol{x},\boldsymbol{x}_i)k(\boldsymbol{x},\boldsymbol{x}')k(\boldsymbol{x}',\boldsymbol{x}_j)\,\mathrm{d}\nu(\boldsymbol{x})\mathrm{d}\nu(\boldsymbol{x}')\\ &\overset{(A.4)}{=} (2\pi)^{3D/2}|\boldsymbol{\Lambda}|^{3/2}\mathcal{N}(\boldsymbol{x}_i;\boldsymbol{\mu},\boldsymbol{\Lambda}+\boldsymbol{\Sigma})\\ &\qquad \int_{\mathcal{X}} \mathcal{N}\big(\boldsymbol{x}';(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}(\boldsymbol{\Lambda}^{-1}\boldsymbol{x}_i+\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}),(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}\big)\mathcal{N}(\boldsymbol{x}';\boldsymbol{x}_j,\boldsymbol{\Lambda})\mathcal{N}(\boldsymbol{x}';\boldsymbol{\mu},\boldsymbol{\Sigma})\,\mathrm{d}\boldsymbol{x}'\\ &\overset{(A.4)}{=} (2\pi)^{3D/2}|\boldsymbol{\Lambda}|^{3/2}\mathcal{N}(\boldsymbol{x}_i;\boldsymbol{\mu},\boldsymbol{\Lambda}+\boldsymbol{\Sigma})\,\mathcal{N}(\boldsymbol{x}_j;\boldsymbol{\mu},\boldsymbol{\Lambda}+\boldsymbol{\Sigma})\\ &\qquad \mathcal{N}\big((\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}(\boldsymbol{\Lambda}^{-1}\boldsymbol{x}_i+\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu});(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}(\boldsymbol{\Lambda}^{-1}\boldsymbol{x}_j,+\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}),2(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}\big)\\ &= (2\pi)^{3D/2}|\boldsymbol{\Lambda}|^{3/2}\,|\boldsymbol{I}+\boldsymbol{\Lambda}\boldsymbol{\Sigma}^{-1}|\,\mathcal{N}(\boldsymbol{x}_i;\boldsymbol{\mu},\boldsymbol{\Lambda}+\boldsymbol{\Sigma})\,\mathcal{N}(\boldsymbol{x}_j;\boldsymbol{\mu},\boldsymbol{\Lambda}+\boldsymbol{\Sigma})\,\mathcal{N}\big(\boldsymbol{x}_i;\boldsymbol{x}_j,2(\boldsymbol{\Lambda}+\boldsymbol{\Lambda}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda})\big).\end{aligned}$$

For the last step we have written out the exponent of the last Gaussian to simplify (omitting the factor of $-1/2$)

$$\begin{aligned}&\big((\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}(\boldsymbol{\Lambda}^{-1}\boldsymbol{x}_i+\cancel{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}}-\boldsymbol{\Lambda}^{-1}\boldsymbol{x}_j-\cancel{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}})\big)^\top\frac{1}{2}(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})\big((\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}(\boldsymbol{\Lambda}^{-1}(\boldsymbol{x}_i-\boldsymbol{x}_j))\big)\\ &= (\boldsymbol{\Lambda}^{-1}(\boldsymbol{x}_i-\boldsymbol{x}_j))^\top(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}\frac{1}{2}\cancel{(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})(\boldsymbol{\Lambda}^{-1}+\boldsymbol{\Sigma}^{-1})^{-1}}(\boldsymbol{\Lambda}^{-1}(\boldsymbol{x}_i-\boldsymbol{x}_j))\\ &= \frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{x}_j)^\top(\boldsymbol{\Lambda}+\boldsymbol{\Lambda}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda})^{-1}(\boldsymbol{x}_i-\boldsymbol{x}_j).\end{aligned}$$

Furthermore, the factor $|\boldsymbol{I}+\boldsymbol{\Lambda}\boldsymbol{\Sigma}^{-1}|$ appears when rewriting the Gaussian, as care has to be taken of the normalization constant.

For stationary kernels, the remaining kernel integral in (2.30) is easy to solve: $k(\boldsymbol{x},\boldsymbol{x})$ integrates to 1 against measures that integrate to 1.

B.2 Optimal design for BQ

We derive here the acquisition functions from Section 3.2 and motivate the definition of the canonical squared correlation (3.5).

B.2.1 Derivation of integral variance reduction

We start with the derivation of the integral variance reduction (IVR) acquisition. Define the Gram matrix including noise as $\boldsymbol{G} = \boldsymbol{K} + \sigma^2 \boldsymbol{I}$. Also note that for better readability, we sometimes write the posterior variance of the integral as $\mathfrak{v}_{\mathcal{D}} = \mathbb{V}[\mathrm{Z} \mid \mathcal{D}]$; both notations are equivalent. The difference in variance of the integral over two consecutive updates is

$$
\begin{aligned}
\Delta\mathbb{V} &= \mathbb{V}[\mathrm{Z} \mid \mathcal{D}] - \mathbb{V}[\mathrm{Z} \mid \mathcal{D} \cup \{\boldsymbol{X}_\star, \boldsymbol{y}_\star\}] \\
&= \boldsymbol{\kappa}(\boldsymbol{X} \cup \boldsymbol{X}_\star)^\top \boldsymbol{G}_{\boldsymbol{X} \cup \boldsymbol{X}_\star}^{-1} \boldsymbol{\kappa}(\boldsymbol{X} \cup \boldsymbol{X}_\star) \; - \; \boldsymbol{\kappa}^\top \boldsymbol{G}^{-1} \boldsymbol{\kappa} \\
&= \begin{bmatrix} \boldsymbol{\kappa} \\ \boldsymbol{\kappa}_\star \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{G} & k(\boldsymbol{X}, \boldsymbol{X}_\star) \\ k(\boldsymbol{X}_\star, \boldsymbol{X}) & \boldsymbol{G}_\star \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\kappa} \\ \boldsymbol{\kappa}_\star \end{bmatrix} - \boldsymbol{\kappa}^\top \boldsymbol{G}^{-1} \boldsymbol{\kappa} \\
&\overset{(??)}{=} \begin{bmatrix} \boldsymbol{\kappa} \\ \boldsymbol{\kappa}_\star \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} k(\boldsymbol{X}_\star, \boldsymbol{X}) \boldsymbol{G}^{-1} & -\boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} \\ -\boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} k(\boldsymbol{X}_\star, \boldsymbol{X}) \boldsymbol{G}^{-1} & \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{\kappa} \\ \boldsymbol{\kappa}_\star \end{bmatrix} \\
&= \boldsymbol{\kappa} \left(\boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} k(\boldsymbol{X}_\star, \boldsymbol{X}) \boldsymbol{G}^{-1} \right) \boldsymbol{\kappa} - \boldsymbol{\kappa} \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} \boldsymbol{\kappa}_\star - \boldsymbol{\kappa}_\star \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} k(\boldsymbol{X}_\star, \boldsymbol{X}) \boldsymbol{G}^{-1} \boldsymbol{\kappa} + \boldsymbol{\kappa}_\star \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} \boldsymbol{\kappa}_\star \\
&= \left(\boldsymbol{\kappa}_\star - \boldsymbol{\kappa} \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) \right) \boldsymbol{C}_{\star \mid \mathcal{D}}^{-1} \left(\boldsymbol{\kappa}_\star - \boldsymbol{\kappa} \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) \right) \\
&= \mathbb{V}[\mathrm{Z} \mid \mathcal{D}] \, \rho_{\mathcal{D}}^2(\boldsymbol{X}_\star)
\end{aligned}
$$

with the noise-corrected posterior covariance

$$
\boldsymbol{C}_{\star \mid \mathcal{D}} = \boldsymbol{K}_\star - k(\boldsymbol{X}_\star, \boldsymbol{X}) \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) + \sigma^2 \boldsymbol{I}.
$$

The other variance-based versions (NIV, IP, IPI) follow straightforwardly from the IVR acquisition function.

B.2.2 Derivation of mutual information

In standard BQ, the joint distribution of the integral Z and any prospective new data points $\mathbf{y}_\star$ after having observed data $\mathcal{D}$ is

$$
p(Z, \boldsymbol{y}_\star) = \mathcal{N}\left(\begin{bmatrix} Z \\ \boldsymbol{y}_\star \end{bmatrix}; \begin{bmatrix} \mathfrak{m}_{\mathcal{D}} \\ \boldsymbol{m}_{\mathcal{D}}(\boldsymbol{X}_\star) \end{bmatrix}, \begin{bmatrix} \mathfrak{v}_{\mathcal{D}} & \boldsymbol{\kappa}_{\star \mid \mathcal{D}}^\top \\ \boldsymbol{\kappa}_{\star \mid \mathcal{D}} & \boldsymbol{C}_{\star \mid \mathcal{D}} \end{bmatrix} \right) \tag{B.1}
$$

The (almost) properly expanded covariance matrix in (B.1) is

$$
\mathbb{C}[\mathrm{Z}, \mathbf{y}_\star] = \begin{bmatrix} \iint_{\mathcal{X}} (k(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{x}, \boldsymbol{X}) \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{x}')) \, \mathrm{d}\nu(\boldsymbol{x}) \mathrm{d}\nu(\boldsymbol{x}') & \int_{\mathcal{X}} (k(\boldsymbol{x}, \boldsymbol{X}_\star) - k(\boldsymbol{x}, \boldsymbol{X}) \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star)) \, \mathrm{d}\nu(\boldsymbol{x}) \\ \int_{\mathcal{X}} (k(\boldsymbol{X}_\star, \boldsymbol{x}) - k(\boldsymbol{X}_\star, \boldsymbol{X}) \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{x})) \, \mathrm{d}\nu(\boldsymbol{x}) & \boldsymbol{K}_\star - k(\boldsymbol{X}_\star, \boldsymbol{X}) \boldsymbol{G}^{-1} k(\boldsymbol{X}, \boldsymbol{X}_\star) + \sigma^2 \boldsymbol{I} \end{bmatrix},
$$

so $\boldsymbol{\kappa}_{\star \mid \mathcal{D}}^\top$ denotes the posterior kernel mean at locations $\boldsymbol{X}_\star$.

The mutual information between Z and $\mathbf{y}_\star$ can be written in terms of entropy as

$$
I[\mathrm{Z}; \mathbf{y}_\star] = H[\mathrm{Z}] + H[\mathbf{y}_\star] - H[\mathrm{Z}, \mathbf{y}_\star]. \tag{B.2}
$$

The entropy of a D-dimensional multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is

$$
H = \frac{D}{2} \log(2\pi e) + \frac{1}{2} \log \det \boldsymbol{\Sigma}.
$$

The entropy is independent of the mean, and therefore in GPS, independent of the observations, since function evaluations do not enter

the posterior covariance. Plugging (B.1) into the expression for the entropy and applying the block matrix determinant rule (A.3), the mutual information (B.2) turns into

$$I[\mathsf{Z}, \mathbf{y}_\star] = \frac{1 + N_\star - (N_\star + 1)}{2} \log(2\pi e) + \frac{1}{2}\left(\log \mathfrak{v}_\mathcal{D} + \log\det \boldsymbol{C}_{\star\,|\,\mathcal{D}} - \log\left(\mathfrak{v}_\mathcal{D} \det \boldsymbol{C}_{\star\,|\,\mathcal{D}} - \boldsymbol{\kappa}_{\star\,|\,\mathcal{D}}^\top \boldsymbol{C}_{\star\,|\,\mathcal{D}}^{-1} \boldsymbol{\kappa}_{\star\,|\,\mathcal{D}}\right)\right)$$

$$\begin{aligned} &= -\frac{1}{2}\log\left(1 - \frac{\boldsymbol{\kappa}_{\star\,|\,\mathcal{D}}^\top \boldsymbol{C}_{\star\,|\,\mathcal{D}}^{-1} \boldsymbol{\kappa}_{\star\,|\,\mathcal{D}}}{\mathfrak{v}_\mathcal{D}}\right) \\ &= -\frac{1}{2}\log\left(1 - \rho^2_\mathcal{D}(\boldsymbol{X}_\star)\right), \end{aligned}$$

using the definition of $\rho^2_\mathcal{D}$ (3.5).

B.2.3 Summary of the acquisition functions

The following table summarizes the functional form of the optimal acquisitions:

Table B.1: Functional form of optimal BQ acquisition schemes.

acronym	name	origin	$\alpha(\rho)$	sanity
MI	mutual information	$I[\mathsf{Z}; \mathbf{y}_\star]$	$-1/2 \log(1-\rho^2)$	✓
IVR	integral variance reduction	$\frac{\mathbb{V}[\mathsf{Z}\,\vert\,\mathcal{D}] - \mathbb{V}[\mathsf{Z}\,\vert\,\mathcal{D}\cup\mathcal{D}_\star]}{\mathbb{V}[\mathsf{Z}\,\vert\,\mathcal{D}]}$	ρ^2	✓
IPI	integral precision increase	$\frac{\mathbb{V}^{-1}[\mathsf{Z}\,\vert\,\mathcal{D}\cup\mathcal{D}_\star] - \mathbb{V}^{-1}[\mathsf{Z}\,\vert\,\mathcal{D}]}{\mathbb{V}^{-1}[\mathsf{Z}\,\vert\,\mathcal{D}]}$	$\frac{\rho^2}{1-\rho^2}$	✓
IP	integral precision	$\frac{\mathbb{V}^{-1}[\mathsf{Z}\,\vert\,\mathcal{D}\cup\mathcal{D}_\star]}{\mathbb{V}^{-1}[\mathsf{Z}\,\vert\,\mathcal{D}]}$	$\frac{1}{1-\rho^2}$	✗
NIV	negative integral variance	$-\frac{\mathbb{V}[\mathsf{Z}\,\vert\,\mathcal{D}\cup\mathcal{D}_\star]}{\mathbb{V}[\mathsf{Z}\,\vert\,\mathcal{D}]}$	$\rho^2 - 1$	✗

Details on Bayesian Quadrature on Riemannian Manifolds

C

The additional details about Chapter 6 have been prepared by Christian Fröhlich, except for parts on the DCV acquisition function.

C.1 Aspects of Riemannian geometry

C.1.1 Geodesic equations

The energy or action functional of a curve γ with time derivative $\dot{\gamma}(t)$ is defined as

$$E(\gamma) = \frac{1}{2}\int_0^1 \underbrace{\langle \dot{\gamma}(t), \boldsymbol{M}(\gamma(t))\dot{\gamma}(t)\rangle}_{=:\mathscr{L}} \,\mathrm{d}t.$$

In physics, the argument of the integral is known as *Lagrangian* and we therefore abbreviate the inner product as $\mathscr{L} := \langle \dot{\gamma}(t), \boldsymbol{M}(\gamma(t))\dot{\gamma}(t)\rangle$. Geodesics are the stationary curves of this functional which are solutions to the Euler-Lagrange equations. We are interested in the minimizers, i.e., shortest paths. Minimizing curve energy instead of length avoids the issue of arbitrary reparameterization. Let γ^d denote the d-th coordinate of the curve γ at time t and $M_{dd'}$ the metric component at row d and column d', if it is represented as a matrix. We leave sums over repeated indices implicit (Einstein summation convention). Energy-minimizing paths can be found by applying the Euler-Lagrange equations to the functional E and solving for the curve $\gamma \in \mathbb{R}^D$. They give rise to a system of D coupled, 2^{nd} order differential equations, the d^{th} component of which reads

$$\frac{\partial \mathscr{L}}{\partial \gamma^d} = \frac{\partial}{\partial t}\frac{\partial \mathscr{L}}{\partial \dot{\gamma}^d}, \quad \text{for } d \in 1, \ldots, D.$$

We first consider the left-hand side

$$I := \frac{\partial \mathscr{L}}{\partial \gamma^d} = \frac{1}{2}\frac{\partial M_{ij}}{\partial \gamma^d}\dot{\gamma}^i\dot{\gamma}^j,$$

which holds due to independence of the coordinates. The right-hand side is

$$II := \frac{\partial}{\partial t}\left[M_{id}\dot{\gamma}^i\right] = \frac{\partial M_{id}}{\partial \gamma^j}\dot{\gamma}^i\dot{\gamma}^j + M_{id}\ddot{\gamma}^i.$$

We expand this using a small index rearrangement trick

$$II = \frac{1}{2}\frac{\partial M_{id}}{\partial \gamma^j}\dot{\gamma}^i\dot{\gamma}^j + \frac{1}{2}\frac{\partial M_{jd}}{\partial \gamma^i}\dot{\gamma}^i\dot{\gamma}^j + M_{id}\ddot{\gamma}^i.$$

This allows us to write $I = II \Leftrightarrow II - I = 0$ as

$$M_{id}\ddot{\gamma}^i + \frac{1}{2}\left(\frac{\partial M_{id}}{\partial \gamma^j} + \frac{\partial M_{jd}}{\partial \gamma^i} - \frac{\partial M_{ij}}{\partial \gamma^d}\right)\dot{\gamma}^j + M_{id}\ddot{\gamma}^i = 0.$$

the next step is to left multiply with the inverse metric tensor and plug in the *Christoffel symbols* defined as follows

$$\Gamma^d_{ij} = \frac{1}{2}M^{-1}_{dh}\left(\frac{\partial M_{ih}}{\partial \gamma^j} + \frac{\partial M_{jh}}{\partial \gamma^i} - \frac{\partial M_{ij}}{\partial \gamma^h}\right),$$

so we finally obtain the geodesic equations in the canonical form

$$\ddot{\gamma}^d + \Gamma^d_{ij}\dot{\gamma}^j\dot{\gamma}^j = 0, \quad \text{for } d \in 1, \ldots, D.$$

We assume our manifold to be *geodesically complete* [200], which means that geodesics can be infinitely extended, i.e., their domain is $\mathbb{R}$. As a consequence, the exponential map is then defined on the whole tangent space. In theory, the exponential map $\mathrm{Exp}_{\boldsymbol{\mu}}(\cdot)$ is a *diffeomorphism* only in some open neighborhood around $\boldsymbol{\mu}$ and thus it only admits a smooth inverse, i.e., $\mathrm{Log}_{\boldsymbol{\mu}}(\cdot)$, in said neighborhood. However, we assume this to be true on the whole manifold in practice to keep the analysis tractable. For long geodesics on high-curvature data manifolds, often $\mathrm{Log}_{\boldsymbol{\mu}}(\mathrm{Exp}_{\boldsymbol{\mu}}(\boldsymbol{v})) \neq \boldsymbol{v}$. This is rather unproblematic since if $\|\mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)\|$ is high, the responsibility r_{nk} will be low (see Section C.2), so this logarithmic map will play a minimal role in the Mahalanobis distance of the LAND density. Thus, the optimization process on its own favors mean and covariances such that the density is concentrated in sufficiently small neighborhoods where the exponential map approximately admits an inverse.

[200]: Pennec (2006), 'Intrinsic statistics on Riemannian manifolds: Basic tools for geometric measurements'

C.1.2 Covariance and precision matrices

We here elaborate on Footnote 1 of the paper. The Riemannian normal distribution [200] is defined using the precision matrix $\boldsymbol{\Gamma}$. This matrix lives on the tangent space $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M}$, i.e., it may be represented as a matrix in $\mathbb{R}^{D\times D}$, where D is the dimension of the tangent space, which is equal to the topological dimension of the manifold. In our applied setting, D matches the dimension of the data space, as we view the whole $\mathbb{R}^D$ as the manifold. We can use the tangent space 'covariance' matrix $\boldsymbol{\Sigma} = \boldsymbol{\Gamma}^{-1}$ for our reasoning and the optimization process. However, to obtain the true covariance on the manifold $\mathcal{M}$, a subtle correction is necessary [200]

$$\boldsymbol{\Sigma}_{\mathcal{M}} = \mathbb{E}\left[\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^\top\right] = \frac{1}{\mathcal{C}}\int_{\mathcal{M}} \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})^\top \exp\left(-\frac{1}{2}\left\langle \mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x}), \boldsymbol{\Gamma}\,\mathrm{Log}_{\boldsymbol{\mu}}(\boldsymbol{x})\right\rangle\right) \mathrm{d}\mathcal{M}(\boldsymbol{x}),$$

with respect to the density on the manifold. For conceptual ease, we focus on the tangent space view in the paper. To plot the eigenvectors of the ADK LAND covariance (Figure 6.17), we used the exponential map on the tangent space covariance matrix, i.e., we evaluate and plot $\mathrm{Exp}_{\boldsymbol{\mu}}(\boldsymbol{v}_{1:2})$, where $\boldsymbol{v}_{1:2}$ are the eigenvectors of $\boldsymbol{\Sigma}$.

C.1.3 Geodesic Solvers

To solve the geodesic equations, we combine two solvers, which have different strengths and weaknesses. By chaining them together, we obtain a more robust computational pipeline.

First, we make use of the fast and robust fixed-point solver (FP) introduced by Arvanitidis et al. [12]. This solver pursues a GP-based approach that avoids the often ill-behaved Jacobians of the geodesic ODE system. However, the resulting logarithmic maps are subject to significant approximation error, depending on the curvature of the manifold. The parameters of this solver are as follows:

[12]: Arvanitidis et al. (2019), 'Fast and robust shortest paths on manifolds learned from data'

Parameter	Value	Description
$\texttt{iter}_{\texttt{max}}$	1000	maximum number of iterations
`N`	10	number of mesh nodes.
`tol`	0.1	tolerance used to evaluate solution correctness.
σ	10^{-4}	noise of the GP.

For MNIST, we set $\texttt{iter}_{\texttt{max}} = 500$, and $\texttt{tol} = 0.2$, since this high-curvature manifold easily leads to failing geodesics.

The second solver we employ is a precise, albeit less robust one. This is the BVP solver available in the module `scipy.integrate.solve_bvp`. On high-curvature manifolds, this solver often fails (especially for long curves) and takes a significant amount of time to run. When it succeeds, however, the logarithmic maps are reliable. For this solver, we set the maximum number of mesh nodes to 100 and the tolerance to 0.1. We empirically found that choosing a high maximum number of mesh nodes (e.g., 500) can lead to high runtimes for failing geodesic computations.

To obtain fast and robust geodesics, these solvers may be chained together, i.e., we initialize the BVP solver with the FP solution, which is often worth the extra effort for speedup and improved robustness. For initialization, we use 20 mesh nodes, evenly spaced on the FP solution. If the FP solver already failed, it is very unlikely for the BVP solver to succeed, so we abort the computation.

Furthermore, we exploit previously computed BVP solutions: assume we want to compute $\operatorname{Log}_{\boldsymbol{\mu}_t}(\boldsymbol{x})$. We search for past results $\operatorname{Log}_{\boldsymbol{\mu}_t^*}(\boldsymbol{x})$, with $t^* < t$, $t^* = \arg\min \|\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t*}\|$ and $\|\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t*}\| < \epsilon_d$, where we choose $\epsilon_d = 0.5$. Since we compute logarithmic maps for data points $\boldsymbol{x}_{1:N}$, which do not change during LAND optimization, we can use them as hash keys in a dictionary, where we store the solutions. Looking up the solution is then linear in the number of previous LAND iterations. If such a solution is found, the FP is skipped and the solution is used to directly initialize the BVP solver.

For the exponential maps, we use `scipy.integrate.solve_ivp` with a tolerance of 10^{-3}.

C.2 The LAND algorithm

Algorithm C.1 and Algorithm C.2 show pseudocode for the optimization of the LAND mixture likelihood (6.2). For completeness, we also state the gradients required for the gradient descent procedure.

Algorithm C.1 LAND mixture main loop

Input: data $\boldsymbol{x}_{1:N}$, manifold $\mathcal{M}$ with Exp and Log operators, max. number of iterations t_{max}, initial stepsize $\alpha_{\boldsymbol{\mu}}^1 \in \mathbb{R}$, gradient tolerance $\epsilon_{\nabla_{\boldsymbol{\mu}}}$, likelihood tolerance $\epsilon_{\mathcal{L}}$
Output: estimates $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \mathcal{C}_k, \pi_k)_{1:K}$
Initialize LAND parameters $\left(\boldsymbol{\mu}_k^1, \boldsymbol{\Sigma}_k^1, \mathcal{C}_k^1, \pi_k^1\right)_{1:K}$, $t \leftarrow 1$.
repeat
  Expectation step: $r_{nk} = \frac{\pi_k p(x_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{l=1}^K \pi_l p(x_n|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}$
  Maximization step:
  for $k = 1$ **to** K **do**
    Compute $\mathcal{C}_k^t(\boldsymbol{\mu}_k^t, \boldsymbol{\Sigma}_k^t)$ // apply BQ
    Compute $d_{\boldsymbol{\mu}_k}\mathcal{L}(\boldsymbol{\mu}_k^t, \boldsymbol{\Sigma}_k^t)$ using (C.1)
    if $||d_{\boldsymbol{\mu}_k}\mathcal{L}|| < \epsilon_{\nabla_{\boldsymbol{\mu}}}$ **then**
      Continue
    end if
    $\boldsymbol{\mu}_k^{t+1} \leftarrow \mathrm{Exp}_{\boldsymbol{\mu}_k^t}(\alpha_{\boldsymbol{\mu}}^t d_{\boldsymbol{\mu}_k}\mathcal{L})$
    Compute $\mathrm{Log}_{\boldsymbol{\mu}_k^{t+1}}(\boldsymbol{x}_{1:N})$
    Compute $\mathcal{C}_k^{t+1}(\boldsymbol{\mu}_k^{t+1}, \boldsymbol{\Sigma}_k^t)$ // apply BQ
    $\boldsymbol{\Sigma}_k^{t+1} \leftarrow \mathrm{update}_{\boldsymbol{\Sigma}_k^t}$ using Algorithm C.2
    $\pi_k^t = \frac{1}{N}\sum_{n=1}^N r_{nk}$
  end for
  if $\mathcal{L}^{t+1} < \mathcal{L}^t$ **then**
    $\alpha_{\boldsymbol{\mu}}^{t+1} \leftarrow 1.1 \cdot \alpha_{\boldsymbol{\mu}}^t$ // optimism
  else
    $\alpha_{\boldsymbol{\mu}}^{t+1} \leftarrow 0.75 \cdot \alpha_{\boldsymbol{\mu}}^t$ // pessimism
  end if
  $t \leftarrow t + 1$
until $||\mathcal{L}^{t+1} - \mathcal{L}^t|| \leq \epsilon_{\mathcal{L}}$ **or** $t = t_{max}$

Algorithm C.2 LAND update$_{\boldsymbol{\Sigma}_k}$ on the symmetric positive definite manifold $\mathcal{S}_+$

Input: Covariance $\boldsymbol{\Sigma}_k^t$, mean $\boldsymbol{\mu}_k$, max. line search iterations $t_{max,\boldsymbol{\Sigma}}$, last stepsize α_k, initial stepsize $\alpha_1 = 1.0$, sufficient decrease factor $c_0 = 0.5$, contraction factor $c_1 = 0.5$
Output: $\boldsymbol{\Sigma}_k^{t+1}$, α_k (for reuse)

Function $\mathrm{Exp}_{\boldsymbol{X}}^{+}(\boldsymbol{\Xi})$:
  // define the exp. map on the $\mathcal{S}_+$ manifold, where $\boldsymbol{X}$ is an SPD matrix and $\boldsymbol{\Xi}$ is a tangent vector, i.e., a symmetric matrix
  return $\boldsymbol{X}^{\frac{1}{2}} \exp\left(\boldsymbol{X}^{-\frac{1}{2}}\boldsymbol{\Xi}\boldsymbol{X}^{-\frac{1}{2}}\right)\boldsymbol{X}^{\frac{1}{2}}$, where exp denotes the matrix exponential.
EndFunction

Function $\left(\|\cdot\|_{\boldsymbol{X}}^{+}\right)(\boldsymbol{\Xi})$: // define the norm of a vector $\boldsymbol{\Xi}$ in the tangent space of $\boldsymbol{X} \in \mathcal{S}_+$
  $\boldsymbol{X} \leftarrow \boldsymbol{L}\boldsymbol{L}^{\top}$ // cholesky decomposition
  return $\|\boldsymbol{L}^{-1}\boldsymbol{\Xi}\boldsymbol{L}^{-\intercal}\|_2$
EndFunction

for $i = 1$ **to** 2 **do** // outer gradient descent loop
  Compute (or retrieve from cache) $\mathcal{L}(\boldsymbol{\Sigma}_k^t)$
  Compute (or retrieve from cache) Euclidean gradient $\nabla_{\boldsymbol{\Sigma}_k^t}\mathcal{L}(\Sigma_k^t)$ using (C.2)
  Obtain manifold gradient: $g := \nabla_{\boldsymbol{\Sigma}_k^t;\mathcal{S}_+} = \frac{1}{2}\boldsymbol{\Sigma}_k^t\left(\nabla_{\boldsymbol{\Sigma}_k^t} + \nabla_{\boldsymbol{\Sigma}_k^t}^{\top}\right)\boldsymbol{\Sigma}_k^t$
  if α_k **is None or** $\alpha_k = 0$ **then**
    $\alpha_k \leftarrow \frac{\alpha_0}{\|g\|}$
  end if
  $\boldsymbol{\Sigma}_k^{t+1} \leftarrow \mathrm{Exp}_{\boldsymbol{\Sigma}_k^t}^{+}(-\alpha_k \cdot g)$
  Compute $\mathcal{C}_k(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k^{t+1})$ // apply BQ
  Evaluate LAND objective $\mathcal{L}(\boldsymbol{\Sigma}_k^{t+1})$

  $j \leftarrow 1$
  while $\mathcal{L}(\boldsymbol{\Sigma}_k^{t+1}) > \mathcal{L}(\boldsymbol{\Sigma}_k^t) - c_0 \cdot \alpha_k \cdot \|g\|^2$ and $j \leq t_{max,\boldsymbol{\Sigma}}$ **do** // line search subroutine
    $\alpha_k \leftarrow \alpha_k \cdot c_1$ // while no sufficient decrease, contract
    $\boldsymbol{\Sigma}_k^{t+1} \leftarrow \mathrm{Exp}_{\boldsymbol{\Sigma}_k^t}^{+}(-\alpha_k \cdot g)$
    Compute $\mathcal{C}_k(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k^{t+1})$ // apply BQ
    Evaluate LAND objective $\mathcal{L}(\boldsymbol{\Sigma}_k^{t+1})$
    $j \leftarrow j + 1$
  end while
  if $\mathcal{L}(\boldsymbol{\Sigma}_k^{t+1}) > \mathcal{L}^t(\boldsymbol{\Sigma}_k^t)$ **then**
    $\alpha_k \leftarrow 0$
  end if
  if $j \neq 2$ **then**
    $\alpha_k = 1.3 \cdot \alpha_k$ // optimism
  end if
  $t \leftarrow t + 1$
end for

For $\boldsymbol{\mu}$, we use the steepest descent direction as in [10]

[10]: Arvanitidis et al. (2016), 'A locally adaptive normal distribution'

$$d_{\boldsymbol{\mu}_k}\mathcal{L} = \sum_{n=1}^{N} r_{nk}\mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n) - \frac{\mathcal{Z}_k \cdot R_k}{\mathcal{C}_k(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \int_{\mathcal{T}_{\boldsymbol{\mu}_k}\mathcal{M}} \boldsymbol{v} f_{\boldsymbol{\mu}_k}(\boldsymbol{v}) \mathcal{N}(\boldsymbol{v}; \mathbf{0}, \boldsymbol{\Sigma}_k)\, \mathrm{d}\boldsymbol{v}, \tag{C.1}$$

where the vector-valued integral stems from BQ and $R_k = \sum_{n=1}^{N} r_{nk}$, $\mathcal{Z}_k = \sqrt{(2\pi)^d |\boldsymbol{\Sigma}_k|}$.

[10] decomposed the precision $\boldsymbol{\Sigma}_k^{-1} = \boldsymbol{A}^\top \boldsymbol{A}$ for unconstrained optimization using gradient descent. We opt for a more principled approach by exploiting geometric structure of the symmetric positive definite (SPD) manifold, to which the covariance is confined. More specifically, we use the *bi-invariant* metric [29]. Under this metric, geodesics from $\boldsymbol{A}$ to $\boldsymbol{B}$ may be parameterized as $\gamma(t) = \boldsymbol{A}^{\frac{1}{2}} \left(\boldsymbol{A}^{-\frac{1}{2}} \boldsymbol{B}^{\frac{1}{2}} \boldsymbol{A}^{-\frac{1}{2}}\right)^t \boldsymbol{A}^{\frac{1}{2}}, \quad 0 \leq t < 1$, and the distance from $\boldsymbol{A}$ to $\boldsymbol{B}$ is $d(\boldsymbol{A}, \boldsymbol{B}) = \left\| \log \boldsymbol{A}^{-\frac{1}{2}} \boldsymbol{B}^{\frac{1}{2}} \boldsymbol{A}^{-\frac{1}{2}} \right\|_2$. The name stems from the fact that this distance is invariant under multiplication with any invertible square matrix $\boldsymbol{\Xi}$, i.e., $d(\boldsymbol{A}, \boldsymbol{B}) = d(\boldsymbol{\Xi} \cdot \boldsymbol{A}, \boldsymbol{\Xi} \cdot \boldsymbol{B})$. For manifold gradient descent, we calculate the Euclidean gradient and then project it onto the manifold. We begin with the first term

[29]: Bhatia (2009), *Positive definite matrices*

$$\nabla_{\boldsymbol{\Sigma}_k} \left(\sum_{n=1}^{N} r_{nk} \left[\frac{1}{2} \langle \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n), \boldsymbol{\Sigma}_k^{-1} \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n) \rangle \right] \right) = -\frac{1}{2} \sum_{n=1}^{N} r_{nk} \boldsymbol{\Sigma}_k^{-\intercal} \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n) \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)^\top \boldsymbol{\Sigma}_k^{-\intercal}.$$

For the gradient of the normalization constant we get

$$\begin{aligned}
\nabla_{\boldsymbol{\Sigma}_k} \log(\mathcal{C}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)) &= \frac{1}{\mathcal{C}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \int_{\mathcal{M}} \nabla_{\boldsymbol{\Sigma}_k} \exp\left(\frac{1}{2} \langle \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}), \boldsymbol{\Sigma}^{-1} \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}) \rangle \right) \mathrm{d}\mathcal{M}_{\boldsymbol{x}} \\
&= \frac{1}{2 \cdot \mathcal{C}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \int_{\mathcal{M}} \boldsymbol{\Sigma}_k^{-\intercal} \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}) \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x})^\top \boldsymbol{\Sigma}_k^{-\intercal} \exp\left(-\frac{1}{2} \langle \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}), \boldsymbol{\Sigma}^{-1} \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}) \rangle \right) \mathrm{d}\mathcal{M}_{\boldsymbol{x}} \\
&= \frac{1}{2 \cdot \mathcal{C}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \int_{\mathcal{T}_{\boldsymbol{\mu}_k}\mathcal{M}} \boldsymbol{\Sigma}_k^{-\intercal} \boldsymbol{v}\boldsymbol{v}^\top f_{\boldsymbol{\mu}_k}(\boldsymbol{v}) \boldsymbol{\Sigma}_k^{-\intercal} \exp\left(-\frac{1}{2} \langle \boldsymbol{v}, \boldsymbol{\Sigma}^{-1} \boldsymbol{v} \rangle \right) \mathrm{d}\boldsymbol{v}.
\end{aligned}$$

Taking this together, we obtain the gradient

$$\begin{aligned}
\nabla_{\boldsymbol{\Sigma}_k}\mathcal{L} = &-\frac{1}{2} \sum_{n=1}^{N} r_{nk} \boldsymbol{\Sigma}_k^{-\intercal} \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n) \mathrm{Log}_{\boldsymbol{\mu}_k}(\boldsymbol{x}_n)^\top \boldsymbol{\Sigma}_k^{-\intercal} \\
&+ \frac{R_k}{2 \cdot \mathcal{C}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \int_{\mathcal{T}_{\boldsymbol{\mu}_k}\mathcal{M}} \boldsymbol{\Sigma}_k^{-\intercal} \boldsymbol{v}\boldsymbol{v}^\top f_{\boldsymbol{\mu}_k}(\boldsymbol{v}) \boldsymbol{\Sigma}_k^{-\intercal} \exp\left(-\frac{1}{2} \langle \boldsymbol{v}, \boldsymbol{\Sigma}^{-1} \boldsymbol{v} \rangle \right) \mathrm{d}\boldsymbol{v},
\end{aligned} \tag{C.2}$$

where the matrix-valued integral again stems from BQ. To project the Euclidean gradient $\nabla_{\boldsymbol{\Sigma}_k}$ onto the tangent space of a SPD matrix $\boldsymbol{\Sigma}_k$, we simply calculate $\frac{1}{2}\boldsymbol{\Sigma}_k \left(\nabla_{\boldsymbol{\Sigma}_k} + \nabla_{\boldsymbol{\Sigma}_k}^\top\right) \boldsymbol{\Sigma}_k$. We optimize with gradient descent and a deterministic manifold line search as a subroutine, which adaptively chooses its step lengths. This procedure as well as the SPD manifold are conveniently available in the `Pymanopt` [245] library.

[245]: Townsend et al. (2016), 'Pymanopt: A Python toolbox for optimization on manifolds using automatic differentiation'

In sum, the optimization process is as follows: we cycle through the components K. After taking a single steepest-direction step for $\boldsymbol{\mu}_k$, we perform two gradient descent steps for $\boldsymbol{\Sigma}_k$, each of which may use up to 4 steps in the line search subroutine to satisfy a sufficient decrease criterion. We provide pseudocode for the covariance update in Alg. C.2. The optimizer uses the following hyperparameters:

Parameter	Value	Description
t_{max}	-	update each component $\mathtt{t_{max}}$ times.
α_{μ}^{1}	-	initial stepsize for mean updates.
$\epsilon_{\nabla_{\mu}}$	-	tolerance for mean gradients
$\epsilon_{\mathcal{L}}$	2	likelihood tolerance
$t_{max,\mathbf{\Sigma}}$	4	max. $\mathbf{\Sigma}$ line search steps.
α_1	1.0	initial step size ($\mathbf{\Sigma}$ line search).
c_0	0.5	sufficient decrease factor ($\mathbf{\Sigma}$ line search).
c_1	0.5	contraction factor ($\mathbf{\Sigma}$ line search)

Cells with unspecified values (-) imply that the value of the respective parameter is not equal across all experiments and problems. Experiment-specific parameter details are in Section 6.5.

C.3 The directional cumulative variance

The DCV acquisition function is

$$\bar{\alpha}(\hat{v}) = \int_0^\infty \alpha(\beta\hat{v})\,\mathrm{d}\beta = \int_0^\infty k_{\mathsf{f}\,|\,\mathcal{D}}(\beta\hat{v},\beta\hat{v})\nu(\beta\hat{v})^2\,\mathrm{d}\beta,$$

with derivative

$$\frac{\partial}{\partial\hat{v}}\bar{\alpha}(\hat{v}) = \int_0^\infty \beta\nu(\beta\hat{v})\left[2k_{\mathsf{f}\,|\,\mathcal{D}}(\beta\hat{v},\beta\hat{v})\frac{\partial}{\partial\beta\hat{v}}\nu(\beta\hat{v}) + \nu(\beta\hat{v})\frac{\partial}{\partial\beta\hat{v}}k_{\mathsf{f}\,|\,\mathcal{D}}(\beta\hat{v},\beta\hat{v})\right]\mathrm{d}\beta.$$

Since the integration measure is Gaussian, i.e., $\nu(\beta\hat{v}) = \mathcal{N}(\beta\hat{v};0,\mathbf{\Sigma})$, its derivative is

$$\frac{\partial}{\partial\beta\hat{v}}\nu(\beta\hat{v}) = -\nu(\beta\hat{v})\mathbf{\Sigma}^{-1}\beta\hat{v}.$$

For simplicity, we always use WSABI-L in combination with DCV, so the derivative of the variance of the warped GP is

$$\begin{aligned}\frac{\partial}{\partial\beta\hat{v}}k_{\mathsf{f}\,|\,\mathcal{D}}(\beta\hat{v},\beta\hat{v}) &= \frac{\partial}{\partial\beta\hat{v}}\left[m_{\mathsf{g}\,|\,\mathcal{D}}(\beta\hat{v})^2 k_{\mathsf{g}\,|\,\mathcal{D}}(\beta\hat{v},\beta\hat{v})\right]\\ &= 2m_{\mathsf{g}\,|\,\mathcal{D}}(\beta\hat{v})k_{\mathsf{g}\,|\,\mathcal{D}}(\beta\hat{v},\beta\hat{v})\frac{\partial}{\partial\beta\hat{v}}m_{\mathsf{g}\,|\,\mathcal{D}}(\beta\hat{v}) + \frac{\partial}{\partial\beta\hat{v}}m_{\mathsf{g}\,|\,\mathcal{D}}k_{\mathsf{g}\,|\,\mathcal{D}}(\beta\hat{v},\beta\hat{v})m_{\mathsf{g}\,|\,\mathcal{D}}(\beta\hat{v})^2.\end{aligned}$$

the derivative of the DCV acquisition function is significantly more costly to evaluate than the objective, because it requires predictive gradients of the underlying GP. Instead of using a quadrature routine like `scipy.quad`, which would evaluate the integral for every dimension sequentially, we use Simpson's rule on 50 evenly spaced points between 0 and $v_{\max}$ (defined below). Since these are multiple univariate integrals of a smooth function, the errors are practically negligible.

The scalar $v_{\max}$ simultaneously constitutes an upper bound for the integration and the length of the exponential map. A bound is reasonable since longer exponential maps are slower to compute and the integration measure concentrates the mass near the center, so very far-away locations become irrelevant. For a sensible bound, we use the chi-square distribution:

$$\langle v\cdot\hat{v}, \mathbf{\Sigma}^{-1}v\cdot\hat{v}\rangle = \chi_p^2$$

by choosing a high value $p = 99.5\%$, we make sure that there is no significant amount of mass outside of this isoprobability contour. Note that this limit applies only to the computation of exponential maps and the collection of observations, not to the main quadrature itself.

Since $\hat{v}$ is constrained to lie on the unit hypersphere, we employ manifold gradient descent with a line search subroutine. Conveniently, the line search only evaluates the objective and not its gradient, which saves a significant amount of time. Overall, optimizing this acquisition function is costly, however.

For completeness, we briefly describe the geometry of the unit (hyper)sphere. If the tangent space of our data manifold is $\mathcal{T}_{\boldsymbol{\mu}}\mathcal{M} = \mathbb{R}^D$, then a direction in this tangent space is a point on S^{D-1}, which we represent as a unit norm vector in $\mathbb{R}^D$. For a point $\boldsymbol{x}$ on the sphere and a tangent vector $\boldsymbol{\xi}$, which lies in the plane touching the sphere tangentially, the exponential map is $\mathrm{Exp}_{\boldsymbol{x}}(\boldsymbol{\xi}) = \cos(\|\boldsymbol{\xi}\|_2)\boldsymbol{x} + \sin(\|\boldsymbol{\xi}\|_2)\frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|_2}$. However, the optimizer uses a *retraction map* $\mathrm{Retr}_{\boldsymbol{x}}(\boldsymbol{\xi}) = \frac{\boldsymbol{x}+\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|_2}$ instead of the exponential map to take a descent step. To obtain the gradient on the manifold, the Euclidean gradient is orthogonally projected onto the tangent plane.

The gradient descent is allowed a maximum of 15 steps in the "error vs. runtime experiment", whereas in the boxplot experiment we decrease this number to 5, as this experiment focuses more on speed given a fixed number of samples. The line search may use up to 5 steps. We set the optimism of the line search to 2.0 and the initial stepsize to 1.0. If a descent step has norm less than 10^{-10}, the optimization is aborted.

After an exponential map is computed according to DCV, we discretize the resulting straight line in the tangent space into 30 evenly spaced points and sequentially select 6 points using the standard WSABI objective, updating the GP after each observation.

C.3.1 Analytical solution of DCV

The integral (6.6) comes with a closed-form solution which we state here. It did not turn out useful in practice, because for a typical number of BQ nodes, the cost of evaluating this solution becomes prohibitive. We provide intuition about this issue further below.
For the cumulative variance acquisition function, we need to compute univariate integrals of the form

$$\Psi(\hat{\boldsymbol{v}}, \boldsymbol{a}, \boldsymbol{A}) := \int_0^\infty \mathcal{N}(\beta\hat{\boldsymbol{v}}, \boldsymbol{a}, \boldsymbol{A})\, d\beta \;=\; \sigma_\beta(\hat{\boldsymbol{v}})\sqrt{2\pi}\, \exp\left(\frac{\mu_\beta^2(\hat{\boldsymbol{v}})}{2\sigma_\beta^2(\hat{\boldsymbol{v}})}\right)\, \Phi_0\left(\frac{\mu_\beta(\hat{\boldsymbol{v}})}{\sigma_\beta(\hat{\boldsymbol{v}})}\right)\, \mathcal{N}(\boldsymbol{a}, 0, \boldsymbol{A})$$

with the cumulative Gaussian

$$\Phi_0(x) = \int_{-\infty}^{x} \mathcal{N}(x, 0, 1)\, dx \;=\; \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right]$$

and

$$\mu_\beta(\hat{\boldsymbol{v}}) = \frac{\hat{\boldsymbol{v}}^\top \boldsymbol{A}^{-1}\boldsymbol{a}}{\hat{\boldsymbol{v}}^\top \boldsymbol{A}^{-1}\hat{\boldsymbol{v}}},$$
$$\sigma_\beta(\hat{\boldsymbol{v}}) = \frac{1}{\sqrt{\hat{\boldsymbol{v}}^\top \boldsymbol{A}^{-1}\hat{\boldsymbol{v}}}}.$$

These terms have been checked numerically.

We derive the DCV acquisition for WSABI-L with constant prior mean function in model space m_0 and posterior weights $\boldsymbol{w} = \boldsymbol{K}^{-1}(\sqrt{2(\boldsymbol{y}+\delta)} - m_0)$ where $[\boldsymbol{K}]_{ij} = k(\boldsymbol{v}_i, \boldsymbol{v}_j)$. Also, we assume the integration measure $\mathcal{N}(\boldsymbol{a}, \boldsymbol{A})$ for generality, noting that in our case we are dealing with a zero-mean Gaussian $\boldsymbol{a} = 0$.

$$\begin{aligned}
\bar{\alpha}(\hat{\boldsymbol{v}}) \;&=\; \int_0^\infty m^2_{\mathrm{g}\,|\,\mathcal{D}}(\beta\hat{\boldsymbol{v}})\, k_{\mathrm{g}\,|\,\mathcal{D}}(\beta\hat{\boldsymbol{v}}, \beta\hat{\boldsymbol{v}})\, \mathcal{N}(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A})^2\, d\beta \\
&=\; \int_0^\infty (m_0 + k(\boldsymbol{\beta}\hat{\boldsymbol{v}}, \boldsymbol{V})\boldsymbol{w})^2 \left(\theta^2 - k(\beta\hat{\boldsymbol{v}}, \boldsymbol{V})\boldsymbol{K}^{-1}k(\boldsymbol{V}, \beta\hat{\boldsymbol{v}})\right) \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A}\right)^2\, d\beta \\
&=\; m_0^2\theta^2 \int_0^\infty \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A}\right)^2\, d\beta \\
&\quad +\; 2\,\theta^2 m_0 \sum_{i=1}^{N_{\mathrm{BQ}}} w_i \int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i)\, \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A}\right)^2\, d\beta \\
&\quad +\; \theta^2 \sum_{i,j=1}^{N_{\mathrm{BQ}}} w_i w_j \int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i) k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_j)\, \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A}\right)^2\, d\beta \\
&\quad -\; m_0^2 \sum_{i,j=1}^{N_{\mathrm{BQ}}} [\boldsymbol{K}^{-1}]_{ij} \int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i) k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_j)\, \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A}\right)^2\, d\beta \\
&\quad -\; 2\, m_0 \sum_{i,j,k=1}^{N_{\mathrm{BQ}}} [\boldsymbol{K}^{-1}]_{ij} w_k \int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i) k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_j) k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_k)\, \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A}\right)^2\, d\beta \\
&\quad -\; \sum_{i,j,k,l=1}^{N_{\mathrm{BQ}}} [\boldsymbol{K}^{-1}]_{ij} w_k w_l \int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i) k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_j) k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_k) k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_l)\, \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \boldsymbol{A}\right)^2\, d\beta
\end{aligned}$$

With (A.4) and (A.5) we can integrate each of these terms. With the following definition

$$\boldsymbol{\Sigma}_n = (n\boldsymbol{\Lambda}^{-1} + 2\boldsymbol{A}^{-1})^{-1} \tag{C.3}$$

we can write these individual terms as

$$\int_0^\infty \mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \frac{\boldsymbol{A}}{2}\right)\, d\beta = \Psi(\hat{\boldsymbol{v}}, \boldsymbol{a}, \boldsymbol{A}/2)$$

$$\int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i)\,\mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \frac{\boldsymbol{A}}{2}\right) d\beta \;=\; \theta^2\sqrt{(2\pi)^D|\boldsymbol{\Lambda}|}\;\Psi\left(\hat{\boldsymbol{v}}; \boldsymbol{\Sigma}_1(\boldsymbol{\Lambda}^{-1}\boldsymbol{v}_i + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Sigma}_1\right)\,\mathcal{N}(\boldsymbol{v}_i, \boldsymbol{a}, \boldsymbol{\Lambda} + 1/2\boldsymbol{A})$$

$$\begin{aligned}\int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i)k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_j)\,\mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \frac{\boldsymbol{A}}{2}\right) d\beta \;&= \\ = \;\theta^4(2\pi)^D|\boldsymbol{\Lambda}|\;\Psi&\left(\hat{\boldsymbol{v}}; \boldsymbol{\Sigma}_2(\boldsymbol{\Lambda}^{-1}(\boldsymbol{v}_i + \boldsymbol{v}_j) + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Sigma}_2\right) \\ \times\,\mathcal{N}&\left(\boldsymbol{v}_j; \boldsymbol{\Sigma}_1(\boldsymbol{\Lambda}^{-1}\boldsymbol{v}_i + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_1\right)\,\mathcal{N}(\boldsymbol{v}_i, \boldsymbol{a}, \boldsymbol{\Lambda} + 1/2\boldsymbol{A})\end{aligned}$$

$$\begin{aligned}\int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i)k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_j)k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_k)\,\mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \frac{\boldsymbol{A}}{2}\right) d\beta \;&= \\ = \;\theta^6(2\pi)^{3D/2}|\boldsymbol{\Lambda}|^{3/2}\;\Psi&\left(\hat{\boldsymbol{v}}; \boldsymbol{\Sigma}_3(\boldsymbol{\Lambda}^{-1}(\boldsymbol{v}_i + \boldsymbol{v}_j + \boldsymbol{v}_k) + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Sigma}_3\right) \\ \times\,\mathcal{N}&\Big(\boldsymbol{v}_k; \boldsymbol{\Sigma}_2(\boldsymbol{\Lambda}^{-1}(\boldsymbol{v}_i + \boldsymbol{v}_j) + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_2\Big)\,\mathcal{N}\left(\boldsymbol{v}_j; \boldsymbol{\Sigma}_1(\boldsymbol{\Lambda}^{-1}\boldsymbol{v}_i + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_1\right)\mathcal{N}(\boldsymbol{v}_i, \boldsymbol{a}, \boldsymbol{\Lambda} + 1/2\boldsymbol{A})\end{aligned}$$

$$\begin{aligned}\int_0^\infty k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_i)k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_j)k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_k)k(\beta\hat{\boldsymbol{v}}, \boldsymbol{v}_l)\,\mathcal{N}\left(\beta\hat{\boldsymbol{v}}; \boldsymbol{a}, \frac{\boldsymbol{A}}{2}\right) d\beta \;&= \\ = \;\theta^8(2\pi)^{2D}|\boldsymbol{\Lambda}|^2\;\Psi&\left(\hat{\boldsymbol{v}}; \boldsymbol{\Sigma}_4(\boldsymbol{\Lambda}^{-1}(\boldsymbol{v}_i + \boldsymbol{v}_j + \boldsymbol{v}_k + \boldsymbol{v}_l) + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Sigma}_4\right) \\ \times\,\mathcal{N}&\left(\boldsymbol{v}_l; \boldsymbol{\Sigma}_3(\boldsymbol{\Lambda}^{-1}(\boldsymbol{v}_i + \boldsymbol{v}_j + \boldsymbol{v}_k) + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_3\right)\,\mathcal{N}\left(\boldsymbol{v}_k; \boldsymbol{\Sigma}_2(\boldsymbol{\Lambda}^{-1}(\boldsymbol{v}_i + \boldsymbol{v}_j) + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_2\right) \\ \times\,\mathcal{N}&\left(\boldsymbol{v}_j; \boldsymbol{\Sigma}_1(\boldsymbol{\Lambda}^{-1}\boldsymbol{v}_i + 2\boldsymbol{A}^{-1}\boldsymbol{a}), \boldsymbol{\Lambda} + \boldsymbol{\Sigma}_1\right)\,\mathcal{N}(\boldsymbol{v}_i, \boldsymbol{a}, \boldsymbol{\Lambda} + 1/2\boldsymbol{A})\end{aligned}$$

The last term is the culprit for the evaluation cost of $\mathcal{O}(M^4)$. However, the term $\boldsymbol{v}_i + \boldsymbol{v}_j + \boldsymbol{v}_k + \boldsymbol{v}_l$ has $\binom{N+k-1}{k}$ distinct terms only, and this symmetry could be used in a smart way to ease computation.

In our specific case, $\boldsymbol{A} = \boldsymbol{\Sigma}, \boldsymbol{a} = 0$ and $\boldsymbol{\Lambda} = \lambda^2\boldsymbol{I}$. Because we need to compute all the inverses $\boldsymbol{\Sigma}_n = (n\lambda^{-2}\boldsymbol{I} + \boldsymbol{\Sigma}^{-1})^{-1}$, it is desirable to have the eigendecomposition of $\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^\top$ where $\boldsymbol{D}$ is diagonal and $\boldsymbol{U}$ unitary. Cost is not an issue here since we are dealing with relatively low dimensions anyway. Then the eigendecomposition of (C.3) is

$$\begin{aligned}\boldsymbol{\Sigma}_n &= (n\lambda^{-2}\boldsymbol{I} + \boldsymbol{U}\boldsymbol{D}^{-1}\boldsymbol{U}^\top)^{-1} \\ &= \boldsymbol{U}(n\lambda^{-2}\boldsymbol{I} + \boldsymbol{D}^{-1})^{-1}\boldsymbol{U}^\top.\end{aligned}$$

We find that evaluating (6.6) by numerical integration is way more efficient than the analytical solution for the number of tangent vectors that we consider, so this path has not been pursued further.

Derivatives for Entropy Search

D

In order to compute a first-order approximation to the objective function in entropy search, we need the derivatives of $\hat{p}_{\min}$ w.r.t. the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. The algorithm requires the following derivative, where $\lambda = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\lambda}\log p_{\min} &\approx \frac{1}{\hat{p}_{\min}}\frac{\mathrm{d}\hat{p}_{\min}}{\mathrm{d}\lambda}\\ &= \frac{1}{\hat{p}_{\min}}\int \mathrm{d}\boldsymbol{f}\ \frac{\mathrm{d}\mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\mathrm{d}\lambda}\prod_{j\neq i}^{N_R}\mathbb{1}\left[[\boldsymbol{M}\boldsymbol{f}]_j > 0\right]\\ &= \frac{1}{\hat{p}_{\min}}\mathbb{E}\left[\frac{\mathrm{d}\log\mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\mathrm{d}\lambda}\right],\end{aligned}$$

using $\frac{\mathrm{d}\mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\mathrm{d}\lambda} = \mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})\frac{\mathrm{d}\log\mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\mathrm{d}\lambda}$. Hence, all we need is to compute the derivatives of the log normal distribution w.r.t. its parameters, and the expected values thereof w.r.t. the integrand. The required derivatives are

$$\frac{\mathrm{d}\log\mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\mathrm{d}\mu_i} = \left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{f}-\boldsymbol{\mu})\right]_i,$$

$$\frac{\mathrm{d}\log\mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\mathrm{d}\boldsymbol{\Sigma}_{ij}} = \frac{1}{2}\left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{f}-\boldsymbol{\mu})(\boldsymbol{f}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}\right]_{ij}$$

and the second derivative

$$\frac{\mathrm{d}^2\mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})}{\mathrm{d}\mu_i\,\mathrm{d}\mu_j} = \mathcal{N}(\boldsymbol{f},\boldsymbol{\mu},\boldsymbol{\Sigma})\left(\left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{f}-\boldsymbol{\mu})(\boldsymbol{f}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}\right]_{ij}\right)$$

Hence we only need $\mathbb{E}_{p_{\min}}[(\boldsymbol{f}-\boldsymbol{\mu})]$ and $\mathbb{E}_{p_{\min}}[(\boldsymbol{f}-\boldsymbol{\mu})(\boldsymbol{f}-\boldsymbol{\mu})^\top]$ to compute the following gradients,

$$\frac{\mathrm{d}\log p_{\min}}{\mathrm{d}\mu_i} \approx \frac{1}{\hat{p}_{\min}}\mathbb{E}_{\hat{p}_{\min}}\left[\left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{f}-\boldsymbol{\mu})\right]_i\right],$$

$$\frac{\mathrm{d}\log p_{\min}}{\mathrm{d}\boldsymbol{\Sigma}_{ij}} \approx \frac{1}{\hat{p}_{\min}}\mathbb{E}_{\hat{p}_{\min}}\left[\frac{1}{2}\left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{f}-\boldsymbol{\mu})(\boldsymbol{f}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}\right]_{ij}\right],$$

and the Hessian w.r.t. $\boldsymbol{\mu}$,

$$\frac{\mathrm{d}^2\log p_{\min}}{\mathrm{d}\mu_i\,\mathrm{d}\mu_j} = 2\frac{\mathrm{d}\log\hat{p}_{\min}}{\mathrm{d}\boldsymbol{\Sigma}_{ij}} - \frac{\mathrm{d}\log p_{\min}}{\mathrm{d}\mu_i}\frac{\mathrm{d}\log p_{\min}}{\mathrm{d}\mu_j}.$$

Bibliography

[1] L. Acerbi. 'Variational Bayesian Monte Carlo'. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., 2018 (cited on pages 36, 47, 48).

[2] L. Acerbi. 'Variational Bayesian Monte Carlo with noisy likelihoods'. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 8211–8222 (cited on pages 36, 47).

[3] C. Agrell. 'Gaussian Processes with linear operator inequality constraints'. In: *Journal of Machine Learning Research* 20.135 (2019), pp. 1–36 (cited on pages 97, 110).

[4] B. Ajna and T. Dalenius. 'Några tillämpningar av statistiska idéer på numerisk integration'. In: *Nordisk Matematisk Tidskrift* 8.4 (1960), pp. 145–152 (cited on page 6).

[5] R. T. Akella, K. Azizzadenesheli, M. Ghavamzadeh, A. Anandkumar, and Y. Yue. 'Deep Bayesian quadrature policy optimization'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 8. 2021, pp. 6600–6608 (cited on page 37).

[6] M. A. Alvarez, L. Rosasco, and N. D. Lawrence. 'Kernels for vector-valued functions: A review'. In: *Foundations and Trendső in Machine Learning* 4.3 (2012), pp. 195–266 (cited on pages 60, 61, 63).

[7] A. Amadei, A. B. Linssen, and H. J. Berendsen. 'Essential dynamics of proteins'. In: *Proteins: Structure, Function, and Bioinformatics* 17.4 (1993), pp. 412–425 (cited on page 89).

[8] L. N. Andersen, P. J. Laub, and L. Rojas-Nandayapa. 'Efficient simulation for dependent rare events with applications to extremes'. In: *Methodology and Computing in Applied Probability* 20.1 (2018), pp. 385–409 (cited on page 96).

[9] N. Aronszajn. 'Theory of reproducing kernels'. In: *Transactions of the American mathematical society* 68.3 (1950), pp. 337–404 (cited on page 25).

[10] G. Arvanitidis, L. K. Hansen, and S. Hauberg. 'A locally adaptive normal distribution'. In: *Advances in Neural Information Processing Systems*. Ed. by D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett. 2016, pp. 4251–4259 (cited on pages 77, 80, 82, 130).

[11] G. Arvanitidis, L. K. Hansen, and S. Hauberg. 'Latent space oddity: On the curvature of deep generative models'. In: *6th International Conference on Learning Representations, ICLR 2018*. 2018 (cited on page 81).

[12] G. Arvanitidis, S. Hauberg, P. Hennig, and M. Schober. 'Fast and robust shortest paths on manifolds learned from data'. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. Ed. by K. Chaudhuri and M. Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR. PMLR, 2019, pp. 1506–1515 (cited on pages 80, 126).

[13] G. Arvanitidis, S. Hauberg, and B. Schölkopf. 'Geometrically enriched latent spaces'. In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by A. Banerjee and K. Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, Apr. 2021, pp. 631–639 (cited on page 81).

[14] J. R. Ashford and R. R. Sowden. 'Multi-variate probit analysis'. In: *Biometrics* (1970), pp. 535–546 (cited on page 97).

[15] S.-K. Au and J. L. Beck. 'Estimation of small failure probabilities in high dimensions by subset simulation'. In: *Probabilistic Engineering Mechanics* 16.4 (2001), pp. 263–277 (cited on pages 96, 101, 103).

[16] S.-K. Au and J. L. Beck. 'First excursion probabilities for linear systems by very efficient importance sampling'. In: *Probabilistic Engineering Mechanics* 16.3 (2001), pp. 193–207 (cited on page 104).

[17] D. Azzimonti and D. Ginsbourger. 'Estimating orthant probabilities of high-dimensional Gaussian vectors with an application to set estimation'. In: *Journal of Computational and Graphical Statistics, Taylor & Francis, 2018, 27 (2), pp.255-267* 27.2 (2017), pp. 255–267 (cited on pages 97, 98).

[18] F. Bach. 'On the equivalence between kernel quadrature rules and random feature expansions'. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 714–751 (cited on page 36).

[19] F. Bach, S. Lacoste-Julien, and G. Obozinski. 'On the equivalence between herding and conditional gradient algorithms'. In: *ICML 2012 International Conference on Machine Learning*. 2012 (cited on page 46).

[20] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy. 'BoTorch: A framework for efficient Monte-Carlo Bayesian optimization'. In: *Advances in Neural Information Processing Systems 33*. 2020 (cited on page 7).

[21] R. Bardenet, A. Hardy, et al. 'Monte Carlo with determinantal point processes'. In: *The Annals of Applied Probability* 30.1 (2020), pp. 368–417 (cited on page 49).

[22] S. Bartels, J. Cockayne, I. Ipsen, and P. Hennig. 'Probabilistic linear solvers: A unifying view'. In: *Statistics and Computing* 29.6 (2019), pp. 1249–1263 (cited on page 7).

[23] T. Bayes. 'An essay towards solving a problem in the doctrine of chances'. In: *Philosophical transactions of the Royal Society of London* 53 (1763), pp. 370–418 (cited on page 5).

[24] J. Bect, L. Li, and E. Vazquez. 'Bayesian subset simulation'. In: *SIAM/ASA Journal on Uncertainty Quantification* 5.1 (2017), pp. 762–786 (cited on page 104).

[25] A. Belhadji, R. Bardenet, and P. Chainais. 'Kernel quadrature with DPPs'. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc., 2019 (cited on page 50).

[26] R. Bellman, R. Bellman, and R. Corporation. *Dynamic programming*. Rand Corporation research study. Princeton University Press, 1957 (cited on page 3).

[27] P. Benner, A. Cohen, M. Ohlberger, and K. Willcox. *Model reduction and approximation: theory and algorithms*. Vol. 15. SIAM, 2017 (cited on page 60).

[28] M. Betancourt. 'A conceptual introduction to Hamiltonian Monte Carlo'. In: (Jan. 10, 2017) (cited on page 54).

[29] R. Bhatia. *Positive definite matrices*. Vol. 24. Princeton University Press, 2009 (cited on page 130).

[30] C. M. Bishop. 'Pattern recognition and machine learning'. In: Springer-Verlag, 2006 (cited on pages 6, 80).

[31] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. 'Variational inference: A review for statisticians'. In: *Journal of the American statistical Association* 112.518 (2017), pp. 859–877 (cited on page 6).

[32] D. Bolin and F. Lindgren. 'Excursion and contour uncertainty regions for latent Gaussian models'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 77.1 (Jan. 2015), pp. 85–106 (cited on page 96).

[33] N. Bosch, P. Hennig, and F. Tronarp. 'Calibrated adaptive probabilistic ODE solvers'. In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by A. Banerjee and K. Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, 2021, pp. 3466–3474 (cited on page 7).

[34] N. Bosch, F. Tronarp, and P. Hennig. 'Pick-and-mix information operators for probabilistic ODE solvers'. In: *arXiv preprint arXiv:2110.10770* (2021) (cited on page 8).

[35] Z. I. Botev. 'The normal law under linear restrictions: Simulation and estimation via minimax tilting'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 79.1 (2016), pp. 125–148 (cited on pages 97, 98, 100).

[36] F.-X. Briol, C. J. Oates, M. Girolami, and M. A. Osborne. 'Frank-Wolfe Bayesian quadrature: probabilistic integration with theoretical guarantees'. In: *arXiv:1506.02681 [stat.ML]* (2015) (cited on pages 45, 46).

[37] F.-X. Briol, C. J. Oates, M. Girolami, M. A. Osborne, and D. Sejdinovic. 'Probabilistic integration: A role in statistical computation?' In: *Statistical Science* 34.1 (Feb. 2019), pp. 1–22 (cited on pages 7, 15, 24, 27, 28, 48, 49, 116).

[38] F.-X. Briol. 'Statistical computation with kernels'. PhD thesis. 2018 (cited on pages 28, 29, 48, 49).

[39] F.-X. Briol, C. J. Oates, J. Cockayne, W. Y. Chen, and M. Girolami. 'On the sampling problem for kernel quadrature'. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by D. Precup and Y. W. Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, Aug. 2017, pp. 586–595 (cited on page 48).

[40] M. P. do Carmo. *Riemannian geometry*. Birkhäuser, 1992 (cited on page 79).

[41] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. 'Stan: A probabilistic programming language'. In: *Journal of statistical software* 76.1 (2017), pp. 1–32 (cited on pages 51, 54).

[42] W. F. Caselton and J. V. Zidek. 'Optimal monitoring network designs'. In: *Statistics & Probability Letters* 2.4 (1984), pp. 223–227 (cited on page 43).

[43] F. Cérou, P. Del Moral, T. Furon, and A. Guyader. 'Sequential Monte Carlo for rare event estimation'. In: *Statistics and Computing* 22.3 (2012), pp. 795–908 (cited on pages 101, 104).

[44] H. Chai and R. Garnett. 'Improving quadrature for constrained integrands'. In: *Proceedings of Machine Learning Research*. Ed. by K. Chaudhuri and M. Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 2019, pp. 2751–2759 (cited on pages 33–36, 46).

[45] H. Chai, J.-F. Ton, M. A. Osborne, and R. Garnett. 'Automated model selection with Bayesian quadrature'. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 931–940 (cited on pages 37, 116).

[46] K. Chaloner and I. Verdinelli. 'Bayesian experimental design: A review'. In: *Statistical Science* (1995), pp. 273–304 (cited on pages 39, 41, 42).

[47] Y.-I. Chen and Y.-M. Chang. 'Identification of the minimum effective dose for right-censored survival data'. In: *Computational Statistics & Data Analysis* 51.6 (2007), pp. 3213–3222 (cited on page 96).

[48] J. Child. *The geometrical lectures of Isaac Barrow*. 3. Open Court Publishing Company, 1916 (cited on page 3).

[49] O. A. Chkrebtii, D. A. Campbell, B. Calderhead, M. A. Girolami, et al. 'Bayesian solution uncertainty quantification for differential equations'. In: *Bayesian Analysis* 11.4 (2016), pp. 1239–1267 (cited on page 7).

[50] C. W. Clenshaw and A. R. Curtis. 'A method for numerical integration on an automatic computer'. In: *Numerische Mathematik* 2.1 (1960), pp. 197–205 (cited on page 31).

[51] J. Cockayne, C. Oates, T. J. Sullivan, and M. Girolami. 'Bayesian probabilistic numerical methods'. In: *SIAM Review* 61.4 (2019), pp. 756–789 (cited on pages 6, 7, 30).

[52] J. Cockayne, C. Oates, T. Sullivan, and M. Girolami. 'Probabilistic numerical methods for PDE-constrained Bayesian inverse problems'. In: *AIP Conference Proceedings*. Author(s), 2017 (cited on page 7).

[53] T. D. Cook. 'Sequential Bayesian quadrature'. PhD thesis. The University of Wisconsin-Madison, 1993 (cited on page 45).

[54] R. T. Cox. 'Probability, frequency and reasonable expectation'. In: *American journal of physics* 14.1 (1946), pp. 1–13 (cited on page 5).

[55] P. Craig. 'A new reconstruction of multivariate normal orthant probabilities'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70.1 (Feb. 2008), pp. 227–243 (cited on page 97).

[56] J. P. Cunningham, P. Hennig, and S. Lacoste-Julien. 'Gaussian probabilities and expectation propagation'. In: *arXiv e-prints*, arXiv:1111.6832 (Nov. 2011) (cited on pages 97, 98, 104, 106).

[57] S. Da Veiga and A. Marrel. 'Gaussian process modeling with inequality constraints'. en. In: *Annales de la Faculté des sciences de Toulouse : Mathématiques* Ser. 6, 21.3 (2012), pp. 529–555 (cited on pages 97, 110).

[58] D. J. Daley and J. Gani. *Epidemic modelling: An introduction*. Cambridge Studies in Mathematical Biology. Cambridge University Press, 1999 (cited on page 71).

[59] P. Damien and S. G. Walker. 'Sampling truncated normal, beta, and gamma densities'. In: *Journal of Computational and Graphical Statistics* 10.2 (2001), pp. 206–215 (cited on page 100).

[60] P. Davis and P. Rabinowitz. *Methods of numerical integration*. Academic Press, 1983 (cited on pages 3, 29, 31).

[61] F. de Roos, A. Gessner, and P. Hennig. 'High-dimensional Gaussian process inference with derivatives'. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021, pp. 2535–2545 (cited on pages 11, 12).

[62] F. de Roos and P. Hennig. 'Active probabilistic inference on matrices for pre-conditioning in stochastic optimization'. In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by K. Chaudhuri and M. Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 2019, pp. 1448–1457 (cited on page 7).

[63] M. P. Deisenroth, M. F. Huber, and U. D. Hanebeck. 'Analytic moment-based Gaussian process filtering'. In: *Proceedings of the 26th annual international conference on machine learning*. 2009, pp. 225–232 (cited on page 36).

[64] M. P. Deisenroth, R. D. Turner, M. F. Huber, U. D. Hanebeck, and C. E. Rasmussen. 'Robust filtering and smoothing with Gaussian processes'. In: *IEEE Transactions on Automatic Control* 57.7 (2011), pp. 1865–1871 (cited on page 36).

[65] P. Del Moral, A. Doucet, and A. Jasra. 'Sequential Monte Carlo samplers'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.3 (2006), pp. 411–436 (cited on pages 48, 104).

[66] P. Diaconis. 'Bayesian numerical analysis'. In: *Statistical Decision Theory and Related Topics IV* 1 (1988), pp. 163–175 (cited on pages 6, 15, 30).

[67] P. Diaconis and S. Holmes. 'Three examples of Monte-Carlo Markov chains: At the interface between statistical computing, computer science, and statistical mechanics'. In: *Discrete Probability and Algorithms*. Springer New York, 1995, pp. 43–56 (cited on pages 95, 101).

[68] J. Dick, F. Y. Kuo, and I. H. Sloan. 'High-dimensional integration: the quasi-Monte Carlo way'. In: *Acta Numerica* 22 (2013), p. 133 (cited on page 48).

[69] S. Duane, A. Kennedy, B. J. Pendleton, and D. Roweth. 'Hybrid Monte Carlo'. In: *Physics Letters B* 195.2 (1987), pp. 216–222 (cited on page 54).

[70] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin. 'A general safety framework for learning-based control in uncertain robotic systems'. In: *IEEE Transactions on Automatic Control* (2018) (cited on page 96).

[71] M. Fisher, C. Oates, C. Powell, and A. Teckentrup. 'A locally adaptive Bayesian cubature method'. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 1265–1275 (cited on page 32).

[72] J. Fitzsimons, A. Al Ali, M. Osborne, and S. Roberts. 'A general framework for fair regression'. In: *Entropy* 21.8 (2019), p. 741 (cited on page 37).

[73] G. Flato et al. 'Evaluation of climate models'. In: *Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, 2013. Chap. 9, pp. 741–866 (cited on pages 3, 60).

[74] A. I. J. Forrester, A. Sóbester, and A. J. Keane. 'Multi-fidelity optimization via surrogate modelling'. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 463.2088 (2007), pp. 3251–3269 (cited on pages 60, 69, 109).

[75] M. Frank, P. Wolfe, et al. 'An algorithm for quadratic programming'. In: *Naval research logistics quarterly* 3.1-2 (1956), pp. 95–110 (cited on page 46).

[76] J. P. French and S. R. Sain. 'Spatio-temporal exceedance locations and confidence regions'. In: *Ann. Appl. Stat.* 7.3 (Sept. 2013), pp. 1421–1449 (cited on page 96).

[77] C. Fröhlich, A. Gessner, P. Hennig, B. Schölkopf, and G. Arvanitidis. 'Bayesian quadrature on Riemannian data manifolds'. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021, pp. 3459–3468 (cited on pages 11, 77).

[78] R. Garnett. *Bayesian optimization*. in preparation. Cambridge University Press, 2022 (cited on pages 7, 108).

[79] H. I. Gassmann, I. Deák, and T. Szántai. 'Computing multivariate normal probabilities: A new look'. In: *Journal of Computational and Graphical Statistics* 11.4 (2002), pp. 920–949 (cited on page 97).

[80] H. Ge, K. Xu, and Z. Ghahramani. 'Turing: A language for flexible probabilistic inference'. In: *International Conference on Artificial Intelligence and Statistics, AISTATS 2018*. 2018, pp. 1682–1690 (cited on pages 51, 54).

[81] A. E. Gelfand and A. F. Smith. 'Sampling-based approaches to calculating marginal densities'. In: *Journal of the American statistical association* 85.410 (1990), pp. 398–409 (cited on page 54).

[82] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995 (cited on page 56).

[83] S. Geman and D. Geman. 'Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images'. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1984), pp. 721–741 (cited on page 54).

[84] M. G. Genton, D. E. Keyes, and G. Turkiyyah. 'Hierarchical decompositions for the computation of high-dimensional multivariate normal probabilities'. In: *Journal of Computational and Graphical Statistics* 27.2 (2018), pp. 268–277 (cited on pages 97, 98).

[85] M. G. Genton, Y. Ma, and H. Sang. 'On the likelihood function of Gaussian max-stable processes'. In: *Biometrika* (2011), pp. 481–488 (cited on page 96).

[86] A. Genz. 'Numerical computation of multivariate normal probabilities'. In: *Journal of Computational and Graphical Statistics* 1.2 (1992), pp. 141–149 (cited on pages 97, 100, 105).

[87] A. Genz. 'Numerical computation of rectangular bivariate and trivariate normal and t probabilities'. In: *Statistics and Computing* 14.3 (Aug. 2004), pp. 251–260 (cited on page 97).

[88] A. Genz and F. Bretz. *Computation of multivariate normal and t probabilities*. Vol. 195. Springer Science & Business Media, 2009 (cited on page 97).

[89] A. Gessner, J. Gonzalez, and M. Mahsereci. 'Active multi-information source Bayesian quadrature'. In: *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*. Ed. by A. Globerson and R. Silva. Vol. 115. Proceedings of Machine Learning Research. AUAI Press, 2019, p. 245 (cited on pages 11, 59).

[90] A. Gessner, O. Kanjilal, and P. Hennig. 'Integrals over Gaussians under linear domain constraints'. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by S. Chiappa and R. Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020, pp. 2764–2774 (cited on pages 11, 95).

[91] J. Geweke. 'Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint probabilities'. In: *Computing science and statistics: Proceedings of the 23rd symposium on the interface*. Fairfax, Virginia: Interface Foundation of North America, Inc. 1991, pp. 571–578 (cited on pages 97, 100).

[92] D. T. Gillespie. 'A general method for numerically simulating the stochastic time evolution of coupled chemical reactions'. In: *Journal of Computational Physics* 22.4 (1976), pp. 403–434 (cited on page 72).

[93] D. T. Gillespie. 'Exact stochastic simulation of coupled chemical reactions'. In: *The Journal of Physical Chemistry* 81.25 (1977), pp. 2340–2361 (cited on page 72).

[94] D. Ginsbourger, R. Le Riche, and L. Carraro. 'Kriging is well-suited to parallelize optimization'. In: *Computational intelligence in expensive optimization problems*. Springer, 2010, pp. 131–162 (cited on page 47).

[95] G. Giordano, F. Blanchini, R. Bruno, P. Colaneri, A. Di Filippo, A. Di Matteo, and M. Colaneri. 'Modelling the COVID-19 epidemic and implementation of population-wide interventions in Italy'. In: *Nature medicine* 26.6 (2020), pp. 855–860 (cited on page 71).

[96] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic. 'Multilevel splitting for estimating rare event probabilities'. In: *Operations Research* 47.4 (1999), pp. 585–600 (cited on page 101).

[97] J. González, Z. Dai, P. Hennig, and N. Lawrence. 'Batch Bayesian optimization via local penalization'. In: *Artificial intelligence and statistics*. PMLR. 2016, pp. 648–657 (cited on page 47).

[98] `GPy`. *GPy: A Gaussian process framework in Python*. `http://github.com/SheffieldML/GPy`. since 2012 (cited on page 68).

[99] I. S. Gradshteyn and I. M. Ryzhik. *Table of integrals, series, and products*. Academic press, 2014 (cited on pages 3, 15).

[100] M. Graham and A. Storkey. 'Continuously tempered Hamiltonian Monte Carlo'. English. In: *The Conference on Uncertainty in Artificial Intelligence (UAI 2017)*. Conference on Uncertainty in Artificial Intelligence , UAI 2017 ; Conference date: 11-08-2017 Through 15-08-2017. 2017 (cited on page 6).

[101] A. Griewank. *Automatic differentiation*. Princeton Companion to Applied Mathematics, Nicolas Higham Ed., Princeton University Press, 2014 (cited on page 3).

[102] T. Gunter, M. A. Osborne, R. Garnett, P. Hennig, and S. J. Roberts. 'Sampling for inference in probabilistic models with fast Bayesian quadrature'. In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. 2014, pp. 2789–2797 (cited on pages 7, 33, 34, 36, 46, 83).

[103] S. Hamid, S. Schulze, M. A. Osborne, and S. J. Roberts. 'Marginalising over stationary kernels with Bayesian quadrature'. In: *arXiv preprint arXiv:2106.07452* (2021) (cited on page 37).

[104] J. B. Hamrick and T. L. Griffiths. 'Mental rotation as Bayesian quadrature'. In: *NIPS 2013 Workshop on Bayesian Optimization in Theory and Practice*. 2013 (cited on page 37).

[105] W. K. Hastings. 'Monte Carlo sampling methods using Markov chains and their applications'. In: (1970) (cited on page 53).

[106] A. J. Hayter and Y. Lin. 'The evaluation of two-sided orthant probabilities for a quadrivariate normal distribution'. In: *Computational Statistics* 27.3 (Sept. 2012), pp. 459–471 (cited on pages 97, 98).

[107] A. J. Hayter and Y. Lin. 'The evaluation of trivariate normal probabilities defined by linear inequalities'. In: *Journal of Statistical Computation and Simulation* 83.4 (2013), pp. 668–676 (cited on page 97).

[108] P. Hennig, M. A. Osborne, and M. Girolami. 'Probabilistic numerics and uncertainty in computations'. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 471.2179 (2015) (cited on pages 4, 7, 15, 30).

[109] P. Hennig. 'Probabilistic interpretation of linear solvers'. In: *SIAM Journal on Optimization* 25.1 (2015), pp. 234–260 (cited on page 7).

[110] P. Hennig and R. Garnett. 'Exact sampling from determinantal point processes'. In: *CoRR* abs/1609.06840 (2016) (cited on page 50).

[111] P. Hennig and S. Hauberg. 'Probabilistic solutions to differential equations and their application to Riemannian statistics'. In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014*. Vol. 33. JMLR Workshop and Conference Proceedings. JMLR.org, 2014, pp. 347–355 (cited on pages 80, 91).

[112] P. Hennig, M. A. Osborne, and H. P. Kersting. *Probabilistic Numerics: Computation as Machine Learning*. Cambridge University Press, 2022 (cited on pages 6, 7, 15, 30).

[113] P. Hennig and C. J. Schuler. 'Entropy search for information-efficient global optimization.' In: *Journal of Machine Learning Research* 13.6 (2012) (cited on pages 42, 97, 108).

[114] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. 'Predictive entropy search for efficient global optimization of black-box functions'. In: *Advances in Neural Information Processing Systems*. 2014, pp. 918–926 (cited on page 110).

[115] H. W. Hethcote. 'The mathematics of infectious diseases'. In: *SIAM Review* 42.4 (2000), pp. 599–653 (cited on page 71).

[116] M. D. Hoffman and A. Gelman. 'The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo'. In: (Nov. 18, 2011) (cited on page 54).

[117] J. B. Hough, M. Krishnapur, Y. Peres, B. Virág, et al. 'Determinantal processes and independence'. In: *Probability surveys* 3 (2006), pp. 206–229 (cited on page 49).

[118] P. Hummel. *Sparse inference for Bayesian quadrature*. Student report. 2020 (cited on pages 21, 24).

[119] W. Humphrey, A. Dalke, and K. Schulten. 'VMD – Visual Molecular Dynamics'. In: *Journal of Molecular Graphics* 14 (1996), pp. 33–38 (cited on page 90).

[120] R. Huser and A. C. Davison. 'Composite likelihood estimation for the BrownResnick process'. In: *Biometrika* 100.2 (Feb. 2013), pp. 511–518 (cited on page 96).

[121] F. Huszár and D. Duvenaud. 'Optimally-weighted herding is Bayesian quadrature'. In: (2012), pp. 377–386 (cited on pages 36, 45).

[122] L. Isserlis. 'On a formula for the product moment coefficient of any order of a normal frequency distribution in any number of variables'. In: *Biometrika* 12.1-2 (Nov. 1918), pp. 134–139 (cited on page 34).

[123] S. Iwazaki, Y. Inatsu, and I. Takeuchi. 'Bayesian quadrature optimization for probability threshold robustness measure'. In: *arXiv preprint arXiv:2006.11986* (2020) (cited on page 36).

[124] J. Oettershagen. 'Construction of optimal cubature algorithms with applications to econometrics and uncertainty quantification'. Dissertation. Institut für Numerische Simulation, Universität Bonn, 2017 (cited on page 46).

[125] E. T. Jaynes. *Probability theory: The logic of science*. Cambridge University Press, 2003 (cited on pages 4, 5).

[126] H. Joe. 'Approximations to multivariate normal rectangle probabilities based on conditional expectations'. In: *Journal of the American Statistical Association* 90.431 (1995), pp. 957–964 (cited on page 97).

[127] H. Kahn and T. E. Harris. 'Estimation of particle transmission by random sampling'. In: *National Bureau of Standards applied mathematics series* 12 (1951), pp. 27–30 (cited on page 101).

[128] M. Kanagawa and P. Hennig. 'Convergence guarantees for adaptive Bayesian quadrature methods'. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 6237–6248 (cited on pages 36, 45, 47, 48).

[129] M. Kanagawa, P. Hennig, D. Sejdinovic, and B. K. Sriperumbudur. 'Gaussian processes and kernel methods: A review on connections and equivalences'. In: (July 6, 2018) (cited on pages 18, 24–26).

[130] M. Kanagawa, B. K. Sriperumbudur, and K. Fukumizu. 'Convergence guarantees for kernel-based quadrature rules in misspecified settings'. In: *Advances in Neural Information Processing Systems 29*. 2016, pp. 3288–3296 (cited on pages 29, 50).

[131] O. Kanjilal and C. Manohar. 'Markov chain splitting methods in structural reliability integral estimation'. In: *Probabilistic Engineering Mechanics* 40 (2015), pp. 42–51 (cited on page 103).

[132] T. Karvonen. *Kernel-based and Bayesian methods for numerical integration; Ydinperusteiset ja bayesilaiset menetelmät numeerisessa integroinnissa*. en. G5 Artikkeliväitöskirja. 2019 (cited on pages 28–30).

[133] T. Karvonen, M. Kanagawa, and S. Särkkä. 'On the positivity and magnitudes of Bayesian quadrature weights'. In: *Statistics and Computing* 29.6 (2019), pp. 1317–1333 (cited on page 29).

[134] T. Karvonen, C. J. Oates, and S. Särkkä. 'A Bayes-Sard cubature method'. In: *arXiv preprint arXiv:1804.03016* (2018) (cited on page 32).

[135] T. Karvonen and S. Särkkä. 'Classical quadrature rules via Gaussian processes'. In: *ArXiv e-prints* (2017) (cited on page 32).

[136] M. Katzfuss and J. Guinness. 'A general framework for Vecchia approximations of Gaussian processes'. In: *Statistical Science* 36.1 (2021), pp. 124–141 (cited on page 19).

[137] M. C. Kennedy and A. O'Hagan. 'Predicting the output from a complex computer code when fast approximations are available'. In: *Biometrika* 87.1 (2000), pp. 1–13 (cited on page 60).

[138] W. O. Kermack and A. G. McKendrick. 'A contribution to the mathematical theory of epidemics'. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 115.772 (1927), pp. 700–721 (cited on page 71).

[139] H. Kersting and P. Hennig. 'Active uncertainty calibration in Bayesian ODE Solvers'. In: *Proceedings of the 32nd Conference on Uncertainty in Artificial Intelligence (UAI 2016)*. AUAI Press, 2016, pp. 309–318 (cited on page 36).

[140] H. Kersting, N. Krämer, M. Schiegg, C. Daniel, M. Tiemann, and P. Hennig. 'Differentiable likelihoods for fast inversion of likelihood-free dynamical systems'. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 5198–5208 (cited on page 7).

[141] H. Kersting, T. J. Sullivan, and P. Hennig. 'Convergence rates of Gaussian ODE filters'. In: *Statistics and Computing* 30.6 (2020), pp. 1791–1816 (cited on page 7).

[142] G. S. Kimeldorf and G. Wahba. 'A correspondence between Bayesian estimation on stochastic processes and smoothing by splines'. In: *The Annals of Mathematical Statistics* 41.2 (1970), pp. 495–502 (cited on page 6).

[143] D. P. Kingma and M. Welling. 'Auto-encoding variational Bayes'. In: *2nd International Conference on Learning Representations, ICLR 2014*. Ed. by Y. Bengio and Y. LeCun. 2014 (cited on page 81).

[144] C.-W. Ko, J. Lee, and M. Queyranne. 'An exact algorithm for maximum entropy sampling'. In: *Operations Research* 43.4 (1995), pp. 684–691 (cited on pages 42, 45).

[145] A. Kolmogorov. 'Die elementare Wahrscheinlichkeitsrechnung'. In: vol. 2. Springer-Verlag, 1933 (cited on page 5).

[146] J. H. Kotecha and P. M. Djuric. 'Gibbs sampling approach for generation of truncated multivariate Gaussian random variables'. In: *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*. Vol. 3. 1999, 1757–1760 vol.3 (cited on page 100).

[147] N. Krämer and P. Hennig. 'Linear-time probabilistic solution of boundary value problems'. In: *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*. 2021 (cited on page 7).

[148] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. 'Near-optimal sensor placements: Maximizing information while minimizing communication cost'. In: *Proceedings of the 5th international conference on Information processing in sensor networks*. 2006, pp. 2–10 (cited on page 60).

[149] A. Krause, A. Singh, and C. Guestrin. 'Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies'. In: *Journal of Machine Learning Research* 9.8 (2008), pp. 235–284 (cited on pages 41–43, 45).

[150] D. P. Kroese, T. Taimre, and Z. I. Botev. *Handbook of Monte Carlo methods*. Vol. 706. John Wiley & Sons, 2013 (cited on pages 6, 51, 52, 95, 101).

[151] M. Kuss, C. E. Rasmussen, and R. Herbrich. 'Assessing approximate inference for binary Gaussian process classification.' In: *Journal of machine learning research* 6.10 (2005) (cited on pages 6, 106).

[152] A. Lagnoux and P. Lezaud. 'Multilevel branching and splitting algorithm for estimating rare event probabilities'. In: *Simulation Modelling Practice and Theory* 72 (2017), pp. 150–167 (cited on page 101).

[153] P. S. Laplace. *Théorie analytique des probabilités*. Courcier, 1820 (cited on page 5).

[154] F. Larkin. 'Gaussian measure in Hilbert space and applications in numerical analysis'. In: *The Rocky Mountain Journal of Mathematics* (1972), pp. 379–421 (cited on page 6).

[155] E. Lawrence, D. Bingham, C. Liu, and V. N. Nair. 'Bayesian inference for multivariate ordinal data using parameter expansion'. In: *Technometrics* 50.2 (2008), pp. 182–191 (cited on page 97).

[156] L. Le Gratiet and J. Garnier. 'Recursive co-kriging model for design of computer experiments with multiple levels of fidelity'. In: *International Journal for Uncertainty Quantification* 4.5 (2014), pp. 365–386 (cited on page 60).

[157] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. 'Gradient-based learning applied to document recognition'. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cited on page 88).

[158] J. Lee. *Introduction to Riemannian manifolds*. Springer, 2018 (cited on page 79).

[159] E. Levina and P. J. Bickel. 'Maximum likelihood estimation of intrinsic dimension'. In: *Advances in neural information processing systems*. 2005, pp. 777–784 (cited on page 116).

[160] S. Levine. 'Reinforcement learning and control as probabilistic inference: Tutorial and review'. In: *arXiv preprint arXiv:1805.00909* (2018) (cited on page 39).

[161] X. Liao, H. Li, and L. Carin. 'Quadratically gated mixture of experts for incomplete data classification'. In: *Proceedings of the 24th International Conference on Machine learning*. ACM. 2007, pp. 553–560 (cited on page 97).

[162] D. V. Lindley. 'On a measure of the information provided by an experiment'. In: *The Annals of Mathematical Statistics* 27.4 (1956), pp. 986–1005 (cited on page 41).

[163] A. F. López-Lopera, F. Bachoc, N. Durrande, and O. Roustant. 'Finite-dimensional Gaussian approximation with linear inequality constraints'. In: *SIAM/ASA Journal on Uncertainty Quantification* 6.3 (2018), pp. 1224–1255 (cited on pages 97, 110).

[164] A. F. López-Lopera, S. John, and N. Durrande. 'Gaussian process modulated cox processes under linear inequality constraints'. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR. 2019, pp. 1997–2006 (cited on page 97).

[165] Y. Ma, R. Garnett, and J. Schneider. 'Active area search via Bayesian quadrature'. In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*. Vol. 33. Proceedings of Machine Learning Research. PMLR, 2014, pp. 595–603 (cited on page 37).

[166] D. J. C. MacKay. 'Information-based objective functions for active data selection'. In: *Neural Computation* 4.4 (July 1992), pp. 590–604 (cited on pages 39–42).

[167] D. J. MacKay. 'Bayesian interpolation'. In: *Neural computation* 4.3 (1992), pp. 415–447 (cited on page 5).

[168] D. J. MacKay. *Information theory, inference and learning algorithms*. Cambridge University Press, 2003 (cited on pages 5, 40, 41, 51).

[169] M. Mahsereci and P. Hennig. 'Probabilistic line searches for stochastic optimization'. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett. 2015, pp. 181–189 (cited on page 7).

[170] G. Malkomes, C. Schaff, and R. Garnett. 'Bayesian optimization for automated model selection'. In: *Proceedings of the Workshop on Automatic Machine Learning*. Ed. by F. Hutter, L. Kotthoff, and J. Vanschoren. Vol. 64. Proceedings of Machine Learning Research. PMLR, June 2016, pp. 41–47 (cited on page 116).

[171] B. Matérn. 'Spatial variation'. In: (1960) (cited on page 20).

[172] R. E. Melchers and A. T. Beck. *Structural reliability analysis and prediction*. John Wiley & Sons, 2018 (cited on page 96).

[173] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. 'Equation of state calculations by fast computing machines'. In: *The journal of chemical physics* 21.6 (1953), pp. 1087–1092 (cited on page 53).

[174] A. Meurer et al. 'SymPy: Symbolic computing in Python'. In: *PeerJ Computer Science* 3 (Jan. 2017), e103 (cited on page 15).

[175] T. Minka. *Deriving quadrature rules from Gaussian processes*. Tech. rep. Statistics Department, Carnegie Mellon University, 2000 (cited on pages 15, 30, 44).

[176] T. P. Minka. 'Expectation propagation for approximate Bayesian inference'. In: *CoRR* abs/1301.2294 (2013) (cited on pages 6, 106).

[177] T. Miwa, A. J. Hayter, and S. Kuriki. 'The evaluation of general non-centred orthant probabilities'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65.1 (2003), pp. 223–234 (cited on page 97).

[178] J. Močkus. 'On Bayesian methods for seeking the extremum'. In: *Optimization Techniques IFIP Technical Conference: Novosibirsk, July 1–7, 1974*. Ed. by G. I. Marchuk. Springer Berlin Heidelberg, 1975, pp. 400–404 (cited on page 7).

[179] J. Močkus. 'Application of Bayesian approach to numerical methods of global and stochastic optimization'. In: *Journal of Global Optimization* 4.4 (1994), pp. 347–365 (cited on page 7).

[180] D. P. Moualeu, M. Weiser, R. Ehrig, and P. Deuflhard. 'Optimal control for a tuberculosis model with undetected cases in Cameroon'. In: *Communications in Nonlinear Science and Numerical Simulation* 20.3 (2015), pp. 986–1003 (cited on page 71).

[181] J. J. Mulgrave and S. Ghosal. 'Bayesian inference in nonparanormal graphical models'. In: *Bayesian Anal.* (2018) (cited on page 97).

[182] I. Murray, R. Adams, and D. MacKay. 'Elliptical slice sampling'. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Vol. 9. PMLR, May 2010, pp. 541–548 (cited on pages 55, 98).

[183] C. A. Naesseth, F. Lindsten, and T. B. Schön. 'Elements of sequential Monte Carlo'. In: *Foundations and Trendső in Machine Learning* 12.3 (2019), pp. 307–392 (cited on page 104).

[184] R. M. Neal. 'Annealed importance sampling'. In: *Statistics and computing* 11.2 (2001), pp. 125–139 (cited on page 56).

[185] R. M. Neal. 'Slice sampling'. In: *Ann. Statist.* 31.3 (June 2003), pp. 705–767 (cited on page 55).

[186] R. M. Neal. 'MCMC using Hamiltonian dynamics'. In: *Published as Chapter 5 of the Handbook of Markov Chain Monte Carlo, 2011* (June 9, 2012) (cited on page 54).

[187] T. Nguyen, S. Gupta, H. Ha, S. Rana, and S. Venkatesh. 'Distributionally robust Bayesian quadrature optimization'. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 1921–1931 (cited on page 36).

[188] N. Nomura. 'Computation of multivariate normal probabilities with polar coordinate systems'. In: *Journal of Statistical Computation and Simulation* 84.3 (2014), pp. 491–512 (cited on page 97).

[189] N. Nomura. 'Evaluation of Gaussian orthant probabilities based on orthogonal projections to subspaces'. In: *Statistics and Computing* 26.1 (Jan. 2016), pp. 187–197 (cited on page 97).

[190] A. O'Hagan. 'Bayes-Hermite quadrature'. In: *Journal of Statistical Planning and Inference* 29 (1991), pp. 245–260 (cited on pages 6, 7, 15, 31).

[191] A. O'Hagan. 'Some Bayesian numerical analysis'. In: *Bayesian Statistics* 4.345–363 (1992), pp. 4–2 (cited on page 6).

[192] C. J. Oates, J. Cockayne, F.-X. Briol, and M. Girolami. 'Convergence rates for a class of estimators based on Steins method'. In: *Bernoulli* 25.2 (2019), pp. 1141–1159 (cited on page 48).

[193] M. Osborne, R. Garnett, Z. Ghahramani, D. K. Duvenaud, S. J. Roberts, and C. E. Rasmussen. 'Active learning of model evidence using Bayesian quadrature'. In: *Advances in Neural Information Processing Systems 25*. 2012, pp. 46–54 (cited on page 34).

[194] M. Osborne, R. Garnett, S. Roberts, C. Hart, S. Aigrain, and N. Gibson. 'Bayesian quadrature for ratios'. In: *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*. Vol. 22. Proceedings of Machine Learning Research. PMLR, 2012, pp. 832–840 (cited on page 36).

[195] A. Pakman and L. Paninski. 'Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians'. In: *Journal of Computational and Graphical Statistics* 23.2 (2014), pp. 518–542 (cited on pages 97, 98, 100).

[196] A. Paleyes, M. Pullin, M. Mahsereci, N. Lawrence, and J. González. 'Emulation of physical processes with emukit'. In: *Second Workshop on Machine Learning and the Physical Sciences, NeurIPS*. 2019 (cited on pages 7, 37).

[197] E. Papaleo, P. Mereghetti, P. Fantucci, R. Grandori, and L. De Gioia. 'Free-energy landscape, principal component analysis, and structural clustering to identify representative conformations from molecular dynamics simulations: The Myoglobin case'. In: *Journal of Molecular Graphics and Modelling* 27.8 (2009), pp. 889–899 (cited on page 89).

[198] S. Paul, K. Chatzilygeroudis, K. Ciosek, J.-B. Mouret, M. Osborne, and S. Whiteson. 'Robust reinforcement learning with Bayesian optimisation and quadrature'. In: *Journal of Machine Learning Research* 21 (2020), pp. 1–31 (cited on page 37).

[199] B. Peherstorfer, K. Willcox, and M. Gunzburger. 'Survey of multifidelity methods in uncertainty propagation, inference, and optimization'. In: *SIAM Review* 60.3 (2018), pp. 550–591 (cited on page 60).

[200] X. Pennec. 'Intrinsic statistics on Riemannian manifolds: Basic tools for geometric measurements'. In: *Journal of Mathematical Imaging and Vision* 25.1 (2006), p. 127 (cited on pages 77, 81, 126).

[201] P. Perdikaris, M. Raissi, A. Damianou, N. D. Lawrence, and G. E. Karniadakis. 'Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling'. In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 473.2198 (2017) (cited on page 61).

[202] K. B. Petersen, M. S. Pedersen, et al. 'The matrix cookbook'. In: *Technical University of Denmark* 7.15 (2008), p. 510 (cited on page 119).

[203] I. Phinikettos and A. Gandy. 'Fast computation of high-dimensional multivariate normal probabilities'. In: *Computational Statistics & Data Analysis* 55.4 (2011), pp. 1521–1529 (cited on pages 97, 98).

[204] R. Piessens, E. de Doncker-Kapenga, C. W. Überhuber, and D. K. Kahaner. *QUADPACK: A subroutine package for automatic integration*. Vol. 1. Springer Science & Business Media, 2012 (cited on pages 30, 32).

[205] H. Poincaré. *Calcul des probabilités*. Vol. 1. Gauthier-Villars, 1912 (cited on page 6).

[206] L. Pronzato. 'Performance analysis of greedy algorithms for minimising a Maximum Mean Discrepancy'. In: *arXiv preprint arXiv:2101.07564* (2021) (cited on pages 36, 45).

[207] J. Prüher and S. Särkkä. 'On the use of gradient information in Gaussian process quadratures'. In: *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE. 2016, pp. 1–6 (cited on page 36).

[208] J. Prüher and M. Šimandl. 'Bayesian quadrature in nonlinear filtering'. In: *2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*. Vol. 1. IEEE. 2015, pp. 380–387 (cited on page 36).

[209] J. Prüher and M. Šimandl. 'Bayesian quadrature variance in sigma-point filtering'. In: *Informatics in Control, Automation and Robotics 12th International Conference, ICINCO 2015 Colmar, France, July 21-23, 2015 Revised Selected Papers*. Springer. 2016, pp. 355–370 (cited on page 36).

[210] J. Prüher, F. Tronarp, T. Karvonen, S. Särkkä, and O. Straka. 'Student-t process quadratures for filtering of non-linear systems with heavy-tailed noise'. In: *2017 20th International Conference on Information Fusion (Fusion)*. IEEE. 2017, pp. 1–8 (cited on page 36).

[211] C. E. Rasmussen and Z. Ghahramani. 'Bayesian Monte Carlo'. In: *Advances in Neural Information Processing Systems 15*. Max-Planck-Gesellschaft. Cambridge, MA, USA: MIT Press, 2003, pp. 489–496 (cited on pages 47, 48, 51).

[212] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for machine learning*. Vol. 2. Adaptive Computation and Machine Learning. Cambridge, MA, USA: MIT Press, 2006, p. 248 (cited on pages 17–20, 35, 97, 98, 106).

[213] D. J. Rezende, S. Mohamed, and D. Wierstra. 'Stochastic backpropagation and approximate inference in deep generative models'. In: *Proceedings of the 31th International Conference on Machine Learning*. Vol. 32. JMLR Workshop and Conference Proceedings. JMLR.org, 2014, pp. 1278–1286 (cited on page 81).

[214] C. P. Robert, G. Casella, and G. Casella. *Monte Carlo statistical methods*. Vol. 2. Springer, 2004 (cited on page 52).

[215] C. P. Robert. 'Simulation of truncated normal variables'. In: *Statistics and Computing* 5.2 (June 1995), pp. 121–125 (cited on page 100).

[216] S. Ross. *Simulation*. Knovel Library. Elsevier Science, 2012 (cited on pages 95, 101).

[217] C. Runge. 'Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten'. In: *Zeitschrift für Mathematik und Physik* 46.224-243 (1901), p. 20 (cited on page 31).

[218] A. Sard. 'Best approximate integration formulas; best approximation formulas'. In: *American Journal of Mathematics* 71.1 (1949), pp. 80–91 (cited on page 32).

[219] S. Särkkä. *Bayesian filtering and smoothing*. 3. Cambridge University Press, 2013 (cited on page 31).

[220] S. Särkkä, J. Hartikainen, L. Svensson, and F. Sandblom. 'On the relation between Gaussian process quadratures and sigma-point methods'. English. In: *Journal of Advances in Information Fusion* 11.1 (June 2016), pp. 31–46 (cited on page 36).

[221] J. Schmidt, N. Krämer, and P. Hennig. *A probabilistic state space model for joint inference from differential equations and data*. 2021 (cited on page 71).

[222] M. Schober, D. K. Duvenaud, and P. Hennig. 'Probabilistic ODE solvers with Runge-Kutta means'. In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger. Vol. 27. Curran Associates, Inc., 2014 (cited on page 21).

[223] M. Schober, S. Särkkä, and P. Hennig. 'A probabilistic model for the numerical solution of initial value problems'. In: *Statistics and Computing* 29.1 (2019), pp. 99–122 (cited on page 7).

[224] I. J. Schoenberg. 'Contributions to the problem of approximation of equidistant data by analytic functions: Part A. On the problem of smoothing or graduation. A first class of analytic approximation formulae'. In: *Quarterly of Applied Mathematics* 4.1 (1946), pp. 45–99 (cited on page 30).

[225] I. J. Schoenberg. 'Contributions to the problem of approximation of equidistant data by analytic functions. Part B. On the problem of osculatory interpolation. A second class of analytic approximation formulae'. In: *Quarterly of Applied Mathematics* 4.2 (1946), pp. 112–141 (cited on page 30).

[226] B. Schölkopf, R. Herbrich, and A. J. Smola. 'A generalized representer theorem'. In: *International conference on computational learning theory*. Springer. 2001, pp. 416–426 (cited on page 26).

[227] S. L. Seyler, A. Kumar, M. F. Thorpe, and O. Beckstein. 'Path similarity analysis: A method for quantifying macromolecular pathways'. In: *PLOS Computational Biology* 11.10 (Oct. 2015), pp. 1–37 (cited on page 89).

[228] A. Shah, A. Wilson, and Z. Ghahramani. 'Student-t processes as alternatives to Gaussian processes'. In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*. Ed. by S. Kaski and J. Corander. Vol. 33. Proceedings of Machine Learning Research. PMLR, Apr. 2014, pp. 877–885 (cited on pages 22, 36).

[229] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. 'Taking the human out of the loop: a review of Bayesian optimization'. In: *Proceedings of the IEEE* 104.1 (Dec. 2016), p. 28 (cited on pages 7, 42).

[230] C. E. Shannon. 'A mathematical theory of communication'. In: *The Bell system technical journal* 27.3 (1948), pp. 379–423 (cited on page 40).

[231] J. Skilling. 'Bayesian numerical analysis'. In: *Physics and Probability: Essays in Honor of Edwin T. Jaynes*. Ed. by W. T. Grandy Jr and P. W. Milonni. Cambridge University Press, 1993, pp. 207–222 (cited on page 6).

[232] J. Skilling. 'Nested sampling'. In: *AIP Conference Proceedings*. Vol. 735. 1. American Institute of Physics. 2004, pp. 395–405 (cited on pages 6, 56).

[233] M. T. Smith, M. A. Álvarez, and N. D. Lawrence. 'Gaussian process regression for binned data'. In: *arXiv preprint arXiv:1809.02010* (2018) (cited on page 37).

[234] E. Snelson, C. E. Rasmussen, and Z. Ghahramani. 'Warped Gaussian processes'. In: *Advances in neural information processing systems* 16 (2004), pp. 337–344 (cited on page 33).

[235] N. Spellmon, X. Sun, N. Sirinupong, B. Edwards, C. Li, and Z. Yang. 'Molecular dynamics simulation reveals correlated inter-lobe motion in protein Lysine Methyltransferase SMYD2'. In: *PLOS ONE* 10.12 (2015), e0145758 (cited on page 89).

[236] M. L. Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 2012 (cited on page 20).

[237] I. Steinwart and A. Christmann. *Support vector machines*. Springer Science & Business Media, 2008 (cited on pages 25, 27).

[238] D. Straub, R. Schneider, E. Bismut, and H.-J. Kim. 'Reliability analysis of deteriorating structural systems'. In: *Structural Safety* 82 (2020), p. 101877 (cited on page 96).

[239] Q. Su, X. Liao, C. Chen, and L. Carin. 'Nonlinear statistical learning with truncated Gaussian graphical models'. In: *Proceedings of The 33rd International Conference on Machine Learning*. Vol. 48. PMLR, June 2016, pp. 1948–1957 (cited on page 96).

[240] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018 (cited on page 39).

[241] K. Tanaka. *Kernel quadrature by applying a point-wise gradient descent method to discrete energies*. 2021 (cited on page 46).

[242] R. Thiébaut and H. Jacqmin-Gadda. 'Mixed models for longitudinal left-censored repeated measures'. In: *Computer Methods and Programs in Biomedicine* 74.3 (2004), pp. 255–260 (cited on page 96).

[243] M. Titsias. 'Variational learning of inducing variables in sparse Gaussian processes'. In: *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*. Ed. by D. van Dyk and M. Welling. Vol. 5. Proceedings of Machine Learning Research. PMLR, Apr. 2009, pp. 567–574 (cited on page 19).

[244] A. Tosi, S. Hauberg, A. Vellido, and N. D. Lawrence. 'Metrics for probabilistic geometries'. In: *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, UAI 2014, Quebec City, Quebec, Canada, July 23-27, 2014*. Ed. by N. L. Zhang and J. Tian. AUAI Press, 2014, pp. 800–808 (cited on page 81).

[245] J. Townsend, N. Koep, and S. Weichwald. 'Pymanopt: A Python toolbox for optimization on manifolds using automatic differentiation'. In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 4755–4759 (cited on page 130).

[246] G. A. Tribello and P. Gasparotto. 'Using dimensionality reduction to analyze protein trajectories'. In: *Frontiers in Molecular Biosciences* 6 (2019), p. 46 (cited on page 89).

[247] F. Tronarp, T. Karvonen, and S. Särkkä. 'Mixture representation of the Matérn class with applications in state space approximations and Bayesian quadrature'. In: *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE. 2018, pp. 1–6 (cited on page 24).

[248] C. F. Van Loan. 'The ubiquitous Kronecker product'. In: *Journal of computational and applied mathematics* 123 (2000) (cited on page 62).

[249] A. Vehtari, A. Gelman, D. Simpson, B. Carpenter, and P.-C. Bürkner. 'Rank-normalization, folding, and localization: An improved $\hat{R}$ for assessing convergence of MCMC (with discussion)'. In: *Bayesian Analysis* 16.2 (June 2021) (cited on page 56).

[250] W. P. Vijverberg. 'Monte Carlo evaluation of multivariate normal probabilities'. In: *Journal of Econometrics* 76.1 (1997), pp. 281–307 (cited on page 97).

[251] P. Virtanen et al. 'SciPy 1.0: Fundamental algorithms for scientific computing in Python'. In: *Nature Methods* 17 (2020), pp. 261–272 (cited on page 32).

[252] E. Wagstaff, S. Hamid, and M. Osborne. 'Batch selection for parallelisation of Bayesian quadrature'. In: *arXiv preprint arXiv:1812.01553* (2018) (cited on pages 37, 47, 86).

[253] G. Wahba. *Spline models for observational data*. SIAM, 1990 (cited on pages 21, 30).

[254] J. Wang, S. C. Clark, E. Liu, and P. I. Frazier. 'Parallel Bayesian global optimization of expensive functions'. In: *Operations Research* 68.6 (2020), pp. 1850–1865 (cited on page 97).

[255] J. Wang, J. Cockayne, O. A. Chkrebtii, T. J. Sullivan, and C. J. Oates. 'Bayesian numerical methods for nonlinear partial differential equations'. In: *Stat. Comput.* 31.5 (2021), p. 55 (cited on page 7).

[256] O. Wani, A. Scheidegger, J. P. Carbajal, J. Rieckermann, and F. Blumensaat. 'Parameter estimation of hydrologic models using a likelihood function for censored and binary observations'. In: *Water Research* 121 (2017), pp. 290–301 (cited on page 96).

[257] D. Weichert and A. Kister. 'Bayesian optimization for min max optimization'. In: *arXiv preprint arXiv:2107.13772* (2021) (cited on page 101).

[258] H. Wendland. *Scattered data approximation*. Vol. 17. Cambridge University Press, 2004 (cited on page 21).

[259] H. Wendland and C. Rieger. 'Approximate interpolation with applications to selecting smoothing parameters'. In: *Numerische Mathematik* 101.4 (2005), pp. 729–748 (cited on page 26).

[260] J. Wenger and P. Hennig. 'Probabilistic linear solvers for machine learning'. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 6731–6742 (cited on page 7).

[261] J. Wenger, N. Krämer, M. Pförtner, J. Schmidt, N. Bosch, N. Effenberger, J. Zenn, A. Gessner, T. Karvonen, F.-X. Briol, M. Mahsereci, and P. Hennig. *ProbNum: Probabilistic numerics in Python*. 2021 (cited on page 37).

[262] C. Withers. 'The moments of the multivariate normal'. In: *Bulletin of the Australian Mathematical Society* 32.1 (1985), pp. 103–107 (cited on page 34).

[263] A. Wolf and K. N. Kirschner. 'Principal component and clustering analysis on molecular dynamics data of the ribosomal L11·23S subdomain'. In: *Journal of Molecular Modeling* 19.2 (2013), pp. 539–549 (cited on page 89).

[264] A. Wu, M. C. Aoi, and J. W. Pillow. 'Exploiting gradients and Hessians in Bayesian optimization and Bayesian quadrature'. In: *arXiv preprint arXiv:1704.00060* (2017) (cited on page 36).

[265] X. Xi, F.-X. Briol, and M. Girolami. 'Bayesian quadrature for multiple related integrals'. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. G. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 5373–5382 (cited on pages 60, 63, 85).

[266] F. Yousefi, M. T. Smith, and M. A. Álvarez. 'Multi-task learning for aggregated data using Gaussian processes'. In: *arXiv preprint arXiv:1906.09412* (2019) (cited on page 37).

[267] L. Zhang, B. Carpenter, A. Gelman, and A. Vehtari. 'Pathfinder: Parallel quasi-Newton variational inference'. In: *arXiv preprint arXiv:2108.03782* (2021) (cited on page 56).

[268] S. Zhou, P. Giulani, J. Piekarewicz, A. Bhattacharya, and D. Pati. 'Reexamining the proton-radius problem using constrained Gaussian processes'. In: *Phys. Rev. C* 99 (5 May 2019), p. 055202 (cited on page 96).

[269] T. Zhou and Y. Peng. 'Adaptive Bayesian quadrature based statistical moments estimation for structural reliability analysis'. In: *Reliability Engineering & System Safety* 198 (2020), p. 106902 (cited on page 37).

[270] H. Zhu, X. Liu, R. Kang, Z. Shen, S. Flaxman, and F.-X. Briol. 'Bayesian probabilistic numerical integration with tree-based models'. In: (June 9, 2020) (cited on page 36).