Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

PROCEEDINGS A

rspa.royalsocietypublishing.org

Research

CrossMark
click for updates

Cite this article: Hennig P, Osborne MA, Girolami M. 2015 Probabilistic numerics and uncertainty in computations. *Proc. R. Soc. A* **471**: 20150142. http://dx.doi.org/10.1098/rspa.2015.0142

Received: 2 March 2015
Accepted: 3 June 2015

Subject Areas:
statistics, computational mathematics, artificial intelligence

Keywords:
numerical methods, probability, inference, statistics

Author for correspondence:
Philipp Hennig
e-mail: phennig@tue.mpg.de

THE ROYAL SOCIETY
PUBLISHING

Probabilistic numerics and uncertainty in computations

Philipp Hennig[1], Michael A. Osborne[2] and Mark Girolami[3]

[1]Department of Empirical Inference, Max Planck Institute for Intelligent Systems, Tübingen, Germany
[2]Department of Engineering Science, University of Oxford, Oxford, UK
[3]Department of Statistics, University of Warwick, Warwick, UK

We deliver a call to arms for *probabilistic numerical methods*: algorithms for numerical tasks, including linear algebra, integration, optimization and solving differential equations, that return uncertainties in their calculations. Such uncertainties, arising from the loss of precision induced by numerical calculation with limited time or hardware, are important for much contemporary science and industry. Within applications such as climate science and astrophysics, the need to make decisions on the basis of computations with large and complex data have led to a renewed focus on the management of numerical uncertainty. We describe how several seminal classic numerical methods can be interpreted naturally as probabilistic inference. We then show that the probabilistic view suggests new algorithms that can flexibly be adapted to suit application specifics, while delivering improved empirical performance. We provide concrete illustrations of the benefits of probabilistic numeric algorithms on real scientific problems from astrometry and astronomical imaging, while highlighting open problems with these new algorithms. Finally, we describe how probabilistic numerical methods provide a coherent framework for identifying the uncertainty in calculations performed with a combination of numerical algorithms (e.g. both numerical optimizers and differential equation solvers), potentially allowing the diagnosis (and control) of error sources in computations.

© 2015 The Authors. Published by the Royal Society under the terms of the Creative Commons Attribution License http://creativecommons.org/licenses/by/4.0/, which permits unrestricted use, provided the original author and source are credited.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

1. Introduction

Probability theory is the quantitative framework for scientific inference [1]. It codifies how observations (data) combine with modelling assumptions (prior distributions and likelihood functions) to give evidence for or against a hypothesis and values of unknown quantities. There is continuing debate about how prior assumptions can be chosen and validated (e.g. [2, §1.2.3]). However, the role of probability as the language of uncertainty is rarely questioned; that is, as long as the subject of inference is a physical variable. What if the quantity in question is a mathematical statement, the solution to a computational task? Does it make sense to assign a probability measure $p(x)$ over the solution of a linear system of equations $Ax = b$ if A and b are known? If so, what is the meaning of $p(x)$, and can it be identified with a notion of 'uncertainty'? If one sees the use of probability in statistics as a way to remove 'noise' from 'signal', it seems misguided to apply it to a deterministic mathematical problem. But noise and stochasticity are themselves difficult to define precisely. Probability theory does not rest on the notion of randomness (*aleatory* uncertainty), but extends to quantifying *epistemic* uncertainty, arising solely from missing information.[1] Connections between deterministic computations and probabilities have a long history. Erdös & Kac [3] showed that the number of distinct prime factors in an integer follows a normal distribution. Their statement is precise, and useful for the analysis of factorization algorithms [4], even though it is difficult to 'sample' from the integers. It is meaningful without appealing to the concept of epistemic uncertainty. Probabilistic and deterministic methods for inference on physical quantities have shared dualities from very early on: Legendre introduced the method of least-squares in 1805 as a deterministic *best fit* for data without a probabilistic interpretation. Gauss' 1809 probabilistic formulation of the exact same method added a generative stochastic model for how the data might be assumed to have arisen. Legendre's least-squares is a useful method without the generative interpretation, but the Gaussian formulation adds the important notion of uncertainty (also interpretable as model capacity) that would later become crucial in areas such as the study of dynamical systems.[2]

Several authors [12–14] have shown that the language of probabilistic inference can be applied to numerical problems, using a notion of uncertainty about the result of an intractable or incomplete computation, and giving rise to methods we will here call *probabilistic numerics*.[3] In such methods, uncertainty regularly arises solely from the lack of information inherent in the solution of an 'intractable' problem: a quadrature method, for example, has access only to finitely many function values of the integrand; an exact answer would, in principle, require infinitely many such numbers. Algorithms for problems such as integration and optimization proceed iteratively, each iteration providing information improving a running estimate for the correct answer. Probabilistic numerics provides methods that, in place of such estimates, update probability measures over the space of possible solutions. As Diaconis noted [12], it appears that Poincaré proposed such an approach already in the nineteenth century. The recent explosive growth of automated inference, and the increasing importance of numerics for science, has given this idea new urgency.

This article connects recent results, promising applications and central questions for probabilistic numerics. We collate results showing that a number of basic, popular numerical methods can be identified with families of probabilistic inference procedures. The probability measures arising from this new interpretation of established methods can offer improved

[1] Many resources discussing epistemic uncertainty can be found, at the time of writing, at http://understanduncertainty.org (the authors are not affiliated with this web page).

[2] In fact, the very same connection between least-squares estimation and Gaussian inference has been re-discovered repeatedly, simply because least-squares estimation has been re-discovered repeatedly after Legendre, under names such as ridge regression [5], Kriging [6], Tikhonov's method [7] and so on. The fundamental connection is that the normal distribution is the exponential of the square ℓ_2 norm. Because the exponential is a monotonic function, minimizing an ℓ_2-regularized ℓ_2 loss is equivalent to maximizing the product of Gaussian prior and likelihood. In this sense, this paper is adding numerical mathematicians to the list as yet another group of re-discoverers of Gaussian inference. Ironically, this list includes Gauss himself, because Gaussian elimination, introduced in the very same paper as the Gaussian distribution itself [8], can be interpreted as a conjugate-direction method [9], and thus as Gaussian regression [10]. See also [11].

[3] The probabilistic numerics community website can be found at http://www.probabilistic-numerics.org.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

performance, enticing new functionality and conceptual clarity; we demonstrate this with examples drawn from astrometry and computational photography. The article closes by pointing out the propagation of uncertainty through computational pipelines as a guiding goal for probabilistic numerics.

It may be helpful to separate the issues discussed here clearly from other areas of overlap between statistics and numerical mathematics: we here focus on well-posed deterministic problems, identifying degrees of uncertainty arising from the computation itself. This is in contrast to the notion of uncertainty quantification (e.g. [15,16]), which identifies degrees of freedom in ill-posed problems, and where epistemic uncertainty arises from the set-up of the computation, rather than the computation itself. It also differs from several concepts of stochastic numerical methods, which use (aleatory) random numbers either to quantify uncertainty from repeated computations (e.g. [17]) or to reduce computational cost through randomly chosen projections (see, for example, [18,19]).

It is also important to note that, as a matter of course, existing frameworks already analyse and estimate the error created by a numerical algorithm. Theoretical *analysis* of computational errors generally yields convergence rates—bounds up to an unknown constant—made under certain structural assumptions. Where probabilistic numerical methods are derived from classic ones, as described below, they naturally inherit such analytical properties. At runtime, the error is also *estimated* for the specific problem instance. Such runtime error estimation is frequently performed by monitoring the dynamics of the algorithm's main estimate (see [20, §II.4, for ODEs], [21, §4.5, for quadrature]). Similarly, in optimization problems, the magnitude of the gradient is often used to monitor the algorithm's progress. Such error estimates are informal, as are the solution estimates computed by the numerical method itself. They are meant to be used locally, mostly as criteria for the termination of a method and the adaptation of its internal parameters. They cannot typically be interpreted as a property (e.g. the variance) of a posterior probability measure, and thus cannot be communicated to other algorithms, and thus cannot be embedded in a larger framework of error estimation. They also do not usually inform the design of the numerical algorithm itself; instead, they are a diagnostic tool added *post hoc*. Below, we argue that the estimation of errors should be given a formal framework, and that probability theory is uniquely suited for this task. Describing numerical computations as inference on a latent quantity yields a joint, consistent, framework for the construction of solution and error estimates. The inference perspective can provide a natural intuition that may suggest extensions and improvements. And the probabilistic framework provides a *lingua franca* for numerical computations, which allows the communication of uncertainty between methods in a chain of computation.

2. Probabilistic numerical methods from classical ones

Numerical algorithms estimate quantities not directly computable, using the results of more readily available computations. Even existing numerical methods can thus be seen as inference rules, reasoning about latent quantities from 'observables' or 'data'. At face value, this connection between inference and computation is vague. But several recent results have shown that it can be made rigorous, such that established deterministic rules for various numerical problems can be understood as maximum *a posteriori* estimates under specific priors (hypothesis classes) and likelihoods (observation models). The recurring picture is that there is a one-to-many relationship between a classic numerical method for a specific task and a family of probabilistic priors which give the same maximum posterior estimate but differing measures of uncertainty. Choosing one member of this family amounts to fitting an uncertainty, a task we call *uncertainty calibration*. The result of this process is a numerical method that returns a point estimate surrounded by a probability measure of uncertainty, such that the point estimate inherits the proved theoretical properties of the classic method, and the uncertainty offers new functionality.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

(a) Quadrature

We term the probabilistic numeric approach to quadrature *Bayesian quadrature.* Diaconis [12] may have been first to point out a clear connection between a Gaussian process regression model and a deterministic quadrature rule, an observation subsequently generalized by Wahba [22, §8] and O'Hagan [23], and also noted by [24]. Details can be found in these works; here we construct an intuitive example highlighting the practical challenges of assigning uncertainty to the result of a computation. For concreteness, consider $f(x) = \exp[-\sin^2(3x) - x^2]$ (black in figure 1*a*). Evidently, f has a compact symbolic form and $f(x)$ can be computed for virtually any $x \in \Re$ in nanoseconds. It is a wholly deterministic object. Nevertheless, the real number

$$F = \int_{-3}^{3} f(x)\,\mathrm{d}x \tag{2.1}$$

has no simple analytic value, in the sense that it cannot be natively evaluated in low-level code. Quadrature rules offer 'black box' estimates of F. These rules have been optimized so heavily (e.g. [21]) that they could almost be called 'low level', but their results do not come with the strict error bounds of floating-point operations; instead, assumptions about f are necessary to bound error. Perhaps the simplest quadrature rule is the trapezoid rule, which amounts to linear interpolation of f (red line in figure 1*a*(i)): evaluate $f(x_i)$ on a grid $-3 = x_1 < x_2 < \cdots < x_N = 3$ of N points, and compute

$$\hat{F}_{\text{midpoint}} = \sum_{i=2}^{N} \frac{1}{2}[f(x_i) + f(x_{i-1})](x_i - x_{i-1}). \tag{2.2}$$

(b) Bayes–Hermite quadrature

A probabilistic description of F requires a joint probability distribution over f and F. Since f lies in an infinite-dimensional Banach or Hilbert space, no Lebesgue measure can be defined. But Gaussian process measures can be well-defined over such spaces, and offer a powerful framework for quadrature [22,25]. In particular, we may choose to model f on $[-3, 3]$ by $p(f) = \mathcal{GP}(f; 0, k)$, a Gaussian process with vanishing mean $\mu = 0$ and the linear spline covariance function [26] (a stationary variant of the Wiener process)

$$k(x, x') = c(1 + b - \tfrac{1}{3}b|x - x'|), \quad \text{for some } c, b > 0. \tag{2.3}$$

More precisely, this assigns a probability measure over the function values $f(x)$ on $[-3, 3]$, such that any restriction to finitely many evaluations $\boldsymbol{y} = [f(x_1), \ldots, f(x_M)]$ at $X = [x_1, \ldots, x_N]$ is jointly Gaussian distributed with zero mean and covariance $\mathrm{cov}[f(x_i), f(x_j)] = k(x_i, x_j)$. Samples drawn from this process are shown in grey in figure 1*a*(i). These sample paths represent the hypothesis class associated with this Gaussian process prior: they are continuous, but not differentiable, in mean square expectation [27, §2.2]. Because Gaussian processes are closed under linear projections (e.g. [27, p. 27]), this distribution over f is identified with a corresponding univariate Gaussian distribution [26],

$$p(F) = \mathcal{N}\left[F; 0, \iint_{-3}^{3} k(\tilde{x}, \tilde{x}')\,\mathrm{d}\tilde{x}\,\mathrm{d}\tilde{x}' = c\left(1 + \frac{b}{3}\right)\right], \tag{2.4}$$

on F. The strength of this formulation is that it provides a clear framework under which observations $f(x_i)$ can be incorporated. The measure on $p(F)$ conditioned on the collected function values is another scalar Gaussian distribution

$$p(F \mid \boldsymbol{y}) = \mathcal{N}\left[F; \int_{-3}^{3} k(\tilde{x}, X)K^{-1}\boldsymbol{y}\,\mathrm{d}\tilde{x}, \iint_{-3}^{3} k(\tilde{x}, \tilde{x}') - k(\tilde{x}, X)K^{-1}k(X, \tilde{x}')\,\mathrm{d}\tilde{x}\,\mathrm{d}\tilde{x}'\right], \tag{2.5}$$

where $k(\tilde{x}, X) = [k(\tilde{x}, x_1), \ldots, k(\tilde{x}, x_N)]$ and K is the symmetric positive definite matrix with elements $K_{ij} = k(x_i, x_j)$. The final expression in the brackets, the posterior covariance, can be

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

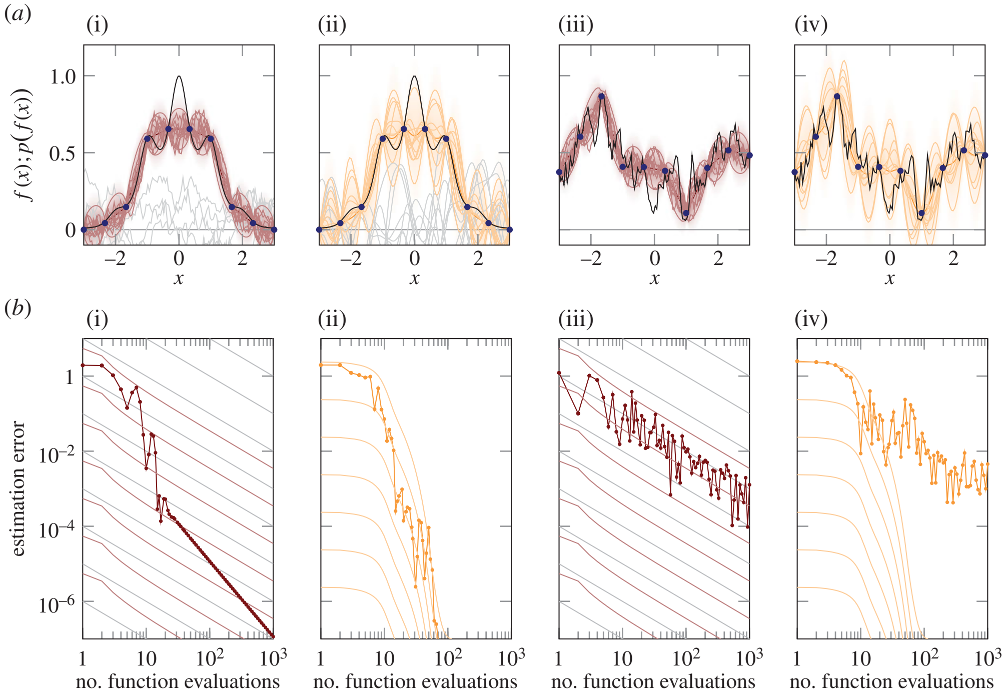

Figure 1. Quadrature rules, illustrating the challenge of uncertainty calibration. (*a*(i)(ii)) Function, $f(x)$ (black line), is approximately integrated using two different Gaussian process priors (*a*(i) linear spline; *a*(ii) exponentiated quadratic), giving posterior distributions and mean estimates. Grey lines are functions sampled from the prior. The thick coloured line is the posterior mean, thin lines are posterior samples and the delineation of two marginal standard deviations. The shading represents posterior probability density. (*b*(i)(ii)) As the number of evaluation points increases, the posterior mean (thick line with points) converges to the true integral value; note the more rapid convergence of the exponentiated-quadratic prior. The posterior covariance provides an error estimate whose scale is defined by the posterior mean alone (each thin coloured line in the plots corresponds to a different instance of such an estimate). But it is only a meaningful error estimate if it is matched well to the function's actual properties. (*b*(i)) shows systematic difference between the convergence of the real error and the convergence of the estimated error under the linear spline, whereas convergence of the estimated error under the exponentiated-quadratic prior is better calibrated to the real error. Grey grid lines in the background, bottom left, correspond to $\mathcal{O}(N^{-1})$ convergence of the error in the number N of function evaluations. (*a*(iii)(iv),*b*(iii)(iv)) The same experiment repeated with a function f drawn from the spline kernel prior. For this function, the trapezoid rule is the optimal statistical estimator of the integral (note well-calibrated error measure in *b*(iii)), while the Gaussian kernel GP is strongly over-confident.

optimized with respect to X to *design* a 'most informative' dataset. For the spline kernel (2.3), this leads to placing X on a regular, equidistant, grid [26].

The spline kernel k of (2.3) is a piecewise linear function with a single point of non-differentiability at $x = x'$. The posterior mean $k(\tilde{x}, X)K^{-1}\boldsymbol{y}$ is a weighted sum of N such kernels centred on the evaluation points X and constrained by the likelihood to pass through the values $\boldsymbol{y}$ at the nodes X. Thus, the posterior mean is a linear spline interpolant of the evaluations, and the posterior mean of equation (2.5) is exactly equal to the trapezoid-rule estimate from X (see also [12]). That is, Bayesian quadrature with a linear spline prior provides a probabilistic interpretation of the trapezoidal rule; it supplements the estimate with a full probability distribution characterizing uncertainty. The corresponding conditional on f is a Gaussian process whose mean is the piecewise linear red function in figure 1*a*(i) (the figure also represents the conditional distribution $p(f \mid f(x_1), \dots, f(x_N))$ as a red cloud, and some samples). Various other, more elaborate, quadrature rules (e.g. higher order spline interpolation and Chebyshev polynomials) can be cast probabilistically in analogous ways, simply by changing the covariance function k [12,22,23].

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

(c) Added value, and challenge, of probabilistic output

A non-probabilistic analysis of the trapezoid rule is that, for sufficiently regular functions f, this rule converges at a rate of at least $\mathcal{O}(N^{-1})$ to the correct value for F as N increases (e.g. [21, eq. 2.1.6]) (other quadrature rules, like cubic splines, have better convergence for differentiable functions). Figure 1*b*(i) shows the mean estimate (equivalently, the trapezoid rule) as a thick line. Grey help lines represent $\mathcal{O}(N^{-1})$ convergence. Clearly, the asymptotic behaviour is steeper than those lines.[4] So the theoretical bound is correct, but it is also of little practical value: no linear bound is tight from start to convergence.

On the probabilistic side, the standard deviation of the conditional (2.5) is regularly interpreted as an estimate of the imprecision of the mean estimate. In fact, this error estimate (thin red lines in figure 1*a*(i)) is analogous to the deterministic linear convergence bound for continuous functions. There is a family of such error bounds associated with the same posterior mean, with each line corresponding to a different value for the unknown constant in the bound and different values for the parameters b and c in the covariance.[5] It may seem as though the probabilistic interpretation had added nothing new, but since this view identifies quadrature rules of varying assumptions as parameter choices in Gaussian process regression, it embeds seemingly separate rules in a hierarchical space of models, from which models with good error modelling can be selected by hierarchical probabilistic inference. This can be done without collecting additional data points [25, §5].

More generally, the probabilistic numeric viewpoint provides a principled way to manage the parameters of numerical procedures. Where Markov chain Monte Carlo procedures might require the hand-tuning of parameters such as step sizes and annealing schedules, Bayesian quadrature allows the machinery of statistical inference procedure to be brought to bear upon such parameters. A practical example of the benefits of approximate Bayesian inference for the (hyper-)parameters of a Bayesian quadrature procedure is given by [28].

The function f used in this example is much smoother than typical functions under the Gaussian process prior distribution associated with the trapezoid rule (shown as thin grey samples in figure 1*a*(i)). Figure 1*a*(ii),*b*(ii) (using orange) shows analogous experiments with the exponentiated-quadratic covariance function $k(x,x') = \theta^2 \exp(-(x-x')^2/\lambda^2)$, corresponding to a very strong smoothness assumption on f [29] (see grey samples in figure 1*a*(ii)), giving a very quickly converging estimate. In this case, the 'error bars' provided by the standard deviation converge in a qualitatively comparable way.

Of course, the faster convergence of this quadrature rule based on the exponentiated-quadratic covariance prior is not a universal property. It is the effect of a much stronger set of prior assumptions. If the true integrand is rougher than expected under this prior, the quadrature estimate arising from this prior can be quite wrong. Figure 1*a*(iii)(iv),*b*(iii)(iv) shows analogous experiments on an integrand that is a true sample from a Gaussian process with the spline covariance (2.3). In this case, the spline prior is the optimal statistical estimator by construction, and its error estimate is perfectly calibrated (figure 1*a*(iii),*b*(iii)), while the exponentiated-quadratic kernel gives over-confident, and inefficient, estimates (figure 1*a*(iv),*b*(iv)).

Identifying the optimal regression model from a larger class, just based on the collected function values, requires more computational work than to fix a regressor from the start. But it also gives better calibrated uncertainty. Contemporary general-purpose quadrature implementations (e.g. [21]) remain lightweight by recursively re-using previous computations. The above experiments show that it is possible to design Bayesian quadrature rules with well-calibrated posterior error estimates, but it remains a question how small the computational overhead from a probabilistic computation over these methods can be made. Even so, formulating quadrature as probabilistic regression precisely captures a trade-off between *prior assumptions*

[4] This, too, is a well-known result: if the integrand is differentiable, rather than just continuous, the trapezoid rule has quadratic-convergence rate [21, eq. 2.1.12].

[5] The initial behaviour of the red lines in the figure is a function of the scale b, which relates to assumptions about the rate at which the asymptotic quadratic-convergence is approached.

inherent in a computation and the computational effort required in that computation to achieve a certain precision. Computational rules arising from a strongly constrained hypothesis class can perform much better than less restrictive rules *if the prior assumptions are valid*. In the numerical setting—in contrast to many empirical situations in statistics—it is often possible to precisely check whether a particular prior assumption is valid: the machine performing the computation has access, at least in principle, to a formal, complete, description of its task, in the form of the source code describing the task (the integrand, the optimization objective, etc.). Using this source code, it is possible, for example, to test, at runtime, whether, and how many times, an integrand is continuously differentiable (e.g. [30]). Using this information will in future allow the design of improved quadrature methods. Once one knows that general-purpose quadrature methods effectively use a Gaussian process prior over function values, it is natural to ask whether this prior actually incorporates the salient information available for one's specific problem. Including such information in the prior leads to customized, 'tailored', numerical methods that can perform better.

Probabilistic inference also furnishes the framework required to tackle numerics tasks using decision theory. For quadrature, the decision problem to be solved is that of node selection: that is, the determination of the optimal positions for points (or nodes) at which to evaluate the integrand. With the definition of an appropriate loss function, such as the posterior variance of the integral, such nodes can be optimally selected by minimizing the expected loss function. It seems clear that this approach can improve upon the simple uniform grids of many traditional quadrature methods, and can enable active learning: where surprising evaluations can influence the selection of future nodes.

This decision-theoretic approach stands in contrast with that adopted by randomized, Monte Carlo, approaches to quadrature. Such approaches generate nodes with the use of a pseudo-random number generator, injecting additional epistemic uncertainty (about the value of the generator's outputs) into a procedure designed to reduce the uncertainty in an integral. It is worth noting that the use of pseudo-random generators burdens the procedure with additional computational overhead: pseudo-random numbers are cheap, but not free. The principal feature of Monte Carlo approaches is their conservative nature: the Monte Carlo policy will always, eventually, take an additional node arbitrarily close to an existing node. The disadvantage of this strategy is its waste of valuable evaluations: the convergence rate of Monte Carlo techniques, $\mathcal{O}(N^{-1/2})$, is clearly improved upon by both traditional quadrature and Bayesian quadrature methods. This problem is only worsened by the common discarding of evaluations known as 'burn-in' and 'thinning'. The advantage of Monte Carlo, of course, is its robustness to even highly non-smooth integrands. However, Bayesian quadrature can realize more value from evaluations by exploiting known structure (e.g. smoothness) in the integrand.

(d) Empirical evaluation of Bayesian quadrature for astrometry

We illustrate this on an integration problem drawn from astrometry, the measurement of the motion of stars. In order to validate astrometric analysis, we aim to recover the number of planets present in synthetic data generated (with a known number of planets) to mimic that produced by an astrometric facility such as the GAIA satellite [31].[6] Here, quadrature's task is to compute the model evidence (marginal likelihood) $Z = \int p(\mathcal{D} \mid \theta, \mathcal{M}) p(\theta \mid \mathcal{M}) \, \mathrm{d}\theta$ of a model $\mathcal{M}$ for orbital motions, where $\mathcal{D}$ are the gathered observations and θ are the model parameters. Specifically, it is of interest to compare the evidence for models including differing numbers of exoplanets. For the following example, to provide a focal problem upon which to compare quadrature methods, we compute the evidence of a model with two such planets on data generated with a two-planet model. The corresponding integral is analytically intractable, with a multi-modal integrand (the likelihood $p(\mathcal{D} \mid \theta, \mathcal{M})$) and a 19-dimensional θ rendering the quadrature problem challenging.

[6] The authors are grateful to H. Parviainen and S. Aigrain for providing data and code examples.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

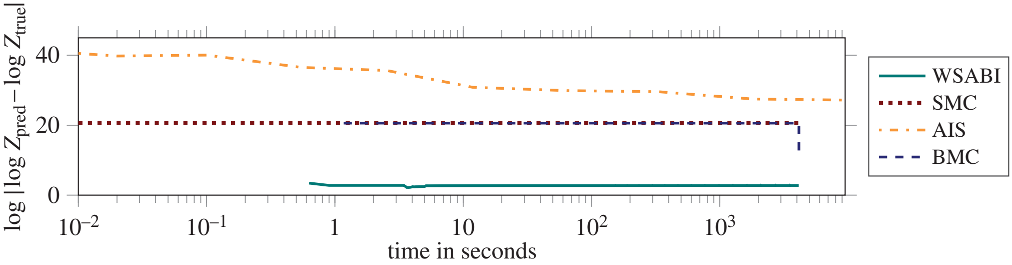

Figure 2. We compare different quadrature methods in computing the 19-dimensional integral giving model evidence in an astrometry application. The Bayesian quadrature algorithm (WSABI) that employs active selection of nodes, along with prior knowledge of the smoothness and non-negativity of the integrand, converges faster than the Monte Carlo approaches: simple Monte Carlo (SMC) and annealed importance sampling (AIS). Note that Bayesian Monte Carlo (BMC) is a Bayesian quadrature algorithm that uses the same samples as SMC, explaining their similar performance. Note also the performance improvement offered by WSABI over BMC, suggesting the crucial role played by the active selection of nodes.

As the probabilistic method, we use a recent algorithm: warped, sequential, active Bayesian integration (WSABI[7]) [32]. WSABI is a Bayesian quadrature algorithm that uses an internal probabilistic model that is well-calibrated to the suspected properties of the problem's integrand. Firstly, like the exponentiated quadratic, WSABI's covariance function is suitable for smooth integrands, as are expected for the problem. Secondly, and beyond what is achievable with classic quadrature rules, WSABI explicitly encodes the fact that the integrand (the likelihood $p(\mathcal{D} \mid \theta, \mathcal{M})$) is strictly positive. WSABI also makes use of a further opportunity afforded by the probabilistic numeric approach: it actively selects nodes so as to minimize the uncertainty in the integral. This final contribution permits nodes to be selected that are far more informative than gridded or randomly selected evaluations. We compare this algorithm against two different Monte Carlo approaches to the problem: annealed importance sampling (AIS) [33] (which was implemented with a Metropolis–Hastings sampler) and simple Monte Carlo (SMC). We additionally compared against Bayesian Monte Carlo (BMC) [24], a Bayesian quadrature algorithm using the simpler exponentiated-quadratic model, and whose nodes were taken from the same samples selected by SMC. 'Ground truth' (Z_{true}) was obtained through exhaustive SMC sampling (10^6 samples). The results in figure 2 show that the probabilistic quadrature method achieves improved precision drastically faster than the Monte Carlo estimates. It is important to point out that the plot's abscissa is 'wall-clock' time, not algorithmic steps. Probabilistic algorithms need not be expensive.

(e) Linear algebra

Computational linear algebra covers various operations on matrices. We will here focus on linear optimization, where $\boldsymbol{b} \in \mathbb{R}^N$ is known and the task is to find $\boldsymbol{x} \in \mathbb{R}^N$ such that $A\boldsymbol{x} = \boldsymbol{b}$ where $A \in \mathbb{R}^{N\times N}$ is symmetric positive definite. We assume access to projections $A\boldsymbol{s}$ for arbitrary $\boldsymbol{s} \in \mathbb{R}^N$. If N is too large for exact inversion of A, a widely known approach is the method of conjugate gradients (CG) [9], which produces a convergent sequence $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_M$ of improving estimates for $\boldsymbol{x}$. Each iteration involves one matrix-vector multiplication and a small number of linear operations, to produce the update $\boldsymbol{s}_i = \boldsymbol{x}_i - \boldsymbol{x}_{i-1}$, and an 'observation' $\boldsymbol{y}_i = A\boldsymbol{s}_i$. The good performance of CG has been analysed extensively (e.g. [34, §5.1]).

We will use the shorthands $S_M = [\boldsymbol{s}_1, \ldots, \boldsymbol{x}_M]$ and $Y_M = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M]$ for the set of projection directions and 'observations' after M iterations of the method. Defining a probabilistic numerical algorithm requires a joint probability measure $p(Y_M, A, \boldsymbol{b}, \boldsymbol{x} \mid S_M)$ over all involved variables, conditioned on the algorithm's active design decisions; and an 'action rule' (design, policy, control law) describing how the algorithm should collect data. Recent work by Hennig [10] describes such

[7] Matlab code for WSABI is available at https://github.com/OxfordML/wsabi.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

a model which exactly reproduces the sequence $\{\boldsymbol{x}_i\}_{i=1,\dots,M\leq N}$ of CG: as prior, choose a Gaussian measure over the (assumed to exist) inverse $H = A^{-1}$ of A

$$p(H) = \mathcal{N}(\boldsymbol{H}; \boldsymbol{H}_0, \Gamma(W \otimes W)\Gamma^\intercal). \tag{2.6}$$

This implies a Gaussian prior over $x = Hb$, and a non-Gaussian prior over A. In (2.6), $\mathcal{N}$ is a Gaussian distribution over the N^2 elements of H stacked into a vector $\boldsymbol{H}$. Γ is the symmetrization operator ($\Gamma \boldsymbol{A} = \frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^\intercal)$), and $\otimes$ is the Kronecker product ($\Gamma(W \otimes W)\Gamma^\intercal$ is also known as the symmetric Kronecker product [35]). As projections are presumed noise-free, the likelihood is the Dirac distribution, the limit of a Gaussian with vanishing width, which can also be seen as performing a strict conditioning of the prior on the observed function values,

$$p(Y_M \mid A, \boldsymbol{b}, \boldsymbol{x}, S_M) = \prod_i \delta(A\boldsymbol{s}_i - \boldsymbol{y}_i) = \prod_i \delta(\boldsymbol{s}_i - H\boldsymbol{y}_i). \tag{2.7}$$

The action rule at iteration i is to move to $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \alpha\hat{H}_i(A\boldsymbol{x}_i - \boldsymbol{b})$, where $\hat{H}_i$ is the posterior mean $p(H \mid Y_M)$. The optimal step α can be computed exactly in a linear computation.

Equation (2.6) requires choices for $\boldsymbol{H}_0$ and W. The prior mean is set to unit, $\boldsymbol{H}_0 = \boldsymbol{I}$. If $W \in \mathbb{R}^{N\times N}$ is positive definite, then $p(H)$ assigns finite measure to every symmetric $H \in \mathbb{R}^{N\times N}$, and the algorithm converges to the true $\boldsymbol{x}$ in at most N steps, assuming exact computations [10]. In this sense it is non-restrictive, but of course some matrices are assigned more density than others. If W is set to a value among the set $\{W \in \mathbb{R}^{N\times N} \mid W \text{ symm. positive definite and } WY_M = \sigma S_M, \sigma \in \mathbb{R}_+\}$ (this includes the true matrix $W = H$, but does not require access to it), then the resulting algorithm *exactly* reproduces the iteration sequence $\{\boldsymbol{x}_i\}_{i=0,\dots,N}$ of the CG method, and can be implemented in the exact same way. However, this derivation also provides something new: a Gaussian posterior distribution $p(H \mid \boldsymbol{y}) = \mathcal{N}(H; H_M, \Gamma(W_M \otimes W_M)\Gamma^\intercal)$ over H. Details can be found in [10]; for the present discussion it suffices to know that both the posterior mean H_M and covariance parameter W_M are of manageable form. In particular, $H_M = \boldsymbol{I} + UEU^\intercal$ with a diagonal matrix $E \in \mathbb{R}^{2M\times 2M}$ and 'skinny' matrices $U \in \mathbb{R}^{N\times 2M}$, which are a function of the steps $\boldsymbol{s}$ and observed projections $\boldsymbol{y}$. So, as in the case of quadrature, there is a family of Gaussian priors of varying width (scaled by σ), such that all members of the family give the same posterior mean estimate. And this posterior mean estimate is identical to a classic numerical method (CG). But each member of the family gives a different posterior covariance—a different uncertainty estimate.

It is an interesting question to which degree the uncertainty parameter W_M can be designed to give a meaningful error estimate. Some answers can be found in [10]. Interestingly, the equivalence class of prior covariances W_0 that match CG in the mean estimate has more degrees of freedom than the number M of observations (matrix-vector multiplications) collected by the algorithm during its typical runtime. Fitting the posterior uncertainty W_M thus requires strong regularization. The method advocated in [10] constructs such a regularized estimator exclusively from scalar numbers already collected during the run of the CG method, thus keeping computational overhead very small.

But, as in quadrature, there are valuable applications for the probabilistic formulation that do not strictly require a well-calibrated *width* of the posterior. Applications that make primary use of the posterior mean may just require *algebraic structure* in the prior, up to an arbitrary scaling constant, to incorporate available, helpful, information. We will not do this here and instead highlight another use of probabilities over point estimates: the propagation of knowledge from one linear problem to another related problem.

The approach described in the following is known in the numerical linear algebra community as the *recycling of Krylov sequences* [36]. However, while the framework of classic numerical analysis required challenging and bespoke derivation of this result, it follows naturally in the probabilistic viewpoint as the extension of a parametric regressor to a filter on a time-varying process. The probabilistic formulation of computation uses the universal and unique language of inference to enable the solution of similar problems across the breadth of numerics using similar techniques. This is in contrast to the compartmentalized state of current numerics, which demands distinct expert knowledge of each individual numeric problem in order to

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

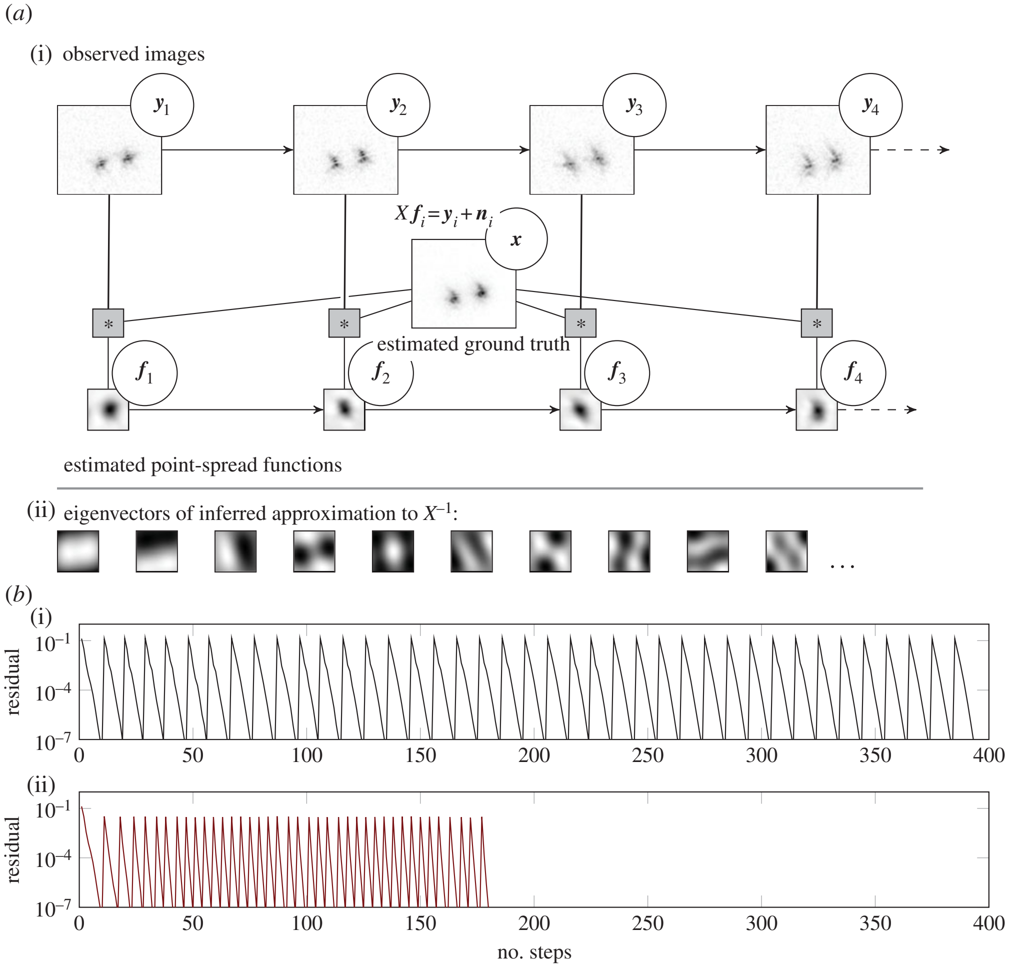

Figure 3. Solving sequences of linear problems by using the probabilistic interpretation of CG, also known as the recycling of Krylov sequences. (*a*(i)) A sequence of observed astronomical images $\boldsymbol{y}_i$ is modelled as the convolution of a stationary true image $\boldsymbol{x}$ with a time-varying point-spread function $\boldsymbol{f}_i$ (the matrix X encodes the convolution with $\boldsymbol{x}$). Each individual deconvolution task requires one run of a linear solver, here chosen to be the method of CG. (*b*) If each problem is solved independently, each instance of the solver progresses similarly (*b*(i) optimization residuals over time; each 'jump' in the residual is the start of a new frame/deconvolution problem). If the posterior mean implied by the probabilistic interpretation of the solver is communicated from one problem to the next, the solvers progress increasingly faster (*b*(ii) note decreasing number of steps for each deconvolution problem, and decrease, by about one order of magnitude, of initial residuals). (*a*(ii)) Over time, the vectors spanning this posterior mean for X^{-1} converge to a generic basis for point-spread functions.

make progress towards it. Furthermore, there is social value in making results accessible to any practitioner with a graduate knowledge of statistics.

In many set-ups, the same A features in a sequence of problems $Ax_t = b_t$, $t = 1, \ldots$. In others, a map A_t changes slightly from step t to step $t+1$. Figure 3 describes a blind deconvolution problem from astronomical imaging:[8] the task is to remove an unknown linear blur from a sequence $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K$ of astronomical images [37]. Atmospherical disturbances create a blur that continuously varies over time. The model is that each frame $\boldsymbol{y}_i$ is the noisy result of convolution of the same ground truth image $\boldsymbol{x}$ with a spatially varying blur kernel $\boldsymbol{f}_i$, i.e. $\boldsymbol{y}_i = \boldsymbol{f}_i\boldsymbol{x} + \boldsymbol{n}_i$, where $\boldsymbol{n}_i$ is white Gaussian noise. A matrix X encodes the convolution operation as $\boldsymbol{y}_i = X\boldsymbol{f}_i + \boldsymbol{n}_i$ (it is possible

[8] The authors thank Harmeling *et al.* [37] for providing data.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

to ensure X is positive definite). A blind deconvolution algorithm iterates between estimating X and estimating the $\boldsymbol{f}_i$ [37]. Each iteration thus requires the solution of K linear problems (with fixed X) to find the $\boldsymbol{f}_i$, then one larger linear problem to find $\boldsymbol{x}$. The naïve approach of running separate instances of CG wastes information, because the K linear problems all share the matrix X, and from one iteration to the next, the matrix X changes less and less as the iterations approach convergence. Instead, information can be *propagated* between related computations using the probabilistic interpretation of CG, by starting each computation in the sequence with a prior mean H_0 set to the posterior mean H_M of the preceding problem. Owing to the low-rank structure of H_M, this has low cost. To prevent a continued rise in computational costs as more and more linear problems are solved, H_M can be restricted to a fixed rank approximation after each inner loop, also at low cost. Figure 3*b* shows the increase in computational efficiency: for the baseline of solving 40 linear problems independently in sequence, each converges about equally fast (figure 3*b*(i); each 'jump' is the start of a new problem). Figure 3*b*(ii) shows optimization progress of the same 40 problems when information is propagated from one problem to the next. The first problem amounts to standard CG, while subsequent iterations can make increasingly better use of the available information. The figure also shows the dominant eigenvectors of the inferred posterior means of X^{-1} after $K = 40$ subsequent linear problems, which converge to a relatively generic basis for point-spread functions. Although not strictly correct, this scheme can be intuitively understood as *inferring a pre-conditioner* across the sequence of problems, by Bayesian filtering (the technical caveat is that the described scheme does not re-scale the linear problem itself, as a pre-conditioner would, it just shifts the initial solution estimate).

(i) Further areas

These examples highlight the areas of quadrature and linear algebra. Analogous results, identifying existing numerical methods with maximum *a posteriori* estimators, have been established in other areas, too. We do not experiment with them here, but they help complete the picture of numerical methods as inference across problem boundaries, as follows.

In nonlinear optimization, quasi-Newton methods such as the Broyden–Fletcher–Goldfarb–Shanno (BFGS) rule are deeply connected to CG (in linear problems, BFGS and CG are the same algorithm [38]). The BFGS algorithm can be interpreted as a specific kind of autoregressive generalization of the Gaussian model for CG [39]. Among other things, this allows explicit modelling of noise on evaluated gradients, a pressing issue in large-scale machine learning [40].

One currently particularly exciting area for probabilistic numerics is ordinary differential equations; more specifically, the solution of initial value problems (IVPs) of the form $(\mathrm{d}\boldsymbol{x}/\mathrm{d}t)(t) = f(\boldsymbol{x}(t), t)$, where $x : \mathbb{R}_+ \to \mathbb{R}^N$ is a real-valued curve parametrized by t known to start at the initial value $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$. Explicit Runge–Kutta methods are a basic and well-studied tool for such problems [20]—while not necessarily state of the art, they nevertheless perform well on many problems, and are conceptually very clear. From the inference perspective, Runge–Kutta methods are linear extrapolation rules. At increasing nodes $t_0 < t_1 < \cdots < t_i < t_s$ they repeatedly construct 'estimates' $\hat{\boldsymbol{x}}(t_i) \approx \boldsymbol{x}(t_i)$ for the true solution which is used to collect an 'observation' $\boldsymbol{y}_i = f(\hat{\boldsymbol{x}}(t_i), t_i)$, such that the estimate is a linear combination of previous observations,

$$\hat{\boldsymbol{x}}(t_i) = \boldsymbol{x}_0 + \sum_{j<i} w_{ij}\boldsymbol{y}_j. \tag{2.8}$$

Crucially, the weights w_{ij} are chosen such that, after s evaluations, the estimate has high convergence order: $\|\boldsymbol{x}(t_s) - \hat{\boldsymbol{x}}(t_s)\| = \mathcal{O}(h^p)$, with an order $p \leq s$ (typically, $p \leq 5$).

It is important to point out the central role that linear extrapolation, and linear computations more generally, play again here, as they did in the other numerical settings discussed above. It is not an oversimplification to note that numerical methods often amount to efficiently projecting an intractable problem into a tractable linear computation. Since the Gaussian family is closed under all linear operations, it is, perhaps, no surprise that Gaussian distributions play a central role in probabilistic re-interpretations of existing numerical methods. In the case of IVPs, Gaussian

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

process extrapolation was previously suggested by Skilling [14] as a tool for their solution. Starting from scratch, Skilling arrived at a method that shares the linear structure of equation (2.8), but does not have the strong theoretical underpinning of Runge–Kutta methods; in particular, Skilling's method does not share the high convergence order of Runge–Kutta methods. But the probabilistic formulation allows for novel theoretical analysis of its own [41], and new kinds of applications, such as the marginalization over posterior uncertainty in subsequent computational steps and finite prior uncertainty over the initial value [42]. A considerably vaguer but much earlier observation was made, on the side of numerical mathematics, by Nordsieck [43], who noted that a class of methods he proposed, and which were subsequently captured in a wider nomenclature of ODE solvers, bore a resemblance to linear electrical filters. These, in turn, are closely connected to Gaussian process regression through the notion of Markov processes.

Recently, Schober *et al.* [44] showed that these connections between Gaussian regression and the solution of IVPs, hitherto only conceptual, can be made tight. There is a family of Gauss–Markov priors that, used as extrapolation rules, give posterior Gaussian processes whose mean function exactly matches members of the Runge–Kutta family. (Thanks to its Markov property, the corresponding inference method can be implemented as a signal filter, and thus in linear computational complexity, like Runge–Kutta methods). Hence, as in the other areas of numerics, there is now a family of methods that returns the trusted point estimates of an established method, while giving a new posterior uncertainty estimate allowing new functionality.

3. Discussion

(a) General recipe for probabilistic numerical algorithms

These recent results, identifying probabilistic formulations for classic numerical methods, highlight a general structure. Consider the problem of approximating the intractable variable z, if the algorithm has the ability to choose 'inputs' $\boldsymbol{x} = \{x_i\}_{i=1,...}$ for computations that result in numbers $\boldsymbol{y}(\boldsymbol{x}) = \{y_i(x_i)\}_{i=1,...}$. A blueprint for the definition of probabilistic numerical methods requires two main ingredients:

(i) A *generative model* $p(z, \boldsymbol{y}(\boldsymbol{x}))$ for all variables involved—that is, a joint probability measure over the intractable quantity to be computed, and the tractable numerical quantities computed in the process of the algorithm. Like all (sufficiently structured) probability measures, this joint measure can be written as

$$p(z, \boldsymbol{y}(\boldsymbol{x})) = p(z)p(\boldsymbol{y}(\boldsymbol{x}) \mid z), \tag{3.1}$$

i.e. separated into a *prior* $p(z)$ and a *likelihood* $p(\boldsymbol{y}(\boldsymbol{x}) \mid z)$. The prior encodes a hypothesis class over solutions, and assigns a typically non-uniform measure over this class. The likelihood explains how the collected tractable numbers $\boldsymbol{y}$ relate to z. It has the basic role of describing the numerical task. Often, in classic numerical problems, the likelihood is a deterministic conditioning rule, a point measure.

(ii) A *design*, *action* rule, or *policy* r, such that

$$x_{i+1} = r(p(z, \boldsymbol{y}(\boldsymbol{x})), \boldsymbol{x}_{1:i}, \boldsymbol{y}_{1:i}), \tag{3.2}$$

encoding how the algorithm should collect numbers. (Here $\boldsymbol{x}_{1:i}$ should be understood as the actions taken in the preceding steps 1 to i, and similarly for $\boldsymbol{y}_{1:i}$). This rule can be simple; for example, it could be independent of collected data (regular grids for integration). Or it might have a Markov-type property that the decision at i only depends on $k < i$ previous decisions (for example in ODE solvers). Sometimes, these rules can be shown to be associated with the minimization of some empirical loss function, and thus be given a decision-theoretic motivation. This is, for example, the case for regular grids in quadrature rules [26].

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

Table 1. Probabilistic description of several basic numerical problems (shortened notation for brevity). In quadrature, (symmetric positive definite) linear optimization, nonlinear optimization and the solution of ordinary differential equation IVPs, classic methods can be cast as maximum *a posteriori* estimation under Gaussian priors. In each case, the likelihood function is a strict conditioning, because observations are assumed to be noise-free. Because numerical methods are active (they decide which computations to perform), they require a decision rule. This is often 'greedy': evaluation under the posterior mean estimate. The exception is integration, which is the only area where the estimated solution of the numerical task is not required to construct the next evaluation.

problem class	integration	linear opt.	nonlinear opt.	ODE IVPs
inferred z	$z = f; \int f(x)\,\mathrm{d}x$	$z = A^{-1}; Ax = b$	$z = B = \nabla\nabla^{\intercal} f$	$z'(t) = f(z(t), t)$
classic method	Gaussian quad.	conjugate gradients	BFGS	Runge–Kutta
$p(z)$	$\mathcal{GP}(f; \mu, k)$	$\mathcal{N}(A^{-1}; M, V)$	$\mathcal{GP}(z; \mu, k)$	$\mathcal{GP}(z; \mu, k)$
$p(\mathbf{y} \mid z)$	$\mathbb{I}(f(x_i) = y_i)$	$\mathbb{I}(y_i = Ax_i)$	$\mathbb{I}(y_i = Bx_i)$	$\mathbb{I}(y_i = z'(t))$
decision rule	minimize post. variance	gradient at est. solution	gradient under est. Hessian	evaluate at est. solution

The aforementioned results show that classic, base-case algorithms for several fundamental numerical problems can be cast as maximum *a posteriori* inference in specific cases of this description; typically under Gaussian priors, and often under simple action rules, such as uniform gridding (in quadrature) or greedy extrapolation (in linear algebra, optimization and the solution of ODEs). Table 1 gives a short summary.

(b) Current limitations

Numerical methods have undergone centuries of development and analysis. The result is a mature set of algorithms that have been ingrained in the scientific tool-set. By contrast, the probabilistic viewpoint suggested here is an emerging area. Many questions remain unanswered, and many aspects of practical importance are missing: formal analysis is at an early stage. Efficient and stable implementations are still in development. Convincing use-cases from various scientific disciplines are only beginning to emerge. We hope that the reader will take these issues as a motivation to contribute, rather than a hold-up. As the use of large-scale computation, simulation and the use of data permeate the quantitative sciences, there is clearly a need for a formal theory of uncertainty in computation.

(c) New paths for research

In our opinion, the match between probabilistic inference and existing numerical methods lays a firm foundation for the analysis of probabilistic numerical methods. We see two primary, complementary goals, as follows.

Firstly, implicit prior assumptions can now be questioned. This could be done in an 'aggressive' way, in the hope of finding either algorithms with faster convergence on a smaller set of problems satisfying stronger assumptions (as in the quadrature example of §3b). Conversely, a 'conservative' re-definition of prior assumptions might improve robustness at increased computational cost. A particularly important aspect in this regard is the action rule r. Wherever r is a function of previously collected 'data' (known as *adaptive design* in statistics and *active learning* in machine learning), a bias can occur. Where the collected data also influence the result of future actions (as in ODE solvers), a more severe problem, an *exploration–exploitation trade-off*, can arise. Checking for such biases, and potentially correcting them, can increase computational cost. But in some applications that require high robustness this effort can pay off.

Secondly, the modelling assumptions, in particular the likelihood, can be extended to increase the reach of existing methods to new settings. A first point of interest is the explicit

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

modelling of uncertainty or noise on the evaluations themselves. This generalization, which would be challenging to construct from a classical standpoint, is often straightforward once a probabilistic interpretation has been found. It may be as simple as replacing the point-mass likelihood functions in table 1 with Gaussian distributions. A prominent case of this aspect is the optimization of noisy functions, such as arises, for example, in the training of large-scale machine learning architectures from subsets of a large or infinite dataset.

Other ideas, such as the propagation of knowledge between problems, as in figure 3, are just as difficult to motivate and study in a classic formulation, but suggest themselves quite naturally in the probabilistic formulation.

There are also practical considerations that shape the research effort. Gaussian distributions play an important role, at least where the inferred quantity is continuous valued. This is not incidental: to a large degree, the point of a numerical method is to turn an intractable computation into a sequence of linear computations. The Gaussian exponential family is closed under linear projections, thus ideally suited for this task.

Efficient adaptation of model hyper-parameters is crucial for a well-calibrated posterior measure. Models with fixed parameters often simply reproduce existing analytic bounds; only through parameter adaptation can uncertainty be actively 'fitted'. Doing so is perhaps more challenging than elsewhere in statistics because numerical methods are 'inner-loop' algorithms used to solve more complex, higher level computations. It is important to find computationally lightweight parameter estimation methods, perhaps at the cost of accepting some limitations in model flexibility.

Although the fundamental insight that numerical methods solve inference problems is not new, the study of probabilistic numerical methods is still young. Recent work has made progress, exposing a wealth of enticing applications in the process. We conclude this text by highlighting a most promising, if distant, application motivating ongoing research.

(d) A vision: chained numerical methods communicating uncertainty

Fuelled by ubiquitous collection and communication of data, several academic and industrial fields are now interested in systems that use observations to adapt to, and interact with, their data source in an autonomous way. Figure 4 shows a conceptual sketch of an autonomous machine aiming to solve a given task by using observations (data) D to build a probabilistic model $p(x_t \mid D, \theta_t)$ of variables x_t that describe the environment's dynamics through model parameters θ_t. The model can be used to predict future states $x_{t+\delta t}$ as a function of actions a_t chosen by the machine. The goal is to choose actions that, over time, maximize some measure of utility that encodes the task. This requires a sequence of numerical steps: inference on x requires marginalization and expectations, i.e. integration. Fitting θ involves optimization. Prediction of $x_{t+\delta t}$ may entail solving differential equations. All three areas have linear base cases (inference in linear regression, optimization of quadratic functions, the solution of linear ODEs). The combination of a sequence of 'black-box' numerical methods in such automated set-ups gives rise to new challenges. Each method receives a point estimate from its precursor, performs its local computation (and adds its local error), and hands the result on. Errors can accumulate in unexpected ways along this chain, but modelling their accumulation provides value: it may be unnecessary to run a numerical method to convergence if its inputs are already known to be only rough estimates.

Specifically, numerical methods allowing for probabilistic inputs and outputs turn the sketch of figure 4 into a factor graph [45], and allow propagation of uncertainty estimates along the chain of computation, through message passing [46]. This would identify sources of computational error, allowing: the active management of a computational budget across the chain; the dedication of finite computer resources to steps that dominate the overall error; and the truncation of computations early once they reach sufficient precision. Uncertainty propagation through computations has been studied widely before ([47] gives a review), but the available algorithms focus on the effects of set-up uncertainties on the outcome of a computation, rather than the

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

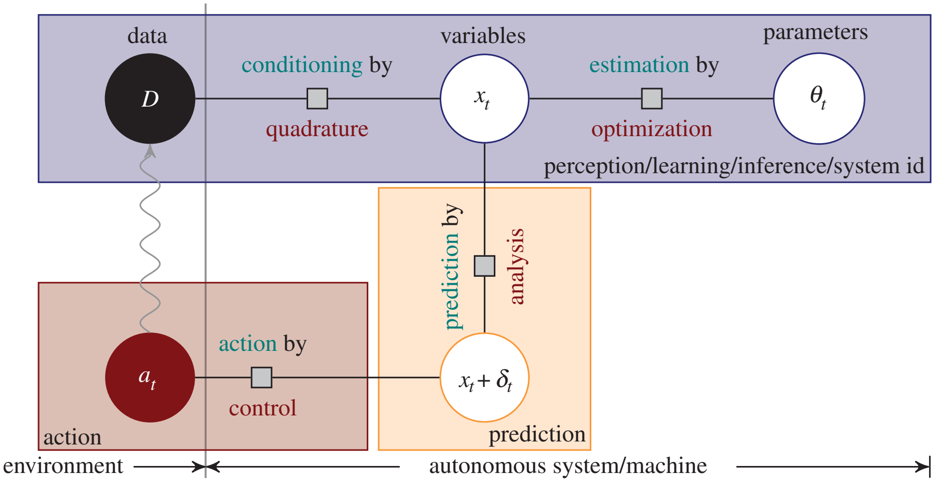

Figure 4. Sketch of an autonomous system, collecting data to build a parametrized model of the environment. Predictions of future states from the model can be used to choose an action strategy. If intermediate operations are solved by numerical methods, both computational errors and inherent uncertainty should be propagated across the pipeline to monitor and target computational effort.

computation itself. This new functionality explicitly requires calibrated probabilistic uncertainty at each step of the computation, *at runtime*. Classic abstract convergence analyses cannot be used for this kind of estimation.

4. Conclusion

Numerical tasks can be interpreted as inference problems, giving rise to probabilistic numerical methods. Established algorithms for many tasks can be cast explicitly in this light. Doing so establishes connections between seemingly disparate problems, yields new functionality and can improve performance on structured problems. To allow interpretation of the posterior as a statement of uncertainty, care must be taken to ensure well-calibrated priors and models. But even where the uncertainty interpretation is not (yet) rigorously established, the probabilistic formulation already allows for the encoding of prior information about problem structure, including the propagation of collected information among problem instances, leading to improved performance. Many open questions remain for this exciting field. In the long run, probabilistic formulations may allow the propagation of uncertainty through pipelines of computation, and thus the active control of computational effort through hierarchical, modular computations.

Authors' contributions. All three authors contributed content and editing equally.

Competing interests. The authors declare they have no conflicting or competing interests.

Funding. P.H. is funded by the Emmy Noether Programme of the German Research Community (DFG). M.G. is funded by an Engineering and Physical Sciences Research Council (EPSRC), EP/J016934/2, Established Career Research Fellowship, a Royal Society Wolfson Research Merit Award and EPSRC Programme Grant—Enabling Quantification of Uncertainty for Large Scale Inverse Problems—EP/K034154/1.

Acknowledgements. The authors express their gratitude to the two anonymous referees for several helpful comments. They are also grateful to Catherine Powell, University of Manchester, for pointing out the connection to recycled Krylov sequence methods.

References

1. Jeffreys H. 1961 *Theory of probability*. Oxford, UK: Oxford University Press.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

2. Hutter M. 2010 *Universal artificial intelligence*. Berlin, Germany: Springer.
3. Erdös P, Kac M. 1940 The Gaussian law of errors in the theory of additive number theoretic functions. *Am. J. Math.* **62**, 738–742.
4. Knuth D, Pardo L. 1976 Analysis of a simple factorization algorithm. *Theor. Comput. Sci.* **3**, 321–348. (doi:10.1016/0304-3975(76)90050-5)
5. Hoerl A, Kennard R. 2000 Ridge regression: biased estimation for nonorthogonal problems. *Technometrics* **42**, 80–86. (doi:10.1080/00401706.2000.10485983)
6. Krige D. 1951 A statistical approach to some mine valuation and allied problems on the Witwatersrand. PhD thesis, University of the Witwatersrand, South Africa.
7. Tikhonov AN. 1943 On the stability of inverse problems. *Dokl. Akad. Nauk SSSR* **39**, 195–198.
8. Gauss C. 1809 *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*. Hamburg, Germany: F. Perthes & I.H. Besser.
9. Hestenes M, Stiefel E. 1952 Methods of conjugate gradients for solving linear systems. *J. Res. Natl Bur. Stand.* **49**, 409–436. (doi:10.6028/jres.049.044)
10. Hennig P. 2015 Probabilistic interpretation of linear solvers. *SIAM J. Optim.* **25**, 234–260. (doi:10.1137/140955501)
11. Shental O, Siegel P, Wolf J, Bickson D, Dolev D. 2008 Gaussian belief propagation solver for systems of linear equations. In *IEEE Int. Symp. on Information Theory, Toronto, Ontario, 6–11 July 2008*, pp. 1863–1867. Piscataway, NJ: IEEE.
12. Diaconis P. 1988 Bayesian numerical analysis. *Stat. Decis. Theory Relat. Top.* **IV**, 163–175.
13. O'Hagan A. 1992 Some Bayesian numerical analysis. *Bayesian Stat.* **4**, 345–363.
14. Skilling J. 1991 Bayesian solution of ordinary differential equations. In *Maximum entropy and Bayesian methods* (eds CR Smith, GJ Erickson, PO Neudorfer), pp. 23–37. Fundamental Theories of Physics, vol. 50. Dordrecht, The Netherlands: Springer.
15. Kennedy MC, O'Hagan A. 2001 Bayesian calibration of computer models. *J. R. Stat. Soc. B* **63**, 425–464. (doi:10.1111/1467-9868.00294)
16. Kiureghian AD, Ditlevsen O. 2009 Aleatory or epistemic? Does it matter?. *Struct. Saf.* **31**, 105–112. (doi:10.1016/j.strusafe.2008.06.020)
17. Iman RL, Helton JC. 1988 An investigation of uncertainty and sensitivity analysis techniques for computer models. *Risk Anal.* **8**, 71–90. (doi:10.1111/j.1539-6924.1988.tb01155.x)
18. Liberty E, Woolfe F, Martinsson P-G, Rokhlin V, Tygert M. 2007 Randomized algorithms for the low-rank approximation of matrices. *Proc. Natl Acad. Sci. USA* **104**, 20 167–20 172. (doi:10.1073/pnas.0709640104)
19. Halko N, Martinsson P-G, Tropp J. 2011 Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.* **53**, 217–288. (doi:10.1137/090771806)
20. Nörsett E, Wanner S, Hairer G. 1987 *Solving ordinary differential equations I—nonstiff problems*. Berlin, Germany: Springer.
21. Davis P, Rabinowitz P. 2007 *Methods of numerical integration*. Mineola, NY: Courier Dover.
22. Wahba G. 1990 *Spline models for observational data*. CBMS-NSF Regional Conferences Series in Applied Mathematics, no. 59. Philadelphia, PA: Society for Industrial and Applied Mathematics.
23. O'Hagan A. 1991 Bayes–Hermite quadrature. *J. Stat. Plan. Inference* **29**, 245–260. (doi:10.1016/0378-3758(91)90002-V)
24. Rasmussen CE, Ghahramani Z. 2003 Bayesian Monte Carlo. In *Advances in neural information processing systems*, vol. 15 (eds S Becker, S Thrun, K Obermayer), pp. 505–512. Cambridge, MA: MIT Press.
25. Rasmussen C, Williams C. 2006 *Gaussian processes for machine learning*. Cambridge, MA: MIT Press.
26. Minka T. 2000 Deriving quadrature rules from Gaussian processes. Technical Report. Carnegie Mellon University, Pittsburgh, PA, USA.
27. Adler R. 1981 *The geometry of random fields*. New York, NY: Wiley.
28. Osborne M, Duvenaud D, Garnett R, Rasmussen C, Roberts S, Ghahramani Z. 2012 Active learning of model evidence using Bayesian quadrature. In *Advances in neural information processing systems*, vol. 25 (eds F Pereira, CJC Burges, L Bottou, KQ Weinberger), pp. 46–54. Red Hook, NY: Curran Associates, Inc.
29. van der Vaart A, van Zanten J. 2011 Information rates of nonparametric Gaussian process methods. *J. Mach. Learn. Res.* **12**, 2095–2119.

Downloaded from http://rspa.royalsocietypublishing.org/ on July 22, 2015

30. Griewank A. 2000 *Evaluating derivatives: principles and techniques of algorithmic differentiation*. Frontiers in Applied Mathematics, no. 19. Philadelphia, PA: Society for Industrial and Applied Mathematics.
31. Sozzetti A, Giacobbe P, Lattanzi M, Micela G, Morbidelli R, Tinetti G. 2014 Astrometric detection of giant planets around nearby M dwarfs: the *Gaia* potential. *Mon. Not. R. Astron. Soc.* **437**, 497–509. (doi:10.1093/mnras/stt1899)
32. Gunter T, Osborne MA, Garnett R, Hennig P, Roberts S. 2014 Sampling for inference in probabilistic models with fast bayesian quadrature. In *Advances in neural information processing systems*, vol. 27 (eds Z Ghahramani, M Welling, C Cortes, ND Lawrence, KQ Weinberger), pp. 2789–2797. Red Hook, NY: Curran Associates, Inc.
33. Neal R. 2001 Annealed importance sampling. *Stat. Comput.* **11**, 125–139. (doi:10.1023/A:1008923215028)
34. Wright J, Nocedal S. 1999 *Numerical optimization*. Berlin, Germany: Springer.
35. van Loan C. 2000 The ubiquitous Kronecker product. *J. Comput. Appl. Math.* **123**, 85–100. (doi:10.1016/S0377-0427(00)00393-9)
36. Parks M, de Sturler E, Mackey G, Johnson D, Maiti S. 2006 Recycling Krylov subspaces for sequences of linear systems. *SIAM J. Sci. Comput.* **28**, 1651–1674. (doi:10.1137/040607277)
37. Harmeling S, Hirsch M, Sra S, Schölkopf B. 2009 Online blind deconvolution for astronomical imaging. In *2009 IEEE Int. Conf. on Computational Photography* (*ICCP*)*, San Francisco, CA, 16–17 April 2009*, pp. 1–7. Piscataway, NJ: IEEE.
38. Nazareth L. 1979 A relationship between the BFGS and conjugate gradient algorithms and its implications for new algorithms. *SIAM J. Numer. Anal.* **16**, 794–800. (doi:10.1137/0716059)
39. Hennig P, Kiefel M. 2013 Quasi-Newton methods—a new direction. *J. Mach. Learn. Res.* **14**, 834–865.
40. Hennig P. 2013 Fast probabilistic optimization from noisy gradients. In *Proc. of the 30th Int. Conf. on Machine Learning, Atlanta, GA, 16–21 June 2013*. See http://jmlr.org/proceedings/papers/v28/hennig13.pdf.
41. Chkrebtii O, Campbell D, Girolami M, Calderhead B. 2013 Bayesian uncertainty quantification for differential equations. (http://arxiv.org/abs/1306.2365)
42. Hennig P, Hauberg S. 2014 Probabilistic solutions to differential equations and their application to Riemannian statistics. Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. *Journal of Machine Learning Research: Workshops and Conference Proceedings* **33**.
43. Nordsieck A. 1962 On numerical integration of ordinary differential equations. *Math. Comput.* **16**, 22–49. (doi:10.1090/S0025-5718-1962-0136519-5)
44. Schober M, Duvenaud D, Hennig P 2014 Probabilistic ODE solvers with Runge-Kutta means. In *Advances in neural information processing systems*, vol. 27 (eds Z Ghahramani, M Welling, C Cortes, ND Lawrence, KQ Weinberger), pp. 739–747. Red Hook, NY: Curran Associates, Inc.
45. Frey B, Kschischang F, Loeliger H-A, Wiberg N. 1997 Factor graphs and algorithms. In *Proc. 35th Allerton Conf. on Communications, Control, and Computing*, pp. 666–680. Champaign-Urbana, IL: University of Illinois.
46. Lauritzen S, Spiegelhalter D. 1988 Local computations with probabilities on graphical structures and their application to expert systems. *J. R. Stat. Soc.* **50**, 157–224.
47. Lee S, Chen W. 2009 A comparative study of uncertainty propagation methods for black-box-type problems. *Struct. Multidiscip. Optim.* **37**, 239–253. (doi:10.1007/s00158-008-0234-7)