Wiley StatsRef:
Statistics Reference Online

Computational Statistics and Data Science in the Twenty-first Century

By *Andrew J. Holbrook*[1], *Akihiko Nishimura*[2], *Xiang Ji*[3], *and Marc A. Suchard*[1]

Keywords: *computational statistics, data science, big N, big P, big M, sparse regression, Bayesian phylogenetics, statistical software, parallel computing, quantum computing*

Abstract: Data science has arrived, and computational statistics is its engine. As the scale and complexity of scientific and industrial data grow, the discipline of computational statistics assumes an increasingly central role among the statistical sciences. An explosion in the range of real-world applications means the development of more and more specialized computational methods, but five Core Challenges remain. We provide a high-level introduction to computational statistics by focusing on its central challenges, present recent model-specific advances, and preach the ever-increasing role of nonsequential computational paradigms such as multicore, many-core, and quantum computing. Data science is bringing major changes to computational statistics, and these changes will shape the trajectory of the discipline in the twenty-first century.

1 Introduction

We are in the midst of the data science revolution. In October 2012, the Harvard Business Review famously declared data scientist the sexiest job of the twenty-first century[1]. By September 2019, Google searches for the term "data science" had multiplied over sevenfold[2], one multiplicative increase for each intervening year. In the United States between the years 2000 and 2018, the number of bachelor's degrees awarded in either statistics or biostatistics increased over 10-fold (382–3964), and the number of doctoral degrees almost tripled (249–688)[3]. In 2020, seemingly every major university has established or is establishing its own data science institute, center, or initiative.

Data science[4, 5] combines multiple preexisting disciplines (e.g., statistics, machine learning, and computer science) with a redirected focus on creating, understanding, and systematizing workflows that turn real-world data into actionable conclusions. The ubiquity of data in all economic sectors and scientific disciplines makes data science eminently relevant to cohorts of researchers for whom the discipline of

[1] University of California, Los Angeles, CA, USA
[2] Johns Hopkins University, Baltimore, MD, USA
[3] Tulane University, New Orleans, LA, USA

DOI: 10.1002/9781118445112.stat08324

statistics was previously closed off and esoteric. Data science's emphasis on practical application only enhances the importance of *computational statistics*, the interface between statistics and computer science primarily concerned with the development of algorithms producing either statistical inference[1] or predictions. Since both of these products comprise essential tasks in any data scientific workflow, we believe that the pan-disciplinary nature of data science only increases the number of opportunities for computational statistics to evolve by taking on new applications[2] and serving the needs of new groups of researchers.

This is the natural role for a discipline that has increased the breadth of statistical application from the beginning. First put forward by R.A. Fisher in 1936[6, 7], the permutation test allows the scientist (who owns a computer) to test hypotheses about a broader swath of functionals of a target population while making fewer statistical assumptions[8]. With a computer, the scientist uses the bootstrap[9, 10] to obtain confidence intervals for population functionals and parameters of models too complex for analytic methods. Newton–Raphson optimization and the Fisher scoring algorithm facilitate linear regression for binary, count, and categorical outcomes[11, 12]. More recently, Markov chain Monte Carlo (MCMC)[13, 14] has made Bayesian inference practical for massive, hierarchical, and highly structured models that are useful for the analysis of a significantly wider range of scientific phenomena.

While computational statistics increases the diversity of statistical applications historically, certain central difficulties exist and will continue to remain for the rest of the twenty-first century. In Section 2, we present the first class of Core Challenges, or challenges that are easily quantifiable for generic tasks. Core Challenge 1 is Big N, or statistical inference when the number "N" of observations or data points is large; Core Challenge 2 is Big P, or statistical inference when the model parameter count "P" is large; and Core Challenge 3 is Big M, or statistical inference when the model's objective or density function is multimodal (having many modes "M")[3]. When large, each of these quantities brings its own unique computational difficulty. Since well over 2.5 exabytes (or 2.5×10^{18} bytes) of data come into existence each day[15], we are confident that Core Challenge 1 will survive well into the twenty-second century.

But Core Challenges 2 and 3 will also endure: data complexity often increases with size, and researchers strive to understand increasingly complex phenomena. Because many examples of big data become "big" by combining heterogeneous sources, big data often necessitate big models. With the help of two recent examples, Section 3 illustrates how computational statisticians make headway at the intersection of big data and big models with model-specific advances. In Section 3.1, we present recent work in Bayesian inference for big N and big P regression. Beyond the simplified regression setting, data often come with structures (e.g., spatial, temporal, and network), and correct inference must take these structures into account. For this reason, we present novel computational methods for a highly structured and hierarchical model for the analysis of multistructured and epidemiological data in Section 3.2.

The growth of model complexity leads to new inferential challenges. While we define Core Challenges 1–3 in terms of generic target distributions or objective functions, Core Challenge 4 arises from inherent difficulties in treating complex models generically. Core Challenge 4 (Section 4.1) describes the difficulties and trade-offs that must be overcome to create fast, flexible, and friendly "algo-ware". This Core Challenge requires the development of statistical algorithms that maintain efficiency despite model structure and, thus, apply to a wider swath of target distributions or objective functions "out of the box". Such generic algorithms typically require little cleverness or creativity to implement, limiting the amount of time data scientists must spend worrying about computational details. Moreover, they aid the development of flexible statistical software that adapts to complex model structure in a way that users easily understand. But it is not enough that software be flexible and easy to use: mapping computations to computer hardware for optimal implementations remains difficult. In Section 4.2, we argue that Core Challenge 5, effective use of computational resources such as central processing units (CPU), graphics processing units (GPU), and quantum computers, will become increasingly central to the work of the computational statistician as data grow in magnitude.

Wiley StatsRef: Statistics Reference Online, © 2014–2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

2 Core Challenges 1 – 3

Before providing two recent examples of twenty-first century computational statistics (Section 3), we present three easily quantified Core Challenges within computational statistics that we believe will always exist: big N, or inference from many observations; big P, or inference with high-dimensional models; and big M, or inference with nonconvex objective – or multimodal density – functions. In twenty-first century computational statistics, these challenges often co-occur, but we consider them separately in this section.

2.1 Big *N*

Having a large number of observations makes different computational methods difficult in different ways. A worst case scenario, the *exact* permutation test requires the production of $N!$ datasets. Cheaper alternatives, resampling methods such as the Monte Carlo permutation test or the bootstrap, may require anywhere from thousands to hundreds of thousands of randomly produced datasets[8, 10]. When, say, population means are of interest, each Monte Carlo iteration requires summations involving N expensive memory accesses. Another example of a computationally intensive model is Gaussian process regression[16, 17]; it is a popular nonparametric approach, but the exact method for fitting the model and predicting future values requires matrix inversions that scale $\mathcal{O}(N^3)$. As the rest of the calculations require relatively negligible computational effort, we say that matrix inversions represent the *computational bottleneck* for Gaussian process regression.

To speed up a computationally intensive method, one only needs to speed up the method's computational bottleneck. We are interested in performing Bayesian inference[18] based on a large vector of observations $\mathbf{x} = (x_1, \ldots, x_N)$. We specify our model for the data with a likelihood function $\pi(\mathbf{x}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \pi(x_n|\boldsymbol{\theta})$ and use a prior distribution with density function $\pi(\boldsymbol{\theta})$ to characterize our belief about the value of the P-dimensional parameter vector $\boldsymbol{\theta}$ *a priori*. The target of Bayesian inference is the posterior distribution of $\boldsymbol{\theta}$ conditioned on $\mathbf{x}$

$$\pi(\boldsymbol{\theta}|\mathbf{x}) = \pi(\mathbf{x}|\boldsymbol{\theta})\pi(\boldsymbol{\theta}) \Big/ \int \pi(\mathbf{x}|\boldsymbol{\theta})\pi(\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta} \tag{1}$$

The denominator's multidimensional integral quickly becomes impractical as P grows large, so we choose to use the MetropolisHastings (M – H) algorithm to generate a Markov chain with stationary distribution $\pi(\boldsymbol{\theta}|\mathbf{x})$[13, 19, 20]. We begin at an arbitrary position $\boldsymbol{\theta}^{(0)}$ and, for each iteration $s = 0, \ldots, S$, randomly generate the proposal state $\boldsymbol{\theta}^*$ from the transition distribution with density $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(s)})$. We then accept proposal state $\boldsymbol{\theta}^*$ with probability

$$a = \min\left(1, \frac{\pi(\boldsymbol{\theta}^*|\mathbf{x})q(\boldsymbol{\theta}^{(s)}|\boldsymbol{\theta}^*)}{\pi(\boldsymbol{\theta}^{(s)}|\mathbf{x})q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(s)})}\right) \tag{2}$$

The ratio on the right no longer depends on the denominator in Equation (1), but one must still compute the likelihood and its N terms $\pi(x_n|\boldsymbol{\theta}^*)$.

It is for this reason that likelihood evaluations are often the computational bottleneck for Bayesian inference. In the best case, these evaluations are $\mathcal{O}(N)$, but there are many situations in which they scale $\mathcal{O}(N^2)$[21, 22] or worse. Indeed, when P is large, it is often advantageous to use more advanced MCMC algorithms that use the gradient of the log-posterior to generate better proposals. In this situation, the log-likelihood gradient may also become a computational bottleneck[21].

Wiley StatsRef: Statistics Reference Online, © 2014 – 2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

2.2 Big *P*

One of the simplest models for big P problems is ridge regression[23], but computing can become expensive even in this classical setting. Ridge regression estimates the coefficient $\boldsymbol{\theta}$ by minimizing the distance between the observed and predicted values $\mathbf{y}$ and $\mathbf{X}\boldsymbol{\theta}$ along with a weighted square norm of $\boldsymbol{\theta}$:

$$\hat{\boldsymbol{\theta}} = \text{argmin}\{\|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|^2 + \|\boldsymbol{\Phi}^{1/2}\boldsymbol{\theta}\|^2\} = (\mathbf{X}^\intercal\mathbf{X} + \boldsymbol{\Phi})^{-1}\mathbf{X}^\intercal\mathbf{y}$$

For illustrative purposes, we consider the following direct method for computing $\hat{\boldsymbol{\theta}}$.[4] We can first multiply the $N \times P$ design matrix $\mathbf{X}$ by its transpose at the cost of $\mathcal{O}(N^2P)$ and subsequently invert the $P \times P$ matrix $(\mathbf{X}^\intercal\mathbf{X} + \boldsymbol{\Phi})$ at the cost of $\mathcal{O}(P^3)$. The total $\mathcal{O}(N^2P + P^3)$ complexity shows that (i) a large number of parameters is often sufficient for making even the simplest of tasks infeasible and (ii) a moderate number of parameters can render a task impractical when there are a large number of observations. These two insights extend to more complicated models: the same complexity analysis holds for the fitting of generalized linear models (GLMs) as described in McCullagh and Nelder[12].

In the context of Bayesian inference, the length P of the vector $\boldsymbol{\theta}$ dictates the dimension of the MCMC state space. For the M-H algorithm (Section 2.1) with P-dimensional Gaussian target and proposal, Gelman *et al*.[25] show that the proposal distribution's covariance should be scaled by a factor inversely proportional to P. Hence, as the dimension of the state space grows, it behooves one to propose states $\boldsymbol{\theta}^*$ that are closer to the current state of the Markov chain, and one must greatly increase the number S of MCMC iterations. At the same time, an increasing P often slows down rate-limiting likelihood calculations (Section 2.1). Taken together, one must generate many more, much slower MCMC iterations. The wide applicability of latent variable models[26] (Sections 3.1 and 3.2) for which each observation has its own parameter set (e.g., $P \propto N$) means M-H simply does not work for a huge class of models popular with practitioners.

For these reasons, Hamiltonian Monte Carlo (HMC)[27] has become a popular algorithm for fitting Bayesian models with large numbers of parameters. Like M-H, HMC uses an accept step (Equation 2). Unlike M-H, HMC takes advantage of additional information about the target distribution in the form of the log-posterior gradient. HMC works by doubling the state space dimension with an auxiliary Gaussian "momentum" variable $\mathbf{p} \sim \text{Normal}_P(\mathbf{0}, \mathbf{M})$ independent to the "position" variable $\boldsymbol{\theta}$. The constructed Hamiltonian system has energy function given by the negative logarithm of the joint distribution

$$H(\boldsymbol{\theta}, \mathbf{p}) \propto -\log(\pi(\boldsymbol{\theta}|\mathbf{X}) \times \exp(-\mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}/2)) \propto -\log\pi(\boldsymbol{\theta}|\mathbf{X}) + \mathbf{p}^T\mathbf{M}^{-1}\mathbf{p}/2$$

and we produce proposals by simulating the system according to Hamilton's equations

$$\dot{\boldsymbol{\theta}} = \frac{\partial}{\partial \mathbf{p}} H(\boldsymbol{\theta}, \mathbf{p}) = M^{-1}\mathbf{p}/2$$

$$\dot{\mathbf{p}} = -\frac{\partial}{\partial \boldsymbol{\theta}} H(\boldsymbol{\theta}, \mathbf{p}) = \nabla \log\pi(\boldsymbol{\theta}|\mathbf{X})$$

Thus, the momentum of the system moves in the direction of the steepest ascent for the log-posterior, forming an analogy with first-order optimization. The cost is repeated gradient evaluations that may comprise a new computational bottleneck, but the result is effective MCMC for tens of thousands of parameters[21, 28]. The success of HMC has inspired research into other methods leveraging gradient information to generate better MCMC proposals when P is large[29].

2.3 Big *M*

Global optimization, or the problem of finding the minimum of a function with arbitrarily many local minima, is NP-complete in general[30], meaning – in layman's terms – it is impossibly hard. In the absence

Wiley StatsRef: Statistics Reference Online, © 2014–2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

of a tractable theory, by which one might prove one's global optimization procedure works, brute-force grid and random searches and heuristic methods such as particle swarm optimization[31] and genetic algorithms[32] have been popular. Due to the overwhelming difficulty of global optimization, a large portion of the optimization literature has focused on the particularly well-behaved class of *convex* functions[33, 34], which do not admit multiple local minima. Since Fisher introduced his "maximum likelihood" in 1922[35], statisticians have thought in terms of maximization, but convexity theory still applies by a trivial negation of the objective function. Nonetheless, most statisticians safely ignored *concavity* during the twentieth century: exponential family log-likelihoods are log-concave, so Newton–Raphson and Fisher scoring are guaranteed optimality in the context of GLMs[12, 34].

Nearing the end of the twentieth century, multimodality and nonconvexity became more important for statisticians considering high-dimensional regression, that is, regression with many covariates (big P). Here, for purposes of interpretability and variance reduction, one would like to induce *sparsity* on the weights vector $\hat{\boldsymbol{\theta}}$ by performing best subset selection[36, 37]:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \mathbb{R}^P}{\operatorname{argmin}} ||\mathbf{y} - \mathbf{X}\boldsymbol{\theta}||_2^2 \quad \text{subject to} \quad ||\boldsymbol{\theta}||_0 \leq k \tag{3}$$

where $0 < k \leq P$, and $||\cdot||_0$ denotes the ℓ_0-norm, that is, the number of nonzero elements. Because best subset selection requires an immensely difficult nonconvex optimization, Tibshirani[38] famously replaces the ℓ_0-norm with the ℓ_1-norm, thereby providing sparsity, while nonetheless maintaining convexity.

Historically, Bayesians have paid much less attention to convexity than have optimization researchers. This is most likely because the basic theory[13] of MCMC does not require such restrictions: even if a target distribution has one million modes, the well-constructed Markov chain explores them all in the limit. Despite these theoretical guarantees, a small literature has developed to tackle multimodal Bayesian inference[39–42] because multimodal target distributions *do* present a challenge in practice. In analogy with Equation (3), Bayesians seek to induce sparsity by specifiying priors such as the spike-and-slab[43–45], for example,

$$\mathbf{y} \sim \text{Normal}_N\ (\mathbf{X}\boldsymbol{\Gamma}\boldsymbol{\theta}, \sigma^2\mathbf{I}_N) \quad \text{for} \quad [\boldsymbol{\Gamma}]_{pp'} = \begin{cases} \gamma_p \sim \text{Bernoulli}\ (\pi) & p = p' \\ 0 & p \neq p' \end{cases} \quad \text{and} \quad \pi \in (0, 1)$$

As with the best subset selection objective function, the spike-and-slab target distribution becomes heavily multimodal as P grows and the support of $\boldsymbol{\Gamma}$'s discrete distribution grows to 2^P potential configurations.

In the following section, we present an alternative Bayesian sparse regression approach that mitigates the combinatorial problem along with a state-of-the-art computational technique that scales well both in N and P.

3 Model-Specific Advances

These challenges will remain throughout the twenty-first century, but it is possible to make significant advances for specific statistical tasks or classes of models. Section 3.1 considers Bayesian sparse regression based on continuous shrinkage priors, designed to alleviate the heavy multimodality (big M) of the more traditional spike-and-slab approach. This model presents a major computational challenge as N and P grow, but a recent computational advance makes the posterior inference feasible for many modern large-scale applications.

And because of the rise of data science, there are increasing opportunities for computational statistics to grow by enabling and extending statistical inference for scientific applications previously outside of mainstream statistics. Here, the science may dictate the development of structured models with complexity

DOI: 10.1002/9781118445112.stat08324

possibly growing in N and P. Section 3.2 presents a method for fast phylogenetic inference, where the primary structure of interest is a "family tree" describing a biological evolutionary history.

3.1 Bayesian Sparse Regression in the Age of Big *N* and Big *P*

With the goal of identifying a small subset of relevant features among a large number of potential candidates, sparse regression techniques have long featured in a range of statistical and data science applications[46]. Traditionally, such techniques were commonly applied in the "$N \leq P$" setting, and correspondingly computational algorithms focused on this situation[47], especially within the Bayesian literature[48].

Due to a growing number of initiatives for large-scale data collections and new types of scientific inquiries made possible by emerging technologies, however, increasingly common are datasets that are "big N" and "big P" at the same time. For example, modern observational studies using health-care databases routinely involve $N \approx 10^5 \sim 10^6$ patients and $P \approx 10^4 \sim 10^5$ clinical covariates[49]. The UK Biobank provides brain imaging data on $N = 100\ 000$ patients, with $P = 100 \sim 200\ 000$, depending on the scientific question of interests[50]. Single-cell RNA sequencing can generate datasets with N (the number of cells) in millions and P (the number of genes) in tens of thousands, with the trend indicating further growths in data size to come[51].

3.1.1 Continuous shrinkage: alleviating big *M*

Bayesian sparse regression, despite its desirable theoretical properties and flexibility to serve as a building block for richer statistical models, has always been relatively computationally intensive even before the advent of "big N and big P" data[45, 52, 53]. A major source of its computational burden is severe posterior multimodality (big M) induced by the discrete binary nature of spike-and-slab priors (Section 2.3). The class of *global–local* continuous shrinkage priors is a more recent alternative to shrink θ_ps in a more continuous manner, thereby alleviating (if not eliminating) the multimodality issue[54, 55]. This class of prior is represented as a scale mixture of Gaussians:

$$\theta_p \mid \lambda_p, \tau \sim \text{Normal}_N(0, \tau^2\lambda_p^2),\ \ \lambda_p \sim \pi_{\text{local}}(\cdot),\ \ \tau \sim \pi_{\text{global}}(\cdot)$$

The idea is that the *global scale* parameter $\tau \leq 1$ would shrink most θ_ps toward zero, while the *local scale* λ_ps, with its heavy-tailed prior $\pi_{\text{local}}(\cdot)$, allow a small number of $\tau\lambda_p$ and hence θ_ps to be estimated away from zero. While motivated by two different conceptual frameworks, the spike-and-slab can be viewed as a subset of global–local priors in which $\pi_{\text{local}}(\cdot)$ is chosen as a mixture of delta masses placed at $\lambda_p = 0$ and $\lambda_p = \sigma/\tau$. Continuous shrinkage mitigates the multimodality of spike-and-slab by smoothly bridging small and large values of λ_p.

On the other hand, the use of continuous shrinkage priors does not address the increasing computational burden from growing N and P in modern applications. Sparse regression posteriors under global–local priors are amenable to an effective Gibbs sampler, a popular class of MCMC we describe further in Section 4.1. Under the linear and logistic models, the computational bottleneck of this Gibbs sampler stems from the need for repeated updates of $\boldsymbol{\theta}$ from its conditional distribution

$$\boldsymbol{\theta} \mid \tau, \boldsymbol{\lambda}, \boldsymbol{\Omega}, \mathbf{y}, \mathbf{X} \sim \text{Normal}_p(\boldsymbol{\Phi}^{-1}\mathbf{X}^\intercal\boldsymbol{\Omega}\mathbf{y}, \boldsymbol{\Phi}^{-1})) \text{ for } \boldsymbol{\Phi} = \mathbf{X}^\intercal\boldsymbol{\Omega}\mathbf{X} + \tau^{-2}\boldsymbol{\Lambda}^{-2} \tag{4}$$

where $\boldsymbol{\Omega}$ is an additional parameter of diagonal matrix and $\boldsymbol{\Lambda} = \text{diag}(\boldsymbol{\lambda})$.[5] Sampling from this high-dimensional Gaussian distribution requires $\mathcal{O}(NP^2 + P^3)$ operations with the standard approach[58]: $\mathcal{O}(NP^2)$ for computing the term $\mathbf{X}^\intercal\boldsymbol{\Omega}\mathbf{X}$ and $\mathcal{O}(P^3)$ for Cholesky factorization of $\boldsymbol{\Phi}$. While an alternative

Wiley StatsRef: Statistics Reference Online, © 2014–2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

approach by Bhattacharya *et al*.[48] provides the complexity of $\mathcal{O}(N^2P + N^3)$, the computational cost remains problematic in the big N and big P regime at $\mathcal{O}(\min\{N^2P, NP^2\})$ after choosing the faster of the two.

3.1.2 Conjugate gradient sampler for structured high-dimensional Gaussians

The *conjugate gradient* (CG) sampler of Nishimura and Suchard[57] combined with their *prior-preconditioning* technique overcomes this seemingly inevitable $\mathcal{O}(\min\{N^2P, NP^2\})$ growth of the computational cost. Their algorithm is based on a novel application of the CG method[59, 60], which belongs to a family of *iterative methods* in numerical linear algebra. Despite its first appearance in 1952, CG received little attention for the next few decades, only making its way into major software packages such as MATLAB in the 1990s[61]. With its ability to solve a large and structured linear system $\boldsymbol{\Phi\theta} = \boldsymbol{b}$ via a small number of matrix–vector multiplications $\boldsymbol{\nu} \to \boldsymbol{\Phi\nu}$ without ever explicitly inverting $\boldsymbol{\Phi}$, however, CG has since emerged as an essential and prototypical algorithm for modern scientific computing[62, 63].

Despite its earlier rise to prominence in other fields, CG has not found practical applications in Bayesian computation until rather recently[57, 64]. We can offer at least two explanations for this. First, being an algorithm for solving a deterministic linear system, it is not obvious how CG would be relevant to Monte Carlo simulation, such as sampling from $\text{Normal}_P(\boldsymbol{\mu}, \boldsymbol{\Phi}^{-1})$; ostensively, such a task requires computing a "square root" $\boldsymbol{L}$ of the precision matrix so that $Var(\boldsymbol{L}^{-1}\boldsymbol{z}) = \boldsymbol{L}^{-1}\boldsymbol{L}^{-\intercal} = \boldsymbol{\Phi}^{-1}$ for $\boldsymbol{z} \sim \text{Normal}_P(\boldsymbol{0}, \boldsymbol{I}_P)$. Secondly, unlike direct linear algebra methods, iterative methods such as CG have a variable computational cost that depends critically on the user's choice of a preconditioner and thus *cannot* be used as a "black-box" algorithm.[6] In particular, this novel application of CG to Bayesian computation is a reminder that other powerful ideas in other computationally intensive fields may remain untapped by the statistical computing community; knowledge transfers will likely be facilitated by having more researchers working at intersections of different fields.

Nishimura and Suchard[57] turns CG into a viable algorithm for Bayesian sparse regression problems by realizing that (i) we can obtain a Gaussian vector $\boldsymbol{b} \sim \text{Normal}_P(\mathbf{X}^\intercal\boldsymbol{\Omega}\mathbf{y}, \boldsymbol{\Phi})$ by first generating $\boldsymbol{z} \sim \text{Normal}_P(\boldsymbol{0}, \boldsymbol{I}_P)$ and $\boldsymbol{\zeta} \sim \text{Normal}_N(\boldsymbol{0}, \boldsymbol{I}_N)$ and then setting $\boldsymbol{b} = \mathbf{X}^\intercal\boldsymbol{\Omega}\mathbf{y} + \mathbf{X}^\intercal\boldsymbol{\Omega}^{1/2}\boldsymbol{\zeta} + \tau^{-1}\boldsymbol{\Lambda}^{-1}\boldsymbol{z}$ and (ii) subsequently solving $\boldsymbol{\Phi\theta} = \boldsymbol{b}$ yields a sample $\boldsymbol{\theta}$ from the distribution (4). The authors then observe that the mechanism through which a shrinkage prior induces sparsity of θ_ps also induces a tight clustering of eigenvalues in the prior-preconditioned matrix $\tau^2\boldsymbol{\Lambda\Phi\Lambda}$. This fact makes it possible for prior-preconditioned CG to solve the system $\boldsymbol{\Phi\theta} = \boldsymbol{b}$ in K matrix–vector operations of form $\boldsymbol{\nu} \to \boldsymbol{\Phi\nu}$, where K roughly represents the number of significant θ_ps that are distinguishable from zeros under the posterior. For $\boldsymbol{\Phi}$ having a structure as in (4), $\boldsymbol{\Phi\nu}$ can be computed via matrix–vector multiplications of form $\boldsymbol{\nu} \to \mathbf{X}\boldsymbol{\nu}$ and $\boldsymbol{w} \to \mathbf{X}^\intercal\boldsymbol{w}$, so each $\boldsymbol{\nu} \to \boldsymbol{\Phi\nu}$ operation requires a fraction of the computational cost of directly computing $\boldsymbol{\Phi}$ and then factorizing it.

Prior-preconditioned CG demonstrates an order of magnitude speedup in posterior computation when applied to a comparative effectiveness study of atrial fibrillation treatment involving $N = 72\,489$ patients and $P = 22\,175$ covariates[57]. Though unexplored in their work, the algorithm's heavy use of matrix–vector multiplications provides avenues for further acceleration. Technically, the algorithm's complexity may be characterized as $\mathcal{O}(NPK)$, for the K matrix–vector multiplications by $\mathbf{X}$ and $\mathbf{X}^\intercal$, but the theoretical complexity is only a part of the story. Matrix–vector multiplications are amenable to a variety of hardware optimizations, which in practice can make orders of magnitude difference in speed (Section 4.2). In fact, given how arduous manually optimizing computational bottlenecks can be, designing algorithms so as to take advantage of common routines (as those in Level 3 BLAS) and their ready-optimized implementations has been recognized as an effective principle in algorithm design[65].

DOI: 10.1002/9781118445112.stat08324

3.2 Phylogenetic Reconstruction

While big N and big P regression adapts a classical statistical task to contemporary needs, the twenty-first century is witnessing the application of computational statistics to the entirety of applied science. One such example is the tracking and reconstruction of deadly global viral pandemics. Molecular phylogenetics has become an essential analytical tool for understanding the complex patterns in which rapidly evolving pathogens propagate throughout and between countries, owing to the complex travel and transportation patterns evinced by modern economies[66], along with other factors such as increased global population and urbanization[67]. The advance in sequencing technology is generating pathogen genomic data at an ever-increasing pace, with a trend to real time that requires the development of computational statistical methods that are able to process the sequences in a timely manner and produce interpretable results to inform national/global public health organizations.

The previous three Core Challenges are usually interwound such that the increase in the sample size (big N) and the number of traits (big P) for each sample usually happen simultaneously and lead to increased heterogeneity that requires more complex models (big M). For example, recent studies in viral evolution have seen a continuing increase in the sample size that the West Nile virus, Dengue, HIV, and Ebola virus studies involve 104, 352, 465, and 1610 sequences[68–71], and the GISAID database has collected 92 000 COVID-19 genomic sequences by the end of August 2020[72].

To accommodate the increasing size and heterogeneity in the data and be able to apply the aforementioned efficient gradient-based algorithms, Ji *et al.*[73] propose a linear-time algorithm for calculating an $O(N)$-dimensional gradient on a tree w.r.t. the sequence evolution. The linear-time gradient algorithm calculates each branch-specific derivative through a preorder traversal that complements the postorder traversal from the likelihood calculation of the observed sequence data at the tip of the phylogeny by marginalizing over all possible hidden states on the internal nodes. The pre- and postorder traversals complete the Baum's forward–backward algorithm in a phylogenetic framework[74]. The authors then apply the gradient algorithm with HMC (Section 2.2) samplers to learn the branch-specific viral evolutionary rates.

Thanks to these advanced computational methods, one can employ more flexible models that lend themselves to more realistic reconstructions and uncertainty quantification. Following a random-effects relaxed clock model, they model the evolutionary rate r_p of branch p on a phylogeny as the product of a global treewise mean parameter μ and a branch-specific random effect ϵ_p. They model the random-effect ϵ_ps as independent and identically distributed from a lognormal distribution such that ϵ_p has mean 1 and variance ψ^2 under a hierarchical model where ψ is the scale parameter. To accommodate the difference in scales of the variability in the parameter space for the HMC sampler, the authors adopt preconditioning with adaptive mass matrix informed by the diagonal entries of the Hessian matrix. More precisely, the nonzero diagonal elements of the mass matrix truncate the values from the first s HMC iterations of $H_{pp}^{(s)} = \frac{1}{\lfloor s/k \rfloor} \sum_{s:s/k \in \mathbb{Z}^+} \left[-\frac{\partial^2}{\partial^2 \theta_p} \log \pi(\boldsymbol{\theta}) \Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(s)}} \right] \approx \mathbb{E}_{\pi(\theta)} \left[-\frac{\partial^2}{\partial^2 \theta_i} \log \pi(\boldsymbol{\theta}) \right]$ so that the matrix remains positive-definite and numerically stable. They estimate the treewise (fixed-effect) mean rate μ with posterior mean 4.75 (95% Bayesian credible interval: $4.05, 5.33) \times 10^{-4}$ substitutions per site per year with rate variability characterized by scale parameter with posterior mean $\psi = 1.26[1.06, 1.45]$ for serotype 3 of Dengue virus with a sample size of 352[69]. Figure 1 illustrates the estimated maximum clade credible evolutionary tree of the Dengue virus dataset.

The authors report relative speedup in terms of the effective sample size per second (ESS/s) of the HMC samplers compared to a univariate transition kernel. The "vanilla" HMC sampler with an identity mass matrix gains 2.2× speedup for the minimum ESS/s and 2.5× speedup for the median ESS/s, whereas the "preconditioned" HMC sampler gains 16.4× and 7.4× speedups, respectively. Critically, the authors make

Wiley StatsRef: Statistics Reference Online, © 2014–2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

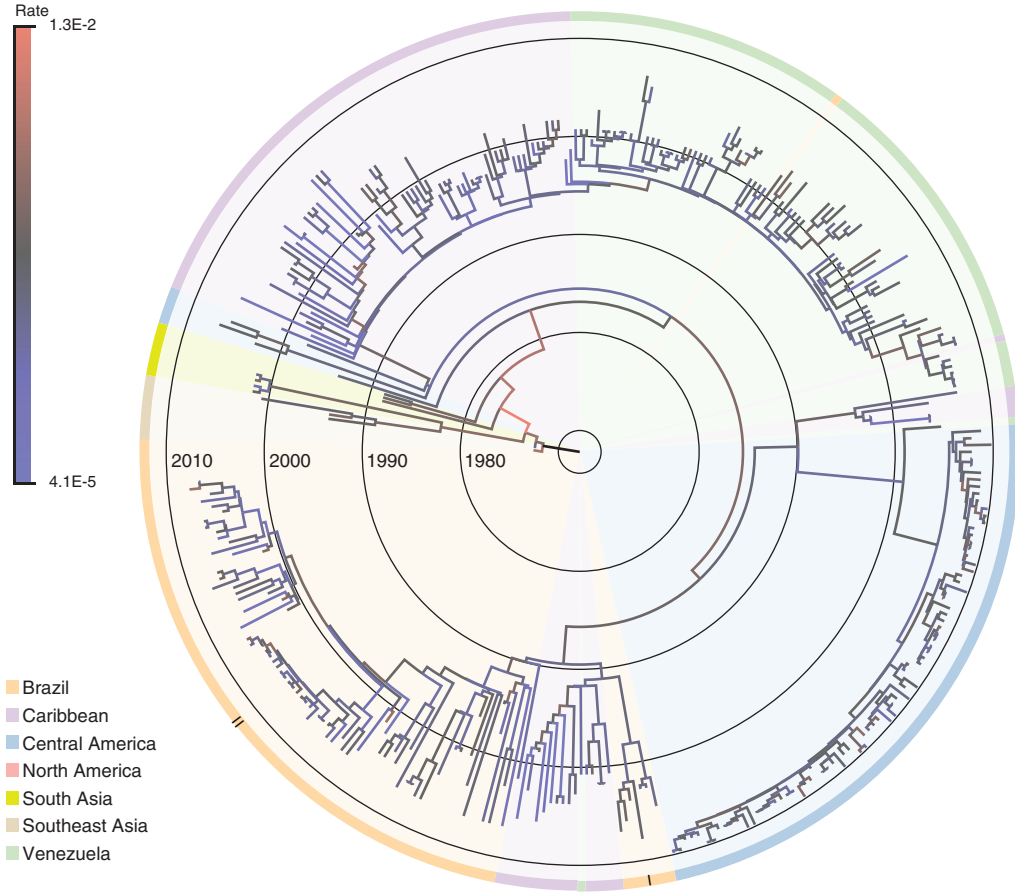

Figure 1. A nontraditional and critically important application in computational statistics is the reconstruction of evolutionary histories in the form of phylogenetic trees. Here is a maximum clade credible tree of the Dengue virus example. The dataset consists of 352 sequences of the serotype 3 of the Dengue virus. Branches are color coded by the posterior means of the branch-specific evolutionary rates according to the color bar on the top left. The concentric circles indicate the timescale with the year numbers. The outer ring indicates the geographic locations of the samples by the color code on the bottom left. '**I**' and '**II**' indicate the two Brazilian lineages as in the original study.

these performance gains available to scientists everywhere through the popular, open-source software package for viral phylogenetic inference *Bayesian evolutionary analysis by sampling trees* (BEAST)[75]. In Section 4.1, we discuss how software package such as BEAST addresses Core Challenge 4, the creation of fast, flexible, and friendly statistical algo-ware.

4 Core Challenges 4 and 5

Section 3 provides examples of how computational statisticians might address Core Challenges 1–3 (big N, big P, and big M) for individual models. Such advances in computational methods must be accompanied

DOI: 10.1002/9781118445112.stat08324

by easy-to-use software to make them accessible to end users. As Gentle *et al*.[76] put it, "While referees and editors of scholarly journals determine what statistical theory and methods are published, the developers of the major statistical software packages determine what statistical methods are used." We would like statistical software to be widely applicable yet computationally efficient at the same time. Trade-offs invariably arise between these two desiderata, but one should nonetheless strive to design algorithms that are general enough to solve an important class of problems and as efficiently as possible in doing so.

Section 4.1 presents Core Challenge 4, achieving "algo-ware" (a neologism suggesting an equal emphasis on the statistical algorithm and its implementation) that is sufficiently efficient, broad, and user-friendly to empower everyday statisticians and data scientists. Core Challenge 5 (Section 4.2) explores the mapping of these algorithms to computational hardware for optimal performance. Hardware-optimized implementations often exploit model-specific structures, but good, general-purpose software should also optimize common routines.

4.1 Fast, Flexible, and Friendly Statistical Algo-Ware

To accommodate the greatest range of models while remaining simple enough to encourage easy implementation, inference methods should rely solely on the quantities that can be computed algorithmically for any given model. The log-likelihood (or log-density in the Bayesian setting) is one such quantity, and one can employ the computational graph framework[77, 78] to evaluate conditional log-likelihoods for any subset of model parameters as well as their gradients via backpropagation[79]. Beyond being efficient in terms of the first three Core Challenges, an algorithm should demonstrate robust performance on a reasonably wide range of problems without extensive tuning if it is to lend itself to successful software deployment.

HMC (Section 2.2) is a prominent example of a general-purpose algorithm for Bayesian inference, only requiring the log-density and its gradient. The generic nature of HMC has opened up possibilities for complex Bayesian modeling as early as Neal[80], but its performance is highly sensitive to model parameterization and its three tuning parameters, commonly referred to as trajectory length, step size, and mass matrix[27]. Tuning issues constitute a major obstacle to the wider adoption of the algorithm, as evidenced by the development history of the popular HMC-based probabilistic programming software Stan[81], which employs the *No-U-Turn* sampler (NUTS) of Hoffman and Gelman[82] to make HMC user-friendly by obviating the need to tune its trajectory length. Bayesian software packages such as Stan empirically adapt the remaining step size and mass matrix[83]; this approach helps make the use of HMC automatic though is not without issues[84] and comes at the cost of significant computational overhead.

Although HMC is a powerful algorithm that has played a critical role in the emergence of general-purpose Bayesian inference software, the challenges involved in its practical deployment also demonstrate how an algorithm – no matter how versatile and efficient at its best – is not necessarily useful unless it can be made easy for practitioners to use. It is also unlikely that one algorithm works well in all situations. In fact, there are many distributions on which HMC performs poorly[83, 85, 86]. Additionally, HMC is incapable of handling discrete distributions in a fully general manner despite the progresses made in extending HMC to such situations[87, 88].

But broader applicability comes with its own challenges. Among sampling-based approaches to Bayesian inference, the Gibbs sampler[89, 90] is, arguably, the most versatile of the MCMC methods. The algorithm simplifies the task of dealing with a complex multidimensional posterior distribution by factorizing the posterior into simpler conditional distributions for blocks of parameters and iteratively updating parameters from their conditionals. Unfortunately, the efficiency of an individual Gibbs sampler depends on its specific factorization and the degree of dependence between its blocks of parameters. Without a careful design or in the absence of effective factorization, therefore, Gibbs samplers' performance may lag behind alternatives such as HMC[91].

Wiley StatsRef: Statistics Reference Online, © 2014 – 2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

On the other hand, Gibbs samplers often require little tuning and can take advantage of highly optimized algorithms for each conditional update, as done in the examples of Section 3. A clear advantage of the Gibbs sampler is that it tends to make software implementation quite modular; for example, each conditional update can be replaced with the latest state-of-the-art samplers as they appear[92], and adding a new feature may amount to no more than adding a single conditional update[75]. In this way, an algorithm may not work in a completely model-agnostic manner but with a broad enough scope can serve as a valuable recipe or meta-algorithm for building model-specific algorithms and software. The same is true for optimization methods. Even though its "E"-step requires a derivation (by hand) for each new model, the EM algorithm[93] enables maximum-likelihood estimation for a wide range of models. Similarly, variational inference (VI) for approximate Bayes requires manual derivations but provides a general framework to turn posterior computation into an optimization problem[94]. As meta-algorithms, both EM and VI expand their breadth of use by replacing analytical derivations with Monte Carlo estimators but suffer losses in statistical and computational efficiency[95, 96]. Indeed, such trade-offs will continue to haunt the creation of fast, flexible, and friendly statistical algo-ware well into the twenty-first century.

4.2 Hardware-Optimized Inference

But successful statistical inference software must also interact with computational hardware in an optimal manner. Growing datasets require the computational statistician to give more and more thought to how the computer implements any statistical algorithm. To effectively leverage computational resources, the statistician must (i) identify the routine's computational bottleneck (Section 2.1) and (ii) algorithmically map this rate-limiting step to available hardware such as a multicore or vectorized CPU, a many-core GPU, or – in the future – a quantum computer. Sometimes, the first step is clear theoretically: a naive implementation of the high-dimensional regression example of Section 3.1 requires an order $\mathcal{O}(N^2P)$ matrix multiplication followed by an order $\mathcal{O}(P^3)$ Cholesky decomposition. Other times, one can use an instruction-level program profiler, such as INTEL VTUNE (Windows, Linux) or INSTRUMENTS (OSX), to identify a performance bottleneck. Once the bottleneck is identified, one must choose between computational resources, or some combination thereof, based on relative strengths and weaknesses as well as natural parallelism of the target task.

Multicore CPU processing is effective for parallel completion of multiple, mostly independent tasks that do not require intercommunication. One might generate 2 to, say, 72 independent Markov chains on a desktop computer or shared cluster. A positive aspect is that the tasks do not have to involve the same instruction sets at all; a negative is *latency*, that is, that the slowest process dictates overall runtime. It is possible to further speed up CPU computing with single instruction, multiple data (SIMD) or vector processing. A small number of vector processing units (VPUs) in each CPU core can carry out a single set of instructions on data stored within an extended-length register. Intel's streaming SIMD extensions (SSE), advance vector extensions (AVX), and AVX-512 allow operations on 128-, 256-, and 512-bit registers. In the context of 64-bit double precision, theoretical speedups for SSE, AVX, and AVX-512 are two-, four-, and eightfold. For example, if a computational bottleneck exists within a for-loop, one can unroll the loop and perform operations on, say, four consecutive loop bodies at once using AVX[21, 22]. Conveniently, languages such as OPENMP[97] make SIMD loop optimization transparent to the user[98]. Importantly, SIMD and multicore optimization play well together, providing multiplicative speedups.

While a CPU may have tens of cores, GPUs accomplish fine-grained parallelization with thousands of cores that apply a single instruction set to distinct data within smaller workgroups of tens or hundreds of cores. Quick communication and shared cache memory within each workgroup balance full parallelization across groups, and dynamic on- and off-loading of the many tasks hide the latency that is so problematic for multicore computing. Originally designed for efficiently parallelized matrix math calculations arising

DOI: 10.1002/9781118445112.stat08324

from image rendering and transformation, GPUs easily speed up tasks that are tensor multiplication intensive such as deep learning[99] but general-purpose GPU applications abound. Holbrook *et al*.[21] provide a larger review of parallel computing within computational statistics. The same paper reports a GPU providing 200-fold speedups over single-core processing and 10-fold speedups over 12-core AVX processing for likelihood and gradient calculations while sampling from a Bayesian multidimensional scaling posterior using HMC at scale. Holbrook *et al*.[22] report similar speedups for inference based on spatiotemporal Hawkes processes. Neither application involves matrix or tensor manipulations.

A quantum computer acts on complex data vectors of magnitude 1 called qubits with gates that are mathematically equivalent to unitary operators[100]. Assuming that engineers overcome the tremendous difficulties involved in building a practical quantum computer (where practicality entails simultaneous use of many quantum gates with little additional noise), twenty-first century statisticians might have access to quadratic or even exponential speedups for extremely specific statistical tasks. We are particularly interested in the following four quantum algorithms: quantum search[101], or finding a single 1 amid a collection of 0s, only requires $\mathcal{O}(\sqrt{N})$ queries, delivering a quadratic speedup over classical search; quantum counting[102], or finding the number of 1s amid a collection of 0s, only requires $\mathcal{O}(\sqrt{N/M})$ (where M is the number of 1s) and could be useful for generating p-values within Monte Carlo simulation from a null distribution (Section 2.1); to obtain the gradient of a function (e.g., the log-likelihood for Fisher scoring or HMC) with a quantum computer, one only needs to evaluate the function once[103] as opposed to $\mathcal{O}(P)$ times for numerical differentiation, and there is nothing stopping the statistician from using, say, a GPU for this single function call; and finally, the HHL algorithm[104] obtains the scalar value $\mathbf{q}^T\mathbf{M}\mathbf{q}$ for the P-vector $\mathbf{q}$ satisfying $\mathbf{A}\mathbf{q} = \mathbf{b}$ and $\mathbf{M}$ and $P \times P$ matrix in time $\mathcal{O}(\log(P\kappa^2))$, delivering an exponential speedup over classical methods. Technical caveats exist[105], but HHL may find use within high-dimensional hypothesis testing (big P). Under the null hypothesis, one can rewrite the score test statistic

$$\mathbf{u}^T(\hat{\theta}_0)\, \mathcal{I}^{-1}(\hat{\theta}_0)\, \mathbf{u}(\hat{\theta}_0) \quad \text{as} \quad \mathbf{u}^T(\hat{\theta}_0)\, \mathcal{I}^{-1}(\hat{\theta}_0)\, \mathcal{I}(\hat{\theta}_0)\, \mathcal{I}^{-1}(\hat{\theta}_0)\, \mathbf{u}(\hat{\theta}_0)$$

for $\mathcal{I}(\hat{\theta}_0)$ and $\mathbf{u}(\hat{\theta}_0)$, the Fisher information and log-likelihood gradient evaluated at the maximum-likelihood solution under the null hypothesis. Letting $\mathbf{A} = \mathcal{I}(\hat{\theta}_0) = \mathbf{M}$ and $\mathbf{b} = \mathbf{u}(\hat{\theta}_0)$, one may write the test statistic as $\mathbf{q}^T\mathbf{M}\mathbf{q}$ and obtain it in time logarithmic in P. When the model design matrix $\mathbf{X}$ is sufficiently sparse – a common enough occurrence in large-scale regression – to render $\mathcal{I}(\hat{\theta}_0)$ itself sparse, the last criterion for the application of the HHL algorithm is met.

5 Rise of Data Science

Core Challenges 4 and 5 – fast, flexible, and user-friendly algo-ware and hardware-optimized inference – embody an increasing emphasis on application and implementation in the age of data science. Previously undervalued contributions in statistical computing, for example, hardware utilization, database methodology, computer graphics, statistical software engineering, and the human–computer interface[76], are slowly taking on greater importance within the (rather conservative) discipline of statistics. There is perhaps no better illustration of this trend than Dr. Hadley Wickham's winning the prestigious COPSS Presidents' Award for 2019

> [for] influential work in statistical computing, visualization, graphics, and data analysis; for developing and implementing an impressively comprehensive computational infrastructure for data analysis through R software; for making statistical thinking and computing accessible to large audience; and for enhancing an appreciation for the important role of statistics among data scientists[106].

Wiley StatsRef: Statistics Reference Online, © 2014–2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

This success is all the more impressive because Presidents' Awardees have historically been contributors to statistical theory and methodology, not Dr. Wickham's scientific software development for data manipulation[107–109] and visualization[110, 111].

All of this might lead one to ask: *does the success of data science portend the declining significance of computational statistics and its Core Challenges?* Not at all! At the most basic level, data science's emphasis on application and implementation underscores the need for computational thinking in statistics. Moreover, the scientific breadth of data science brings new applications and models to the attention of statisticians, and these models may require or inspire novel algorithmic techniques. Indeed, we look forward to a golden age of computational statistics, in which statisticians labor within the intersections of mathematics, parallel computing, database methodologies, and software engineering with impact on the entirety of the applied sciences. After all, significant progress toward conquering the Core Challenges of computational statistics requires that we use every tool at our collective disposal.

Acknowledgments

AJH is supported by NIH grant K25AI153816. MAS is supported by NIH grant U19AI135995 and NSF grant DMS1264153.

End Notes

1. *Statistical inference* is an umbrella term for hypothesis testing, point estimation, and the generation of (confidence or credible) intervals for population functionals (mean, median, correlations, etc.) or model parameters.
2. We present the problem of phylogenetic reconstruction in Section 3.2 as one such example arising from the field of molecular epidemiology.
3. The use of "N" and "P" to denote observation and parameter count is common. We have taken liberties in coining the use of "M" to denote mode count.
4. A more numerically stable approach has the same complexity[24].
5. The matrix parameter $\boldsymbol{\Omega}$ coincides with $\boldsymbol{\Omega} = \sigma^{-2}\mathbf{I}_N$ for linear regression and $\boldsymbol{\Omega} = \text{diag}(\boldsymbol{\omega})$ for auxiliary Pólya-Gamma parameter $\boldsymbol{\omega}$ for logistic regression[56, 57].
6. See Nishimura and Suchard[57] and references therein for the role and design of a preconditioner.

Related Articles

Bayesian Inference; **Big Data**; **Big Data in Biosciences**; **Computer-Intensive Statistical Methods**; **Computers and Statistics**; **Data Science**; **Gibbs Sampling**; **Hamiltonian Monte Carlo**; **Image Restoration and Reconstruction**; **Importance Sampling including the Bootstrap**; **Machine Learning**; **Multivariate Data Visualisation**; **Markov Chain Monte Carlo (MCMC)**; **Statistical Graphics**; **Statistical Software**; **Software Reliability**

References

[1] Davenport, T.H. and Patil, D. (2012) Data scientist. *Harvard Bus. Rev.*, **90**, 70–76.
[2] Google Trends (2020) Data source: Google trends. *https://trends.google.com/trends* (accessed 12 July 2020)

DOI: 10.1002/9781118445112.stat08324

[3] American Statistical Association (2020) *Statistics Degrees Total and By Gender*, *https://ww2.amstat.org/misc/StatTable1987-Current.pdf* (accessed 01 June 2020).
[4] Cleveland, W.S. (2001) Data science: an action plan for expanding the technical areas of the field of statistics. *Int. Stat. Rev.*, **69**, 21–26.
[5] Donoho, D. (2017) 50 Years of data science. *J. Comput. Graph. Stat.*, **26**, 745–766.
[6] Fisher, R.A. (1936) Design of experiments. *Br Med J 1.3923*, 554–554.
[7] Fisher, R.A. (1992) Statistical methods for research workers, in Kotz S., Johnson N.L. (eds) *Breakthroughs in Statistics*, Springer Series in Statistics (Perspectives in Statistics). Springer, New York, NY. (Especially Section 21.02). https://doi.org/10.1007/978-1-4612-4380-9_6.
[8] Wald, A. and Wolfowitz, J. (1944) Statistical tests based on permutations of the observations. *Ann. Math. Stat.*, **15**, 358–372.
[9] Efron B. (1992) Bootstrap methods: another look at the jackknife, in Kotz S., Johnson N.L. (eds) *Breakthroughs in Statistics*. Springer Series in Statistics (Perspectives in Statistics). Springer, New York, NY, pp. 569–593. https://doi.org/10.1007/978-1-4612-4380-9_41.
[10] Efron, B. and Tibshirani, R.J. (1994) *An Introduction to the Bootstrap*, CRC press.
[11] Bliss, C.I. (1935) The comparison of dosage-mortality data. *Ann. Appl. Biol.*, **22**, 307–333 (Fisher introduces his scoring method in appendix).
[12] McCullagh, P. and Nelder, J. (1989) *Generalized Linear Models*, 2nd edn, Chapman and Hall, London. Standard book on generalized linear models.
[13] Tierney, L. (1994) Markov chains for exploring posterior distributions. *Ann. Stat.*, **22**, 1701–1728.
[14] Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011) *Handbook of Markov Chain Monte Carlo*, CRC press.
[15] Chavan, V., Phursule, R.N. (2014) Survey paper on big data. *Int. J. Comput. Sci. Inf. Technol.*, **5**, 7932–7939.
[16] Williams, C.K. and Rasmussen, C.E. (1996) Gaussian processes for regression. Advances in Neural Information Processing Systems, pp. 514–520.
[17] Williams, C.K. and Rasmussen, C.E. (2006) *Gaussian Processes for Machine Learning*, vol. 2, MIT press, Cambridge, MA.
[18] Gelman, A., Carlin, J.B., Stern, H.S. *et al*. (2013) *Bayesian Data Analysis*, CRC press.
[19] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N. *et al*. (1953) Equation of state calculations by fast computing machines. *J. Chem. Phys.*, **21**, 1087–1092.
[20] Hastings, W.K. (1970) Monte Carlo sampling methods using Markov chains and their applications, *Biometrika*, **57** (1), 97–109. https://doi.org/10.1093/biomet/57.1.97.
[21] Holbrook, A.J., Lemey, P., Baele, G. *et al*. (2020) Massive parallelization boosts big Bayesian multidimensional scaling. *J. Comput. Graph. Stat.*, 1–34.
[22] Holbrook, A.J., Loeffler, C.E., Flaxman, S.R. *et al*. (2021) Scalable Bayesian inference for self-excitatory stochastic processes applied to big American gunfire data, *Stat. Comput.* **31**, 4.
[23] Seber, G.A. and Lee, A.J. (2012) *Linear Regression Analysis*, vol. 329, John Wiley & Sons.
[24] Trefethen, L.N. and Bau, D. (1997) Numerical linear algebra. *Soc. Ind. Appl. Math.*
[25] Gelman, A., Roberts, G.O., Gilks, W.R. (1996) Efficient metropolis jumping rules. *Bayesian Stat.*, **5**, 42.
[26] Van Dyk, D.A. and Meng, X.-L. (2001) The art of data augmentation. *J. Comput. Graph. Stat.*, **10**, 1–50.
[27] Neal, R.M. (2011) MCMC using Hamiltonian dynamics, in Steve Brooks, Andrew Gelman, Galin L. Jones and Xiao-Li Meng (eds) *Handbook of Markov Chain Monte Carlo*, Chapman and Hall/CRC Press, 113–162.
[28] Holbrook, A., Vandenberg-Rodes, A., Fortin, N., and Shahbaba, B. (2017) A Bayesian supervised dual-dimensionality reduction model for simultaneous decoding of LFP and spike train signals. *Stat*, **6**, 53–67.
[29] Bouchard-Côté, A., Vollmer, S.J., and Doucet, A. (2018) The bouncy particle sampler: a nonreversible rejection-free Markov chain Monte Carlo method. *J. Am. Stat. Assoc.*, **113**, 855–867.
[30] Murty, K.G. and Kabadi, S.N. (1985) Some NP-Complete Problems in Quadratic and Nonlinear Programming. *Tech. Rep.*
[31] Kennedy, J. and Eberhart, R. (1995) *Particle Swarm Optimization*. Proceedings of ICNN'95-International Conference on Neural Networks, vol. 4, pp. 1942–1948. IEEE.
[32] Davis, L. (1991) *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York.
[33] Hunter, D.R. and Lange, K. (2004) A tutorial on MM algorithms. *Am. Stat.*, **58**, 30–37.
[34] Boyd, S., Boyd, S.P., and Vandenberghe, L. (2004) *Convex Optimization*, Cambridge University Press.
[35] Fisher, R.A. (1922) On the mathematical foundations of theoretical statistics. *Philos. Trans. R. Soc. London, Ser. A*,**222**,309–368.
[36] Beale, E., Kendall, M., and Mann, D. (1967) The discarding of variables in multivariate analysis. *Biometrika*, **54**, 357–366.
[37] Hocking, R.R. and Leslie, R. (1967) Selection of the best subset in regression analysis. *Technometrics*, **9**, 531–540.
[38] Tibshirani, R. (1996) Regression shrinkage and selection via the lasso. *J. R. Stat. Soc., Ser. B*, **58**,267–288.
[39] Geyer, C. (1991) Markov chain Monte Carlo maximum likelihood, In *Computing science and statistics: Proceedings of 23rd Symposium on the Interface Interface Foundation, Fairfax Station*, 156–163.

DOI: 10.1002/9781118445112.stat08324

[40] Tjelmeland, H. and Hegstad, B.K. (2001) Mode jumping proposals in MCMC. *Scand. J. Stat.*, **28**, 205–223.
[41] Lan, S., Streets, J., and Shahbaba, B. (2014) *Wormhole Hamiltonian Monte Carlo*. Twenty-Eighth AAAI Conference on Artificial Intelligence.
[42] Nishimura, A. and Dunson, D. (2016) Geometrically tempered Hamiltonian Monte Carlo. *arXiv preprint arXiv:1604.00872*.
[43] Mitchell, T.J. and Beauchamp, J.J. (1988) Bayesian variable selection in linear regression. *J. Am. Stat. Assoc.*, **83**, 1023–1032.
[44] Madigan, D. and Raftery, A.E. (1994) Model selection and accounting for model uncertainty in graphical models using Occam's window. *J. Am. Stat. Assoc.*, **89**, 1535–1546.
[45] George, E.I. and McCulloch, R.E. (1997) Approaches for Bayesian variable selection. *Statistica Sinica*, **7**, 339–373.
[46] Hastie, T., Tibshirani, R., and Wainwright, M. (2015) *Statistical Learning with Sparsity: The Lasso and Generalizations*, CRC Press.
[47] Friedman, J., Hastie, T., and Tibshirani, R. (2010) Regularization paths for generalized linear models via coordinate descent. *J. Stat. Softw.*, **33**, 1.
[48] Bhattacharya, A., Chakraborty, A., and Mallick, B.K. (2016) Fast sampling with Gaussian scale mixture priors in high-dimensional regression. *Biometrika*, **103**, 985–991.
[49] Suchard, M.A., Schuemie, M.J., Krumholz, H.M. *et al*. (2019) Comprehensive comparative effectiveness and safety of first-line antihypertensive drug classes: a systematic, multinational, large-scale analysis. *The Lancet*, **394**, 1816–1826.
[50] Passos, I.C., Mwangi, B., and Kapczinski, F. (2019) *Personalized Psychiatry: Big Data Analytics in Mental Health*, Springer.
[51] Svensson, V., da Veiga Beltrame, E., and Pachter, L. (2019) A curated database reveals trends in single-cell transcriptomics. *bioRxiv 742304*.
[52] Nott, D.J. and Kohn, R. (2005) Adaptive sampling for Bayesian variable selection. *Biometrika*, **92**, 747–763.
[53] Ghosh, J. and Clyde, M.A. (2011) Rao–Blackwellization for Bayesian variable selection and model averaging in linear and binary regression: a novel data augmentation approach. *J. Am. Stat. Assoc.*, **106**,1041–1052.
[54] Carvalho, C.M., Polson, N.G., and Scott, J.G. (2010) The horseshoe estimator for sparse signals. *Biometrika*, **97**,465–480.
[55] Polson, N.G. and Scott, J.G. (2010) Shrink globally, act locally: sparse Bayesian regularization and prediction. *Bayesian Stat.*, **9**, 501–538.
[56] Polson, N.G., Scott, J.G., and Windle, J. (2013) Bayesian inference for logistic models using Pólya–Gamma latent variables. *J. Am. Stat. Assoc.*, **108**, 1339–1349.
[57] Nishimura, A. and Suchard, M.A. (2018) Prior-preconditioned conjugate gradient for accelerated gibbs sampling in "large n & large p" sparse Bayesian logistic regression models. *arXiv:1810.12437*.
[58] Rue, H. and Held, L. (2005) *Gaussian Markov Random Fields: Theory and Applications*, CRC Press.
[59] Hestenes, M.R. and Stiefel, E. (1952) Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.*, **49**, 409–436.
[60] Lanczos, C. (1952) Solution of systems of linear equations by minimized iterations. *J. Res. Nat. Bur. Stand.*, **49**, 33–53.
[61] Van der Vorst, H.A. (2003) *Iterative Krylov Methods for Large Linear Systems*, vol. 13, Cambridge University Press.
[62] Cipra, B.A. (2000) The best of the 20th century: editors name top 10 algorithms. *SIAM News*, **33**, 1–2.
[63] Dongarra, J., Heroux, M.A., and Luszczek, P. (2016) High-performance conjugate-gradient benchmark: a new metric for ranking high-performance computing systems. *Int. J. High Perform. Comput. Appl.*, **30**, 3–10.
[64] Zhang, L., Zhang, L., Datta, A., and Banerjee, S. (2019) Practical Bayesian modeling and inference for massive spatial data sets on modest computing environments. *Stat. Anal. Data Min.*, **12**, 197–209.
[65] Golub, G.H. and Van Loan, C.F. (2012) *Matrix Computations*, vol. 3, Johns Hopkins University Press.
[66] Pybus, O.G., Tatem, A.J., and Lemey, P. (2015) Virus evolution and transmission in an ever more connected world. *Proc. R. Soc. B: Biol. Sci.*, **282**, 20142878.
[67] Bloom, D.E., Black, S., and Rappuoli, R. (2017) Emerging infectious diseases: a proactive approach. *Proc. Natl. Acad. Sci. U.S.A.*, **114**, 4055–4059.
[68] Pybus, O.G., Suchard, M.A., Lemey, P. *et al*. (2012) Unifying the spatial epidemiology and molecular evolution of emerging epidemics. *Proc. Natl. Acad. Sci. U.S.A.*, **109**, 15066–15071.
[69] Nunes, M.R., Palacios, G., Faria, N.R. *et al*. (2014) Air travel is associated with intracontinental spread of dengue virus serotypes 1–3 in Brazil. *PLoS Negl. Trop. Dis.*, **8**, e2769.
[70] Bletsa, M., Suchard, M.A., Ji, X. *et al*. (2019) Divergence dating using mixed effects clock modelling: an application to HIV-1. *Virus Evol.*, **5**, vez036.
[71] Dudas, G., Carvalho, L.M., Bedford, T. *et al*. (2017) Virus genomes reveal factors that spread and sustained the Ebola epidemic. *Nature*, **544**, 309–315.
[72] Elbe, S. and Buckland-Merrett, G. (2017) Data, disease and diplomacy: GISAID's innovative contribution to global health. *Glob. Chall.*, **1**, 33–46.
[73] Ji, X., Zhang, Z., Holbrook, A. *et al*. (2020) Gradients do grow on trees: a linear-time O(N)-dimensional gradient for statistical phylogenetics. *Mol. Biol. Evol.*, **37**, 3047–3060.

DOI: 10.1002/9781118445112.stat08324

[74] Baum, L. (1972) An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process. *Inequalities*, **3**, 1–8.
[75] Suchard, M.A., Lemey, P., Baele, G. *et al*. (2018) Bayesian phylogenetic and phylodynamic data integration using BEAST 1.10. *Virus Evol.*, **4**, vey016.
[76] Gentle, J.E., Härdle, W.K., and Mori, Y. (eds) (2012) How computational statistics became the backbone of modern data science, in *Handbook of Computational Statistics*, Springer, pp. 3–16.
[77] Lunn, D., Spiegelhalter, D., Thomas, A., and Best, N. (2009) The BUGS project: evolution, critique and future directions. *Stat. Med.*, **28**, 3049–3067.
[78] Bergstra, J., Breuleux, O., Bastien, F. *et al*. (2010) *Theano: A CPU and GPU Math Expression Compiler*. Proceedings of the Python for Scientific Computing Conference (SciPy) Oral Presentation.
[79] Rumelhart, D.E., Hinton, G.E., and Williams, R.J. (1986) Learning representations by back-propagating errors. *Nature*, **323**, 533–536.
[80] Neal, R.M. (1996) *Bayesian Learning for Neural Networks*, Springer-Verlag.
[81] Gelman, A. (2014) Petascale Hierarchical Modeling Via Parallel Execution. U.S. Department of Energy. Report No: DE-SC0002099.
[82] Hoffman, M.D. and Gelman, A. (2014) The no-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.*, **15**, 1593–1623.
[83] Stan Development Team (2018) Stan Modeling Language Users Guide and Reference Manual. Version 2.18.0.
[84] Livingstone, S. and Zanella, G. (2019) On the robustness of gradient-based MCMC algorithms. *arXiv:1908.11812*.
[85] Mangoubi, O., Pillai, N.S., and Smith, A. (2018) Does Hamiltonian Monte Carlo mix faster than a random walk on multimodal densities? *arXiv:1808.03230*.
[86] Livingstone, S., Faulkner, M.F., and Roberts, G.O. (2019) Kinetic energy choice in Hamiltonian/hybrid Monte Carlo. *Biometrika*, **106**, 303–319.
[87] Dinh, V., Bilge, A., Zhang, C., and Matsen IV, F.A. (2017) *Probabilistic Path Hamiltonian Monte Carlo*. Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 1009–1018.
[88] Nishimura, A., Dunson, D.B., and Lu, J. (2020) Discontinuous Hamiltonian Monte Carlo for discrete parameters and discontinuous likelihoods. *Biometrika*, **107**, 365–380.
[89] Geman, S. and Geman, D. (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, **PAMI-6**, 721–741.
[90] Gelfand, A.E. and Smith, A.F. (1990) Sampling-based approaches to calculating marginal densities. *J. Am. Stat. Assoc.*, **85**, 398–409.
[91] Monnahan, C.C., Thorson, J.T., and Branch, T.A. (2017) Faster estimation of Bayesian models in ecology using Hamiltonian Monte Carlo. *Methods Ecol. Evol.*, **8**, 339–348.
[92] Zhang, Z., Zhang, Z., Nishimura, A. *et al*. (2020) Large-scale inference of correlation among mixed-type biological traits with phylogenetic multivariate probit models. *Ann. Appl. Stat.*
[93] Dempster, A.P., Laird, N.M., and Rubin, D.B. (1977) Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc., Ser. B*, **39**, 1–22.
[94] Jordan, M.I., Ghahramani, Z., Jaakkola, T.S., and Saul, L.K. (1999) An introduction to variational methods for graphical models. *Mach. Learn.*, **37**, 183–233.
[95] Wei, G.C. and Tanner, M.A. (1990) A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *J. Am. Stat. Assoc.*, **85**, 699–704.
[96] Ranganath, R., Gerrish, S., and Blei, D.M. (2014) *Black Box Variational Inference*. Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics.
[97] Dagum, L. and Menon, R. (1998) OpenMP: an industry standard API for shared-memory programming. *IEEE Comput. Sci. Eng.*, **5**, 46–55.
[98] Warne, D.J., Sisson, S.A., and Drovandi, C. (2019) Acceleration of expensive computations in Bayesian statistics using vector operations. *arXiv preprint arXiv:1902.09046*.
[99] Bergstra, J., Bastien, F., Breuleux, O. *et al*. (2011) *Theano: Deep Learning on GPUS with Python*. NIPS 2011, BigLearning Workshop, Granada, Spain vol. 3, pp. 1–48. Citeseer.
[100] Nielsen, M.A. and Chuang, I. (2002) *Quantum computation and quantum information*, Cambridge University Press.
[101] Grover, L.K. (1996) *A Fast Quantum Mechanical Algorithm for Database Search*. Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pp. 212–219.
[102] Boyer, M., Brassard, G., Høyer, P., and Tapp, A. (1998) Tight bounds on quantum searching. *Fortschritte der Physik: Progress of Physics*, **46**, 493–505.
[103] Jordan, S.P. (2005) Fast quantum algorithm for numerical gradient estimation. *Phys. Rev. Lett.*,**95**, 050501.
[104] Harrow, A.W., Hassidim, A., and Lloyd, S. (2009) Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, **103**, 150502.

Wiley StatsRef: Statistics Reference Online, © 2014–2021 John Wiley & Sons, Ltd.

DOI: 10.1002/9781118445112.stat08324

[105] Aaronson, S. (2015) Read the fine print. *Nat. Phys.*, **11**, 291–293.
[106] COPSS (2020) Committee of Presidents of Statistical Societies, *https://community.amstat.org/copss/awards/winners* (accessed 31 August 2020).
[107] Wickham, H. (2007) Reshaping data with the reshape package. *J. Stat. Soft.*, **21**, 1–20.
[108] Wickham, H. (2011) The split-apply-combine strategy for data analysis. *J. Stat. Soft.*, **40**, 1–29.
[109] Wickham, H. (2014) Tidy data. *J. Stat. Soft.*, **59**, 1–23.
[110] Kahle, D. and Wickham, H. (2013) ggmap: spatial visualization with ggplot2. *R J.*, **5**, 144–161.
[111] Wickham, H. (2016) *ggplot2: Elegant Graphics for Data Analysis*, Springer.

DOI: 10.1002/9781118445112.stat08324