Background to ILike

A general framework, which covers many applications, is that we have a latent (unobserved) process $\mathbf{X}$, whose distribution depends on some parameters of interest θ. We do not observe $\mathbf{X}$ itself, but instead data $\mathbf{Y}$ that contains partial information about $\mathbf{X}$. Note that in some applications $\mathbf{X}$ may have a specific meaning (e.g. the genealogy of a sample of chromosomes), whilst in others it may be a modelling construct (e.g. the allocation of data to components in a mixture model) introduced to help with the analysis. This framework includes so-called missing-data models, state-space models, hierarchical models and latent-variable models amongst others. Note that the ideas below can apply more generally – and we are focussing on this framework just to help make the ideas concrete. It is simple to define the likelihood for such models

$$p(\mathbf{y}|\theta) = \int p(\mathbf{y}|\mathbf{x}, \theta)p(\mathbf{x}|\theta)\mathrm{d}\mathbf{x}, \tag{1}$$

however, the challenge comes from the fact that calculating the required integral is often not possible analytically. Inference is performed either by maximising the likelihood $p(\mathbf{y}|\theta)$ or within the Bayesian paradigm where a prior $p(\theta)$ is introduced and the aim is then to calculate the posterior distribution $p(\theta|\mathbf{y}) \propto p(\theta)p(\mathbf{y}|\theta)$. I-Like research is driven by recent developments in 6 areas:

(B1) Pseudo marginal computations and particle MCMC. In many applications, whilst we cannot calculate $p(\mathbf{y}|\theta)$, we can obtain Monte Carlo estimates of it: for example one can approximate (1) through importance sampling

$$\hat{p}(\mathbf{y}|\theta) = N^{-1} \sum_{i=1}^{N} p(\mathbf{y}, \mathbf{x}^{(i)}|\theta)/q(\mathbf{x}^{(i)}|\mathbf{y}, \theta), \tag{2}$$

where $\mathbf{x}^{(i)}$ are drawn from $q(\cdot|\mathbf{y}, \theta)$. A key question is how we can use such estimates within a Bayesian approach to estimating θ. One approach is to take a standard MCMC algorithm for sampling from $p(\theta|\mathbf{y})$. Such an algorithm will involve repeatedly proposing new values of θ to move to, and then accepting these with an appropriate probability such that the ensuing Markov chain has $p(\theta|\mathbf{y})$ as its stationary distribution. The problem with implementing this is that the acceptance probability depends on the intractable likelihood $p(\mathbf{y}|\theta)$. A naive way to proceed would replace the $p(\mathbf{y}|\theta)$s by Monte Carlo estimates. Intuitively we would expect such an approach to lead to an MCMC algorithm with the wrong stationary distribution, but may hope that if the Monte Carlo estimates are accurate then this wrong stationary distribution would be close to $p(\theta|\mathbf{y})$ as N increases. Remarkably, [1] have shown that, if implemented in the correct way, such an approach is not approximate in that the stationary distribution of the MCMC algorithm is still $p(\theta|\mathbf{y})$. Whilst the variance of the Monte Carlo estimator of the likelihood, controlled by N above, does not affect validity of the resulting MCMC algorithm, it does affect its mixing properties.

This type of idea has been shown to be particularly useful in the context of inference in state-space models. Particle MCMC [2] gives a way of combining existing Monte Carlo methods for state-space models, namely particle filters, with MCMC. At its simplest it involves running an MCMC algorithm that targets $p(\theta|\mathbf{y})$, but using the particle filter to produce estimates of $p(\mathbf{y}|\theta)$ that are used within the accept-reject step of the MCMC algorithm. More complicated and efficient versions, that target $p(\theta, \mathbf{x}|\mathbf{y})$, also exist. These ideas are related to the pseudo-marginal approach, and again have the property that whilst the likelihood is replaced by an estimate, the resulting MCMC algorithm is still exact (in that it has the correct target distribution). It has been shown that particle MCMC can be substantially more efficient than standard MCMC algorithms.

(B2) Likelihood-free methods. So called *likelihood-free methods* are actually likelihood-based methods for inference, but methods where the need to calculate a likelihood is replaced through the use of simu-

lation from the underlying model. Simply put, the basic idea is that we can approximate the likelihood for a given parameter value by computing the proportion of data sets simulated from the model with that parameter value that are *similar to* the observed data. One popular approach is Approximate Bayesian Computation (ABC) [3]. Here "similar to" is defined in terms of appropriately chosen summaries of the data, $S(\cdot)$, and some kernel $K(\cdot,\cdot)$ that measures the discrepancy between sets of summaries. Formally this leads to an approximation to the likelihood which can be defined as

$$p_{\text{ABC}}(\mathrm{y}|\theta) = \int K(S(\mathbf{y}), S(\mathbf{u}))p(\mathbf{u}|\theta)\mathrm{d}\mathbf{u}. \tag{3}$$

This in turn leads to an approximate posterior $p_{\text{ABC}}(\theta|\mathrm{y}) \propto p(\theta)p_{\text{ABC}}(\mathrm{y}|\theta)$. The advantage of using this approximation stems from the fact that it is possible to sample from $p_{\text{ABC}}(\theta|\mathrm{y})$ using only the ability to simulate new data sets $\mathbf{u}$ from $p(\mathbf{u}|\theta)$ [4]. Alternative likelihood-free approaches include Indirect Inference [5], and simulated likelihood methods [6] amongst others. The popularity of these methods is due to their efficiency in complex-model situations where simulation is straightforward, where often there are no other likelihood-based options for inference. Furthermore they are easy to adapt to a range of models, as only the simulation algorithm needs to be changed.

(B3) Composite and pseudo likelihoods. A pragmatic approach to inference is often to use an approximate likelihood. In many cases these can be viewed in terms of the likelihood for an approximation to the model of interest, such as the PAC likelihood [7]; or based on approximation to the likelihood, such as via truncating a Taylor-expansion as in the INLA method [8]. A key open question is how to evaluate the accuracy of the approximations and the effect this has on inferences. Alternatively there are generic approximate likelihood methods — perhaps the most general and relevant for analysing complex models and large data being composite likelihood. This approach creates a log-likelihood as a weighted sum of the log-likelihoods for different subsets of the data. For example, for $\mathbf{y} = (y_1, \ldots, y_n)$ we could consider all individual data points, and all pairs of data points [9]

$$Cl(\theta) = \sum_{i=1}^{n} w_i \log p(y_i|\theta) + \sum_{i=2}^{n}\sum_{j=1}^{i-1} w_{ij} \log p(y_i, y_j|\theta). \tag{4}$$

Theory exists which gives conditions under which maximising such a composite likelihood will lead to a consistent estimator, and also expressions for the asymptotic variance of this estimator. One motivation for using composite likelihoods is that they can give robust estimates: as correct model specification is needed only for the subsets used, and thus the results are not affected by any assumptions about higher dependencies in the data: so the above composite likelihood (4) only requires modelling the marginal and pairwise distributions, and does not impose any assumptions about the further dependence among, say, triples of data.

(B4) Simulation and inference for intractable models. Many models of interest are intractable: often because the underlying latent process needed to define a tractable distribution $p(\mathbf{y}|\mathbf{x})$ is infinite dimensional. Two important examples are diffusion processes, where $\mathbf{x}$ is the sample path of the diffusions, and Dirichlet process models, where $\mathbf{x}$ describes the weights and parameters of the infinitely many classes in the model. It seems inevitable that for such models we need to resort to approximations, where the infinite-dimensional $\mathbf{x}$ is replaced by a finite-dimension approximation. For the diffusion example, such an x may be the value of the path at a fixed grid of time-points; for the Dirichlet process model, these are often defined in terms of truncating the number of classes. However, for many such models, simulation and inference are possible without resorting to approximation. The key idea underlying these methods is that it is possible to replace x by a finite, but random dimensional, object, $\mathbf{U}$ say, in such a way that

$$p(\mathbf{y}|\mathbf{x}) = \mathrm{E}(p(\mathbf{y}|\mathbf{U})),$$

where $p(\mathbf{y}|\mathbf{U})$ is tractable. For Dirichlet process models U is defined in terms of a random truncation [10]; for diffusions, such a U consists of the value of the path a random number of randomly chosen time-points [11, 12]. Whilst the initial work on diffusion had limited applicability, recent work has developed generalisations (based around continuous-time extensions of sequential importance sampling) which enable such ideas to be applied to most continuous-time Markov process models. In some cases, these methods give us a way of simulating an event of a certain probability, p, even when we cannot calculate p. As a simple example, if X is the diffusion satisfying SDE

$$dX_t = \alpha(X_t)dt + dB_t\ , \quad X_0 = 0$$

on time interval $[0, T]$, with corresponding probability law $\mathbb{P}$ then under weak regularity conditions, the Radon-Nikodym derivative with respect to Wiener measure $\mathbb{W}$ is given by

$$\frac{d\mathbb{P}}{d\mathbb{W}} = G(X) = \exp\left\{\int_0^T \alpha(X_{s-})dX_s - \frac{1}{2}\int_0^T \alpha(X_{s-})^2 ds\right\}.$$

Whilst we cannot calculate $G(X)$, [11] show how we can simulate from an event of probability $p = K_1 G(X)$ for some known constant $K_1 \leq 1$. However, within an MCMC algorithm the acceptance probability will be a function of this unknown probability p. If we can simulate events with probability p, can we use such simulations to implement an appropriate MCMC algorithm? This is a classical problem in computational probability, termed the Bernoulli factory problem. One example is: given an ability to simulate from events of probability p we wish to construct events of probability $2p$. Whilst the existence of an algorithm for doing this was established in [13] the first practical algorithm of this type was recently developed in [14] and exploits the structure of analytic functions in a simulation context. There are prospects of being able to generalise this to the case of events of the probability $f(p)$ for suitably regular f, and further to the transformation of entire random variables.

(B5) Adaptive Monte Carlo. The implementation of the most popular and flexible Monte Carlo methods currently used requires the user to make a significant number of choices. These choices are known to be crucial in order to ensure good performance, traditionally measured in terms of the variability of the estimators derived from samples generated by the procedure. In the context of Markov chain Monte Carlo algorithms the choice of the proposal distribution is key to the performance of the procedure. For example it is well known that the distribution of the increments of a random walk based Metropolis-Hastings algorithm should capture the dependence structure and scale heterogeneity of the probability distribution to be explored. In general an MCMC algorithm is parameterised by some parameter $\gamma \in \Gamma$ for some set Γ and the user is required to choose among a family of MCMC transition probabilities $\{P_\gamma, \gamma \in \Gamma\}$ in order to design an efficient algorithm. The optimal parameter is usually unknown a priori. Adaptive MCMC [15] algorithms aim to automate the choice of γ by using the history of the chain. Implementing this is non-trivial, as most methods of adaptation violate the Markov property, and we need to ensure the resulting adaptive MCMC algorithm still has the correct stationary distribution. There are two key ingredients to these algorithms: the specification of an optimality criterion and the design of efficient optimisation procedures. Well known theoretical criteria are the speed of convergence and the asymptotic variance of ergodic averages, which may however be difficult to optimize practically. Instead simpler and more tractable proxies have been proposed, such as the acceptance probability at stationarity for random walk or Langevin diffusion based Metropolis-Hastings algorithms. It has been shown that such criteria can be efficiently optimized by MCMC algorithms which iteratively improve their performance by learning from past samples. Key to the validity and efficiency of these algorithms is the updating rule for the parameter to be optimized, and in particular the central notion of vanishing adaptation. This important property allows one to recover asymptotically the correct ergodicity properties lost by the introduction of the learning mechanism above and required for steady state optimality criteria.

(B6) Modern many-core computer architecture. The 20th Century saw single-threaded computational power increase exponentially. However, as we reach the limit of single processing speed manufacturers are looking towards many-core architectures: it is possible to provide more processing power by putting more cores onto a single die. The result of this trend is that computational algorithms which take advantage of multiple threads can see significant linear speedup with the number of cores available. For concreteness we will describe just one example of many-core architectures, Graphical processing units (GPUs). These are specialized processors with dedicated memory that conventionally perform floating point operations required for rendering graphics. In response to commercial demand for real-time graphics rendering, the current generation of GPUs have evolved into many-core processors that are specifically designed to perform data-parallel computation. The main difference between GPUs and central processing units (CPUs) is that GPUs devote proportionally more transistors to arithmetic logic units and less to caches and flow control. Moreover, GPUs have high-speed memory access with very low latency and very high bandwidth. They are dedicated and local which is important for private data security. This has led us as well as others to explore their potential as computing devices for statistical computation to good effect [16, 17]. Algorithms suited to many-core GPU simulation will exploit data-parallel computation. This is a computation that has been parallelized by distributing the data amongst computing nodes. It can be contrasted with a task-parallel computation, in which the distribution of computing tasks is emphasized. One framework that is used to accomplish data-parallelism is "single instruction, multiple data" (SIMD), in which multiple processors execute the same instructions on different data. In general, if a computing task is well-suited to SIMD parallelization then it will be well-suited to GPU computation. In particular, data-parallel computations where the ratio of arithmetic operations to memory operations is high are able to attain maximum performance from a GPU, as the volume of very fast arithmetic instruction can 'hide' the relatively slow memory accesses. Many standard methods, such as MCMC or sequential Monte Carlo, are not naturally suited to implementation on GPUs or other multi-core architectures. To have maximum impact, next generation statistical algorithms will need to exploit the structure and respect the constraints of multi-core architectures.

[1] Andrieu, C. and Roberts, G. O. The pseudo-marginal approach for efficient monte carlo computations. *Annals of Statistics* **37**, 697–725 (2009).

[2] Andrieu, C., Doucet, A., and Holenstein, R. Particle Markov chain Monte Carlo (with Discussion). *Journal of the Royal Statistical Society, Series B* **62**, 269–342 (2010).

[3] Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., and Feldman, M. W. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Molecular Biology and Evolution* **16**, 1791–1798 (1999).

[4] Marjoram, P., Molitor, J., Plagnol, V., and Tavare, S. Markov chain Monte Carlo without likelihoods. *PNAS* **100**, 15324–15328 (2003).

[5] Gouriéroux, C. and Ronchetti, E. Indirect inference. *Journal of Applied Econometrics* **8**, s85–s118 (1993).

[6] Wood, S. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature* **466**(7310), 1102–1104 (2010).

[7] Li, N. and Stephens, M. Modelling LD, and identifying recombination hotspots from SNP data. *Genetics* **165**, 2213–2233 (2003).

[8] Rue, H., Martino, S., and Chopin, N. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations (with discussion) . *Journal of the Royal Statistical Society, Series B* **71**, 319–392 (2009).

[9] Cox, D. R. and Reid, N. A note on pseudolikelihood constructed from marginal densities. *Biometrika* **91**(3), 729–737 (2004).

[10] Papaspiliopoulos, O. and Roberts, G. O. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika* **95**, 169–186 (2008).

[11] Beskos, A., Papaspiliopoulos, O., and Roberts, G. O. A factorisation of diffusion measure and finite sample path constructions. *Methodology and Computing in Applied Probability* **10**, 85–104 (2008).

[12] Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P. Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society Series B* **68**, 333–382 (2006).

[13] Keane, M. S. and O'Brien, G. L. A Bernoulli factory. *Computer Simulation* **4**, 213–219 (1994).

[14] Latuszynski, K., Kosmidis, I., Papaspiliopoulos, O., and Roberts, G. Simulating Events of Unknown Probabilities via Reverse Time Martingales. *Random Structures and Algorithms* **38**(4), 442–453 (2011).

[15] Andrieu, C. and Thoms, J. A tutorial on adaptive MCMC. *Statistics and Computing* **18**, 343–373 (2008).

[16] Suchard, M., Wang, Q., Chan, C., Frelinger, J., Cron, A., and West, M. Understanding gpu programming for statistical computation: Studies in massively parallel massive mixtures. *Journal of Computational and Graphical Statistics* **19**(2), 419–438 (2010).

[17] Lee, A., Yau, C., Giles, M., Doucet, A., and Holmes, C. On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods. *Journal of Computational and Graphical Statistics* **19**(4), 769–789 (2010).