Estimating the Probability that a Function Observed with Noise is Convex

Nanjing Jian, Shane G. Henderson
School of Operations Research and Information Engineering, Cornell University
nj227@cornell.edu, sgh9@cornell.edu

Consider a real-valued function that can only be observed with stochastic noise at a finite set of design points
within a Euclidean space. We wish to determine whether there exists a convex function that goes through
the true function values at the design points. We develop an asymptotically consistent Bayesian sequential
sampling procedure that estimates the posterior probability of this being true. In each iteration, the posterior
probability is estimated using Monte Carlo simulation. We offer three variance reduction methods – change
of measure, acceptance-rejection, and conditional Monte Carlo. Numerical experiments suggest that the
conditional Monte Carlo method should be preferred.

Key words: convexity detection; Bayesian sequential models; variance reduction; likelihood ratio;
    conditional Monte Carlo

1.   Introduction

Our goal in this paper is to develop a method to determine whether a real-valued function
$g : S \subseteq \mathbb{R}^d \to \mathbb{R}$, observed at a finite set of points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_r$ in S with noise from a
stochastic simulation model, is convex in the sense that a convex function f exists that
coincides with g at $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_r$. That is, does there exist a convex function f that goes
through the points $(\boldsymbol{x}_1, g(\boldsymbol{x}_1)), (\boldsymbol{x}_2, g(\boldsymbol{x}_2)), \ldots, (\boldsymbol{x}_r, g(\boldsymbol{x}_r))$? For example, g could be the
expected profit in an inventory problem where the demand ξ is random, the starting
inventory $\boldsymbol{x}$ of the day can take integer values $S = \{0, 1, \ldots, \infty\}$, and we only observe a
simulation estimate of g at $\boldsymbol{x}$. We might choose a few integer values in S to test whether the
expected profit is convex as a function of the starting inventory. Or g might represent the
(true) expected waiting time for an ambulance as a function of base locations represented
in latitude-longitude coordinates, where we observe g with noise through a stochastic

simulation model of ambulance operations. We might then choose a finite set of base-location options in the city to test whether the waiting time is convex with regard to the location of the ambulance bases.

Convexity is a key structural property that can be exploited in many ways. Assuming the minimum is attained, one can use gradient-based methods (for smooth functions) or a cutting-plane based method (for nonsmooth functions) to quickly find the minimum, or bounds on the minimum, e.g., Nesterov (2004), Glynn and Infanger (2013). Even if a function is not globally convex, one might use our methodology to identify regions around local minima in which the restriction of the objective function is convex ("basins of attraction"). Such basins can provide information on the stability of a solution (Vogel 1988).

Our methodology is computationally demanding, so we do not envisage it being used only once prior to the selection of an optimization algorithm, which is then applied to solve an optimization problem only once. Rather, we contend that it is usually the case that optimization models are repeatedly solved, often with the same model structure but with different data. In such cases, one would expect that the presence of convexity (or not) would usually be preserved from one data set to another. So then we believe it appropriate to explore convexity *once* on *one* data set using our methodology, and to use the results to inform subsequent effort on *many* instances of the optimization problem that differ in modest ways, perhaps in terms of input parameters or problem-specific data. In this view, the computational cost of exploring the function once with our methodology is amortized over the subsequent analysis.

Beyond optimization, convexity can also provide insights into qualitative model behavior in simulation applications and elsewhere. This is especially useful when we have a sequence of similar simulation models with different inputs but the same structural properties, as mentioned earlier.

Most studies on convexity detection use a frequentist hypothesis-testing framework, and can be categorized with regard to how the null hypothesis is defined. The first category uses an infinite-dimensional functional approach. It defines the null hypothesis as $g \in \mathcal{C}$, where $\mathcal{C}$ is the cone of convex functions in an appropriate function space, whereas the alternative hypothesis is $g \notin \mathcal{C}$. The representative paper Juditsky and Nemirovski (2002) models g as a Gaussian process assuming smoothness under the null hypothesis, and uses the L^r distance between g and $\mathcal{C}$ as the test statistic.

The second category is closer to our paper in that it works with finite-dimensional vectors. The null hypothesis is $\boldsymbol{g} \in \mathbb{C}$, where $\boldsymbol{g}$ is the restriction of g to a finite set of points, and $\mathbb{C}$ is the set of convex functions restricted to the same set of points. In this case, the noisy evaluations of $\boldsymbol{g}$ are modeled by a Gaussian random vector. Although not targeted specifically for testing convexity, Silvapulle and Sen (2001) describes a general approach for testing whether a finite-dimensional Gaussian vector lies within a closed convex cone. They use the distance between the Gaussian vector and the cone as the test statistic, and show that this distance follows a so-called chi-bar-square distribution, the tail probability of which can be evaluated using simulation.

The third category fits a regression model with Gaussian noise on the observations of g and tests the hypothesis of convexity through the estimated model parameters. This approach is essentially testing whether there exists a convex function that could have generated the observed finite set of function values. In one dimension and under regularity conditions, Baraud et al. (2005) show that testing for the regression parameter is equivalent to testing $\boldsymbol{g} \in \mathbb{C}$, when $\mathbb{C}$ is defined with regard to nonnegative second order Vandermonde determinants. Their test is based on the idea that if a one-dimensional function is convex, then the sample mean of the function values in a partition should be lower than certain linear combinations of the function values in neighboring partitions. Others fit cubic splines on the observations and use the second order derivatives at the knots to test for convexity, e.g. Diack and Thomas-Agnan (1998), Wang and Meyer (2011) and Meyer (2012). For higher dimensions, Abrevaya and Jiang (2005) work with small and localized sets of data points and count all the possible convex and concave simplices to construct a test statistic. Lau (1978) uses a second-order parametric model for the data points, and is a good survey of early literature.

A closely related field to convexity tests is convex regression, where one fits a regression model to observations under the constraint that the fitted model is convex. Work in this direction includes Judge and Takayama (1966), Allon et al. (2007), Seijo and Sen (2010), Hannah and Dunson (2011), Lim and Glynn (2012), among which Seijo and Sen (2010) provides a review of past work.

In this paper, we give a Bayesian sequential algorithm that iteratively collects noisy evaluations of an unknown function g on a fixed and finite set of design points $\boldsymbol{x}$, and uses them to estimate the posterior probability that the function, when restricted to the design

points, is convex. Our approach differs from previous research in two main ways. First, since function estimates are obtained via Monte Carlo simulation, we can only observe the function g on a finite set of points $\boldsymbol{x}$. Thus the best we can provide is a statistical guarantee that there exists a convex function that coincides with the unknown function g at those points. Even if there exists such a convex function, it does not imply that g itself is convex, although nonexistence does imply nonconvexity of g. Second, we use a Bayesian conjugate prior model that updates a posterior on the function values every time we collect a set of new samples. Then the posterior probability of convexity is estimated separately through Monte Carlo simulation. Instead of having a fixed running time, as is the case with hypothesis testing, our algorithm can be stopped at any stage to output an estimated probability of convexity. Indeed, the Bayesian approach avoids the difficulty in frequentist sequential hypothesis testing where one should condition on the outcomes of previous hypothesis tests when considering the distribution of a test statistic. The Bayesian framework seems to us to be more straightforward in both concept and implementation.

Our overall approach is to successively update a posterior distribution on the vector of (true) function values $\boldsymbol{g}$. In doing so we assume that the noise in the estimated function values is normally distributed and adopt a conjugate prior so that we can use standard posterior updates. In simulation the normality assumption is common and usually reasonable, since one can average multiple (finite variance) replications to obtain a single approximately normally distributed replication through the central limit theorem. Normality is not essential to our approach; other distributions could be assumed, but the use of a conjugate prior is key to keeping the computation manageable. We restrict attention to the normal assumption for brevity and simplicity, and because our primary intended application area is in simulation, where the assumption can almost always be at least approximately satisfied (through averaging).

For a given posterior distribution, computing the probability of convexity for a sample from the posterior appears to be difficult. We use Monte Carlo to estimate this probability, providing three methods for reducing the variance of the Monte Carlo estimator of the posterior probability of convexity. The change of measure and acceptance-rejection methods reuse samples obtained in an earlier iteration to construct an unbiased estimator for the current iteration. These two methods can be useful in any sequential algorithm in a Bayesian framework, but need to be used with caution due to heavy-tailed behavior of

the likelihood ratio that is needed in these methods. The conditional Monte Carlo method takes advantage of the spherical property of Gaussian and t distributions. It can be applied to the more general problem of estimating the probability that a Gaussian or t-distributed random vector lies in a polyhedron, which might arise, e.g., in solving linear feasibility problems described in Szechtman and Yücesan (2016).

We view the Bayesian methodology developed herein, where we repeatedly estimate the posterior probability of convexity as n, the number of simulation replications, increases, as being well suited to exploratory analysis aimed at developing knowledge of the structure of the function g. Certainly, a key strength of the Bayesian approach, relative to hypothesis tests, is that it allows an analyst to explore several values of n, interactively increasing n until satisfied with the results. Hypothesis tests do not afford this flexibility, so in our opinion are not as well suited to the exporatory analysis we envisage. Our statistical guarantees are weaker than those of formal hypothesis tests, in that the confidence intervals we generate on the posterior probability of convexity hold only marginally in n and not *jointly* in n (they are not confidence bands). However, plotting the confidence intervals as a function of n, as we do, provides a visual sense of whether the function g (restricted to the selected points) is convex or not, along with a sense of the sample size needed to establish convexity or not with some reliability. It is conceivable that one could further develop the Bayesian methodology to develop a more rigorous test, but we do not pursue that line, again because of our focus on an exploratory tool.

This paper is built upon Jian et al. (2014), with the addition of new results, complete proofs, new variance reduction methods, and more extensive numerical results. Full proofs of several results can be found in the online supplement.

Notation: We use upper case Latin letters for random variables or sets, and lower case Latin letters for deterministic variables. Vectors are in bold, and matrices are in upper case Greek letters. We use A^T to denote the transpose of the matrix A. For a set S, S° is its interior. We use $\Rightarrow$ for convergence in distribution, and $\mathbb{1}\{B\}$ is the indicator function that takes the value 1 on the event B and 0 otherwise.

2.   Problem Statement and Assumptions

Suppose we can obtain noisy evaluations of a real-valued function $g : S \to \mathbb{R}$ over a fixed and finite set of design points $\boldsymbol{x} = \{\boldsymbol{x}_i : i = 1, 2, \ldots, r\}$ in $S \subseteq \mathbb{R}^d$. Ideally, we would like to know

whether the function g is convex on S or not. In the absence of any regularity assumptions on g, such as are assumed in Juditsky and Nemirovski (2002), it appears that this question cannot be addressed in finite time. The rest of this paper is focused on giving a statistical guarantee on the convexity of the finite-dimensional vector obtained by restricting the function g to the r points in $\boldsymbol{x}$, i.e., we restrict attention to $\boldsymbol{g} = (g(\boldsymbol{x}_1), g(\boldsymbol{x}_2), \ldots, g(\boldsymbol{x}_r)) \in \mathbb{R}^r$.

DEFINITION 1. Given a finite set of points $\boldsymbol{x}$, we define a vector $\boldsymbol{g}$ to be convex if and only if there exists a convex function g whose values on $\boldsymbol{x}$ coincide with $\boldsymbol{g}$, i.e. $g(\boldsymbol{x}) = \boldsymbol{g}$.

DEFINITION 2. Define $\mathbb{C} = \mathbb{C}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_r) \subseteq \mathbb{R}^r$ to be the set of all convex vectors on $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_r)$, so that $\boldsymbol{g}$ is convex if and only if $\boldsymbol{g} \in \mathbb{C}$.

The notion of vector convexity is weaker than (the usual) functional convexity, since if a function g is convex then its restriction $\boldsymbol{g}$ on $\boldsymbol{x}$ is convex under our definition. The converse is not true since we can arbitrarily extend $\boldsymbol{g}$.

The set $\mathbb{C}$ is a convex cone. In Section 5, we will see that $\boldsymbol{g} \in \mathbb{C}$ if and only if a certain linear system is feasible, and the linear system is then a tool to verify vector convexity. We will also show that $\boldsymbol{g}$ is strictly convex (in the sense that a strictly convex function g exists that agrees with $\boldsymbol{g}$ on $\boldsymbol{x}$) iff $\boldsymbol{g} \in \mathbb{C}^\circ$, the interior of $\mathbb{C}$, in the online supplement.

We use a Bayesian approach, regarding $\boldsymbol{g}$ as an unknown realization from the prior distribution of a random vector $\boldsymbol{f}$. Let $(\boldsymbol{\xi}_j : j = 1, 2, \ldots)$ be an i.i.d. sequence of r-dimensional Gaussian random vectors, each with mean vector $\boldsymbol{0}$ and covariance matrix Γ, that is independent of $\boldsymbol{f}$. Our jth observation is then $\boldsymbol{Y}_j$, where $\boldsymbol{Y}_j = \boldsymbol{f} + \boldsymbol{\xi}_j$, $j = 1, 2, \ldots$. We denote the ith component of the vector $\boldsymbol{Y}_j$ by Y_{ij}, and interpret it as the jth sampled function value at the point $\boldsymbol{x}_i$.

To elaborate, we view $\boldsymbol{g}$ as a deterministic vector of function values. In the Bayesian structure within which we work, the vector $\boldsymbol{g}$ is modeled through a random vector $\boldsymbol{f}$ that is sampled from a prior distribution at the outset, and we subsequently accumulate evidence on the value of $\boldsymbol{f}$ (and thus $\boldsymbol{g}$) through the samples $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots$, updating our beliefs through the posterior distribution. In principle, with sufficient replications we can recover the value of $\boldsymbol{f}$; our progress towards this goal is reflected through the posterior distribution.

We observe the value of $\boldsymbol{f}$ with additive noise, through the vector outputs of successive simulation replications $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots$. Consistent with much simulation literature, conditional on $\boldsymbol{f}$, the observed values $\boldsymbol{Y}_j, j = 1, 2, \ldots$ are assumed to be Gaussian, which can at least

be approximately satisfied by averaging multiple simulation replications to yield a single Y_j. This is an approximation, but it affords considerable computational advantages, as we will see later.

The covariance matrix Γ is not necessarily diagonal, i.e., we do not necessarily constrain the observations to be (conditionally) independent between sampled points, conditional on f. Thus, Common Random Numbers (CRN) can be employed within our framework. CRN induces positive correlation on the r dimensions of the noise ξ, so that the structure of the underlying "true" function f can be better preserved than would be possible with conditionally independent observations (Chen et al. 2012). For simplicity, we assume throughout that the covariance matrix Γ is positive definite.

3. Sequential Algorithm

Initially, before any sampling, we fix the r points $x_i, i = 1, \ldots, r$ and the prior mean μ_0 and covariance matrix Λ_0 of the assumed Gaussian prior distribution of f. At the beginning of the nth iteration $(n = 1, 2, \ldots)$, we obtain a new observation Y_n, and use that to update the posterior distribution on the function values, as described in Section 4. The information collected thus far is denoted $\mathscr{A}_n$, which is the sigma field generated from $\mathscr{A}_0$ and $\{Y_j, j = 1, \ldots, n\}$. Thus, $\{\mathscr{A}_n\}_{n=0,1,2,\ldots}$ is a filtration. Once the posterior distribution has been updated, we separately estimate the posterior probability of convexity, $P(f \in \mathbb{C} | \mathscr{A}_n)$ as discussed in Section 5. At the end of each iteration, we can choose either to stop, or to continue with the current posterior as the prior of the next iteration. More precisely, the algorithm is as follows.

Algorithm 1 A sequential method for testing for convexity of the function

---

Require: The Gaussian prior $P(\boldsymbol{f} \in \cdot | \mathscr{A}_0)$, with hyperparameters $\{\boldsymbol{\mu_0}, \Lambda_0\}$ of the function

values $\boldsymbol{f}$.

1: Initialize $n = 0$.

2: **repeat**

3:    Set $n = n + 1$.

4:    Obtain a new vector $\boldsymbol{y}_n$ of r noisy function values at $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_r$.

5:    Update the posterior distribution of $\boldsymbol{f} | \mathscr{A}_n$ from the new samples $\boldsymbol{y}_n$ using $\boldsymbol{f} | \mathscr{A}_{n-1}$ as

   the prior, as in Section 4.

6:    Estimate $p_n = P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$ from the distribution of $\boldsymbol{f} | \mathscr{A}_n$ using the Monte Carlo

   method described in Section 5, obtaining a confidence interval $[\hat{p}_n - h_n, \hat{p}_n + h_n]$.

7: **until** stopped

8: **return** A confidence interval $[\hat{p}_n - h_n, \hat{p}_n + h_n]$ of p.

In Step 6, the posterior probability that $\boldsymbol{f}$ is convex is estimated using Monte Carlo simulation by sampling m times from the posterior distribution $\boldsymbol{f} | \mathscr{A}_n$. Compared to obtaining the samples $\boldsymbol{Y}_j$, which involves running the full simulation model, sampling from the posterior distribution is computationally inexpensive, entailing sampling from a Gaussian or t distribution, depending on whether the variance is known or unknown. Depending on the computational cost of running the full simulation model, we can also skip the estimation of p_n for some n and enter Step 6 only for selected values of n.

4.   Posterior Updates

We use a conjugate prior to update our belief about $\boldsymbol{f} | \mathscr{A}_n$ in each iteration n. Since we assumed that $\boldsymbol{Y} - \boldsymbol{f} \sim N(\boldsymbol{0}, \Gamma)$, this conjugate prior is normal-normal when Γ is known, and normal-inverse-Wishart when Γ is unknown. In this section we give the updating formula of the posterior distribution of $\boldsymbol{f} | \mathscr{A}_n$ in Step 5 of Algorithm 1 under these two scenarios, given the prior $\boldsymbol{f} | \mathscr{A}_{n-1}$. The formulae given here are standard; see, e.g., DeGroot (1970), Gelman et al. (2003), or Bernardo and Smith (2008).

4.1.   Conjugate Prior under Known Sampling Variance

First, before any sampling we select a non-informative Gaussian prior with zero mean and large variance, i.e. $\boldsymbol{f} | \mathscr{A}_0 \sim N(\boldsymbol{\mu_0}, \Lambda_0)$ in which $\mu_0 = \boldsymbol{0} \in \mathbb{R}^r$ and $\Lambda_0 \in \mathbb{R}^{r \times r}$ is a diagonal

matrix with diagonal values that are large relative to the sampling variance (the diagonal of Γ). Alternative parameters for the Gaussian prior could be used in the presence of more information, and would not change the algorithm.

At iteration $n > 1$, the posterior from the last iteration $\boldsymbol{f}|\mathscr{A}_{n-1} \sim N(\boldsymbol{\mu_{n-1}}, \Lambda_{n-1})$ is used as the prior for the current iteration. We then obtain $s \geq 1$ new objective-function samples $(y_{ij}, j = 1, 2, \ldots, s)$ at each of the design points $\boldsymbol{x}_i, i = 1, 2, \ldots, r$. The mean $\boldsymbol{\mu_n}$ and covariance Λ_n of the posterior $\boldsymbol{f}|\mathscr{A}_n \sim N((\boldsymbol{\mu_n}, \boldsymbol{\Lambda_n})$ are updated by

$$\Lambda_n^{-1} = \Lambda_{n-1}^{-1} + s\Gamma^{-1}$$

$$\boldsymbol{\mu_n} = \Lambda_n(\Lambda_{n-1}^{-1}\boldsymbol{\mu_{n-1}} + s\Gamma^{-1}\bar{\boldsymbol{y}}), \tag{1}$$

where the i-th component of the r-dimensional vector $\bar{\boldsymbol{y}}$ is $s^{-1} \sum_{j=1}^{s} \boldsymbol{y}_{ij}$. One can adaptively choose the sample size s in each iteration, but for simplicity we use $s = 1$, meaning that only one new sample is obtained in each iteration.

The updating equation (1) involves matrix inversion. To reduce the computational effort, we use Cholesky factorization and the Sherman-Morrison-Woodbury formula as detailed in the online supplement.

4.2. Conjugate Prior under Unknown Sampling Variance

When the sampling variance Γ is unknown, the inverse-Wishart distribution provides a conjugate prior. First, we use an uninformative Jeffrey's prior, where the prior joint distribution on $\boldsymbol{f}$ and Γ is proportional to $|\Gamma|^{-(r+1)/2}$ (Gelman et al. 2003) and $|A|$ denotes the determinant of the matrix A. To construct Jeffrey's prior, an initial set of r-dimensional samples $\boldsymbol{y}_j = (y_{ij}, i = 1, 2, \ldots, s), j = 1, 2, \ldots, s_0$ are used to estimate the parameters of the normal distribution for the mean $\boldsymbol{f}$ and the Inverse-Wishart distribution (Inv-Wishart) for the variance Γ. The initial sample size s_0 can be any positive integer. For a prior that reflects the data without being too costly, we choose $s_0 = r + 1$, where r is the number of design points. This choice also ensures that the inverse-Wishart distribution is concentrated on covariance matrices that are positive definite. More specifically (Gelman et al. 2003),

$$\Gamma|\mathscr{A}_0, \boldsymbol{y} \sim \text{Inv-Wishart}_{v_0}(\Xi_0^{-1})$$

$$\boldsymbol{f}|\Gamma, \mathscr{A}_0, \boldsymbol{y} \sim N(\mu_0, \Gamma/\kappa_0), \tag{2}$$

where

$$\boldsymbol{\mu_0} = \frac{1}{s_0}\sum_{j=1}^{s_0}\boldsymbol{y}_j = \bar{\boldsymbol{y}}; \quad \kappa_0 = s_0; \quad \upsilon_0 = s_0 - 1; \quad \Xi_0 = \left(\sum_{j=1}^{s_0}(\boldsymbol{y}_j - \bar{\boldsymbol{y}})(\boldsymbol{y}_j - \bar{\boldsymbol{y}})^T\right)^{-1}.$$

In iteration $n \geq 1$, we obtain s samples y_{ij} on each of the points $\boldsymbol{x}_i, i = 1, \ldots, r$ and update the posterior of

$$\Gamma|\mathscr{A}_n \sim \text{Inv-Wishart}_{\upsilon_n}(\Xi_n^{-1})$$

$$\boldsymbol{f}|\Gamma, \mathscr{A}_n \sim N(\boldsymbol{\mu_n}, \Gamma/\kappa_n) \tag{3}$$

by (Gelman et al. 2003):

$$\boldsymbol{\mu_n} = \frac{\kappa_{n-1}}{\kappa_{n-1} + s}\boldsymbol{\mu_{n-1}} + \frac{s}{\kappa_{n-1} + s}\bar{\boldsymbol{y}}; \quad \kappa_n = \kappa_{n-1} + s; \quad \upsilon_n = \upsilon_{n-1} + s;$$

$$\Xi_n = \Xi_{n-1} + S + \frac{\kappa_{n-1}s}{\kappa_{n-1} + s}(\bar{\boldsymbol{y}} - \boldsymbol{\mu_{n-1}})(\bar{\boldsymbol{y}} - \boldsymbol{\mu_{n-1}})^T, \tag{4}$$

where the i-th component of the r dimensional vector $\bar{\boldsymbol{y}}$ is defined as $\bar{\boldsymbol{y}}_i = \sum_{j=1}^s y_{ij}/s, i = 1, \ldots, r$, and the $r \times r$ matrix S is the sum of squared errors $\sum_{j=1}^s (\boldsymbol{y}_j - \bar{\boldsymbol{y}})(\boldsymbol{y}_j - \bar{\boldsymbol{y}})^T$. For simplicity we again choose $s = 1$ when updating, so that $\boldsymbol{y}_j = \bar{\boldsymbol{y}}$ and $S = 0$.

If a random $r \times r$ matrix Γ has the Inverse-Wishart distribution with parameters υ and Ξ^{-1}, whose density is proportional to $|\Xi|^{\upsilon/2}|\Gamma|^{-(\upsilon-r-1)/2}\exp\{-\text{tr}(\Xi\Gamma^{-1})/2\}$, where $\text{tr}(\cdot)$ is the trace of a matrix, the inverse Γ^{-1} has the Wishart distribution with parameters υ and Ξ. The Wishart distribution is a higher-dimensional generalization of the χ^2 distribution, and thus can be expressed similarly as the sum of squares of Gaussian random vectors. To generate a Wishart$_\upsilon(\Xi)$ distributed random matrix Γ, we generate $\upsilon > r$ independent, r-dimensional random vectors W_i distributed as $N(0, \Xi)$, and return $\Gamma = \sum_{i=1}^\upsilon W_i W_i^T$.

With the posterior covariance distributed as Inverse-Wishart and the posterior mean distributed as Gaussian conditioning on the covariance, the marginal distribution of the posterior mean $\boldsymbol{f}|\mathscr{A}_n$ is

$$\boldsymbol{f}|\mathscr{A}_n \sim t_{(\upsilon_n - r + 1)}(\boldsymbol{\mu_n}, \Xi_n/(\kappa_n(\upsilon_n - r + 1))), \tag{5}$$

where $t_{(\upsilon_n - r + 1)}(\boldsymbol{\mu_n}, \Xi_n/(\kappa_n(\upsilon_n - r + 1)))$ is a multivariate Student-t distribution with $(\upsilon_n - r + 1)$ degrees of freedom, location parameter $\boldsymbol{\mu_n}$, and scale matrix $\Xi_n/(\kappa_n(\upsilon_n - r + 1))$. The density function of $\boldsymbol{f}|\mathscr{A}_n$ is thus proportional to $|\Xi_n/(\kappa_n(\upsilon_n - r + 1))|^{-1/2}\{1 + (\boldsymbol{f} - \boldsymbol{\mu_n})^T[\Xi_n/(\kappa_n(\upsilon_n - r + 1))]^{-1}(\boldsymbol{f} - \boldsymbol{\mu_n})\}^{-(\upsilon_n+1)/2}$ (Gelman et al. 2003). One can generate such a random vector by exploiting the elliptical nature of the distribution; see Section 7.3. (Although that section does not explicitly give a generation algorithm, an algorithm should be clear from the arguments given there.)

5.   Convexity

Recall that $\boldsymbol{g} = (g(\boldsymbol{x}_1), g(\boldsymbol{x}_2), \ldots, g(\boldsymbol{x}_r))$, and the vector $\boldsymbol{g}$ is defined to be convex if and only if there exists a convex function that coincides with $\boldsymbol{g}$ on the set of points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r$. Equivalently, for each $i = 1, \ldots, r$ there exists a hyperplane $\{\boldsymbol{a}_i^T \boldsymbol{x} + b_i : \boldsymbol{x} \in \mathbb{R}^d\}$ that goes through $(\boldsymbol{x}_i, g(\boldsymbol{x}_i))$ and lies at or below all the other points $(\boldsymbol{x}_j, g(\boldsymbol{x}_j)), j \neq i$ (Murty 1988, p.539; Atlason et al. 2004). That is, $\boldsymbol{g}$ is convex if and only if there exists feasible solutions $\boldsymbol{a}_i \in \mathbb{R}^d, i = 1, \ldots, r$ and $\boldsymbol{b} \in \mathbb{R}^r$ to the linear system

$$
\begin{aligned}
\boldsymbol{a}_i^T \boldsymbol{x}_i + b_i &= g(\boldsymbol{x}_i), \quad \text{for all } i \in \{1, \ldots, r\} \\
\boldsymbol{a}_i^T \boldsymbol{x}_j + b_i &\leq g(\boldsymbol{x}_j), \quad \text{for all } i \in \{1, \ldots, r\} \text{ and } j \neq i, \, j \in \{1, \ldots, r\},
\end{aligned}
\tag{LS}
$$

with b_i being the i-th component of $\boldsymbol{b}$. Let the set of all $\boldsymbol{g}$ such that the corresponding LS is feasible be $\mathbb{C}$, which denotes the set of all convex vectors $\boldsymbol{g}$ with regard to the r design points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r$.

This large linear system can also be decomposed into r sub-systems, indexed by $i = 1, \ldots, r$:

$$
\begin{aligned}
\boldsymbol{a}_i^T \boldsymbol{x}_i + b_i &= g(\boldsymbol{x}_i) \\
\boldsymbol{a}_i^T \boldsymbol{x}_j + b_i &\leq g(\boldsymbol{x}_j), \quad \text{for all } j \neq i \text{ and } j \in \{1, \ldots, r\},
\end{aligned}
\tag{LS(i)}
$$

each with the variables $\boldsymbol{a}_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}$.

Transforming the question of whether a vector is convex to the feasibility of r linear systems allows us to use Monte Carlo simulation to estimate the posterior probability of convexity at the end of each iteration n. We first simulate m random samples from the posterior distribution of $\boldsymbol{f}|\mathscr{A}_n$. Then, for each generated sample, we determine feasibility (or lack thereof) for the linear systems (LS(i), $i = 1, \ldots, r$) in sequence. If *any* linear system is infeasible then we stop (skip the rest of the systems) and conclude that this generated sample is not convex, since one cannot define an appropriate hyperplane. The probability $P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$ is then estimated by the sample average of the indicators of convexity for each sample as described more formally in Algorithm 2.

6.   Asymptotic Validity of the Main Algorithm

We now establish that the posterior probability of convexity converges to 1 or 0, depending on whether $\boldsymbol{g}$ is convex or not, with one qualification. If $\boldsymbol{g}$ is convex but not strictly convex then it lies on the boundary of $\mathbb{C}$, and then certain arbitrarily small perturbations of the

---

Algorithm 2 Subroutine used in Step 6 of Algorithm 1 to estimate $P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$.

Require: The posterior marginal density of $\boldsymbol{f} | \mathscr{A}_n$ from (1) or (5).

1: Generate independent samples $\{\boldsymbol{y}_n^1, \boldsymbol{y}_n^2, \ldots, \boldsymbol{y}_n^m\}$ from the posterior marginal density of $\boldsymbol{f} | \mathscr{A}_n$.

2: **for** k from 1 to m **do**

3:     Set $\mathbb{1}\left\{\boldsymbol{y}_n^k \in \mathbb{C}\right\} = 1$.

4:     **for** i from 1 to r **do**

5:        Set $g(\boldsymbol{x}_i)$ as the i-th component of $\boldsymbol{y}_n^k$, $i = 1, \ldots, r$.

6:        Solve for the feasibility of LS(i).

7:        **if** LS(i) is infeasible **then**

8:           Set $\mathbb{1}\left\{\boldsymbol{y}_n^k \in \mathbb{C}\right\} = 0$.

9:           BREAK the inner loop and go to next k.

10:        **end if**

11:     **end for**

12: **end for**

13: **return**   The center $\hat{p}_n = \sum_{k=1}^{m} \mathbb{1}\left\{\boldsymbol{y}_n^k \in \mathbb{C}\right\} / m$ and half-width $h_n = 1.96 s_n / \sqrt{m}$ of a 95% confidence interval for $P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$, where s_n is the sample standard deviation of $\mathbb{1}\left\{\boldsymbol{y}_n^k \in \mathbb{C}\right\}$, $k = 1, \ldots, m$.

---

function values $\boldsymbol{g}$ will yield points outside $\mathbb{C}$. Since we estimate the function values $\boldsymbol{g}$ using simulation, we cannot rule out such perturbations, and so we should not expect the posterior probability of convexity to converge to 1 or 0.

The formal statement of convergence is with respect to the probability space containing both the prior from which $\boldsymbol{f}$ is sampled, and the data. We show that when $\boldsymbol{f}$ is strictly convex the posterior probability of convexity converges to 1, and when $\boldsymbol{f}$ is not convex the posterior probability of convexity converges to 0. The remaining case where $\boldsymbol{f}$ lies on the boundary of $\mathbb{C}$ has probability 0 under our prior, which has a density with respect to Lebesgue measure.

THEOREM 1. *Let $p_n = P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$ be the n-iteration posterior probability that $\boldsymbol{f}$ is convex as in Algorithm 1. As the number of iterations $n \to \infty$, $p_n - \mathbb{1}\{\boldsymbol{f} \in \mathbb{C}\} \to 0$ a.s.*

Jian et al. (2014) has a sketch of the proof in the known variance case. We provide a complete proof that covers both the known and unknown variance cases in the online supplement.

The result of Theorem 1 relates to the exact posterior probability of convexity, which we estimate using Monte Carlo. We next show that the Monte Carlo estimator from Section 5 of the exact probability converges to the same indicator provided that the Monte Carlo sample sizes increase without bound, through a uniform law of large numbers.

COROLLARY 1. *Let p_n^m be the m-sample estimator of $P(\boldsymbol{f} \in \mathbb{C}|\mathscr{A}_n)$ from Algorithm 2. As $n \to \infty$ and $m = m(n) \to \infty$, $p_n^m - \mathbb{1}\{\boldsymbol{f} \in \mathbb{C}\} \to 0$ in probability.*

Proof. We have $|p_n^m - \mathbb{1}\{\boldsymbol{f} \in \mathbb{C}\}| \le |p_n^m - p_n| + |p_n - \mathbb{1}\{\boldsymbol{f} \in \mathbb{C}\}|$, where $p_n = P(\boldsymbol{f} \in \mathbb{C}|\mathscr{A}_n)$ and $p_n^m = \frac{1}{m}\sum_{k=1}^m \mathbb{1}\{\boldsymbol{y}_n^k \in \mathbb{C}\}$. Let $\epsilon > 0$ be arbitrary. For the first term, Chebyshev's inequality gives

$$P(|p_n^m - p_n| > \epsilon) = EP\left(\left|\frac{1}{m}\sum_{k=1}^m \mathbb{1}\{\boldsymbol{y}_n^k \in \mathbb{C}\} - P(\boldsymbol{f} \in \mathbb{C}|\mathscr{A}_n)\right| > \epsilon \middle| \mathscr{A}_n\right)$$
$$\le E\left(\frac{Var(\mathbb{1}\{\boldsymbol{y}_n^k \in \mathbb{C}\}|\mathscr{A}_n)}{m\epsilon^2}\right) \le \frac{1}{4m\epsilon^2} \to 0$$

as $n \to \infty$ since $m = m(n) \to \infty$ as $n \to \infty$. This shows that $p_n^m - p_n \to 0$ in probability as $n \to \infty$. Also Theorem 1 shows that $|p_n - \mathbb{1}\{\boldsymbol{f} \in \mathbb{C}\}| \to 0$ in probability as $n \to \infty$.

7. Variance Reduction Methods

In this section, we improve the vanilla Monte Carlo method through three variance-reduction methods. The change of measure and acceptance-rejection methods are likelihood-ratio-based methods that reuse samples generated in an earlier iteration, and the conditional Monte Carlo method reduces the variance through smoothing.

7.1. Change of Measure

Algorithm 2 can be computationally costly due to the need to solve up to mr linear feasibility problems $LS(i)$, where m is the number of Monte Carlo samples and r is the number of design points. We can reduce the computational effort by reusing samples generated in a previous iteration through a change-of-measure method. The resulting estimator is based on the same principle used in the score-function method for simulation optimization (Rubinstein and Shapiro 1990), and that used in "green simulation" (Feng and Staum

2015). We will see that the resulting estimator is unbiased and has finite variance, but does not perform as well as we might hope.

Recall that in iteration n, Algorithm 2 generates m i.i.d. samples $\{\boldsymbol{y}_n^k : k = 1, 2, \ldots, m\}$ from the posterior marginal distribution of $\boldsymbol{f}|\mathscr{A}_n$ and produces m indicators $\{\mathbb{1}\{\boldsymbol{y}_n^k \in \mathbb{C}\} : k = 1, 2, \ldots, m\}$ of convexity. To reuse these samples, in iteration $n + \ell$, we instead output

$$\hat{p}_{n+\ell} = \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}\{\boldsymbol{y}_n^k \in \mathbb{C}\} L_{n+\ell,n}(\boldsymbol{y}_n^k) \tag{6}$$

as an estimate of $p_{n+\ell} = P(\boldsymbol{f} \in \mathbb{C}|\mathscr{A}_{n+\ell})$, where $L_{n+\ell,n}(\cdot) = \phi_{n+\ell}(\cdot)/\phi_n(\cdot)$ is the likelihood ratio of the densities of $\boldsymbol{f}|\mathscr{A}_{n+\ell}$ and $\boldsymbol{f}|\mathscr{A}_n$.

THEOREM 2. *The change of measure estimator $\hat{p}_{n+\ell} = \mathbb{1}\{\boldsymbol{Y}_n \in \mathbb{C}\}L_{n+\ell,n}(\boldsymbol{Y}_n)$ is (conditionally) unbiased and has finite conditional variance, conditional on $\mathscr{A}_{n+\ell}$ for any n and $\ell \geq 1$.*

The proof for the known Γ case can be found in Jian et al. (2014), and we provide a proof for the unknown Γ case in the online supplement.

Given that the change of measure estimator is unbiased and has finite variance, it is tempting to generate a single sample and re-use it for many iterations to save computational effort. Unfortunately, such an estimator has poor empirical performance. Figure 3 gives an example where the estimated probability of convexity is greater than 1. This happens especially later in the run when all of the linear systems are feasible, and the likelihood ratios $L_{n+\ell,n}(\boldsymbol{y}_k^n)$ occasionally take very large values.

Occasional large values of the likelihood ratio $L_{n+\ell,n}(\boldsymbol{y})$ might arise when the sample $\boldsymbol{y}$ is generated within the tail of ϕ_n. Indeed, Proposition 1 below shows that in at least one special case, $L_{n+\ell,n}$ has a heavy tail given any sampling trajectory $\mathscr{A}_n$. At first sight, this may appear to contradict Theorem 2, which states that given the posterior information $\mathscr{A}_{n+\ell}$ in iteration $n + \ell$, the change of measure estimator is bounded. But notice that in Theorem 2 we are conditioning on more information than in Proposition 1. In effect, Proposition 1 shows that given the posterior information $\mathscr{A}_n$ in iteration n, the change of measure estimator in iteration $n+\ell$ could have poor behavior, depending on the (random) samples that are used to update $\boldsymbol{f}|\mathscr{A}_{n+\ell}$ from $\boldsymbol{f}|\mathscr{A}_n$. Thus there is no contradiction between these two results. Proposition 1 shows that the change of measure estimator could exhibit volatile behavior when extreme values of the likelihood ratio arise at values that were

sampled in iteration n. However, we make no claim about how likely such values are to arise. Numerical experiments given later show that indeed the change of measure estimator is volatile.

PROPOSITION 1. *When Γ is known and $r = 1$, given $\mathscr{A}_n$, $C_n = \sup_{\boldsymbol{y} \in \mathbb{R}^r} L_{n+1,n}(\boldsymbol{y})$ asymptotically (as $n \to \infty$) has the same distribution as $e^{\chi_1^2}$, where χ_1^2 is a non-central chi-square random variable with 1 degree of freedom.*

In fact, the proof for Proposition 1 in the online supplement also applies when n is finite. In that case $\ln C_n$, conditional on $\mathscr{A}_n$, is a non-central chi-square random variable scaled by a constant of order $O(1/n)$ and shifted by another constant of order $O(1/n)$. The conclusion of Proposition 1 can be generalized to $r > 1$ when Γ is known and diagonal. Indeed, when Γ is diagonal the likelihood ratio decomposes into a product, so that $\ln C_n = \ln(\sup_{\boldsymbol{y} \in \mathbb{R}^r} L_{n+1,n}(\boldsymbol{y})) = \sup_{\boldsymbol{y} \in \mathbb{R}^r} \ln(\prod_{i=1}^r L_{n+1,n}(\boldsymbol{y}_i)) = \sum_{i=1}^r \sup_{\boldsymbol{y}_i \in \mathbb{R}} \ln L_{n+1,n}(\boldsymbol{y}_i)$. Proposition 1 then allows us to conclude that, conditional on $\mathscr{A}_n$, this is asymptotically conditionally distributed as $\chi_r^2/2$, where χ_r^2 is a non-central chi-square random variable with r degress of freedom. Since the tail probability of $\chi_r^2/2$ at a given point increases in r, we expect this heavy tail behavior to be more significant as r increases, i.e., as the number of design points increases. We conjecture that the likelihood ratio is similarly heavy-tailed in the cases where Γ is known but not necessarily diagonal and when Γ is unknown.

In summary, conditional on $\mathscr{A}_{n+\ell}$, the estimator $\hat{p}_{n+\ell}$ is unbiased and has finite variance, but its distribution may be heavy tailed given $\mathscr{A}_n$ only, depending on the samples obtained to update to $\boldsymbol{f}|\mathscr{A}_{n+\ell}$. Thus this estimator needs to be used with caution. We suggest that if the method is to be used, then one should do so with small ℓ, e.g., $\ell < 5$, based on simulation experiments described later.

7.2. Acceptance/Rejection

The change of measure estimator reuses all the samples obtained in an earlier iteration by outputting a Monte Carlo estimator that scales each indicator $\{I_n^k = \mathbb{1}\{\boldsymbol{Y}_n^k \in \mathbb{C}\} : k = 1, 2, \ldots, m\}$ by a likelihood ratio $L_{n+\ell,n}(\boldsymbol{Y}_n^k) = \phi_{n+1}(\boldsymbol{Y}_n^k)/\phi_n(\boldsymbol{Y}_n^k)$, where ϕ_n is the posterior density. An alternative is to reuse a subset of the samples from the previous iteration through acceptance-rejection.

Suppose that in iteration n, we have m i.i.d. Monte Carlo samples $\{\boldsymbol{y}_n^k : k = 1, 2, \ldots, m\}$ from $\boldsymbol{f}|\mathscr{A}_n$, together with the indicators $\{I_n^k = \mathbb{1}\{\boldsymbol{y}_n^k \in \mathbb{C}\} : k = 1, 2, \ldots, m\}$. Then, at iteration $n+1$, the k-th sample $\boldsymbol{y}_k^n$ will be accepted (reused) with probability $L_{n+1,n}(\boldsymbol{y}_k^n)/c$,

where $c \geq \sup\{L_{n+1,n}(\boldsymbol{y}) : \boldsymbol{y} \in \mathbb{R}^r\}$. If the accepted indices are $A \subseteq \{1, 2, \ldots, m\}$, then $m - |A|$ additional samples can be generated from $\boldsymbol{f}|\mathscr{A}_{n+1}$ to ensure a total of m samples. The estimator is then just the usual Monte Carlo estimator based on all m samples, i.e., $\hat{p}_{n+1} = \left(\sum_{k \in A} \mathbb{1}\left\{\boldsymbol{y}_k^n \in \mathbb{C}\right\} + \sum_{k=1}^{m-|A|} \mathbb{1}\left\{\boldsymbol{y}_k^{n+1} \in \mathbb{C}\right\} \right)/m$.

When the sampling variance Γ is known, optimization shows that c is given by

$$\left\{ \frac{|\Lambda_n|}{|\Lambda_{n+1}|} \exp\left[(\Lambda_{n+1}^{-1}\boldsymbol{\mu_{n+1}} - \Lambda_n^{-1}\boldsymbol{\mu_n})^T \Gamma (\Lambda_{n+1}^{-1}\boldsymbol{\mu_{n+1}} - \Lambda_n^{-1}\boldsymbol{\mu_n}) + \boldsymbol{\mu_n}^T \Lambda_n^{-1} \boldsymbol{\mu_n} - \boldsymbol{\mu_{n+1}}^T \Lambda_{n+1}^{-1} \boldsymbol{\mu_{n+1}} \right] \right\}^{1/2},$$

where the parameters $\boldsymbol{\mu_n}, \Lambda_n, \boldsymbol{\mu_{n+1}}, \Lambda_{n+1}$ are defined as in Section 3. When Γ is unknown, c is the maximum of a ratio of polynomials and does not have a closed form, so we calculate it numerically.

The acceptance-rejection estimator is simply an average of i.i.d. samples, like the pure Monte Carlo estimator. The difference lies in how the samples are obtained. The probability of accepting a sample generated in iteration n is $1/c$, so the efficiency of this method is related to the constant c. According to Proposition 1, the likelihood $L_{n+1,n}(y)$ can take very large values, meaning that c can often be large. When c is large, very few of the earlier samples might be reused, so the majority of the m samples needed in the $(n+1)$th iteration are new. This may lower the efficiency of the acceptance-rejection method.

7.3.   Conditional Monte Carlo

Denote the upper hemisphere of the $(r-1)$ spherical shell, $\{z \in \mathbb{R}^r : \|z\| = 1, z_r \geq 0\}$ by S_+^{r-1}. We can view a sample from the posterior distribution as consisting of both a direction Z chosen from S_+^{r-1} and a step size T taking both positive and negative values along that direction, along with the linear transformation to obtain the appropriate scale matrix and then a translation by the mean. We condition on the direction Z, and integrate the posterior over the interval of step sizes $[t_{\min}, t_{\max}]$ that yield points inside the convexity cone $\mathbb{C}$. Averaging the results over a number of uniformly generated directions gives the desired estimator.

For convenience, let $E_n(\cdot) = E(\cdot|\mathscr{A}_n)$ and $P_n(\cdot) = P(\cdot|\mathscr{A}_n)$. We write $X = TZ$, so that in the known variance case, $X \sim N(0, I)$, and in the unknown variance case $X \sim t_{\nu_n}(0, I)$. Then

$$P(\boldsymbol{f} \in \mathbb{C}|\mathscr{A}_n) = E_n\left(\mathbb{1}\left\{ \Lambda_n^{1/2} X + \boldsymbol{\mu_n} \in \mathbb{C} \right\} \right)$$
$$= E_n\left(\mathbb{1}\left\{ T\Lambda_n^{1/2} Z + \boldsymbol{\mu_n} \in \mathbb{C} \right\} \right), \text{ for } Z \text{ uniform on } S_+^{r-1}$$

$$= E_n \left(E_n \left(\mathbb{1} \left\{ T \Lambda_n^{1/2} Z + \boldsymbol{\mu_n} \in \mathbb{C} \right\} | Z \right) \right)$$

$$= E_n (P_n (T \in [t_{\min}(Z), t_{\max}(Z)] \, | Z))$$

$$= E_n (F_{T|Z}(t_{\max}(Z)) - F_{T|Z}(t_{\min}(Z))).$$

Here $F_{T|Z}$ is the conditional distribution function of T given $\mathscr{A}_n$ and Z. (We shall see that T is independent of Z.) Thus, the posterior probability $P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$ can be estimated using $F_{T|Z}$ and a way to calculate $t_{\max}(Z)$ and $t_{\min}(Z)$. Theorem 3 gives the former, and linear programs LS(i) (below) give the latter.

THEOREM 3 (**Distribution of** $T|Z$). *When the sampling variance* Γ *is known,* $F_{T|Z}(t) = (1 + sign(t) F_{\chi_r^2}(t^2))/2$, *where* $F_{\chi_r^2}(\cdot)$ *is the (cumulative) distribution function of a* χ^2 *r.v. with* r *degrees of freedom. When* Γ *is unknown,* $F_{T|Z}(t) = (1 + sign(t) F_{F(r,\nu_n)}(t^2/r))/2$, *where* $F_{F(r,\nu_n)}$ *is the distribution function of the F distribution with* r *and* ν_n *degrees of freedom.*

Proof Sketch. A detailed proof of Theorem 3 based on the "change of variables" technique is provided in the online supplement. Here we give a short proof that provides richer insight into the result, but relies on a step that is essentially a consequence of the change of variables argument.

$$F_{T|Z}(t) = P_n(T \le t | Z) = \begin{cases} P_n(T \le 0 | Z) + P_n(0 \le T \le t | Z), & \text{when } t \ge 0 \\ P_n(T \le 0 | Z) - P_n(0 \le T \le -t | Z), & \text{when } t < 0 \end{cases}$$

$$= 1/2 + \text{sign}(t) P(||X||^2 \le t^2 | Z)/2, \text{ for } X = TZ \tag{7}$$

$$= 1/2 + \text{sign}(t) P(||X||^2 \le t^2)/2.$$

The last step in (7) depends on the independence of $||X||^2$ and Z, as established in the proof in the online supplement. Intuitively, this result is a consequence of the structure of elliptical distributions, as discussed in, e.g., Joe (2014), in that such random vectors can be generated by independently generating the direction X, scaling by a square root of the scale matrix, selecting the distance T along the scaled direction independently of X, and finally adding on the mean.

When Γ is known, $X \sim N(0, I)$, so $||X||^2 \sim \chi_r^2$.

When Γ is unknown, $X \sim t_{\nu_n}(0, I) = N/\sqrt{Y/\nu_n}$ for independent $N \sim N(0, I)$ and $Y \sim \chi_{\nu_n}^2$. Therefore

$$||X||^2 = \frac{N^T N}{Y/\nu_n}$$

where $N^T N \sim \chi_r^2$, so $||X||^2/r \sim F(r, \nu_n)$. $\square$

To find $t_{\min}(Z)$ and $t_{\max}(Z)$, we can solve linear programs with objectives minimizing or maximizing t, with decision variables $t \in \mathbb{R}, \boldsymbol{a} \in \mathbb{R}^{r \times d}, \boldsymbol{b} \in \mathbb{R}^d$, and the constraints (LS), replacing $g(\boldsymbol{x})$ by $\boldsymbol{\mu} + (\Lambda^{1/2} Z)t$:

$$t_{\min} = \min t \quad (t_{\max} = \max t)$$

$$\text{s.t.} \quad \boldsymbol{a}^T \boldsymbol{x} + \boldsymbol{b} = \boldsymbol{\mu} + (\Lambda^{1/2} Z)t \tag{LP}$$

$$\boldsymbol{a}_i^T \boldsymbol{x}_j + \boldsymbol{b}_i \leq \boldsymbol{\mu_j} + (\Lambda^{1/2} Z)_j t, \text{ for all } i \in \{1, \ldots, r\} \text{ and } j \neq i, j \in \{1, \ldots, r\}$$

The linear program LP can be decomposed into r smaller LP's, with constraints LS(i) and variables $t \in \mathbb{R}, \boldsymbol{a}_i \in \mathbb{R}^r, b_i \in \mathbb{R}$:

$$t_{\min}(i) = \min t \quad (t_{\max}(i) = \max t)$$

$$\text{s.t.} \quad \boldsymbol{a}_i^T \boldsymbol{x}_i + b_i = \boldsymbol{\mu} + (\Lambda^{1/2} Z)t \tag{LP(i)}$$

$$\boldsymbol{a}_i^T \boldsymbol{x}_j + b_i \leq \boldsymbol{\mu_j} + (\Lambda^{1/2} Z)_j t, \text{ for all } j \neq i, j \in \{1, \ldots, r\},$$

and then

$$t_{\min} = \max_{i=1,2,\ldots,r} t_{\min}(i), \quad \text{and} \quad t_{\max} = \min_{i=1,2,\ldots,r} t_{\max}(i).$$

This decomposition does not bring as much speed improvement as LS(i) does, because all the decomposed linear programs must be solved.

Now we have all the pieces needed for the conditional Monte Carlo method.

Algorithm 3 A conditional Monte Carlo estimator $\widetilde{p}_n$ for $p_n = P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$.

Require: Posterior distribution of $\boldsymbol{f} | \mathscr{A}_n$ obtained from Algorithm 1 with mean $\boldsymbol{\mu_n}$ and covariance Λ_n; Number of Monte Carlo samples m needed

1: **for** $k = 1, \ldots, m$ **do**

2:    Uniformly generate a vector z_k on the surface of a unit sphere (by generating a standard Gaussian and normalizing it to a unit vector).

3:    Determine integration boundaries $t_{\min}(z_k)$ and $t_{\max}(z_k)$.

4:    Set $\widetilde{P}_n(k) = F_{T|Z}(t_{\max}(z_k)) - F_{T|Z}(t_{\min}(z_k))$.

5: **end for**

6: Calculate the mean p_n^m and standard deviation s_n^m of $(\widetilde{P}_n(k) : k = 1, 2, \ldots, m)$.

7: **return** $\widetilde{p}_n = p_n^m$ as an estimator of $P(\boldsymbol{f} \in \mathbb{C} | \mathscr{A}_n)$, along with the half-width $\tilde{h}_n = 1.96 s_n^m / \sqrt{m}$ of a 95% confidence interval.

Relative to the other variance-reduction methods, conditional Monte Carlo takes much longer to produce an estimate in each iteration because it needs to solve two linear programs LP and cannot "skip" any of them as can be done when solving the decomposed feasibility problems LS(i).

8.   Numerical Results

In this section we show numerical results on some test functions, assuming the more realistic case that the sampling variance is unknown. To select the r sample points for a test function in d dimensions, we first sample $d+1$ points uniformly at random within the (assumed compact) sample space S, and for each such random point, we generate a uniform random direction on the surface of the unit sphere. Each point-direction pair defines a line segment within S. Then we sample 3 points uniformly at random on each line segment. This method generates $r = 3(d+1)$ sample points. We select the points to lie on line segments because doing so seems to improve the performance of the convexity test relative to just sampling points uniformly within S. We have used $3(d+1)$ sample points partly to keep the computation minimal, thereby enabling us to relatively easily explore the behavior of the algorithms and estimators in this section. In practice, one would likely use more points, though it is unclear exactly how many points should be chosen. The number of points is likely related to how certain one wishes to be about convexity or lack thereof. In each iteration of the sequential algorithm, we use $m = 100$ Monte Carlo samples from the posterior predictive distribution to estimate a 95% confidence interval for p_n. With each of our estimators, one can easily adjust the sample size m to achieve a desired accuracy in the confidence interval widths of the estimators of p_n; using $m = 100$ gave reasonable results in our experiments. We discuss the choice of n in Section 9.

Our procedure is implemented in Matlab and freely available in an online repository (Jian 2017). The repository contains two versions. The first version uses only a standard Matlab installation, solving linear programs using the built-in `linprog` function (Mathworks 2016). The second version requires the installation of the packages `CVX` (Grant and Boyd 2014, 2008), which is a package for specifying and solving convex programs, and `Gurobi` (Gurobi Optimization 2016), a commercial optimization solver. We suggest the second version if a user has the requisite licenses, since `Gurobi` seems more robust than `linprog`. For example, we have found cases where `linprog` was not able to find a feasible

solution, whereas `Gurobi` did. However, because of the overhead of `CVX` in setting up the linear program in a format that `Gurobi` is able to read, `linprog` is usually faster when the problem dimension is low. When the problem dimension is high, the inefficiency of the interior-point-method used by `linprog` outweighs the overhead of `CVX`. Figure 1 compares solving times in seconds by these two solvers for the linear programs (LP), tested with the sample function $f(\boldsymbol{x}) = ||\boldsymbol{x}||^2, \boldsymbol{x} \in [-1,1]^d$ for different values of the problem dimension d.
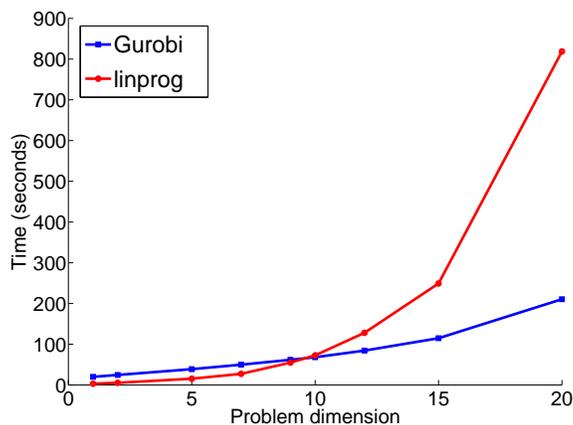
Figure 1   **The solving time vs. testing function dimension for the two linear programs in the conditional Monte Carlo method using `Gurobi` and `linprog`. `Gurobi` is faster when the dimension d exceeds 10, where $r = 33$ sample points are used.**

All test cases are run on a desktop with a 4-core Intel Core i7-3770 3.40 GHz processor with 16G memory, running Matlab R2013a on 64-bit Windows 7.

8.1. A Strictly Convex Function

We use $f(\boldsymbol{x}) = ||\boldsymbol{x}||^2$ in this section as the test function.

First, we compare vanilla Monte Carlo with the variance reduction methods in Section 7, showing 95% confidence intervals for the estimated probability of convexity, the time per iteration, and the efficiency per iteration. Here the efficiency of the Monte Carlo estimator $\tilde{p}_n$ is defined as the inverse of the product of the computational time per replication and the variance of one replication; see, e.g., Glynn and Whitt (1992). We first take the dimension $d = 1$, on the sample space $[-1,1]$. The sampling covariance matrix has equal constant variances of 0.01 on the diagonal, and we use a Gaussian kernel of $10^{-4} \exp\{-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2/2\}$ for the off-diagonal components. Hence the noise at different points is positively correlated, and the correlation is stronger between closer points (Rasmussen and Williams 2005).

We use `linprog` instead of `Gurobi` to avoid the time overhead incurred by `CVX`, since the dimension $d = 1$. For the change of measure method, a new set of samples is obtained every iteration for the first 30 iterations, and every 5 iterations thereafter. For the acceptance-rejection method we start to reuse samples only after the first 30 iterations. Thus the first 30 iterations of these two methods are exactly the same as vanilla Monte Carlo.
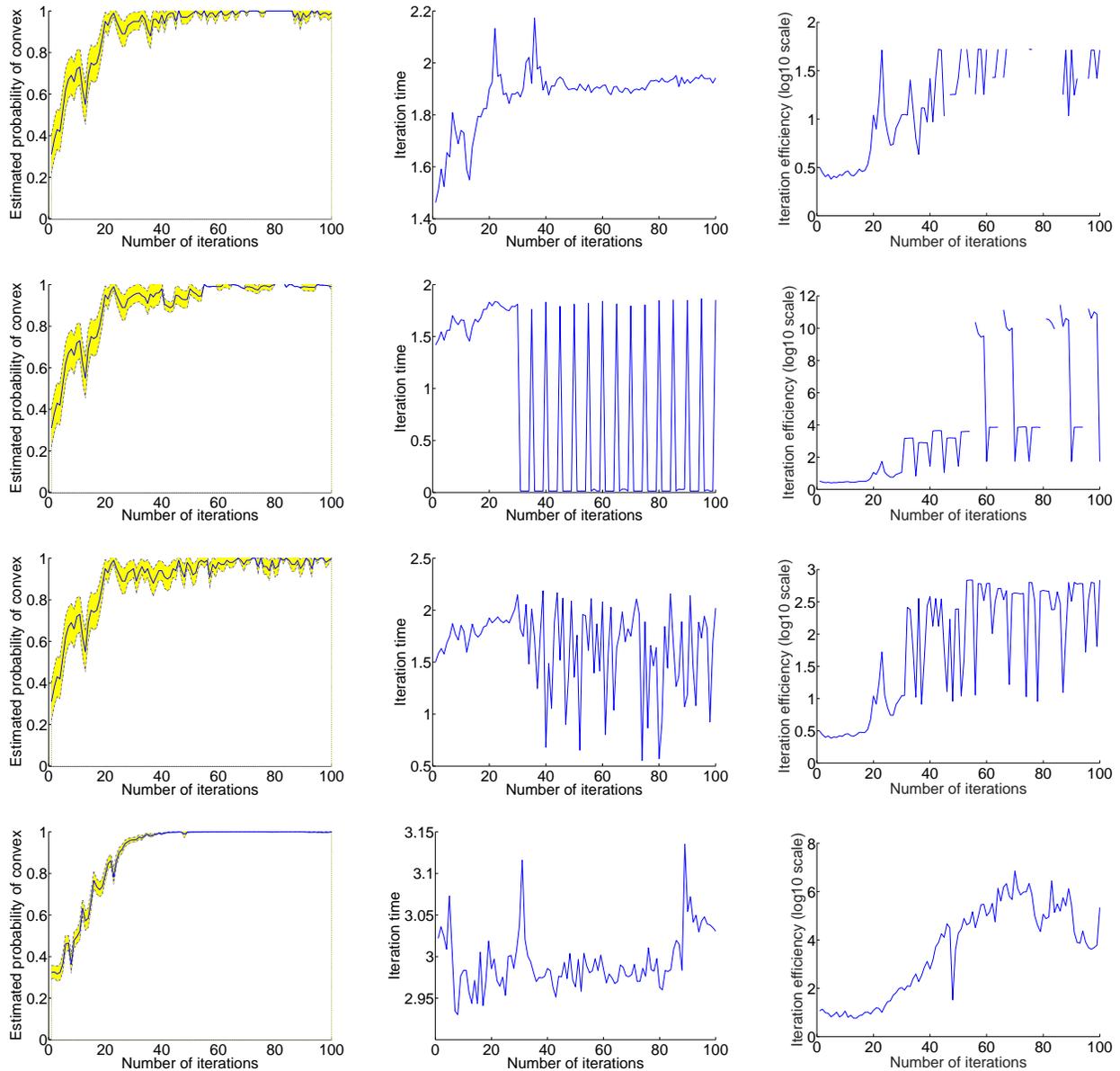

Figure 2   Comparisons of the estimated probability of convexity, the iteration time (in seconds), and the log (base 10) efficiency (left to right) of vanilla Monte Carlo, change of measure, acceptance-rejection, and conditional-Monte-Carlo (top to bottom) methods applied to a one-dimensional strictly convex function.

Figure 2 shows that the estimated probability of convexity increases to 1 for all methods. Later in the iterations, the change of measure method can return greater-than-one estimates due to the poor behavior of the likelihood ratio as discussed in Section 7.1. Among all methods, the conditional Monte Carlo method has the smallest variance but takes the longest time to compute. (This difference in the computational time becomes more significant for higher-dimensional test functions when we later experiment on a 30-dimensional function.) For the 1-dimensional convex function here, taking both computational time and variance into consideration, we observe that conditional Monte Carlo has the highest overall efficiency. The efficiency of vanilla Monte Carlo is the lowest. The efficiency plots occasionally break when the sample variance of the estimator is 0, where all the linear systems (LS) are feasible. This also happens for the change of measure method because that method corresponds with vanilla Monte Carlo every 5 iterations after the first 30. The efficiency of the change of measure method and the acceptance-rejection method both increase whenever they reuse the samples from a previous iteration because the linear feasibility problems need not be solved. The change of measure method occasionally has a very large efficiency because of the small sample variance of the estimate. This happens in later iterations when the posterior density is very concentrated. In this case all the Monte Carlo samples are close to the mean, giving almost identical posterior densities and similar likelihood ratios. When all reused samples are convex (corresponding to the iterations where the vanilla Monte Carlo method has infinite estimated efficiency), the change of measure estimator has almost 0 variance. However, due to the heavy tail behavior of the likelihood ratio, it is risky to trust the change of measure estimator values, as we see when the change of measure method estimates a probability greater than 1. The acceptance-rejection estimator has slightly lower estimated efficiency, but the estimator is more trustworthy in that it is statistically identical to vanilla Monte Carlo.

Consider now the 30-dimensional test function $f(\boldsymbol{x}) = ||\boldsymbol{x}||^2, \boldsymbol{x} \in [-10, 10]^{30}$ with $r = 3(d+1) = 93$ sample points. The covariance matrix Γ has diagonal entries $\Gamma_{ii} = 0.04f^2(\boldsymbol{x}_i)$, and off-diagonal entries $\Gamma_{ij} = 10^{-2} \exp\{-||\boldsymbol{x}_i - \boldsymbol{x}_j||^2/2\}0.04f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)$. Hence the variance depends on the function value, and there is also modest positive correlation between any two design points depending on the distance between them. As before, for the change of measure method, a new set of samples is obtained every iteration for the first 30 iterations, and every 5 iterations thereafter, and for the acceptance-rejection method we start to

reuse samples only after the first 30 iterations. We find that the change of measure and acceptance-rejection methods do not work very well on this example. Indeed, according to Proposition 1, the heavy-tail behavior of the likelihood ratio becomes more severe with more design points. With the likelihood ratio often taking very large values, the change of measure estimates evaluate to large values with wide confidence intervals, as shown in Figure 3 (notice the y-axis scale). Due to the same reason, the acceptance-rejection method reduces to vanilla Monte Carlo by rejecting almost all previous samples, so we omit that method from the results in Figure 3. In early iterations, conditional Monte Carlo takes more than 6 minutes to generate an estimate using `CVX` with `Gurobi` (`linprog` takes over 1 hour), and the iteration efficiency is around 0.20. In comparison, the vanilla Monte Carlo method only takes 80 seconds per iteration at the beginning of the iteration by solving the decomposed LS(i), giving around the same level of iteration efficiency. However, towards the end of the 100 iterations, conditional Monte Carlo is able to reduce the variance of the estimated probability so well that the efficiency improves beyond that of vanilla Monte Carlo. Therefore we recommend using conditional Monte Carlo (with `CVX` + `Gurobi`) if one can afford the running time, and vanilla Monte Carlo otherwise or when `CVX` is not installed.

8.2. A Non-Convex Function

Consider the function $f(\boldsymbol{x}) = -||\boldsymbol{x}||^2, \boldsymbol{x} \in [-1, 1]^d$. In order to make the problem "harder," we choose the covariance matrix Γ to be $d^2/4$ on the diagonal, so that the sampling standard deviation is bigger than half of the function value, and 0 on the off-diagonal. Figure 4 gives the results from the vanilla Monte Carlo estimator, with acceptance-rejection applied after the initial 30 iterations, for varying dimensions. The estimated probabilities of convexity hover near zero over all iterations, especially in lower dimensions. This is perhaps intuitive: with few iterations the noise in the estimated function values dominates, and in the presence of large noise *any* function will appear to be nonconvex, while after many iterations, the nonconvexities of the (true) function dominate and are detected.
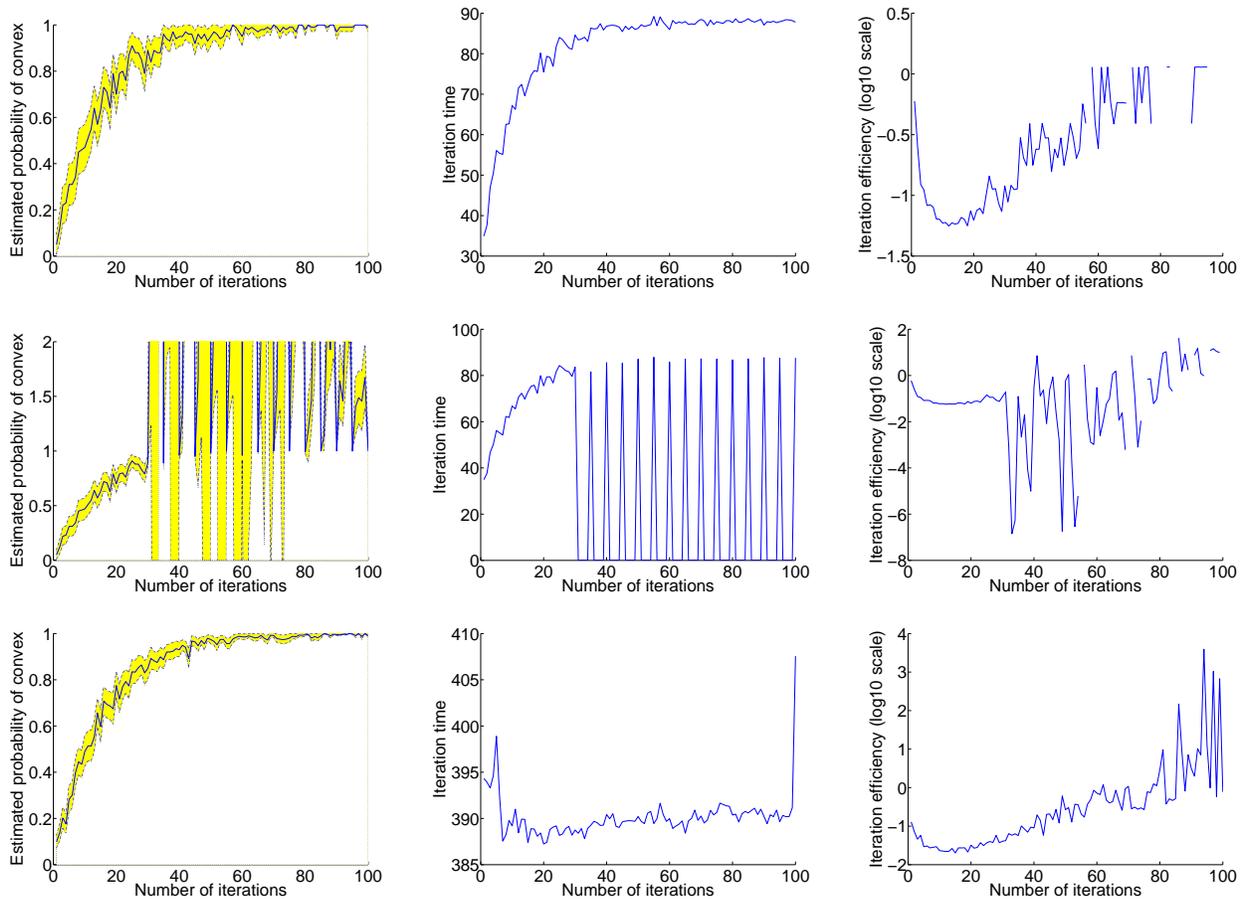
Figure 3   The estimated probability of convexity, the iteration time (in seconds), and the log (base 10) efficiency (left to right) of the vanilla Monte Carlo, change of measure, and conditional Monte Carlo methods (top to bottom) applied to a 30-dimensional strictly convex function.
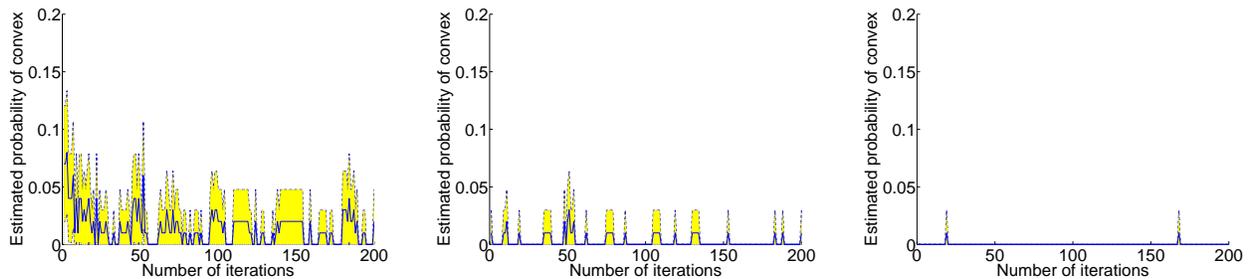

Figure 4   The estimated probability of convexity for the simple strictly non-convex function in dimensions 3 and 5 and 10 (from left to right). The estimates when the function is one-dimensional are not shown because they were effectively identically 0.

8.3. Linear Function

Linear functions are convex but lie on the boundary of the cone $\mathbb{C}$. Theorem 1 does not inform us of the likely behavior of our algorithm in this case, because the event that the

function lies on the boundary of $\mathbb{C}$ has measure 0 in the context of that result. Thus the posterior probability of convexity could converge to any number between 0 and 1 or not converge at all. Here we use a one-dimensional linear function $f(\boldsymbol{x}) = 0, \boldsymbol{x} \in [-1, 1]$, with a sampling covariance matrix that equals 10^{-4} on the diagonal and 0 on the off-diagonal.
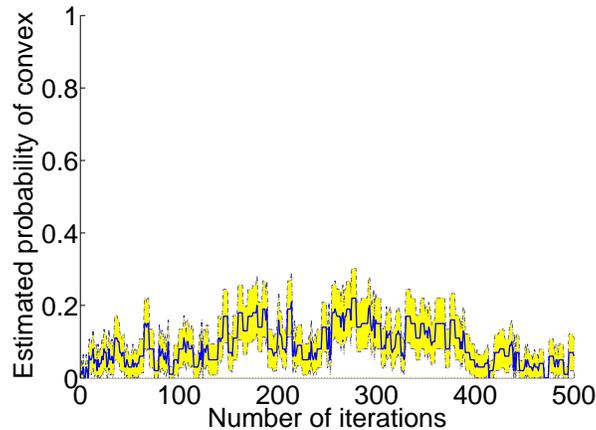

Figure 5   **The estimated probability of convexity for a 1-dimensional linear function. The mean does not appear to converge.**

As shown in Figure 5, the estimated probability does not converge to 0 or 1, but stays close to 0. When we increase the dimension, e.g., to 5, and keep the sampling variance the same, the estimated probabilities stay at 0 throughout the first 100 iterations. Changing the sampling variance does not change the qualitative nature of results because when the function is zero-valued the sampling variance only changes the "scale" of the observations. These results are perhaps to be expected because a linear function would only appear convex when the function noise at all design points "happens to" form a strictly convex function.

8.4.   Output of a Simulation

Finally, we have also tested our algorithm on a more realistic example similar to the "Ambulances in a Square" problem from SimOpt (Pasupathy and Henderson 2007). In this problem, patient calls arrive in a one kilometer unit square $[0, 1]^2$ according to a Poisson process at a constant rate of 1 call every 2 hours. The (x, y) locations of the calls are i.i.d. and distributed with a density proportional to $1.6 - (|x - 0.8| + |y - 0.8|)$. Upon receiving a call, a nearest ambulance is dispatched, traveling to the scene at a constant speed of 60 km/h. Once arriving at the scene, the ambulance spends a Gamma-distributed scene time

with mean 45 minutes and standard deviation 15 minutes, then returns to the base at a speed of 40 km/h if no other call is received. We are interested in the mean response time (time from when the call is received until the ambulance arrives at the call location) as a function of the location(s) of the ambulance base(s).

We sampled the base locations of the ambulance along ($4 \times$ the number of bases $+$ 1) random lines in the unit square, with 3 points sampled on each line. Each base has two coordinates, so this is equivalent to $3(2d+1)$ design points, where d is the dimension of the sample space. We are using more design points than in our previous test cases because we wanted to try more points (and consequently more computation) on a real case. Similar to our other experiments, we obtain a sample of the mean response time on each set of sampled base locations from running the simulation until 360 calls receive a response (approximately 30 days). The mean response times of the sampled base locations are evaluated using common random numbers, which compares the locations using the exact same random call arrivals and scene times. The convexity of the mean response time as a function of the ambulance base locations is tested with one, two, and three ambulance bases, using the conditional Monte Carlo method. The estimated probabilities vs. iteration are plotted in Figure 6.
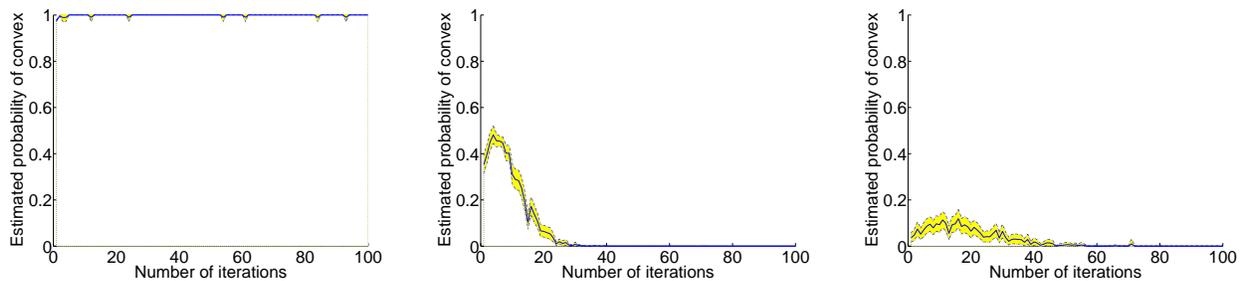

Figure 6   **The estimated probability of convexity for the mean response time as a function of the base locations, when the number of bases is one, two, and three.**

It seems that the mean response time is convex as a function of the base location when there is only one base, while it is not convex for more than one ambulance base. This agrees with our intuition that the location of a single ambulance base should have one global minimizer in the unit square. By plotting the posterior mean function, we found that the minimizer is located near the point $[0.46, 0.54]$, near, but slightly offset from, the mode $[0.8, 0.8]$, to balance the travel time to the farther corner $[0, 0]$. However, when there

is more than one ambulance base, the objective does not have a single minimizer due to symmetry and the interactions between bases.

9.   Conclusion

Given a function that can be observed on a finite number of points in the presence of noise, we have suggested a sequential algorithm to estimate the posterior probability that the function is convex. The method models the function values on a fixed set of design points using a Bayesian conjugate model, and estimates the probability of convexity by Monte Carlo simulation, using samples of the function values from the posterior distribution. This Bayesian procedure gives sequential estimates for function convexity. It is useful when a function is expensive to evaluate, e.g., the output of a large simulation, or when its values can only be obtained on a constrained set of points, e.g., a function defined on a discrete domain, and is primarily an exploratory tool to help an analyst develop an understanding of the geometry of an optimization problem.

To improve the efficiency of our algorithm we introduced three variance reduction methods - change of measure, acceptance-rejection, and conditional Monte Carlo. The first two methods reuse samples obtained in an earlier iteration to calculate an estimator in the current iteration. However, they both rely on the likelihood ratio of normal or Student-t posterior densities, which we prove could take extreme values due to its heavy-tail behavior. In our computational results, we observe that the change of measure method may give poor (e.g., greater than 1) estimates of the probability, and the acceptance-rejection method rejects most of the earlier samples and reduces to vanilla Monte Carlo when the number of design points is large. Finally, the conditional Monte Carlo method takes the longest time to compute but is the most effective in variance reduction, giving the highest efficiency among all methods. We recommend using it with `CVX` and `Gurobi`, especially for high-dimensional functions, to ensure reasonable computational time.

How should one choose n, the simulation runlength at each of the r design points? In our experiments, we increased n until the confidence intervals for p_n appeared to remain near 1 (suggesting convexity) or 0 (non-convexity). We suggest this exploratory procedure as a reasonable rule of thumb, recognizing that it is only a heuristic. More advanced stopping rules that offer some kind of overall statistical guarantee might be possible, which might even lead to the development of an hypothesis testing procedure. However, such a goal is

not in line with the exploratory aims of the present paper. Moreover, developing such rules would likely require considerable effort that we view to be beyond the scope of this paper.

In our experimentation, we also found the location of design points to be important when exploring an unknown function. Despite the fact that we only determine the convexity of a vector based on the pre-chosen design points, it would be helpful if the points are representative of the sample space S. Without knowing anything about the underlying function, a good starting point is to choose the design points such that they span the entire space. As we gain better knowledge with the sequential procedure, it is possible to expand the design points dynamically, and consequently have different sample sizes at each point. The method of choosing where and how much to sample in each iteration is left as an open problem.

A package containing the main algorithm and all variance reduction alternatives is available on Github (Jian 2017).

Acknowledgments

References

Abrevaya, Jason, Wei Jiang. 2005. A nonparametric approach to measuring and testing curvature. *Journal of Business & Economic Statistics* **23** 1–19. URL http://EconPapers.repec.org/RePEc:bes:jnlbes:v:23:y:2005:p:1-19.

Allon, Gad, Michael Beenstock, Steven Hackman, Ury Passy, Alexander Shapiro. 2007. Nonparametric estimation of concave production technologies by entropic methods. *Journal of Applied Econometrics* **22** 795–816. doi:10.1002/jae.918. URL http://dx.doi.org/10.1002/jae.918.

Atlason, J. M. A. Epelman, S. G. Henderson. 2004. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research* **127** 333–358.

Baraud, Y., S. Huet, B. Laurent. 2005. Testing convex hypotheses on the mean of a Gaussian vector. application to testing qualitative hypotheses on a regression function. *The Annals of Statistics* **33** 214–257.

Bernardo, José M., Adrian F. M. Smith. 2008. *Bayesian Theory*. John Wiley & Sons, Inc., 240–376. doi:10.1002/9780470316870.ch5. URL http://dx.doi.org/10.1002/9780470316870.ch5.

Chen, X., B. E. Ankenman, B. L. Nelson. 2012. The effects of common random numbers on stochastic kriging metamodels. *ACM TOMACS* **22** Article 7.

DeGroot, Morris H. 1970. *Optimal Statistical Decisions*. McGraw-Hill, New York, NY. URL http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+021834997&sourceid=fbw_bibsonomy.

Diack, C. A. T., C. Thomas-Agnan. 1998. A nonparametric test of the non-convexity of regression. *Nonparametric Statistics* **9** 335–362.

Feng, Mingbin, Jeremy Staum. 2015. Green simulation designs for repeated experiments. *Proceedings of the 2015 Winter Simulation Conference*. WSC '15, IEEE Press, Piscataway, NJ, USA, 403–413. URL http://dl.acm.org/citation.cfm?id=2888619.2888663.

Gelman, Andrew, John B. Carlin, Hal S. Stern, Donald B. Rubin. 2003. *Bayesian Data Analysis*. 2nd ed. Chapman & Hall/CRC Texts in Statistical Science, Chapman and Hall/CRC. URL http://www.worldcat.org/isbn/158488388X.

Glasserman, Paul. 2004. *Monte Carlo methods in financial engineering*. Springer, New York. URL http://www.amazon.com/Financial-Engineering-Stochastic-Modelling-Probability/dp/0387004513/ref=pd_sim_b_68?ie=UTF8&refRID=1AN8JXSDGMEV2RPHFC2A.

Glynn, Peter W., Gerd Infanger. 2013. Simulation-based confidence bounds for two-stage stochastic programs. *Mathematical Programming* **138** 15–42. doi:10.1007/s10107-012-0621-0. URL http://dx.doi.org/10.1007/s10107-012-0621-0.

Glynn, Peter W., Ward Whitt. 1992. The asymptotic efficiency of simulation estimators. *Oper. Res.* **40** 505–520. doi:10.1287/opre.40.3.505. URL http://dx.doi.org/10.1287/opre.40.3.505.

Golub, Gene H., Charles F. Van Loan. 1996. *Matrix Computations*. 3rd ed. Johns Hopkins Studies in Mathematical Sciences, The Johns Hopkins University Press. URL http://www.worldcat.org/isbn/0801854148.

Grant, Michael, Stephen Boyd. 2008. Graph implementations for nonsmooth convex programs. V. Blondel, S. Boyd, H. Kimura, eds., *Recent Advances in Learning and Control*. Lecture Notes in Control and Information Sciences, Springer-Verlag Limited, 95–110. http://stanford.edu/~boyd/graph_dcp.html.

Grant, Michael, Stephen Boyd. 2014. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx.

Gurobi Optimization, Inc. 2016. Gurobi optimizer reference manual. URL http://www.gurobi.com.

Hannah, L. A., D. B. Dunson. 2011. Multivariate Convex Regression with Adaptive Partitioning. *ArXiv e-prints* .

Jian, N. 2017. Matlab package for convexity detection. Created Dec. 22, 2016. https://github.com/njian/convexity.

Jian, N., S. G. Henderson, S. R. Hunter. 2014. Sequential detection of convexity from noisy function evaluations. A. Tolk, S. D. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, J. A. Miller, eds., *Proceedings of the*

30   **Jian and Henderson:** *Estimating the Probability of Convexity*

Article submitted to *INFORMS Journal on Computing*; manuscript no. (Please, provide the manuscript number!)

2014 Winter Simulation Conference. Institute of Electrical and Electronics Engineers, Inc., Piscataway, NJ, 3892–3903. doi:10.1109/WSC.2014.7020215.

Joe, Harry. 2014. *Dependence Modeling with Copulas*. Chapman and Hall/CRC, New York.

Judge, G. G., T. Takayama. 1966. Inequality restrictions in regression analysis. *Journal of the American Statistical Association* **61** pp. 166–181. URL http://www.jstor.org/stable/2283052.

Juditsky, A., A. Nemirovski. 2002. On nonparametric tests of positivity/monotonicity/convexity. *The Annals of Statistics* **30** 498–527.

Kaplan, Wilfred. 1991. *Advanced Calculus (4th Edition)*. 4th ed. Addison Wesley Publishing Company. URL http://www.worldcat.org/isbn/0201578883.

Lau, Lawrence J. 1978. Testing and imposing monoticity, convexity, and quasi-convexity constraints. *Electronic Journal of Statistics* **1** 409–453.

Lim, Eunji, Peter W. Glynn. 2012. Consistency of multidimensional convex regression. *Operations Research* **60** 196–208. URL http://dblp.uni-trier.de/db/journals/ior/ior60.html#LimG12.

Mathworks. 2016. Documentation for linprog. Accessed Feb. 17, 2017. https://www.mathworks.com/help/optim/ug/linprog.html.

Meyer, Mary C. 2012. Constrained penalized splines. *Canadian Journal of Statistics* **40** 190–206. doi:10.1002/cjs.10137. URL http://dx.doi.org/10.1002/cjs.10137.

Murty, K. G. 1988. *Linear Complementarity, Linear and Nonlinear Programming*. Heldermann Verlag, Berlin.

Nesterov, Yurii. 2004. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic.

Pasupathy, Raghu, Shane G. Henderson. 2007. Ambulance bases. Accessed May. 15, 2014. http://simopt.org/wiki/index.php?title=Ambulances_in_a_square.

Rasmussen, Carl Edward, Christopher K. I. Williams. 2005. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Rubinstein, R. Y., A. Shapiro. 1990. Optimization of static simulation models by the score function method. *Mathematics and Computers in Simulation* **32** 373–392.

Seijo, E., B. Sen. 2010. Nonparametric Least Squares Estimation of a Multivariate Convex Regression Function. *ArXiv e-prints* .

Silvapulle, Mervyn J., Pranab K. Sen. 2001. *Constrained Statistical Inference: Order, Inequality, and Shape Restrictions*, chap. 3. John Wiley & Sons, Inc., 59–141. doi:10.1002/9781118165614.ch3. URL http://dx.doi.org/10.1002/9781118165614.ch3.

Szechtman, Roberto, Enver Yücesan. 2016. A bayesian approach to feasibility determination. *Proceedings of the 2016 Winter Simulation Conference*. WSC '16, IEEE Press, Piscataway, NJ, USA, 782–790. URL http://dl.acm.org/citation.cfm?id=3042094.3042203.

Vogel, S. 1988. Stability results for stochastic programming problems. *Optimization* **19** 269–288.

Wang, Jianqiang C., Mary C. Meyer. 2011. Testing the monotonicity or convexity of a function using regression splines. *Canadian Journal of Statistics* **39** 89–107. doi:10.1002/cjs.10094. URL http://dx.doi.org/10.1002/cjs.10094.

Williams, D. 1991. *Probability with Martingales*. Cambridge mathematical textbooks, Cambridge University Press. URL http://books.google.com/books?id=RnOJeRpkOSEC.