Subregion-Adaptive Integration of Functions Having a Dominant Peak [1]

Alan Genz[2] and Robert E. Kass[3]

[1]to appear in *Journal of Computational and Graphical Statistics.*
[2]Department of Mathematics, Washington State University, Pullman, WA 99164-3113.
[3]Department of Statistics, Carnegie Mellon University, Pittsburgh, PA 15213-3890.

ABSTRACT

Many statistical multiple integration problems involve integrands that have a dominant peak. In applying numerical methods to solve these problems, statisticians have paid relatively little attention to existing quadrature methods and available software developed in the numerical analysis literature. One reason these methods have been largely overlooked, even though they are known to be more efficient than Monte Carlo for well-behaved problems of low dimensionality, may be that when applied naively they are poorly suited for peaked-integrand problems. In this paper we use transformations based on "split-t" distributions to allow the integrals to be efficiently computed using a subregion-adaptive numerical integration algorithm. Our split-t distributions are modifications of those suggested by Geweke (1989) and may also be used to define Monte Carlo importance functions. We then compare our approach to Monte Carlo. In the several examples we examine here, we find subregion-adaptive integration to be substantially more efficient than importance sampling.

Key Words: Importance sampling, Monte Carlo, multiple integrals, numerical quadrature, Bayesian computation, split-t distributions.

1 Introduction

In this paper we consider the numerical evaluation of multidimensional integrals in the form

$$I(f) = \int f(\boldsymbol{\theta}) d\boldsymbol{\theta},$$

where $\boldsymbol{\theta}$ is an m-dimensional vector of integration variables with m being not too large, typically $1 \leq m \leq 8$. We assume that the integrand $f(\boldsymbol{\theta})$ is characterized by the presence of a dominant peak. We call the basic integration strategy discussed here *subregion-adaptive integration* and describe it in Section 2. There are two essential features of the method. First, it dynamically allocates points at which the integrand is to be evaluated into those parts of the integration region where the integrand is most variable; that is, it attempts to find the important regions for the integrand and concentrate its effort there. Second, the rules used in each subregion provide exact integrals of low-order polynomials, and do so with a substantial reduction in the number of function evaluations compared to Gaussian product rules. In addition, when this strategy is implemented, the function evaluations over each subregion may be efficiently re-used to determine whether the region should be further subdivided. This approach has become established in the numerical analysis literature over roughly the past 20 years, and has been implemented in readily available public-domain software (in the NAG and CMLIB libraries).

Despite their apparent success and availability, however, these routines do not appear to have been used very much by statisticians, at least in the kind of problem we are discussing. The reason is simple: used in a naively straightforward way, for integrals having a dominant peak they are very inefficient. Our purpose here is to describe an effective "front end" to that methodology, so that problems with dominant peaks may be transformed to a domain in which that software should perform well.

Although the approach would apply to a diversity of problems, our statistical examples will be drawn from Bayesian analysis, so we will write $f(\boldsymbol{\theta}) = g(\boldsymbol{\theta})\tilde{L}(\boldsymbol{\theta})$ where $g(\boldsymbol{\theta})$ is some elementary real-valued function (e.g., $g(\boldsymbol{\theta}) = \theta_j$, a component of $\boldsymbol{\theta}$) and $\tilde{L}(\boldsymbol{\theta})$ is proportional to a posterior density function that has a dominant peak at its mode $\tilde{\boldsymbol{\theta}}$. We will assume that essentially all of the contribution to the integral comes from values of the integrand in some neighborhood of $\tilde{\boldsymbol{\theta}}$. In applications there are often many such integrals, for various alternative choices of $g(\boldsymbol{\theta})$, that need to be evaluated.

Many other strategies for integration are available, especially in Bayesian analysis (see Evans and Swartz, 1995, for an overview). Our investigation of subregion-adaptive integration was motivated by the need for a method that would be generally applicable and efficient in reasonably well-behaved problems. Such a method could become the basis of a single generic integration function in a programmming package such as Lisp-Stat or S-Plus.

The existing software we would like to take advantage of assumes the domain of integration is a product region of the form $[a_1, b_1] \times [a_2, b_2] ... \times [a_m, b_m]$. The naive approach to applying it in the problem defined above would be to try to find a suitable box of this form surrounding $\tilde{\boldsymbol{\theta}}$. For example, in a Bayesian analysis one might use some multiple of approximate marginal posterior standard deviations, taken from a Normal approximation to the posterior, as half-widths for intervals centered at the components of $\tilde{\boldsymbol{\theta}}$. The region over which the integrand has non-negligible values, however, will typically comprise only

a small fraction of such a multidimensional box. Thus, the software will typically either spend an extremely long time finding the region that matters, or it will miss it entirely and fail to evaluate the integral correctly.

The method we adopt here uses the additional information provided by the second derivatives of $\log \tilde{L}(\boldsymbol{\theta})$ at $\tilde{\boldsymbol{\theta}}$, together with evaluations of $\tilde{L}(\boldsymbol{\theta})$ (along rays emanating from $\tilde{\boldsymbol{\theta}}$) that help determine its shape. A transformation is then introduced to bring the domain to $[0,1] \times [0,1] \times \cdots \times [0,1]$ in such a way that, over subregions of this new domain, the integrand will be well approximated by low-order polynomials. In Section 3 we describe these transformations, which are based on split-t distributions. An initial step involves orthogonalization: we transform the axes to make the negative second derivative matrix, evaluated at the mode, equal to the identity.

The split-t transformations could be used with Monte Carlo integration, rather than subregion-adaptive integration, on $[0,1] \times [0,1] \times \cdots \times [0,1]$. This would be equivalent to using the split-t distributions as importance functions for Monte Carlo integration. Viewed in that light, our algorithm for choosing split-t distributions is a modification of that used by Geweke (1989) (see also Evans, 1991), and we believe it will be useful in that role. Subregion-adaptive integration would be expected to provide greater accuracy, however, because of its use of polynomial rules together with its adaptive character, which should enable it to recover well from failures of the transformation to model the shape of $\tilde{L}(\boldsymbol{\theta})$. In Section 4, we compare these two approaches in several simple examples, and in Section 5 we summarize our findings and conclusions.

2 Subregion Adaptive Numerical Integration

2.1 Background

Multidimensional numerical integration has traditionally been considered a difficult computational problem. The primary cause of the difficulties has been attributed to the "curse of dimensionality". This "curse" arises when a multidimensional integral over a product region is approximated in a natural way using a product of one dimensional integration rules. An appropriate one-dimensional *rule* (or formula)

$$\int_a^b f(\theta) d\theta \;\simeq\; \sum_{i=1}^{N} w_i f(\theta_i)$$

is used for each dimension. When these one-dimensional rules are combined in a product form we have the multidimensional product rule approximation

$$I(f) \;\simeq\; B_N(f) \;=\; \sum_{i_1=1}^{N} w_{1,i_1} \sum_{i_2=1}^{N} w_{2,i_2} \ldots \sum_{i_m=1}^{N} w_{m,i_m} f(\theta_{1,i_1}, \theta_{2,i_2}, \ldots, \theta_{m,i_m}).$$

This product rule requires $M = N^m$ values of the integrand, which increases rapidly with N even for modest values of m. The one-dimensional rules that have been used to construct product rules are usually polynomial integrating rules (e.g., Naylor and Smith, 1982). With these rules the points θ_i and weights w_i are chosen so that for a particular N, the rule will integrate exactly all polynomials with degree $\leq d$, for some d that increases with N

(the "Gauss" rules have the optimal $d = 2N - 1$). This polynomial integrating property is inherited by the product rules, which therefore converge very rapidly to $I(f)$ when $f(\boldsymbol{\theta})$ is sufficiently smooth. However, this rapid convergence might only be apparent when N is fairly large, and so for many ("cursed") practical problems it is not possible to obtain an accurate approximation to $I(f)$ in an acceptable amount of time using a sequence of product rules.

An extensive amount of research work (see Berntsen and Espelid, 1982, Genz and Malik, 1983, Genz, 1986, Keast, 1979, Keast and Lyness, 1979, Cools and Haegemans, 1994 and Cools and Dellaportas, 1994, for a selection of references) has been devoted to finding multidimensional polynomial integrating rules that do not require nearly as many points as the product rules. For example, the Genz-Malik rules require substantially fewer integrand values for increasing m and d than the corresponding Gauss rules. The degree 7 and 9 Genz-Malik rules have been used extensively as local integration rules for the type of subregion adaptive algorithms that are discussed in Section 3. For degree 7, the number of points increases with dimensionality roughly at the rate m^3 rather than at the product-rule rate of 4^m, and for degree 9, the number of points increases with dimensionality roughly at the rate m^4 rather than at the product-rule rate of 5^m.

A feature of these low M rules is that the weights w_i are not always positive. While this has been perceived by some statisticians (see Shaw, 1988a and 1988b and Dellaportas and Wright, 1991) as an obstacle to using these rules with positive integrands, the results presented in the examples given at the end of this paper do not support this perception. From a theoretical point of view, if we want convergence of a sequence of integration rules to the integral of a continuous function, it is necessary that the sum of the absolute values of the integration rule weights remain bounded (Engels, 1980, Chapter 4). This condition, for an integration rule sequence based on increasing d, has not been established for many of the non-product polynomial integrating rules. However, the way these rules are used in subregion adaptive integration algorithms, where d is fixed throughout the computation and the chosen rule is applied to smaller and smaller subregions, does provide a sequence of rules that satisfy the bounded absolute weight sum condition.

2.2 Overview of Subregion Adaptive Algorithms

With *subregion* adaptive methods, a finer and finer subdivision of the original integration region is dynamically constructed, with smaller subregions concentrated where the integrand is most irregular. Within each subregion a moderate degree *local* integration rule is used to provide an estimate for the integral. These local results are combined to produce the global estimate. Although various basic rule types could be used, polynomial integrating basic rules are usually used with the subregion adaptive methods because they can provide rapid convergence once the subdivision has been refined enough that a low degree polynomial approximation can provide an accurate approximation to the integrand. Typical input for this type of algorithm consists of (i) a description of the initial integration region R, (ii) the integrand, (iii) an error tolerance ϵ and (iv) a limit k_{max} on the total number of subregions allowed (this can either be used to limit the maximum time allowed or to limit the maximum workspace allowed).

Along with the chosen basic integration rule B we must also have an associated error

estimation rule E. In this section we let B_i be the approximation to the integral in a subregion R_i obtained using the basic rule, and let E_i be the estimate for the absolute error in B_i. If at some stage in the algorithm R has been subdivided into k subregions, the relevant pieces of information are kept in a list $S = \{(R_1, B_1, E_1), (R_2, B_2, E_2), ..., (R_k, B_k, E_k)\}$. Initially $R_1 = R$ with $k = 1$, B_1 and E_1 are computed. There are many possible adaptive strategies that may be used to dynamically refine the list S. A generic *globally* adaptive algorithm has a main loop with the following form:

Globally Adaptive Integration Algorithm

- Input: S, $f(\boldsymbol{\theta})$, an absolute error tolerance ϵ and a work limit W
- Algorithm:
  1. Begin with S, and global estimates for $I(f)$ and the error.
  2. **While** (error $> \epsilon$) and (total work $<$ W) **Do**
     (a) Determine a new subdivision by subdividing a largest error subregion.
     (b) Apply a local integration rule to $f(\boldsymbol{\theta})$ in the new subregion.
     (c) Update the subregion list and estimates for $I(f)$ and the error.
     Endwhile
- Output: Estimates for $I(f)$ and the absolute error.

In order to avoid the "curse", the subdivision step 2(a) in the generic globally adaptive algorithm must be done in a controlled way. A natural subdivision strategy subdivides the chosen region into 2^m pieces by halving along each coordinate axis, but suffers from the "curse", and does not take account of the potential for differences in the behavior of the integrand as $\boldsymbol{\theta}$ varies in different directions. A better strategy was developed by van Dooren and de Ridder (1976). With this strategy, the subdivision step 2(a) divides the selected subregion in half along the coordinate axis where the integrand is (locally) most rapidly changing. This allows the computational time to increase slowly and adaptation to occur only in those variables that cause most of the variation in the integrand. This strategy has been used in several algorithms, the most recent being an algorithm developed by Berntsen, Espelid, and Genz (1991a, with double precision Fortran implementation DCUHRE, 1991b), for a vector of integrals. Most statistical integration problems involve a set of similar integrands that differ only by some elementary factors, and have a common integration region. With these problems, a significant part of the computation required for each integrand (the computation of the posterior density) is the same for all of the integrands. The DCUHRE software, which we have used for the examples described later in this paper, computes simultaneous approximations to a vector of similar integrals over the same integration region, and can therefore take advantage of this special feature of statistical integration problems.

2.3 Better Error Estimation

The basic error estimation method for subregion adaptive integration algorithms uses, for some subregion R, a difference of two integration rules

$$\mathbf{E(f)} \;=\; |\mathbf{B(f)} \;-\; \mathbf{B'(f)}| \;\simeq\; |\int_R \mathbf{f(x)}d\mathbf{x} \;-\; \mathbf{B(f)}|.$$

The rule $\mathbf{B}'(\mathbf{f})$ usually has degree $d' < d$ (the degree of $\mathbf{B}(\mathbf{f})$) and uses a subset of the points for $\mathbf{B}(\mathbf{f})$. Simple error estimates of this type are often conservative (they are usually a better estimate for the error in $\mathbf{B}'(\mathbf{f})$) and sometimes unreliable. If combined with results from other basic rules they can be made more robust (Berntsen, Espelid and Genz, 1991a), but the effect of these more sophisticated techniques is usually an even more conservative estimate. The result is that even though the subregion adaptive algorithm that uses these careful error estimates may be producing very accurate integral results, it does not know when to stop, and therefore often continues a calculation much longer than is necessary. While it is important for the adaptive algorithm to use conservative error estimates for local subdivision decisions, the sum of these conservative estimates can be an overly conservative final error estimate. Conservativeness in software like DCUHRE is needed to maintain a relatively high level of reliability for a broad range of integrands, but for better-behaved problems the error estimation strategy should be modified.

An approach that has shown promise on several different problems (see Genz and Kass, 1991) is to use a combination of the conservative error estimates from the adaptive software, and differences between integral estimates from successive calls to the software. To be specific, let $\mathbf{E}_N$ and $\mathbf{I}_N$ be the respective error and integral estimates produced by the adaptive software using N values of the integrand. The revised estimate takes the form

$$\hat{\mathbf{E}}_N = |\mathbf{I}_N - \mathbf{I}_{N/2}| + w(N)\mathbf{E}_N,$$

where $w(N)$ is a heuristically chosen weight function for the conservative error estimates. In most practical multiple integration computations, it is not known in advance how much work will be needed to achieve a prescribed accuracy level, so a typical strategy is to request intermediate results from the integration software with $N = M, 2M, 4M, \ldots$ (with the initial M fairly small) and stop when there is a negligible change in successive results. The type of error estimate just given implements this strategy, but also places some weight on the more conservative estimates from all of the basic integration rules. The choice for $w(N)$ that we have used is $w(N) = (M/N)^\alpha$ for $N > M$, with $\hat{\mathbf{E}}_M = \mathbf{E}_M$, initially, and $\alpha = 0.5$. This choice allows the conservative estimates to initially have more weight, but these conservative estimates are given less weight as the adaptive algorithm proceeds. The parameter $\alpha = 0.5$ was chosen after some experimentation, to reflect behavior of the observed errors in several test problems.

3 Transformations

Subregion adaptive integration algorithms require repeated subdivision of the integration region. The globally adaptive algorithm described in the previous section chooses at each stage a priority subregion to be halved along some coordinate axis. When the integrand is highly peaked, the region over which the peak occurs may be very small and it is possible for it to be missed by a set of evaluation points. If this occurs at one stage, it could easily also occur at the next so that the strategy could fail to find the peak and, hence, fail to correctly evaluate the integral. For this reason it is important to use available knowledge about the location and shape of the dominant peak.

We begin with the modal value $\tilde{\boldsymbol{\theta}}$ and the modal covariance matrix Σ, which is the inverse of the negative Hessian matrix of $\log \tilde{L}(\boldsymbol{\theta})$ at $\tilde{\boldsymbol{\theta}}$. One possibility would be to define

cutoffs for each component of $\boldsymbol{\theta}$, i.e., $\theta_i \pm c \cdot \sigma_i$ for the i-th component, where σ_i is the square-root of the i-th diagonal element Σ_{ii} and c is some suitable number such as $c = 5$. This would be assuming that the essential mass of the integrand could be captured by a spherical Normal approximation extending c approximate standard deviation units in each component direction. This naive method, however, is not very successful. First, it ignores the covariance structure; second, it forces the algorithm to search for important contributions to the integrand in regions where they do not occur (e.g., near the corners of the box), and third, it presumes that the tails drop at the rate of a Normal density. In many cases it thereby leaves the problem too difficult and the algorithm will not terminate in the allotted time.

Our alternative is to transform the region of integration to $[0,1]^m$ using a suitable distribution function. We begin by reviewing in Section 3.1 the modal approximation transformation, which would assume that $\tilde{L}(\boldsymbol{\theta})$ has the form of a Normal density. Although it is not a method we recommend, it becomes a part of the overall approach we follow and it does avoid the first two difficulties mentioned above that are faced by the naive cutoff method. The idea is to apply an orthonormalizing transformation to obtain new coordinates which would, if the original $\boldsymbol{\theta}$ were Normally distributed with mean $\tilde{\boldsymbol{\theta}}$ and variance matrix Σ, be independent and standard Normal. Then, each new coordinate is transformed by the standard Normal distribution function to $[0,1]$. In Section 3.2 we describe the modification of this method that deals with the third of the difficulties mentioned above by allowing the tail behavior to be non-Normal.

The transformation methods we are describing to facilitate subregion-adaptive integration could also be used with Monte Carlo: they effectively define importance functions. That is, performing Monte Carlo integration over $[0,1]^m$ following the modal approximation transformation is equivalent to using the modal Normal approximation to define an importance function. We will compare subregion-adaptive methods with importance sampled Monte Carlo in the examples of Section 4.

We note that it will be important to obtain good numerical approximations for both the mode and the modal covariance matrix (Kass, 1987) since the completion of the transformation construction process, and the stable evaluation of the posterior at the integration points, depend on these quantities.

3.1 The Modal Approximation Transformation

The use of a transformation of variables can be viewed as a method of preconditioning the integrand so that the transformed integral is easier to compute. ¿From this point of view, a good transformation is one that concentrates the integrand evaluation points in the most important subregions of the integration region. For integrands with a dominant peak, one possible general transformation method is based on the assumption (Chen, 1985) that the factor $\tilde{L}(\boldsymbol{\theta})$ is approximately multivariate normal. This means $\tilde{L}(\boldsymbol{\theta}) \simeq Ke^{-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})^t\Sigma^{-1}(\boldsymbol{\theta}-\boldsymbol{\mu})}$ with $\boldsymbol{\mu} = \tilde{\boldsymbol{\theta}}$ and K being the normalizing constant. Here, it is advisable to initially transform the variables to improve this approximation. For instance, positive quantities could be transformed by logarithms. To determine $\boldsymbol{\mu}$ and Σ multivariate optimization techniques can be applied to the function $\log(\tilde{L}(\boldsymbol{\theta}))$ to obtain the mode $\boldsymbol{\mu}$ and the modal covariance matrix Σ. Although numerical optimization for functions of many variables may itself be

difficult, we do not deal with this part of the problem. In many applications good initial values are available and we have assumed that there is a pronounced peak in the integrand. The time for the approximate solution of this type of numerical optimization problem will usually be insignificant compared to the numerical integration time.

Once Σ and $\boldsymbol{\mu}$ are available, the Cholesky decomposition CC^t of Σ can be found. Then, using the transformation $\mathbf{y} = C^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu})$ (and the result $\boldsymbol{\theta}^t\Sigma^{-1}\boldsymbol{\theta} = \mathbf{y}^t C^t C^{-t} C^{-1} C\mathbf{y} = \mathbf{y}^t\mathbf{y}$), and Normal transformations $z_i = \Phi(y_i)$ on the $\mathbf{y}$ components yields

$$\int_{-\infty}^{\infty} ... \int_{-\infty}^{\infty} g(\boldsymbol{\theta})\tilde{L}(\boldsymbol{\theta})d\boldsymbol{\theta} = (\sqrt{2\pi})^m |C| \int_0^1 ... \int_0^1 g(\boldsymbol{\mu} + C\mathbf{y}(\mathbf{z}))h(\mathbf{y}(\mathbf{z}))\mathbf{dz},$$

where $\mathbf{y}(\mathbf{z}) = (\Phi^{-1}(z_1), ..., \Phi^{-1}(z_m))^t$ and $h(\mathbf{y}) = e^{\frac{1}{2}\mathbf{y}^t\mathbf{y}}\tilde{L}(\boldsymbol{\mu} + C\mathbf{y})$.

This general transformation is straightforward to apply and as long as the stated assumption of approximate normality is correct, this method should be reasonably efficient because the transformed integrand should be approximately constant.

3.2 Split-t Transformations

A possible problem with the modal approximation transformation occurs when $\tilde{L}(\theta)$ has tail behavior that is not accurately modeled by the multivariate Normal distribution. There are several schemes that might improve on the modal approximation transformation. The method that we consider here is to use a modification of the split-t transformation technique discussed by Geweke (1989, 1991). With this technique, we again (as in the previous section) use the Cholesky decomposition CC^t of Σ to define $\mathbf{y} = C^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu})$, but the transformation $z_i = \Phi(y_i)$ is modified by splitting at the origin and using alternative functions in each direction that could have (i) the form of Student's t distribution function and (ii) scale factors different than 1 (which would be standard deviations in the Normal case). The function $\tilde{L}(\boldsymbol{\theta})$ is investigated along the directions indicated by the Cholesky decomposition of the modal covariance matrix, in order to determine the appropriate degrees of freedom and scale factors. For our method we start with the same basic model as Geweke, but we have a different algorithm for choosing the scale factors. We also extend Geweke's model by allowing different degrees of freedom parameters for each dimension, and we provide an explicit algorithm for choosing these parameters.

To be more specific, we first consider the one dimensional integral

$$I(\tilde{L}) = \int_{-\infty}^{\infty} \tilde{L}(\theta)d\theta.$$

Given the mode μ and the modal variance σ^2, we define $y = (\theta - \mu)/\sigma$ and write $I(\tilde{L})$ as

$$I(\tilde{L}) = \sigma \int_{-\infty}^{\infty} \tilde{L}(\mu + \sigma y)dy = \sigma \int_{-\infty}^{\infty} L(y)dy,$$

and then use an assumption that $L(y) \doteq Ke^{-y^2/2}$ to motivate the final transformation $z = \Phi(y)$, where Φ is the standard Normal distribution function. Thus, we obtain

$$I(\tilde{L}) = \sigma \int_0^1 \tilde{L}(\mu + \sigma\Phi^{-1}(z))\phi(z)^{-1}dz.$$

In order to handle more general tail behavior for $L(y)$, we use the Student's t family of distributions. With this family, we model $L(y)$ using

$$L(y) \doteq K(1 + \frac{y^2}{\nu\delta^2})^{-(\nu+1)/2}$$

with $\nu > 0$ (where K is the normalizing constant). To account for possibly different tail behavior to the left and right of the mode at $y = 0$, we will allow different δ and ν values to be used for negative and positive y. This then will give us a "split-t" model for $L(y)$.

The determination of good values for δ and ν based on several evaluations of $L(y)$ is a nonlinear approximation problem that could be solved using a variety of methods. We choose to examine the value of $L(y)$ at several points $\pm y$ and then fit values δ^- and ν^- for y negative, and δ^+ and ν^+ for y positive. We first determine approximate $\delta^\pm$ values by approximately solving $\log(L(\pm\alpha\delta^\pm)/L(0)) = -1.25$ with $\alpha = \sqrt{2.5}$ (the equation is solved twice: once for δ^- and once for δ^+). The value $\alpha = \sqrt{2.5}$ was chosen because the log of the scaled (divided by K) model, when evaluated at $y = \pm\alpha\delta$, is $-(\nu+1)\log(1+2.5/\nu)/2 \doteq -1.25$, with less than 5% relative error for all $\nu \geq 0.6$. The use of $\alpha = \sqrt{2.5}$ therefore allows us to determine good approximate values for $\delta^\pm$ independent of ν. (We expect values of ν much less than 0.6 to be rarely needed.)

Having found $\delta^\pm$, we need to determine some $\nu^\pm$ values so that the model approximates $L(y)$ for both positive and negative y. We already know that the model provides a good approximation to $L(y)$ at $y = \pm\sqrt{2.5}\delta^\pm$ for all $\nu \geq 0.6$, and at $y = 0$. We would like to choose $\nu^\pm$ values so that the model approximates $L(y)$ at some other y values. We could pick some appropriate β and solve $\log(L(\pm\beta\delta^\pm)/L(0)) = -(\nu^\pm+1)\log(1+\beta^2/\nu^\pm)/2$ for $\nu^\pm$, in order to get an exact fit at $y = \pm\beta\delta^\pm$. We choose instead to get an approximate fit at two other sets of y values, $y = \pm\delta^\pm$ and $y = \pm 2\delta^\pm$. In order to accomplish this, we determine the $\nu^\pm$ values that minimize

$$\left|\frac{\nu+1}{2}\log\left(1+\frac{4}{\nu}\right) + \log\left(\frac{L(\pm 2\delta^\pm)}{L(0)}\right)\right| + \left|\frac{\nu+1}{2}\log\left(1+\frac{1}{\nu}\right) + \log\left(\frac{L(\pm\delta^\pm)}{L(0)}\right)\right|.$$

At this stage we should have approximate values for both sets of parameters, δ^- and ν^- and δ^+ and ν^+, with

$$L(y) \doteq K(1 + \frac{y^2}{\nu^\pm(\delta^\pm)^2})^{-(\nu^\pm+1)/2}$$

for $y = 0$, $y = \pm\delta^\pm$, $y = \pm\alpha\delta^\pm$ and $y = \pm 2\delta^\pm$. The procedure could be iterated to further refine $\delta^\pm$ and $\nu^\pm$, but we found that further iteration did not really improve the quality of the model, with typical test problems. The algorithm we used assumes a Normal approximation for $\nu \geq \nu_{max}$. It is used twice: once for the δ^- and ν^- parameters, and once for the δ^+ and ν^+ parameters.

Algorithm for Determining Split Transformation Parameters

- Solve $\log(L(\pm\alpha\delta^\pm)/L(0)) = -1.25$ for $\delta^\pm$ to 5% relative accuracy, using the Secant method, with $\alpha = \sqrt{2.5}$.

- Determine ν^- and ν^+ in the range $(0, \nu_{max})$ to minimize

$$\left|\frac{\nu+1}{2}\log\left(1+\frac{4}{\nu}\right)-\log\left(\frac{L(\pm 2\delta^{\pm})}{L(0)}\right)\right|+\left|\frac{\nu+1}{2}\log\left(1+\frac{1}{\nu}\right)-\log\left(\frac{L(\pm\delta^{\pm})}{L(0)}\right)\right|.$$

In the examples in Section 4 we used $\nu_{max} = 8$ and minimized over integer values of ν. As we note in Section 4.1, however, there might be cases where it will be better to let ν range below the value 1.

The values of $\delta^+, \delta^-, \nu^+, \nu^-$ resulting from this algorithm are used to construct the final transformation for $I(f)$. Instead of $y = \Phi^{-1}(z)$, we use

$$y = T^{-1}(z) = \begin{cases} \delta^{\pm} t^{-1}(\nu^{\pm}, z) & \text{if } \pm(z-0.5) > 0 \text{ and } \nu^{\pm} < \nu_{max} \\ \delta^{\pm}\Phi^{-1}(z) & \text{if } \pm(z-0.5) > 0 \text{ and } \nu^{\pm} = \nu_{max} \end{cases}$$

with $z\epsilon[0,1]$, $t(\nu, y) = N_\nu^{-1}\int_{-\infty}^{y}(1+s^2/\nu)^{-(\nu+1)/2}ds$ and N_ν chosen so that $t(\nu,\infty) = 1$.

For multidimensional problems, we need to use this algorithm for each of the integration variables. Assuming that the mode $\boldsymbol{\mu}$ and modal covariance Cholesky factor C have been found, we define $L(y) = \tilde{L}(\boldsymbol{\mu} + c_{ii}\mathbf{e}_i y)$ (where $\mathbf{e}_i$ is the standard i^{th} Euclidean basis vector), for the i^{th} variable, for $i = 1, 2, ..., m$. We then use the split transformation parameter selection algorithm twice for each i, to obtain transformation parameters $\nu_i^{\pm}$ and $\delta_i^{\pm}$. The work needed to determine the mode, modal covariance matrix and split transformation parameters will usually be insignificant compared to the work for the integration. The tests we have completed indicate that most of the work for the construction of our transformation comes in the determination of the mode, and this work can be significantly reduced if a good initial value is given to start the numerical optimization. After the mode has been found, we need $O(m^3)$ work to determine Σ and C. The determination of the split-t parameters requires only $O(m)$ work, because we need a few $L(y)$ values for each of the integration variables in order to fine good $\delta^{\pm}$ and $\nu^{\pm}$ parameters.

This completes our discussion of the transformation selection process. The selected transformations will allow the original domain of integration (which we assume to be effectively infinite) to be transformed to the unit m-cube:

$$\int_{-\infty}^{\infty}...\int_{-\infty}^{\infty} g(\boldsymbol{\theta})\tilde{L}(\boldsymbol{\theta})d\boldsymbol{\theta} = |C|\int_0^1...\int_0^1 g(\boldsymbol{\mu}+C\mathbf{y}(\mathbf{z}))h(\mathbf{y}(\mathbf{z}))\mathbf{dz},$$

where $\mathbf{y}(\mathbf{z}) = (T_1^{-1}(z_1), ..., T_m^{-1}(z_m))^t$ and $h(\mathbf{y}(\mathbf{z})) = (\prod_{i=1}^m J_i(z_i))\tilde{L}(\boldsymbol{\mu}+C\mathbf{y}(\mathbf{z}))$, with the transformation Jacobian factors J_i defined by

$$J_i(z_i) = \begin{cases} \delta_i^{\pm} N_{\nu_i^{\pm}}(1+z_i^2/\nu_i^{\pm})^{(\nu_i^{\pm}+1)/2} & \text{if } \pm z_i > 0 \text{ and } \nu_i^{\pm} < \nu_{max} \\ \delta_i^{\pm}\sqrt{2\pi}e^{z_i^2/2} & \text{if } \pm z_i > 0 \text{ and } \nu_i^{\pm} = \nu_{max}. \end{cases}$$

If the transformation process has been successful, $h(\mathbf{y})$ should be approximately constant, and therefore fairly easy to integrate with standard numerical integration methods for the hypercube.

It remains possible that some tails of the posterior will be thicker than those detected by our transformation algorithm, so that $h(\mathbf{y})$ becomes unbounded at some places along

the boundary of the transformed integration region. In principle, if the original integral is finite, then the transformations we use will result in a finite transformed integral, even though the integrand may be unbounded in parts of the integration region; the integration rules we use do not require integrand values at the boundary of the integration region, and our algorithm can adapt to boundary singularities. In practice, an unbounded $h(\mathbf{y})$ could slow convergence. This slow convergence would typically be observed and would serve as a warning. We have not come across real examples where tail thickness has caused misleading results with this method. On the other hand, such examples could be constructed and the possibility of inaccuracy due to undetected heavy tails should be kept in mind.

4 Examples

We now illustrate our subregion-adaptive scheme and compare it with importance sampled Monte Carlo integration. The Monte Carlo integration method used for all of our tests is an antithetic variate enhanced version of crude Monte Carlo. With this standard technique (see Davis and Rabinowitz, 1984 or Evans and Swartz, 1992) for improving the convergence of crude Monte Carlo, for every random point $\mathbf{z}\epsilon[0,1]^m$ at which we sample the transformed integrand, we also sample at the point $(1,1,...,1)-\mathbf{z}$. In Section 4.1 we use a one-dimensional example to show how important the tails in the transformation can be. In Sections 4.2-4.4 we apply the technique to three multidimensional problems. In the calculations for all of these examples we have scaled the posterior $\tilde{L}$ by adding the constant $-\log(\tilde{L}(\boldsymbol{\mu}))$ to $\log(\tilde{L})$ during the computation of $\tilde{L}$. This avoids underflow problems that can occur with some statistical problems.

4.1 A One Dimensional Example

In this section we consider the computation of integrals using a Pearson Type IV density function (see Johnson and Kotz, 1970, p. 12). These integrals have the form

$$I(g) = \int_{-\infty}^{\infty} g(\theta) \frac{e^{\frac{-\rho\nu}{2}(\frac{\pi}{2} - atan(\frac{\theta-\lambda}{\omega\sqrt{\nu}}))}}{(1+\frac{(\theta-\lambda)^2}{\omega^2\nu})^{\frac{\nu+1}{2}}} d\theta\,.$$

The exact mode and modal variance for the density function are given by

$$\mu = \lambda + \rho\omega\nu^{1.5}/(1+\nu)$$

and

$$\sigma^2 = (\omega^2(\nu + 2\nu^2 + \nu^3(1+\rho^2)))/(1+\nu)^3.$$

In order to investigate the behavior of the automatic transformation selection method described in the previous section we fixed $(\lambda,\omega,\rho,\nu) = (0,1,20,4)$ and used $g = 1$, θ and θ^2.

We first approximately computed $I(1)$, $I(\theta)$ and $I(\theta^2)$ using no transformation, but with the integration domain truncated to the interval [-2000,2000]. This particular truncated domain was determined by using domains of the form $[-\alpha,\alpha]$ for different choices of α and selecting the smallest domain where the results were accurate to at least three decimal digits. Computation with the truncated interval [-2000,2000] required 165 integrand

evaluations to compute results accurate to three or four decimal digits with a simple one dimensional globally adaptive algorithm. Larger domains could have been used to achieve more accurate results, but at a higher total integrand evaluation cost.

Next, we computed the three integrals $I(1)$, $I(\theta)$ and $I(\theta^2)$ using an unsplit Normal transformation with a simple one dimensional subregion-adaptive algorithm. This computation terminated after 765 integrand evaluations had been made, with only one or two digit accuracy in the results. Then we computed the integrals using the transformation selected by the algorithm in Section 3.2. The algorithm selected a split Normal/Cauchy model with 0.66 and 1.69 as respective scale factors (i.e. $(\nu^-, \delta^-, \nu^+, \delta^+) = (8, 0.66, 1, 1.69)$). This computation required only 45 integrand evaluations to achieve three to four digit accuracy. Further investigation revealed that the poor results obtained using the Normal transformation are mostly because of the limited integration domain that results from the use of finite precision arithmetic. For $z \simeq 1$ we theoretically expect very large values for $\Phi^{-1}(z)$, but standard software returns maximum values for $\Phi^{-1}(z)$ in the range 5-6 (single precision, with range 6-7 for double precision). For the test problem considered in this section where $\sigma = 14.34$, the upper limit for the actual integration domain when the inverse Normal transformation is used is therefore less than 100, and this is too small to accurately account for the tail behavior in the test problem integrand. Similar results were obtained with other values of skewness parameter ρ. In this example ν was restricted to be at least 1 and it is likely that better results would have been obtained if we had allowed ν to be less than one.

We then computed the three integrals using importance sampled Monte Carlo integration. This method required about 250,000 integrand evaluations for only two-digit accuracy with the split transformation and 6,000,000 integrand evaluations for two-digit accuracy with the Normal transformation. Three-digit accuracy would require approximately one hundred times as much work.

We also used this setting to illustrate an important difficulty that can arise with sequential Gauss-Hermite schemes (Naylor and Smith, 1982; Dellaportas and Wright, 1991). We applied a sequence of N-point Gauss-Hermite rules, for $N = 1, 2, ..., 50$, to $e^{x^2/2} g(\mu + \sigma x)\tilde{L}(\mu + \sigma x)$. For $N = 50$ the results were accurate to only two digits for $I(1)$ and only one digit for $I(\theta)/I(1)$, but the relative difference between results for $N = 50$ and $N = 49$ were approximately 10^{-4} for $I(1)$ and 10^{-3} for $I(\theta)/I(1)$, suggesting a much higher level of accuracy than was present. Although Gauss-Hermite rules and the generalizations developed by Dellaportas and Wright (1991) can be very effective for problems where the posterior density is approximately Normal, these results suggest that they should be used with caution when there is serious doubt about approximate Normality. The approach we have taken may be better able to adapt to moderate posterior non-Normality.

4.2 A Nonlinear Regression Example: BOD

Data on biochemical oxygenation demand (BOD) were analyzed by Bates and Watts (1988) and subsequently by Tanner (1993) using a two-parameter nonlinear regression model $y_i = \theta_1[1 - \exp(-\theta_2 x_i)] + \epsilon_i$, with Normal constant-variance errors. We use the prior $1/(360\sigma)$ on $(\theta_1, \theta_2, \sigma)$ over the region $(0, 60) \times (0, 6) \times (0, \infty)$. (Tanner allowed θ_1 and θ_2 to take negative values, which seems to us scientifically unreasonable.) After integrating out σ the

integrands have the form

$$I(g) = \int g(\theta_1, \theta_2)[S(\theta_1, \theta_2)]^{-3} \frac{1}{45\pi^3} d\theta_1 d\theta_2.$$

This example is interesting because the posterior is markedly non-Normal (see the cited references for plots of the likelihood and posterior contours).

Results using $g(\theta) = 1$, $g(\theta) = \theta_1$, and $g(\theta) = \theta_2$, shown in Tables 4.2.1 and 4.2.2, were computed using the DCUHRE software. The errors given are estimated errors from the adaptive integration software combined with the error estimation method described in Section 2.4. For comparison, we also applied an importance-sampled Monte-Carlo method combined with the split transformations. These results are given in Table 4.2.3. There the reported errors are standard errors from the Monte-Carlo method. The results from longer computer runs, which we believe to be accurate to three decimal digits, are $(\bar{1}, \bar{\theta}_1, \bar{\theta}_2) = (2.24, 18.8, 1.16)$.

Table 4.2.1: Adaptive Modal Results

	Expected Values		
$\tilde{L}(\theta_1, \theta_2)$'s	$\bar{1}$	$\bar{\theta}_1$	$\bar{\theta}_2$
333	1.380	19.216	0.5673
925	1.443	19.320	0.5695
2109	1.519	19.247	0.5894
4477	1.609	19.229	0.6112
9213	1.681	19.258	0.6247
Err. Est.	.0074	0.881	0.0408

Table 4.2.2: Adaptive Split Results

	Expected Values		
$\tilde{L}(\theta_1, \theta_2)$'s	$\bar{1}$	$\bar{\theta}_1$	$\bar{\theta}_2$
333	1.893	18.438	0.8288
925	1.930	18.648	0.9078
2109	2.044	18.582	1.0977
4477	2.069	18.589	1.0849
9213	2.131	18.435	1.1529
Err. Est.	.0067	0.426	0.1060

Table 4.2.3: Monte-Carlo Split Results

	Expected Values		
$\tilde{L}(\theta_1, \theta_2)$'s	$\bar{1}$	$\bar{\theta}_1$	$\bar{\theta}_2$
4500	1.81	18.8	0.76
Std. Err.	0.20	1.6	0.36

The splitting algorithm chose a t on 2 d.f. for the transformation of θ_2 in the positive direction, with a scale factor of 1.4 (for the negative direction of θ_2 and both directions of θ_1 the algorithm fitted Normal distributions with scale factors close to 1), and this led to dramatically improved answers in comparison with the modal Normal transformation.

There is also a substantial improvement over Monte Carlo in this example. (Although the Monte Carlo estimate of expectation of θ_1 was quite accurate, its large standard error indicates that this has occurred by chance.) Note that our estimated errors are reasonably close to the actual errors. An additional interesting result, not displayed here, was based on 9,300 Monte Carlo samples using the modal Normal approximation, which is equivalent to modal Normal importance sampling. The expectation approximations were very inaccurate, and also had Monte Carlo standard errors that were misleadingly small by an order of magnitude (presumably because the tails of the distribution were inadequately sampled). A version of Gauss-Hermite integration we implemented also performed very poorly in this problem.

4.3 Stanford Heart Transplant Data Example

This example has been considered in a number of papers (e.g. Naylor and Smith, 1982). The integrals of interest have the form

$$I(g) = \int_0^\infty \int_0^\infty \int_0^\infty g(\lambda, \tau, p)\tilde{L}(\lambda, \tau, p)d\lambda d\tau dp,$$

with

$$\tilde{L}(\lambda, \tau, p) = \prod_{i=1}^n \frac{p\lambda^p}{(\lambda+x_i)^{p+1}} \prod_{i=n+1}^N (\frac{\lambda}{\lambda+x_i})^p \prod_{j=1}^m \frac{\tau p\lambda^p}{(\lambda+y_j+\tau z_j)^{p+1}} \prod_{j=m+1}^M (\frac{\lambda}{\lambda+y_j+\tau z_j})^p.$$

We first make the transformation $(\lambda, \tau, p) = (e^{\theta_1}, e^{\theta_2}, e^{\theta_3})$, and the integrals now have the form

$$I(g) = \int_{-\infty}^\infty \int_{-\infty}^\infty \int_{-\infty}^\infty g(e^{\theta_1}, e^{\theta_2}, e^{\theta_3})\tilde{L}(e^{\theta_1}, e^{\theta_2}, e^{\theta_3})e^{(\theta_1+\theta_2+\theta_3)}d\boldsymbol{\theta}.$$

Then, numerical optimization is applied to the function $\log(\tilde{L}(e^{\theta_1}, e^{\theta_2}, e^{\theta_3})) + \theta_1 + \theta_2 + \theta_3$ to obtain the mode μ and modal covariance matrix Σ, with Cholesky factor C. It is important to include the $\theta_1 + \theta_2 + \theta_3$ terms (that arise from the Jacobian for the transformation to the $\boldsymbol{\theta}$ variables) in the objective function during the optimization, because the transformation selection and integration algorithms also use these terms. The maximum value is approximately -375, so the constant 375 was added to $\log(\tilde{L})$ during the integral computations in order to avoid underflow problems when computing $\tilde{L}$.

Results in Tables 4.3.1 and 4.3.2 were computed using the DCUHRE software. The results are similar because the transformation selection algorithm selected a normal/normal split for all three variables with the scale factors $\delta^{\pm} \simeq 1$. The errors given are estimated errors from the adaptive integration software combined with the error estimation method described in Section 2.4. For comparison, we also used an importance sampled Monte-Carlo method combined with the split transformations, for the integration. These results are given in Table 4.3.3. The reported errors are standard errors from the Monte-Carlo method. The results from longer computer runs which we believe to be accurate to four decimal digits are $(\bar{1}, \bar{\lambda}, \bar{\tau}, \bar{p})$ = (0.2969, 32.60, 1.047, 0.4969). (We believe Naylor and Smith, 1982, incorrectly reported the exact value of $\bar{\tau}$ to be 1.042.) Once again, as in the previous example, the best results are produced by the transformed subregion adaptive numerical

integration; the error estimates are reasonably close to the actual errors and considerably smaller than the Monte Carlo standard errors.

Table 4.3.1: Adaptive Modal Results

	Expected Values			
$\tilde{L}(\lambda,\tau,p)$'s	$\bar{1}$	$\bar{\lambda}$	$\bar{\tau}$	$\bar{p}$
315	0.2949	32.521	1.0323	0.4980
855	0.2958	32.530	1.0386	0.4977
2025	0.2961	32.556	1.0406	0.4976
4365	0.2963	32.585	1.0421	0.4976
9135	0.2964	32.597	1.0441	0.4975
Err. Est.	0.0005	0.037	0.0033	0.0007

Table 4.3.2: Adaptive Split Results

	Expected Values			
$\tilde{L}(\lambda,\tau,p)$'s	$\bar{1}$	$\bar{\lambda}$	$\bar{\tau}$	$\bar{p}$
315	0.2943	32.455	1.0364	0.4967
855	0.2947	32.488	1.0378	0.4972
2025	0.2955	32.507	1.0396	0.4970
4365	0.2962	32.573	1.0408	0.4973
9135	0.2964	32.604	1.0426	0.4974
Err. Est.	0.0005	0.049	0.0031	0.0003

Table 4.3.3: Monte-Carlo Split Results

	Expected Values			
$\tilde{L}(\lambda,\tau,p)$'s	$\bar{1}$	$\bar{\lambda}$	$\bar{\tau}$	$\bar{p}$
4500	0.3032	31.86	1.110	0.4885
Std. Err.	0.0036	0.95	0.037	0.0092

4.4 A Hierarchical Model Example

In this section we consider an integral that arose in a Hierarchical Model problem. An article by Carlin, Kass, Lerch and Huguenard (1992) considers two cognitive models for predicting error rates in computer-based tasks using the cognitive psychological concept of human working memory. One of the models (the full model) requires the evaluation of six-dimensional integrals. A reduced model involves five-dimensional integrals. Here, we consider the reduced model integrals.

The required integrals have the form

$$I(g) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{0}^{\infty}\int_{-\infty}^{\infty} g(\boldsymbol{\lambda})L(\boldsymbol{\lambda})\pi(\boldsymbol{\lambda})d\boldsymbol{\lambda}$$

with $\boldsymbol{\lambda} = (\gamma, \mu_\theta^{(1)}, \mu_\theta^{(2)}, \sigma_\theta, \alpha)^t$, and $g = 1$ and $\boldsymbol{\lambda}$. The prior density function $\pi(\boldsymbol{\lambda})$ is given

by

$$\pi(\boldsymbol{\lambda}) \;=\; \frac{e^{-\frac{1}{2}((\frac{\mu_\theta^{(1)}-a_1}{b_1})^2+(\frac{\mu_\theta^{(2)}-a_2}{b_2})^2+(\frac{\gamma-\mu_\gamma}{\sigma_\gamma})^2+(\frac{2}{d\sigma_\theta^2})+(\frac{\alpha-\mu_\alpha}{\sigma_\alpha})^2)}}{\sqrt{2\pi}b_1\sqrt{2\pi}b_2\sqrt{2\pi}\sigma_\gamma c d^c \sigma_\theta^{2(c+1)}\sqrt{2\pi}\sigma_\alpha},$$

and the likelihood function $L(\boldsymbol{\lambda})$ has the form

$$L(\boldsymbol{\lambda}) \;=\; \prod_{i=1}^{2}\prod_{j=1}^{10}\int_{-\infty}^{\infty}\frac{e^{-\frac{1}{2}(\frac{\theta_j^{(i)}-\mu_\theta^{(i)}}{\sigma_\theta})^2}}{\sqrt{2\pi}\sigma_\theta}w(\theta_j^{(i)},\boldsymbol{\lambda})d\theta_j^{(i)},$$

with

$$w(\theta_j^{(i)},\boldsymbol{\lambda}) = \prod_{k=1}^{3}\prod_{l=1}^{3}\frac{(e^{\theta_j^{(i)}+\gamma(k-1)+\alpha[l/2]})^{z_{i,j,k,l}}}{(1+e^{\theta_j^{(i)}+\gamma(k-1)+\alpha[l/2]})^{n_{i,j,k,l}}},$$

where [x] is used to denote the integer part of x and the $z_{i,j,k,l}$ and $n_{i,j,k,l}$ values come from experimental data. Putting everything together, the computation $I(f)$ could actually be considered a twenty-five dimensional numerical integration problem. However, we treat the problem as a five dimensional integral with a difficult integrand, part of which is a product of twenty one-dimensional integrals. Because variable σ_θ has limits 0 and ∞, the initial transformation $\sigma'_\theta \;=\; \log(\sigma_\theta)$ was used. Numerical optimization is then used to obtain the mode $\hat{\lambda}$ and modal covariance matrix Σ, with Cholesky factor C, and an additive log posterior constant, that turned out to be 232. Then the integral $I(g)$ can be put into the form

$$I(g) = |C|\int_0^1 ... \int_0^1 e^{w_4} g(\tilde{\boldsymbol{\lambda}}(\mathbf{z}))L(\tilde{\boldsymbol{\lambda}}(\mathbf{z}))\pi(\tilde{\boldsymbol{\lambda}}(\mathbf{z}))d\mathbf{z},$$

where $\tilde{\boldsymbol{\lambda}}(\mathbf{z}) = (w_1, w_2, w_3, e^{w_4}, w_5)^t$ is defined using $\mathbf{w} = \hat{\boldsymbol{\lambda}} + C\mathbf{y}(\mathbf{z})$ with

$$\mathbf{y}(\mathbf{z}) = (T_1^{-1}(z_1), T_2^{-1}(z_2), ..., T_5^{-1}(z_5))^t.$$

Results in Table 4.4.1 were computed using the DCUHRE software with the split transformation. The transformation selection algorithm chose inverse normal transformations for all cases, will all scale parameters approximately 1.0, so the modal and split transformation results were almost identical. Results in Table 4.4.2 were computed for comparison using an importance sampled Monte-Carlo algorithm, with the split transformations. The results from longer computer runs which we believe to be accurate to four decimal digits are $(\bar{1}, \bar{\gamma}, \bar{\mu}_\theta^{(1)}, \bar{\mu}_\theta^{(2)}, \bar{\sigma}, \bar{\alpha})$ = (0.1360, 1.016, -4.252, -4.859, 0.1773, 1.794). As in the previous two examples, transformed subregion-adaptive numerical integration outperforms Monte Carlo; the error estimates are again reasonably close to the actual errors and considerably smaller than the importance sampled Monte Carlo standard errors.

Table 4.4.1: Adaptive Split Transformation Results

	Expected Values					
$L(\boldsymbol{\lambda})$'s	$\bar{1}$	$\bar{\gamma}$	$\bar{\mu}_\theta^{(1)}$	$\bar{\mu}_\theta^{(2)}$	$\bar{\sigma}$	$\bar{\alpha}$
339	0.1309	1.0155	-4.250	-4.851	0.1707	1.7941
791	0.1331	1.0155	-4.250	-4.854	0.1714	1.7943
1921	0.1345	1.0155	-4.252	-4.855	0.1721	1.7944
4181	0.1352	1.0152	-4.251	-4.854	0.1730	1.7939
8927	0.1353	1.0155	-4.251	-4.856	0.1744	1.7937
Err. Est.	0.0005	0.0030	0.012	0.015	0.0037	0.0053

Table 4.4.2: Monte-Carlo Split Transformation Results

	Expected Values					
$L(\boldsymbol{\lambda})$'s	$\bar{1}$	$\bar{\gamma}$	$\bar{\mu}_\theta^{(1)}$	$\bar{\mu}_\theta^{(2)}$	$\bar{\sigma}$	$\bar{\alpha}$
4500	0.1386	1.012	-4.250	-4.84	0.190	1.796
Std. Err.	0.0036	0.032	0.096	0.15	0.017	0.053

5 Concluding Remarks

The purpose of the work reported here was to extend applicability of subregion-adaptive software and evaluate its effectiveness in an important class of statistical integration problems. We were not attempting to provide methodology for especially difficult integration problems, but rather an efficient scheme for analytically intractable yet well-behaved low-dimensional integrands, which arise frequently in simple Bayesian analyses.

We used split-t distributions to transform the domain of integration to the unit cube, where the previously existing software could be employed. These transformations were chosen because similar ones were shown by Geweke (1989) to be effective as importance functions for Monte Carlo integration. We expected this approach to be more efficient, for most problems of low dimensionality, because subregion-adaptive integration is known to be more efficient than Monte Carlo for regular problems of modest dimensionality on the unit cube (and importance sampling with a probability distribution is equivalent to crude Monte Carlo on the unit cube after transformation by the corresponding distribution function).

The subregion-adaptive error estimates were reasonably close to the actual errors. We may compare these error estimates to the corresponding Monte Carlo standard errors to get an idea of the gain in efficiency. Computing the efficiency ratio [(Monte Carlo standard error)/(subregion-adaptive error)]^2 for the results in Examples 2-4 (using as subregion-adaptive error the difference between the long-run values and the values reported on the second-to-last line, based on almost 4500 function evaluations) we find efficiency is improved by large factors: in only one case was the ratio less than 10 and in several it was more than 1000 (the median efficiency ratio among the 13 expectations reported there is 38, with quartiles of 20 and 900). We do not wish to overemphasize these numbers, but it is clear that subregion-adaptive integration offers substantial gains in efficiency judged in terms of accuracy for a given number of function evaluations. Thus, to achieve a given level of accuracy we would expect substantial reductions in running time for the subregion-adaptive scheme we propose. Our conclusion is not that Monte Carlo should be abandoned,

but rather that for routine evaluation of low-dimensional integrals having a dominant peak split-t-transformed subregion-adaptive integration will often be much more efficient.

The form of the transformation function matters. As in importance sampling, distributions with tails that are too thin can lead to very inefficient or even disastrous results. This was illustrated in the Pearson-family example. On the other hand, any probability distribution used in importance sampling may also be used to define a transformation to the unit cube; thus, by analogous procedures, it should be possible to improve any importance sampling method for integrating smooth functions of modest dimensionality.

We should acknowledge that all of our examples involve calculation of normalizing constants and expectations. These are important and sometimes challenging problems (e.g., DiCiccio, Kass, Raftery, and Wasserman, 1997), but marginal densities are often of interest as well. Although the method described here should be applicable, there are implementation issues that would need further attention before it could be applied effectively to marginal density calculation. This is one class of problems for which sampling methods are particularly well suited. We view subregion-adaptive integration as a complementary approach most useful, as it currently exists, for the calculation of normalizing constants and expectations.

The development of software to perform the analyses presented here is nearly complete. This software includes other transformations (such as unsplit multivariate Student-t) and other numerical integration methods such as adaptive importance sampling, modified Gauss-Hermite (Genz and Keister, 1996) and spherical-radial integration (Monahan and Genz, 1996a,b). The software will be made available through standard public domain internet sources, including *statlib*. Work on constructing a version for use with the *S* statistical computing environment is also underway.

ACKNOWLEDGMENTS

This work was supported by grants from the National Science Foundation and the National Institutes of Health. We appreciate the comments of several referees, which improved our presentation.

REFERENCES

Bates, D.M. and Watts, D.G. (1988) *Nonlinear Regression Analysis and its Applications.* John Wiley and Sons, New York.

Berntsen, J. and Espelid, T. O. (1982), On the Construction of Higher Degree Three Dimensional Imbedded Integration Rules, *SIAM J. Numer. Anal.*, 25, 222-234.

Berntsen, J., Espelid, T.O. and Genz, A. (1991a), An Adaptive Algorithm for the Approximate Calculation of Multiple Integrals, *ACM TOMS* 17, 437-451.

Berntsen, J., Espelid, T.O. and Genz, A. (1991b), Algorithm 698: DCUHRE- An Adaptive Multiple Integration Routine for a Vector of Integrals, *ACM TOMS* 17, 452-456.

Carlin, B. P., Kass, R. E., Lerch, J. and Huguenard, B. (1992), Predicting Working Memory Failure: A Subjective Bayesian Approach to Model Selection, *Journal of the American Statistical Association*, 87, 319-327.

Chen, C.F. (1985), On Asymptotic Normality of Limiting Density Functions with Bayesian Implications, *J. Royal Statist. Soc.* 47, 540-546.

Cools, R. and Dellaportas, P. (1994). The role of embedded integration rules in Bayesian statistics. Technical report, Athens University of Economics.

Cools, R. and Haegemans, A. (1994). An Imbedded Family of Cubature Formulas for n-Dimensional Product Regions, *J. Comput. Appl. Math.* 51, 251-262.

Davis, P. J. and Rabinowitz P. (1984), *Methods of Numerical Integration*, Academic Press, New York.

Dellaportas, P. and Wright, D. (1991), Positive Imbedded Integration in Bayesian Analysis, *Statistics and Computing* 1, 1-12.

DiCiccio, T.J., Kass, R.E., Raftery, A.E., and Wasserman, L. (1997), Computing Bayes factors by combining simulation and asymptotic approximations, *J. Amer. Statist. Assoc.*, to appear.

Engels, H. (1980), *Numerical Quadrature and Cubature*, Academic Press, New York.

Evans, M. (1991) Adaptive importance sampling and chaining, in N. Flournoy and R. K. Tsutakawa (eds.) *Statistical Numerical Integration*, Contemporary Mathematics, 115, American Mathematical Society, Providence, Rhode Island, 137-143.

Evans, M. and Swartz, T. (1992). Some integration strategies for problems in statistical inference. *Computing Science and Statistics*, 24, 310-317.

Evans, M. and Swartz, T. (1995). Methods for approximating integrals in statistics with special emphasis on Bayesian integration problems. *Statist. Science*, 10, 254-272.

Gelfand, A. and Smith, A. (1990), Sampling-based approaches to calculating marginal densities, *J. Amer. Stat. Assoc.* 85, 398-409.

Genz, A. (1986), Fully Symmetric Interpolatory Rules for Multiple Integrals, *SIAM J. Numer. Anal.*, 23, 1273-1283.

Genz, A. and Kass, R. E. (1991), An Application of Subregion Adaptive Numerical Integration to a Bayesian Inference Problem, 441-444 in *Computing Science and Statistics*, 23, E. Keramidas (Ed.), Interface Foundation of America.

Genz, A. and Keister, B. (1996), Fully symmetric interpolatory rules for multiple integrals over infinite regions, *J. Compu. Apl. Math.*, 72, to appear.

Genz, A. and Malik, A. A. (1983), An Imbedded Family of Fully Symmetric Numerical Integration Rules, *SIAM Journal of Numerical Analysis*, 20, 580-587.

Gelfand, A. E. and Smith, A. F. M. (1990), Sampling Based Approaches to Calculating Marginal Densities, *J. Amer. Stat. Assoc.*, 85, 398-409.

Geweke, J. (1989), Bayesian Inference in Econometric Models Using Monte Carlo Integration, *Econometrica*, 57, 1317-1340.

Geweke, J. (1991), Generic, Algorithmic Approaches to Monte Carlo Integration in Bayesian Inference, 117-135 in *Statistical Multiple Integration*, N. Flournoy and R. K. Tsutakawa (Eds.), Contemporary Mathematics, 115, American Mathematical Society, Providence, Rhode Island.

Johnson, N. L. and Kotz, S. (1970), *Continuous Univariate Distributions-I*, John Wiley and Sons, New York.

Kass, R. E. (1987), Computing observed information by finite differences, *Communications in Statistics: Simulation and Computation*, 16, 587-599.

Keast, P. (1979) Some Fully Symmetric Rules for Product Spaces, *J. IMA*, 23, 251-264.

Keast, P. and Lyness, J. N. (1979) On the Structures of Fully Symmetric Multidimensional Quadrature Rules, *SIAM J. Numer. Anal.*, 16, 11-29.

Monahan, J. and Genz, A. (1996a), A comparison of omnibus methods for Bayesian computation, *Computing Science and Statistics*, 27, to appear.

Monahan, J. and Genz, A. (1996b), Spherical-radial integration rules for Bayesian computation, Technical Report, Department of Mathematics, Washington State University.

NAG, Numerical Algorithms Group Limited, Wilkinson House, Jordan Hill Road, Oxford, OX2 8DR, UK

Naylor, J. C. and Smith, A. F. M. (1982), Applications of a Method for the Efficient Computation of Posterior Distributions, *Appl. Stat.*, 31, 214-225.

Shaw, J. E. H. (1988a), A Quasirandom Approach to Integration in Bayesian statistics, *Ann. Statist.*, 16, 895-914.

Shaw, J. E. H. (1988b), Aspects of Numerical Integration and Summarisation, in *Bayesian Statistics 3*, J. Bernardo, H. H. Degroot, D. V. Lindley and A. F. M. Smith (Eds.), Oxford University Press, 411-428.

Tanner, M. (1993), *Tools for Statistical Inference* Springer-Verlag, New York.

van Dooren, P. and de Ridder, L. (1976), An Adaptive Algorithm for Numerical Integration over an N-Dimensional Rectangular Region, *J. Comp. Appl. Math.* 2, 207-217.