Adaptive Multidimensional Integration: VEGAS Enhanced

G. Peter Lepage

Department of Physics, Cornell University, Ithaca, NY, 14853

Abstract

We describe a new algorithm, VEGAS+, for adaptive multidimensional Monte Carlo integration. The new algorithm adds a second adaptive strategy, adaptive stratified sampling, to the adaptive importance sampling that is the basis for its widely used predecessor VEGAS. Both VEGAS and VEGAS+ are effective for integrands with large peaks, but VEGAS+ can be much more effective for integrands with multiple peaks or other significant structures aligned with diagonals of the integration volume. We give examples where VEGAS+ is 2–19× more accurate than VEGAS. We also show how to combine VEGAS+ with other integrators, such as the widely available MISER algorithm, to make new hybrid integrators. For a different kind of hybrid, we show how to use integrand samples, generated using MCMC or other methods, to optimize VEGAS+ before integrating. We give an example where preconditioned VEGAS+ is more than 100× as efficient as VEGAS+ without preconditioning. Finally, we give examples where VEGAS+ is more than 10× as efficient as MCMC for Bayesian integrals with $D = 3$ and 21 parameters. We explain why VEGAS+ will often outperform MCMC for small and moderate sized problems.

1. Introduction

Classic VEGAS is an algorithm for adaptive multidimensional Monte Carlo integration [1]. It is widely used in particle physics: for example, in Monte Carlo event generators [2], and to evaluate low-order [3] and high-order [4] Feynman diagrams and cross sections numerically. It has also been used in other fields for a variety of applications including, for example, path integrals for chemical physics [5] and option pricing in applied finance [6], Bayesian statistics for astrophysics [7, 8, 9] and medical statistics [10], models of neuronal networks [11], wave-function overlaps for atomic physics [12], topological integrals for condensed matter physics [13], and so on.

Monte Carlo integration is unusually robust. It makes few assumptions about the integrand — it needn't be analytic or even continuous — and provides useful measures for the uncertainty and reliability of its results. This makes it well suited to multidimensional integration, and in particular adaptive multidimensional integration. Adaptive strategies are essential for integration in high dimensions because important structures in integrands often occupy small fractions of the integration volume. One might not think, for example, of the interior of a sphere of radius 0.5 enclosed by a unit hypercube as a "sharp peak," but it occupies only 0.0000025% of the hypercube in $D = 20$ dimensions.[1]

Classic VEGAS is very effective for integrands with sharp peaks, which are common in particle physics applications and Bayesian integrals. While it is particularly effective for integrals that are separable (into a product of one-dimensional integrals), it also works very well for non-separable integrals with large peaks. It works less well for integrands with multiple peaks or other important structures aligned with diagonals of the integration volume, although it is much better than Simple Monte Carlo integration.

In this paper, we describe a modification of classic VEGAS, which we call VEGAS+, that adds a second adaptive strategy to classic VEGAS's adaptive *importance sampling* [14]. The second strategy is a form of adaptive *stratified sampling* [14] that makes VEGAS+ far more effective in dealing with multiple peaks and diagonal structures in integrands, given enough integrand samples.

Email address: g.p.lepage@cornell.edu (G. Peter Lepage)

[1] VEGAS+, the algorithm described in this paper, can find the 20-D sphere with 10 iterations of $N_{\rm ev} = 10^7$ integrand samples each. Another 10 iterations gives an estimate for the volume that is accurate to 0.05%.

We begin in Sec. 2 by reviewing the algorithm for (iterative) adaptive importance sampling used in classic VEGAS. We discuss in particular the variable map used by VEGAS to switch to new integration variables that flatten the integrand's peaks. This is how VEGAS implements importance sampling. We describe the new algorithm VEGAS+ in Sec. 3 and give examples that illustrate how and why it works. We also discuss its limitations and how they can be mitigated.

The map used by VEGAS to implement importance sampling can be used in conjunction with other integration algorithms to create new hybrid algorithms. We show how this is done in Sec. 4, where we combine VEGAS+ with the MISER algorithm [15]. There we also show how to use sample data, from a Markov Chain Monte Carlo (MCMC) or other peak-finding algorithm, to optimize the VEGAS map before integrating; this can greatly reduce the cost of integrating functions with multiple high, narrow peaks.

Our conclusions are in Sec. 5. We continue in two appendices with further VEGAS+ examples that illustrate its use for adaptive multidimensional summation, high-order Feynman diagrams, and Bayesian curve fitting, where we compare VEGAS+ with MCMC for a 3-dimensional and a 21-dimensional problem.

When comparing algorithms we do not look at computer run times because these are too sensitive to implementation details. For realistic applications the cost of evaluating the integrand usually exceeds other costs, so we measure efficiency by the number of integrand evaluations used. Where statistical uncertainties are quoted, they correspond to one standard deviation;[2] small differences (e.g., 10%) between uncertainties are not significant. The VEGAS+ examples here were analyzed using a widely available Python/Cython implementation first released in 2013 [16]. The integrands are written in (vectorized) Python or, in one case, Fortran77.

One change since classic VEGAS was introduced 45 years ago is the enormous increase in speed of computers. As a result VEGAS+ integrals using 10^8 or 10^9 integrand evaluations require only minutes for simple integrands on a modern laptop. VEGAS+ is also easily configured for parallel computing [17].

2. Adaptive Importance Sampling

In this section we review the adaptive strategy used in the original VEGAS algorithm: its remapping of the integration variables in each direction to optimize Monte Carlo estimates of the integral. The strategy is an implementation of the standard technique of importance sampling. In the following sections we review how VEGAS implements this strategy, first for one-dimensional integrals and then for multidimensional integrals.

2.1. Remapping the Integration Variable

For one-dimensional integrals, the VEGAS strategy is to replace the original integral

$$I = \int_a^b dx\, f(x) \tag{1}$$

by an equivalent integral over a new variable y,

$$I = \int_0^1 dy\, J(y) f(x(y)), \tag{2}$$

where $J(y)$ is the Jacobian of the transformation. The transformation is chosen to minimize the uncertainty in a Monte Carlo evaluation of the integral in y-space.

A Simple Monte Carlo estimate of the y-integral (Eq. (2)) is given by

$$I \approx I_{\rm MC} \equiv \frac{1}{N_{\rm ev}} \sum_y J(y) f(x(y)) \tag{3}$$

[2] Error estimates from VEGAS and VEGAS+ in most examples come from multiple iterations which were combined as described in Sec. 2.5, with good χ^2s in each case. Estimated values for the integrals agreed with the exact values to within errors (i.e., mostly within $\pm 1\sigma$).

where the sum is over $N_{\rm ev}$ random points y uniformly distributed between 0 and 1. The estimate $I_{\rm MC}$ is itself a random number whose mean is the exact value I of the integral and whose variance is

$$\sigma_I^2 = \frac{1}{N_{\rm ev}}\Big(\int_0^1 dy\, J^2(y) f^2(x(y)) - I^2\Big) \tag{4}$$

$$= \frac{1}{N_{\rm ev}}\Big(\int_a^b dx\, J(y(x)) f^2(x) - I^2\Big). \tag{5}$$

The standard deviation σ_I is an indication of the possible error in the Monte Carlo estimate. The variance σ_I^2 can also be estimated from the Monte Carlo data:

$$\sigma_{\rm MC}^2 \equiv \frac{1}{N_{\rm ev}-1}\Big(\frac{1}{N_{\rm ev}}\sum_y J^2(y) f^2(x(y)) - I_{\rm MC}^2\Big) \tag{6}$$

The distribution of the Monte Carlo estimates $I_{\rm MC}$ becomes Gaussian in the limit of large $N_{\rm ev}$, assuming $J(y)f(x(y))$ is sufficiently integrable. There are non-Gaussian corrections, but they vanish quickly with increasing $N_{\rm ev}$. For example, the Monte Carlo estimates for I and σ_I^2 would be uncorrelated with Gaussian statistics, but here have a nonzero correlation that vanishes like $1/N_{\rm ev}^2$:

$$\begin{aligned}\big\langle (I_{\rm MC} - I)(\sigma_{\rm MC}^2 - \sigma_I^2)\big\rangle &= \big\langle (I_{\rm MC} - I)^3\big\rangle \\ &= \frac{1}{N_{\rm ev}^2}\int_0^1 dy\, \big(J(y) f(x(y)) - I\big)^3.\end{aligned} \tag{7}$$

2.2. VEGAS *Map and Importance Sampling*

VEGAS implements the variable map described in the previous section by dividing the x-axis into N_g intervals bounded by:

$$\begin{aligned} x_0 &= a \\ x_1 &= x_0 + \Delta x_0 \\ &\cdots \\ x_{N_g} &= x_{N_g-1} + \Delta x_{N_g-1} = b. \end{aligned} \tag{8}$$

where $N_g = 1000$ is typical. The transformed variable has value $y = i/N_g$ at point x_i, and varies linearly with x between x_is. Thus

$$x(y) \equiv x_{i(y)} + \Delta x_{i(y)}\, \delta(y) \tag{9}$$

relates x to y, where functions $i(y)$ and $\delta(y)$ are the integer and fractional parts of yN_g, respectively:

$$i(y) \equiv \text{floor}(yN_g) \tag{10}$$

$$\delta(y) \equiv yN_g - i(y). \tag{11}$$

This transformation maps the interval $[0, 1]$ in y-space onto the the original integration region $[a, b]$ in x-space. Intervals of varying widths Δx_i in x-space map into intervals of uniform width $\Delta y = 1/N_g$ in y-space. The Jacobian for this transformation,

$$J(y) = N_g \Delta x_{i(y)} \equiv J_{i(y)}, \tag{12}$$

is a step function whose values are determined by the interval widths Δx_i.

Substituting this Jacobian into Eq. (5) for the uncertainty in a Monte Carlo integration gives

$$\sigma_I^2 = \frac{1}{N_{\rm ev}}\Big(\sum_i J_i \int_{x_i}^{x_i+\Delta x_i} dx\, f^2(x) - I^2\Big). \tag{13}$$

Treating the J_i as independent variables, subject to the constraint

$$\sum_i \frac{\Delta x_i}{J_i} = \sum_i \Delta y = 1, \tag{14}$$

it is easy to show that σ_I^2 is minimized when

$$\frac{J_i^2}{\Delta x_i} \int_{x_i}^{x_i+\Delta x_i} dx\, f^2(x) = \text{constant}. \tag{15}$$

That is, the grid is optimal when the average value of $J^2(y(x))f^2(x)$ in each interval Δx_i is the same for every interval.

A transformation with this property greatly reduces the standard deviation when the integrand has high peaks. The Jacobian flattens the peaks in y-space and stretches them out, because

$$J \equiv \left|\frac{dx}{dy}\right| \propto \frac{1}{|f(x)|} \tag{16}$$

becomes small near a peak. This means that a uniform Monte Carlo in y-space concentrates integrand samples around the peaks in x-space. Each interval Δx_i receives on average the same number of samples ($= N_{\text{ev}}/N_g$), and the smallest intervals are placed where $|f(x)|$ is largest (because $J_i \propto \Delta x_i$). This concentrates samples in the most important regions, which is why this method is called importance sampling.

2.3. Iterative Adaptation

The set of x_is defined above constitutes the VEGAS map. To optimize Monte Carlo integration, VEGAS varies the Jacobian of the transformation by varying the interval sizes Δx_i, while keeping the sum of all Δx_is constant. This is done iteratively. First VEGAS estimates the integral with a uniform grid, accumulating information about the integrand in the process. This information is then used to construct an improved grid, and VEGAS makes a new estimate of the integral. Again information about the integrand is accumulated in the process, and used to further improve the grid. In this fashion, the VEGAS map adapts to the integrand over several iterations.

To illustrate how this is done, we continue with the example above where we generate a Monte Carlo estimate of the integral in y-space (Eq. (2)) by sampling the integrand at N_{ev} random points y (Eq. (3)). Given some initial grid, VEGAS accumulates the average value of $J^2 f^2$ for each interval in the grid while sampling the integrand to estimate the integral:

$$d_i \equiv \frac{1}{n_i} \sum_{x(y)\in\Delta x_i} J^2(y) f^2(x(y)), \tag{17}$$

where $n_i \approx N_{\text{ev}}/N_g$ is the number of samples in interval Δx_i. The averages d_i are used to refine the grid. The grid is optimal when all of the d_is are equal (Eq. (15)), so VEGAS adjusts the grid intervals Δx_i to make the d_i more constant across the integration region.

This algorithm, like most adaptive algorithms, tends to overreact in the early stages of optimizing its grid, since it has rather poor information concerning the integrand at this stage. It is important, therefore, to dampen the refinement process so as to avoid rapid, destabilizing changes in the grid. In VEGAS, the d_is are first smoothed and normalized:

$$d_i \to \frac{1}{\sum_i d_i} \times \begin{cases} (7d_0 + d_1)/8 & \text{for } i = 0 \\ (d_{i-1} + 6d_i + d_{i+1})/8 & \text{for } i \neq 0, N_g - 1 \\ (d_{N_g-2} + 7d_{N_g-1})/8 & \text{for } i = N_g - 1. \end{cases} \tag{18}$$

Smoothing is particularly important if the integrand has large discontinuities (e.g., step functions). For example, an abrupt increase in the integrand near the upper edge of interval Δx_i might be missed completely by the samples. Without smoothing, VEGAS would see a sudden rise in the function beginning only in Δx_{i+1}, and refine the grid accordingly, thereby missing the small but possibly significant part of the step in interval Δx_i. Smoothing makes this less likely by causing VEGAS to focus some effort on interval Δx_i as well as Δx_{i+1}.

Having smoothed the d_is, VEGAS then compresses their range, to avoid overreacting to atypically large sample values for the integrand. This is done by replacing each d_i with

$$d_i \to \Big(\frac{1-d_i}{\ln(1/d_i)}\Big)^{\alpha}, \tag{19}$$

where $\alpha \geq 0$ is typically of order one.[3] Parameter α can be reduced in situations where VEGAS has trouble finding or holding onto the optimal grid; $\alpha = 0$ implies no grid refinement.

The condition for an optimal map remains that all d_i, now smoothed and compressed, be roughly equal. If the map is not optimal, VEGAS attempts to improve it. First the d_is are treated as continuous quantities, and each d_i is distributed uniformly over its interval Δx_i. Then new intervals, specified by $\{x'_i, \Delta x'_i\}$, are chosen so that each contains an equal fraction of the total $d = \sum_i d_i$. The following algorithm achieves this:

1. Define δd to be the amount of d associated with each interval of the new grid,

$$\delta d \equiv \frac{\sum_i d_i}{N_g}, \tag{20}$$

and initialize the following variables:

$$\begin{aligned} x'_0 &= x_0 \\ x'_{N_g} &= x_{N_g} \\ i &= 0 = \text{index of current new } x' \\ j &= 0 = \text{index of current old } x \\ S_d &= 0 = \text{amount of } d \text{ accumulated.} \end{aligned} \tag{21}$$

2. Increment i. If $i \geq N_g$, the new grid is finished.
3. Skip to the next step if $S_d \geq \delta d$; otherwise add d_j to S_d, increment j, and return to the beginning of this step.
4. Subtract δd from S_d, and compute the boundary of the new interval by interpolation:

$$x'_i = x_j - \frac{S_d}{d_{j-1}}\Delta x_{j-1}. \tag{22}$$

Return to Step 2.

Replacing the old map with the new map, VEGAS proceeds to generate a new Monte Carlo estimate for the integral and another new map. This entire process is repeated until the map has converged and the estimate of the integral is sufficiently accurate.

2.4. Multidimensional Integrals

Monte Carlo integration, even with adaptive importance sampling, is not usually competitive with other algorithms for one-dimensional integration. It rapidly becomes competitive, however, as the dimensionality increases.

The VEGAS algorithm extends the algorithm described above to D-dimensional integrals over variables x^μ (with $\mu = 1 \ldots D$) by replacing each x^μ with a new variable y^μ. The variable transformation is specified by an independent VEGAS map $\{x^\mu_i, \Delta x^\mu_i\}$ for each direction.

The resulting integral is

$$I = \int_0^1 d^D y\, J(y)\, f(x(y)) \tag{23}$$

[3] Classic VEGAS typically defaults to $\alpha = 1.5$. VEGAS+ can use a smaller default value, $\alpha = 0.5$, because of its adaptive stratified sampling.

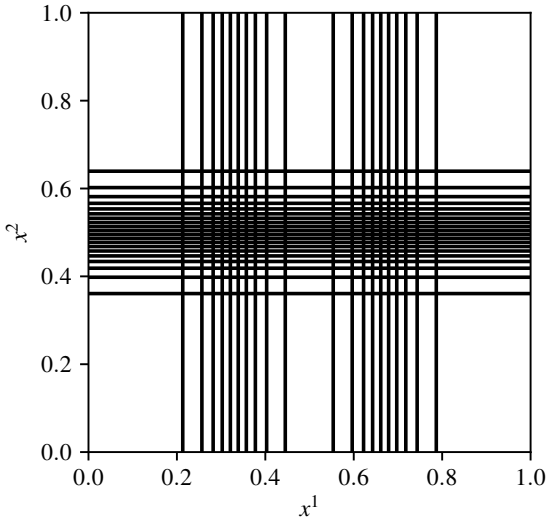

Figure 1: VEGAS map for the $D = 4$ integral defined in Eqs. (26) and Eq. (27). The figure shows grid lines for every 50$^{\text{th}}$ increment along the x^1 and x^2 axes; grids for the x^3 and x^4 axes are the same as for x^2.

where $x(y)$ is the D-dimensional VEGAS map and $J(y)$ its Jacobian. A Simple Monte Carlo estimate of this integral is obtained by sampling the integrand at random points $y = \{y^\mu\}$ distributed uniformly within the unit hypercube at the origin:

$$0 < y^\mu < 1. \tag{24}$$

The integrand samples are also used to calculate the averages

$$d_i^\mu \equiv \frac{1}{n_i^\mu} \sum_{x^\mu(y^\mu)\in\Delta x_i^\mu} J^2(y) f^2(x) \tag{25}$$

for every interval on every integration axis. These are used to improve the grid for each variable after each iteration, following the procedure described in Section 2.3.

Fig. 1 shows the grid corresponding to a VEGAS map optimized for the $D = 4$ dimensional integral

$$\int_0^1 d^4x \left(\mathrm{e}^{-100(\mathbf{x}-\mathbf{r}_1)^2} + \mathrm{e}^{-100(\mathbf{x}-\mathbf{r}_2)^2}\right), \tag{26}$$

where vector $\mathbf{x} = (x^1, x^2, x^3, x^4)$, and

$$\begin{aligned} \mathbf{r}_1 &= (0.33, 0.5, 0.5, 0.5) \\ \mathbf{r}_2 &= (0.67, 0.5, 0.5, 0.5). \end{aligned} \tag{27}$$

The grid concentrates increments near 0.33 and 0.67 for x^1, and near 0.5 in the other directions. Each rectangle in the figure receives, on average, the same number of Monte Carlo integration samples.

This VEGAS map has $N_g = 1000$ increments along each axis. The accuracy of the y-space integrals is typically insensitive to N_g so long as it is large enough. With $N_{\text{ev}} = 10^4$ integrand evaluations, the accuracy improves from 11% with $N_g = 1$ (i.e., no VEGAS map) to 0.3% at $N_g = 100$ and flattens out at 0.1% around $N_g = 700$.

The VEGAS map is particularly effective for integrals like that in Eq. (26), because the integral over each Gaussian can be separated into a product of one-dimensional integrals over each direction. It also works well, however, for

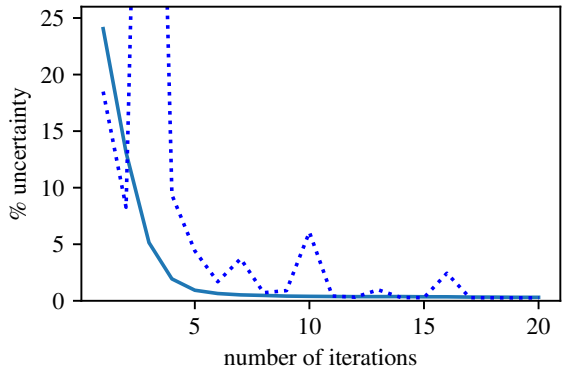

Figure 2: Percent uncertainty (1σ) for each VEGAS iteration in a simple y-space Monte Carlo for the $D = 4$ integral defined in Eqs. (28) and (27). The solid line shows how the uncertainty improves with successive iterations of the refinement process when damping parameter $\alpha = 0.2$; the dotted line shows (unstable) improvement when $\alpha = 1.0$.

other integrands with high peaks that are not separable in this way. The solid line in Fig. 2, for example, shows how the uncertainty in a y-space Monte Carlo is reduced over 20 iterations for the $D = 4$ integral

$$\int_0^1 d^4x \sum_{i=1}^{2} \Theta(|\mathbf{x} - \mathbf{r}_i| < 0.067), \tag{28}$$

where the $\mathbf{r}_i$ are given in Eq. (27) and

$$\Theta(x) = \begin{cases} 1 & \text{if } x = \text{True} \\ 0 & \text{if } x = \text{False.} \end{cases} \tag{29}$$

This integral is harder than the previous one, because the peaks have no shoulders and so are difficult to find — the integrand vanishes over 99.96% of the integration volume. The VEGAS map is nevertheless able to reduce the fractional uncertainty from 24% before adapting to 0.34% after 10–20 iterations, where $N_{\text{ev}} = 10^5$ Monte Carlo samples are used for each iteration. The damping parameter was set lower here, to $\alpha = 0.2$, to ensure smooth adaptation (solid line); setting $\alpha = 1.0$ leads to instability (dotted line).

This last example also illustrates how the VEGAS map can improve integrals over volumes with irregular shapes. Monte Carlo integration works well even with discontinuous integrands and so has no problem with the Θ functions that define the integration volume. The VEGAS map helps find the integration region and concentrate samples in that region. Obviously it is better, where possible, to redefine the integration variables so that the integration region is rectangular.

2.5. Combining and Comparing Iterations

VEGAS, being iterative, generates a series of estimates I_j of the integral (Eq. (3)), each with its own error estimate $\sigma_j \equiv \sigma_{I_j}$ (Eq. (6)). As discussed above, the distribution of I_js for a given VEGAS map is approximately Gaussian when a sufficient number of samples N_{ev} is used (and assuming the integral is square integrable). Then we can combine estimates from separate iterations to obtain a cumulative estimate of the integral and its standard error:

$$\bar{I} = \frac{\sum_j I_j/\sigma_j^2}{\sum_j 1/\sigma_j^2} \tag{30}$$

$$\sigma_{\bar{I}} = \Big(\sum_j \frac{1}{\sigma_j^2}\Big)^{-1/2}. \tag{31}$$

The cumulative estimate is usually superior to the estimates from separate iterations.

It is important to verify that results from different iterations are consistent with each other within the estimated uncertainties. This provides a direct check on the reliability of the error estimates. For example, we can calculate the χ^2 statistic for the estimates:

$$\chi^2 = \sum_j \frac{(I_j - \bar{I})^2}{\sigma_j^2}. \tag{32}$$

We expect χ^2 to be of order the number of iterations (less one) when the I_j are approximately Gaussian, and the estimates of the uncertainties σ_j are reliable. If χ^2 is considerably larger than this, either or both of $\bar{I}$ and $\sigma_{\bar{I}}$ may be unreliable. There are two common causes for χ^2s that are too large.

The first common cause is that early iterations, before the VEGAS map has adapted, may give very poor estimates for the integral and its uncertainty. This happens, for example, when the integrand has high, narrow peaks that are largely missed in early iterations, leading to estimates for the integral and error that are both much too small. A standard remedy is to omit the early iterations from the determinations of $\bar{I}$ and $\sigma_{\bar{I}}$.

The second cause for χ^2s that are too large is that the number of samples $N_{\rm ev}$ is insufficiently large to guarantee Gaussian statistics for the I_j, even after the VEGAS map has fully adapted to the integrand. The threshold for an adequate $N_{\rm ev}$ is highly dependent on the integrand. In practice one finds this threshold through trial and error, by looking to see how large $N_{\rm ev}$ needs to be in order to obtain stable results and a reasonable χ^2. Although the statistical error in $\bar{I}$ can be reduced by increasing either the number of samples $N_{\rm ev}$ or the number of iterations $N_{\rm it}$, it is generally better to increase $N_{\rm ev}$ while keeping $N_{\rm it}$ just large enough to find the optimal VEGAS map and measure a χ^2. $N_{\rm it}$ = 5–20 is usually enough.

There is another, related reason for increasing the number of samples $N_{\rm ev}$ rather than the number of iterations $N_{\rm it}$. While the estimates I_j from individual iterations give unbiased estimates of the integral for any $N_{\rm ev}$, the weighted sum $\bar{I}$ only becomes unbiased when $N_{\rm ev}$ is sufficiently large that non-Gaussian effects are negligible. The leading non-Gaussian effect is the correlation between fluctuations in I_j and σ_j^2 (Eq. (7)). It introduces a bias in the weighted average that vanishes like $1/N_{\rm ev}$ with increasing $N_{\rm ev}$ and so is usually negligible compared to the statistical uncertainty $\sigma_{\bar{I}}$, which vanishes more slowly. For example, the bias in $\bar{I}$ for the second integral discussed above (Eq. (28)) is only about -0.05% when $N_{\rm ev} = 10^5$ (with $\alpha = 0.2$). This is seven times smaller than σ_j, and so is negligible compared to $\sigma_{\bar{I}}$ unless $N_{\rm it} \geq 50$. The bias falls to -0.03% when $N_{\rm ev} = 2 \times 10^5$.

The bias coming from the weighted average can usually be ignored, but it is easy to avoid it completely, if desired. This is done by discarding results from the first 10 or 20 iterations of the VEGAS algorithm (while the grid is adapting), and then preventing the algorithm from adapting in subsequent iterations (by setting damping parameter $\alpha = 0$). The unweighted average of the latter I_js provides an unbiased estimate of the integral, and the unweighted average of the σ_js divided by $\sqrt{N_{\rm it}}$ gives an estimate of the statistical uncertainty. This technique is useful when the number of samples $N_{\rm ev}$ is small, leading to large fluctuations from iteration to iteration in the uncertainties σ_j.

3. Adaptive Stratified Sampling

The y-space (Eq. (23)) integrals in the previous section were estimated using Simple Monte Carlo integration. The VEGAS implementation currently in wide use ("classic VEGAS") improves on this by using stratified Monte Carlo sampling in y-space rather than Simple Monte Carlo [1]. Given $N_{\rm ev}$ samples per iteration, the algorithm divides each y^μ axis ($\mu = 1 \ldots D$) into

$$N_{\rm st} = \text{floor}((N_{\rm ev}/2)^{1/D}), \tag{33}$$

stratifications of width

$$\Delta y_{\rm st} = 1/N_{\rm st}. \tag{34}$$

This divides divide y-space into $N_{\rm st}^D$ hypercubes, each with volume $\Delta y_{\rm st}^D$. The full integral and its variance are the sums of contributions from each hypercube h:

$$\begin{aligned} I &= \sum_h \Delta I_h \approx I_{\rm MC} \\ \sigma_I^2 &= \sum_h \sigma_{\Delta I_h}^2 \approx \sigma_{\rm MC}^2. \end{aligned} \tag{35}$$

Simple Monte Carlo estimates are made for ΔI_h (c.f., Eq. (3)) and $\sigma^2_{\Delta I_h}$ (c.f., Eq. (5)) to obtain estimates $I_{\rm MC}$ and $\sigma^2_{\rm MC}$ for the total integral and its variance, respectively. The number of integrand samples used is

$$n_{\rm ev} \equiv \text{floor}(N_{\rm ev}/N_{\rm st}^D) \geq 2 \tag{36}$$

per hypercube. The standard deviation for a stratified Monte Carlo estimate typically falls with increasing $N_{\rm st}$, potentially as quickly as $1/N_{\rm st}^D \propto 1/N_{\rm ev}$ [14].

We can improve on the stratification strategy used by classic VEGAS by allowing the number of integrand samples n_h used in each hypercube h to vary from hypercube to hypercube. Then the variance in the Monte Carlo estimate for the integral is

$$\sigma_I^2 = \sum_h \frac{\sigma_h^2(Jf)}{n_h} \tag{37}$$

where

$$\begin{aligned}\sigma_h^2(Jf) \equiv \Omega_h \int_{\Omega_h} d^D y\, (J(y) f(x(y)))^2 \\ - \Big(\int_{\Omega_h} d^D y\, J(y) f(x(y)) \Big)^2\end{aligned} \tag{38}$$

and Ω_h is the hypercube's volume in y-space. Varying the n_h independently, constrained by

$$\sum_h n_h = N_{\rm ev}, \tag{39}$$

it is easy to show that σ_I^2 is minimized when

$$n_h \propto \sigma_h(Jf). \tag{40}$$

The innovation in the new VEGAS ("VEGAS+") is to redistribute integrand samples across the hypercubes, according to Eq. (40), after each iteration. The $\sigma_h(Jf)$ are estimated in an iteration using the integrand samples used to estimate the integral. In this way the distribution of integrand samples across hypercubes is optimized over several iterations, at the same time as the VEGAS map is optimized.

In detail, the algorithm for reallocating samples across hypercubes in VEGAS+ is as follows:

1. Choose a somewhat smaller number of stratifications so there are enough samples to allow for significant variation in n_h:

   $$N_{\rm st} = \text{floor}((N_{\rm ev}/4)^{1/D}). \tag{41}$$

   Usually the number stratifications $N_{\rm st}$ is much smaller than the number of increments N_g used in the VEGAS map. The algorithm is slightly more stable if N_g is an integer multiple of $N_{\rm st}$ (or vice versa if $N_{\rm st}$ is larger).
2. During each iteration accumulate estimates

   $$\begin{aligned}\sigma_h^2(Jf) \approx \frac{\Omega_h^2}{n_h} \sum_{y\in\Omega_h} (J(y) f(x(y)))^2 \\ - \Big(\frac{\Omega_h}{n_h} \sum_{y\in\Omega_h} J(y) f(x(y))\Big)^2,\end{aligned} \tag{42}$$

   for each hypercube using the same samples used to estimate the integral.
3. Introduce a damping parameter $\beta \geq 0$ by replacing $\sigma_h(Jf)$ with

   $$d_h \equiv (\sigma_j(JF))^\beta. \tag{43}$$

   Choosing $\beta = 1$ corresponds to the optimal distribution (Eq. (40)), but a somewhat smaller value can help avoid overreaction to random fluctuations. Setting $\beta = 0$ results in n_h values that are all the same — the stratified sampling becomes non-adaptive, as in classic VEGAS. We use $\beta = 0.75$ for the examples in this paper.

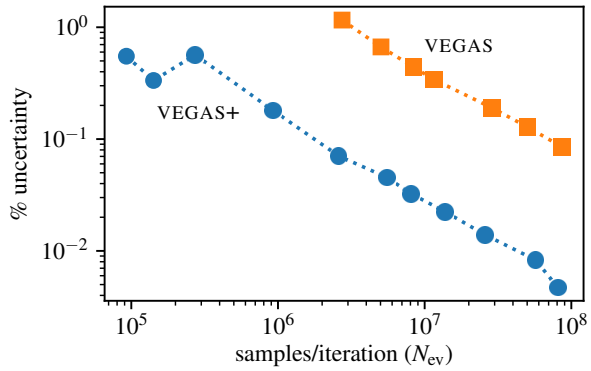

Figure 3: Percent uncertainty (1σ) in estimates of the integral in Eq. (45) from 30 iterations of classic VEGAS (top) and VEGAS+ (bottom) is plotted versus the number $N_{\rm ev}$ of integrand evaluations (samples) used per iteration. Integral estimates from the first ten iterations are ignored in each case. Damping parameter $\alpha = 0.15$ in both cases. Damping parameter $\beta = 0$ for classic VEGAS; $\beta = 0.75$ for VEGAS+. Classic VEGAS becomes unstable below $N_{\rm ev} = 3 \times 10^6$; VEGAS+ is unstable below 1×10^5. The number $N_{\rm st}$ of stratifications per axis used by VEGAS+ varies from 3 to 8 over this range of sample sizes $N_{\rm ev}$.

4. Recalculate the number of samples for each hypercube,

$$n_h = \max(2, N_{\rm ev} d_h / \sum_{h'} d_{h'}), \tag{44}$$

for use in the next iteration. Alternatively, n_h can be set to $N_{\rm ev} d_h / \sum d_{h'} + 2$ which uses more samples but might be more stable. The point of this step is to distribute samples across the hypercubes according to Eq. (40), while guaranteeing at least 2 samples per hypercube (to allow error estimates).

The VEGAS map is also updated after each iteration, as described in Section 2.3. The allocation of integrand samples converges rapidly once the VEGAS map has converged. The optimal VEGAS map for an integrand is independent of the allocation of samples, but reallocating samples according to Eq. (40) can significantly speed the discovery of that optimum because there is better information about the integrand earlier on. The independence of the optimal VEGAS map from the sample allocation improves the algorithm's stability — random fluctuations in the sample allocation are unlikely to trigger big changes in the VEGAS map.

We mention parenthetically that the original Fortran implementation classic VEGAS switched to a different form of adaptive stratified sampling than described here when working in low dimensions with lots of samples per iteration. This algorithm replaces the VEGAS map with a similar grid that stratifies x space, but with stratifications concenterated where the uncertainties are largest (rather than where the function is largest); see the appendix of Ref. [1] for more details. This algorithm can outperform the classic VEGAS algorithm in very low dimensions. For example, it is about three times more accurate for the two-dimensional analogue of the integral in Eq. (45) (next section) with 400,000 samples per iteration. The adaptive stratified sampling technique described above, however, is also about three times more accurate than classic VEGAS for that integral. Typically VEGAS+ does not need this other approach even in low dimensions.

3.1. Diagonal Structure

Adaptive stratified sampling as described in the previous section provides little or no improvement over classic VEGAS for integrals like those in the previous sections, where the Jacobian from the VEGAS map can flatten the integrand's peaks almost completely and spread them out to fill y-space. It is easy, however, to create integrals for which the new adaptive stratified sampling makes a big difference.

Consider, for example, the eight-dimensional integral

$$\int_0^1 d^8x \sum_{i=1}^3 \mathrm{e}^{-50\,|\mathbf{x}-\mathbf{r}_i|} \tag{45}$$

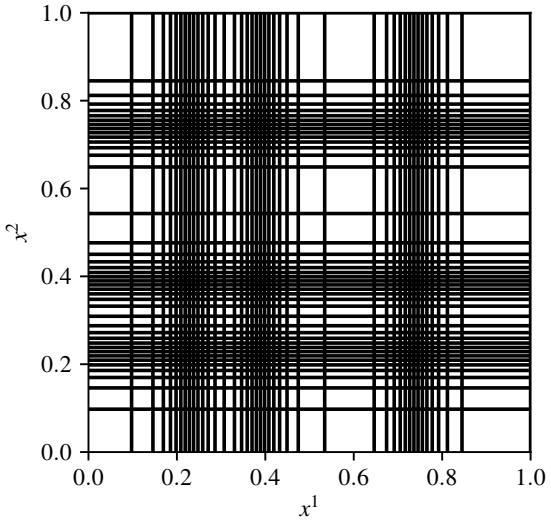

Figure 4: VEGAS map for the integral in Eq. (45). Every 33rd grid line is drawn for axes x^1 and x^2; grids for the other axes are the same.

whose integrand has three narrow peaks along the diagonal of the integration volume, at

$$\begin{aligned}
\mathbf{r}_1 &= (0.23, 0.23, 0.23, 0.23, 0.23, 0.23, 0.23, 0.23) \\
\mathbf{r}_2 &= (0.39, 0.39, 0.39, 0.39, 0.39, 0.39, 0.39, 0.39) \\
\mathbf{r}_3 &= (0.74, 0.74, 0.74, 0.74, 0.74, 0.74, 0.74, 0.74).
\end{aligned} \tag{46}$$

The locations along the diagonal were chosen randomly. Unlike Gaussians, these integrands cannot be factored into a product of separate functions for each direction.

Fig. 3 shows that the uncertainties in the integral estimates from classic VEGAS are 14–19× larger than those from VEGAS+, using the same number of integrand samples N_{ev} per iteration. Measured from $N_{\text{ev}} = 10^6$, the uncertainty generated by N_{it} iterations of the new algorithm falls roughly like

$$\sigma_I \propto \frac{1}{\sqrt{N_{\text{it}}}\, N_{\text{ev}}^{0.9}} \tag{47}$$

with increasing N_{it} and N_{ev} — as expected, a larger N_{ev} is more valuable than a larger N_{it} for a given cost $N_{\text{it}} \times N_{\text{ev}}$. VEGAS+ gives reliable results down to $N_{\text{ev}} = 10^5$; classic VEGAS is unusable below $N_{\text{ev}} = 3 \times 10^6$, where it typically misses out one or more of the three peaks.

What makes this integral challenging for classic VEGAS is the diagonal structure of the integrand. An axis-oriented adaptation strategy, like that used by classic VEGAS, will generally have difficulty handling large structures aligned along diagonals.

The problem for classic VEGAS is obvious from pictures of the optimal VEGAS map for this integrand (Fig. 4). This grid concentrates integrand samples at the three peaks on the diagonal, but also at $3^8 - 3 = 6558$ additional points, where the integrand is very small:

$$\begin{aligned}
&\mathbf{r} = (0.39, 0.23, 0.23, \ldots) \\
&\mathbf{r} = (0.74, 0.23, 0.23, \ldots) \\
&\mathbf{r} = (0.23, 0.39, 0.23, \ldots) \\
&\mathbf{r} = (0.39, 0.39, 0.23, \ldots) \\
&\ldots
\end{aligned} \tag{48}$$

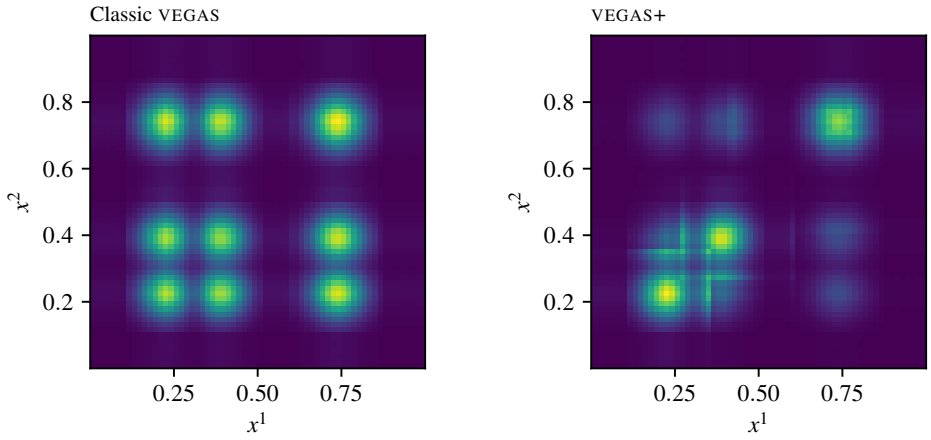

Figure 5: Histograms showing the distribution of $N_{\rm ev} \approx 10^8$ integrand evaluations across the x^1-x^2 plane using classic VEGAS (left) and VEGAS+ (right). The distributions are the same in other directions.

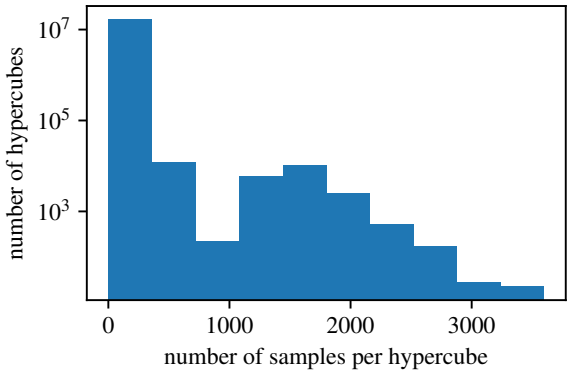

Figure 6: Distribution of integrand samples across hypercubes used by VEGAS+ when evaluating the integral in Eq. (45). There are 8^8 hypercubes and $N_{\rm ev} \approx 10^8$ samples in all.

Integrand samples at these phantom peaks are wasted, greatly reducing the effective number of samples. The adaptive stratification used by VEGAS+ can transfer integration points from the phantom peaks to the real peaks, leading to a much larger effective $N_{\rm ev}$. This is evident from the histograms in Fig. 5 which compare the spatial distributions of $N_{\rm ev} = 10^8$ integrand samples using classic VEGAS (left) and VEGAS+ (right). Classic VEGAS gives equal attention to peaks and phantoms, while VEGAS+ focuses mostly on the peaks. Fig. 6 shows how the samples are distributed across the 8^8 hypercubes used by VEGAS+; more than half of the hypercubes have only 2 samples. Classic VEGAS uses 2 samples in each of 9^8 hypercubes.

Another example is the integral

$$\int_{-1}^{1} d^4x\, \mathrm{e}^{-\mathbf{x}^T H^{-1}\mathbf{x}/4}, \tag{49}$$

where H is the 4×4 Hilbert matrix.[4] This integrand has a sharp ridge along an oblique axis (Fig. 7). Classic VEGAS

[4]The $N \times N$ Hilbert matrix has elements $H_{\mu\nu} = 1/(\mu + \nu - 1)$ for $\mu, \nu = 1 \ldots N$. It is famously ill-conditioned.

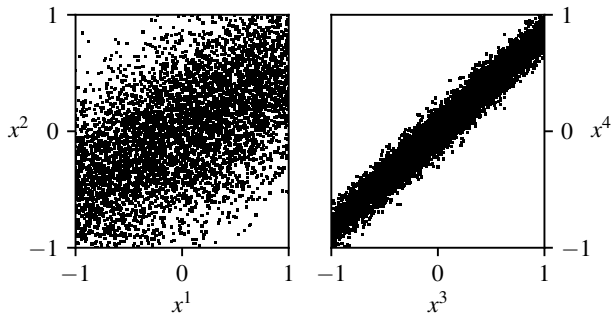

Figure 7: Two views of 10,000 random points distributed with density proportional to the integrand of Eq. (49). One view is projected onto the x^1, x^2 plane (left) and the other onto the x^3, x^4 plane (right). Correlation coefficients for x^1, x^2 and x^3, x^4 are 0.866 and 0.986, respectively.

obtains a 0.25% accurate result from the last five of seven iterations with $N_{\rm ev} = 4 \times 10^5$, while VEGAS+ is about 3× more accurate.

3.2. Limitations and a Variation

The chief limitation of VEGAS+'s adaptive stratified sampling is that it requires at least $N_{\rm st} = 2$ stratifications per direction to have any effect on results. From Eq. (41), this requires

$$N_{\rm ev} \geq 4N_{\rm st}^D \geq 2^{D+2} \tag{50}$$

integrand evaluations per iteration. While this is not much of a restriction for dimension $D = 5$ or 6, adaptive stratified sampling only turns on when $N_{\rm ev} \geq 1.3 \times 10^8$ for $D = 25$. This is still manageable but in practice there will be less and less difference between VEGAS+ and classic VEGAS for higher dimensions.

In some situations it is possible to circumvent this restriction, at least partially, by using different numbers $N_{\rm st}^\mu$ of stratifications in different directions μ. Most integrands have more structure in some directions than in others. Concentrating stratifications in those directions, with fewer stratifications or none in other directions, allows VEGAS+ to use stratified sampling with smaller values of $N_{\rm ev}$. We give an example (with $D = 21$) at the end of Appendix B.

Using a mixed set of stratifications rather than a uniform set can be useful in high dimensions even when the integrand does not have structure concentrated in a lower-dimensional subspace. This is illustrated in Fig. 8(a) where we compare estimates of the integral

$$\int_0^1 d^D x \; \mathrm{e}^{-50\,|\mathbf{x}|} \tag{51}$$

for various dimensions up to $D = 50$ using two different strategies for stratification. The uniform stratification uses the same number of stratifications in every direction, with the number $N_{\rm st}$ determined from Eq. (41) where $N_{\rm ev} = 2.5{\times}10^5$. This value for $N_{\rm st}$ is the largest that allows for an average of 4 samples per hypercube given at most $N_{\rm ev}$ samples per iteration. The mixed stratification uses $N_{\rm st} + 1$ stratifications for the first d directions and $N_{\rm st}$ stratifications otherwise, where again the value for d is the largest that allows for 4 samples per hypercube. The values for d and $N_{\rm st}$ vary with dimension, but $d = 15$ and $N_{\rm st} = 1$ for dimension $D > 15$. So the uniform stratification has only a single hypercube for $D > 15$ and adaptive stratified sampling can not be used. The mixed stratification, on the other hand, has 2^{15} hypercubes for all $D > 15$.

The mixed stratification is significantly more accurate for large dimensions $D > 15$, and continues working all the way out to $D = 50$, while the uniform stratification fails to give useful results above $D = 30$. The difference is mostly because the VEGAS map does not have enough iterations to fully adapt in high dimensions. With the mixed stratification, adaptive stratified sampling is still functioning to some considerable extent above $D = 15$ and so can help the VEGAS map handle the sharp peak at the origin.

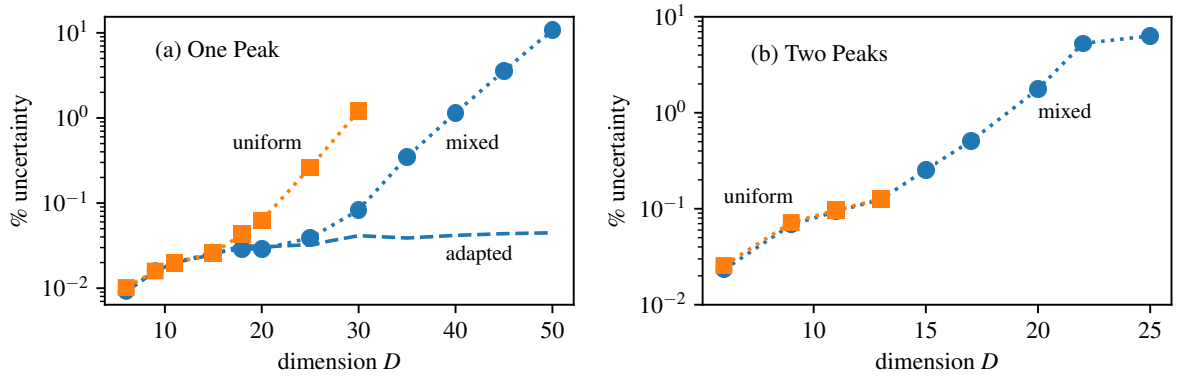

Figure 8: (a) Percent uncertainty (1σ) in estimates of the integral in Eq. (51) from the last 25 of 50 iterations of VEGAS+ with a uniform stratification (top) and a mixed stratification (bottom) plotted versions the dimension. The damping parameters are $\alpha = 0.1$ and $\beta = 0.75$ for the first 25 iterations, while both are set to zero for the remaining iterations (to guarantee unbiased estimates at high dimensions). VEGAS+ is limited to at most 250,000 samples per iteration. The uniform stratification gives unreliable results for $D > 30$. The dashed line shows results from the mixed stratification when the VEGAS map is fully adapted. (b) Percent uncertainty for the integral in Eq. (52) with one fifth as many samples per iteration and five times as many iterations. The uniform stratification gives unreliable results for $D > 13$.

More iterations for adaptation would reduce the uncertainties at large dimension D for both curves in Fig. 8(a) (see the dashed line), since VEGAS maps remain effective for arbitrarily large dimensions once they have converged. More iterations would have little effect, however, on the results in Fig. 8(b) which use one fifth as many samples per iteration but five times as many iterations (ensuring convergence) to estimate the integral

$$\int_0^1 d^D x \left(e^{-4\sum_\mu (x^\mu)^2} + e^{-4\sum_\mu (x^\mu - 1)^2} \right). \tag{52}$$

This integrand has peaks at opposite ends of the integration volume's diagonal. The peaks are fairly broad in low dimensions but become increasingly hard for VEGAS+ to discover as the dimension increases. The integrator with the uniform stratification stops working abruptly at $D = 14$ where it goes from having 2^{13} hypercubes at $D = 13$ to only a single hypercube. It is then unable to find both peaks. With the mixed stratification, adaptive stratified sampling can continue beyond $D = 13$, with 2^{13} hypercubes for all $D > 13$. This partial stratification is enough to stabilize the adaptation of the VEGAS map so that neither peak is lost until much higher dimensions.

The uncertainties in second example (Fig. 8(b)) grow quickly with increasing dimension despite the fully adapted VEGAS map. This is because of the exponential growth (2^{D-2}) in the number of phantom peaks resulting from the diagonal structure of the integrand (see previous section). Mixed stratification allows adaptive stratified sampling to deal with phantom peaks in 13 directions, which helps but still leaves 2^{D-15} phantoms wasting integrand samples. The first example (Fig. 8(a)) has only a single peak and therefore no phantoms; its uncertainties grow very slowly with D once the VEGAS map is fully developed (dashed line).

The improvements shown here from mixed stratification are unusually large; for many applications there is little difference between the two strategies. It is probably useful, nevertheless, to use the mixed strategy as the default rather than the uniform strategy described in Section 3 (and used elsewhere in this paper).

4. VEGAS+ Hybrids

In this section we discuss how VEGAS+ can be combined with other algorithms to make hybrid integrators. We look at two strategies. In the first, we use several iterations of VEGAS+ to generate a VEGAS map $x(y)$ that is optimized for the integrand. We then used a different (adaptive) algorithm to evaluate the y-space integral Eq. (23). This strategy, in effect, replaces VEGAS+'s adaptive stratified sampling with the other algorithm. We illustrate it in Sec. 4.1 by combining VEGAS+ with the widely available MISER algorithm, and a variation on that algorithm, MISER+.

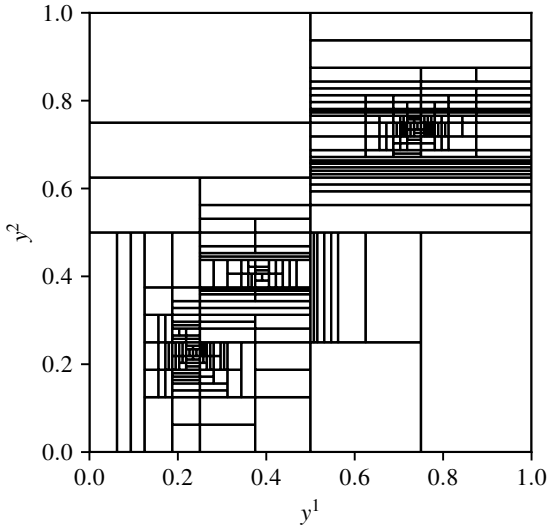

Figure 9: A partition of the integration volume generated by MISER for the $D = 2$ dimensional version of integral Eq. (45). MISER used 3×10^4 samples to generate this partition and estimate the integral. There are 271 sub-volumes.

The second strategy employs a Markov Chain Monte Carlo (MCMC) or other peak-finding algorithm to generate a set of samples $\{x, f(x)\}$ of the integrand. These are used to precondition the VEGAS map before integrating. The sample points x need to cover important regions of the integration volume, but otherwise are unrestricted. The preconditioning makes it easier for VEGAS+ to discover the important regions. In Sec. 4.2 we illustrate this approach with integrands having multiple, narrow peaks. We also compare preconditioned VEGAS+ with a new algorithm optimized for this strategy.

4.1. MISER *and* MISER+

For our first example of a hybrid integrator, we combine VEGAS+ with the MISER algorithm [15]. MISER uses adaptive stratified sampling where the integration volume is recursively partitioned into a large number of rectangular sub-volumes (Fig. 9) to increase the accuracy of the integral's estimate. MISER appears to be more effective for integrands with peaks aligned along diagonals than it is for peaks aligned parallel to integration axes. This is opposite from VEGAS maps, suggesting that the combination might be particularly effective.

In the following examples, we compare VEGAS+ with MISER and with a VEGAS-MISER hybrid. In each case we run VEGAS+ with various values for the number $N_{\rm ev}$ of samples per iteration, and 15 iterations, discarding results from the first 5. We use default values for the damping parameters: $\alpha = 0.5$ and $\beta = 0.75$. To compare, we run MISER with $15N_{\rm ev}$ samples. The VEGAS-MISER hybrid uses 5 iterations of VEGAS+ to develop a VEGAS map $x(y)$ and then estimates the y-space integral Eq. (23) using MISER with $10N_{\rm ev}$ integrand samples.

We also compare these algorithms with a variation on MISER, which we call MISER+. The procedure for MISER+ is:

1. Use MISER with half the sample points to generate and save a partition of the integration space optimized for the integrand $f(x)$. Also save MISER's estimate for
$$\sigma_i^2(f) = \Omega_i \int_{\Omega_i} d^D x\, f^2(x) - \Big(\int_{\Omega_i} d^D x\, f(x) \Big)^2 \tag{53}$$
in each sub-volume Ω_i. Ignore MISER's estimate for the integral.
2. Distribute the remaining half of the integrand samples across the sub-volumes so that the number n_i of samples in each sub-volume is proportional to $\sigma_i(f)$, using the procedure described for VEGAS+ (Eqs. (43 and (44)).

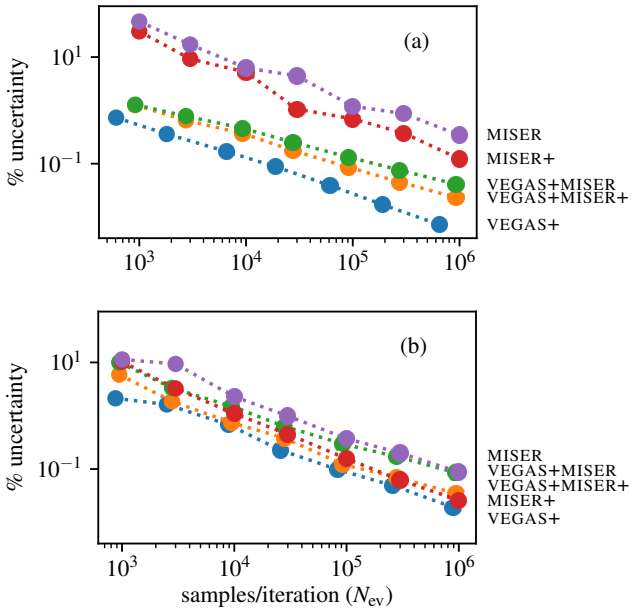

Figure 10: Percent uncertainty (1σ) in estimates of the integral Eq. (54) with peaks specified by a) Eq. (55), and b) Eq. (56). Results are shown for the last 10 of 15 iterations of VEGAS+, with N_{ev} integrand evaluations (samples) per iteration. Corresponding results from MISER and MISER+ use $15N_{\text{ev}}$ samples. The VEGAS hybrids with with MISER and MISER+ use 5 iterations of VEGAS+ to generate a VEGAS map, and then estimate the integral using $10N_{\text{ev}}$ samples with MISER/MISER+.

3. Estimate the integral in each sub-volume of the partition using Simple Monte Carlo with the number of sample points allocated in the previous step. Add the estimates from each sub-volume to obtain an estimate for the total integral (as in Eq. (35)).

The relation between MISER+ and MISER is similar to that between VEGAS+ and classic VEGAS. MISER+ can also be combined with VEGAS maps, as described above.

We did detailed comparisons for two different integrals. The first is the $D = 4$ dimensional integral

$$\int_0^1 d^4x \sum_{i=1}^{3} \mathrm{e}^{-50\,|\mathbf{x}-\mathbf{r}_i|} \tag{54}$$

with peaks aligned parallel to the x^1 axis:

$$\begin{aligned} \mathbf{r}_1 &= (0.23, 0.5, 0.5, 0.5) \\ \mathbf{r}_2 &= (0.39, 0.5, 0.5, 0.5) \\ \mathbf{r}_3 &= (0.74, 0.5, 0.5, 0.5) \end{aligned} \tag{55}$$

The uncertainties in the integral estimates generated by each algorithm for different values of N_{ev} are shown in Fig. 10(a). MISER is 50–100× less accurate than VEGAS+. MISER is also 1.2–2.7× less accurate than MISER+. VEGAS maps are well suited to this integrand, so both MISER and MISER+ see big improvements when used with a VEGAS map. VEGAS+ combined with MISER+ is 7–38× more accurate than MISER+ alone, and only 2–4× less accurate than VEGAS+ by itself.

For the second comparison (Fig. 10(b)), we use the same integral but with the peaks aligned along the diagonal of

the integration volume:

$$\begin{aligned}\mathbf{r}_1 &= (0.23, 0.23, 0.23, 0.23)\\ \mathbf{r}_2 &= (0.39, 0.39, 0.39, 0.39)\\ \mathbf{r}_3 &= (0.74, 0.74, 0.74, 0.74)\end{aligned} \tag{56}$$

Both MISER+ and MISER are significantly more accurate (4–5×) for this diagonal structure than for the axis-aligned structure of the previous integrand. As discussed above, VEGAS maps are less effective for diagonal structures, but the combination of VEGAS+ with MISER+ remains competitive with MISER+ alone. VEGAS+ by itself is 4–6× more accurate than MISER, and 1.5–6× more accurate than MISER+.

We also checked the $D = 8$ dimensional versions of these integrals. For the integrand with axis-aligned peaks, VEGAS+ starts working reliably with 500–1000× fewer integrand samples than MISER or MISER+. Using the last 20 of 30 iterations, with $\alpha = 0.15$ and $N_{\text{ev}} = 10^6$ samples per iteration, VEGAS+ gives 0.03%-accurate estimates of the integral. VEGAS-assisted MISER+ is 3× less accurate, while MISER+ and MISER by themselves are both more than 500× less accurate for a similar number of integrand samples.

The differences are smaller with peaks aligned along the $D = 8$ diagonal because both MISER and MISER+ prefer the diagonal structure. MISER+ is approximately 5× more accurate than MISER, and only 2–3× less accurate than VEGAS+ when N_{ev} is between 5×10^5 and 10^7. Using a VEGAS map with MISER+ gives results that vary in precision between MISER+ and VEGAS+, depending upon N_{ev}. Using VEGAS+ with MISER improves on MISER but is not as accurate as MISER+.

These experiments suggest that combining VEGAS+ with either MISER or MISER+ is a good idea. Where VEGAS maps are effective, the combination can be much more accurate than MISER or MISER+ separately. Where VEGAS maps are less effective, they do not appreciably degrade results compared to the separate algorithms. Typically MISER+ outperforms MISER by factors of 2–5 and so might be the preferred option in combination with VEGAS+. VEGAS+ outperforms all of the other algorithms in all of these tests.

4.2. Preconditioned Integrators

Ref. [18] suggests a different approach to multidimensional integration. They assume that the integrand $f(x)$ has been sampled prior to integration, using MCMC or some other technique. A sample consists of some large number (thousands) of integrand values $\{f(x)\}$ at points $\{x\}$ that are concentrated in regions important to the integral. The samples are used to design an integrator that is customized (preconditioned) for the integrand. The authors describe an algorithm for doing this, but this strategy is also easily implemented using VEGAS+.

To illustrate the approach with VEGAS+, we consider a dimension $D = 8$ integral whose integrand has three very sharp peaks arrayed along the diagonal of the integration volume:

$$\int_0^1 d^8x \sum_{i=1}^{3} \mathrm{e}^{-10^4(\mathbf{x}-\mathbf{r}_i)^2}, \tag{57}$$

where the $\mathbf{r}_i$ are given in Eq. (46). In this case it is easy to generate random sample points whose density is proportional to the integrand. We use 3000 sample points $\{x\}$. The VEGAS map used by VEGAS+ can be trained on the sample data $\{x, f(x)\}$ before integrating. This is done using the method described in Secs. 2.3 and 2.4, but with

$$d_i^\mu \equiv \frac{1}{n_i^\mu} \sum_{x \in \Delta x_i^\mu} J^2(y(x))\, f^2(x), \tag{58}$$

where the sum is over the sample data, n_i^μ is the number of sample data points falling in increment Δx_i^μ, and $y(x)$ is the inverse of the VEGAS map. The VEGAS map converges quickly as the algorithm is iterated, with the same sample data being reused for each iteration. Here we iterate the algorithm 10 times.

Starting with the preconditioned map, we then run VEGAS+ as usual for $N_{\text{it}} = 8$ iterations, with damping parameter $\alpha = 0$ to prevent further adjustment of the VEGAS map. The 8 iterations allow the stratified sampling algorithm to adapt to whatever structure has not been dealt with by the VEGAS map. As we increase the number N_{ev} of integrand

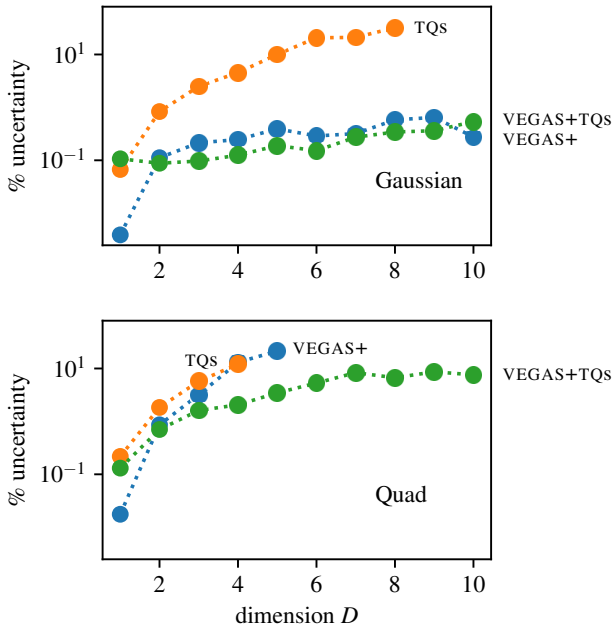

Figure 11: Percent uncertainty (1σ) in estimates of two integrals from Ref. [18] for dimensions D = 1–10. Results are shown for the VEGAS+ (blue) and TQs (orange) algorithms, as well as for a hybrid that combines the VEGAS map with TQs (green). All algorithms were limited to 12,000 integrand evaluations in all. The uncertainties are inferred from the interquartile range of the results from 50 repetitions of each integration.

evaluations per iteration, we find that VEGAS+ starts to give good results when $N_{\rm ev} \approx 10^4$ to 10^5. By $N_{\rm ev} = 10^6$, it is giving 1%-accurate results. VEGAS+ without the preconditioning is unable to find all three peaks reliably until $N_{\rm ev} \approx 10^8$ (and classic VEGAS needs many more evaluations).

We expect a large benefit from preconditioning for extreme multimodal problems like this one. The exact distribution of the sample points $\{x\}$ is not crucial — for example, we get more or less the same results above using points drawn from Gaussians that are twice as wide as in the integrand. What matters is that there are enough samples around all of the peaks. In more general problems, we usually do not know *a priori* where the peaks are or how many there are. Peak-finding algorithms or MCMC might be helpful in such cases. As emphasized in Ref. [18], using preconditioning in this way separates the challenge of finding the integrand's peaks from the challenge of accurate integration once they have been found, allowing us to use different algorithms for the two different tasks, each algorithm optimized for its task.

We compared preconditioned VEGAS+ with one of the algorithms (TQs) from Ref. [18] that is designed for preconditioning. Like MISER, TQs recursively subdivides the integration volume into a large number of sub-volumes (c.f., Fig. 9); but TQs bases this partitioning on the sample $\{x, f(x)\}$ available before integrating. For our comparison we look at two integrals discussed in Ref. [18]: one has a single narrow Gaussian (same width as in Eq. (26)) at the center of a 2×2 hypercube; the other has four narrow Gaussians (same width as in Eq. (26)) spread evenly along the diagonal of a 10×10 hypercube. The paper labels these problems "Gaussian" and "Quad", respectively. These are the easiest and hardest integrals considered there. We examined results for dimension D = 1–10.

Following Ref. [18], we limit each algorithm to approximately 12,000 integrand evaluations for these comparisons. The TQs algorithm uses half of those samples to divide the integration volume into 2000 sub-volumes. The integral is estimated by doing 3-point Simple Monte Carlo integrals over each sub-volume and summing the results.

VEGAS+ needs far fewer samples to optimize the VEGAS map because optimizing the map for each direction is a separate one-dimensional problem that utilizes all of the data. Here we use 1000 samples to create the VEGAS map. The remaining 11,000 samples are allocated across 4 iterations of VEGAS+, again with damping parameter $\alpha = 0$.

Our results are in Fig. 11. VEGAS+ outperforms TQs by more than an order of magnitude on the Gaussian problem

in high dimensions, with only modest growth in the errors as the dimension D increases (the VEGAS+ error is still only 7% by $D = 50$, for example). Not surprisingly, results from the two algorithms are much closer for the Quad problem. Both algorithms become unreliable for dimensions D greater than three or four — 12,000 integrand samples are too few for higher dimensions.

Fig. 11 also shows results for a hybrid approach that combines a VEGAS map with TQs. In the hybrid approach, half of the integrand evaluations are used to create an optimized VEGAS map, as outlined above. Those same samples are then re-used by TQs to partition the integration volume to optimize the y-space integral Eq. (23) with the optimized VEGAS map $x(y)$. The y-space integral is then calculated summing Simple Monte Carlo estimates of the contributions from each sub-volume in the TQs partition. The VEGAS+TQs hybrid gives similar results to VEGAS+ for the Gaussian problem, but outperforms both of the other algorithms on the Quad problem. In particular the hybrid algorithm continues to give usable results even out to dimension $D = 9$–10.

These problems shows how the VEGAS map is effective for dealing with isolated peaks, even when these are arranged along a diagonal of the integration volume. This is because it is able to flatten the peaks. It can't expand the peaks to fill y-space when there are multiple peaks along the diagonal, so it is important to combine the map with algorithms, like VEGAS+ and TQs, that can target sub-regions within the y-space integration volume.

There are problems, of course, where the VEGAS map is of limited use. The Hilbert-matrix Gaussian in Eq. (49) is an example. For this problem, neither TQs nor the VEGAS+TQs hybrid can achieve errors smaller than 50% with only 12,000 samples when $D \geq 3$; VEGAS+ gives errors smaller than 1% for $D = 3$.

5. Conclusions

In this paper we have demonstrated how to combine adaptive stratified sampling with classic VEGAS's adaptive importance sampling in a new algorithm, VEGAS+. The adaptive stratified sampling makes VEGAS+ far more effective than VEGAS for dealing with integrands that have multiple peaks or other structure aligned along diagonals of the integration volume. The added computational cost is negligible compared to the cost of evaluating the integrand. VEGAS+ was 2–19× more accurate than classic VEGAS in our various examples, with errors that typically fell much faster than $1/\sqrt{N_{\rm ev}}$ when increasing the number $N_{\rm ev}$ of integrand evaluations.

In Sec. 4, we showed how to combine VEGAS+ with other algorithms, in effect replacing its adaptive stratified sampling with the other adaptive algorithm. Our experiments with the MISER and TQs algorithms show that such hybrids can be significantly more accurate than the original algorithms. It would be worthwhile to explore these and other options further.

We also showed (Sec. 4.2) how to precondition VEGAS+ using integrand samples generated separately from the integrator (e.g., by an MCMC algorithm). Preconditioning can help stabilize VEGAS+ and improve precision, especially for integrands with multiple narrow peaks. In one example, VEGAS+ without preconditioning required more than 100× as many integrand evaluations as preconditioned VEGAS+ before it began giving reliable results. This again is an area deserving further exploration.

Finally we compared VEGAS+ and MCMC for two Bayesian analyses in Appendix B, one with $D = 3$ parameters and the other with $D = 21$. VEGAS+ was more than 10× as efficient in both cases. There we discuss why we expect VEGAS+ will often outperform MCMC for small and moderate sized problems.

Acknowledgments

We thank T. Kinoshita for sharing his code for the $10^{\rm th}$-order QED correction discussed in the first appendix. This work was supported by the National Science Foundation.

Appendix A. Sums and Feynman Diagrams

VEGAS+ can be used for adaptive multi-dimensional summation, as well as integration. To illustrate, we show how to use VEGAS+ to correct for the finite space-time volume used in lattice QCD simulations. Such corrections are

usually calculated in chiral perturbation theory. A typical example is the contribution from a pion tadpole diagram, which in infinite volume is proportional to the integral (in Euclidean space)

$$I_\pi \equiv \int_{-\infty}^{\infty} d^4k\, f(k), \tag{A.1}$$

where

$$f(k) \equiv 1/(k^2 + m_\pi^2)^2 \tag{A.2}$$

with $m_\pi = 0.135\,\text{GeV}$. This becomes an infinite, 4-dimensional sum,

$$S_\pi \equiv (\Delta k)^4 \sum_{k_n} f(k_n), \tag{A.3}$$

when the theory is confined to a box of side L. Here $k_n^\mu = n^\mu\,\Delta k$ with $n^\mu = 0, \pm1, \pm2\ldots$ and

$$\Delta k = 2\pi/L. \tag{A.4}$$

We take $L = 5\,\text{fm}$. The sum is easily converted to an integral:

$$S_\pi = \int_{-\infty}^{\infty} d^4k\, f(\bar{k}(k)) \tag{A.5}$$

where

$$\bar{k}^\mu(k) \equiv \text{round}(k^\mu/\Delta k)\,\Delta k \tag{A.6}$$

and round(x) is the nearest integer to x. Then the needed correction can be written,

$$I_\pi - S_\pi = 16 \int_0^{\infty} d^4k\,[f(k) - f(\bar{k}(k))] \tag{A.7}$$

$$= 16 \int_0^1 \frac{m_\pi^4\, d^4z}{\prod_\mu (1-z^\mu)^2}\,[f(k) - f(\bar{k}(k))] \tag{A.8}$$

where $k^\mu = m_\pi z^\mu/(1-z^\mu)$. This integral is ultraviolet finite, unlike the original integrals.

This integral is easy for VEGAS+. Using the last 10 of 15 iterations, each with $N_{\text{ev}} = 10^3$ samples, gives results with a 7.5% uncertainty, which is accurate enough for most practical applications. Here damping parameter $\alpha = 0.5$. Setting $N_{\text{ev}} = 10^5$ gives 0.5% errors. Classic VEGAS gives 1.0% for the same N_{ev}.

We also tested VEGAS+ on a 10^{th}-order QED contribution to the muon's magnetic moment. We examined the contributions from the light-by-light diagrams with two vacuum polarization insertions (diagrams VI(a) in Fig. 6 and Table IX of Ref. [19]), using a (Fortran) code for the integrand provided by T. Kinoshita. The integration is over $D = 9$ Feynman parameters. We studied two cases: one where all the fermion-loop particles are muons and the other where they are electrons. The latter has large factors of $\log(m_\mu/m_e)$. We found that VEGAS+ was 3–4× more accurate than classic VEGAS in both cases, yielding uncertainties smaller than 0.01–0.02%, when $N_{\text{ev}} \approx 10^8$.

Appendix B. Bayesian Curve Fitting

VEGAS+'s ability to find and target narrow high peaks in an integrand makes it well suited for evaluating the integrals used in Bayesian analyses. It can be much faster than other popular methods, such as Markov Chain Monte Carlos (MCMC), when applied to small or medium sized problems.

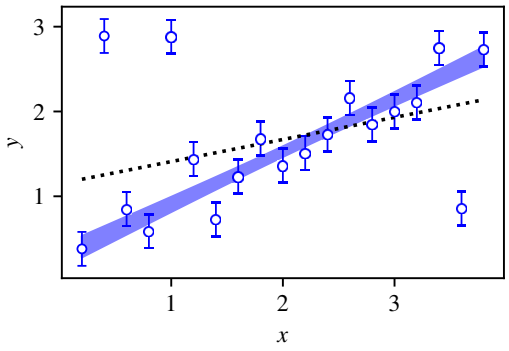

Figure B.12: Bayesian fit (blue band) of data (19 blue data points) with posterior probability Eq. (B.1) and parameters specified by Eqs. (B.7) and (B.9). The dotted line shows the fit from a standard least-squares analysis. The band shows the $\pm 1\sigma$ range around the best fit line.

To illustrate a Bayesian analysis, we consider fitting a straight line $p_0 + p_1 x$ to the data in Fig. B.12 [20]. The error estimates for several of the points are clearly wrong, so we model the data's probability density as a sum of two Gaussians, one with the nominal width and another with 10× that width:

$$P_{\text{data}}(y, \sigma_y | \mathbf{p}, w) \equiv \frac{(1-w)}{\sqrt{2\pi}\sigma_y} \, \mathrm{e}^{-(y-p_0-p_1 x)^2/2\sigma_y^2} + \frac{w}{\sqrt{2\pi}\, 10\sigma_y} \, \mathrm{e}^{-(y-p_0-p_1 x)^2/200\sigma_y^2} \tag{B.1}$$

Here w is the probability of a bad error estimate. Assuming flat priors for the p_μ and w, the Bayesian posterior probability density is proportional to

$$f(\mathbf{p}, w) = P_{\text{prior}}(\mathbf{p}, w) \prod_{i=1}^{19} P_{\text{data}}(y_i, \sigma_y | \mathbf{p}, w), \tag{B.2}$$

where

$$P_{\text{prior}}(\mathbf{p}, w) \propto \Theta(0 < w < 1) \prod_{\mu=1}^{2} \Theta(-5 < p_\mu < 5). \tag{B.3}$$

The Bayesian probability distribution is normalized by computing the 3-dimensional integral

$$I_0 = \int_{-5}^{5} d^2 p \int_0^1 dw \, f(\mathbf{p}, w). \tag{B.4}$$

The mean values for the three parameters are calculated from additional integrals,

$$\langle \mathbf{p} \rangle = \frac{1}{I_0} \int_{-5}^{5} d^2 p \int_0^1 dw \, f(\mathbf{p}, w) \, \mathbf{p}$$
$$\langle w \rangle = \frac{1}{I_0} \int_{-5}^{5} d^2 p \int_0^1 dw \, f(\mathbf{p}, w) \, w, \tag{B.5}$$

and their covariances from still further integrals:

$$\mathrm{cov}_{\mathbf{p}} = \frac{1}{I_0}\int_{-5}^{5} d^2p \int_0^1 dw\, f(\mathbf{p}, w)\, \mathbf{p}\,\mathbf{p}^T - \langle\mathbf{p}\rangle\,\langle\mathbf{p}\rangle^T,$$
$$\mathrm{var}_w = \frac{1}{I_0}\int_{-5}^{5} d^2p \int_0^1 dw\, f(\mathbf{p}, w)\, w^2 - \langle w\rangle^2. \tag{B.6}$$

Additional integrals could provide expectation values $\langle g(\mathbf{p})\rangle$ and/or histograms for arbitrary functions $g(\mathbf{p})$. And so on.

These integrals could be done separately using VEGAS+, but it is generally much better to do them simultaneously, using the same sample points $(\mathbf{p}, w)$ for all of the integrals. This is because the VEGAS+ errors in the different integrals are then highly correlated, leading to significant cancellations in the errors for ratios like Eq. (B.5) and differences like Eq. (B.6). VEGAS+ can estimate the covariances between the estimates of different integrals by summing the covariances coming from each hypercube (estimated using the multivariable generalization of Eq. (6)). Given the covariances, it is possible to account for cancellations in the uncertainties associated with ratios, differences, and other combinations of the integration results.

VEGAS+ with 28,000 integrand samples distributed across 15 iterations gives values for the three model parameters that are accurate to 0.06–0.3% (much more than is needed):

$$\langle\mathbf{p}\rangle_{\text{VEGAS+}} = (0.2817(9),\, 0.6224(4))$$
$$\langle w\rangle_{\text{VEGAS+}} = 0.2628(6). \tag{B.7}$$

We drop the first 5 iterations when estimating parameters. Ignoring correlations between VEGAS+ errors gives results that are 2.5–12× less accurate.

To compare with MCMC [21], we generate 150,000 integrand samples with a MCMC and use the last two thirds of those samples to estimate means for the parameters. The results are are 3–4× less accurate than from VEGAS+, despite using more than 5× as many integrand samples:

$$\langle\mathbf{p}\rangle_{\text{MCMC,5×}} = (0.2832(29), 0.6215(14))$$
$$\langle w\rangle_{\text{MCMC,5×}} = 0.2638(26) \tag{B.8}$$

We estimate the errors for the MCMC simulations by rerunning them several times.

We show the fit line corresponding to the VEGAS+ results for $\langle\mathbf{p}\rangle$ and

$$\mathrm{cov}_{\mathbf{p}} = \begin{pmatrix} 0.0179(2) & -0.00675(8) \\ -0.00675(8) & 0.00318(4) \end{pmatrix} \tag{B.9}$$

in Fig. B.12 (blue band). This is more plausible than the fit suggested by a standard least squares analysis (dotted line). Note that the slope and intercept are anti-correlated, with correlation coefficient −0.89. Classic VEGAS is only somewhat less accurate (40%) than VEGAS+ for these integrals.

Our MCMC analysis is more than an order of magnitude less efficient than VEGAS+ when computing the means and covariances of the fit parameters. The MCMC's precision is limited by its de-correlation time, the number of MCMC steps required between samples to de-correlate them. This accounts for most of the difference here. The last iteration of VEGAS+ from above, for example, uses $N_{\text{ev}} = 1704$ integrand samples to obtain 0.76% and 0.14% errors on the intercept and slope, respectively. (The results in Eq. (B.7) are from the last 10 iterations and so have smaller errors.) From Eq. (B.9), we can estimate that the same number of uncorrelated samples from the posterior distribution Eq. (B.1) would give errors of 1.15% and 0.22%, respectively. So samples from a perfect Monte Carlo (i.e., de-correlation time equals one step) would be slightly less valuable here than samples from VEGAS+, once it has adapted. No general purpose MCMC is perfect, of course. The advantage from VEGAS+ samples grows with increasing N_{ev}, because VEGAS+ errors fall faster than $1/\sqrt{N_{\text{ev}}}$ due to its adaptive stratified sampling (c.f., Eq. (47)).

We also compared VEGAS+ with MCMC for the more difficult problem where there is a separate value of w for each data point. Then w becomes a 19-component vector $\mathbf{w}$ in the equations above, and the integrals are over 21 variables: $\mathbf{p}$ and $\mathbf{w}$. VEGAS+ gives results with better than 1% errors from the last 8 out of 24 iterations, using 162,000 integrand samples in all:

$$\begin{aligned}\langle\mathbf{p}\rangle_{\text{VEGAS+}} &= (0.285(2),\ 0.6172(9))\\ \langle\mathbf{w}\rangle_{\text{VEGAS+}} &= (0.375(3),\ 0.667(3)\ \ldots)\end{aligned} \tag{B.10}$$

An MCMC analysis with 10× as many samples gives results that are 7–13× less accurate than from VEGAS+:

$$\begin{aligned}\langle\mathbf{p}\rangle_{\text{MCMC,10}\times} &= (0.291(26),\ 0.6170(77))\\ \langle\mathbf{w}\rangle_{\text{MCMC,10}\times} &= (0.372(25),\ 0.672(19)\ \ldots)\end{aligned} \tag{B.11}$$

This MCMC analysis used the last third of the samples for estimating parameters.

Classic VEGAS and VEGAS+ are the same for this last example since there are only enough integrand samples to allow a single hypercube in 21 dimensions. This follows because the default behavior in VEGAS+ is to use the same number of stratifications in each direction, and $N_{\text{st}} = 2$ is too many (see Eq. (50)). We can cut the errors for the slope and intercept in half, however, by using $N^{\mu}_{\text{st}} = 46$ stratifications in each of the $\mathbf{p}$ directions, with no stratification ($N^{\mu}_{\text{st}} = 1$) in the other directions. This can be done with the same number of integrand samples as in the unstratified case.

Finally we note that the VEGAS+ results for this problem can be improved (by factors of order 1.5–4) if approximate values for the means of the parameters and their covariance matrix are known ahead of time, for example, from a peak-finding algorithm. Writing the fit parameters as a vector $\mathbf{c}$, this is done by first expressing the deviation from the approximate mean $\mathbf{c}_0$ in terms of the normalized eigenvectors $\mathbf{u}_n$ of the approximate covariance matrix:

$$\mathbf{c} \equiv \mathbf{c}_0 + \sum_n b_n \mathbf{u}_n. \tag{B.12}$$

The integral is then rewritten as an integral over the coefficients b_n rather than the components of $\mathbf{c}$. This transformation reorients the error ellipse so it is aligned with the integration axes, making it easier for the VEGAS map to adapt around the peak. Ref. [7] gives an example of this strategy in use.

References

[1] G. P. Lepage, "A New Algorithm for Adaptive Multidimensional Integration," J. Comp. Phys. **27**, 192–203 (1978).

[2] B. P. Kersevan and E. Richter-Was, "The Monte Carlo event generator AcerMC versions 2.0 to 3.8 with interfaces to PYTHIA 6.4, HERWIG 6.5 and ARIADNE 4.1," Comp. Phys. Comm. **184**, 919-985 (2013).

[3] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H.-S. Shao, T. Stelzer, P. Torriellig and M. Zaroh, "The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations," JHEP07 **(2014)** 079.

[4] T. Aoyama, M. Hayakawa, T. Kinoshita, and M. Nio, "Complete Tenth-Order QED Contribution to the Muon $g-2$," Phys. Rev. Lett. **109**, 111808 (2012).

[5] G. Garberoglio and A. H. Harvey, "Path-Integral calculation of the third virial coefficient of quantum gases at low temperatures," J. Chem. Phys. **134**, 134106 (2011).

[6] G. Campolieti and R. Makarov, "Pricing Path-Dependent Options on State Dependent Volatility Models with a Bessel Bridge," Int. J. Theor. Appl. Finance **10**, 51–88, (2007).

[7] P. Serra, A. Heavens, and A. Melchiorri, "Bayesian Evidence for a cosmological constant using new high-redshift supernova data," Mon. Not. R. Astron. Soc. **379**, 169–175 (2007)

[8] J. L. Sanders, "Probabilistic model for constraining the Galactic potential using tidal streams," Mon. Not. R. Astron. Soc. **443**, 423–431 (2014).

[9] K. Gültekin, D. O. Richstone, K. Gebhardt, T. R. Lauer, S. Tremaine, M. C. Aller, R. Bender, A. Dressler, S. M. Faber, A. V. Filippenko, R. Green, L. C. Ho, J. Kormendy, J. Magorrian, J. Pinkney, and C. Siopis, "The $M-\sigma$ and $M-L$ Relations In Galactic Bulges, and Determination of their Intrinsic Scatter," Astrophys. J. **698**, 198—221 (2009).

[10] J. Ray, Y. M. Marzoukb and H. N. Najm, "A Bayesian approach for estimating bioterror attacks from patient data," Statist. Med. **30**, 101-–126 (2011).

[11] F. M. Atay and A. Hutt, "Neural Fields with Distributed Transmission Speeds and Long-Range Feedback Delays," SIAM J. Appl. Dyn. Sys. **5**, 670–698 (2006).

[12] J. S. Dehesa, T. Koga, R. J. Yáñez, A. R. Plastino, and R. O. Esquivel, "Quantum entanglement in helium," J. Phys. B: At. Mol. Opt. Phys. **45**, 015504 (2012).

[13] D. Varjas, F. de Juan, and Y.-M. Lu, "Bulk invariants and topological response in insulators and superconductors with nonsymmorphic symmetries," Phys. Rev. **B 92**, 195116 (2015).

[14] For a general discussion of importance and stratified sampling see: J. M. Hammersley and D. C. Hanscombe, *Monte Carlo Methods,* Chapt. 5 (Chapman and Hall, London, 1979).

[15] W. H. Press and G. R. Farrar, "Recursive Stratified Sampling For Multidimensional Monte Carlo Integration," Comp. in Phys. **4**, 190 (1990). We used a Python adaptation of the version of the MISER code given in: W. H. Press et al, *Numerical Recipes in C*, 2nd Edition (Cambridge, Cambridge, 2002).

[16] G. Peter Lepage, gplepage/vegas v3.4.5 (2020), Zenodo `http://doi.org/10.5281/zenodo.3897199`. The most recent code is available at: `https://github.com/gplepage/vegas`.

[17] See, for example, the simple implementation of parallel processing, developed by Q. Mason and R. Horgan (private communication), that is used in Ref. [16].

[18] T. Foster, C. L. Lei, M. Robinson, D. Gavaghan, and B. Lambert, "Model Evidence with Fast Tree Based Quadrature," arXiv:2005.11300v1. We used the TQs software provided by the authors at: `https://github.com/thomfoster/treeQuadrature`.

[19] T. Kinoshita and M. Nio, Phys. Rev. **D 73**, 053007 (2006).

[20] This example is adapted from J. Vanderplas' Python blog: `http://jakevdp.github.io/blog/2014/06/06/frequentism-and-bayesianism-2-when-results-differ/`. The data in the figure are included in the examples bundled with the software in Ref. [16].

[21] We used the *emcee* Python package for the MCMC simulations. It is an implementation of the algorithm described in: J. Goodman and J. Weare, "Ensemble Samplers with Affine Invariance," Comm. App. Math. and Comp. Sci. **5**, 65–80 (2010).