Notes on VEGAS

G.Peter Lepage
Newman Laboratory of Nuclear Studies
Cornell University, Ithaca, NY 14850 [*]

DRAFT VERSION—2 Feb. 1990

1 Introduction

This note gives background information on `vegas`, an adaptive Monte Carlo algorithm for multidimensional integration. The algorithm has two components. First an automatic transformation is applied to the integration variables in an attempt to flatten the integrand. Then a Monte Carlo estimate of the integral is made using the transformed variables. The transformation has a restricted form, and is optimized over several iterations of the algorithm: information about the integrand that is collected during one iteration of the Monte Carlo integration is used to improve the transformation used in the next iteration.

The various ingredients that go into the algorithm are discussed in the following sections. A brief discussion of ways in which `vegas` can be extended and/or used in conjunction with other integration methods is also included.

2 Adaptive Maps

The most characteristic feature of `vegas` is the automatic transformation applied to each of the integration variables in an attempt to optimize the Monte Carlo integration. For example, `vegas` might start with an integral like

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

and change the integration variable from x to some $y(x)$, obtaining

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

where $J(y)$ is the Jacobian of the transformation. The integral over y is then estimated using some sort of Monte Carlo method. The variable transformation is chosen so that the product $J(y)\, f(x(y))$ has a form that is optimal for the integration algorithm in use.

In the case of multidimensional integrals, `vegas` applies such a transformation to each integration variable separately.

[*]Computer addresses: `gpl@lnssun1.tn.cornell.edu` and `gpl@crnlnuc.bitnet`.

2.1 Transformation of the Integration Variable

The transformation used by vegas is rather simple. Consider the one dimensional integral introduced above (Eq. (1)). The x-axis is divided into N intervals bounded by

$$\begin{aligned} x_0 &= a \\ x_1 &= x_0 + \Delta x_0 \\ x_2 &= x_1 + \Delta x_1 \\ \cdots & \\ x_N &= x_{N-1} + \Delta x_{N-1} = b. \end{aligned} \tag{3}$$

The points x_i specify the transformation function at the points $y = i/N$ for $i = 0, 1 \ldots N$: i.e., $x(y{=}i/N) = x_i$. Linear interpolation is used between these points, so that in general we have[1]

$$x(y) = x_{i(y)} + \Delta x_{i(y)} \delta(y) \tag{4}$$

where $i(y)$ is the integer part of yN,

$$i(y) = \text{floor}(yN), \tag{5}$$

and $\delta(y)$ is the fractional part of yN,

$$\delta(y) = yN - \text{floor}(yN). \tag{6}$$

This transformation maps the interval $[0, 1]$ onto the original integration region $[a, b]$. The Jacobian is a step function:

$$J(y) = N \Delta x_{i(y)} = J_{i(y)}. \tag{7}$$

2.2 Grid Refinement

The set of x_i's defined above constitutes the vegas grid. To optimize the integration, vegas tunes the Jacobian of the transformation by varying the interval sizes Δx_i, while keeping the sum of all Δx_i's constant. In general 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 concerning the integrand is accumulated in the process, and used to further improve the grid. In this fashion, the grid adapts itself to the integrand over several iterations.

To illustrate how the grid is improved, consider the case where vegas uses crude Monte Carlo integration (in the transformed variable) to estimate the integral in Eq. (1). Then the optimal choice for the Jacobian is one that makes $J(y)\,|f(x(y))|$ as flat as possible. Given some initial grid, vegas samples the integrand at M uniformly distributed points y. While sampling the integrand it accumulates the average value of $J\,|f|$ for each interval of the grid:

$$d_i = \frac{N}{M} \sum_{x(y) \in \Delta x_i} J(y)\,|f(x(y))|. \tag{8}$$

The averages d_i are used in refining the grid. The grid is optimal when all of the d_i's are roughly equal, and so vegas tries to create a new grid for which the d_i's will be uniform in magnitude. (Other definitions of d_i may be more appropriate when using other integration algorithms; in general, we assume there is some such quantity that becomes uniform across the grid when the grid becomes optimal.)

[1]The inverse transformation can be cast in the same terms. However, it is not needed by vegas and so we will not bother further with it here.

This algorithm, like most other 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. Thus it is important to dampen the refinement process so as to avoid rapid, destabilizing changes in the grid. In `vegas` the d_i's are first smoothed:

$$\begin{aligned} d_0 &\rightarrow (d_0 + d_1)/2 \\ d_i &\rightarrow (d_{i-1} + d_i + d_{i+1})/3 \\ d_{N-1} &\rightarrow (d_{N-2} + d_{N-1})/2. \end{aligned} \tag{9}$$

This step is particularly important if the integrand has large discontinuities (e.g., step functions). For example, suppose there is an abrupt increase in the integrand near the upper edge of interval Δx_i. The sample points in that interval might well miss the feature entirely. Without smoothing, `vegas` would see a sudden rise in the function beginning only in interval Δx_{i+1}, and refine the grid accordingly, thereby missing the small, but possibly significant, part of the step in interval Δx_i.

Having smoothed the d_i's `vegas` then compresses their range, again to avoid overreaction to atypically large sample values for the integrand. This is done by first normalizing them,

$$d_i \rightarrow \frac{d_i}{\sum_i d_i}, \tag{10}$$

and then by replacing them by

$$d_i \rightarrow \left(\frac{1 - d_i}{\ln(1/d_i)}\right)^{\alpha} \tag{11}$$

where typically $\alpha \approx 1.5$. Parameter α can be reduced in situations where `vegas` has trouble finding or holding onto the optimal grid ($\alpha \rightarrow 0$ implies no refinement).[2]

The condition for an optimal grid is still that all of the d_i's, now smoothed and compressed, be roughly equal. If the grid is not optimum, `vegas` attempts to improve it. First the d_i's 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 does the trick, starting at the lower end of the old grid and working its way up:

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

   $$\delta d \equiv \frac{\sum_i d_i}{N}; \tag{12}$$

   and initialize the following variables:

   $$\begin{aligned} x'_0 &= x_0; \\ x'_N &= x_N; \\ i &= 0 = \text{index of the current new } x; \\ j &= 0 = \text{index of the current old } x; \\ S_d &= 0 = \text{the amount of } d \text{ currently accumulated.} \end{aligned} \tag{13}$$

2. Increment i. If $i >= N$, the grid is finished.

3. If $S_d \geq \delta d$ then skip to the next step; otherwise add d_j to S_d, increment j, and return to the beginning of this step.

[2]The particular choice of function here is borrowed from Sheppey's program. The actual choice of function does not seem to be too crucial, although the logarithm as in Eq. (11) seems to be a good idea.

4. Subtract δd from S_d, and compute the boundary of a new interval by interpolation:

$$x'_i = x_j - \frac{S_d}{d_{j-1}} \Delta x_{j-1}. \tag{14}$$

Return to Step 2.

3 Integration Strategies

The current `vegas` code chooses one integration algorithm from among a couple of options. These choices are all rather simple; significant improvements may be possible here. We review each of the current options in turn.

3.1 Importance Sampling

In this mode, `vegas` does a straightforward Monte Carlo estimate of the integral using random sample points distributed uniformly throughout the integration region, this after having transformed the integration variables using the adaptive maps described in the last section. So for the one-dimensional integral in Eq. (1), the estimate of the integral is simply

$$I \approx \frac{1}{M} \sum_y J(y)\, f(x(y)), \tag{15}$$

where M is the number of samples used. This estimate is itself a random number, and, when M is large enough, the parent distribution for the estimate is Gaussian with a variance that is approximately

$$\sigma_I^2 \approx \frac{\sum_y J^2(y)\, f^2(x(y))/M - I^2}{M}. \tag{16}$$

Thus σ_I is an estimate of the potential error in the Monte Carlo estimate of I.

By adjusting $J(y)$, `vegas` concentrates integration samples where the integrand is largest in magnitude. This is a form of importance sampling. As discussed in the previous Section, the optimal grid for a one-dimensional integral is one for which $J(y)|f(x(y))|$ is constant, and so `vegas`, given N intervals and M samples of the integrand in each iteration, should adjust the grid so that the averages defined by Eq. (8) are constant throughout. Actually `vegas` uses a slight variation, adjusting the grid so that the quantities

$$d_i = \frac{N}{M} \sum_{x(y)\in\Delta x_i} J^2(y)\, f^2(x(y)) \tag{17}$$

all become approximately equal. This produces the same grid in one dimension if the number of increments N is large enough, and it is the optimal choice for multidimensional integrals (i.e., d_i for a particular interval in a particular integration variable is proportional to the part of the total integral of $J^2\, f^2$ coming from that interval).

3.2 Importance/Stratified Sampling

The variance for estimates of low-dimensional integrals can significantly reduced by stratifying the Monte Carlo samples of the integrand (in the transformed integration volume). So in the case of the one-dimensional integral in Eq. (1), `vegas` transforms to the y variable, as in the last section, and then it divides the y-interval $[0, 1]$ into $L = M/2$ equal subintervals, where M is the allowed number of integrand samples. A two-point Monte Carlo estimate of the integral, $\Delta I \pm \sigma_{\Delta I}$, is made for each subinterval, and these are combined to give the final estimate:

$$\begin{aligned} I &\approx \sum \Delta I \\ \sigma_I &\approx \left(\sum \sigma_{\Delta I}^2\right)^{1/2}. \end{aligned} \tag{18}$$

For n-dimensional integrals, `vegas` divides the y-interval for each direction into L equal subintervals, choosing L so that $2 \times L^n$ is as close as possible to M, the maximum number of samples. These intervals divide the y-integration volume into $N_{\mathrm{hc}} = L^n$ identical hypercubes. A Monte Carlo estimate of the integral is then made in each hypercube using M/N_{hc} points per hypercube, and the results are combined as in the one-dimensional case.

In this mode, `vegas` still adjusts the grid so that the projection of $J^2 f^2$ unto each axis becomes constant. Note that two grids are used in this mode: one that defines the y-variables, and another that defines the stratifications in y-space.

3.3 Stratified Sampling

When working in very low dimensions or with a very large number of integrand samples, `vegas` switches from adaptive importance sampling to adaptive stratified sampling.[3] In this mode, the same grid used to define the variable transformations is used to stratify the final Monte Carlo integration. Estimates of the integral and its uncertainty are as above, but now the optimal grid is different. In this case the grid is chosen not to flatten $J^2\, f^2$, but rather to minimize variations in the variances $\sigma_{\Delta I}^2$ from interval to interval. So in one dimension, for example, `vegas` computes the part of the variance coming from each interval,

$$d_i = \sum_{x(y)\in\Delta x_i} \sigma_{\Delta I}^2, \tag{19}$$

and then refines the grid so as to make the d_i's uniform in i.

It is easy to appreciate why the grid appropriate to importance sampling is different from that appropriate to stratified sampling. Consider, for example, a one-dimensional integrand $f(x)$ that is small and almost flat for $x \ll 0.5$, has a large step around $x = 0.5$, and then is large and almost flat for $x \gg 0.5$. While sample points should be concentrated where the integrand is largest (i.e., $x \gg 0.5$) when using importance sampling, with stratified sampling one concentrates samples where the variation in f is largest, or, more precisely, where the integration errors are largest (i.e., $x = 0.5$). It makes no sense to waste stratifications on the region $x \gg 0.5$; a single stratification, with just two sample points, is quite adequate for this region since the integrand is almost constant there. The key difference with importance sampling is that the algorithm using stratifications knows precisely how many samples are used in each stratification. In importance sampling the number of function evaluations in any given region fluctuates. Large fluctuations in the number of samples for the region $x \gg 0.5$ would result in large fluctuations in the estimate of the integral; and thus an algorithm that does not stratify should concentrate integrand samples in that region so as to minimize fluctuations.

There is no obvious criterion for when to switch from importance/stratified sampling to pure stratified sampling. When working in an importance sampling mode, `vegas` rarely uses fewer than 50 or 100 intervals on each axis when defining the variable transformations. The number of intervals usually must be reduced when switching to stratified sampling, since the total number of integrand samples is then tied to the number of stratifications. In high dimensions, this reduction can seriously impair the algorithm's ability to adapt to the integrand. So `vegas` uses an empirical rule for deciding between the two strategies: adaptive stratified sampling is used if the number of samples permits

[3]Adaptive stratified sampling is the basic technique used in Sheppey's algorithm.

more than 20 or 25 stratifications per axis; otherwise the importance/stratified sampling mode is adopted with 50 or 100 intervals per axis. For some integrands one should switch modes earlier; for others one can wait longer. However this empirical rule gives reasonable performance most of the time, particularly since the stratified sampling option is rarely feasible in high dimensions.

3.4 Statistical Comparison of Iterations

`vegas`, being iterative, usually generates a series of estimates I_α of the integral, each with its own error estimate σ_α. As discussed above, the distribution of I_α's for a given `vegas` grid is approximately Gaussian when a sufficient number of samples is used (and assuming the integral is square integrable). Thus we can combine the estimates from separate iterations to obtain a cumulative estimate of the integral:

$$\begin{aligned} \overline{I} &= \frac{\sum_\alpha I_\alpha/\sigma_\alpha^2}{\sum_\alpha 1/\sigma_\alpha^2} \\ \sigma_{\overline{I}} &\approx \left(\sum_\alpha \frac{1}{\sigma_\alpha^2}\right)^{-1/2}. \end{aligned} \tag{20}$$

The cumulative estimate is generally superior to the estimates from separate iterations. Even more importantly, we can compare the separate iterations to see if they are consistent with each other to within the computed errors. This is a direct check on the quality of the error estimates. Specifically, `vegas` computes the χ^2 statistic for the estimates:

$$\chi^2 = \sum_\alpha \frac{(I_\alpha - \overline{I})^2}{\sigma_\alpha^2}. \tag{21}$$

When the I_α are Gaussian and the standard deviations σ_α are roughly correct, we expect χ^2 to be of order the number of iterations (less one). If χ^2 is considerably larger than this, something is wrong with either $\overline{I}$ or $\sigma_{\overline{I}}$, or with both.

4 New Directions

Watch this spot...