LA-UR-16-23217

Title: OpenMC In Situ Source Convergence Detection

Author(s): Aldrich, Garrett Allen
Dutta, Soumya
Woodring, Jonathan Lee

Intended for: Report

Issued: 2016-05-07

OpenMC *In Situ* Source Convergence Detection

Garrett Aldrich[1, 2], Soumya Dutta[1, 3] and Jonathan Woodring[1]
[1]Los Alamos National Lab, [2]University of California Davis, [3]The Ohio State University

Intro:

We designed and implemented an *in situ* version of particle source convergence for the OpenMC particle transport simulator. OpenMC is a Monte Carlo based particle simulator for neutron criticality calculations [2]. For the transport simulation to be accurate, source particles must converge on a spatial distribution. Typically, convergence is obtained by iterating the simulation by a user-settable, fixed number of steps, and it is assumed that convergence is achieved. We instead implement a method to detect convergence, using the stochastic oscillator for identifying convergence of source particles based on their accumulated Shannon Entropy. Using our *in situ* convergence detection, we are able to detect and begin tallying results for the full simulation once the proper source distribution has been confirmed. Our method ensures that the simulation is not started too early, by a user setting too optimistic parameters, or too late, by setting too conservative a parameter.

Shannon Entropy:

Shannon entropy is a measurement, from information theory, for the amount of uncertainty in a system. When there is more randomness in a system, it has greater entropy, and conversely, when there is more certainty in an event, there is a less amount of entropy. Using Shannon entropy, as a method for detecting convergence of the fissionable source distribution in neutrino transport simulations has been well established [3]. The entropy value is useful because it reduces the effective spatial information of a large number of particles (millions or more) to a single scalar value. This value can be plotted in time, against the number of simulation iterations, to indicate to the user that convergence has been detected.

The Shannon entropy is calculated from the source distribution by defining a regular grid over the simulation domain and binning the number of source particles. The percentage of particles in each cell,

$$S_i = \frac{N_i}{M}$$

Where N_i is the number of particles in cell i and M is the total number of particles in the system. The Shannon entropy for all particles is then defined as,

$$H = -\sum_{i=1}^{n} \mathrm{S_i} \log_2 S_i,$$

Where n is the total number of cells in the regular grid. When H converges around a value, we consider the source distribution to have converged, as well.

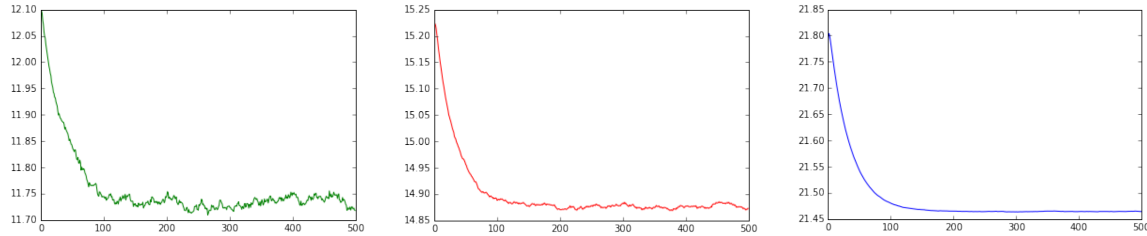


Figure 1: The Shannon entropy over simulation iterations is shown for 100 thousand (green), 1 million (red), and 100 million (blue) source particles in the same benchmarking model. The entropy values increase with the number of particles, but the high frequency noise reduces with the increase.

In Fig. 1, we show the Shannon entropy for three simulation runs using the OpenMC benchmarking model. Each of these simulations was run for 500 time steps, and the Shannon entropy for source particles is plotted against the number of time steps. These graphs show that as the number of particles in the system increase, the total entropy also increases. However, the high-frequency noise in the system is significantly reduced as the number of source particles is increased. Detecting convergence visually is therefore much easier when many source particles are used. Given the possibility of high frequency noise and our desire to do it automatically at run-time, we must introduce an online analysis method to automatically detect convergence.

Stochastic Oscillator:

The stochastic oscillator is an analysis method most often used to indicate trends in stock prices over time. However, it has also been shown to be useful in the detection of convergence in oscillating systems, such as the Shannon entropy of source particles. The stochastic oscillator first normalizes the entropy values within a window, such that when values are increasing over time, the most recent normalized values are close to 1, and when they are decreasing, the normalized values are close to 0. A moving average is also used to smooth the output of the stochastic oscillator.

In particular, the stochastic oscillator for Shannon entropy is defined as,

$$K^n = \frac{H^n - \min\left(H^{n,p}\right)}{\max(H^{n,p}) - \min\left(H^{n,p}\right)}$$

Where K^n is the normalized value, H^n is the entropy at time step n and $H^{n,p}$ is the entropy values over the last p time steps. We further smooth K^n by taking the moving average,

$$A^n = \frac{1}{m}\sum_{i=0}^{m-1} K^{n+i}$$

Source convergence is detected when both the normalized values and their moving average settle on 0.5, indicating that the entropy values are neither increasing nor decreasing. Specifically we test that,

$$|K^n - .5| < \varepsilon \text{ and } |A^n - .5| < \varepsilon$$

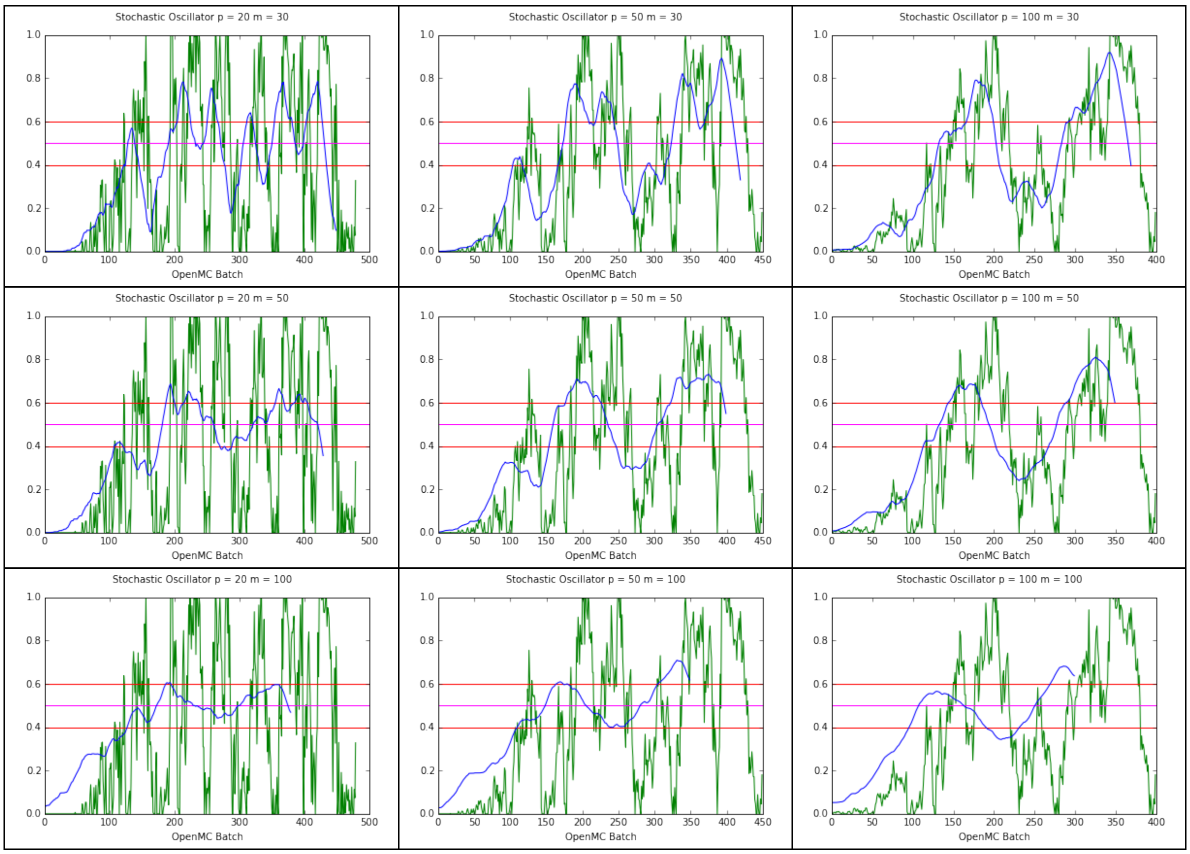


Figure 2: The window sizes and epsilon values can significantly effect the detection of convergence. Here, we show a series of graphs where $p = 20, 50, 100$ ($varying\ on\ the\ y$)$and\ m = 30, 50, 100$ ($varying\ on\ the\ x$). K^n is shown in green, and the moving average A^n is shown in blue. The convergence point, .5, and the epsilon bound (.1) are shown in magenta and red.

While this method works for detecting convergence, the obvious problem is setting the correct parameters for p, m, and, ε. Current literature suggests a set of values, which works well in most cases ($p = 20, m = 50, \varepsilon = .1$) [1], however the detection can fail, producing a false positive. This can happen on difficult simulations where convergence takes many steps. Another concern, especially when implementing the *in situ* detection, is that convergence detection can only occur $p + m$ iterations after convergence has actually happened. This is due to the averaging windows. So while larger values of p, and m can provide a better indication of convergence, it comes at the cost of delayed realization.

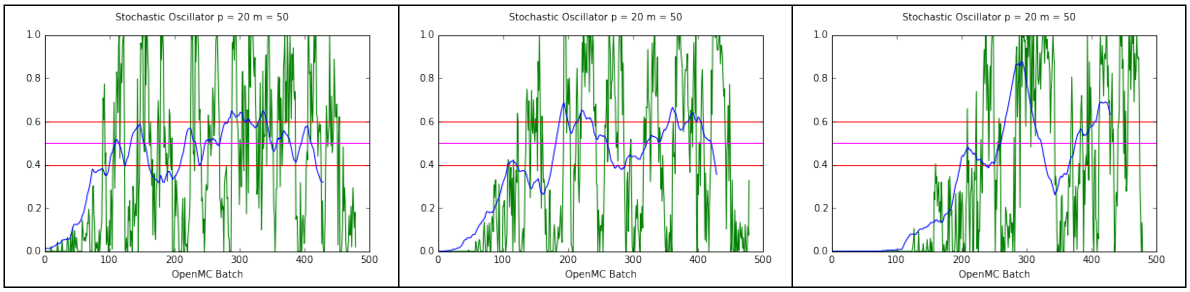


Figure 3 The results of the stoichastic oscillator applied to the 100K, 1M and 100M particle simulations

In Situ Implementation:

We implemented the stochastic oscillator as an *in situ* early convergence test in the OpenMC code base. After each batch has finished, we test for convergence, given that convergence has not already been found, and some maximum number of inactive iterations has not been reached. Once convergence is detected, active batches are started and the results can be tallied.

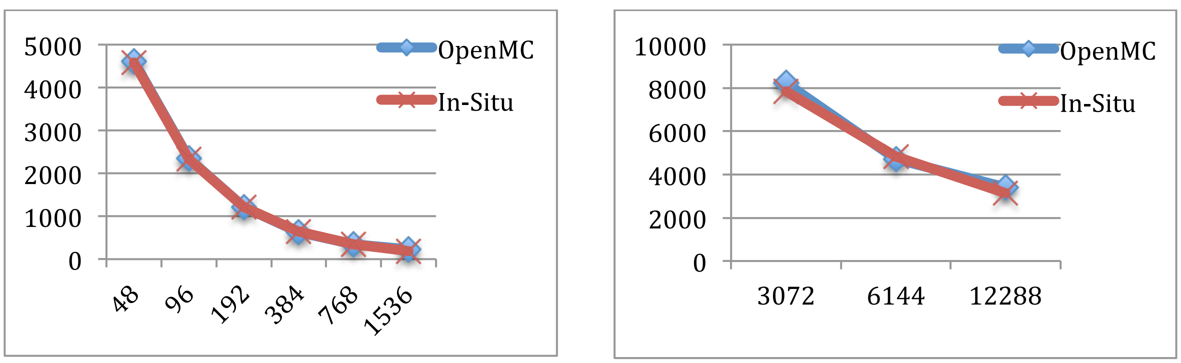
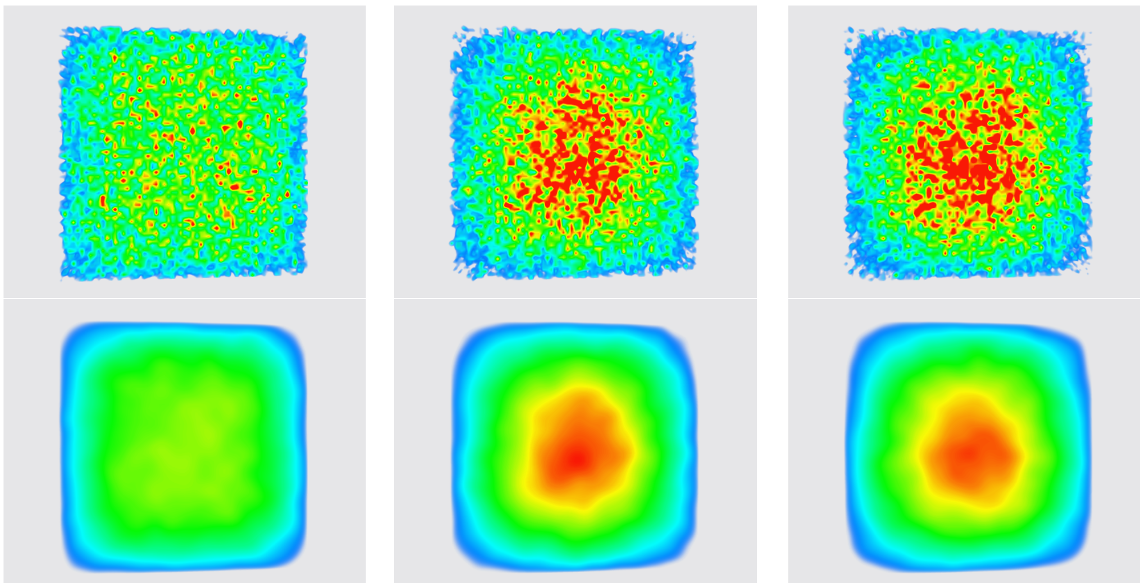


Figure 4 A scaling study using the original code and our *in situ* early convergence detection. The X axis indicates the number of cores, and the Y axis indicates the run time in seconds. On the left, we ran a 1 million particle simulation and on the right 100 million particles were used. These results show that the *in situ* addition to the OpenMC code has negligible impact on performance.

One concern with the analysis is that it would cause a significant performance penalty to the simulation, in particular, due to the communication and synchronization to calculate global entropy after each iteration. In Fig. 4, we show the results of two strong scaling studies, one done with 1 million particles, and the other with 100 million. Both were performed on the LANL Mustang supercomputer, with 24 cores per node. The timing differences between the original OpenMC code and the *in situ* version with convergence detection are nearly identical. To ensure a fair test, we used the same number of inactive batches for the original code.

Conclusion and Future Work:

We implemented the known stochastic oscillator as an *in situ* early convergence detection for source particle distribution in the OpenMC code. We have shown that for the provided benchmark input, the method works well and does not impact performance. More work needs to be done to understand the impact on a greater number of applications, but early indications are positive. The main caveats include setting correct detection parameters for the stochastic oscillator, and the related delay on confirmation of convergence due to the need for temporal smoothing.



(a) (b) (c)

Figure 5 Here, we show a 3D histogram showing the spatial particle density for the 1 million particle simulation. Blue indicates few or no particles, while red is a high density of particles. In the top row we show the raw data, sliced along the x-y plane, on the bottom row the same data is shown, but has been smoothed using a Gaussian filter. The initial time step (a), a time step near the convergence point (b), and several time steps after the source has converged (c) are shown.

In future work, we are interested in applying spatial smoothing on top of the temporal smoothing. In Fig. 5, we show that simple Gaussian smoothing of the source distribution can greatly decrease noise between time steps. This could remove much of the noise we see in the entropy graph, increase the accuracy of convergence detection, and potentially shrink the period of detection.

Bibliography:

[1] Romano, Paul K. "Application of the Stochastic Oscillator to Assess Source Convergence in Monte Carlo Criticality Calculations." *International Conference on Mathematics, Computational Methods & Reactor Physics*. 2009.

[2] Romano, Paul K., and Benoit Forget. "The OpenMC Monte Carlo particle transport code." *Annals of Nuclear Energy* 51 (2013): 274-281.

[3] Ueki, Taro, and Forrest B. Brown. "Stationarity diagnostics using Shannon entropy in monte carlo criticality calculation I: F test." *American Nuclear Society 2002 Winter Meeting, November*. 2002.