LA-UR-17-24406

Title: Verification of MCNP6.2 for Nuclear Criticality Safety Applications

Author(s): Brown, Forrest B.
Rising, Michael Evan
Alwin, Jennifer Louise

Intended for: ANS 2017 Winter Meeting, 2017-10-29/2017-11-02 (Washington, District Of Columbia, United States)
MCNP documentation

Issued: 2017-06-01

Verification of MCNP6.2 for Nuclear Criticality Safety Applications

Forrest Brown, Michael Rising, Jennifer Alwin

Los Alamos National Laboratory, PO Box 1663, Los Alamos, NM, 87544, fbrown@lanl.gov

INTRODUCTION

Several suites of verification/validation benchmark problems were run in early 2017 to verify that the new production release of MCNP6.2 [1,2] performs correctly for nuclear criticality safety applications (NCS). MCNP6.2 results were compared to those from MCNP6.1 [3] and MCNP6.1.1 [4]. MCNP6.2 includes all of the standard features for NCS calculations that have been available for the past 15 years, along with Whisper-1.1 for sensitivity-uncertainty based NCS validation [5]. The standard criticality benchmark suites used for the comparisons are:

- *Verification_Keff* [6] – A suite of criticality problems for which exact analytical results are available. For the current testing, the suite was revised and reconfigured [7] to use the continuous-energy coding portions of MCNP6, the same coding that is used in realistic NCS calculations.
- *Validation_Criticality* [8] – 31 ICSBEP [9] problems, using ENDF/B-VII.1 [10],
- *Validation_Crit_Expanded* [11] - 119 ICSBEP problems, using ENDF/B-VII.1.
- *Validation_Rossi_Alpha* [12] – 13 ICSBEP problems, using ENDF/B-VII.0 and -VII.1.

Over 1.5 x 10^9 active neutrons were run in the course of those calculations. The principal conclusion from the extensive NCS testing is that MCNP6.2 performs correctly, in that results for nearly all problems match results from MCNP6.1 and MCNP6.1.1. In a few cases, results for MCNP6.2 differ by about 1 standard-deviation or less due to known bug fixes or compiler differences. No unusual or unexplained differences were found. In addition, MCNP6.2 was found to run about twice as fast as MCNP6.1. MCNP6.2 is as correct, robust, and reliable for NCS applications as MCNP5, MCNP6.1, and MCNP6.1.1.

METHODOLOGY AND BACKGROUND

Criticality Validation Suites

Previous testing with the *Validation_Criticality* and *Validation_Crit_Expanded* suites used ENDF/B-VII.0 data so that comparisons could be made with the older MCNP5-1.60 code. The current testing used only ENDF/B-VII.1 data, with continuous-energy S(α,β) thermal scattering.

Rossi Alpha Validation Suite

Since the initial work on the *Validation_Rossi_Alpha* test suite was performed in 2011 using MCNP5-1.60 with ENDF/B-VI and -VII.0, these test problems were updated to run with MCNP6 and ENDF/B-VII.1. Various combinations of code and data are used in the present work to show the evolution of the results when compilers and hardware change, when nuclear data is updated, and when minor bug fixes are introduced into the code.

Nuclear Data Libraries

As discussed in [1], there were some changes to the ACE data files distributed with MCNP6.2, but these have little to no effect on most NCS calculations. Hydrogen files now include formerly missing gamma production data; S(α,β) data was revised for SiO_2 and for zirc-hydride at 1200K; and a file *xsdir_mcnp6.2* is now used for MCNP6.2.

MCNP6 Coding Changes

Continuous S(α,β) Numerics

MCNP6.1 had a small, infrequent error in dealing with continuous-energy S(α,β) data at very low energies (e.g., 10^{-5}-10^{-4} eV). This problem was fixed for MCNP6.1.1 and MCNP6.2. While there is insignificant impact on results, there should be some very minor differences in a few results for thermal problems.

After the release of MCNP6.1.1, additional problems were found with round-off errors for some S(α,β) datasets, e.g., zr-h.20t and zr-h.30t. For MCNP6.2, additional checks on this round-off were introduced, and if needed sampling is performed by a different, robust method. Verification testing is unchanged except for a very few cases.

Coincident Surface Treatment

The *universe* and *fill* concepts were introduced into MCNP in the late 1980s. That is, when defining a cell in MCNP input, the cell can be filled with a universe (a collection of cells) rather than a single material. The problem encountered with the original universe/fill treatment occurred when a bounding surface of one or more cells in a universe was coincident with one of the container cell bounding surfaces. MCNP sometimes made a wrong decision on which surface a particle had hit, and lost particles or silent errors were the result.

In the early 1990s, a "fix" for the coincident-surface problem was introduced, first appearing in MCNP4C in 2000. Unfortunately, that fix was flawed and did not account for possible rotations that can be specified for filling a container cell with a (rotated) universe. Lost particles or silent errors could be produced. There was also an absolute tolerance of 0.0001 cm used in the scheme for selecting the surface that was hit.

For MCNP6.2, the coincident surface treatment was revised. During tracking in a cell contained in a universe, distances to the bounding surfaces at all universe levels are examined, and the minimum distance is retained. Each distance has an associated level, with level=0 the "real world," level=1 the next deeper universe in the geometry hierarchy, etc. Then, to allow for round-off in the distance calculations, starting at level=0 distances are examined in order of depth to see if they are within a relative tolerance of $\pm 10^{-6}$ from the minimum distance. The first such distance found is selected, and the remaining distances are ignored. A relative tolerance of $\pm 10^{-6}$ is entirely plausible and consistent as an estimate of possible round-off in distance calculations that use 53-bit precision IEEE standard arithmetic. Retaining the smallest distance (within round-off) at the least-deep level is what is desired. Note that this distance may actually be larger than the distance at a different (deeper) level, but is the correct logical choice to prevent the selection of an incorrect surface distance.

The newly revised coincident-surface treatment is the default for MCNP6.2. The older, flawed treatment can optionally be used instead (e.g., for QA purposes). It is unavoidable that some, but not all, problems that use the universe/fill capabilities will show different results with the new coincident-surface treatment versus the old one, due to the different approaches to dealing with arithmetic round-off. The new coincident surface logic prevents errors when rotated fills are used and is the preferred treatment.

K-Adjoint First K-Effective Estimate

During the calculation of the adjoint-weighted reactor kinetics parameters, MCNP computes an estimate of K_{eff} for a block in the iterated fission probably method. Previously, the block K_{eff} estimate was initialized at the end of the block after the first adjoint-weighted tally scores were made. Consequently, the first estimate of these tallies utilized K_{eff} information from the inactive cycles, introducing a small bias. After the MCNP6.1.1 release, the coding was fixed, with the block-estimate of K_{eff} now initialized at the beginning of the block. This bug fix does change the results of the adjoint-weighted calculation of the reactor kinetics parameters. However, this change is very small, generally much smaller than the statistics of the tallies computed.

Fortran Compiler Issues

An important part of the recent testing was a comparison of results obtained from MCNP6.1 and MCNP6.1.1 compiled with the Intel-12 Fortran compiler versus MCNP6.2 compiled with the Intel-17 Fortran compiler. Fortran compilers are complex software programs, and all such programs have bugs. Testing MCNP using different versions of the Fortran compiler helps to verify that both MCNP and the Fortran compilers are performing correctly for NCS applications. However, it is generally not possible to avoid some minor differences in results caused by different arithmetic round-off between the compilers. Round-off differences are not considered errors.

All of the testing performed recently was done in a parallel mode, using OpenMP threading with 8-16 cpu-cores and the "*-O1*" optimization level. Performance testing showed only small gains in performance with higher optimization levels, at the expense of complications in verification due to small round-off differences.

TESTING RESULTS

The criticality verification/validation suites were run on Mac OS X, Linux, and Windows systems with MCNP6.1, MCNP6.1.1, and MCNP6.2. For Mac OS X, the suites were run on a Mac Pro, 12-core Xeon processor with 2 hyperthreads/core, OS X 10.11.6 & 10.12.4, and 12 threads. For Linux, the suites were run on a single node of a LANL cluster, 8 dual-core Xeon processors, Chaos Linux, and 16 threads. For Windows, the suites were run on a Windows laptop, quad-core I7-4930MX with hyperthreading, Windows 7, and 8 threads.

Verification_Keff Suite

For this suite, MCNP results can be compared to exact results from analytic benchmark problems. In the current testing, MCNP6.2 was run using both multigroup and continuous-energy treatments for 38 analytic benchmark problems. The results from this testing are detailed in [2]. MCNP6.2 gives correct results for all of the analytic problems when run in either multigroup or continuous-energy mode. The absolute RMS accuracy of the results is 3 pcm ± 3 pcm.

Validation_Criticality Suite

Table I shows the K_{eff} results for 31 *ICSBEP* benchmark problems for MCNP6.1, MCNP6.1.1, and MCNP6.2 for a Linux system. For MCNP6.2, results are presented for both the old and new coincident-surface treatments. To simplify the comparisons, the table shows the MCNP6.1 results and differences that arise for MCNP6.1.1 and MCNP6.2. Cases that show differences are highlighted in the tables, and the reasons for the differences are noted. Other detailed results are given in [2].

On Mac OS X, 4 MCNP6.2 problems showed differences from MCNP6.1 or MCNP6.1.1. The differences were less than 2 combined standard deviations. One difference was due to the $S(\alpha,\beta)$ fixes; another to compiler round-off differences; and 2 others to round-off from the new coincident-surface treatment. MCNP6.2 was 1.7 times faster than MCNP6.1.

On Linux, 3 MCNP6.2 problems showed differences from MCNP6.1 or MCNP6.1.1. The differences were less than 2 combined standard deviations. The differences were the same as for Mac OS X, except that the compiler difference did not occur. This is not unexpected, since Mac and Linux compilers sometimes differ in arithmetic round-off. MCNP6.2 was 2.0 times faster than MCNP6.1.

Comparing Mac OS X, Linux, and Windows results for MCNP6.2 from [2] shows agreement in 30 of 31 cases. The one difference for Windows is due to round-off from compiler differences and possibly the slight differences in cpu hardware (I7 on Windows, Xeon on Mac and Linux). The difference on Windows is less than 1 standard-deviation.

Validation_Crit_Expanded Suite

For this benchmark suite, 119 *ICSBEP* benchmark problems were run on both Mac OS X and Linux using MCNP6.1, MCNP6.1.1, and MCNP6.2. Results are given in [2]. For both Mac and Linux, all results for MCNP6.1, MCNP6.1.1, and MCNP6.2 (with old coincident-surface) are identical, and MCNP6.2 shows 11 cases where there are differences of about 1 standard-deviation or less due to the different roundoff for the new coincident-surface treatment. MCNP6.2 is 1.9 times faster than MCNP6.1 on Mac OS X, and 2.2 times faster than MCNP6.1 on Linux.

Validation_Rossi_Alpha Suite

This benchmark suite consists of 13 *ICSBEP* benchmark problems run on Linux using MCNP5-1.60 and MCNP6.1 with ENDF/B-VII.0 data and using MCNP6.1, MCNP6.1.1 and MCNP6.2 with ENDF/B-VII.1 data. Along with the experimental benchmark results, **Table II** shows all of the computed results. The highlighted values indicate a difference compared with the adjacent column to the left and the asterisks indicate the magnitude of the difference. The two differences between MCNP5-1.60 and MCNP6.1 using ENDF/B-VII.0 data are likely due to compiler and hardware round-off differences. Changing the nuclear data from ENDF/B-VII.0 to ENDF/B-VII.1 resulted in all except one benchmark changed, with no consistent trend in the new values with respect to the experimental benchmark values. Finally, MCNP6.1 and MCNP6.1.1 gave identical results, and MCNP6.2 gave different results due to the k-adjoint first k-effective estimate bug fix described previously. While all of the test problems did give very small differences due to this bug fix, most were less than 1 standard-deviation with only two differences observed in the final decimal place shown in the quoted values in **Table II**.

SUMMARY AND CONCLUSIONS

The general conclusions from the recent testing of MCNP6.1, MCNP6.1.1, and MCNP6.2 for NCS applications are:

- MCNP6.1, MCNP6.1.1, and MCNP6.2 perform correctly for NCS applications.
- The *Verification_Keff* results indicate that all versions of MCNP6 are accurate to within 3±3 pcm when exact simple cross-sections are used for analytic benchmarks.
- While small differences were noted for 15 out of 150 *ICSBEP* criticality-only problems and 2 out of 13 *ICSBEP* Rossi-α problems, these are strictly due to arithmetic round-off from different compilers, a minor S(α,β) bug-fix, different arithmetic round-off from the new coincident-surface treatment, or the fixed k-adjoint first k-effective estimate bug, and are not a concern for verification/validation.
- MCNP6.1, MCNP6.1.1, and MCNP6.2 yield the same results on different computer platforms – Mac OS X, Linux, and Windows – for NCS applications.

Criticality safety analysts should consider testing MCNP6.2 on their particular problems and validation suites. No further development of MCNP5 is planned. MCNP6.1 is now 4 years old, and MCNP6.1.1 is now 3 years old. In general, released versions of MCNP are supported only for about 5 years, due to resource limitations. All future MCNP improvements, bug fixes, user support, and new capabilities are targeted only to MCNP6.2 and beyond.

ACKNOWLEDGMENTS

This work was supported by the US DOE-NNSA Nuclear Criticality Safety Program.

REFERENCES

1. F.B. Brown, M.E. Rising, J.L. Alwin, "Release of MCNP6.2 & Whisper-1.1 – Guidance for NCS Users", ANS 2017 NCSD Topical Meeting, Sept. 10-15, Carlsbad NM, LA-UR-17-24260 (2017).
2. F.B. Brown, M.E. Rising, J.L. Alwin, "Verification of MCNP6.2 for Nuclear Criticality Safety Applications", LA-UR-17-23822 (2017).
3. J.T. Goorley, et al., "Initial MCNP6 Release Overview," *Nuclear Technology*, **180**, 298 (2012).
4. J.T. Goorley, "MCNP6.1.1-Beta Release Notes," LA-UR-14-24680 (2014).
5. B.C. Kiedrowski, F.B. Brown, et al., "Whisper: Sensitivity/Uncertainty-Based Computational Methods and Software for Determining Upper Subcritical Limits", *Nucl. Sci. Eng.* **181**, 17-47 [also LA-UR-14-26558] (2015).
6. A. Sood, R.A. Forster, D.K. Parsons, "Analytic Benchmark Test Set for Criticality Code Verification," *Prog. Nucl. Energy*, **42**, 55-106 (2003).
7. F.B. Brown, "The MCNP6 Analytic Criticality Benchmark Suite", LA-UR-16-24255 (2016).
8. RUSSELL D. MOSTELLER, "Validation Suites for MCNP," Proc. ANS 2002 RPSD Topical Meeting, Santa Fe, NM, LA-UR-02-0878 (2002).
9. International Handbook of Evaluated Criticality Safety Benchmark Experiments, NEA No. 7281, OECD NEA (2015).
10. M.B. Chadwick, et al., "ENDF/B-VII.1 Nuclear Data for Science & Technology: Cross Sections, Covariances, Fission Product Yields and Decay Data", *Nuclear Data Sheets*, Vol 112, No. 12, 2887-2996 (2011).
11. R.D. MOSTELLER, "An Expanded Criticality Validation Suite for MCNP", LA-UR-10-06230 Rev3 (2010).
12. R.D. Mosteller, B.C. Kiedrowski, "The Rossi Alpha Validation Suite for MCNP", ICNC 2011, Edinburgh, Scotland, 19-22 Sept. 2011, LA-UR-11-04409 (2011).

Table I. Validation_Criticality – Linux

610_12_71 – 2013, mcnp6.1, Intel-12, endf/b-vii.1
611_12_71 – 2014, mcnp6.1.1, Intel-12, endf/b-vii.1
621_17_71 – 2017, mcnp6.2, Intel-17, endf/b-vii.1, with old coincident-surface treatment
620_17_71 – 2017, mcnp6.2, Intel-17, endf/b-vii.1, with new coincident-surface treatment

	610_12_71_lin		611_12_71_lin		621_17_71_lin		620_17_71_lin		
	keff	std	deltak	std	deltak	std	deltak	std	Reason for diffs
U233 Benchmarks									
JEZ233	1.0000	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
FLAT23	0.9974	(7)	0.0000	(9)	0.0000	(9)	0.0000	(9)	
UMF5C2	0.9960	(7)	0.0000	(9)	0.0000	(9)	0.0000	(9)	
FLSTF1	0.9845	(11)	0.0000	(15)	0.0000	(15)	0.0000	(15)	
SB25	0.9997	(10)	0.0000	(14)	0.0000	(14)	0.0009	(14)	roundoff, coinc-sur
ORNL11	1.0018	(2)	0.0000	(4)	0.0000	(4)	0.0000	(4)	
HEU Benchmarks									
GODIVA	0.9988	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
TT2C11	1.0009	(8)	0.0000	(11)	0.0000	(11)	0.0000	(11)	
FLAT25	1.0034	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
GODIVR	0.9989	(7)	0.0000	(9)	0.0000	(9)	0.0000	(9)	
UH3C6	0.9957	(8)	0.0000	(11)	0.0000	(11)	0.0000	(11)	
ZEUS2	0.9976	(7)	0.0000	(9)	0.0000	(9)	0.0000	(9)	
SB5RN3	0.9945	(13)	0.0000	(18)	0.0000	(18)	0.0000	(18)	
ORNL10	1.0001	(4)	0.0000	(5)	0.0000	(5)	0.0000	(5)	
IEU Benchmarks									
IMF03	1.0019	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
BIGTEN	0.9952	(5)	0.0000	(7)	0.0000	(7)	0.0000	(7)	
IMF04	1.0082	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
ZEBR8H	1.0182	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
ICT2C3	1.0023	(7)	0.0012	(9)*	0.0012	(9)*	0.0012	(9)*	Sab-fix
STACY36	0.9981	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
LEU Benchmarks									
BAWXI2	1.0025	(5)	0.0000	(8)	0.0000	(8)	-0.0004	(8)	roundoff, coinc-sur
LST2C2	0.9960	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
Pu Benchmarks									
JEZPU	0.9990	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
JEZ240	0.9999	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
PUBTNS	0.9980	(7)	0.0000	(9)	0.0000	(9)	0.0000	(9)	
FLATPU	1.0004	(7)	0.0000	(9)	0.0000	(9)	0.0000	(9)	
THOR	0.9976	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
PUSH2O	1.0013	(8)	0.0000	(11)	0.0000	(11)	0.0000	(11)	
HISHPG	1.0121	(5)	0.0000	(8)	0.0000	(8)	0.0000	(8)	
PNL2	1.0050	(10)	0.0000	(14)	0.0000	(14)	0.0000	(14)	
PNL33	1.0068	(7)	0.0000	(9)	0.0000	(9)	0.0000	(9)	
Wall-clock:	18.9 min		10.2 min		9.6 min		9.6 min		
Threads:	16		16		16		16		
Rel. Speed:	1.00		1.86		1.97		1.97		

Table II. Validation Rossi Alpha Test Suite Results

	Benchmark		MCNP5 1.60 ENDF/B-VII.0		MCNP6.1 ENDF/B-VII.0		MCNP6.1 & MCNP6.1.1 ENDF/B-VII.1		MCNP6.2 ENDF/B-VII.1	
	rossi-α	std	rossi-α	std	rossi-α	std	rossi-α	std	rossi-α	std
U233 Benchmarks										
Jezebel-233	-100	(1)	-108	(1)	-108	(1)	-107	(1)	-107	(1)
Flattop-23	-26.7	(5)	-30.2	(4)	-30.2	(4)	-29.8	(4)	-29.8	(4)
HEU Benchmarks										
Godiva	-111	(2)	-113	(1)	-113	(1)	-113	(1)	-113	(1)
Flattop-25	-38.2	(2)	-39.7	(2)	-39.5	(2)	-39.6	(2)	-39.5	(2)
Zeus-1	-0.338	(7)	-0.363	(2)	-0.363	(2)	-0.360	(2)*	-0.360	(2)
Zeus-5	-14.8	(1)	-10.8	(1)	-10.8	(1)	-10.7	(1)	-10.7	(1)
Zeus-6	-3.73	(5)	-4.14	(3)	-4.16	(3)	-4.11	(3)*	-4.10	(3)
IEU Benchmarks										
BIG TEN	-11.7	(1)	-11.8	(1)	-11.8	(1)	-11.7	(1)	-11.7	(1)
STACY-30	-0.0127	(3)	-0.0133	(3)	-0.0133	(3)	-0.0121	(3)***	-0.0121	(3)
STACY-46	-0.0106	(4)	-0.0104	(2)	-0.0104	(2)	-0.0106	(2)	-0.0106	(2)
Pu Benchmarks										
Jezebel	-64.0	(10)	-65	(1)	-65.1	(8)	-63.2	(7)**	-63.2	(7)
Flattop-Pu	-21.4	(5)	-21.0	(3)	-21.0	(3)	-20.2	(3)**	-20.2	(3)
THOR	-19.7	(10)	-20	(1)	-19.7	(7)	-20.6	(7)*	-20.6	(7)

Notes
- All results in 10^4 generations/second
- Color indicates type of diff, * indicates magnitude of diff:

compiler/hardware	* = diff > 1 std
nuclear data	** = diff > 2 std
minor k-adjoint bug fix	*** = diff > 3 std