Runtime Comparisons Between SWMM4 and SWMM5 Using Various Continuous Simulation Model Networks

Carl C. Chan, Robert E. Dickinson and Edward H. Burgess

Abstract

The abstract should be one paragraph of 200 words (plus or minus 50 words). The Stormwater Management Model (SWMM) is a dynamic rainfall–runoff model used for the continuous simulation of runoff quantity and quality. SWMM has recently been redeveloped under a Cooperative Research and Development Agreement (CRADA) between CDM Inc. and the U. S. Environmental Protection Agency (USEPA). One purpose of the CRADA is to improve the numerical stability of the model without changing the fundamental characteristics of the EXTRAN solution and without sacrificing the efficiency of its performance. Extensive QA–QC testing was performed as part of the CRADA to ensure consistency between the SWMM4 and SWMM5 solutions and to compare simulation runtimes between SWMM4 and SWMM5. This paper also describes the approach used to test the simulation speed in SWMM499 and SWMM5 (version Beta E). To ensure data consistency, the historical SWMM4 applications were translated to SWMM5 input format, using the converter software distributed with the SWMM5 package. Results generated from the new SWMM5 were compared to those from the existing SWMM4 model as part of the speed test procedure.

Keywords: up to a maximum of five words or phrases, SWMM4, SWMM5, Euler Solution, Picard solution.

1 Methodology

1.1 Input Datasets

Twenty five sewer network models were selected for the speed test. These datasets were calibrated SWMM4 continuous simulation models, selected from various existing project applications. The sizes of the testing models ranged from ten nodes to 2 500 nodes. These input files cover all appurtenant elements (e.g. orifices, pumps, storages, weirs) in SWMM. Different levels of hydraulic surcharging conditions were also well represented in the selected test datasets.

The example equation inserted here is for style reference only and has no relevance to the content of this paper:

λ(*t*) = λ(*t*0)e*A*(*t* − *t*0) (1)

where:

λ(*t*) = the number of bursts/y/1 000 ft at time *t*,

*t*0 = base year of the analysis,

λ(*t*0) = number of bursts/y/1 000 ft at time *t0*, and

*A* = growth rate coefficient (1/y), with *A* in the range of 0.051/y to 0.151/y, depending on the pipe material and diameter.

1.2 SWMM Engines

SWMM499 and SWMM5 Beta E were chosen to be the testing engines for this exercise. SWMM499, an enhanced version of SWMM44H, is the most recent and final member of the SWMM4.x family. The advantage of using SWMM499 is the ability to produce output files that can be used to directly compare each time step in SWMM5.

1.3 Dynamic Wave Calculation/Integration Methods

Traditionally, SWMM4 has used the modified Euler solution as the sole numerical integration technique in performing dynamic hydraulic analysis. Over the years, two additional variations of this technique were developed (Roesner et al. 1998) and SWMM4 now uses:

1. the original modified Euler solution (ISOL = 0; XROUTE)
2. the enhanced modified Euler solution (ISOL = 1; YROUTE)
3. the iterative under-relaxation Solution (ISOL = 2; ZROUTE)

SWMM5 introduces the Picard solution as the new numerical integration method in the dynamic hydraulic calculation (Dickinson et al. 2004). The Picard method appears to be more robust and, under some conditions, more efficient than the Euler solution. Another new feature in SWMM5 is the variable time step option. SWMM5 has a built-in algorithm that continuously observes the solution accuracy during the course of the computation and adaptively changes the time step size to maintain a consistent level of accuracy. The various SWMM4 and SWMM5 solutions are shown in Table 1.

Table 1 Computation options in SWMM4 and SWMM5.

|  |  |
| --- | --- |
| SWMM4 | SWMM5 |
| 1. Original modified Euler solution (ISOL = 0; XROUTE)  2. Enhanced modified Euler solution (ISOL = 1; YROUTE)  3. Iterative under-relaxation solution (ISOL = 2; ZROUTE) | 1. Euler Solution  variable time step  constant time step  2. Picard Solution  variable time step  constant time step |

The lines for the table can be gotten by applying the standard Word style Table Simple 1.

1.4 Testing Procedure

The selected 25 calibrated SWMM4 continuous simulation models were first simulated in SWMM499, producing results and calibration files for SWMM5. Then the input files were translated to SWMM5 input format using the converter software from the SWMM5 package. The next step was to run the simulations in SWMM5 with various dynamic wave integration methods, with and without variable time step option. Model results were examined to ensure both engines produced similar quantitative results. SWMM4 and SWMM5 produced similar results for all the test models. Runtimes of SWMM4 and SWMM5 were plotted and compared by different computation methods.

All models were run on the same computer workstation for fairness in the comparison. The test computer was equipped with a commonly available Intel Pentium P4 1.6 GHz processor with 256MB RAM.

There are many variables that can affect the speed of a simulation, such as hardware, dataset characteristics, compilers and SWMM solutions. Under certain circumstances, a simple comparison of the runtime duration may not fully describe the speed differences of the two different solutions. For example, a SWMM5 simulation could be faster due to the fact that fewer time steps were required during the course of computation (variable time step = ON) while the actual calculation time required per time step is greater than in SWMM4. In order to avoid this misrepresentation, two measurements of engine speed were developed:

* unit runtime (s/iteration/node); and
* total runtime (s).

Unit runtime was designed to help programmers understand the efficiency of the code. This was used during the engine debugging process. Total runtime was so designated for SWMM users, particularly for continuous simulation where total runtime is a primary concern. Thus simulation runtime was the main focus in speed testing and comparison. Four tests were developed to isolate different parameters that could affect the engine speed:

1. SWMM4 Euler vs SWMM5 Picard (with and without variable time step; non-surcharge condition);
2. SWMM4 Euler vs SWMM5 Euler (with and without variable time step; non-surcharge condition);
3. SWMM4 Euler vs SWMM5 optimized solution (surcharge condition); and
4. Borland Compiler 5.5 vs Microsoft VC Compiler 6.0 (surcharge and non-surcharge conditions).

2 Results and Discussions

2.1 Test 1: SWMM4 Euler vs SWMM5 Picard

The purpose of this test was to identify the speed difference between the SWMM4 Euler solution and the SWMM5 Picard solution. Seven calibrated models were used in this test to compare the speeds of SWMM4 Euler and SWMM5 Picard. The sizes of the sewer networks ranged from 10 nodes to 240 nodes. All models were run using the SWMM4 Euler (ISO=0) with a 5 s fixed time step, the SWMM5 Picard with a 5 s fixed time step, and with the SWMM5 Picard with a variable time step of a maximum 30 s. The simulation period was 31 d. The models were in a non-surcharge condition. Both engines and all methods produced similar quantitative results. Figure 1 shows the speed difference between engines.

The SWMM5 Euler with a constant time step is slightly faster (10%) than the SWMM4 Euler. The SWMM5 Euler with a variable time step is significantly faster (50% to 300%) than the SWMM4 Euler. As the size of the model increases, the relative speed difference between the SWMM5 Euler variable and the SWMM4 Euler increases.

2.2 Test 2: SWMM4 Euler vs SWMM5 Euler

This test was designed to focus on the speed difference between SWMM4 Euler (ISO=0) with a 5 s constant time step, the SWMM5 Euler solution with a 5 s constant time step, and the SWMM5 Euler solution with a variable time step of a maximum 30 s. Seven calibrated non-surcharge models were selected in this test. The sizes of the sewer networks range from 10 nodes to 260 nodes. The simulation period was 31 d.

The results from both engines and methods agreed quantitatively. Figure 2 shows that the runtime of the SWMM5 Euler with a variable time step is similar to the SWMM4 Euler. However, the fixed time step for the SWMM5 Euler is slower than SWMM4. The causes for the slow Euler fixed solution remain uncertain at the time of testing. It could be a combination of minor algorithm flaws or coding inefficiency.

2.3 Test 3: SWMM4 Euler vs SWMM5 Optimized Solution (Surcharge Condition)

The optimized solution described in this test can be defined as the fastest computation in SWMM5. The purpose of this test is to compare the speed of SWMM engines during a surcharge condition.

Five calibrated models were used in this test. The sizes of the models were from 100 nodes to 2 400 nodes. The simulation length was 96 h. Both engines shared similar results quantitatively. Figure 1 shows the speed test result of this exercise. Under a surcharging condition, SWMM5 is much faster than SWMM4 on average. For small models, the speeds of SWMM4 and SWMM5 are similar. As the sizes of the models increase, however, the relative speed difference significantly increases in SWMM5.

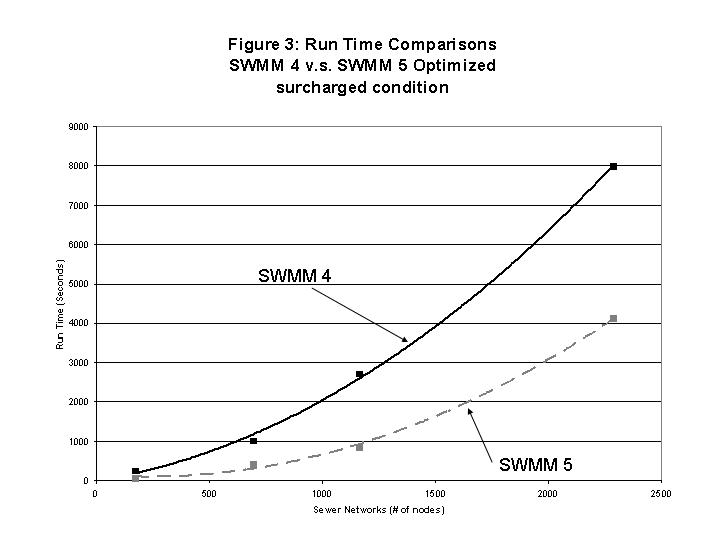


Figure 1 Runtime comparisons: SWMM4 vs SWMM5 optimized (surcharge condition).

The tests above reveal that the Picard solution with a variable time step is the fastest solution in SWMM5 and is significantly faster than SWMM4. The speed benefit dramatically increases when the network size increases. The SWMM5 Picard solution with a fixed time step is slightly faster than SWMM4 although the speed gains decrease with increasing network size. The tests show that the SWMM5 Euler solution with a variable time step is about the same speed as SWMM4. However, the SWMM5 Euler fixed time step is slower than SWMM4. The reason could be some combination of code inefficiency, oversight of calculations, or minor algorithm flaws. The SWMM5 redevelopment team is committed to the improvement of the engine so that it provides a faster and more accurate solution

2.4 Test 4: Borland C SWMM5 vs Microsoft VC SWMM5

During the course of the investigation of the code efficiency, the SWMM5 redevelopment team discovered that the choice of C compiler can impact the computational speed of the engine. The purpose of this test was to help the SWMM5 redevelopment team decide which compiler to use for the official version of SWMM5. Seven models were used in this test, as well as the same datasets used in Test 1. Two compilers were used in this test: Borland C 5.5 (public version) and Microsoft Visual C++ 6.0 (commercial version). Figure 2 shows that the Microsoft Visual C++ compiler has a significantly better performance (55% faster) than the Borland C 5.5 complier.

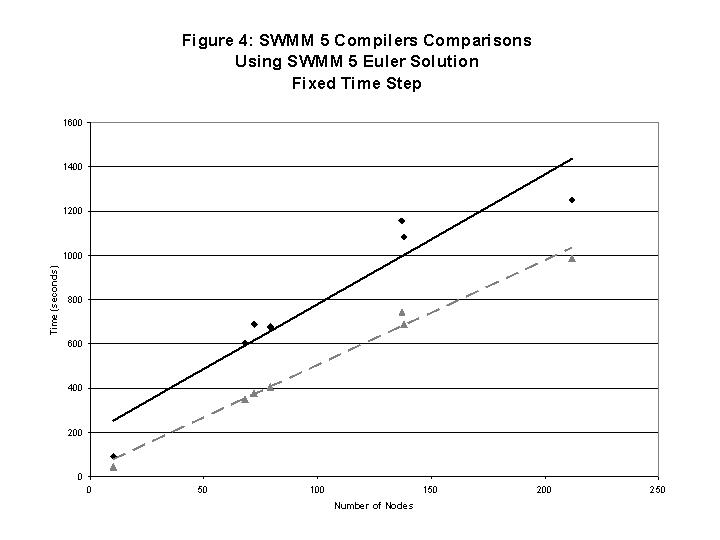


Figure 2 SWMM5 compilers comparisons using SWMM5 Euler solution (fixed time step).

Notice that the Microsoft Visual C++ compiler is two years older than the Borland C 5.5 compiler. The reasons for the different performances remain unknown. The following might explain why Microsoft Visual C++ may be a better choice of compiler for SWMM5.

The test computer uses Windows XP as its operating system. Microsoft Visual C++ is known to be the better compiler for XP.

Borland C 5.5 is public domain (free) software whereas Microsoft Visual C++ is a commercial product. Borland also has a commercial version of a C compiler. It is suspected that the optimization compiling ability in the public domain version is not as proficient as the commercial version.

This test strongly suggests that the SWMM5 redevelopment team should further search for a better compiler for the final release version of SWMM5. Microsoft Visual C ++ 7.0 is now on the market. However, the Borland commercial compiler could be faster than the Microsoft Visual C++ 6.0 compiler.

3 Conclusions

In the past ten years, technological advances in the computer field have been astonishing. As the current trend in the development of high performance desktop computers continues, long term continuous SWMM simulation becomes more feasible in engineering practice. Computation time will become increasingly less of an obstacle to long term continuous hydraulic analysis in the near future. Besides relying on future hardware advancements to improve the computational speed, there are several current ways to increase model efficiency. One is code optimization, in which new algorithms in calculation or data structure may improve computational efficiency. Another possible option to help in gaining speed is to migrate the SWMM program code to the Linux environment. With the SWMM code now entirely written in the C language, migrating to Linux should not be difficult. The new 64-bit microprocessors can also be an option for improving simulation runtimes. Lastly, parallel processing technology should be deployed in SWMM5. CDM has written a SWMM4 engine that can take advantage of parallel processors (Burgess et al.1999). The SWMM5 redevelopment team should review the parallel code and incorporate it into SWMM5.

References

The following references illustrate the most common forms used (book, journal, chapter in book, conference proceedings). We use the author-date system specified in the *Chicago Manual of Style* (16th Edition).

Bennett, A. 2011. “Potable Water: New Technology Enables Use of Alternative Water Sources.” *Filtration+Separation* (March/April): 24–7.

Bremere, I., M. Kennedy, A. Stikker and J. Schippers. 2001. “How Water Scarcity Will Effect the Growth in the Desalination Market in the Coming 25 Years.” *Desalination* 138:7–15.

Brix, H., H. H. Schierup and C. A. Arias. 2007. “Twenty Years Experience with Constructed Wetland Systems in Denmark: What Did We Learn?” *Water Science and Technology* 56 (3): 63–8.

Burgess, E., W.R. Magro, M. Clement, C. Moore and J. Smullen. 2000. "Parallel Processing Enhancement to SWMM/EXTRAN." *Journal of Water Management Modeling* R206-03. doi: 10.14796/JWMM.R206-03.

Chan, C.C., E. Burgess, T. Schade and R.E. Dickinson. 2003. "Design of the SWMM Data Translation and the Testing of SWMM 5 Engine." *Journal of Water Management Modeling* R220-17. doi: 10.14796/JWMM.R220-17.

Dickinson R. E., E. H. Burgess, L. A. Rossman and L.A. Roesner. 2004. “The Modified Euler Solution Algorithm and the Picard Solution Algorithm in SWMM5.” Paper presented at The 2004 Conference on Stormwater and Urban Water Systems Modeling, Toronto, February 19–20, 2004.

Huber, W. C. and R. E. Dickinson. 1988. *Storm Water Management Model User’s Manual Version 4*. Athens, GA: U. S. Environmental Protection Agency. EPA/600/3-88/001A.

Roesner, L. A., R. E. Dickinson and J. A. Aldrich. 1988. *Storm Water Management Model User’s Manual, Version 4, EXTRAN Addendum*. Athens, GA: U. S. Environmental Protection Agency. EPA/600/3-88/001B.

Rossman, L.A., R.E. Dickinson, T. Schade, C.C. Chan, E. Burgess, D. Sullivan and F. Lai. 2003. "SWMM 5 - the Next Generation of EPA's Storm Water Management Model." *Journal of Water Management Modeling* R220-16. doi: 10.14796/JWMM.R220-16.

Schade, T. 2002. *Quality Assurance Project Plan for SWMM Redevelopment*. Cincinnati, OH: U. S. Environmental Protection Agency.

USEPA. 2002. *SWMM Redevelopment Project Plan—Version 5*. Washington DC: U. S. Environmental Protection Agency.

Author Affiliations

For each author, please include in this section (using one paragraph for each author) the author’s name, business/institutional affiliation and geographical location. Please identify the corresponding author as such.

Robert E. Dickinson, Innovyze Inc., Tampa, FL.