AMG Test Problem Set

Version 1.0

- For the definition of terms such as SCCS, see the Sequoia Benchmark Glossary on the Sequoia Benchmark Web site.
- All runs described below are to be done using problem # 1 as described in the AMG tar file's internal documentation.
- All runs are to be done with executables that have been compiled with compile option `-DHYPRE_NO_GLOBAL_PARTITION`.
- For each problem defined below, two runs should be done, one with solver 3, and one with solver 4. Each is chosen by a parameter on the execution line.

Single Core

Set 1 Set the refinements rx = ry = rz = N, for N = 2, 4, 6, 8, 10, 12, … as large as the benchmark system allows. (Recognizing when AMG stops working is not simple. Guidance for maximum system size versus memory per node will be forthcoming for both solver 3 and solver 4.)

Single Node

Set 2a,b **Weak scaling**: For Set 2a, use the command-line parameters rx = ry = rz = 4, and for Set 2b use rx = ry = rz = 12. For each set, run as many of the values of N_{MPI}, Px, Py, and Pz as are possible on the benchmark system from the following list.

N_{MPI}	Px	Py	Pz
1	1	1	1
2	1	1	2
3	1	1	3
4	1	2	2
5	1	1	5
6	1	2	3
7	1	7	1
8	2	2	2
9	3	1	3
10	1	2	5
12	2	2	3
14	1	7	2
15	3	1	5
16	2	4	2
18	3	2	3
20	2	2	5
21	1	7	3
24	3	4	2
28	2	7	2
30	3	2	5
32	4	4	2

For the one case of $N_{MPI} = N_{C/Node}$, an additional run may be done in which the values of Px, Py, and Pz can be permuted to seek the highest figure of merit (FOM)*.

Set 3 **Strong scaling**: Run all values of $N_{MPI} = N_{C/Node}$ in the following list, using the given

values of rx, ry, rz, Px, Py, and Pz.

N_{MPI}	rx	ry	rz	Px	Py	Pz
1	24	28	30	1	1	1
2	24	28	15	1	1	2
3	24	28	10	1	1	3
4	24	14	15	1	2	2
5	24	28	6	1	1	5
6	24	14	10	1	2	3
7	24	4	30	1	7	1
8	12	14	15	2	2	2
9	8	28	10	3	1	3
10	24	14	6	1	2	5
12	12	14	10	2	2	3
14	24	4	15	1	7	2
15	8	28	6	3	1	5
16	12	7	15	2	4	2
18	8	14	10	3	2	3
20	12	14	6	2	2	5
21	24	4	10	1	7	3
24	8	7	15	3	4	2
28	12	4	15	2	7	2
30	8	14	6	3	2	5
32	6	7	15	4	4	2

For the one case of $N_{MPI} = N_{C/Node}$, an additional run may be done in which the values of rx, ry, rz, Px, Py, and Pz can be permuted to seek the highest FOM subject to the following three constraints; $kx = rx \times Px = 24$, $ky = ry \times Py = 28$, and $kz = rz \times Pz = 30$.

If there is not enough memory, kx, ky, and kz can be chosen smaller, but should be close to each other, i.e., kx approx. ky approx. kz .

Set 4 **OpenMP test**: Set the refinements rx = 24, ry = 28, rz = 30. Run with 2 and 4 OpenMP threads, in both cases using one MPI task.

Multiple Nodes

Set 5 **Weak scaling, and large fixed problem size**: Set the refinements rx = ry = rz = 6; N_{MPI} = SCCS.

Vendor should minimize the ratio max(px,py,pz) / min(px,py,pz) for any test case.

Run A **Benchmark system maximum**: Vendor choice of rx, ry, rz, Px, Py, and Pz to maximize the FOM on the benchmark system.

Run B **Large, fixed processor count**: Vendor choice of rx, ry, rz, Px, Py, and Pz with $Px \times Py \times Pz = 2{,}048$ to maximize the FOM on the benchmark system.

Run C **Large, fixed problem size**: Vendor choice of rx, ry, rz, Px, Py, and Pz, with $rx \times ry \times rz \times Px \times Py \times Pz = 2 \times 10^6$ to maximize the FOM on the benchmark system.

The maximun “rx/ry/rz” value that can be used in Run A, B, or C cannot exceed 6.

[*] Figure of merit (FOM) = system_size × (#iterations/solve time) × [a scale factor yet to be specified] where:

solve time is wall-clock time,

system_size = 512 × rx × ry × rz × Px × Py × Pz.

Note: This number is printed out by the benchmark at the end of a run.