

Comparison of Selected Solvers

David Andrs

December 1, 2009

Abstract

This report compares a few selected solver, namely UMFPACK, MUMPS and AztecOO on a simple PDE. Comparison involves mainly time consumption of several computation phases.

1 Solved Problem

The testing problem is a so called Fichera corner problem given by:

$$-\Delta u = f$$

on a domain $(-1, 1)^3 \setminus (0, 1)^3$. Function f is chosen in such a way that the solution is $u = (x^2 + y^2 + z^2)^{\frac{1}{4}}$.

1.1 Measurements

The problem was solved using *hp*-adaptivity (implemented in Hermes3D library) on a machine with Intel(R) Xeon(R) CPU X5450 3.00 GHz, 32 GB of memory. The adaptivity was guided with an exact solution¹.

We measured not only time required for solving, but also time required for assembling and the adaptivity step to see the amount of time being spent in the solver in comparison to the rest of the iterations.

In the histogram-like graphs, *adapt-error* refers to the time required to calculate the error between exact and calculated solutions, *adapt-adapt* means the time required for finding the candidates².

2 Results

The following table shows the number of DOFs, elements, fill-in ratio of the stiffness matrix and the number of elements refined in the adaptivity step.

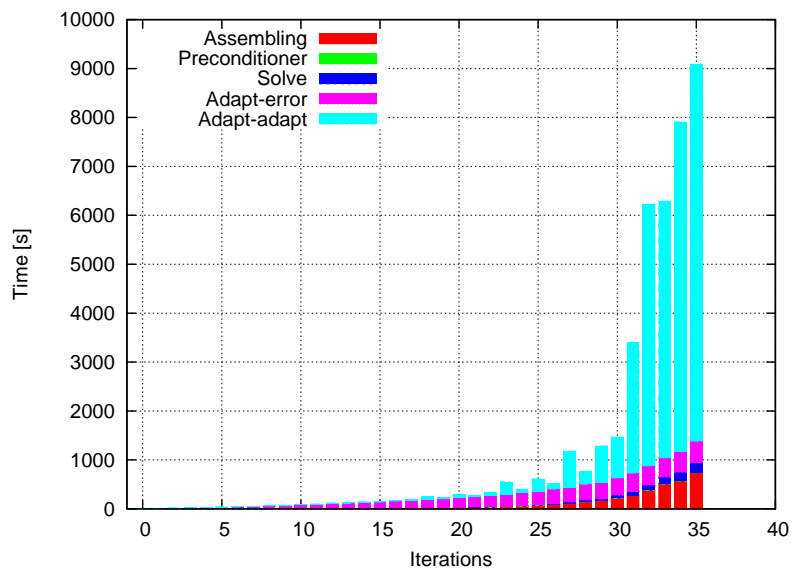
¹Guiding with reference solution ended the calculation after several steps because of memory issues

²The call of `hp.adapt()`

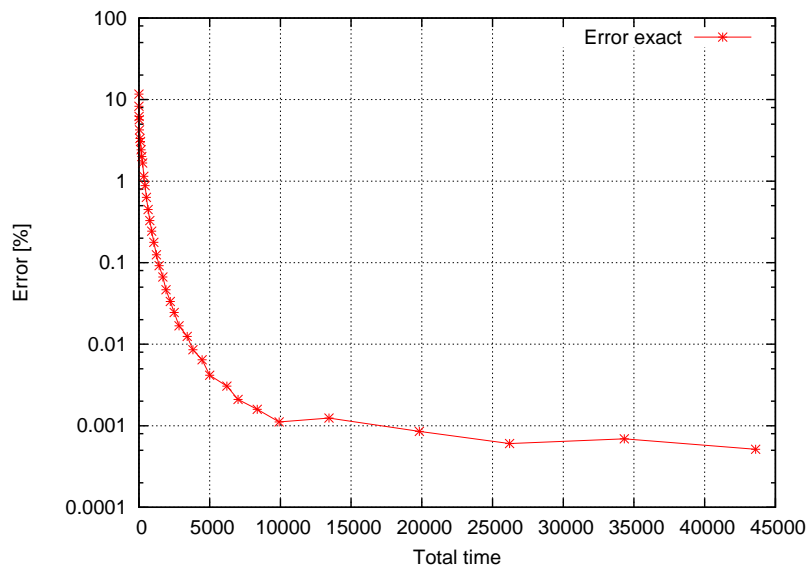
DOFs	Elms	Fill-in	Refined elements
19	7	4.349030e-01	6
79	7	2.924211e-01	4
155	28	2.048283e-01	6
263	49	1.222368e-01	4
339	56	1.081352e-01	4
415	77	1.062389e-01	6
523	98	8.558142e-02	4
599	105	7.606166e-02	7
762	147	6.356390e-02	7
859	154	5.712981e-02	28
1307	196	4.511696e-02	28
1743	203	3.684503e-02	28
2191	245	3.273984e-02	35
2760	252	2.918268e-02	28
3208	294	2.599603e-02	35
3777	301	2.394529e-02	41
4502	343	2.108121e-02	64
5694	350	1.911100e-02	56
7298	392	1.749474e-02	70
9498	399	1.727579e-02	63
10953	441	1.522826e-02	76
13213	448	1.452401e-02	63
14668	490	1.323705e-02	92
17352	497	1.219406e-02	77
21089	539	1.166662e-02	98
23833	546	1.059468e-02	77
27570	588	1.040275e-02	119
31487	595	9.415870e-03	108
36686	637	9.004784e-03	102
40889	644	8.472724e-03	107
46135	686	8.179081e-03	129
53182	693	7.735894e-03	114
62059	735	7.500131e-03	24
66235	735	8.080447e-03	159
73132	763	7.536840e-03	120

2.1 UMFPACK Solver

UMFPACK is direct solver written by Tim Davis (mainly in C).



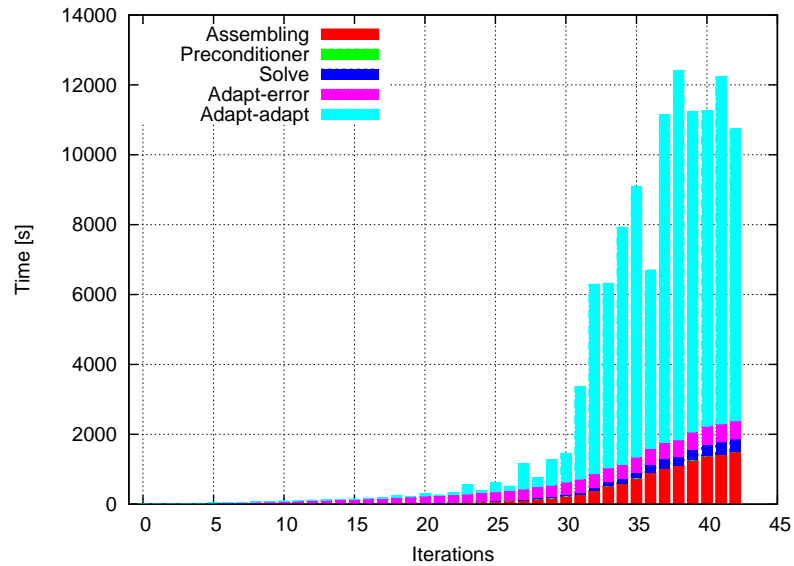
Time for computation phases for UMFPACK solver



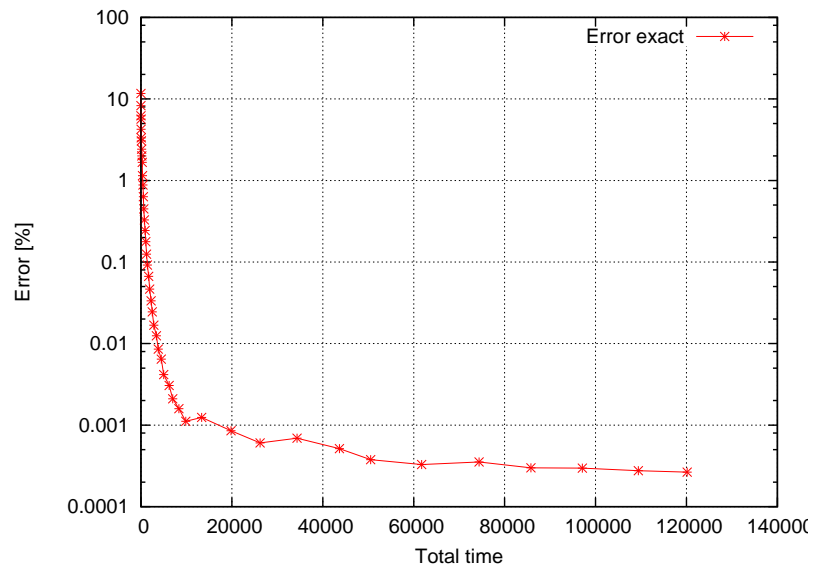
Time convergence for UMFPACK solver

2.2 MUMPS Solver

MUMPS is written in Fortran and it is also a direct solver.



Time for computation phases for MUMPS solver

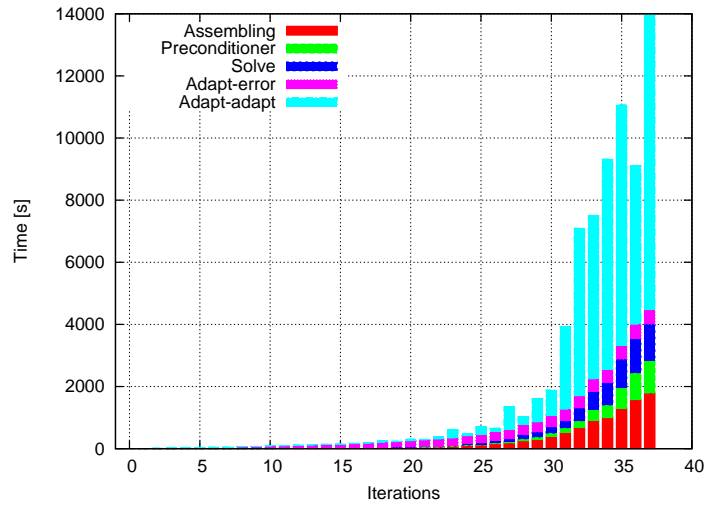


Time convergence for MUMPS solver

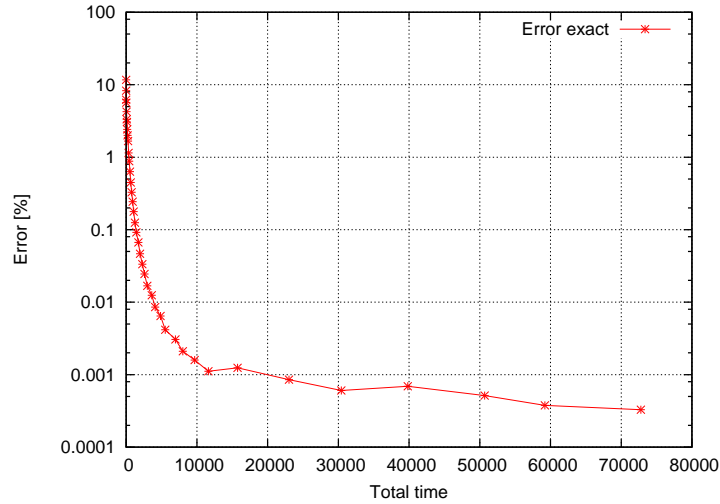
2.3 AztecOO Solver

AztecOO is an iterative solver (it is a part of Trilinos package). We used GMRES method for solving the matrix and compared three configurations: no preconditioner, ML-based preconditioner (using ML package), ILU preconditioner from AztecOO package³

2.3.1 ML Preconditioner



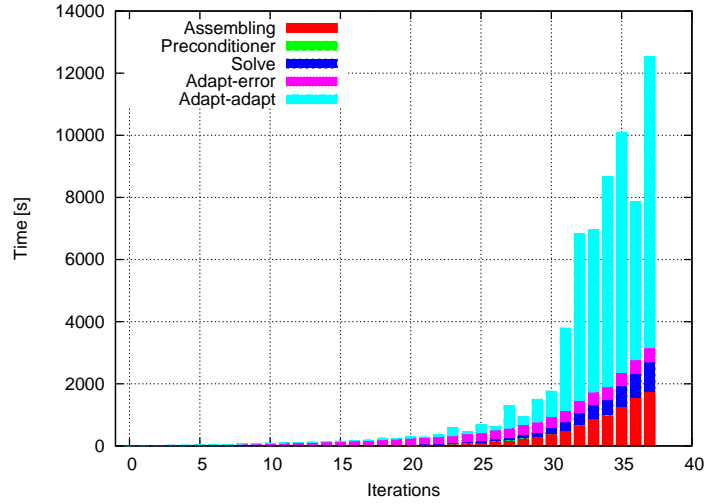
Time for computation phases for AztecOO solver + ML preconditioner



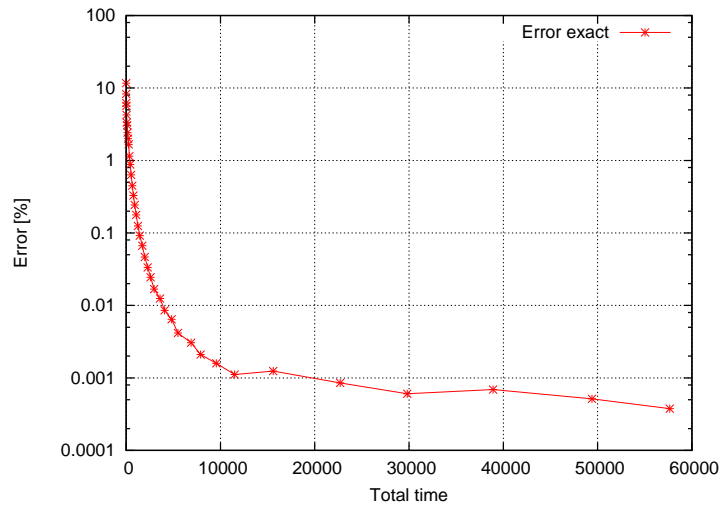
Time convergence for AztecOO solver + ML preconditioner

³It is possible to construct ILU preconditioner using IFPACK package.

2.3.2 ILU Preconditioner

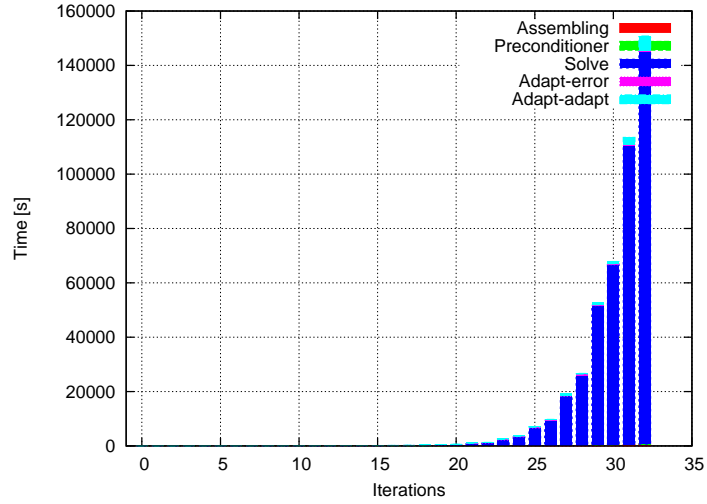


Time for computation phases for AztecOO solver + ILU preconditioner

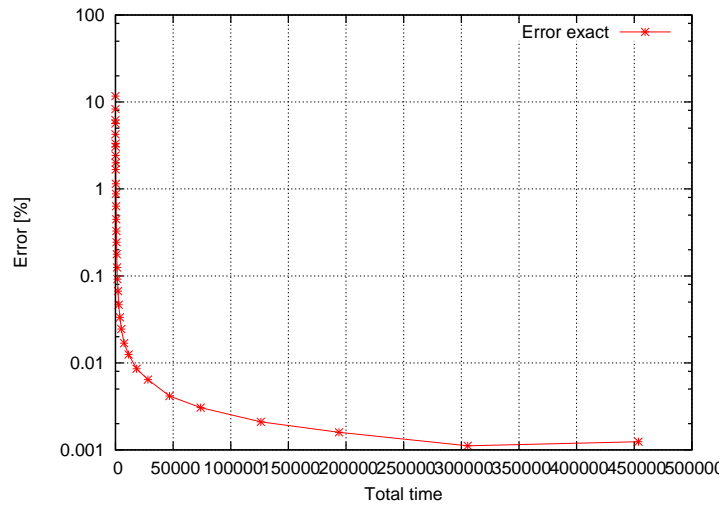


Time convergence for AztecOO solver + ILU preconditioner

2.3.3 No Preconditioner



Time for computation phases for AztecOO solver w/o preconditioner



Time convergence for AztecOO solver w/o preconditioner

2.3.4 Comparison of Preconditioners

The following table compares ML and ILU preconditioner, it shows the number of required iterations, achieved residual and time spend on building the preconditioner and solve cycle.

ML			ILU		
iters	residual	time	iters	residual	time
2	2.353093E-16	0.016947	2	2.537207E-16	0.030756
2	3.515601E-16	0.003209	2	2.205697E-16	0.000444
19	2.166322E-15	0.049038	2	1.007801E-15	0.001371
25	3.978865E-15	0.014691	4	2.243477E-15	0.003592
28	3.160473E-15	0.02097	5	2.65049E-15	0.00674
28	4.353426E-15	0.02979	5	2.723368E-15	0.010392
29	4.861079E-15	0.039673	6	3.404806E-15	0.019226
269	6.086832E-15	0.204468	7	4.224932E-15	0.02477
30	4.180241E-15	0.064209	7	3.537499E-15	0.038492
72	7.616182E-15	0.115405	7	4.243018E-15	0.04493
36	4.399029E-15	0.145797	8	4.699138E-15	0.087022
43	7.056679E-15	0.249239	8	5.678652E-15	0.136225
49	2.707695E-15	0.349953	8	5.463299E-15	0.221937
44	4.478258E-15	0.550336	9	5.680215E-15	0.378899
46	5.857577E-15	0.630528	9	5.957868E-15	0.492461
49	5.888737E-15	0.889523	9	6.724519E-15	0.686324
48	4.539718E-15	1.331869	9	6.266355E-15	0.908205
59	7.69252E-15	2.321451	9	8.790086E-15	1.533473
86	6.523263E-15	5.294865	10	5.607093E-15	2.766849
112	4.502607E-15	12.190696	10	6.685786E-15	5.53006
102	5.369358E-15	13.653861	11	6.532758E-15	6.929876
107	5.79226E-15	21.263538	10	7.470489E-15	11.281467
118	6.847573E-15	24.837364	11	9.302801E-15	13.320248
124	4.532617E-15	34.699528	11	5.885139E-15	19.323008
141	6.507663E-15	54.630892	11	7.62331E-15	31.299904
148	5.211381E-15	66.781695	10	7.465844E-15	38.742596
160	5.147479E-15	106.461068	11	7.811873E-15	60.271149
162	6.402147E-15	126.495611	11	7.494833E-15	80.13377
197	5.939829E-15	197.581339	11	7.715921E-15	110.509599
191	6.698509E-15	229.95946	11	7.370817E-15	136.751555
208	5.587774E-15	313.608761	12	1.322008E-14	191.616296
196	6.255511E-15	376.374599	12	1.096446E-14	261.919472
262	6.392941E-15	623.656462	12	1.013226E-14	386.036404
297	9.501331E-15	934.819222	14	9.620493E-15	446.674736
336	4.525606E-15	1141.209032	17	1.525212E-14	492.046329
353	4.991225E-15	1581.344589	23	1.861955E-14	645.367145
360	6.94084E-15	1969.141275	25	1.031553E-13	754.61065
354	6.132517E-15	2211.799109	51	6.362687E-15	929.116879

2.4 Comparison of Solve Time

The following table compares the solve time for all tested solvers. The numbers show the total time the solver needed for solving the problem.

DOFs	ML	ILU	UMFPACK	MUMPS
19	0.016947	0.030756	0.001272	0.071353
79	0.003209	0.000444	0.000455	0.00062
155	0.049038	0.001371	0.001488	0.001358
263	0.014691	0.003592	0.002783	0.002633
339	0.02097	0.00674	0.004517	0.004318
415	0.02979	0.010392	0.007178	0.006302
523	0.039673	0.019226	0.010186	0.009294
599	0.204468	0.02477	0.014129	0.013128
762	0.064209	0.038492	0.022357	0.019552
859	0.115405	0.04493	0.025572	0.02259
1307	0.145797	0.087022	0.057292	0.060633
1743	0.249239	0.136225	0.086221	0.094665
2191	0.349953	0.221937	0.17089	0.155732
2760	0.550336	0.378899	0.225348	0.179466
3208	0.630528	0.492461	0.350038	0.264033
3777	0.889523	0.686324	0.36595	0.349379
4502	1.331869	0.908205	0.489139	0.455547
5694	2.321451	1.533473	0.860339	0.89071
7298	5.294865	2.766849	1.370579	1.23824
9498	12.190696	5.53006	2.696907	2.271886
10953	13.653861	6.929876	3.267665	3.134226
13213	21.263538	11.281467	5.114654	4.60429
14668	24.837364	13.320248	6.386882	5.658775
17352	34.699528	19.323008	9.712641	7.6026
21098	54.630892	31.299904	12.521468	11.111559
23833	66.781695	38.742596	15.740038	15.483907
27570	106.461068	60.271149	21.908203	20.123818
31487	126.495611	80.13377	27.207123	24.562024
36686	197.581339	110.509599	35.010204	30.497097
40889	229.95946	136.751555	39.921257	35.531281
46135	313.608761	191.616296	56.631107	52.620044
53182	376.374599	261.919472	81.78604	62.149052
62059	623.656462	386.036404	99.411548	87.733926
66235	934.819222	446.674736	132.44629	114.522331
73132	1141.209032	492.046329	172.399927	133.707893

Note: this report compares only CPU times, but not memory requirements. Usually direct solvers needs more memory than iterative solvers. Also direct solvers cannot be used for JFNK

Note2: Using AztecOO solver without preconditioning did not lead to a correct result in a sense that the number of maximum iterations was not enough to achieve prescribed tolerance — which was 10^{-15}