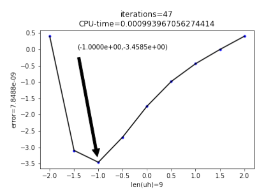
Project3.pdf

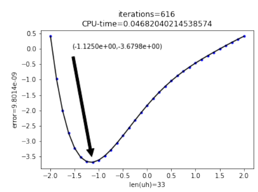
Dario Maddaloni (233187)

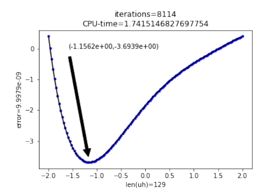
June 2022

Task 1.2

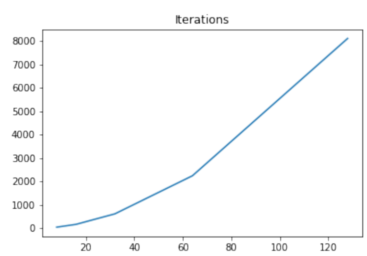
In this task, we apply the one-step Gauss-Seidel function with tol=10^{-8} on the pseudo_residual. In particular, I printed the graph of the solutions given by the function w.r.t. some len(uh).

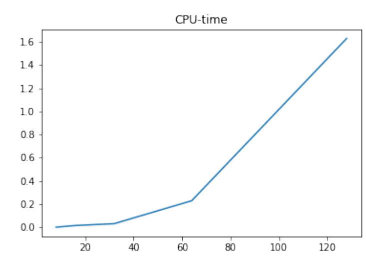






It the following graphs and table, we can see the number of iterations of the one-step Gauss-Seidel and the CPU time (expressed in seconds) needed to attain a pseudo residual smaller 10^{-8}. Note that len(uh) are of the form $(2^n)+1$

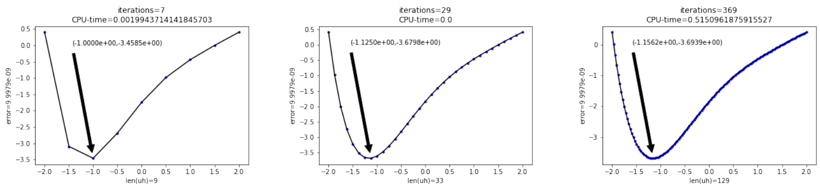




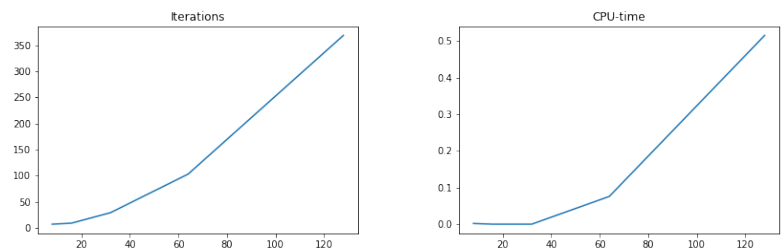
Len(uh)	Iterations GS	Time GS
9	47	0.000997
17	168	0.003987
33	616	0.030249
65	2245	0.247549
129	8114	1.679613

Task 1.3

In this task, we apply the two-grid-step function with tol=10^{-8} on the pseudo_residual. In particular, I printed the graph of the solutions given by the function w.r.t. some len(uh).



It the following graphs and table, we can see the number of iterations of the two-grid-step and its CPU time (expressed in seconds) needed to attain a pseudo residual smaller 10^{-8}. Note that len(uh) are of the form $(2^n) + 1$. Morover, note that the len(uh) are exactly the same as the previous ones used for one-step Gauss-Seidel. Thanks to this, we can compare the CPU times and the number of iterations



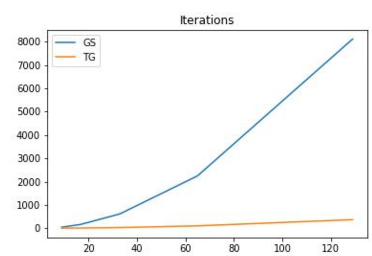
Points	Iterations TG	Time TG
9	7	0.001994
17	9	0.0
33	29	0.0
65	103	0.075798
129	369	0.515096

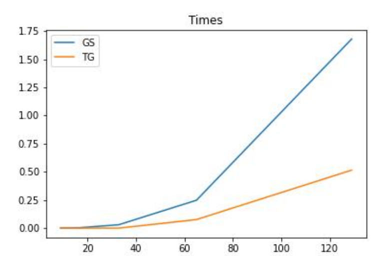
Comparison

In this subsection we will compare the number of iterations and CPU time required to attain a solution of our problem with pseudo residual less than 10^{-8}. I want to throw light on the fact that in order to make the table more readable I highlighted in green the number of iterations and in blue the CPU time (always expressed in seconds)

Points	Iterations GS	Times GS	Iterations TG	Time TG
9	47	0.000997	7	0.001994
17	168	0.003987	9	0.0
33	616	0.030249	29	0.0
65	2245	0.247549	103	0.075798
129	8114	1.679613	369	0.515096

We can see graphically as follows:



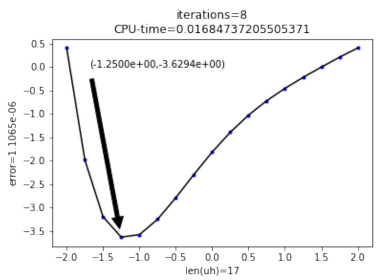


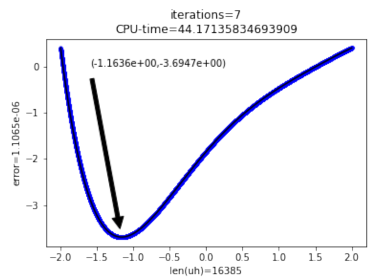
How we could expect the two-grid-step function is faster than the Gauss-Seidel one. Indeed, with the two-grid-step we are able to adjust the errors we are dealing with when we are using big uh

Task 1.5

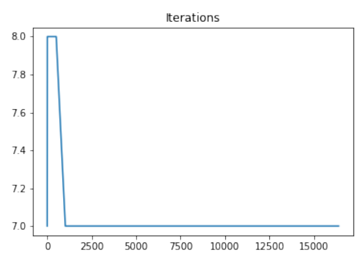
In this task, we apply the v_cycle_step function with tol=10^{-8} on the pseudo_residual. In particular, I printed the graph of the solutions given by the function w.r.t. some len(uh). Note that every page has different parameters α_1 and α_2. Moreover, I want to stress the fact that the v_cycle_step is based on gs-step-1d and it could be seen as a generalization of the two-grid-step with a recursive method.

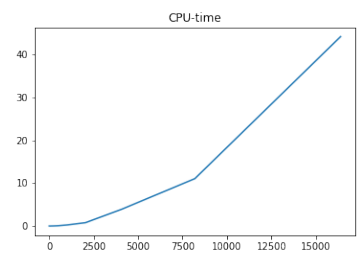
$$\boldsymbol{\alpha_1=\alpha_2=1}$$





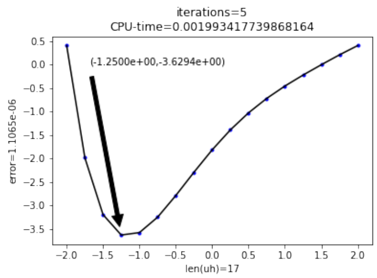
Here is how much time and how many iterations it took

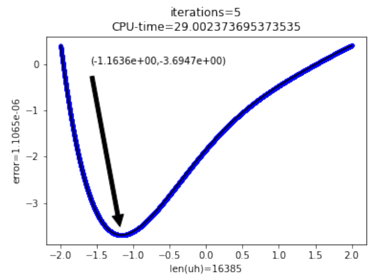




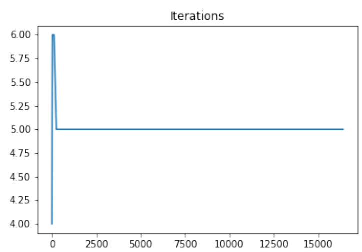
Points	Iterations	Time
9	7	0.002991
17	8	0.006265
33	8	0.021313
65	8	0.009007
129	8	0.021977
257	8	0.035761
513	8	0.076134
1025	7	0.251534
2049	7	0.806048
4097	7	2.803779
8193	7	10.70538
16385	7	47.35675

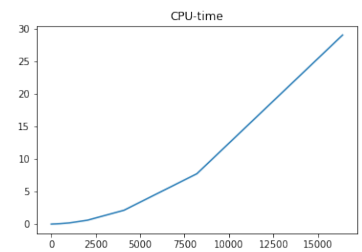
α_1=α_2=**2**





Here is how much time and how many iterations it took





Points	Iterations	Time
9	4	0.000997
17	5	0.001992
33	6	0.005227
65	6	0.006983
129	6	0.018829
257	5	0.028190
513	5	0.063062
1025	5	0.190287
2049	5	0.596706
4097	5	2.097174
8193	5	7.826753
16385	5	28.32951

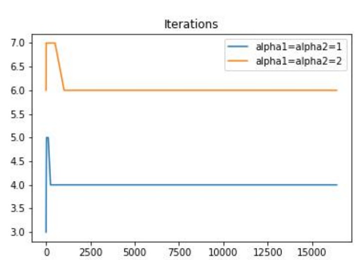
Comparison

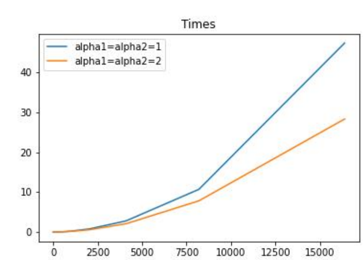
In this subsection we will compare the number of iterations and CPU time required to attain a solution of our problem with pseudo residual less than 10^{-8}. As before, I highlighted in green the number of iterations and in blue the CPU time (always expressed in seconds)

Points	Iterations1	Iterations2
9	4	7
17	5	8
33	6	8
65	6	8
129	6	8
257	5	8
513	5	8
1025	5	7
2049	5	7
4097	5	7
8193	5	7
16385	5	7

Points	Times1	Times2
9	0.002991	0.000997
17	0.006265	0.001992
33	0.021313	0.005227
65	0.009007	0.006983
129	0.021977	0.018829
257	0.035761	0.028190
513	0.076134	0.063062
1025	0.251534	0.190287
2049	0.806048	0.596706
4097	2.803779	2.097174
8193	10.70538	7.826753
16385	47.35675	28.32951

We can graphically see the data as follows:

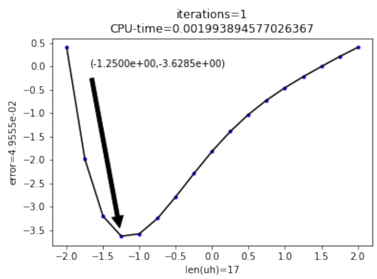


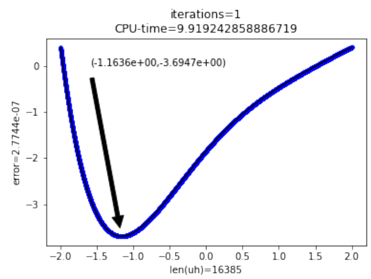


Task 1.7

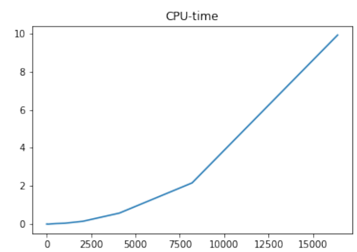
In this task, we apply the full_mg_1d function one. As always, I printed some solutions. Note that every page has different parameters α_1 and α_2 and ν. Moreover, by the fact that we apply the function one we are interesten in investigations over residuals and pseudo-residuals. I computed the infty norm of them

$$\alpha_1 = \alpha_2 = \mathbf{1},\ \nu = \mathbf{1}$$



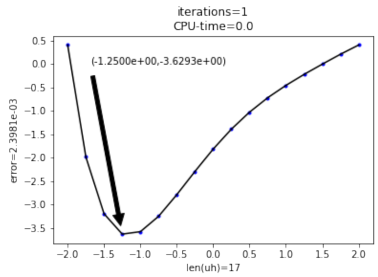


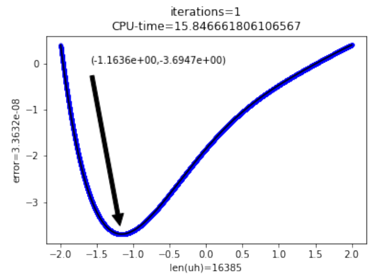
Here is how much time it took



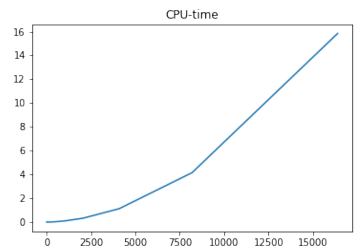
Points	Time	Residuals	Pseudo-residuals
9	0.000995	0.126316	0.054281
17	0.000998	0.244009	0.049554
33	0.008327	1.311873	0.027150
65	0.003988	2.472116	0.009656
129	0.006981	3.362213	0.003283
257	0.008975	3.937192	0.000961
513	0.031501	4.274821	0.000260
1025	0.043539	4.461994	6.80e-05
2049	0.148341	4.562035	1.74e-05
4097	0.599726	4.614246	4.40e-06
8193	2.155994	4.641073	1.10e-06
16385	9.130884	4.654719	2.77e-07

α_1=α_2=1, ν=2



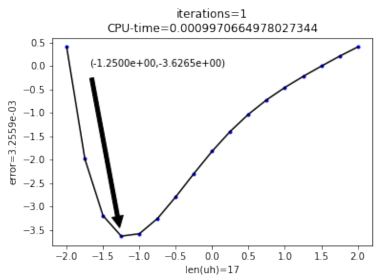


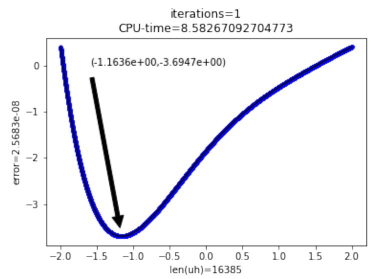
Here is how much time it took



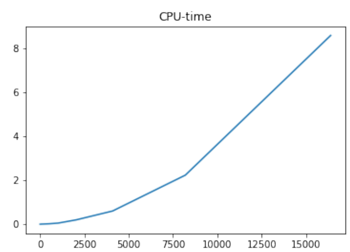
Points	Time	Residuals	Pseudo-residuals
9	0.0	0.002492	0.005277
17	0.015532	0.014194	0.002398
33	0.0	0.096103	0.003185
65	0.018258	0.214694	0.001384
129	0.002067	0.298895	0.000441
257	0.015621	0.348026	0.000123
513	0.048628	0.374443	3.26e-05
1025	0.078115	0.388139	8.39e-06
2049	0.343666	0.395119	2.12e-06
4097	1.124156	0.398644	5.35e-07
8193	4.017902	0.400416	1.34e-07
16385	15.46817	0.401305	3.36e-08

$\alpha_1 = \alpha_2 = \mathbf{2},\ \nu = \mathbf{1}$



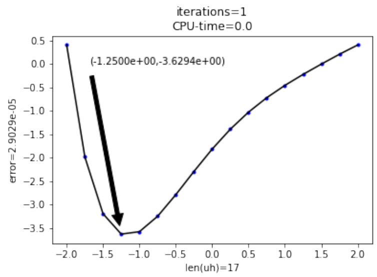


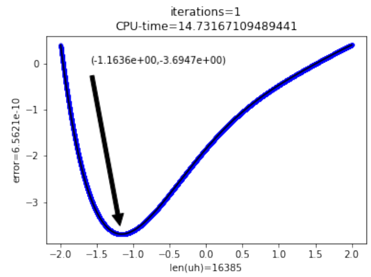
Here is how much time it took



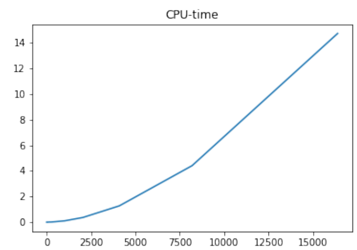
Points	Time	Residuals	Pseudo-residuals
9	0.002899	0.008032	0.002008
17	0.000998	0.052093	0.003255
33	0.0	0.168694	0.002635
65	0.0	0.278040	0.001086
129	0.010624	0.349291	0.000341
257	0.014647	0.389161	9.50e-05
513	0.019601	0.410079	2.50e-05
1025	0.062485	0.420760	6.42e-06
2049	0.172967	0.426152	1.62e-06
4097	0.592068	0.428860	4.08e-07
8193	2.121579	0.430217	1.02e-07
16385	7.705162	0.430896	2.56e-08

$$\alpha_1\mathbf{=}\alpha_2\mathbf{=2,}\ \nu\mathbf{=2}$$





Here is how much time it took



Points	Time	Residuals	Pseudo-residuals
9	0.007496	4.52e-05	1.13e-05
17	0.002119	0.000464	2.90e-05
33	0.003989	0.002863	4.47e-05
65	0.007196	0.006013	2.34e-05
129	0.010608	0.008294	8.10e-06
257	0.021040	0.009614	2.34e-06
513	0.045679	0.010312	6.29e-07
1025	0.120795	0.010670	1.62e-07
2049	0.360009	0.010850	4.13e-08
4097	1.212559	0.010941	1.04e-08
8193	4.264894	0.010986	2.61e-09
16385	16.19967	0.011009	6.56e-10

Comparison

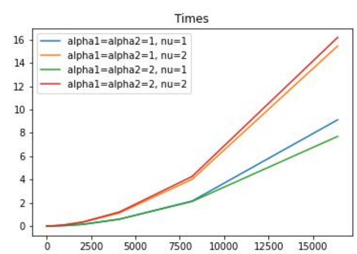
In this subsection, we will compare the CPU time required to apply the full_mg. As before, I highlighted in blue the CPU time (always expressed in seconds). Moreover, I highlighted in grey the residuals and in red the pseudo-residual.

Points	Time (1,1,1)	Times (1,1,2)	Times (2,2,1)	Times (2,2,2)
9	0.000995	0.0	0.002899	0.007496
17	0.000998	0.015532	0.000998	0.002119
33	0.008327	0.0	0.0	0.003989
65	0.003988	0.018258	0.0	0.007196
129	0.006981	0.002067	0.010624	0.010608
257	0.008975	0.015621	0.014647	0.021040
513	0.031501	0.048628	0.019601	0.045679
1025	0.043539	0.078115	0.062485	0.120795
2049	0.148341	0.343666	0.172967	0.360009
4097	0.599726	1.124156	0.592068	1.212559
8193	2.155994	4.017902	2.121579	4.264894
16385	9.130884	15.46817	7.705162	16.19967

Points	Residuals (1,1,1)	Residuals (1,1,2)	Residuals (2,2,1)	Residuals (2,2,2)
9	0.126316	0.002492	0.008032	4.52e-05
17	0.244009	0.014194	0.052093	0.000464
33	1.311873	0.096103	0.168694	0.002863
65	2.472116	0.214694	0.278040	0.006013
129	3.362213	0.298895	0.349291	0.008294
257	3.937192	0.348026	0.389161	0.009614
513	4.274821	0.374443	0.410079	0.010312
1025	4.461994	0.388139	0.420760	0.010670
2049	4.562035	0.395119	0.426152	0.010850
4097	4.614246	0.398644	0.428860	0.010941
8193	4.641073	0.400416	0.430217	0.010986
16385	4.654719	0.401305	0.430896	0.011009

Points	Pseudo-residuals (1,1,1)	Pseudo-residuals (1,1,2)	Pseudo-residuals (2,2,1)	Pseudo-residuals (2,2,2)
9	0.054281	0.005277	0.002008	1.13e-05
17	0.049554	0.002398	0.003255	2.90e-05
33	0.027150	0.003185	0.002635	4.47e-05
65	0.009656	0.001384	0.001086	2.34e-05
129	0.003283	0.000441	0.000341	8.10e-06
257	0.000961	0.000123	9.50e-05	2.34e-06
513	0.000260	3.26e-05	2.50e-05	6.29e-07
1025	6.80e-05	8.39e-06	6.42e-06	1.62e-07
2049	1.74e-05	2.12e-06	1.62e-06	4.13e-08
4097	4.40e-06	5.35e-07	4.08e-07	1.04e-08
8193	1.10e-06	1.34e-07	1.02e-07	2.61e-09
16385	2.77e-07	3.36e-08	2.56e-08	6.56e-10

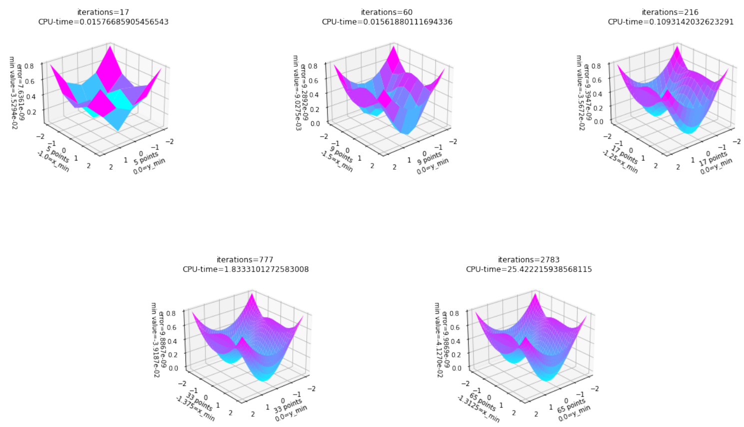
We can graphically see the data as follows: Note that the minimum value



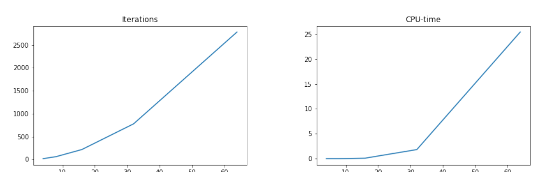
that the function can attain is printed inside every plot of the function

1 Task 2.2(i)

In this task, we apply the one-step Gauss-Seidel, but 2d, function with tol=10^{-8} on the pseudo_residual. In particular, I printed the graph of the solutions given by the function w.r.t. some len(uh).



It the following graphs and table, we can see the number of iterations of the one-step Gauss-Seidel and the CPU time (expressed in seconds) needed to attain a pseudo residual smaller 10^{-8}. Note that len(uh) are of the form $(2^n)+1$

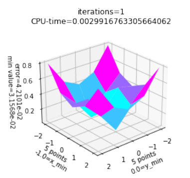


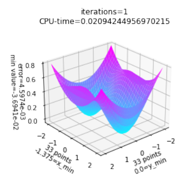
Points	Iterations	Time
5	17	0.015766
9	60	0.015618
17	216	0.109314
33	777	1.833310
65	2783	25.42221

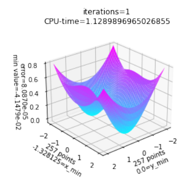
2 Task 2.2(ii)

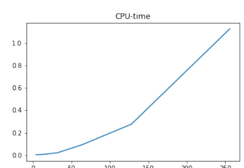
In this task, we apply the full_mg_2d function one. As always, I printed some solutions. Note that every page has different parameters α_1 and α_2 and ν. Moreover, by the fact that we apply the function one we are interested in investigations over residuals and pseudo-residuals. I computed the infty norm of them

$$\alpha_1 = \alpha_2 = \mathbf{1},\ \nu = \mathbf{1}$$



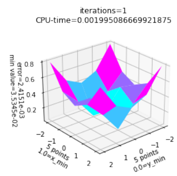


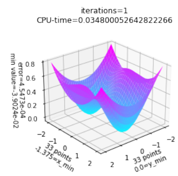


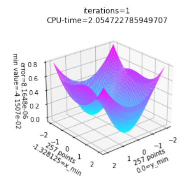


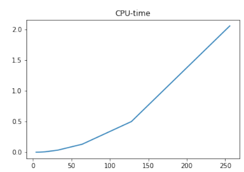
Points	Time	Residual	Pseudo-residual
5	0.002991	0.040263	0.042100
9	0.003988	0.204709	0.050505
17	0.007979	0.244360	0.016883
33	0.020942	0.294235	0.004597
65	0.092752	0.316209	0.001235
129	0.276298	0.326578	0.000318
257	1.128989	0.331244	8.08e-05

α_1=α_2=**1**, ν=**2**



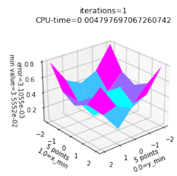


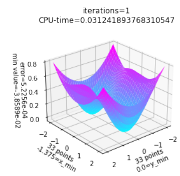


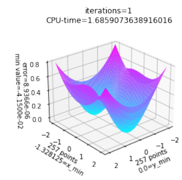


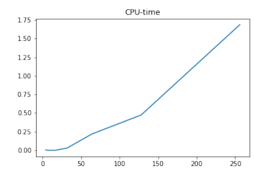
Points	Time	Residual	Pseudo-residual
5	0.001995	0.003652	0.002415
9	0.002991	0.017716	0.003984
17	0.008976	0.025082	0.001547
33	0.034800	0.029102	0.000454
65	0.131700	0.031815	0.000124
129	0.500063	0.032948	3.21e-05
257	2.054722	0.033443	8.16e-06

α_1=α_2=**2**, ν=**1**



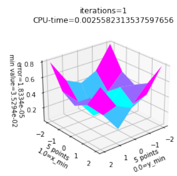


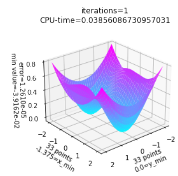


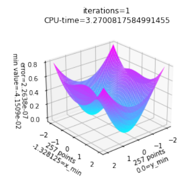


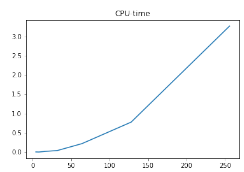
Points	Time	Residual	Pseudo-residual
5	0.004797	0.005033	0.003105
9	0.0	0.033411	0.005397
17	0.0	0.041836	0.001841
33	0.031241	0.043560	0.000522
65	0.219254	0.044760	0.000137
129	0.474570	0.045514	3.53e-05
257	1.685907	0.045928	8.93e-06

α_1=α_2=**2**, ν=**2**









Points	Time	Residual	Pseudo-residual
5	0.002558	1.65e-05	1.833426
9	0.0	0.001278	0.000202
17	0.015617	0.001452	5.82e-05
33	0.038560	0.001117	1.26e-05
65	0.217633	0.001042	3.39e-06
129	0.776211	0.001016	8.89e-07
257	3.270081	0.001028	2.26e-07

Comparison

In this subsection, we will compare the CPU time required to apply the full_mg. As before, I highlighted in blue the CPU time (always expressed in seconds). Moreover, I highlighted in grey the residuals and in red the pseudo-residual.

Points	Time (1,1,1)	Times (1,1,2)	Times (2,2,1)	Times (2,2,2)
5	0.002991	0.001995	0.004797	0.002558
9	0.003988	0.002991	0.0	0.0
17	0.007979	0.008976	0.0	0.015617
33	0.020942	0.034800	0.031241	0.038560
65	0.092752	0.131700	0.219254	0.217633
129	0.276298	0.500063	0.474570	0.776211
257	1.128989	2.054722	1.685907	3.270081

Points	Residual (1,1,1)	Residual (1,1,2)	Residual (2,2,1)	Residual (2,2,2)
5	0.040263	0.003652	0.005033	1.65e-05
9	0.204709	0.017716	0.033411	0.001278
17	0.244360	0.025082	0.041836	0.001452
33	0.294235	0.029102	0.043560	0.001117
65	0.316209	0.031815	0.044760	0.001042
129	0.326578	0.032948	0.045514	0.001016
257	0.331244	0.033443	0.045928	0.001028

Points	Pseudo-residual (1,1,1)	Pseudo-residual (1,1,2)	Pseudo-residual (2,2,1)	Pseudo-residual (2,2,2)
5	0.042100	0.002415	0.003105	1.833426
9	0.050505	0.003984	0.005397	0.000202
17	0.016883	0.001547	0.001841	5.82e-05
33	0.004597	0.000454	0.000522	1.26e-05
65	0.001235	0.000124	0.000137	3.39e-06
129	0.000318	3.21e-05	3.53e-05	8.89e-07
257	8.08e-05	8.16e-06	8.93e-06	2.26e-07