Introduction

The following report describes what Stefano Costa did for the third project of the scientific computing course.

The written code is strongly inspired by the directions left in class, following step by step all the algorithms left and suggested schemes.

I apologize for the length of the report, but I believe it is justified by the detailed graphical analysis of each method and all the results. This allows, in fact, a graphical interpretation of how each method works, and I found it very interesting to study them from this point of view.

As you read the report you will find that it is not necessary to read every verbal part, but it is sufficient to study the graphs given to interpret the efficiency results I obtained. Reading the report in this way takes but a few minutes.

Multigrid 1D

`jacobi_step_1d(uh, fh, omega)`

The following section investigates the weighted Jacobi method to achieve a tolerance of 10^{-8} in the 1D problem stated in the project.

Case: $\omega = 1/3$

h	l	**Iterations**	**Time**	$\min_{x\in\overline{\Omega}}$
0.125	3	200	0.0186s	-0.00575033
0.0625	4	620	0.08429s	-0.00620467
0.03125	5	1742	0.46619s	-0.00637584
0.015625	6	4665	2.92784s	-0.00640042
0.0078125	7	13910	15.5894s	-0.00640434

Investigation of the one-step weighted Jacobi method performance with $\omega = 1/3$

The results follow in graph form:

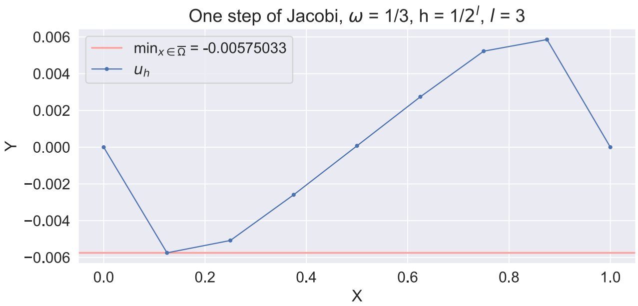

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.125$, $\omega = 1/3$

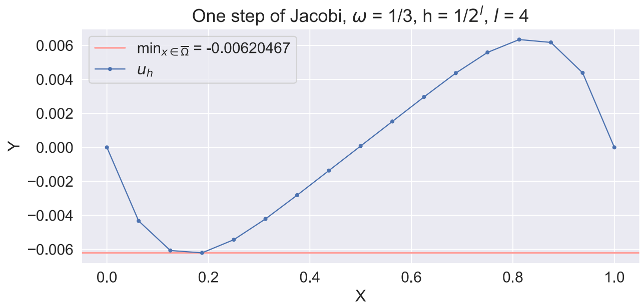

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.0625$, $\omega = 1/3$

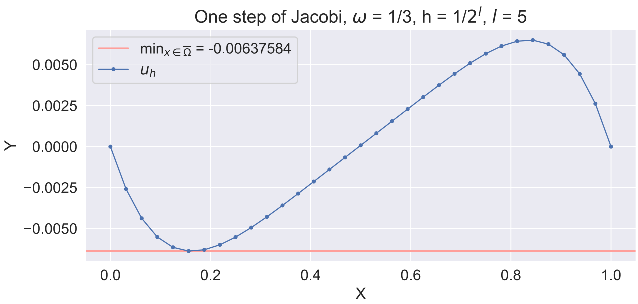

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.03125$, $\omega = 1/3$

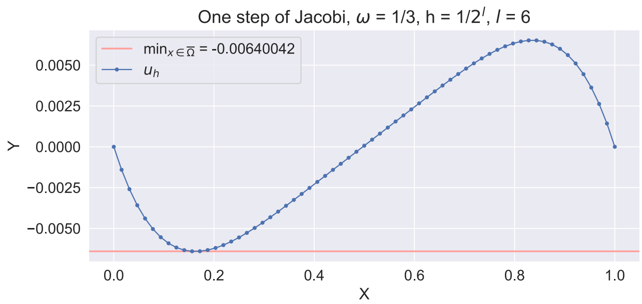

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.015625$, $\omega = 1/3$

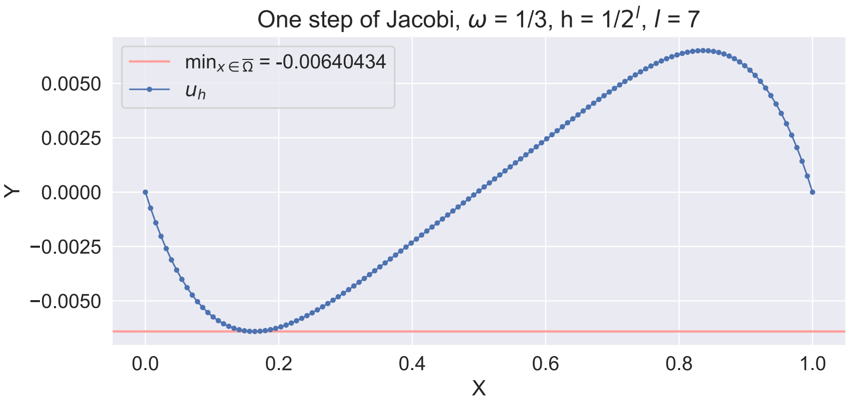

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.0078125$, $\omega = 1/3$

Plotting now the iterations, time and minimum values gives the following.

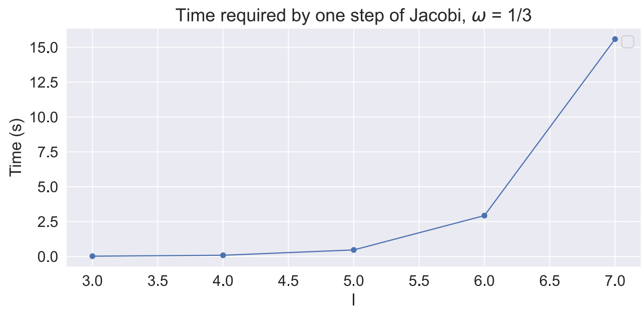

Figure: Graph of the time required by the weighted Jacobi method, $\omega = 1/3$

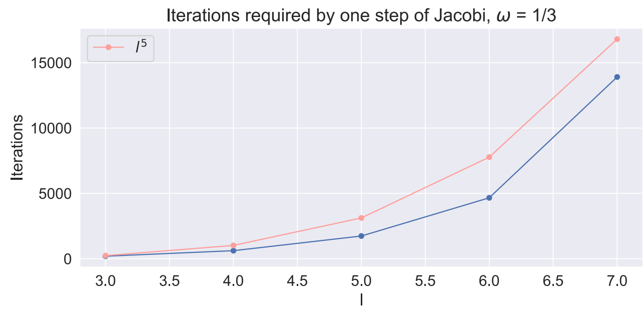

Figure: Graph of the iterations required by the weighted Jacobi method, $\omega = 1/3$

It can be seen that the number of iterations increases as the time required to conclude the computations as l increases.

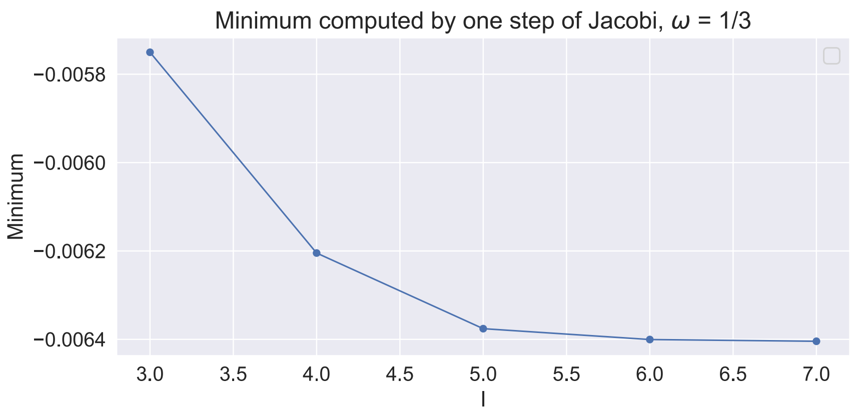

Figure: Graph of the trend of the minimum computed by the weighted Jacobi method, $\omega = 1/3$

The evolution of the minimum, on the other hand, follows an opposite trend to that of the iterations: at the beginning there is a large difference, while at the end the difference is minimal, in the face of a large computational effort. In the face of what is shown in the graph, we are suggested to stop at $l = 6$ or even $l = 5$ to avoid wasting much time for little perceptible improvement.

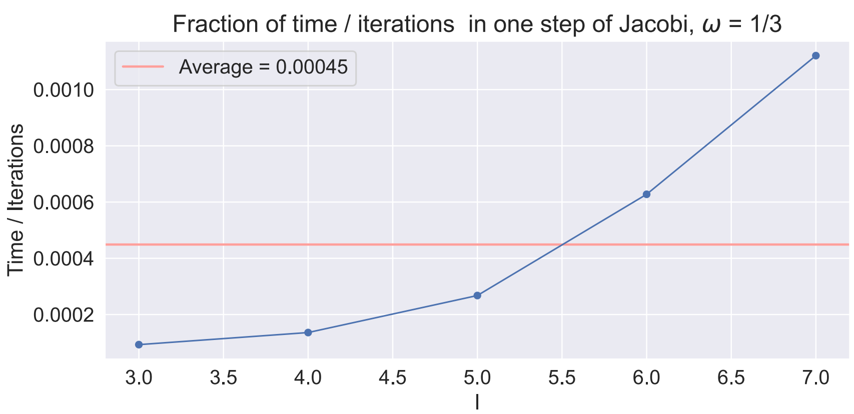

Figure: Graph of the fraction time / iterations required by the weighted Jacobi method, $\omega = 1/3$

It would seem that the ratio of time to iterations increases as l increases. It is possible to assume that for large l, this fraction stabilizes.

In any case, it seems that the best ratio occurs for small values of l, supporting what was said earlier: it is convenient to use this method for small values of l, that is, (relatively) large values of h.

Case: $\omega = 2/3$

h	l	**Iterations**	**Time**	$\min_{x\in\overline{\Omega}}$
0.125	3	111	0.01596s	-0.00575027
0.0625	4	356	0.06599s	-0.0062043
0.03125	5	1057	0.2799s	-0.00637463
0.015625	6	2803	1.6445s	-0.00639951
0.0078125	7	8026	11.12707s	-0.00641078

Investigation of the one-step weighted Jacobi method performance with $\omega = 2/3$

The results follow in graph form:

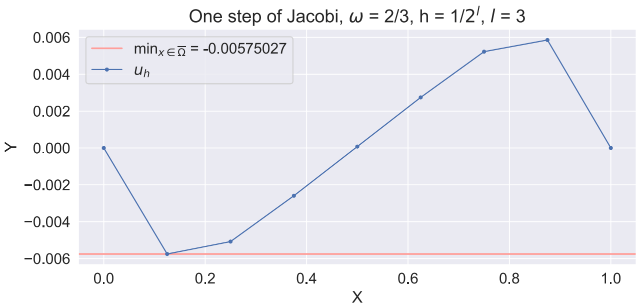

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.125$, $\omega = 2/3$

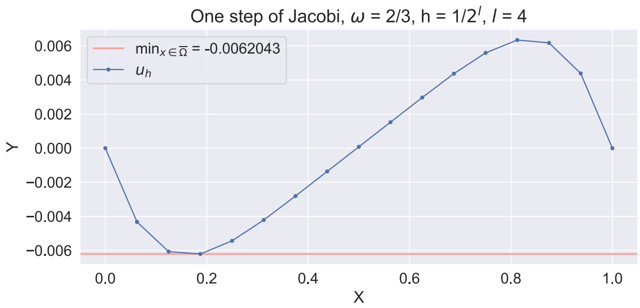

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.0625$, $\omega = 2/3$

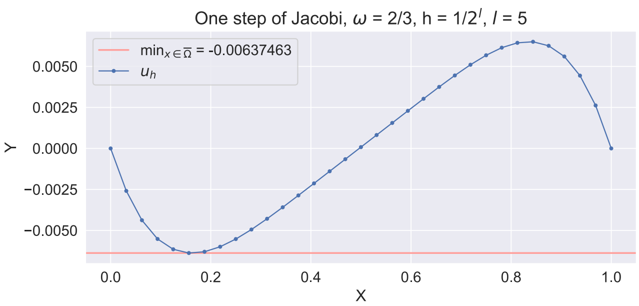

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.03125$, $\omega = 2/3$

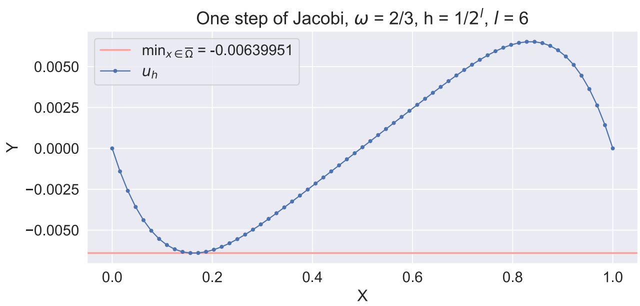

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.015625$, $\omega = 2/3$

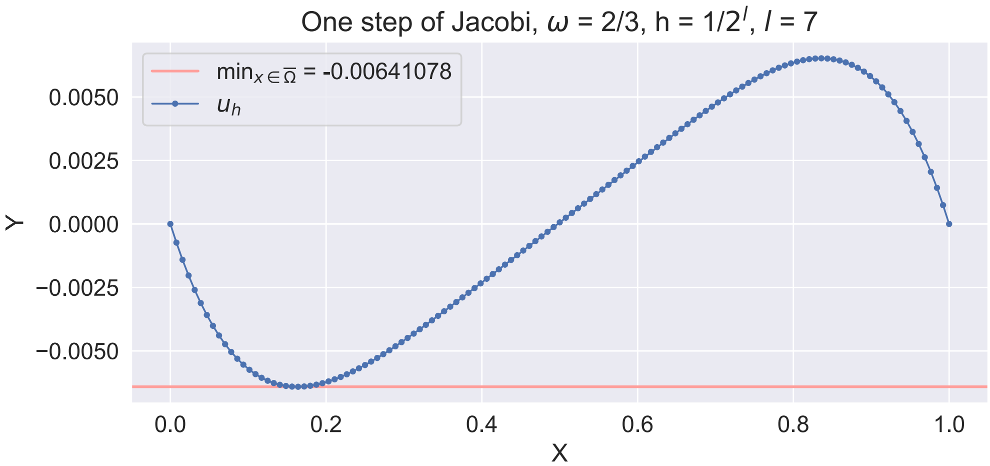

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.0078125$, $\omega = 2/3$

Plotting now the iterations, time and minimum values gives the following.

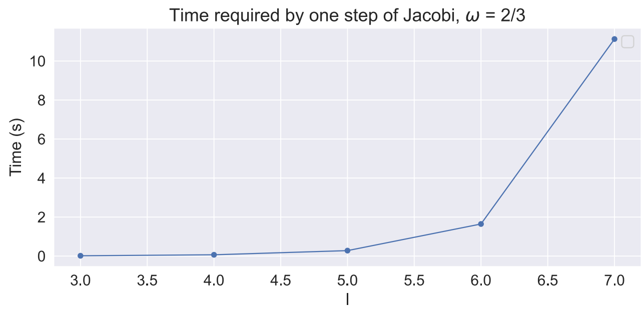

Figure: Graph of the time required by the weighted Jacobi method, $\omega = 2/3$

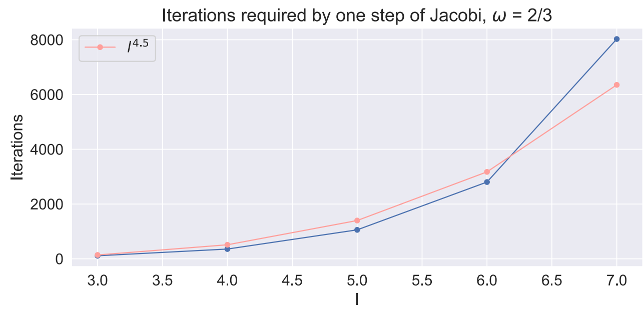

Figure: Graph of the iterations required by the weighted Jacobi method, $\omega = 2/3$

As can be seen from the graph of iterations, these grow less quickly than the $\omega = 1/3$ case, which used to grow about the same as l^5, whereas now it grows as about $l^{4.5}$.

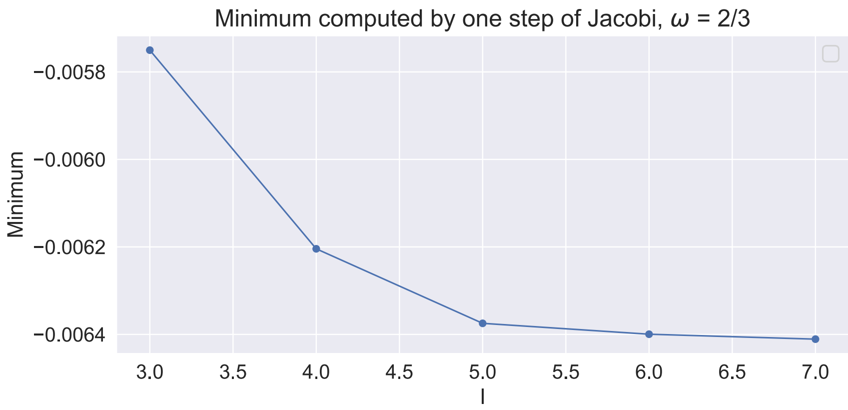

Figure: Graph of the trend of the minimum computed by the weighted Jacobi method, $\omega = 2/3$

Again, however, the trend of the minimum suggests that this method is efficient only for small l.

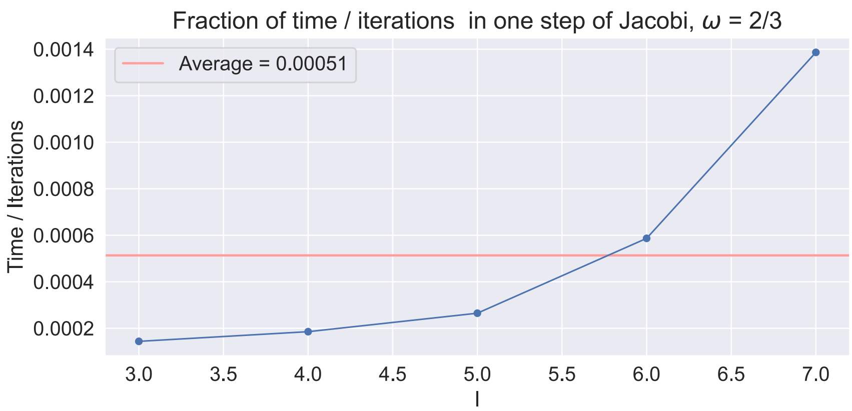

Figure: Graph of the fraction time / iterations required by the weighted Jacobi method, $\omega = 2/3$

Although the number of iterations has greatly decreased, also the empirical times have decreased, so the time/iteration ratio has sligthly increased.

Obviously, to do a more in-depth analysis, it would be necessary to do more tests, to take into account mean and standard deviation, and to see if indeed the times are different. However, with the data we have it is possible to say that the choice $\omega = 2/3$ is the best both in term of iterations and time.

Two-grid correction scheme

The following section investigates the two-grid correction scheme (with ten steps of the weighted Jacobi method on Ω_{2h}) to achieve a tolerance of 10^{-8} in the 1D problem stated in the project.

Code

Only for this case the code is given, since in the `.py` of the project it was not left (because it was not required). Only the $\omega = 1/3$ case is reported since the other case is identical except for the value of omega...

```python
for l in range(3,8):
    n = 2**l
    h = 1 / n

    x = np.linspace(0, 1, n+1)
    fh = np.zeros_like(x)
    fh[:] = f(x)
    fh[0], fh[-1] = 0, 0
    uh = np.zeros_like(x)

    tol = 1e-8
    kJ1h = 0
    kJ12h = 0

    tic = time()
    checkJ1 = tol + 1

    while checkJ1 > tol:
        v = uh.copy()

        # pre-smoothing
        jacobi_step_1d(uh=uh,fh=fh, omega=1/3)
        kJ1h += 1

        # compute rh
        Ahuh = Ahuh_multiplication(uh)
        rh = fh-Ahuh

        # restriction
        r2h = restriction_operator(rh)

        # Solution with 10 steps jacobi
        jacobi_steps = 10
        e2h = np.zeros_like(r2h)
        for i in range(jacobi_steps):
            jacobi_step_1d(uh=e2h,fh=r2h, omega=1/3)
            kJ12h += 1

        # prolongation
        eh = prolongation_operator(e2h)

        # correction
        uh += eh

        # post-smoothing
        err = jacobi_step_1d(uh=uh, fh=fh, omega=1/3)
        kJ1h+=1

        checkJ1 = la.norm(uh-v, np.inf)

    elapsed = round(time() - tic, 5)
```

Analysis of the results follows.

Case: $\omega = 1/3$

h	l	Iterations on Ω_h	Iterations on Ω_{2h}	Time	$\min_{x\in\overline{\Omega}}$
0.125	3	32	160	0.00898s	-0.00575021
0.0625	4	56	280	0.06427s	-0.00620395
0.03125	5	182	910	0.21306s	-0.00637343
0.015625	6	576	2880	1.17574s	-0.00639545
0.0078125	7	1728	8640	8.15326	-0.00640788

Investigation of the two grid correction performance with $\omega = 1/3$

Note that the iterations on Ω_h refer to each individual Jacobi step on Ω_h.

The results follow in graph form:

Figure: Graph of the solution approximated by the two grid correction, $h = 0.125$, $\omega = 1/3$

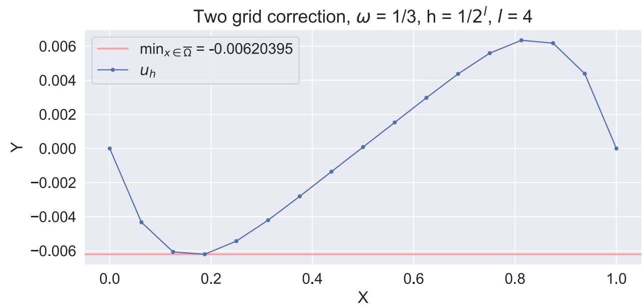

Figure: Graph of the solution approximated by the two grid correction, $h = 0.0625$, $\omega = 1/3$

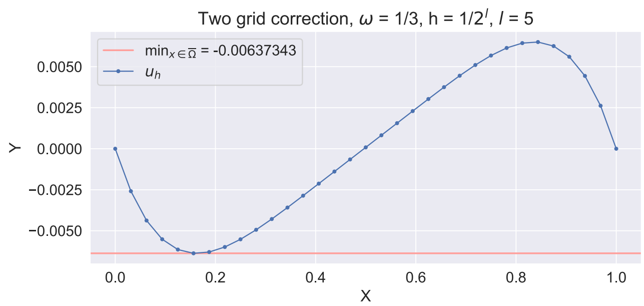

Figure: Graph of the solution approximated by the two grid correction, $h = 0.03125$, $\omega = 1/3$

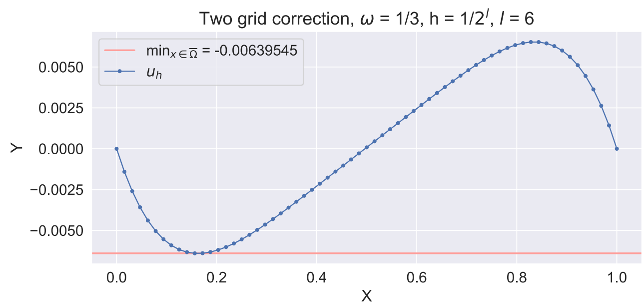

Figure: Graph of the solution approximated by the two grid correction, $h = 0.015625$, $\omega = 1/3$

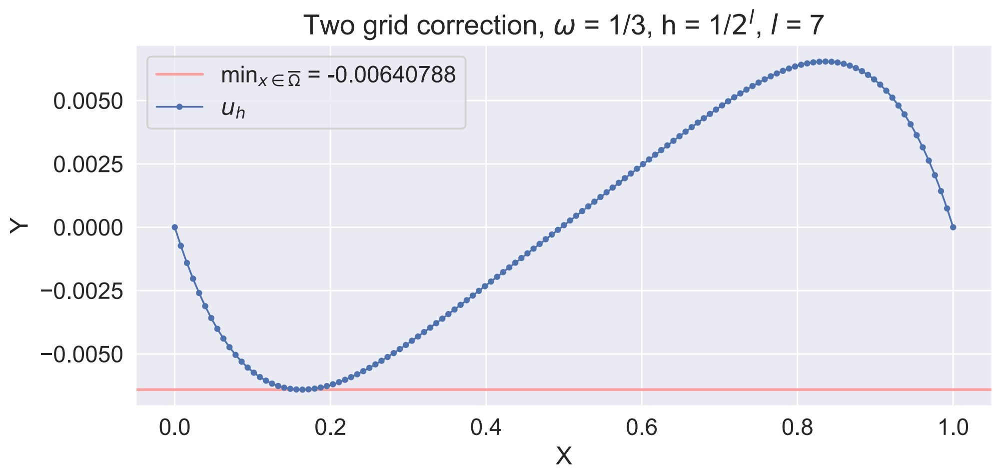

Figure: Graph of the solution approximated by the two grid correction, $h = 0.0078125$, $\omega = 1/3$

Plotting now the iterations, time and minimum values gives the following.

Figure: Graph of the iterations required by the two grid correction on Ω_h, $\omega = 1/3$

Figure: Graph of the iterations required by the two grid correction on Ω_{2h}, $\omega = 1/3$

Clearly we have a trend of less growth on Ω_h, as the approach is precisely to do more iterations on Ω_{2h}.

Figure: Graph of the time required by the two grid correction, $\omega = 1/3$

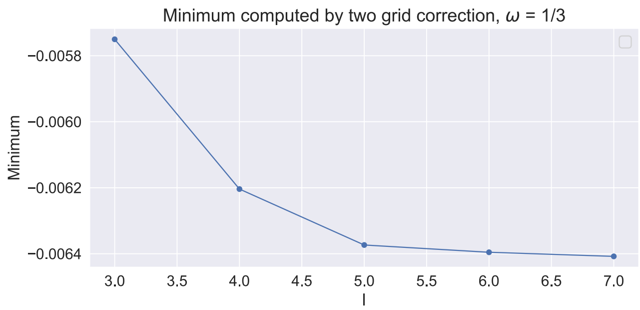

Figure: Graph of the minimum computed by the two grid correction, $\omega = 1/3$

Evidently, time grows exactly as both iteration graphs. Again, as in the previous cases, there is a large increase in time for large values of l against a small change in the minimum value. However, this technique takes much less time.

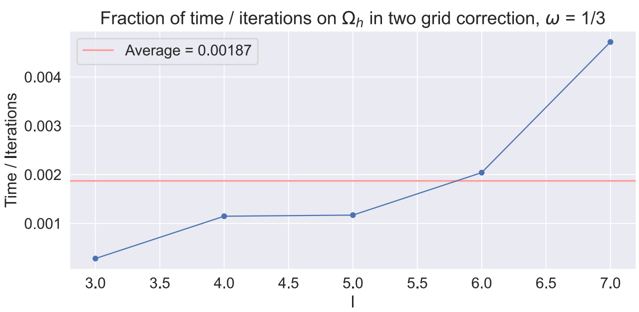

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_h, $\omega = 1/3$

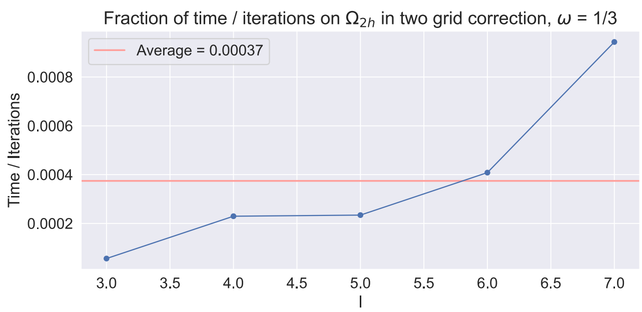

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_{2h}, $\omega = 1/3$

As expected, the time required per iteration is much less (about 10 times as much) on Ω_{2h}. It is also noticeable that the fraction trend is very similar on the two grids.

Case: $\omega = 2/3$

h	l	**Iterations on** Ω_h	**Iterations on** Ω_{2h}	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.125	3	14	70	0.00499s	-0.00575021
0.0625	4	32	160	0.03391s	-0.00620394
0.03125	5	100	500	0.10871s	-0.0063734
0.015625	6	324	1620	0.72696s	-0.00639533
0.0078125	7	1008	5040	3.51848s	-0.00640738

Investigation of the two grid correction performance with $\omega = 2/3$

The results follow in graph form:

Figure: Graph of the solution approximated by the two grid correction, $h = 0.125$, $\omega = 2/3$

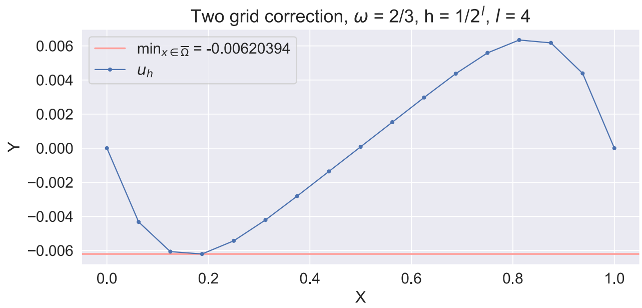

Figure: Graph of the solution approximated by the two grid correction, $h = 0.0625$, $\omega = 2/3$

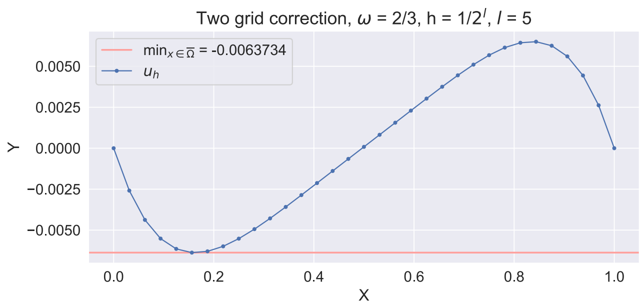

Figure: Graph of the solution approximated by the two grid correction, $h = 0.03125$, $\omega = 2/3$

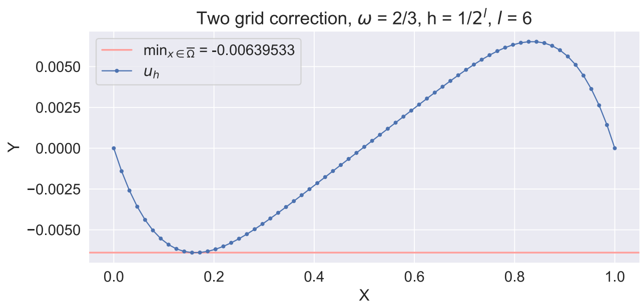

Figure: Graph of the solution approximated by the two grid correction, $h = 0.015625$, $\omega = 2/3$

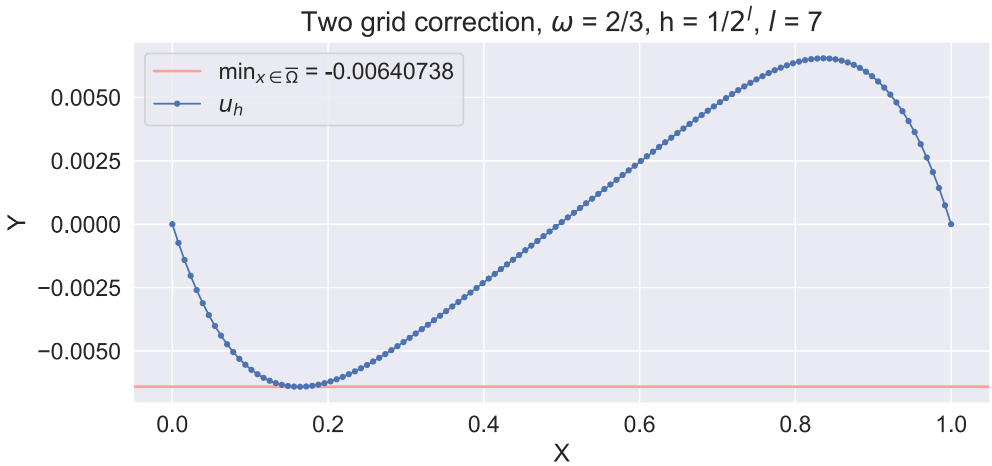

Figure: Graph of the solution approximated by the two grid correction, $h = 0.0078125$, $\omega = 2/3$

Plotting now the iterations, time and minimum values gives the following.

Figure: Graph of the iterations required by the two grid correction on Ω_h, $\omega = 2/3$

Figure: Graph of the iterations required by the two grid correction on Ω_{2h}, $\omega = 2/3$

Again there is a trend of less growth on Ω_h

Figure: Graph of the time required by the two grid correction, $\omega = 2/3$

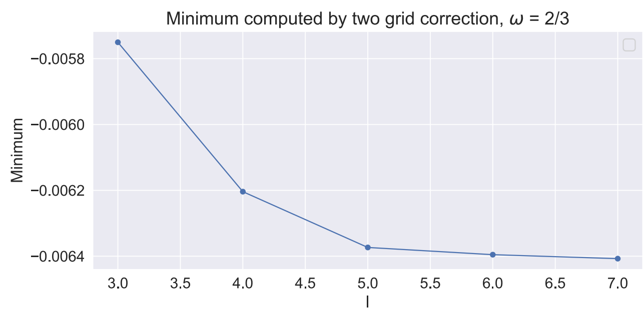

Figure: Graph of the minimum computed by the two grid correction, $\omega = 2/3$

The trend of the minimum and time is the same as the case before, although the time is less overall.

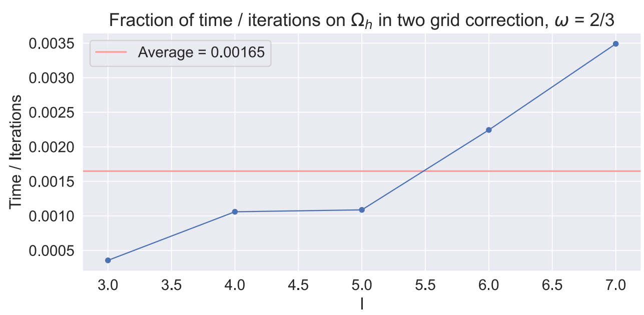

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_h, $\omega = 2/3$

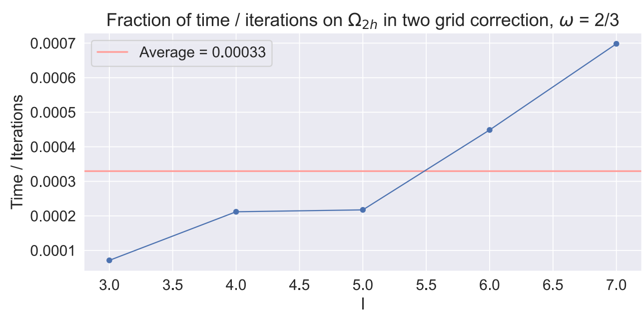

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_{2h}, $\omega = 2/3$

With the fraction time / iterations we have the same situation as in the $\omega = 1/3$ case.

Overall, it is evident that there is an improvement over the single step of the Jacobi method, but most importantly there is a remarkable improvement in terms of time, maximized in the case of $\omega = 2/3$.

`w_cycle_step_1d(uh, fh, omega, alpha1, alpha2)`

The following section investigates the W-cycle scheme to achieve a tolerance of 10^{-8} in the 1D problem stated in the project.

Case: $\alpha_1 = \alpha_2 = 1$, $\omega = 1/3$

h	l	**Iterations**	**Time**	$\min_{x\in\overline{\Omega}}$
0.125	3	14	0.00898s	-0.00575019
0.0625	4	13	0.01599s	-0.00620394
0.03125	5	11	0.03593s	-0.00637337
0.015625	6	10	0.0908s	-0.00639521
0.0078125	7	8	0.16484s	-0.00640688
0.00390625	8	7	0.42933s	-0.00640827
0.001953125	9	5	0.55834s	-0.00640878
0.0009765625	10	4	0.90296s	-0.00640892
0.00048828125	11	3	2.23391s	-0.00640895
0.000244140625	12	2	2.79508s	-0.00640896
0.0001220703125	13	1	2.96425s	-0.00640896
6.103515625e-05	14	1	5.91273	-0.00640896

Investigation of the W-cycle performance with $\omega = 1/3$, $\alpha_1 = \alpha_2 = 1$

Results follow in graph form:

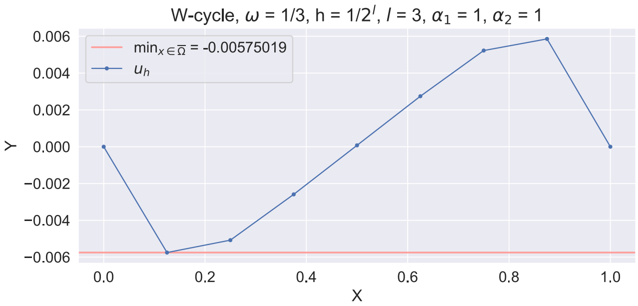

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

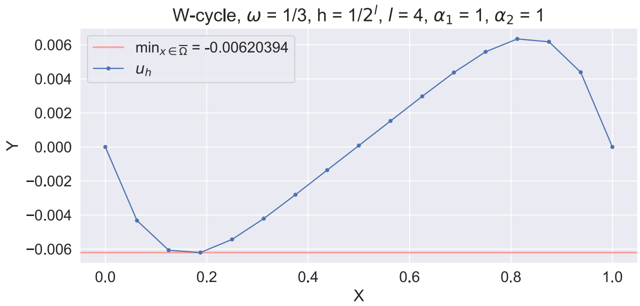

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

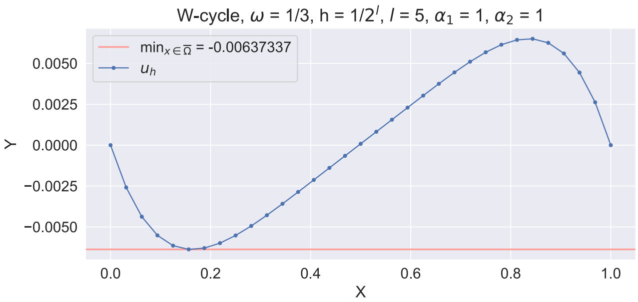

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

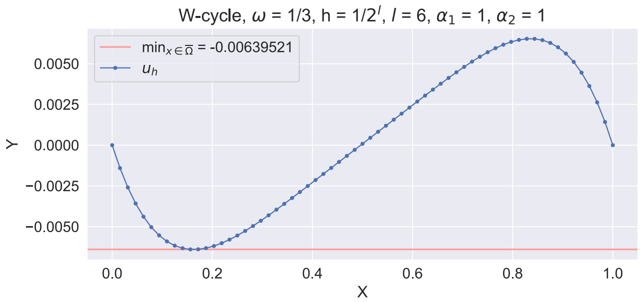

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

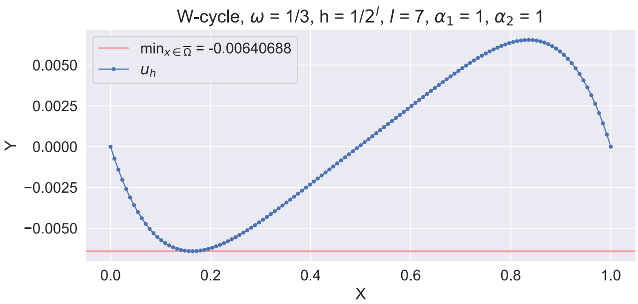

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0078125$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

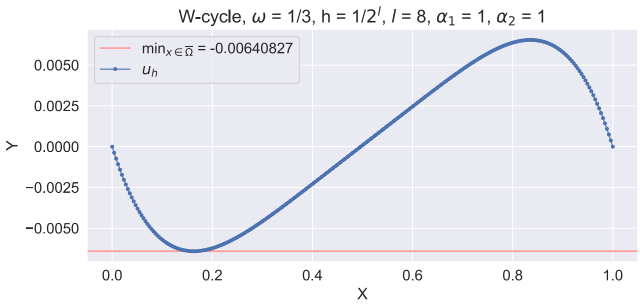

Figure: Graph of the solution approximated by the W-cycle, $h = 0.00390625$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

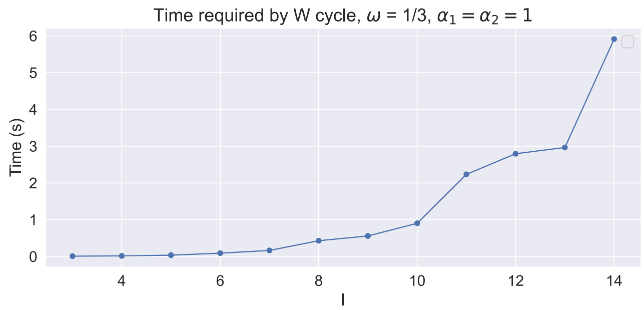

Figure: Graph of the time required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 1$

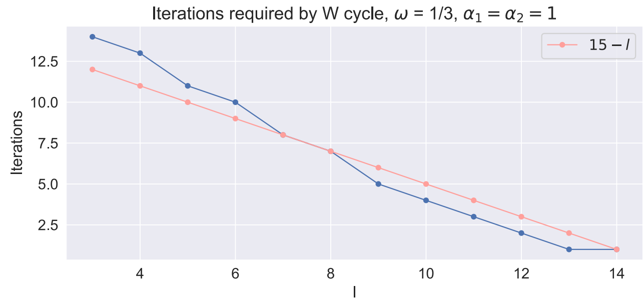

Figure: Graph of the iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 1$

It can be seen that although the time increases very moderately as the value of l increases, the number of iterations decreases.

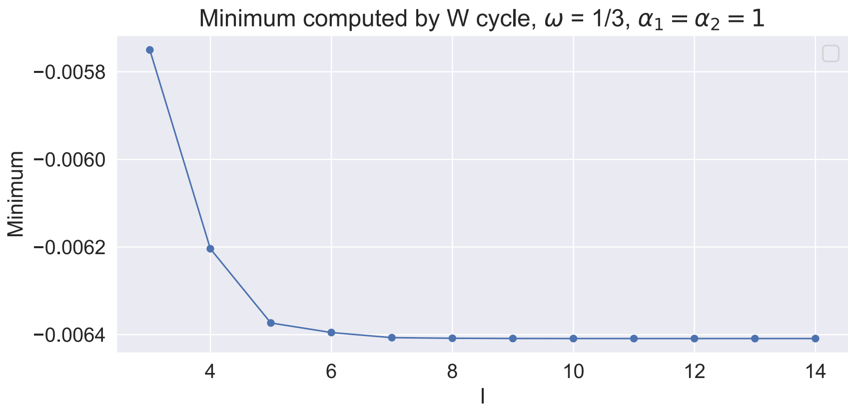

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

Note that in the last three cases the same value is found for the minimum.

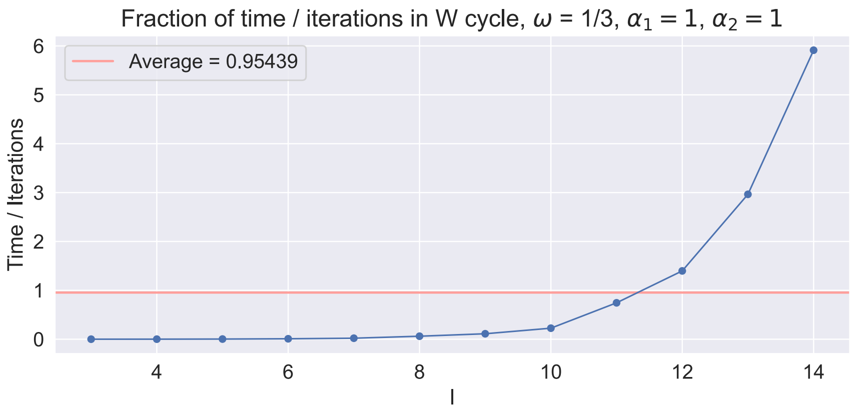

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

Clearly a very high value of the fraction time / iterations is obtained since there is a lowering of iterations with an increase in time. If even analyzing this fraction we would see that this method looks worse than the previous methods, comparing it with the other attributes we see that this method is much better: the times are much less with the same values of h and the same tolerance.

Case: $\alpha_1 = \alpha_2 = 2$, $\omega = 1/3$

h	l	**Iterations**	**Time**	$\min_{x\in\overline{\Omega}}$
0.125	3	8	0.00902s	-0.00575021
0.0625	4	7	0.01793s	-0.00620394
0.03125	5	6	0.02896s	-0.00637338
0.015625	6	5	0.07149s	-0.00639521
0.0078125	7	5	0.21196s	-0.00640688
0.00390625	8	4	0.31244s	-0.00640827
0.001953125	9	3	0.54945s	-0.00640878
0.0009765625	10	2	0.98856s	-0.00640892
0.00048828125	11	2	1.72281s	-0.00640895
0.000244140625	12	1	2.24766s	-0.00640896
0.0001220703125	13	1	4.54823s	-0.00640896
6.103515625e-05	14	1	9.45686s	-0.00640896

Investigation of the W-cycle performance with $\omega = 1/3$, $\alpha_1 = \alpha_2 = 2$

Results follow in graph form:

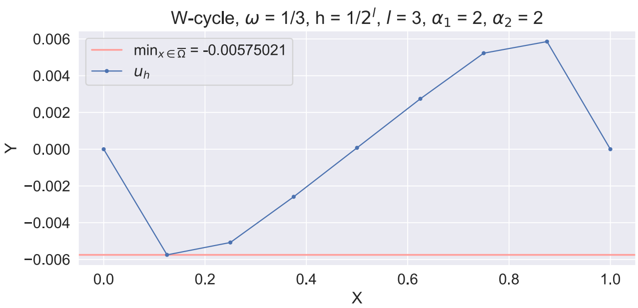

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

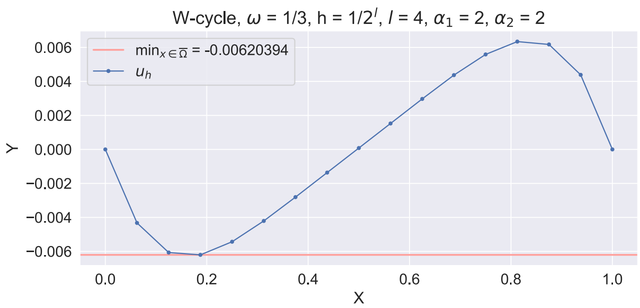

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

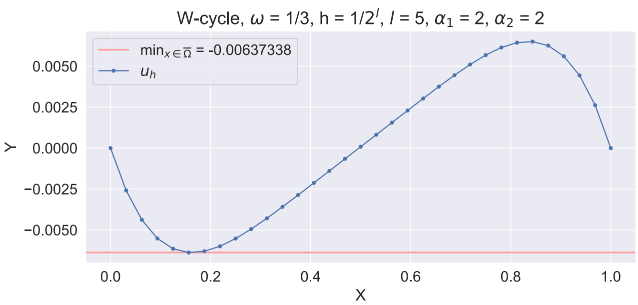

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

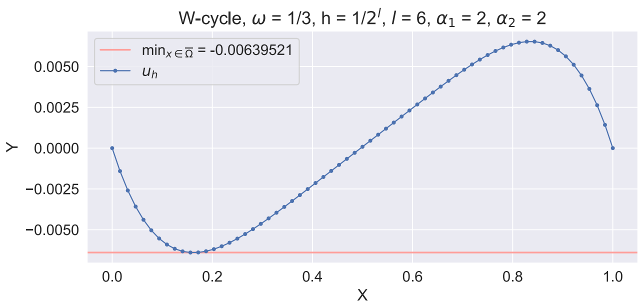

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

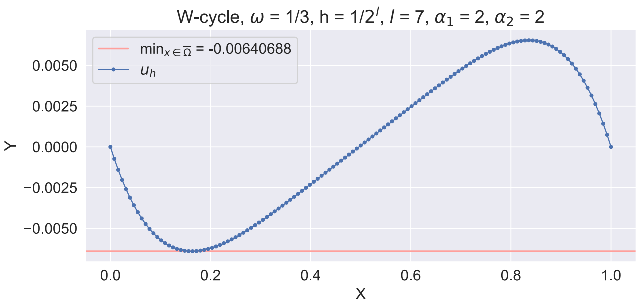

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0078125$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

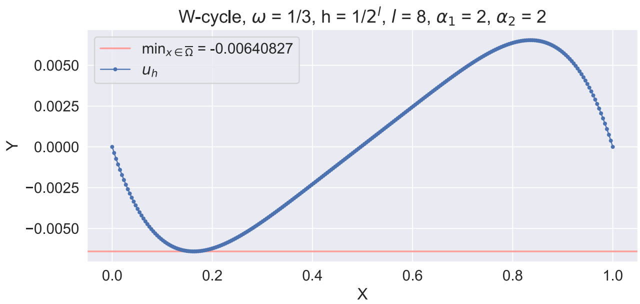

Figure: Graph of the solution approximated by the W-cycle, $h = 0.00390625$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

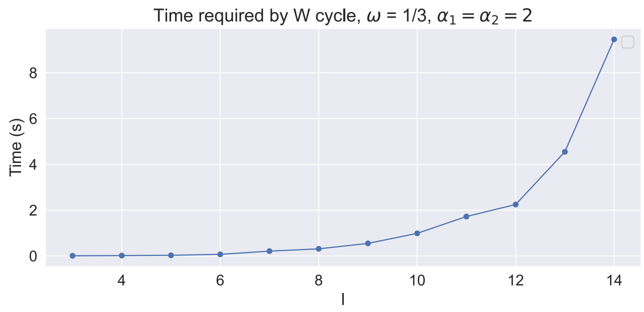

Figure: Graph of the time required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 2$

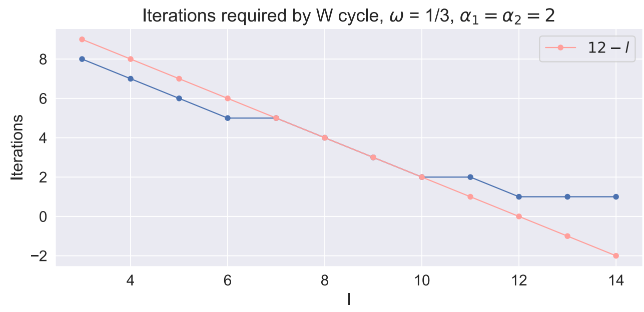

Figure: Graph of the iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 2$

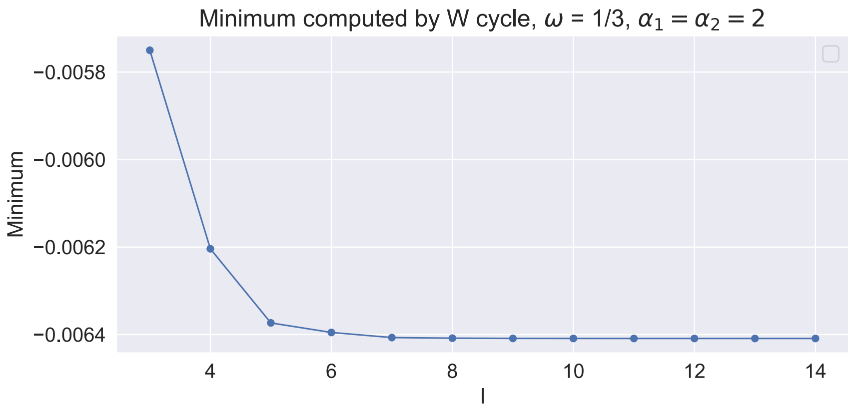

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

Note that in the last three cases the same value is found for the minimum.

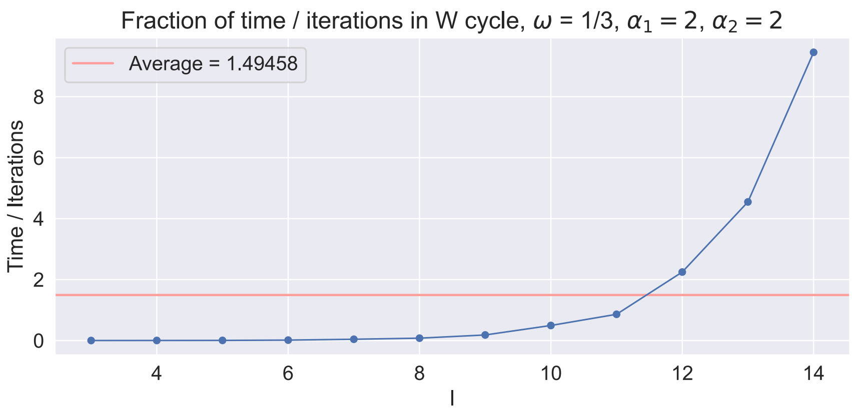

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

Overall, the same results are obtained as in the previous case. The notable difference is in terms of times (at equal iterations), which are significantly higher than the $\alpha_1 = \alpha_2 = 1$ case. So between the two cases the previous one is to be preferred.

Case: $\alpha_1 = \alpha_2 = 1$, $\omega = 2/3$

h	l	**Iterations**	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.125	3	7	0.00499s	-0.00575021
0.0625	4	6	0.00895s	-0.00620393
0.03125	5	6	0.02589s	-0.00637338
0.015625	6	6	0.04887s	-0.00639521
0.0078125	7	5	0.10176s	-0.00640688
0.00390625	8	4	0.19156s	-0.00640827
0.001953125	9	4	0.57345s	-0.00640878
0.0009765625	10	3	0.90831s	-0.00640892
0.00048828125	11	2	1.14874s	-0.00640895
0.000244140625	12	2	2.5875s	-0.00640896
0.0001220703125	13	1	3.27667s	-0.00640896
6.103515625e-05	14	1	6.07724	-0.00640896

Investigation of the W-cycle performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$

Results follow in graph form:

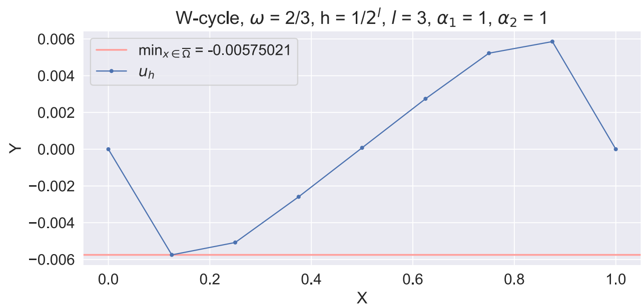

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

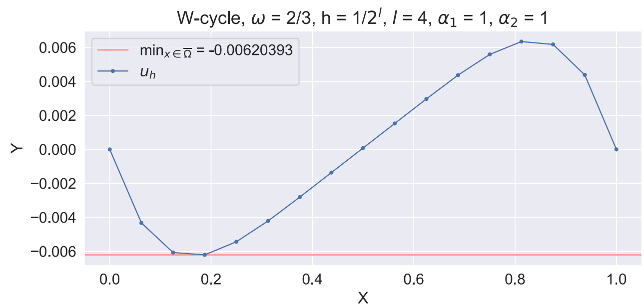

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

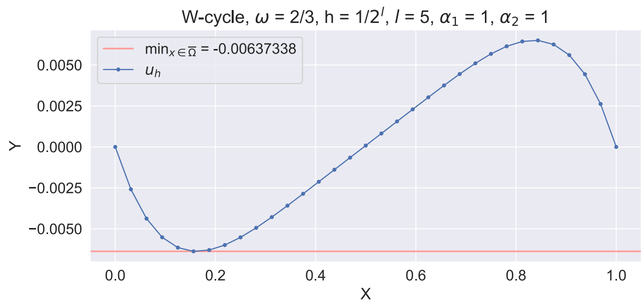

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

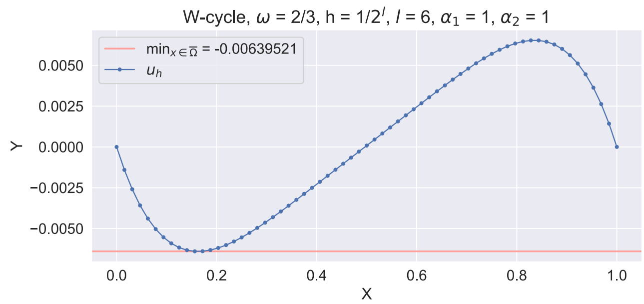

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

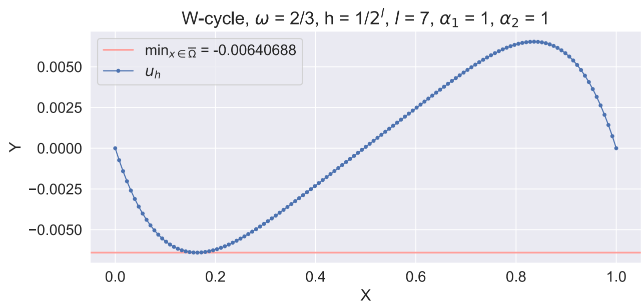

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0078125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

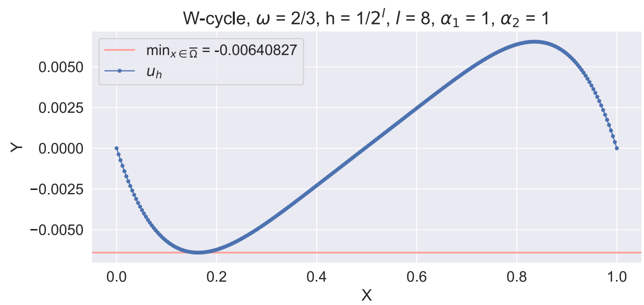

Figure: Graph of the solution approximated by the W-cycle, $h = 0.00390625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

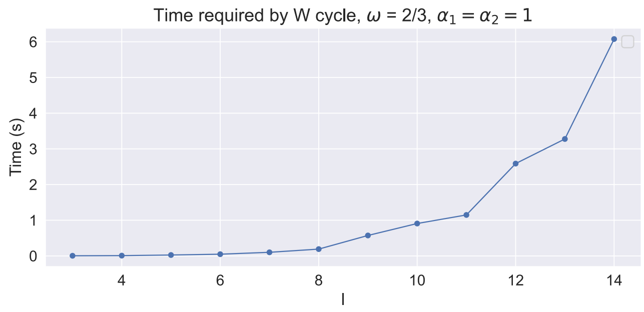

Figure: Graph of the time required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$

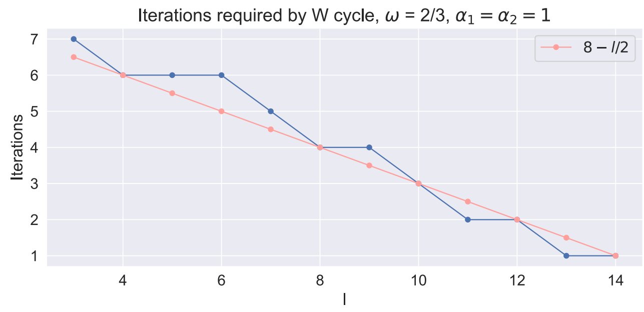

Figure: Graph of the iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$

Overall, we have the same behavior: decrease in iterations and increase in time. However, iterations decrease more slowly than the $\omega = 1/3$ case.

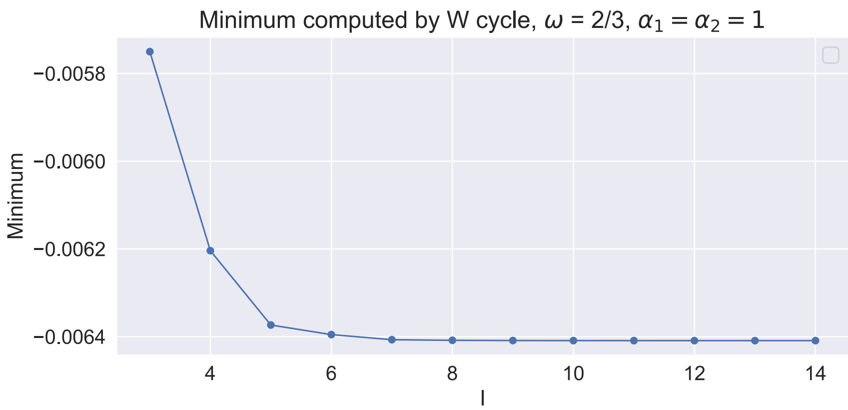

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

Note that in the last three cases the same value is found for the minimum.

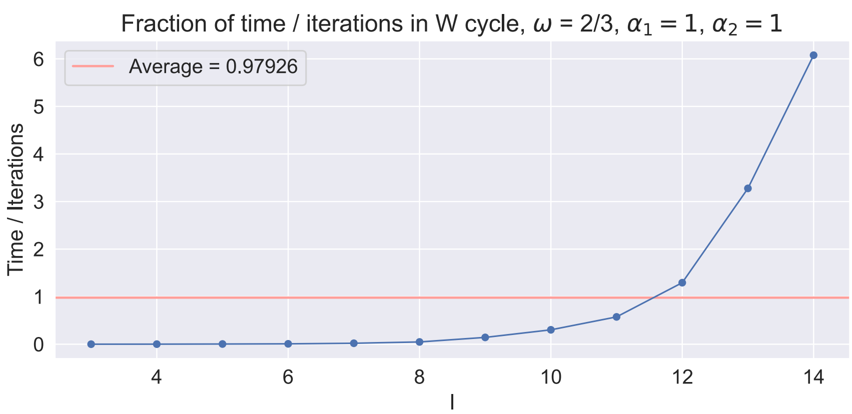

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

We note in conclusion that this approach takes much less time than the previous case, so in these terms it is to be preferred. Comparing the times with the $\omega = 1/3$ case shows that essentially the two methods are similar. The only difference is the number of iterations and how they decrease as l changes.

Case: $\alpha_1 = \alpha_2 = 2$, $\omega = 2/3$

h	l	Iterations	Time	$\min_{x\in\overline{\Omega}}$
0.125	3	5	0.00401s	-0.00575021
0.0625	4	5	0.01s	-0.00620393
0.03125	5	4	0.02394s	-0.00637338
0.015625	6	4	0.05879s	-0.00639522
0.0078125	7	3	0.09974s	-0.00640688
0.00390625	8	3	0.2791s	-0.00640827
0.001953125	9	3	0.61461s	-0.00640878
0.0009765625	10	2	0.86664s	-0.00640892
0.00048828125	11	2	2.06917s	-0.00640895
0.000244140625	12	1	2.27921s	-0.00640896
0.0001220703125	13	1	4.72046s	-0.00640896
6.103515625e-05	14	1	9.76103s	-0.00640896

Investigation of the W-cycle performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$

Results follow in graph form:

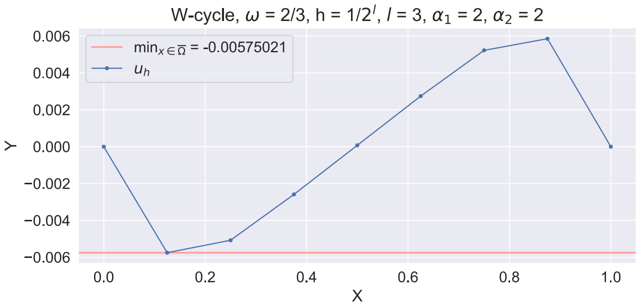

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

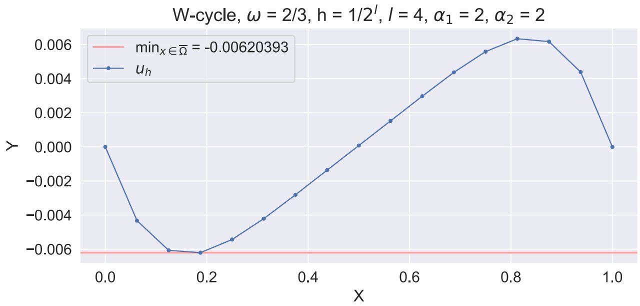

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

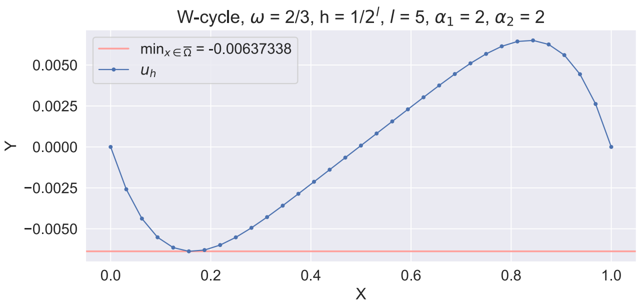

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

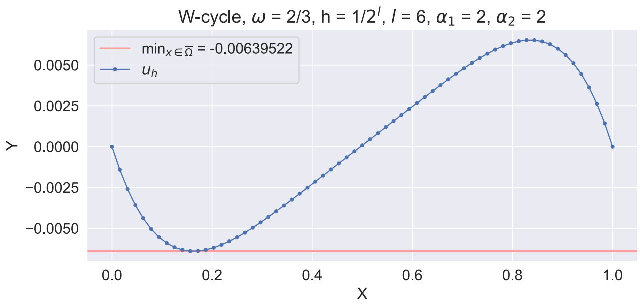

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

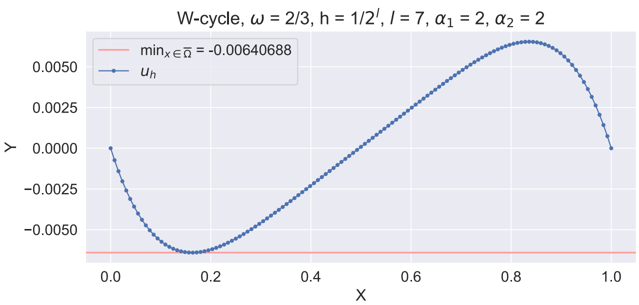

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0078125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

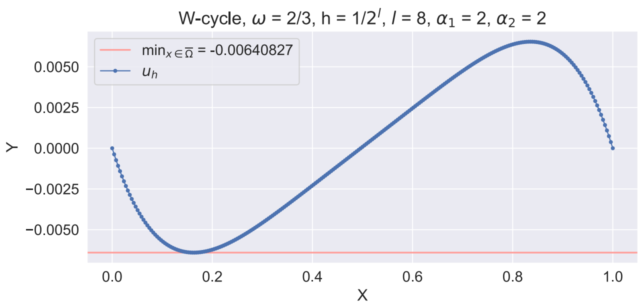

Figure: Graph of the solution approximated by the W-cycle, $h = 0.00390625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

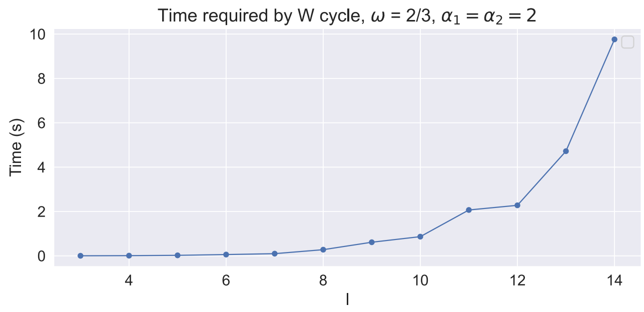

Figure: Graph of the time required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$

Figure: Graph of the iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$

Again the iterations decrease less than the $\omega = 1/3$ case.

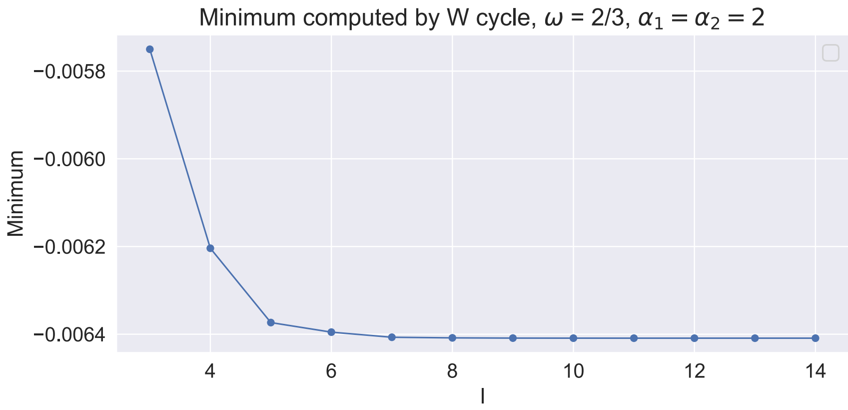

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

Note that in the last three cases the same value is found for the minimum.

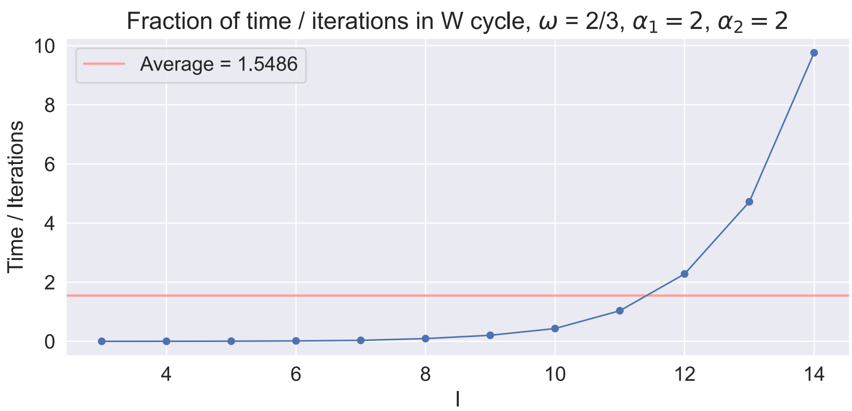

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

Comparing the two $\omega = 2/3$ cases, the $\alpha_1 = \alpha_2 = 1$ case requires more iterations but less time, resulting in a less average time/iteration fraction.

The improvement over the single step of Jacobi's method is undeniable in terms of time. The two grid correction approach is also uncompetitive.

Overall, the $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$ case seems to be the best for the problem analyzed (even if the case $\omega = 1/3$, $\alpha_1 = \alpha_2 = 1$ is very similar).

The methods seem to be faster when $\omega = 2/3$ is chosen, in agreement with what was said with the theory.

`full_mg_1d(uh, fh, omega, alpha1, alpha2, nu)`

The following section investigates a single step of the full grid scheme with $\omega = 2/3$ in the 1D problem stated in the project.

Case: $\alpha_1 = \alpha_2 = 1$, $\omega = 2/3$, $\nu = 1$

h	l	$\lvert r_h \rvert_\infty$	**Time**	$\min_{x\in\overline{\Omega}}$
0.125	3	0.0234099	0.001s	-0.00552572
0.0625	4	0.0316707	0.00199s	-0.00625795
0.03125	5	0.048057	0.00503s	-0.00639142
0.015625	6	0.0730384	0.01201s	-0.0064039
0.0078125	7	0.0903381	0.0379s	-0.00640785
0.00390625	8	0.1025166	0.12997s	-0.00640876
0.001953125	9	0.1097519	0.3772s	-0.00640884
0.0009765625	10	0.1136499	0.45984s	-0.00640894
0.00048828125	11	0.1156382	1.08329s	-0.00640895
0.000244140625	12	0.1166422	2.25359s	-0.00640896
0.0001220703125	13	0.1171467	4.75104s	-0.00640896
6.103515625e-05	14	0.1175173	10.12442s	-0.00640896

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 1$

Results follow in graph form:

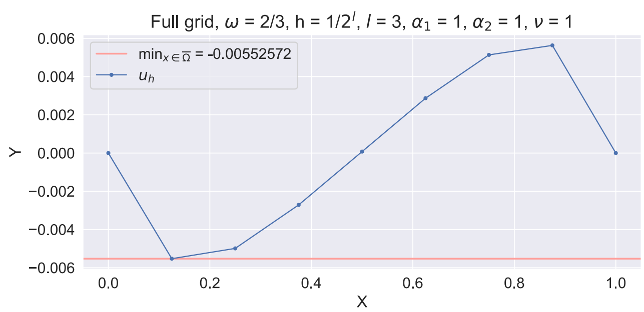

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

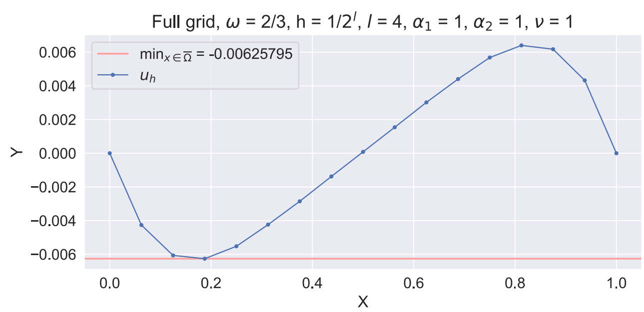

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

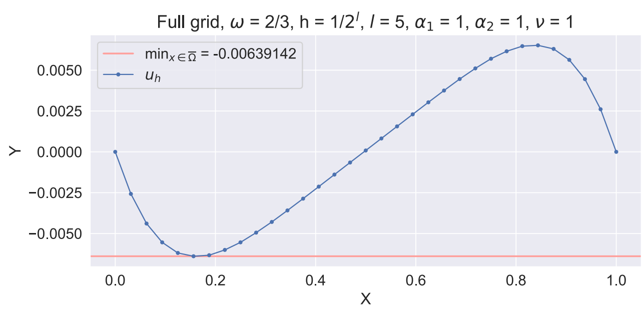

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

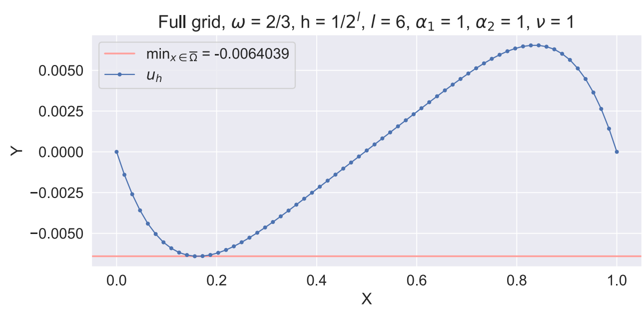

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

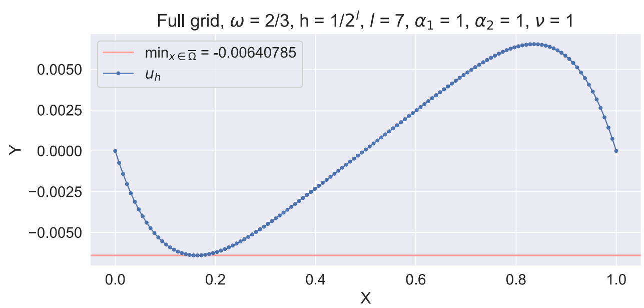

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0078125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

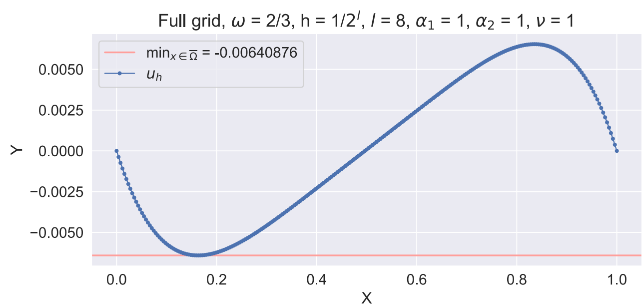

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.00390625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

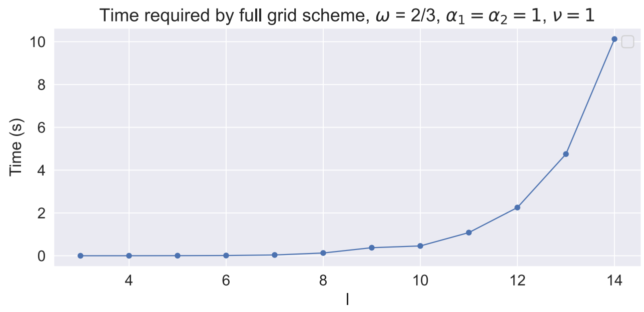

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 1$

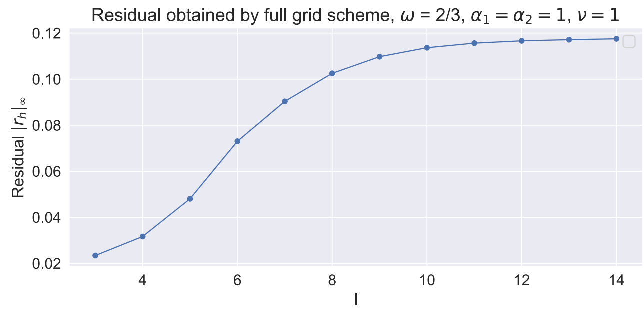

Figure: Graph of the residuals obtained by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 1$

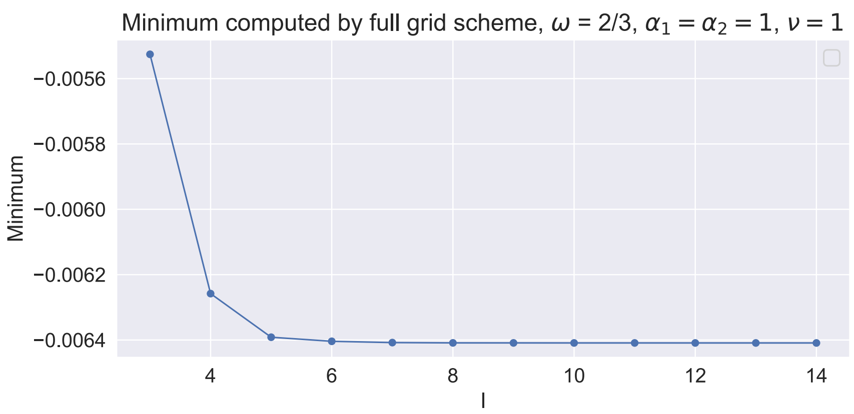

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

The times obtained are not very different (slightly higher) from those of the W-cycle $\alpha_1 = \alpha_2 = 1$ and $\omega = 2/3$, and in fact the full grid algorithm with $\nu = 1$ uses only one call to the algorithm with W-cycle.

It can be seen that the residual increases as the value of h decreases.

Case: $\alpha_1 = \alpha_2 = 1,\ \omega = 2/3,\ \nu = 2$

h	l	$\lvert r_h \rvert_\infty$	**Time**	$\min_{x\in\overline{\Omega}}$
0.125	3	0.0025255	0.00354s	-0.00572901
0.0625	4	0.0035025	0.00399s	-0.00621237
0.03125	5	0.0056125	0.01098s	-0.00637563
0.015625	6	0.0083459	0.02753s	-0.00639617
0.0078125	7	0.0101688	0.06782s	-0.00640699
0.00390625	8	0.0112081	0.27225s	-0.00640832
0.001953125	9	0.0117615	0.55147s	-0.00640878
0.0009765625	10	0.0120469	1.13405s	-0.00640893
0.00048828125	11	0.0122157	2.19992s	-0.00640895
0.000244140625	12	0.0123609	4.39275s	-0.00640896
0.0001220703125	13	0.012434	9.45451s	-0.00640896
6.103515625e-05	14	0.0124706	19.86433s	-0.00640896

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 2$

Results follow in graph form:

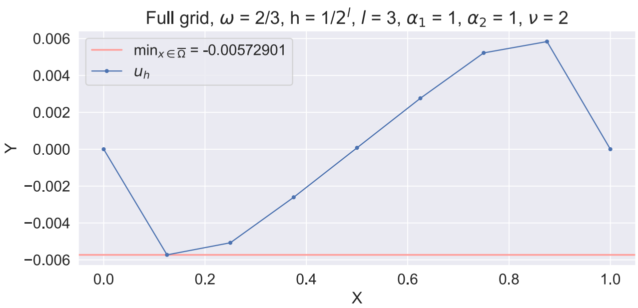

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

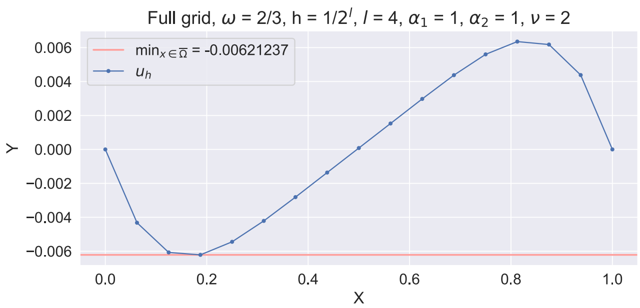

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

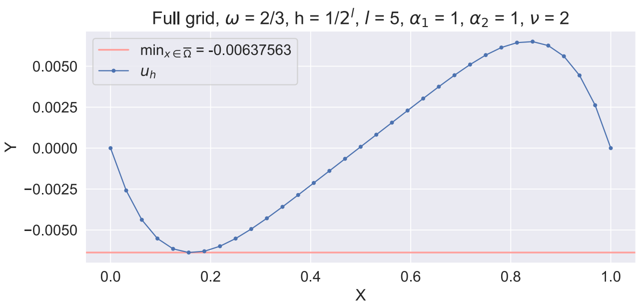

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

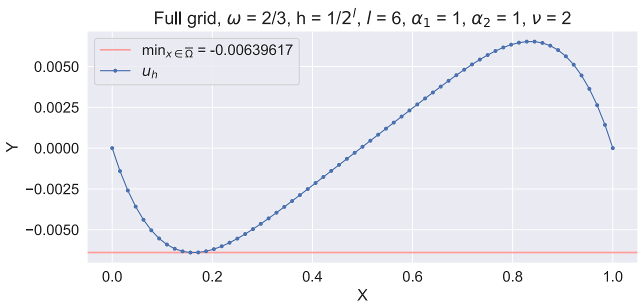

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

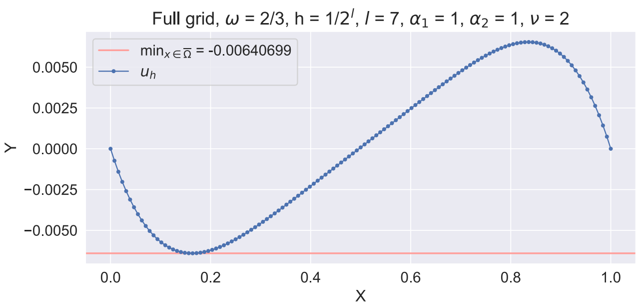

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0078125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

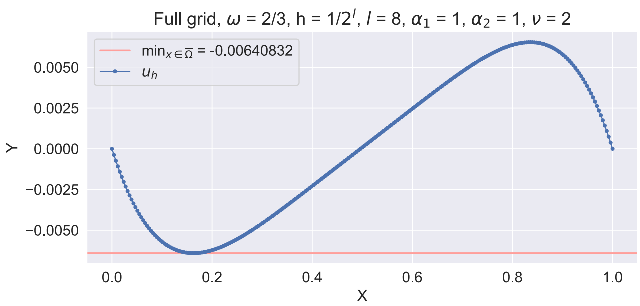

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.00390625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

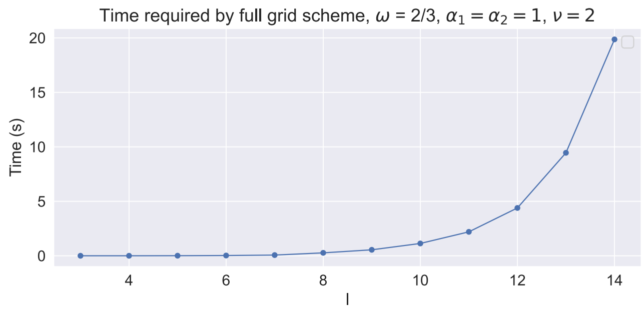

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 2$

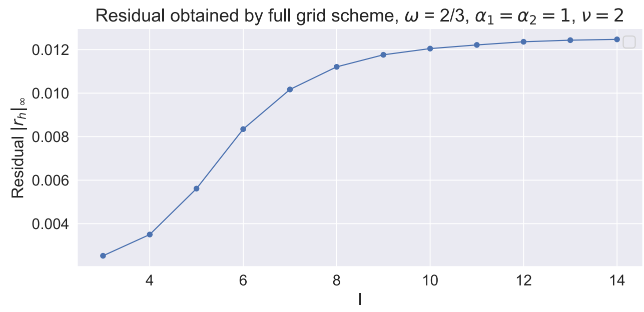

Figure: Graph of the residuals obtained by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 2$

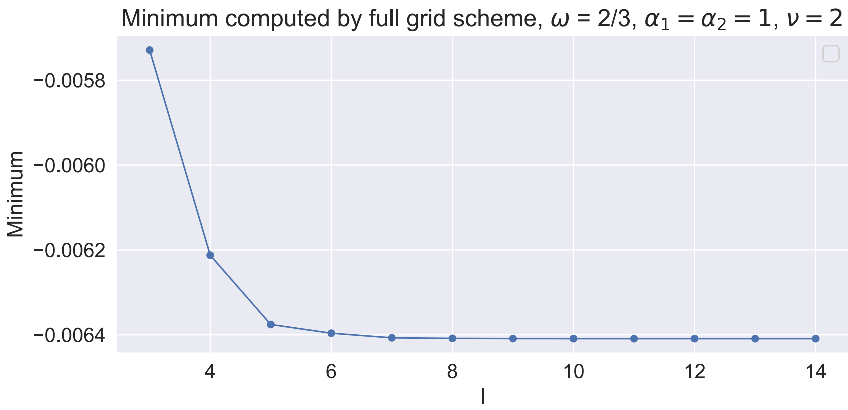

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

The residuals for each l value are substantially reduced tenfold compared with the previous case, at the price of twice as much time.

Overall, the times and residuals have the same behavior, as can be seen from the graphs.

The $\nu = 2$ case is the case where two W-cycle steps are taken, which therefore unsurprisingly doubles the times. Although we have at hand only two cases of ν values, we can conjecture that each call of W-cycle decreases the residual by ten faces.

Case: $\alpha_1 = \alpha_2 = 2$, $\omega = 2/3$, $\nu = 1$

h	l	$\lvert r_h \rvert_\infty$	**Time**	**min**$_{x\in\overline{\Omega}}$
0.125	3	0.0053232	0.001	-0.00566805
0.0625	4	0.0085343	0.00299s	-0.00616639
0.03125	5	0.010598	0.00795s	-0.00636264
0.015625	6	0.0117645	0.02192s	-0.00639342
0.0078125	7	0.0123825	0.05785s	-0.00640636
0.00390625	8	0.0127002	0.20826s	-0.00640817
0.001953125	9	0.0128612	0.33922s	-0.00640874
0.0009765625	10	0.0129422	0.95415s	-0.00640892
0.00048828125	11	0.0129829	1.72752s	-0.00640895
0.000244140625	12	0.0130033	3.48519s	-0.00640896
0.0001220703125	13	0.0130215	7.53795s	-0.00640896
6.103515625e-05	14	0.0130349	15.91308s	-0.00640896

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 1$

Results follow in graph form:

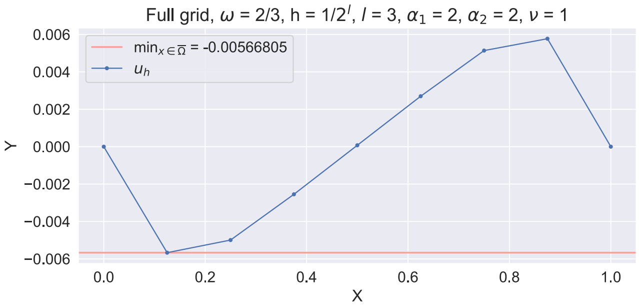

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

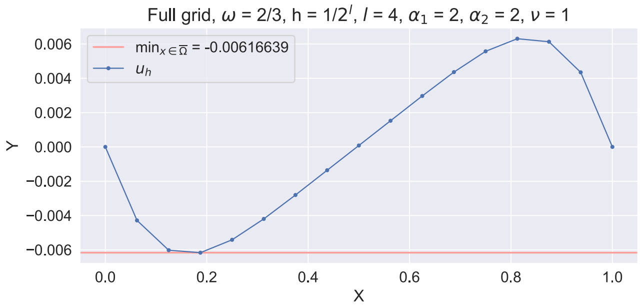

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

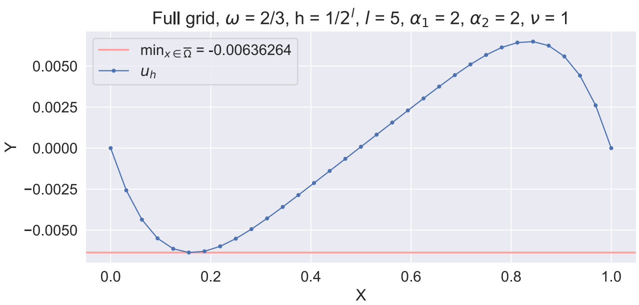

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

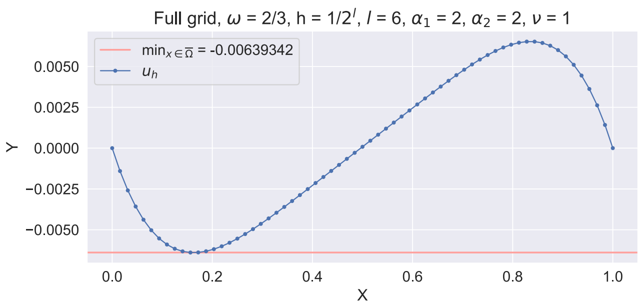

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

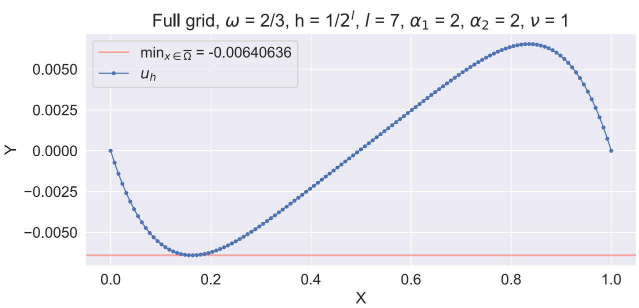

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0078125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

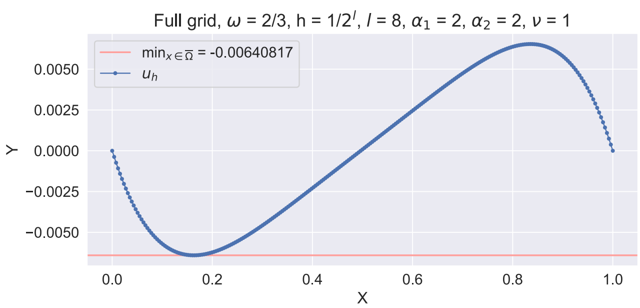

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.00390625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

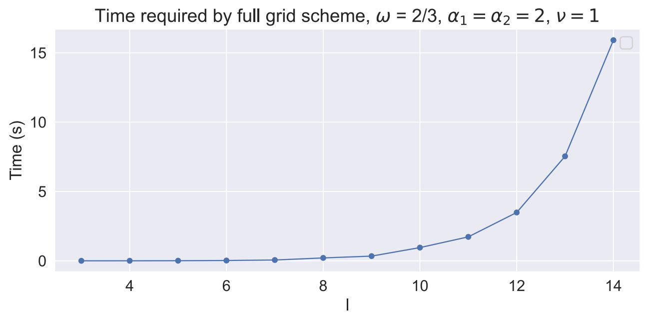

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 1$

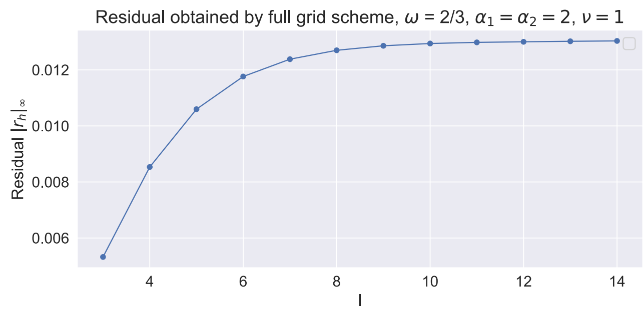

Figure: Graph of the residuals obtained by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 1$

The times increase like the previous case, while the residuals seem to increase like the square root, so differently.

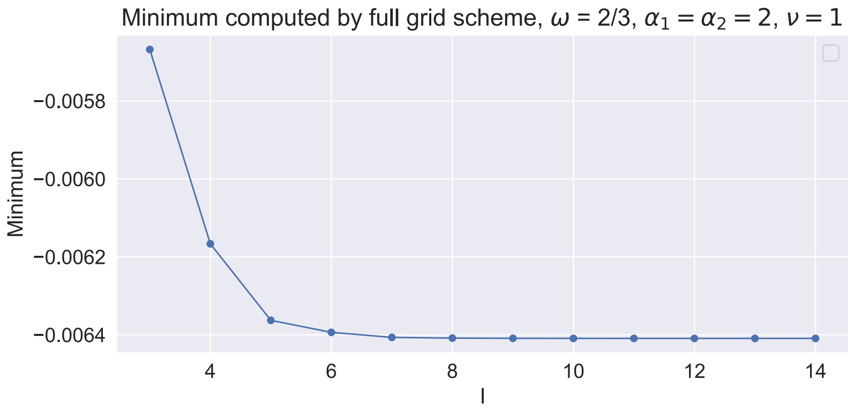

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

This case, however, despite having $\nu = 1$ the times are greater than the W-cycle case with the same alpha and omega.

Compared with the case before, we have better times but slightly worse residuals.

Note that the times are worse than the case $\alpha_1 = \alpha_2 = 1$ but the residuals are much better, just under ten times less, which justifies the times.

Case: $\alpha_1 = \alpha_2 = 2,\ \omega = 2/3,\ \nu = 2$

h	l	$\lvert r_h \rvert_\infty$	**Time**	$\min_{x\in\overline{\Omega}}$
0.125	3	0.0001997	0.00197s	-0.00574619
0.0625	4	0.0003377	0.00603s	-0.00620213
0.03125	5	0.0004132	0.01596s	-0.00637317
0.015625	6	0.000443	0.05086s	-0.0063952
0.0078125	7	0.0004545	0.16131s	-0.00640688
0.00390625	8	0.0004593	0.38649s	-0.00640827
0.001953125	9	0.0004624	0.68617s	-0.00640878
0.0009765625	10	0.0004653	1.54591s	-0.00640892
0.00048828125	11	0.0004666	3.26731s	-0.00640895
0.000244140625	12	0.0004673	6.81137s	-0.00640896
0.0001220703125	13	0.0004677	14.78558s	-0.00640896
6.103515625e-05	14	0.0004678	32.27255s	-0.00640896

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 2$

Results follow in graph form:

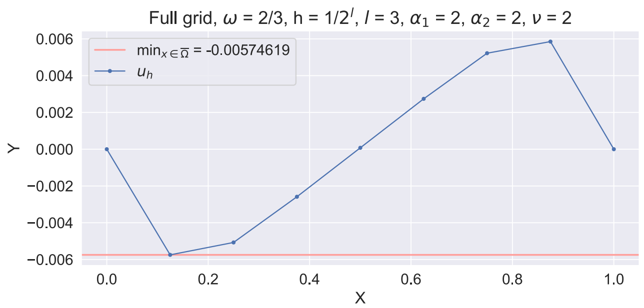

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

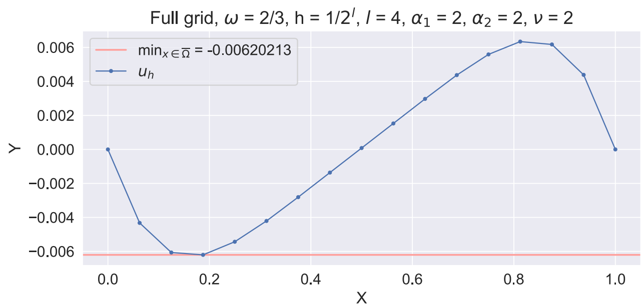

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

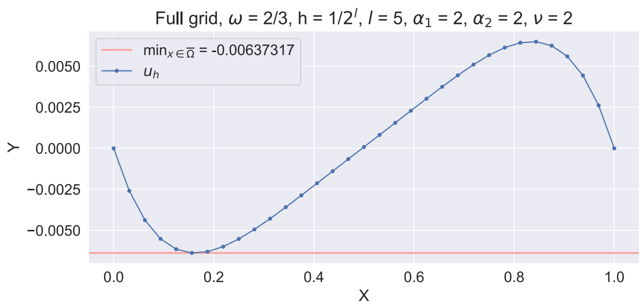

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

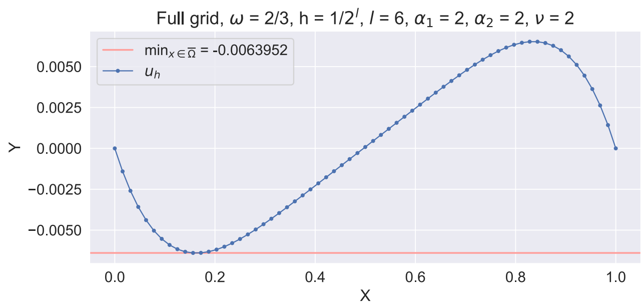

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

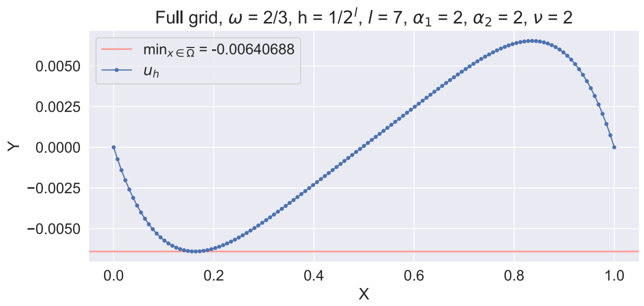

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0078125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

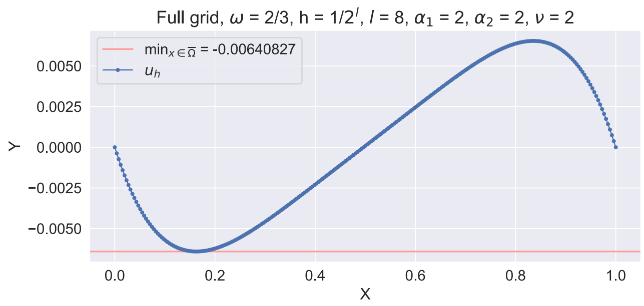

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.00390625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

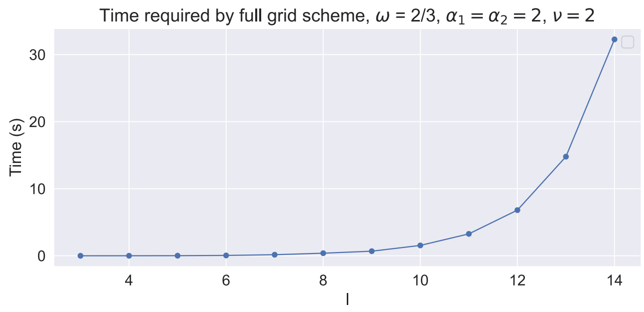

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 2$

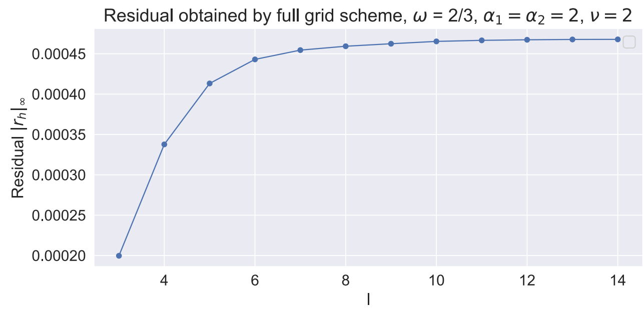

Figure: Graph of the residuals obtained by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 2$

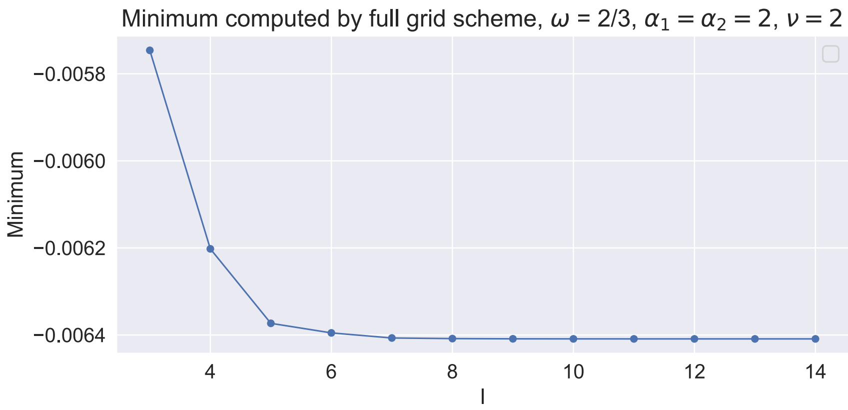

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

Again, the choice $\nu = 2$ doubles the time but greatly improves the residuals, more than tenfold than last case.

So the choice of parameters in the case of the full grid scheme depends on the interests in the results: the choice $\alpha_1 = \alpha_2 = 2$ seems to lead to lower residuals, while the choice $\alpha_1 = \alpha_2 = 1$ seems to result in shorter times.

Conclusion - 1D

Choosing the "best" method is impossible, as there are too many parameters to judge the methods analyzed so far.

Speaking, for example, of ease of implementation, Jacobi's method provides an easily approachable solution. The cons of this choice are the rather high times, even for large values of h.

The two grid correction scheme, easy to implement since it is in fact an easy modification of the Jacobi method, is a notable improvement. Both in terms of iterations, but especially in terms of times, the improvement is remarkable: up to five times faster.

The W-cycle scheme, which is more difficult to implement, gives extraordinarily lower times to the previous methods. With this method it is possible to use much lower values of h without having to wait too long and at the same time obtaining very good approximations. However, the choice of parameters for this scheme also depends on the problem, so testing is necessary.

Finally, the full-grid scheme is similar to the W-cycle in terms of implementation and provides an improvement to the latter in terms of accuracy. Although, in fact, the times are higher, very low residuals are obtained. Again, on the parameters of this scheme it is necessary to do some testing because they also depend on the problem. You can choose parameters that reward either times or residuals.

Analyzing all the methods, for small values of h (and with sometimes very high timescales) several of them arrive at a minimum value of -0.00640896 (even the full grid method with small residuals), which then seems a good choice for an approximation of the minimum value.

It makes no sense to report more than 8 significant figures since the precision required of the methods (some of them at least) was precisely `tol`=10^{-8}.

Multigrid 2D

`jacobi_step_2d(uh, fh, omega)`

The following section investigates the weighted Jacobi method to achieve a tolerance of 10^{-8} in the 2D problem stated in the project.

Case: $\omega = 1/3$

h	l	**Iterations**	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.25	2	103	0.00601s	-0.01128155
0.125	3	368	0.11469s	-0.01147954
0.0625	4	1277	1.67573s	-0.01155188
0.03125	5	4303	21.73375s	-0.01172806
0.015625	6	13975	348.28571s	-0.0117526

Investigation of the one-step weighted Jacobi method performance with $\omega = 1/3$

The results follow in graph form:

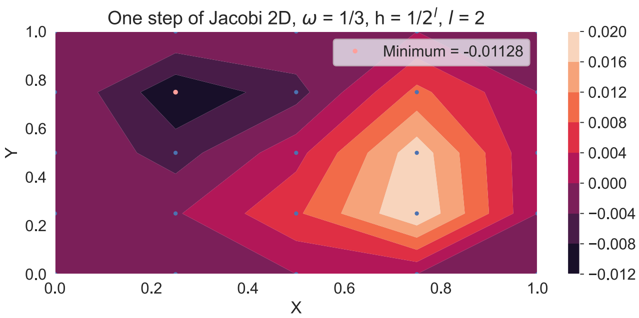

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.25$, $\omega = 1/3$

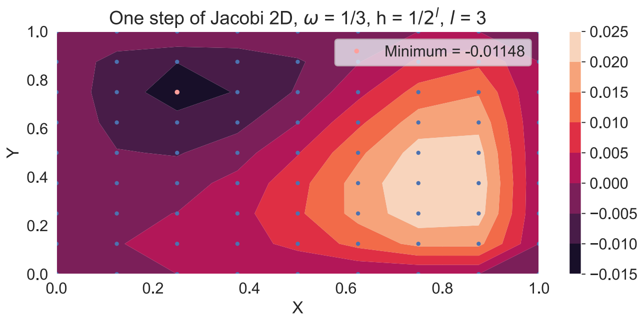

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.125$, $\omega = 1/3$

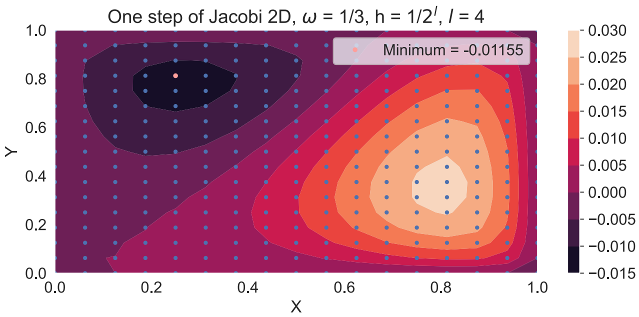

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.0625$, $\omega = 1/3$

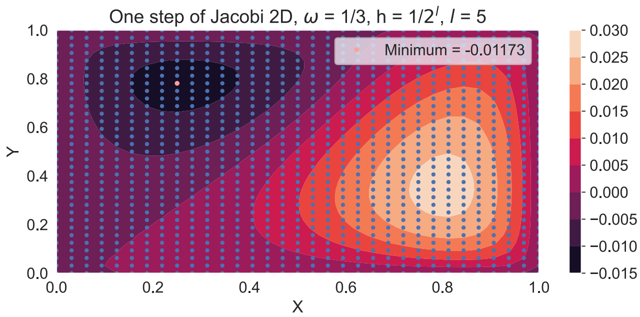

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.03125$, $\omega = 1/3$

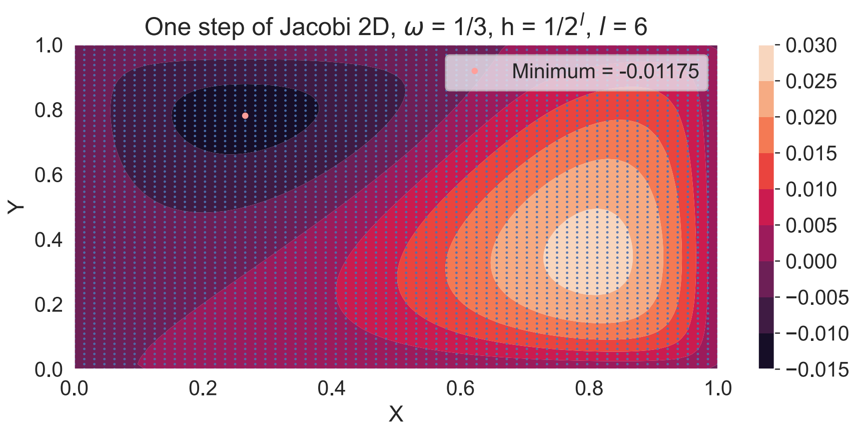

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.015625$, $\omega = 1/3$

Plotting now the iterations, time and minimum values gives the following.

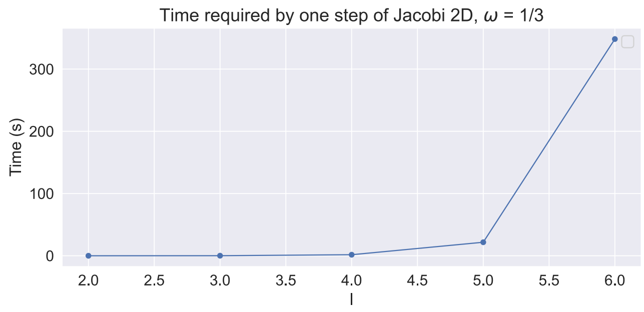

Figure: Graph of the time required by the weighted Jacobi method, $\omega = 1/3$

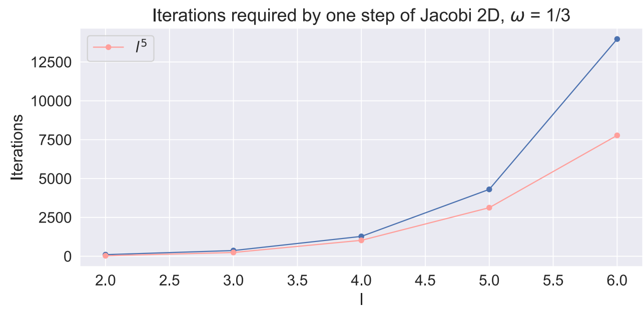

Figure: Graph of the iterations required by the weighted Jacobi method, $\omega = 1/3$

The graphs of time and iterations are similar to the 1D case, although, clearly the times are much greater. As expected, the number of iterations is also higher than the 1D case, but the growth as l changes is very similar and is around l^5.

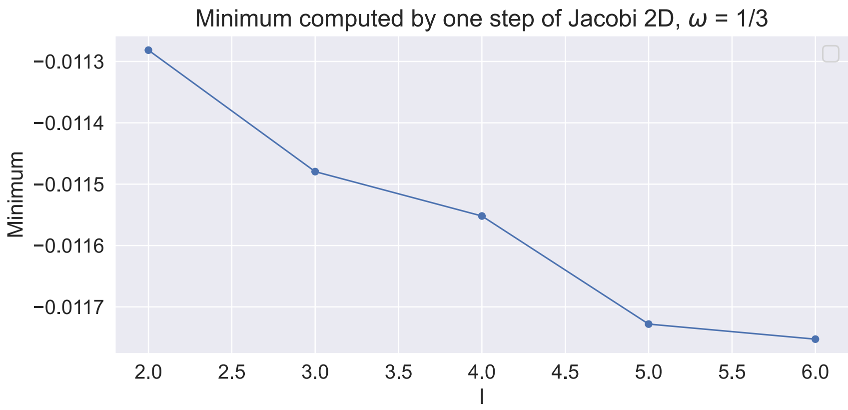

Figure: Graph of the trend of the minimum computed by the weighted Jacobi method, $\omega = 1/3$

The 2D case presents a great difference from the one-dimensional case particularly on the trend of the minimum value, which for each value of l changes significantly.

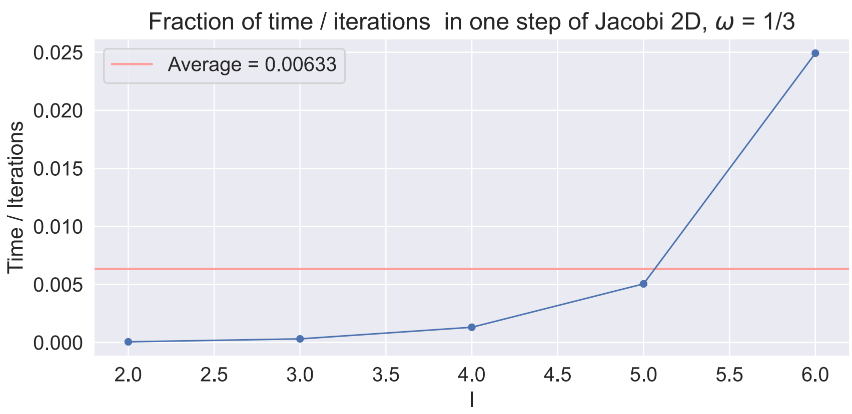

Figure: Graph of the fraction time / iterations required by the weighted Jacobi method, $\omega = 1/3$

The fraction of time/iterations is much higher than the 1D case, it is more than ten times higher. This is not very surprising to us, but it anticipates that the 2D case is much more time-consuming and iteration-consuming. We therefore expect similar situations for the other methods as well.

Case: $\omega = 2/3$

h	l	**Iterations**	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.25	2	52	0.00399s	-0.01128153
0.125	3	195	0.07035s	-0.01147944
0.0625	4	688	1.01005s	-0.01155159
0.03125	5	2353	14.06907s	-0.01172674
0.015625	6	7798	198.25517s	-0.01174709

Investigation of the one-step weighted Jacobi method performance with $\omega = 2/3$

The results follow in graph form:

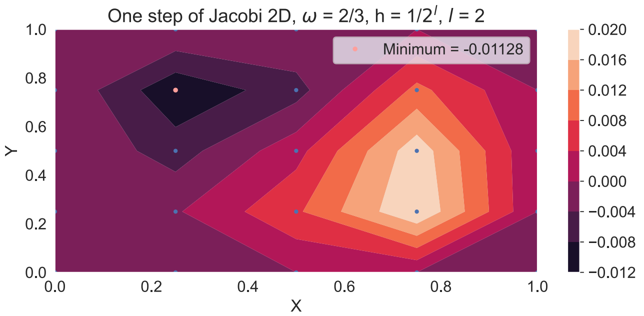

Figure : Graph of the solution approximated by the weighted Jacobi method, $h = 0.25$, $\omega = 2/3$

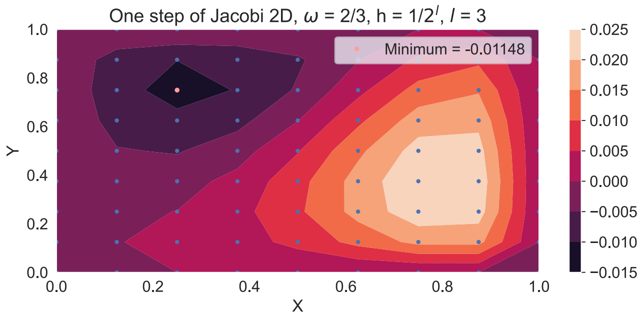

Figure : Graph of the solution approximated by the weighted Jacobi method, $h = 0.125$, $\omega = 2/3$

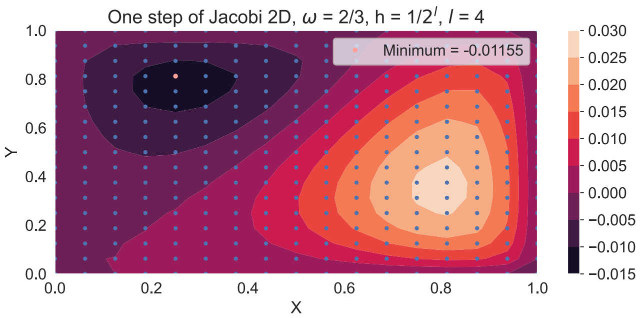

Figure : Graph of the solution approximated by the weighted Jacobi method, $h = 0.0625$, $\omega = 2/3$

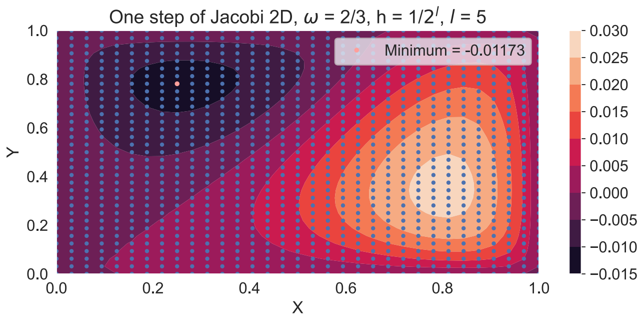

Figure : Graph of the solution approximated by the weighted Jacobi method, $h = 0.03125$, $\omega = 2/3$

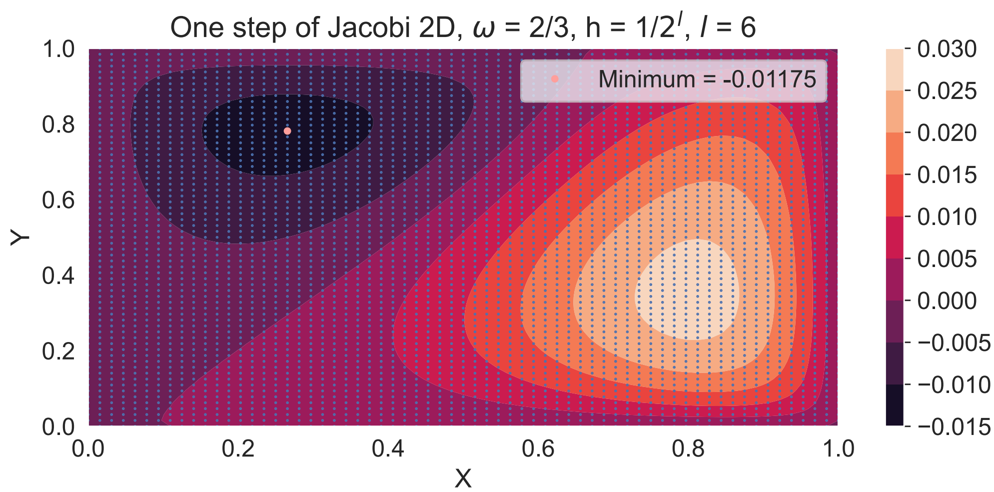

Figure: Graph of the solution approximated by the weighted Jacobi method, $h = 0.015625$, $\omega = 2/3$

Plotting now the iterations, time and minimum values gives the following.

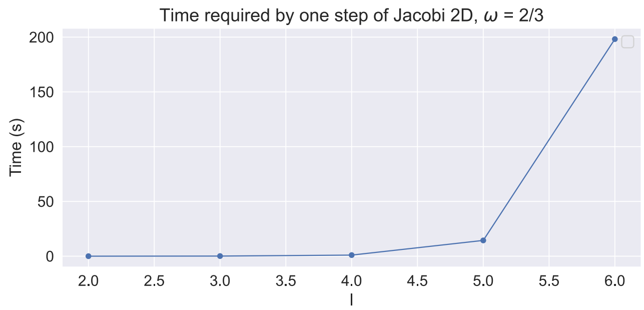

Figure: Graph of the time required by the weighted Jacobi method, $\omega = 2/3$

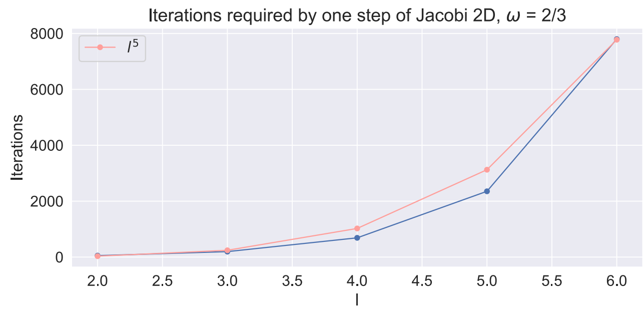

Figure: Graph of the iterations required by the weighted Jacobi method, $\omega = 2/3$

Again, the behaviors of time and iterations are comparable to those in the 1D case. There is a decrease in time and iterations compared with the previous 2D case.

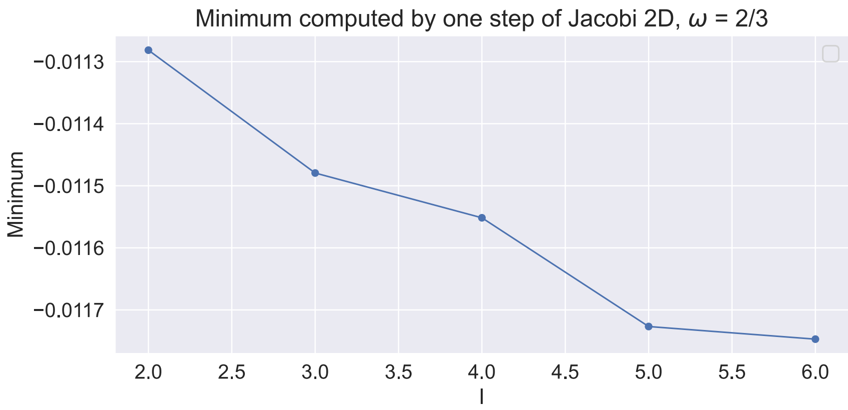

Figure: Graph of the trend of the minimum computed by the weighted Jacobi method, $\omega = 2/3$

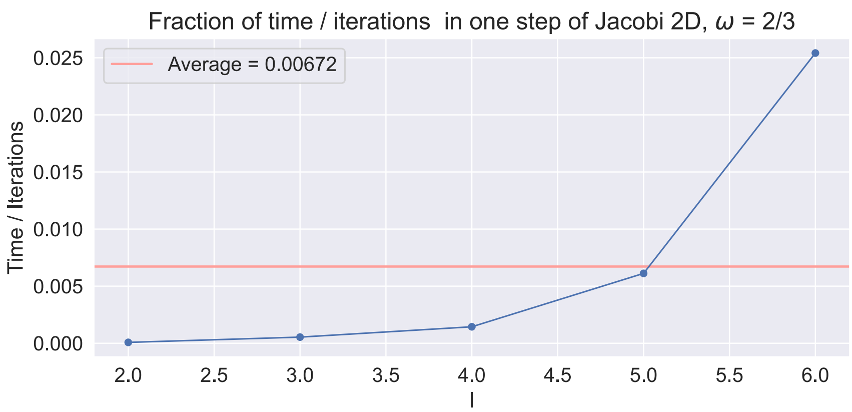

Figure: Graph of the fraction time / iterations required by the weighted Jacobi method, $\omega = 2/3$

The fraction of time/iterations is slightly increased over the previous case because of fewer iterations and less time. Overall, this method is preferred over the previous one in terms of both time and iterations. The significant difference in timing is emphasized.

Again there is a worsening in timing compared to the 1D case, but a slight improvement in the number of iterations.

Two-grid correction scheme

The following section investigates the two-grid correction scheme (with ten steps of the weighted Jacobi method on Ω_{2h}) to achieve a tolerance of 10^{-8} in the 2D problem stated in the project.

Code

Again, only for this case the code is given, since in the `.py` of the project it was not left (because it was not required). Only the $\omega = 1/3$ case is reported since the other case is identical except for the value of omega...

```
for l in range(2,7):
    n = 2**l
    h = 1 / n
    x = np.linspace(0, 1, n+1)
    y = np.linspace(0, 1, n+1)
    x, y = np.meshgrid(x, y)
    fh = np.zeros((n+1,n+1))
    fh[1:n,1:n] = f_dd(x[1:n,1:n], y[1:n,1:n])
    uh = np.zeros((n+1,n+1))
    uh[1:n,1:n] = 0

    tol = 1e-8
    kJ1h = 0
    kJ12h = 0

    tic = time()

    checkJ1 = tol + 1

    while checkJ1 > tol:
        v = uh.copy()

        # pre-smoothing
        checkJ1 = jacobi_step_2d(uh=uh,fh=fh, omega=1/3)
        kJ1h += 1

        # compute rh
        Ahuh = Ahuh_multiplication_dd(uh)
        rh = fh-Ahuh

        # restriction
        r2h = restriction_operator_dd(rh)

        # Solution with 10 steps jacobi
        jacobi_steps = 10
        e2h = np.zeros_like(r2h)
        for i in range(jacobi_steps):
            jacobi_step_2d(uh=e2h,fh=r2h, omega=1/3)
            kJ12h += 1

        # prolongation
        eh = prolongation_operator_dd(e2h)

        # correction
        uh += eh

        # post-smoothing
        jacobi_step_2d(uh=uh, fh=fh, omega=1/3)
        kJ1h+=1

        checkJ1 = la.norm((uh-v).flat, np.inf)

    elapsed = round(time() - tic, 5)
```

Analysis of the results follows.

Case: $\omega = 1/3$

h	l	**Iterations on** Ω_h	**Iterations on** Ω_{2h}	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.25	2	56	280	0.01692s	-0.0112815
0.125	3	62	310	0.06483s	-0.01147936
0.0625	4	92	460	0.47435s	-0.01155131
0.03125	5	314	1570	11.43543s	-0.01172549
0.015625	6	1086	5430	128.31745s	-0.01174185

Investigation of the two grid correction performance with $\omega = 1/3$

The results follow in graph form:

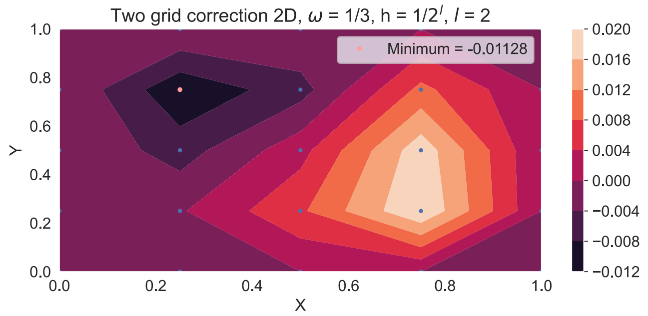

Figure: Graph of the solution approximated by the two grid correction, $h = 0.25$, $\omega = 1/3$

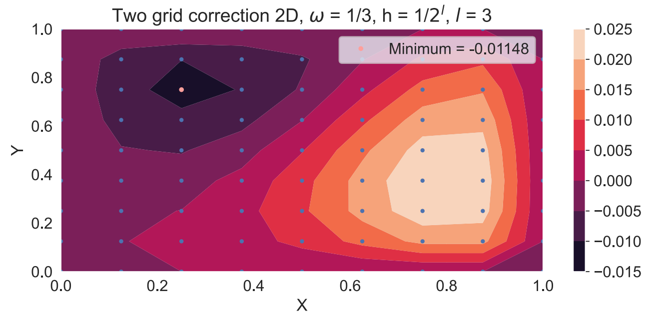

Figure: Graph of the solution approximated by the two grid correction, $h = 0.125$, $\omega = 1/3$

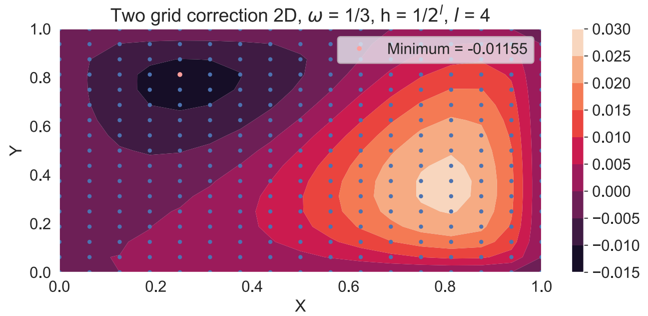

Figure: Graph of the solution approximated by the two grid correction, $h = 0.0625$, $\omega = 1/3$

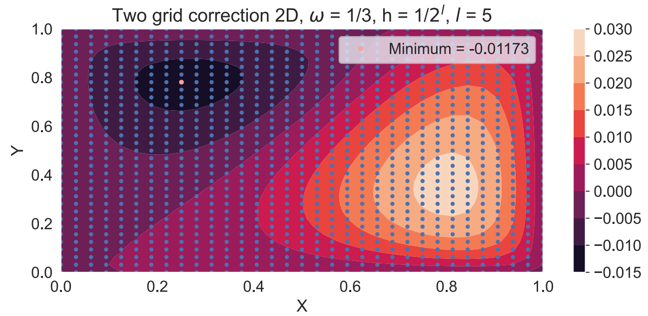

Figure: Graph of the solution approximated by the two grid correction, $h = 0.03125$, $\omega = 1/3$

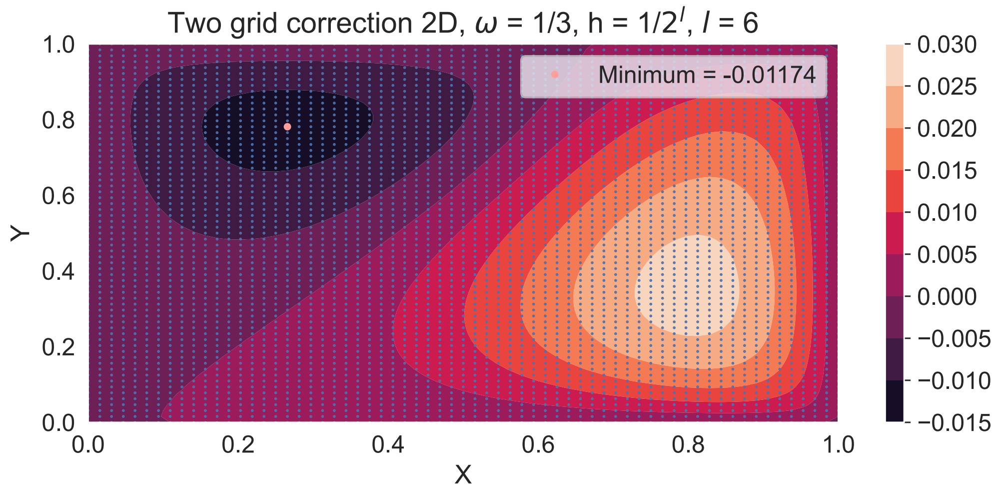

Figure: Graph of the solution approximated by the two grid correction, $h = 0.015625$, $\omega = 1/3$

Plotting now the iterations, time and minimum values gives the following.

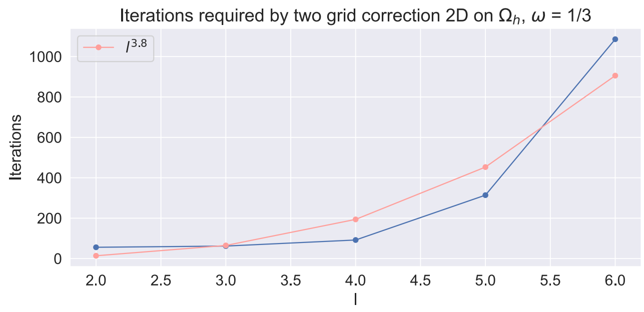

Figure: Graph of the iterations required by the two grid correction on Ω_h, $\omega = 1/3$

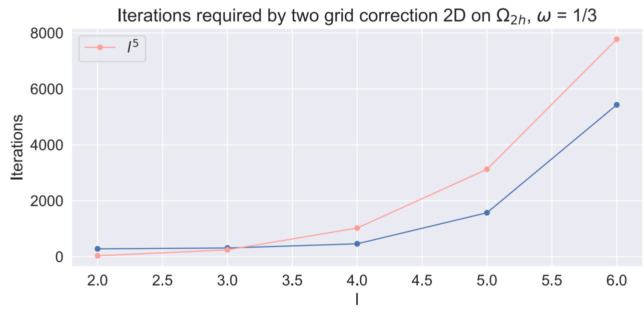

Figure: Graph of the iterations required by the two grid correction on Ω_{2h}, $\omega = 1/3$

The 2D case has a greater increase in the number of iterations than the 1D case.

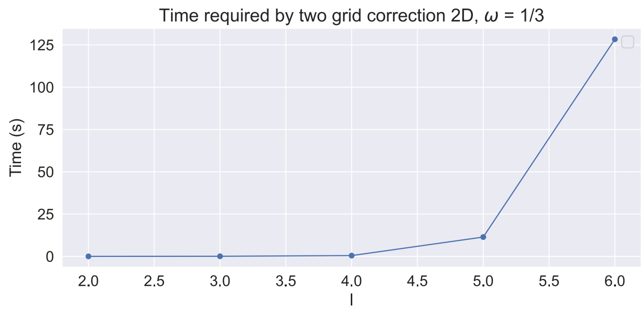

Figure: Graph of the time required by the two grid correction, $\omega = 1/3$

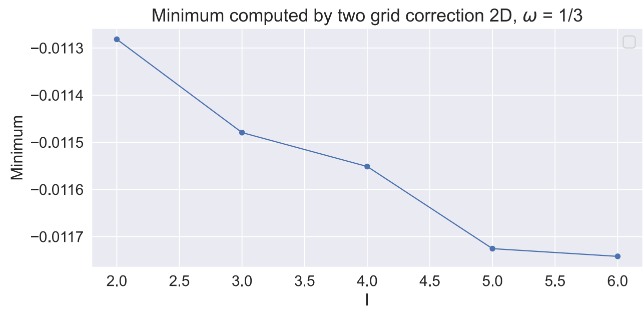

Figure: Graph of the minimum value find by the approximation find with the two grid correction, $\omega = 1/3$

The times required by this method in 2D are much greater, although the graph reports similar behavior to 1D. This is the method that is most affected in terms of time by the transition to the 2D case.

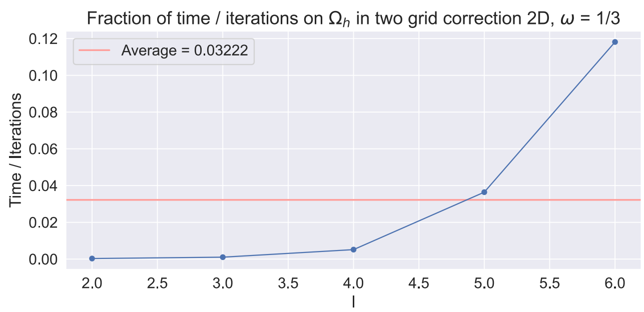

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_h, $\omega = 1/3$

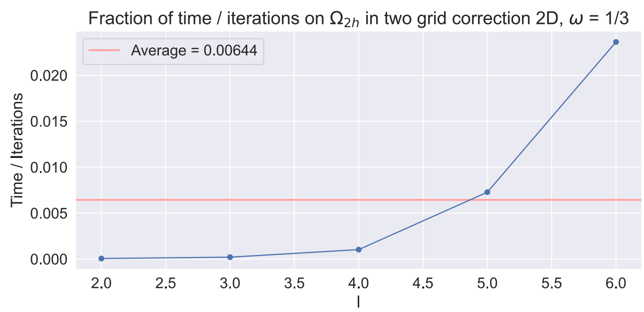

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_{2h}, $\omega = 1/3$

The ratio between the fractions in the two grids is the same as in the 1D case.

However, compared with the 1D case there is a significant worsening: the fractions have worsened by about one hundred times as much. This information does not surprise us by noting the time required and the iterations needed.

The times are still better than the Jacobi method in 2D. You have in this respect a situation similar to the 1D case, where the two grid correction is better than the Jacobi method.

Case: $\omega = 2/3$

h	l	**Iterations on** Ω_h	**Iterations on** Ω_{2h}	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.25	2	28	140	0.01995s	-0.01128151
0.125	3	32	160	0.05421s	-0.01147936
0.0625	4	50	250	0.46866s	-0.01155132
0.03125	5	166	830	3.42119s	-0.01172546
0.015625	6	582	2910	84.25164s	-0.01174172

Investigation of the two grid correction performance with $\omega = 2/3$

The results follow in graph form:

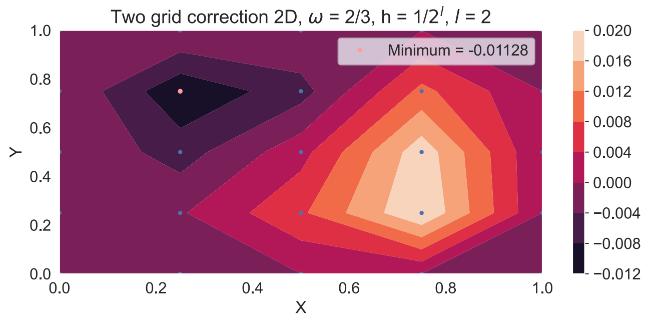

Figure: Graph of the solution approximated by the two grid correction, $h = 0.25$, $\omega = 2/3$

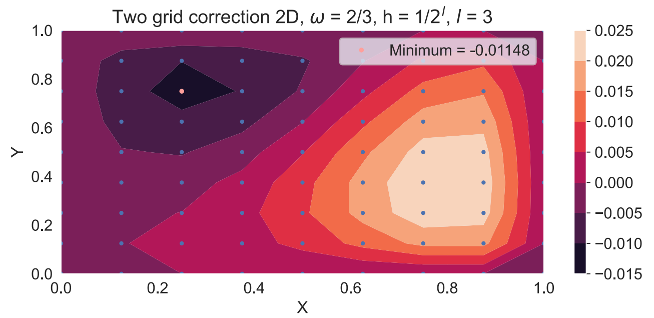

Figure: Graph of the solution approximated by the two grid correction, $h = 0.125$, $\omega = 2/3$

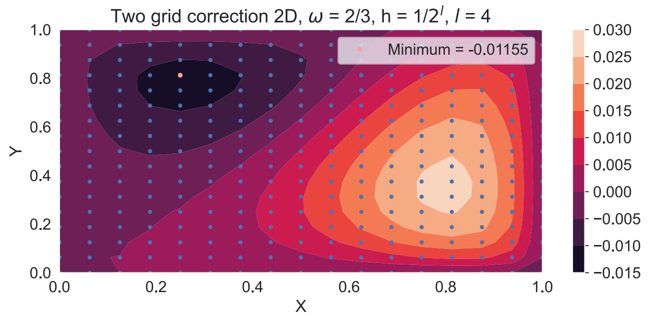

Figure: Graph of the solution approximated by the two grid correction, $h = 0.0625$, $\omega = 2/3$

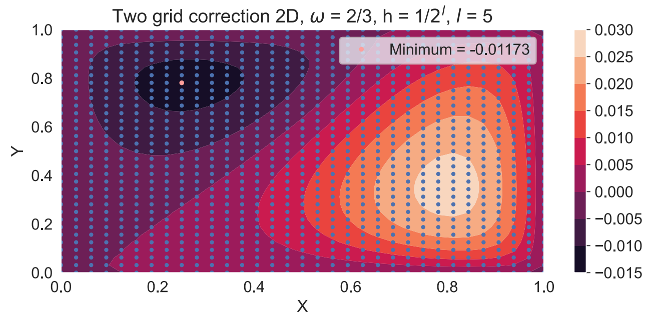

Figure: Graph of the solution approximated by the two grid correction, $h = 0.03125$, $\omega = 2/3$

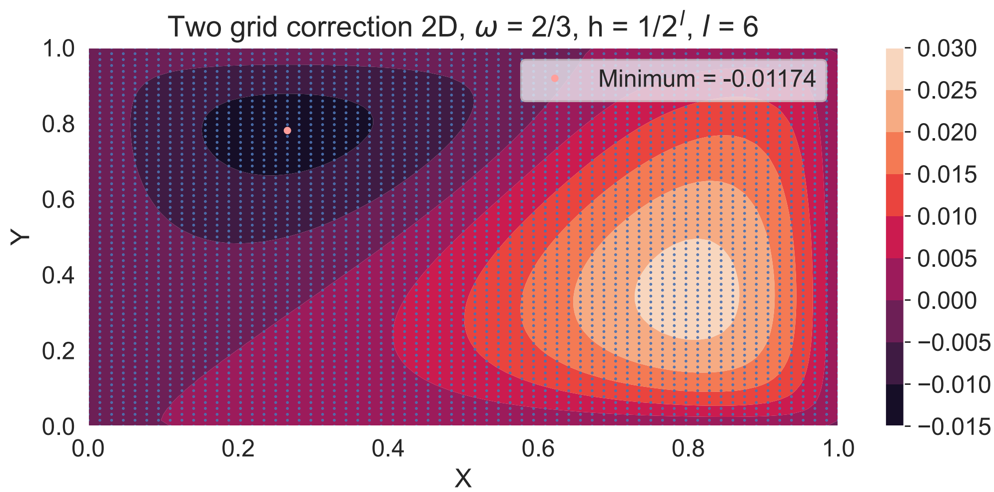

Figure: Graph of the solution approximated by the two grid correction, $h = 0.015625$, $\omega = 2/3$

Plotting now the iterations, time and minimum values gives the following.

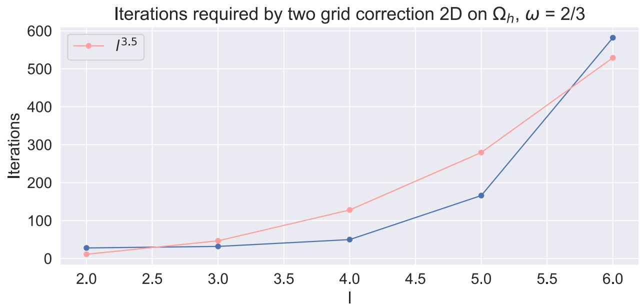

Figure: Graph of the iterations required by the two grid correction on Ω_h, $\omega = 2/3$

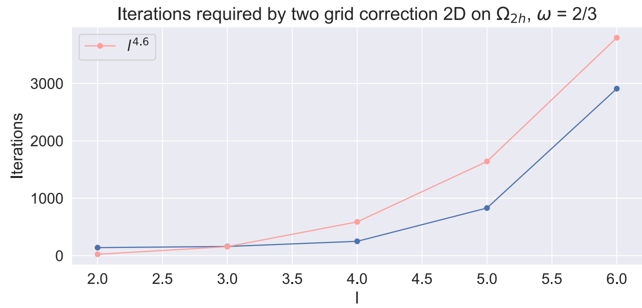

Figure: Graph of the iterations required by the two grid correction on Ω_{2h}, $\omega = 2/3$

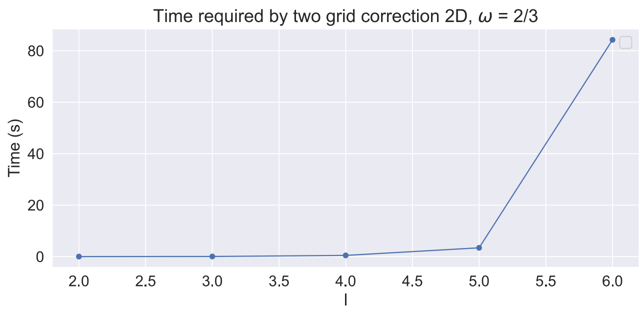

Figure: Graph of the time required by the two grid correction, $\omega = 2/3$

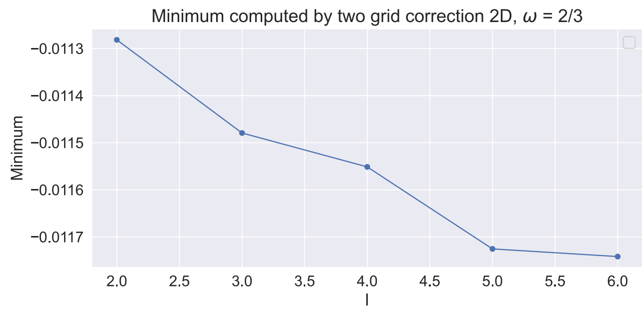

Figure: Graph of the minimum value find by the approximation find with the two grid correction, $\omega = 2/3$

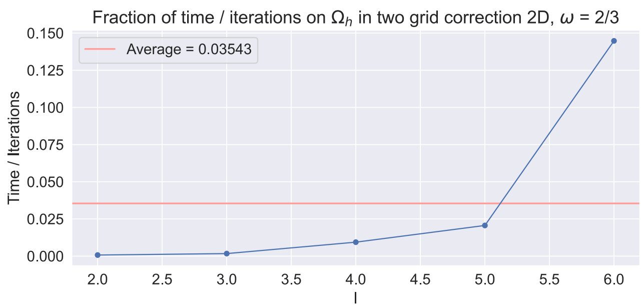

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_h, $\omega = 2/3$

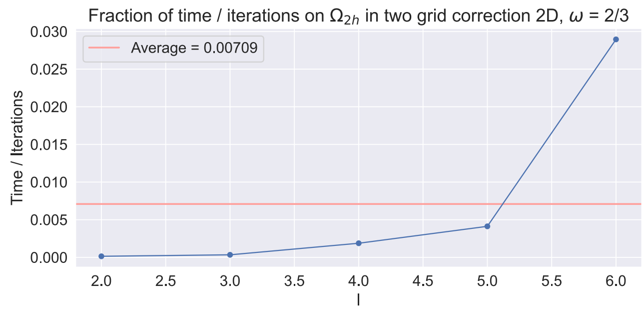

Figure: Graph of the fraction time / iterations required by the two grid correction on Ω_{2h}, $\omega = 2/3$

The same situation occurs as in the previous case of two grid correction: the ratio of the two grids is similar to the 1D case, while overall the fraction of time/iterations is greatly worsened.

Of the two omega cases, the choice $\omega = 2/3$ is preferred in terms of both time and iterations. The considerable difference, and thus the great improvement, is emphasized.

Overall, it is clear that two grid correction is preferable to the Jacobi method, but evidently the behaviors of the methods are different from 1D.

`w_cycle_step_2d(uh, fh, omega, alpha1, alpha2)`

The following section investigates the W-cycle with $\omega = 1/3$ (with ten steps of the weighted Jacobi method on Ω_{2h}) to achieve a tolerance of 10^{-8} in the 2D problem stated in the project.

Case: $\alpha_1 = \alpha_2 = 1$, $\omega = 1/3$

h	l	**Iterations**	**Time**	$\mathbf{min}_{x \in \overline{\Omega}}$
0.25	2	27	0.01197s	-0.01128149
0.125	3	30	0.06084s	-0.01147936
0.0625	4	28	0.30592s	-0.01155132
0.03125	5	25	0.9985s	-0.01172544
0.015625	6	22	5.00512s	-0.01174159
0.0078125	7	19	15.93716s	-0.01174671
0.00390625	8	15	42.6939s	-0.01174864

Investigation of the W-cycle performance with $\omega = 1/3$, $\alpha_1 = \alpha_2 = 1$

Results follow in graph form:

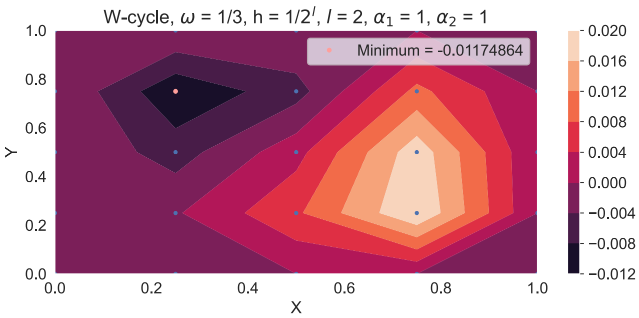

Figure: Graph of the solution approximated by the W-cycle, $h = 0.25$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

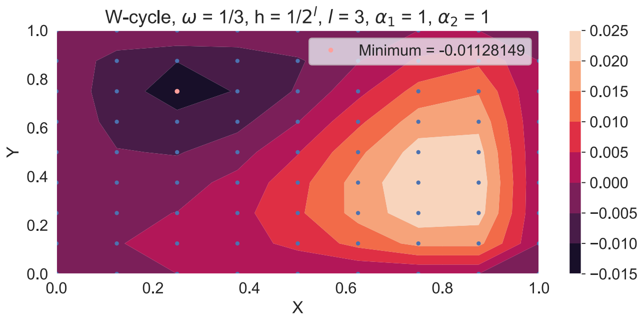

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

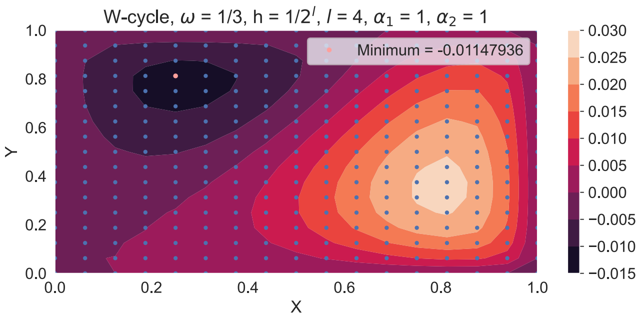

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

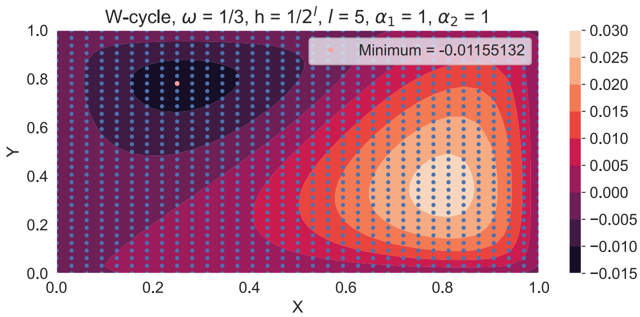

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

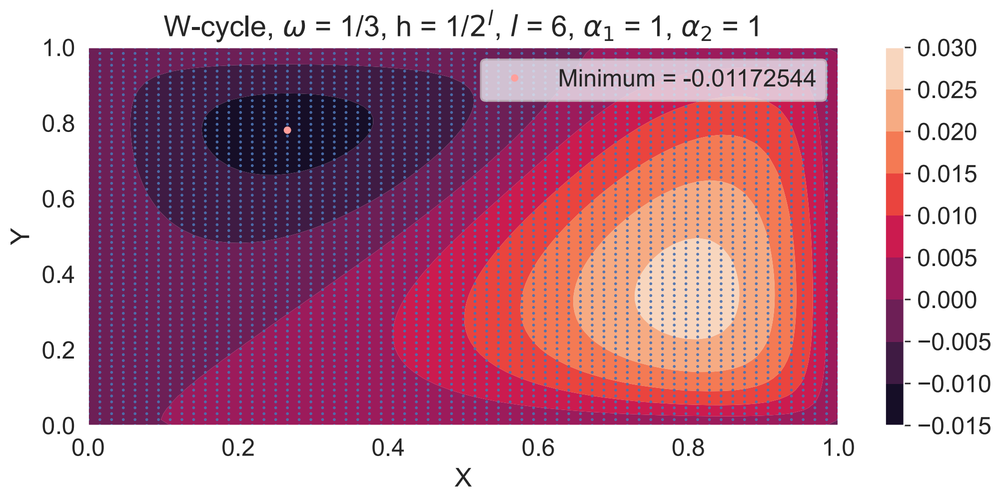

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

No plots are shown for larger l values since the difference cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

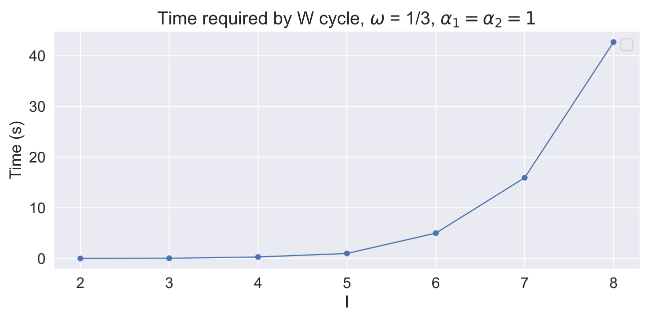

Figure: Graph of the time required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 1$

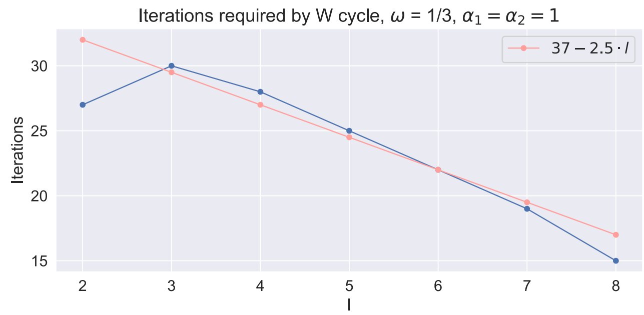

Figure: Graph of the iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 1$

The time trend is similar to the 1D case, with an increase in time. Note that the increase in time is not as noticeable as for the two grid scheme.

The trend of iterations, on the other hand, is different: it decreases with more speed than in the 1D case. Despite this, the number of iterations is about twice as high as the 1D case.

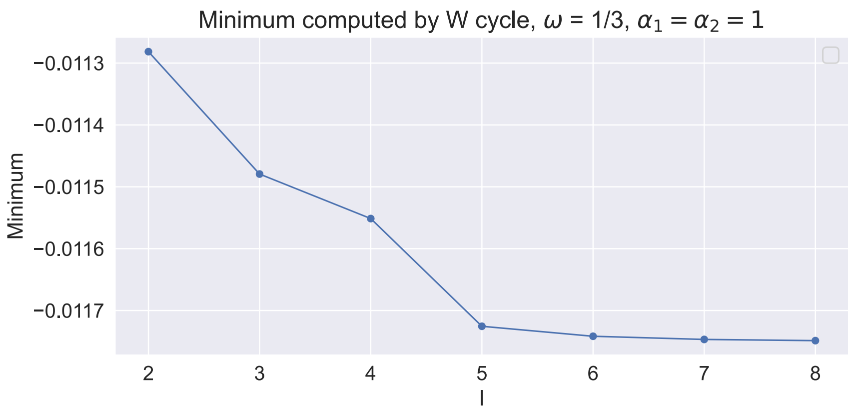

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

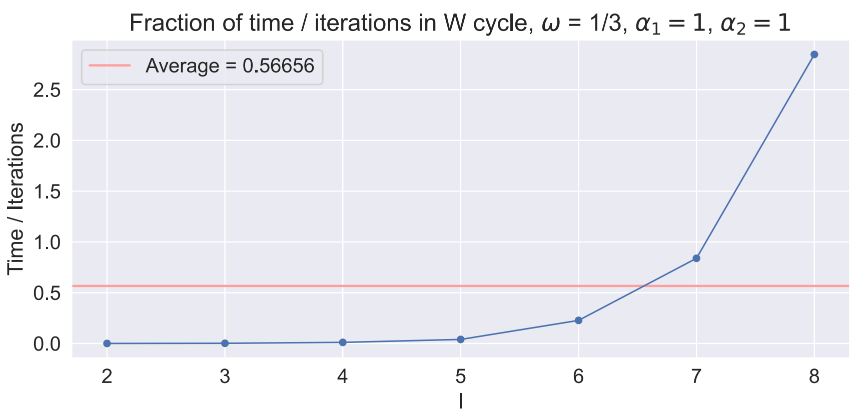

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

There is an interesting improvement in the fraction of time/iterations. Comparing these fractions between 1D and 2D methods then we can conclude that the 2D method is more efficient in this sense.

Case: $\alpha_1 = \alpha_2 = 2$, $\omega = 1/3$

h	l	**Iterations**	**Time**	$\min_{x\in\overline{\Omega}}$
0.25	2	14	0.00798s	-0.0112815
0.125	3	15	0.04687s	-0.01147936
0.0625	4	14	0.23385s	-0.01155132
0.03125	5	13	1.29416s	-0.01172544
0.015625	6	11	3.47643s	-0.01174159
0.0078125	7	10	12.8009s	-0.01174671
0.00390625	8	8	37.35807s	-0.01174864

Investigation of the W-cycle performance with $\omega = 1/3$, $\alpha_1 = \alpha_2 = 2$

Results follow in graph form:

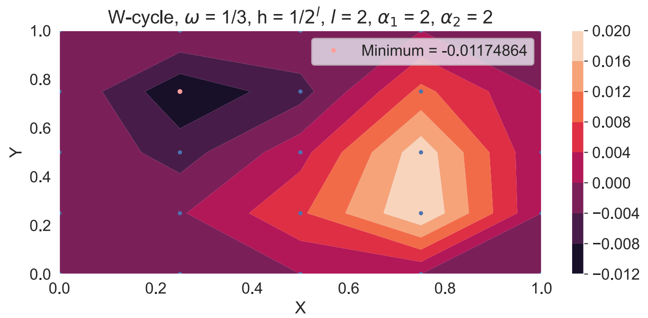

Figure: Graph of the solution approximated by the W-cycle, $h = 0.25$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

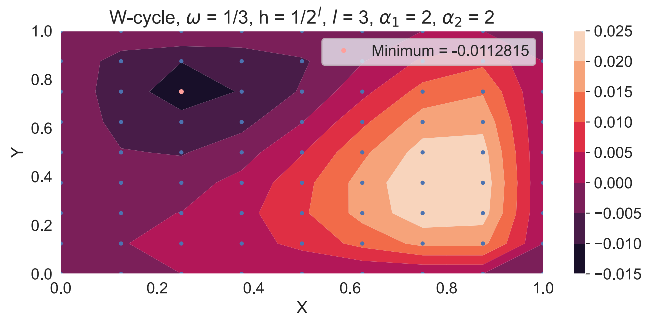

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

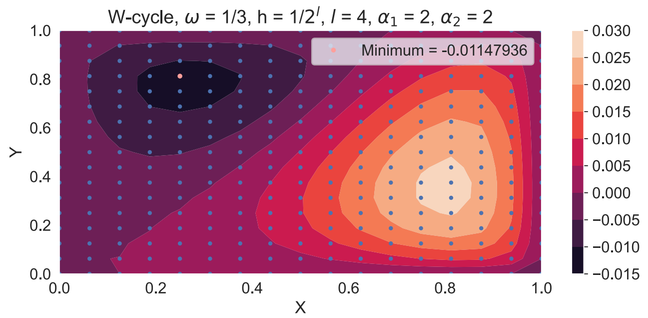

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

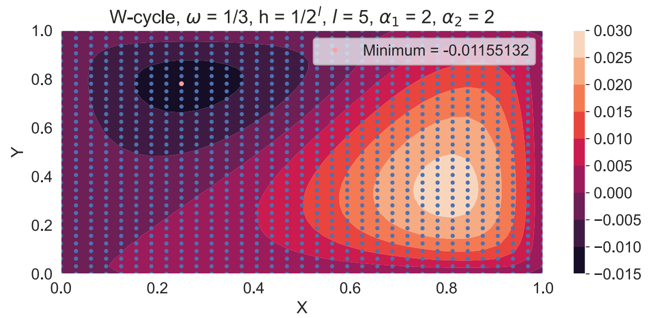

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

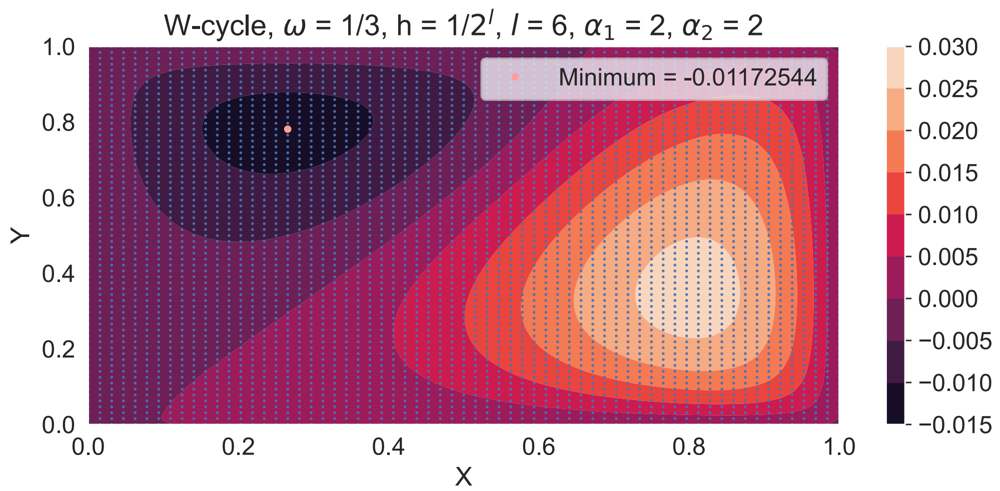

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

No plots are shown for larger l values since the difference cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

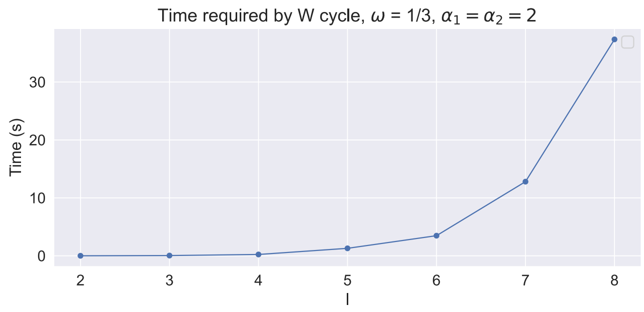

Figure: Graph of the time required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 2$

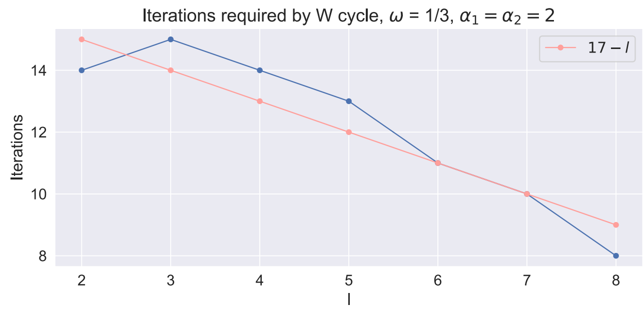

Figure: Graph of the iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = \alpha_2 = 2$

In this case the increase of times with different values of l is similar to the 1D case, and the same is true for the trend of iterations.

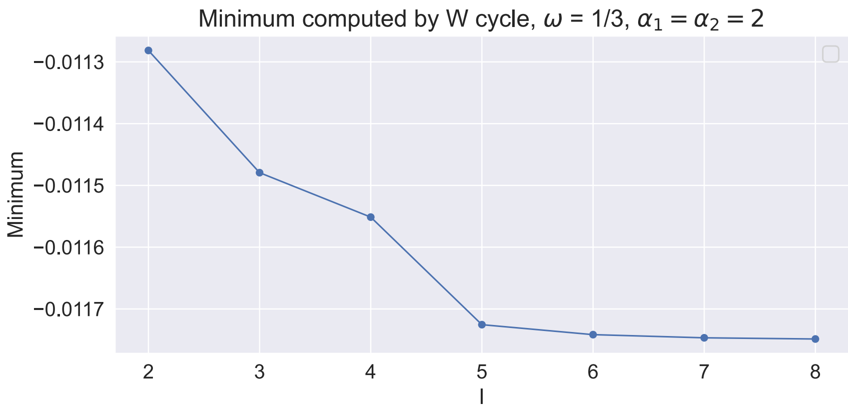

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

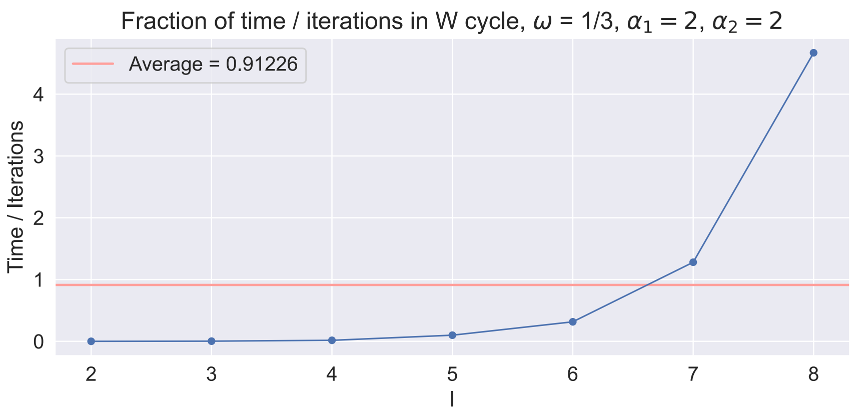

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 1/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

Compared with the previous case, there is an overall improvement in both time and iterations, although the time to iteration ratio has worsened.

Case: $\alpha_1 = \alpha_2 = 1$, $\omega = 2/3$

h	l	**Iterations**	**Time**	$\min_{x\in\overline{\Omega}}$
0.25	2	13	0.00499s	-0.01128151
0.125	3	14	0.02891s	-0.01147936
0.0625	4	14	0.15127s	-0.01155131
0.03125	5	13	0.64355s	-0.01172544
0.015625	6	11	2.38974s	-0.01174159
0.0078125	7	10	9.56507s	-0.01174671
0.00390625	8	8	25.48797s	-0.01174864

Investigation of the W cycle performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$

Results follow in graph form:

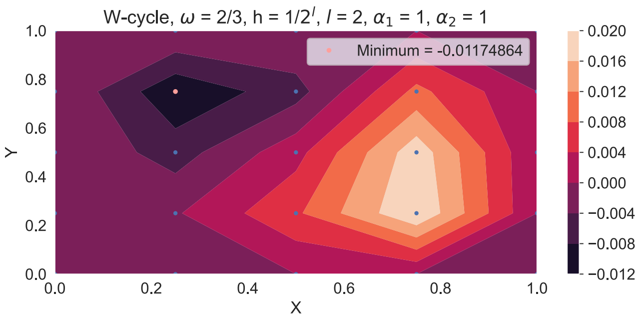

Figure: Graph of the solution approximated by the W-cycle, $h = 0.25$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

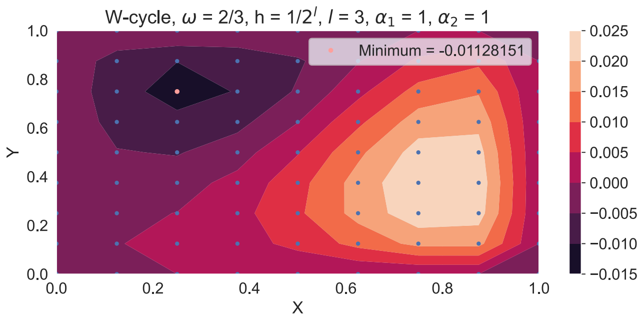

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

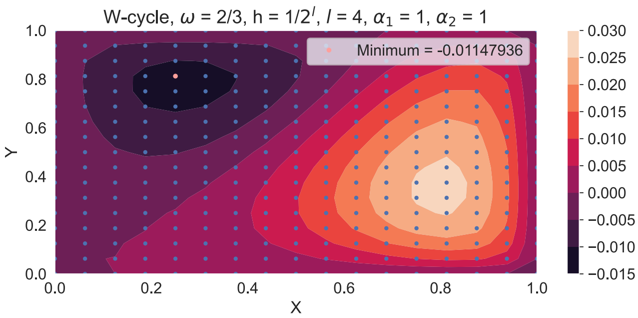

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

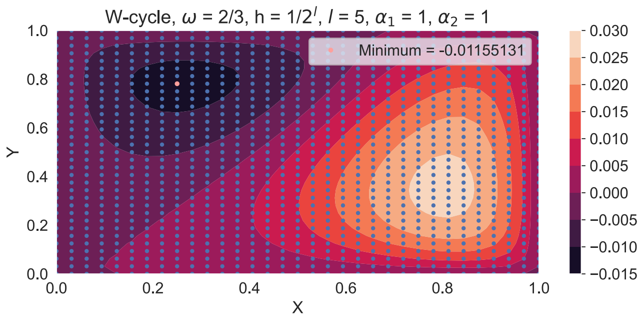

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

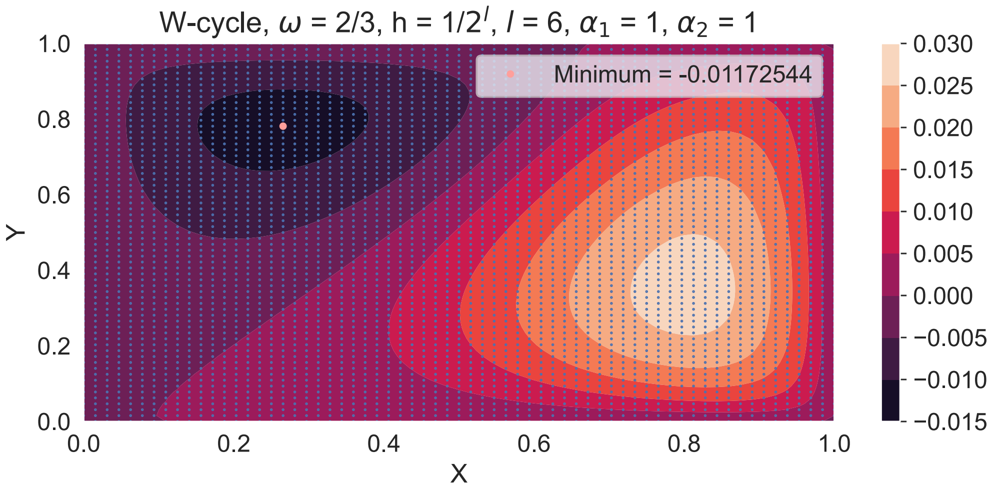

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

No plots are shown for larger l values since the difference cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

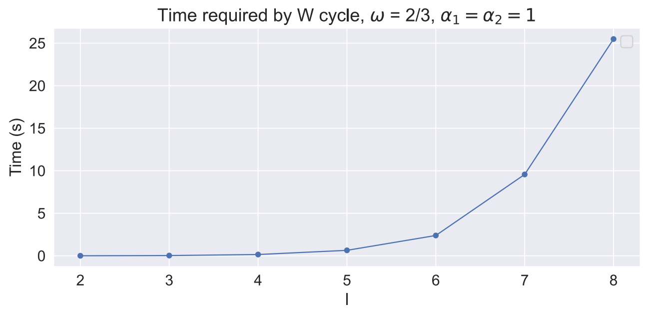

Figure: Graph of the time required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$

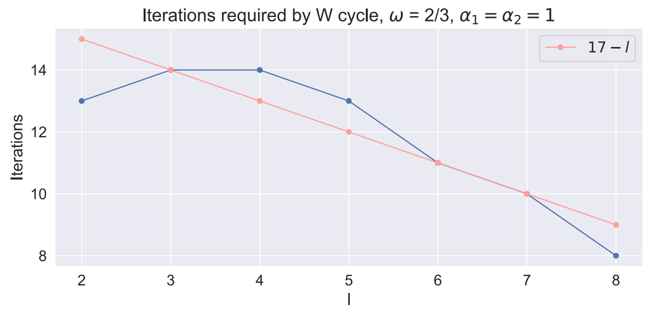

Figure: Graph of the iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$

The 2D case has a faster decrease in the number of iterations than the 1D case, but still much more iterations are necessary.

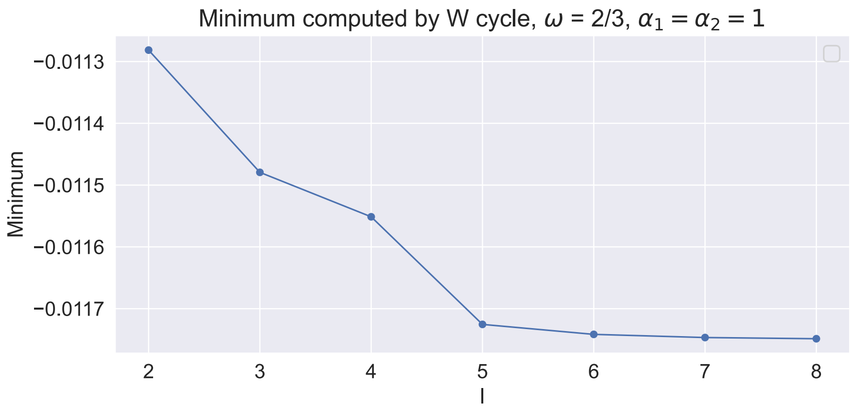

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

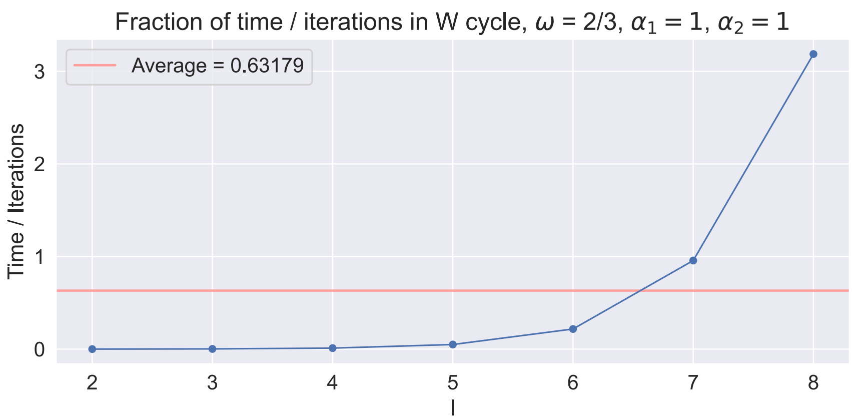

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$

Although you have longer times and more iterations, the time/iteration ratio is lower in the 2D case.
So far, this choice of α and ω gives better results than the other W-cycles in terms of time and iterations.

Case: $\alpha_1 = \alpha_2 = 2$, $\omega = 2/3$

h	l	**Iterations**	**Time**	$\min_{x\in\overline{\Omega}}$
0.25	2	7	0.00397s	-0.01128152
0.125	3	8	0.0379s	-0.01147935
0.0625	4	8	0.13512s	-0.01155131
0.03125	5	7	0.45589s	-0.01172543
0.015625	6	6	2.15296s	-0.01174159
0.0078125	7	5	6.31451s	-0.01174671
0.00390625	8	4	19.23703s	-0.01174864

Investigation of the W cycle performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$

Results follow in graph form:

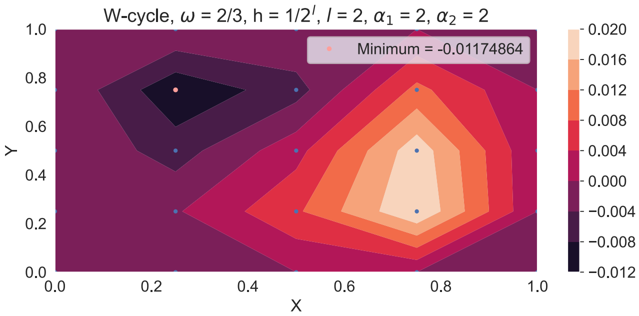

Figure: Graph of the solution approximated by the W-cycle, $h = 0.25$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

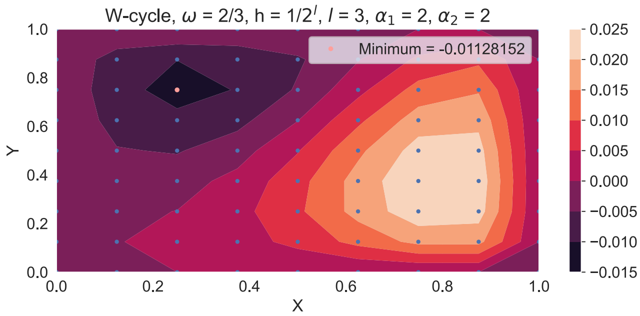

Figure: Graph of the solution approximated by the W-cycle, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

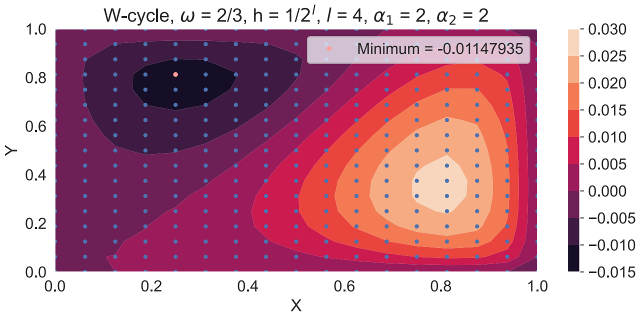

Figure: Graph of the solution approximated by the W-cycle, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

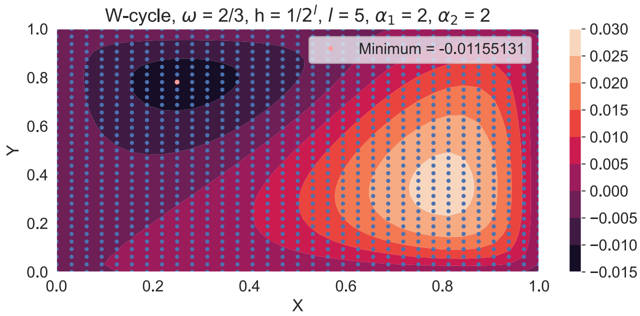

Figure: Graph of the solution approximated by the W-cycle, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

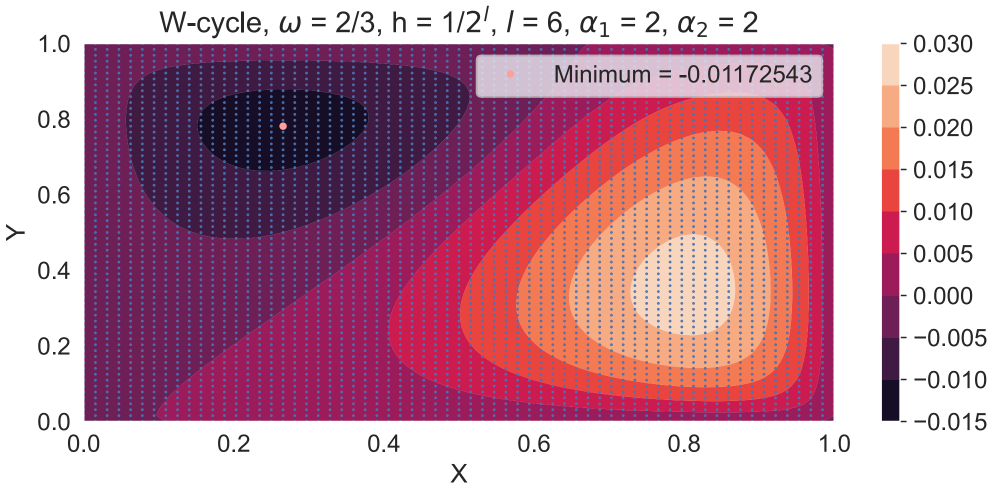

Figure: Graph of the solution approximated by the W-cycle, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

No plots are shown for larger l values since the difference cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

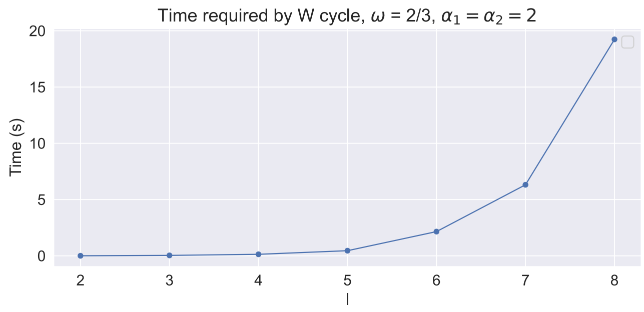

Figure: Graph of the time required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$

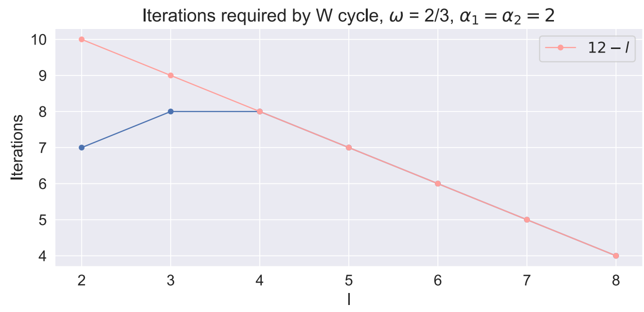

Figure: Graph of the iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$

Again there is a faster decrease in the number of iterations of the 1D case.

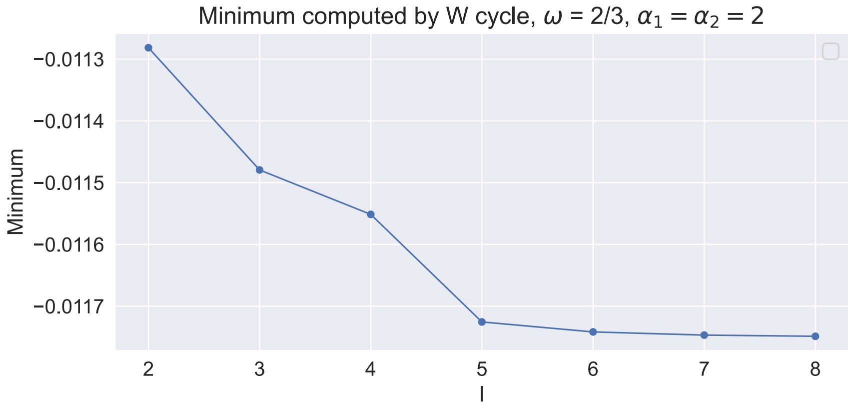

Figure: Graph of the trend of the minimum computed by the W-cycle, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

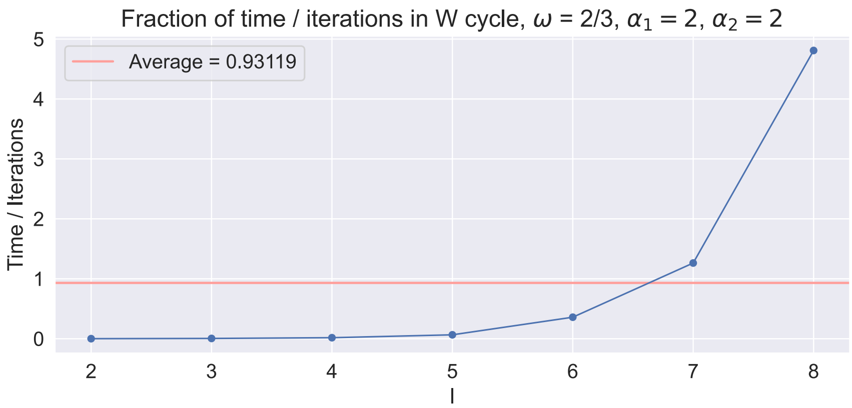

Figure: Graph of the fraction time / iterations required by the W-cycle, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$

Although you have a lower time/iteration ratio, you clearly have longer times and more iterations than the 1D case. However, we can interpret the lower fraction as a higher efficiency of the 2D method.

Of all the W-cycle methods, this is the most efficient in terms of both time and iterations.

`full_mg_2d(uh, fh, omega, alpha1, alpha2, nu)`

The following section investigates a single step of the full grid scheme with $\omega = 2/3$ in the 2D problem stated in the project.

Case: $\alpha_1 = \alpha_2 = 1$, $\omega = 2/3$, $\nu = 1$

h	l	$\lvert r_h \rvert_\infty$	**Time**	$\mathbf{min}_{x \in \overline{\Omega}}$
0.25	2	0.226	0.001s	-0.00751159
0.125	3	0.56286	0.00299s	-0.01001862
0.0625	4	0.77381	0.0279s	-0.01127217
0.03125	5	0.89402	0.13564s	-0.01169105
0.015625	6	0.95234	0.36223s	-0.01173659
0.0078125	7	0.98205	1.51908s	-0.01174518
0.00390625	8	0.99705	5.90759s	-0.01174796

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 1$

Results follow in graph form:

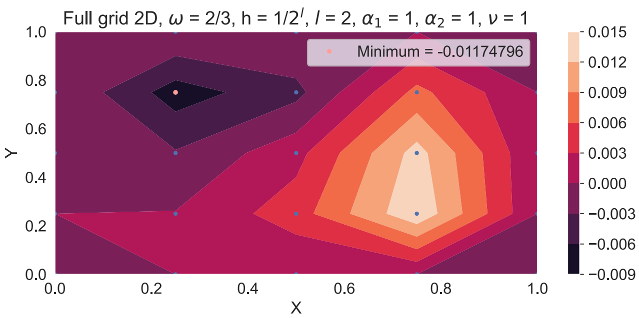

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.25$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

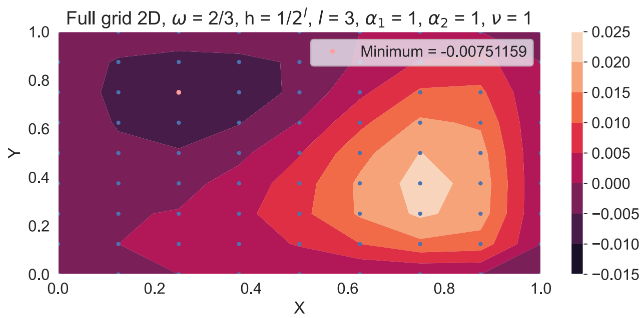

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

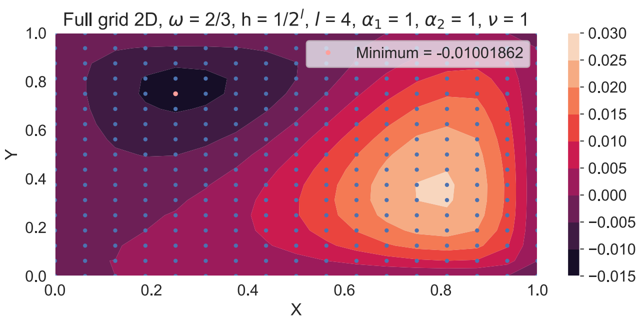

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

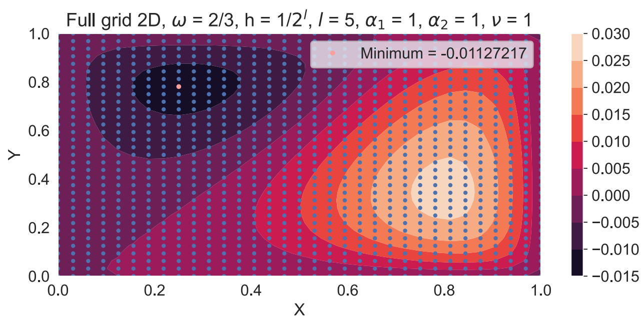

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

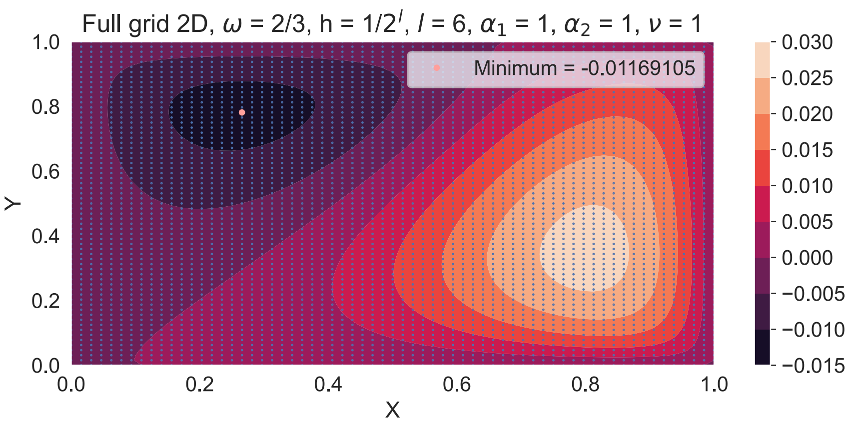

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

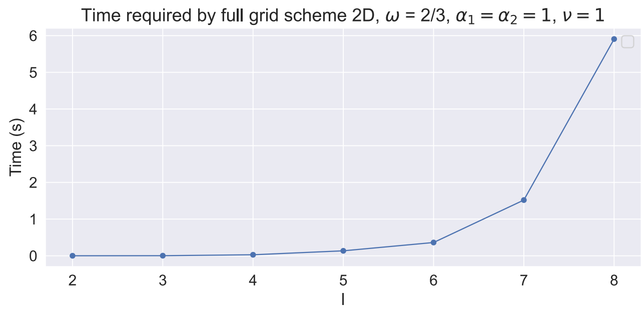

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 1$

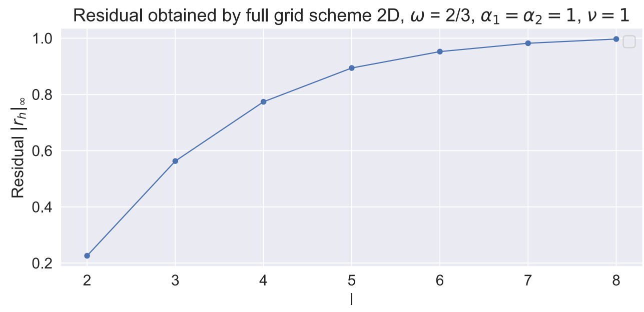

Figure: Graph of the iterations required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 1$

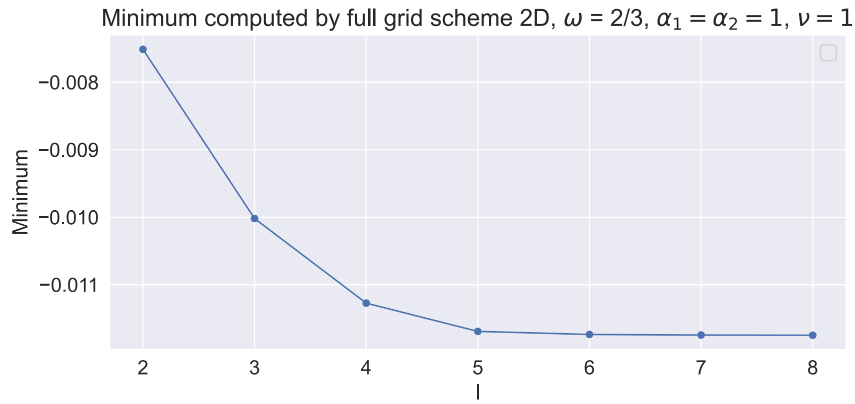

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 1$

It can be seen that the residual increases as h decreases. Despite this, the times are significantly less than the previous cases and are the best times for all the cases analyzed in 2D.

Again you have a large increase in time but mostly extremely larger residuals than the 1D case.

Among the 2D methods, this one is definitely better than the others in terms of timing.

Case: $\alpha_1 = \alpha_2 = 1$, $\omega = 2/3$, $\nu = 2$

h	l	$\lvert r_h \rvert_\infty$	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.25	2	0.06947	0.001s	-0.01003963
0.125	3	0.18607	0.00797s	-0.01113415
0.0625	4	0.25259	0.0748s	-0.01153149
0.03125	5	0.27861	0.19514s	-0.01172597
0.015625	6	0.2896	0.63521s	-0.01174112
0.0078125	7	0.29361	2.61826s	-0.01174655
0.00390625	8	0.2956	12.27261s	-0.01174856

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 2$

Results follow in graph form:

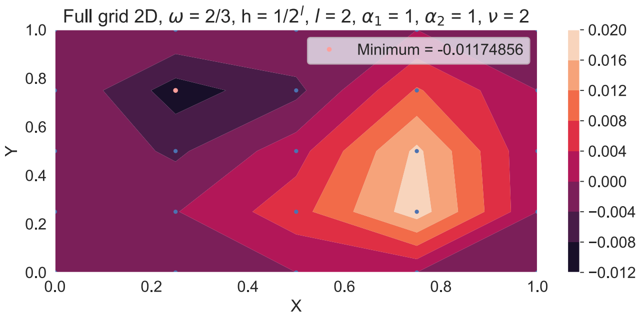

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.25$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

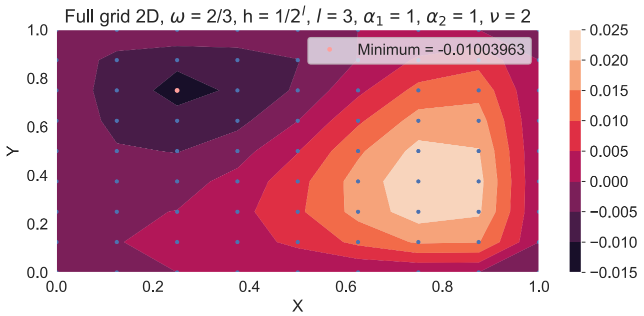

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

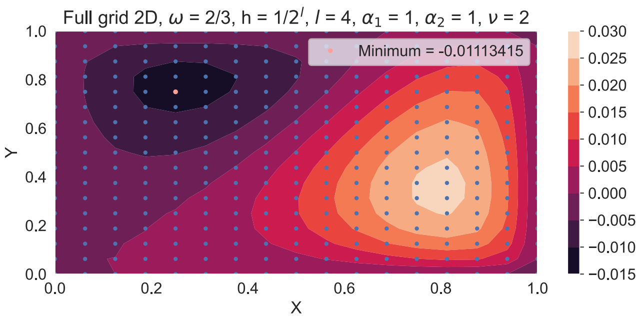

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

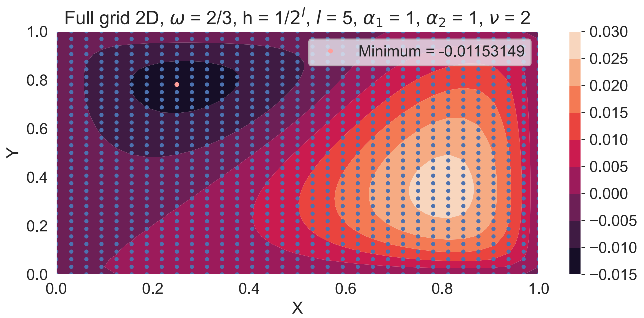

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

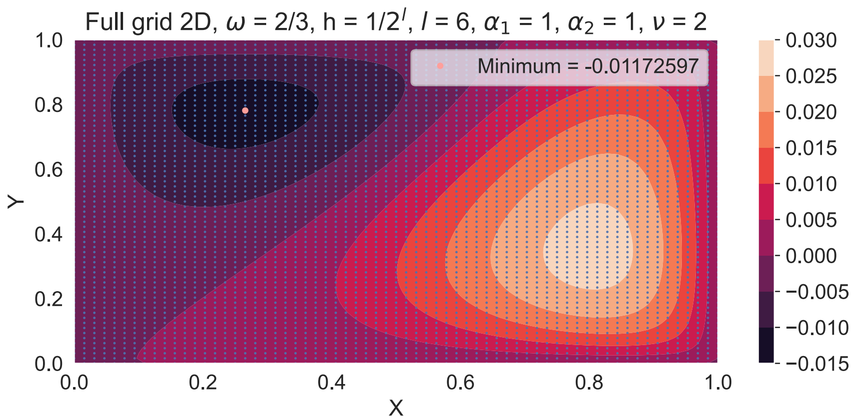

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

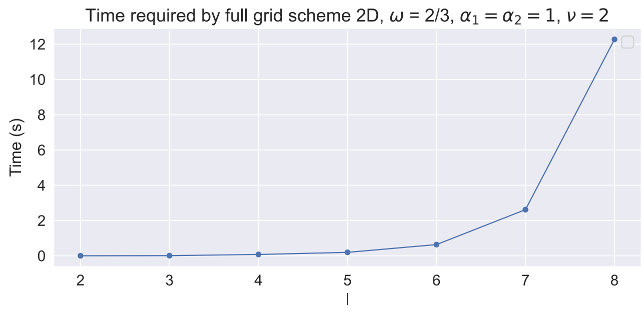

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 2$

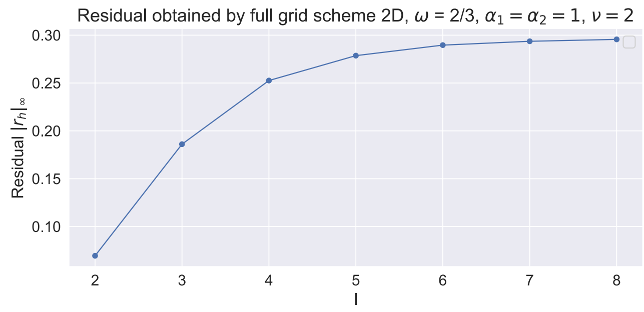

Figure: Graph of the iterations required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 1$, $\nu = 2$

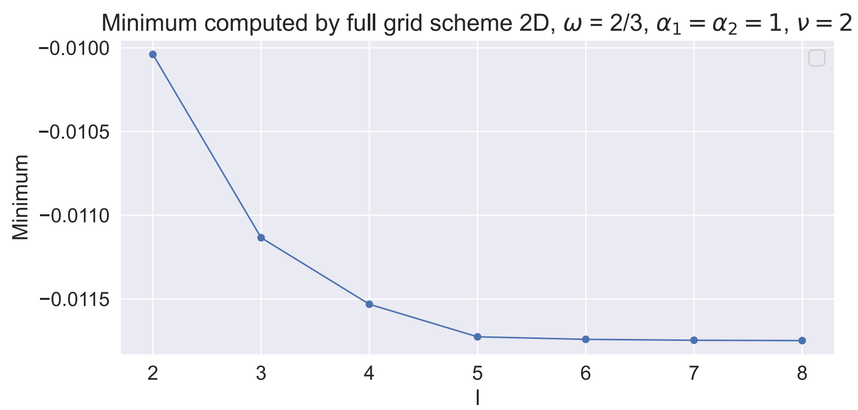

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 1$, $\alpha_2 = 1$, $\nu = 2$

The times are slightly longer than the previous case, but they are very competitive with all other methods. The worse times, however, are justified with a lower residual per l value. This method seems a good compromise between time and residue.

Case: $\alpha_1 = \alpha_2 = 2$, $\omega = 2/3$, $\nu = 1$

h	l	$\|r_h\|_\infty$	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.25	2	0.07092	0.001s	-0.01014942
0.125	3	0.20282	0.00998s	-0.01077131
0.0625	4	0.26918	0.07185s	-0.01143714
0.03125	5	0.29764	0.13469s	-0.01171493
0.015625	6	0.30952	0.48108s	-0.01173956
0.0078125	7	0.31473	2.08978s	-0.01174617
0.00390625	8	0.31704	12.54956s	-0.01174847

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 1$

Results follow in graph form:

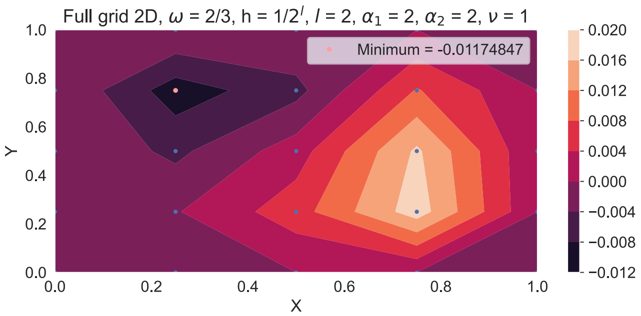

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.25$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

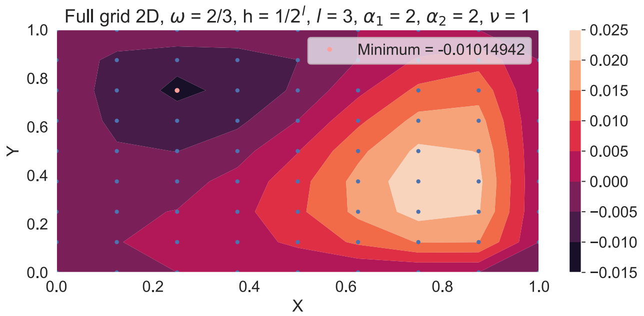

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

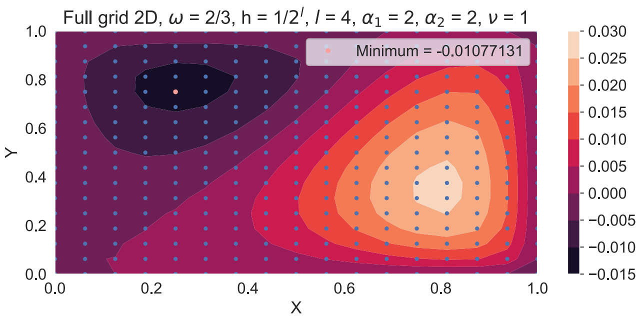

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

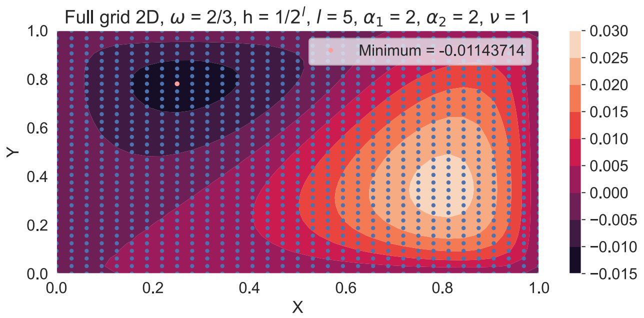

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

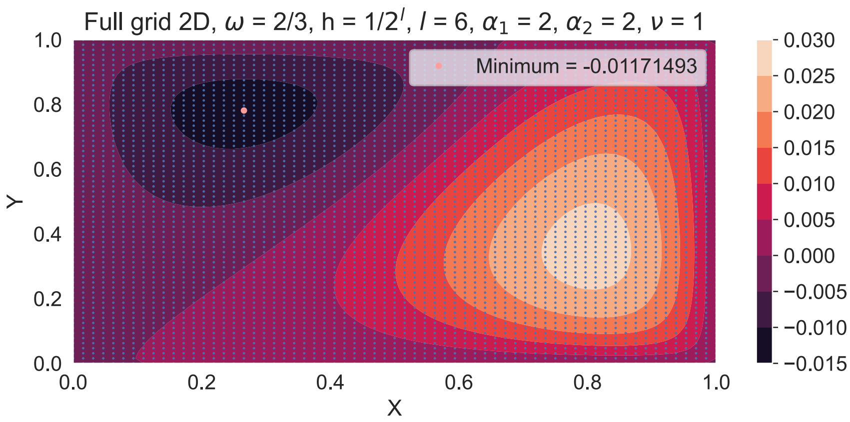

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

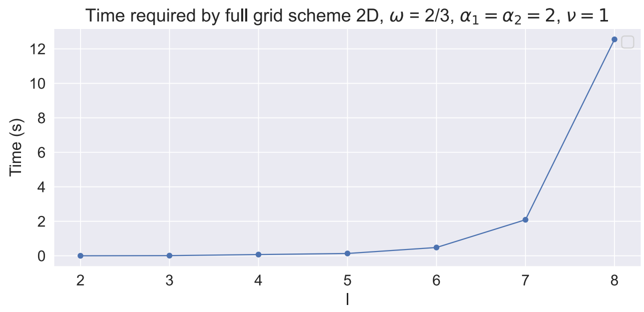

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 1$

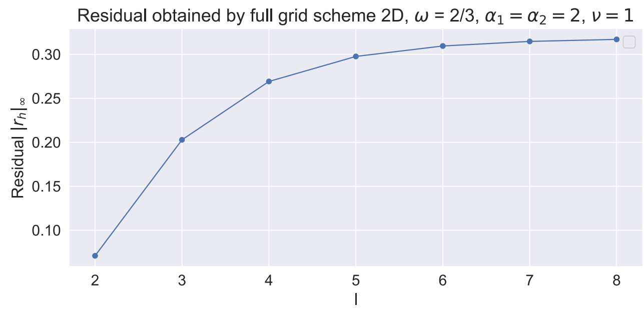

Figure: Graph of the iterations required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 1$

The time and residual trends are similar to all other combinations of parameters in the method.

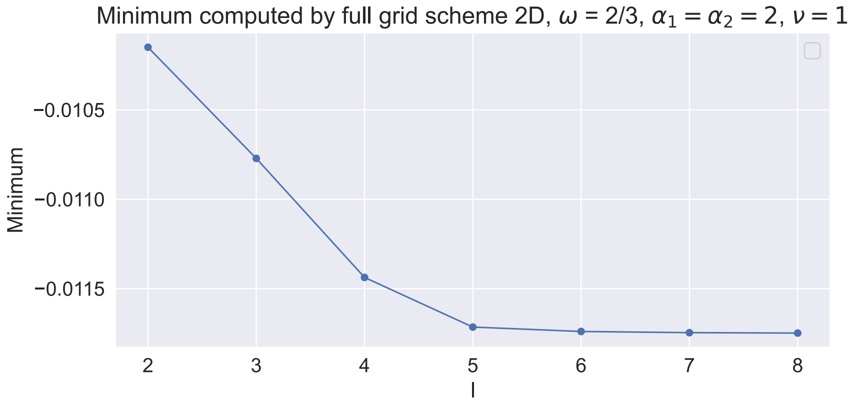

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 1$

The times are slightly less than the previous case, but the residuals are slightly higher. In these terms, it is difficult to say whether there is an improvement over the previous case.

Case: $\alpha_1 = \alpha_2 = 2,\ \omega = 2/3,\ \nu = 2$

h	l	$\lvert r_h \rvert_\infty$	**Time**	$\mathbf{min}_{x\in\overline{\Omega}}$
0.25	2	0.00755	0.00199s	-0.01118349
0.125	3	0.0315	0.01397s	-0.01138317
0.0625	4	0.04141	0.08247s	-0.01154723
0.03125	5	0.04475	0.24679s	-0.01172574
0.015625	6	0.04583	1.12247s	-0.01174158
0.0078125	7	0.04619	4.31917s	-0.0117467
0.00390625	8	0.04633	23.27679s	-0.01174863

Investigation of the full grid scheme performance with $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 2$

Results follow in graph form:

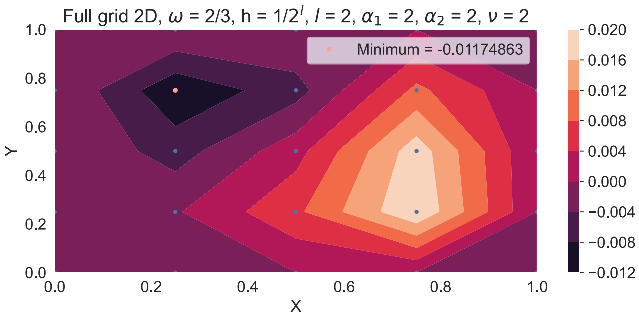

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.25$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

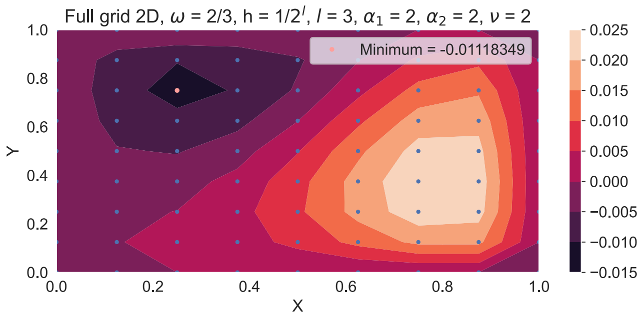

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

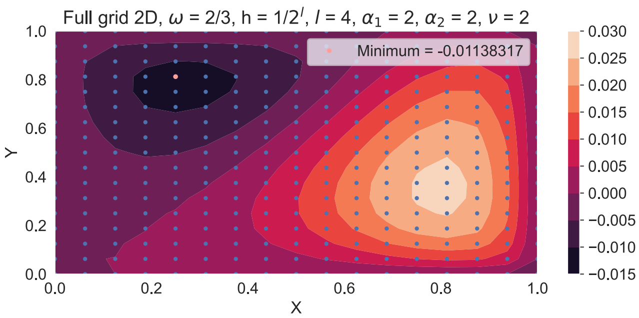

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.0625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

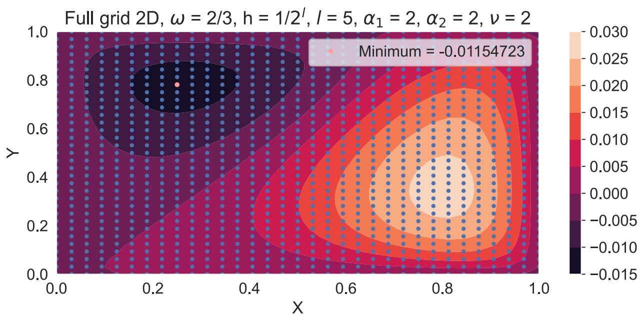

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.03125$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

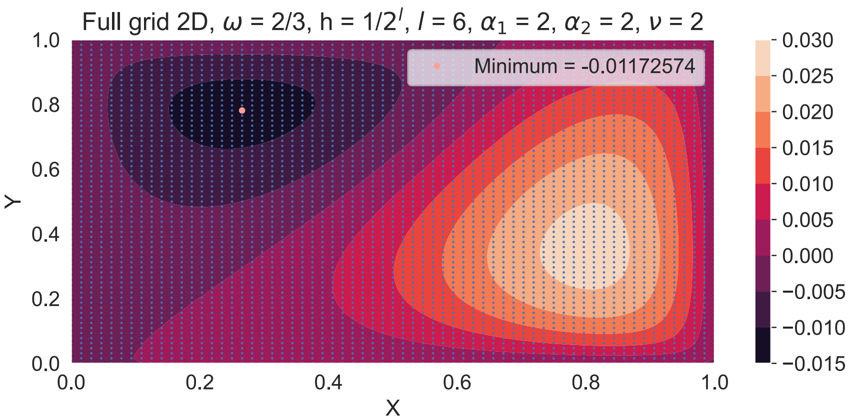

Figure: Graph of the solution approximated by the full grid scheme, $h = 0.015625$, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

No graphs are shown for bigger l, as the improvement in approximation cannot be seen with the naked eye.

Plotting now the iterations, time and minimum values gives the following:

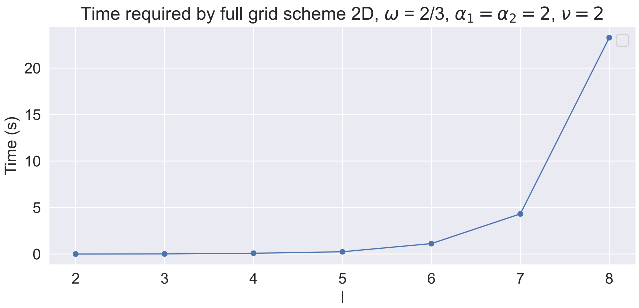

Figure: Graph of the time required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 2$

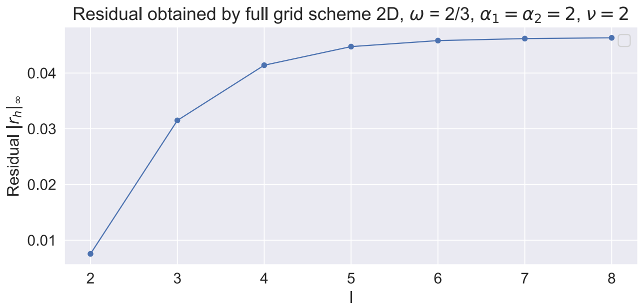

Figure: Graph of the iterations required by the full grid scheme, $\omega = 2/3$, $\alpha_1 = \alpha_2 = 2$, $\nu = 2$

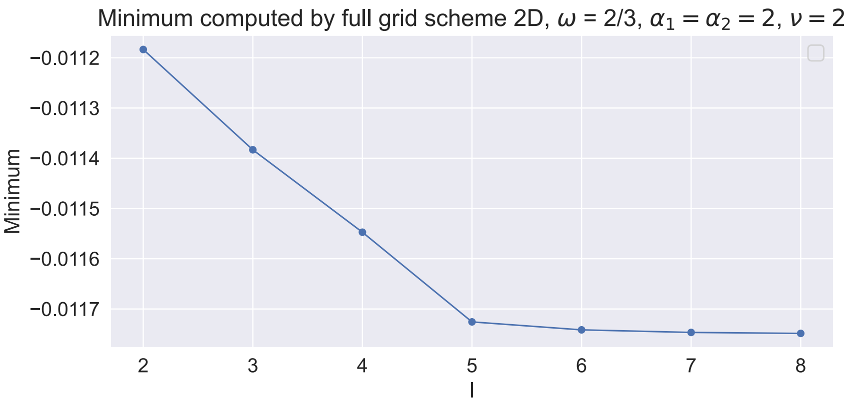

Figure: Graph of the trend of the minimum computed by the full grid scheme, $\omega = 2/3$, $\alpha_1 = 2$, $\alpha_2 = 2$, $\nu = 2$

Again timing and residuals have similar behavior to previous cases. For full grids we get the highest times in this case, but in return we have the lowest residuals.

So the choice of α_i and ν combination depends on what is desired between convergence speed and accuracy (in terms of residuals). This case is preferred for precision, i.e., small residuals.

So results similar to the 1D case were obtained, where some combinations lead to lower residuals and others to lower times.

Conclusion - 2D

Exactly like the 1D case, it is not possible to define which method is best, as there are too many parameters to consider.

Implementing the 2D version of all 1D methods was very simple, of consequence the same consideration made about the complexity of implementation applies to the 2D case.

Again, the Jacobi method offers long times and many iterations at the expense of a very simple implementation. The $\omega = 2/3$ case is better in both iterations and times, with improvement of about twice as much.

The 2D case also shows improvement in the two grid correction case. Nevertheless, it is not an advisable method for values of l greater than 6, as the timescales are inaccessible. Again, the $\omega = 2/3$ case is better in terms of iterations and times.

The W-cycle method is an excellent choice for the problem considered in the project. The times are less overall and so are the iterations, and the best performance of this method is obtained for $\alpha_1 = \alpha_2 = 2$ and $\omega = 2/3$.

Although the full grid scheme is very similar to the W-cycle in the $\nu = 1$ case, there is a noticeable difference in timing between the two methods for the $\alpha_1 = \alpha_2 = 1$ $\omega = 2/3$ case. However, this case also has a large residual.

Like the 1D case, the $\nu = 2$ choice has larger times and smaller residuals.

Clearly, the choice of alpha and nu values depends on the desired residuals and times.

Analyzing all the methods, for small values of h (and with sometimes very high times) several of them arrive at a minimum value of -0.011748. There is a greater discrepancy on the next decimal digits (decimal digits up to the eighth are considered, since a tolerance `tol`$=10^{-8}$ is required)

In the full grid method with smaller residual -0.01174863 is found, which perhaps should be regarded as more reliable.

Overall conclusions

In conclusion, multi grid methods offer better results in terms of time and iterations than the single grid method in both 1D and 2D.

The choice of multi grid method is to be made according to the problem and based on time, iterations and residuals demand.