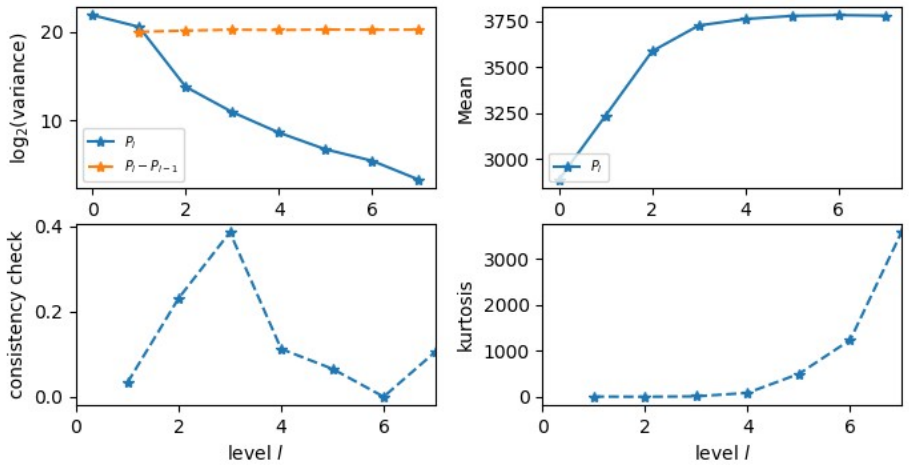
Practical 3

We initialize the problem as follows. We choose 100 initial samples on each level, and impose that the levels used by the MLMC routine are between 2 and 10. For the convergence tests, we used the first 7 levels, with 20’000 paths for each coupling between level l and level l-1.

We first look at the convergence results when we use a refinement factor of M = 3 and an initial number of step at layer 0 of n0 = 9.




```
*********************************************************************************
                *** Convergence tests, kurtosis, telescoping sum check ***
                ******* using N =  20000 samples                 *******
*********************************************************************************

 l     ave(Pf-Pc)     ave(Pf)     var(Pf-Pc)     var(Pf)        kurtosis       check          cost
---------------------------------------------------------------------------------------------------
 0  2.8874e+03  2.8874e+03  3.8290e+06  3.8290e+06  5.7218e+00  0.0000e+00  1.0000e+00
 1  3.5166e+02  3.2361e+03  1.5255e+06  1.0361e+06  7.1935e+00  3.3235e-02  3.0000e+00
 2  3.6206e+02  3.5874e+03  1.4275e+04  1.1418e+06  3.7094e+00  2.3148e-01  9.0000e+00
 3  1.2215e+02  3.7277e+03  1.9826e+03  1.2250e+06  1.2290e+01  3.8606e-01  2.7000e+01
 4  4.0745e+01  3.7631e+03  3.9989e+02  1.1957e+06  8.8003e+01  1.1255e-01  8.1000e+01
 5  1.3628e+01  3.7799e+03  1.0887e+02  1.2382e+06  5.0056e+02  6.5891e-02  2.4300e+02
 6  4.4656e+00  3.7843e+03  4.4435e+01  1.2180e+06  1.2370e+03  5.0735e-04  7.2900e+02
 7  1.5137e+00  3.7809e+03  1.0228e+01  1.2296e+06  3.5736e+03  1.0518e-01  2.1870e+03
```

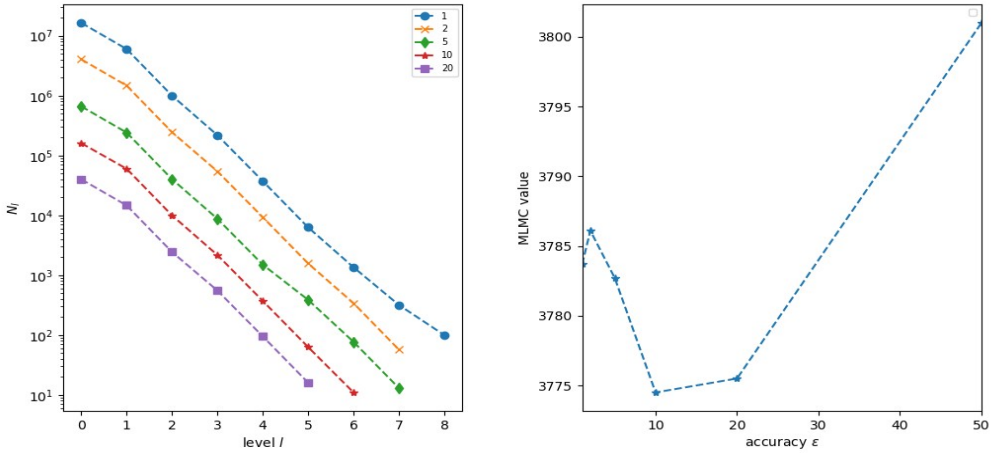
We observe that the mean and the variance of the estimators are of the right order. However, we failed to achieve the same value as in Anderson & Higham (2011), which was around 3713. We believe that this small bias is introduced in the way we deal with the initial steps of the algorithm. Indeed, at first, only G is present in the mix. Hence, the first reaction of the simulation has to be the reaction $G \rightarrow G + M$. But the reaction $M \rightarrow M + P$ happens 40 times more often, so we must discard this reaction if it is triggered first. But even correcting this bias is not sufficient to get to their value.

We now run the main MLMC code to see the number of samples needed in each level, and the associated saving for a fixed error.

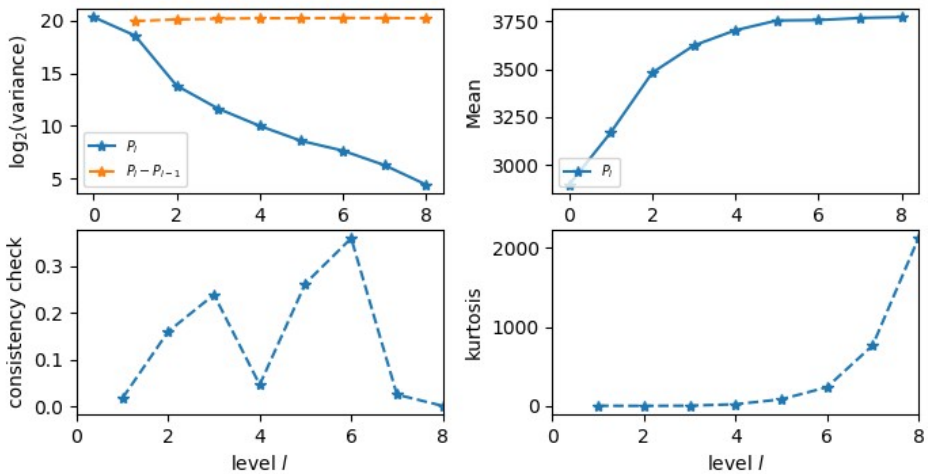
```
***********************************************************
*** Linear regression estimates of MLMC parameters ***
***********************************************************

 alpha = 1.408080  (exponent for MLMC weak convergence)
 beta  = 2.585748  (exponent for MLMC variance)
 gamma = 1.584962  (exponent for MLMC cost)

*******************************
*** MLMC complexity tests ***
*******************************
```

eps	value	mlmc_cost	std_cost	savings	l=2	l=3	l=4	l=5	l=6	l=7	l=8	l=9	l=10
1.0000	3.7837e+03	1.685e+08	3.227e+10	191.54	16436148	5989478	997975	218057	37273	6290	1356	318	100
2.0000	3.7861e+03	4.087e+07	8.964e+08	21.93	4038204	1468972	244762	54359	9299	1570	341	57	
5.0000	3.7827e+03	6.849e+06	1.434e+08	20.94	660806	241253	40198	8761	1499	387	77	13	
10.000	3.7745e+03	1.626e+06	1.184e+07	7.28	160426	59861	9975	2173	372	63	11		
20.000	3.7755e+03	3.972e+05	1.003e+06	2.53	39946	14645	2441	554	95	16			
50.000	3.8010e+03	6.064e+04	1.605e+05	2.65	6343	2163	361	84	14	3			

We see that the savings for eps=1 are huge: a factor 190. Standard Monte Carlo would have been impractical in this case.



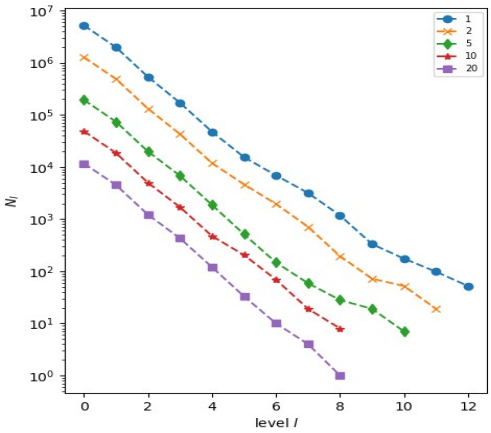
We redo the same experiments with a refinement factor of M = 2 and an initial number of step at layer 0 of n0 = 4. For the convergence tests, we used the first 7 levels, with 20’000 paths for each coupling between level l and level l-1. The results we got are fairly similar to the case M = 3 above.

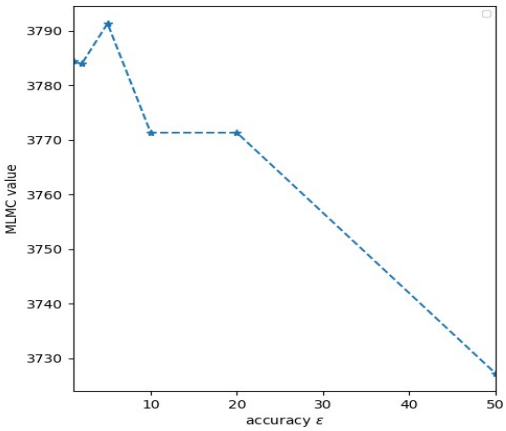


For the number of samples per level for a fixed error, we observe the same trend as in the case M=3.

```
***********************************************************
*** Linear regression estimates of MLMC parameters ***
***********************************************************

 alpha = 0.899725  (exponent for MLMC weak convergence)
 beta  = 1.788249  (exponent for MLMC variance)
 gamma = 1.000000  (exponent for MLMC cost)

*********************************
*** MLMC complexity tests ***
*********************************
```

eps	value	mlmc_cost	std_cost	savings	l = 2	l = 3	l = 4	l = 5	l = 6	l = 7	l = 8	l = 9	l = 10	l = 11	l = 12	l =13	l = 14
1.0000	3.7844e+03	6.326e+07	2.674e+10	422.73	5179697	1985111	534068	170576	47318	15471	6906	3177	1176	332	174	98	52
2.0000	3.7839e+03	1.541e+07	3.343e+09	216.87	1273420	488970	131551	42773	11867	4638	1960	710	195	72	52	19	
5.0000	3.7913e+03	2.247e+06	2.674e+08	118.99	193260	73735	19838	6788	1875	513	147	59	28	19	7		
10.000	3.7713e+03	5.693e+05	4.178e+06	7.34	48650	18554	4992	1698	473	205	69	19	8				
20.000	3.7713e+03	1.335e+05	1.045e+06	7.82	11643	4523	1217	429	120	33	10	4	1				
50.000	3.7272e+03	1.984e+04	4.249e+04	2.14	1779	666	180	67	19	5	2						





We observe that the value for eps=1 is 3784.4, comparable to the 3783.7 we got in the case M=3. Hence, the bias we get compared to the value in the paper is an implementation issue, it is not coming from a discretization error in the MLMC.

The rate of weak convergence is slower : 0.89 for M = 2, and 1.40 for M = 3. We run the experiments with M = 4 as well, obtaining an even bigger rate : 1.86.

We tested as well different generators for Poisson random variables. The random number generator from the standard C++11 library is quite slow. For performance, we used the C++ CPU software at http://people.maths.ox.ac.uk/gilesm/codes/poissinv/ to transform samples from the uniform distribution to a Poisson distribution with specified lambda.