Numerical Study

0.1 Convolution Type MV-SDE

The original MV-SDE is given as :

$$\mathrm{d}X_t = \mathbb{E}\left[\exp\left(-\frac{(X_t - x)^2}{2}\right)\right]\Bigg|_{x=X_t} \mathrm{d}t + \sigma \mathrm{d}W_t, \quad X_0 = \mathcal{N}_{(0,1)}.$$

Note that above is of the form of equation (1.1) in Belomestny-Schoenmakers 2018. In paticular, we are interested in the approximation:

$$\mathrm{d}X_t = \sum_{k=0}^{K} \gamma_k(t)\alpha_k(X_t)\mathrm{d}t + \sigma \mathrm{d}W_t, \quad X_0 = \mathcal{N}_{(0,1)}.$$

The above is of of the form of equation (2.3) in Belomestny-Schoenmakers 2018. In the MV-SDE we see that:

- $K = 0, \cdots, 20$, $d = 1$ and $q = 1$;
- $\varphi_k(x) = \overline{H}_k(x)\mathrm{e}^{-x^2/2}$, $\gamma_k(t) = \mathbb{E}[\varphi_k(X_t)]$, for $k = 0, \cdots, K$;
- $\alpha_k(x) = \pi^{1/4}\left(\frac{1}{2}\right)^{k/2} \frac{x^k}{\sqrt{k!}}\mathrm{e}^{-x^2/4}$, for $k = 0, \cdots, K$;
- $\beta(x) = (\sigma, 0, \cdots, 0)^T$.

Also:

$$\alpha_k'(x) = -\pi^{1/4}\left(\frac{1}{2}\right)^{k/2+1} \frac{x^{k-1}(x^2 - 2k)}{\sqrt{k!}}\mathrm{e}^{-x^2/4}, \qquad \text{for } k = 0, \cdots K.$$

In the above, $\overline{H}_k(x)$ are the normalised Hermite physics polynomials:

$$\overline{H}_k(x) = c_k(-1)^k \mathrm{e}^{x^2} \frac{\mathrm{d}^k}{\mathrm{d}x^k}\left(\mathrm{e}^{-x^2}\right) = c_k \mathrm{e}^{x^2/2}\left(x - \frac{\mathrm{d}}{\mathrm{d}x}\right)^k \mathrm{e}^{-x^2/2},$$

$$c_k = \left(2^k k! \sqrt{\pi}\right)^{-1/2}.$$

In our numerical study, the results and the observations are obtained from the work of two macro functions, both aimed at the approximate calculation of the values of $\gamma_0, \cdots \gamma_K$.

- The first one, which we use to obtain the solution that will play the role of benchmark, is the well-known Euler - Monte Carlo method. It consists in approximating the two expected values with a direct average of $M_1 = 10^7$ simulations of processes $(X_t)_{t=0,\cdots T}$ obtained by dividing the interval $[0, T]$ into $100+1$ time steps and applying Euler's method to each of them. Specifically, Euler's Method calculates the 101 values of a discretized process by obtaining, at each step, the value at time t from the value of the process at the previous time and the value of the time step h, according to the formula:

$$X_{t+1} = X_t + \text{drift}(t) \cdot h + \text{diffusion}(t) \cdot \sqrt{h} \cdot W,$$

where W is the sample of a Standard Normal. In our case the time steps are all equivalent since we divide the interval $[0, T]$ into equispaced points. We observe that the function is structured in such a way that it can also be applied to McKean-Vlasov SDEs, for which Euler's method at step $t+1$ requires that the expected values sought, at time t, are known within the drift and diffusion parts. To obtain these we again use an average over the M_1 simulations of the realisations of the processes at time t.

- The second one is the Stochastic Gradient Descent method. This returns us as output the polynomials $(\mathcal{L}(a))_0(t), \cdots, (\mathcal{L}(a))_K(t)$ which will be the approximations of $\gamma_0, \cdots, \gamma_K$ respectively. As for the choice of the space of polynomials, the two outputs are calculated in the basis of orthogonal Lagrange polynomials. In the test we will change the dimension n of the space of polynomial. Each element of the basis is of the form:

$$g_i(t) := \prod_{j \leq n \text{ and } j \neq n} \left(\frac{t - t_j}{t_i - t_j} \right), \text{ with Chebyshev knots } \frac{T}{2} + \frac{T}{2} \cos\left(\frac{2k+1}{2n+2} \pi \right).$$

Again, we will use to the N Euler steps to find the solution of the SDEs. This time, Euler's method for simulating $Z(\xi, W)$ and $\left(Z^a(\tilde{\xi}, \tilde{W}), \partial_{a_{h,j}} Z^a(\tilde{\xi}, \tilde{W}) \right)$ will have to carry out 3 processes simultaneously: $(X_t)_{t=0,\cdots T}$ and $(Z_t)_{t=0,\cdots T}$ one-dimensional and $(Y_t)_{t=0,\cdots T}$ $(K+1)\times(n+1)$-dimensional. Furthermore, it will be necessary to use at each step the value obtained for the process X in order to calculate Y. We note that X and Z implement the Euler step in the same way as the previous function, but with two different samples of the Brownian. In this simplified algorithm the maps $\mathbf{h}$ and H are taken as the identity and the null function, respectively. By taking the values of the coefficient functions for the MV-SDE relative to the convolution-type model, we obtain that specifically the equations become:

$$\mathrm{d}Z_t = \Big((\mathcal{L}a)_0(t)\alpha_0(Z_t) + \cdots + (\mathcal{L}a)_K(t)\alpha_K(Z_t) \Big) \mathrm{d}t + \sigma \mathrm{d}W_t, \quad Z_0 = \mathcal{N}_{(0,1)}.$$

$$(\mathcal{L}a)_{l=0,\cdots,K}(t) = \sum_{i=1}^{n} a_{l,i} \prod_{\substack{1\le j\le n\\ i\neq j}} \frac{t-t_j}{t_i-t_j}.$$

$$\begin{aligned}
\mathrm{d}Y_t^{i,0} = \big(& \prod_{\substack{1\le j\le n\\ i\neq j}} \frac{t-t_j}{t_i-t_j}\alpha_0(Z_t) \\
& + (\mathcal{L}a)_0(t)\alpha_0'(Z_t)Y_t^{i,0} + \cdots + (\mathcal{L}a)_K(t)\alpha_K'(Z_t)Y_t^{i,0}\big)\mathrm{d}t, \quad Y_0^{i,0} = 0, \\
\vdots & \\
\mathrm{d}Y_t^{i,K} = \big(& \prod_{\substack{1\le j\le n\\ i\neq j}} \frac{t-t_j}{t_i-t_j}\alpha_K(Z_t) \\
& + (\mathcal{L}a)_0(t)\alpha_0'(Z_t)Y_t^{i,K} + \cdots + (\mathcal{L}a)_K(t)\alpha_K'(Z_t)Y_t^{i,K}\big)\mathrm{d}t, \quad Y_0^{i,K} = 0,
\end{aligned}$$

for $j = 0, \cdots, n$ and $k = 0, \cdots, K$. These processes are necessary to calculate the realisation of the gradient for the Stochastic Descent, i.e. the random variable v. Having divided time into N time steps and approximated the integral with a summation the writing of v, component by component, is as follows:

$$\begin{aligned}
v_{j,0}(a;W;\tilde{W}) = 2h\sum_{t=0}^{N} \Big[& (\varphi_0(Z_t^a(W)) - (\mathcal{L}a)_0(t)) \left(\varphi_0'(Z_t^a(\tilde{W}))Y_t^{j,0}(\tilde{W}) - g_j(t)\right) + \cdots \\
& + (\varphi_K(Z_t^a(W)) - (\mathcal{L}a)_K(t))\,\varphi_K'(Z_t^a(\tilde{W}))Y_t^{j,0}(\tilde{W})\Big], \\
\vdots & \\
v_{j,K}(a;W;\tilde{W}) = 2h\sum_{t=0}^{N} \Big[& (\varphi_0(Z_t^a(W)) - (\mathcal{L}a)_0(t))\,\varphi_0'(Z_t^a(\tilde{W}))Y_t^{j,K}(\tilde{W}) + \cdots \\
& + (\varphi_K(Z_t^a(W)) - (\mathcal{L}a)_K(t)) \left(\varphi_K'(Z_t^a(\tilde{W}))Y_t^{j,K}(\tilde{W}) - g_j(t)\right)\Big],
\end{aligned}$$

with $j = 0, \cdots, n$. We conclude highlighting that, before returning the value v, this function averages M realisations obtained corresponding to as many independent simulations of Brownian motions. If this parameter is 1, the method is a classical SGD method, but if taken to ∞ it leads to a GD method, i.e. deterministic descent. This strategy, called Mini Batch, is the core of the numerical analyses we have done. Finally, regarding the choice of *learing rates* necessary for the calculation of v at each iteration, we choose $\eta_m = \dfrac{r_0}{(m+1)^\rho}$, where the factors $r_0 \in (0, +\infty)$ and $\dfrac{1}{2} < \rho \le 1$ will change in the tests.

0.2 Tables and Graphs

The aim of our analysis is to find the number of iterations required for the Gradient Descent method to converge. We therefore explicate the stopping criterion for the iteration of a_m: fixed $\gamma_{0,bench}, \cdots, \gamma_{K,bench}$ obtained from the first function, the iterations stops when:

$$\frac{\| \sum_{i=0}^{n} (a_0)_i g_i - \gamma_{0,bench} \|_{L_2}}{\|\gamma_{0,bench}\|_{L_2}} < 5\% \quad , \cdots , \quad \frac{\| \sum_{i=0}^{n} (a_K)_i g_i - \gamma_{K,bench} \|_{L_2}}{\|\gamma_{K,bench}\|_{L_2}} < 5\%.$$

Therefore, the second function, described in the previous section, will produce as output a solution with a relative error of 5%, in norm L_2, with respect to the benchmark solution. In order to make these results as general and correct as possible, we repeat the same test by varying: dimention $K = 5, 7, 9, 10, 15, 20$, final instant $T = 1$; dimension of the space of polynomials $n = 3$; and values of the parameters $\rho = 0.8, 0.9$ and $r_0 = 5, 10$ of the *learning rates*. In particular, we repeat for each case the same test 10 times and show: the average convergence times, a table that for each combination of ρ and r_0 shows the minimum, maximum and average number of convergence iterations and the graphs of the approximate solutions. The graphs we will show are those with the values of ρ and r_0 which in the tables have the smallest number of average iterations, for each combination of the parameters T, n and M. This is done by varying the value of the Mini Batch $M = 1, 10, 10^2, 10^3$ and the degree of the polynomials n. We group these tables and graphs into sections according to the value of T.

N.B. 0.2.1. *When values such as 'overflow' or* 49999 *appear in the tables, they respectively mean that an overflow occurred during the execution of the programme, i.e. the limit of the value storage capacity was reached (i.e. too large a number); while the second means that the algorithm did not reach convergence within the* 50000 *iterations imposed as a threshold.*

N.B. 0.2.2. *For the first three values of M, i.e. $M = 1, 10, 100$, we check the solution with the benchmark every ten iterations in order to keep execution times down. While for the last two cases, i.e. $M = 1000$, this check takes place every step.*

N.B. 0.2.3. *We specify that the algorithms were written in the Python programming language, specifically in the Jupyter Notebook web application for creating and sharing computational documents. With regard to the algorithm execution times, I write below the specifications of the machine where I ran the tests:*

Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz
Installed RAM 8.00 GB (7.90 GB usable)
System Type 64-bit operating system, x64-based processor

0.3 T = 1, n = 3 and $\epsilon = 5\%$

Bencmark: Euler - Monte Carlo execution time: 3009.03125 MC with 10^6 particles: execution time 347.46875; error: 0.0018321418319757708

K = 5

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	28.43	48.15
$\rho = 0.9$	23.11	34.66

Tabella 1: Average execution times (in seconds s) with $M = 1$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	340	2100	822	650	2160	1539
$\rho = 0.9$	390	1230	714	330	1700	1116

Tabella 2: Number of iterations m to achieve convergence with $M = 1$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	6.72	10.11
$\rho = 0.9$	4.71	4.54

Tabella 3: Average execution times (in seconds s) with $M = 10$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	40	270	141	100	320	247
$\rho = 0.9$	50	160	119	40	200	108

Tabella 4: Number of iterations m to achieve convergence with $M = 10$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	4.35	6.29
$\rho = 0.9$	2.5	5.48

Tabella 5: Average execution times (in seconds s) with $M = 100$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	10	70	27	30	80	44
$\rho = 0.9$	10	40	18	20	50	39

Tabella 6: Number of iterations m to achieve convergence with $M = 100$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	6.89	14.75
$\rho = 0.9$	6.21	10.76

Tabella 7: Average execution times (in seconds s) with $M = 1000$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	5	6	5.2	13	16	14.2
$\rho = 0.9$	4	7	5.1	12	14	12.5

Tabella 8: Number of iterations m to achieve convergence with $M = 1000$

K = 7

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	43.57	71.08
$\rho = 0.9$	30.45	55.93

Tabella 9: Average execution times (in seconds s) with $M = 1$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	640	2160	1342	780	3430	2189
$\rho = 0.9$	460	1640	933	770	2960	1723

Tabella 10: Number of iterations m to achieve convergence with $M = 1$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	8.1	13.65
$\rho = 0.9$	6.45	9.85

Tabella 11: Average execution times (in seconds s) with $M = 10$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	110	490	209	90	560	352
$\rho = 0.9$	70	240	166	110	380	254

Tabella 12: Number of iterations m to achieve convergence with $M = 10$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	4.55	7.37
$\rho = 0.9$	2.7	5.52

Tabella 13: Average execution times (in seconds s) with $M = 100$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	20	60	37	30	90	60
$\rho = 0.9$	20	30	22	20	90	45

Tabella 14: Number of iterations m to achieve convergence with $M = 100$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	6.38	16.2
$\rho = 0.9$	6.35	15.29

Tabella 15: Average execution times (in seconds s) with $M = 1000$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	5	7	5.5	13	16	14
$\rho = 0.9$	4	6	5.5	12	14	13.2

Tabella 16: Number of iterations m to achieve convergence with $M = 1000$

K = 9

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	88.58	137.58
$\rho = 0.9$	53.64	68.69

Tabella 17: Average execution times (in seconds s) with $M = 1$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	1540	3550	2377	1760	5460	3671
$\rho = 0.9$	650	1980	1437	880	3000	1813

Tabella 18: Number of iterations to achieve convergence with $M = 1$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	13.53	21.92
$\rho = 0.9$	8.5	10.56

Tabella 19: Average execution times (in seconds s) with $M = 10$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	170	400	280	160	800	468
$\rho = 0.9$	80	280	177	120	340	231

Tabella 20: Number of iterations to achieve convergence with $M = 10$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	6.2	11.17
$\rho = 0.9$	5.06	8.74

Tabella 21: Average execution times (in seconds s) with $M = 100$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	20	80	38	40	110	69
$\rho = 0.9$	20	50	31	40	80	54

Tabella 22: Number of iterations to achieve convergence with $M = 100$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	9.47	24.37
$\rho = 0.9$	9.61	21.65

Tabella 23: Average execution times (in seconds s) with $M = 1000$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	5	7	5.8	14	16	15
$\rho = 0.9$	5	8	5.9	12	14	13.3

Tabella 24: Number of iterations to achieve convergence with $M = 1000$

K = 10

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	86.38	179.41
$\rho = 0.9$	53.11	91.40

Tabella 25: Average execution times (in seconds s) with $M = 1$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	1270	3750	2148	1640	7430	4460
$\rho = 0.9$	530	2080	1322	1480	3570	2275

Tabella 26: Number of iterations to achieve convergence with $M = 1$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	17.03	25.24
$\rho = 0.9$	11.84	15.16

Tabella 27: Average execution times (in seconds s) with $M = 10$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	150	700	311	280	700	462
$\rho = 0.9$	100	340	217	140	350	279

Tabella 28: Number of iterations to achieve convergence with $M = 10$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	9.56	12.39
$\rho = 0.9$	7.20	10.15

Tabella 29: Average execution times (in seconds s) with $M = 100$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	20	70	51	40	100	66
$\rho = 0.9$	20	80	37	30	90	54

Tabella 30: Number of iterations to achieve convergence with $M = 100$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	12.31	28.44
$\rho = 0.9$	10.56	26.79

Tabella 31: Average execution times (in seconds s) with $M = 1000$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	5	8	6.3	14	16	14.6
$\rho = 0.9$	5	6	5.4	13	15	13.8

Tabella 32: Number of iterations to achieve convergence with $M = 1000$

K = 15

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	216.03	330.01
$\rho = 0.9$	119.60	197.99

Tabella 33: Average execution times (in seconds s) with $M = 1$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	1890	6440	3777	2650	9780	6116
$\rho = 0.9$	1170	3370	2220	1740	5950	3666

Tabella 34: Number of iterations to achieve convergence with $M = 1$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	34.67	85.53
$\rho = 0.9$	44.93	33.83

Tabella 35: Average execution times (in seconds s) with $M = 10$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	240	660	414	450	1130	766
$\rho = 0.9$	150	410	263	240	600	395

Tabella 36: Number of iterations to achieve convergence with $M = 10$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	21.63	40.42
$\rho = 0.9$	14.95	28.65

Tabella 37: Average execution times (in seconds s) with $M = 100$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	30	80	52	70	140	97
$\rho = 0.9$	20	50	36	30	100	68

Tabella 38: Number of iterations to achieve convergence with $M = 100$

	$r_0 = 5$	$r_0 = 10$
$\rho = 0.8$	28.44	68.61
$\rho = 0.9$	26.91	64.04

Tabella 39: Average execution times (in seconds s) with $M = 1000$

	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average	$r_0 = 10$ min	$r_0 = 10$ max	$r_0 = 10$ average
$\rho = 0.8$	5	10	6.7	14	19	16
$\rho = 0.9$	5	11	6.3	13	17	14.7

Tabella 40: Number of iterations to achieve convergence with $M = 1000$

0.4 T = 1, n = 3 and $\epsilon = 1\%$

K = 15

	$r_0 = 2$	$r_0 = 3$	$r_0 = 4$	$r_0 = 5$
$\rho = 0.9$	166.81	229.46	223.73	270.90
$\rho = 0.95$	188.42	177.45	203.26	228.31
$\rho = 0.99$	222.42	160.88	257.22	225.87

Tabella 41: Average execution times (in seconds s) with $M = 100$

	$r_0 = 2$ min	$r_0 = 2$ max	$r_0 = 2$ average	$r_0 = 3$ min	$r_0 = 3$ max	$r_0 = 3$ average	$r_0 = 4$ min	$r_0 = 4$ max	$r_0 = 4$ average	$r_0 = 5$ min	$r_0 = 5$ max	$r_0 = 5$ average
$\rho = 0.9$	140	850	405	210	820	482	320	750	517	440	1170	645
$\rho = 0.95$	170	1040	457	150	810	433	280	780	484	210	840	508
$\rho = 0.99$	240	1230	540	260	760	391	240	1220	601	200	850	521

Tabella 42: Number of iterations to achieve convergence with $M = 100$

0.5 Observations and Conclusions