A Neural Network Approach to Value R&D Compound American Exchange Option

Giovanni Villani¹

Accepted: 6 June 2021 / Published online: 14 July 2021
© The Author(s) 2021

Abstract

In this paper we show as the neural network methodology, coupled with the Least Squares Monte Carlo approach, can be very helpful in valuing R&D investment opportunities. As it is well known, R&D projects are made in a phased manner, with the commencement of subsequent phase being dependent on the successful completion of the preceding phase. This is known as a sequential investment and therefore R&D projects can be considered as compound options. In addition, R&D investments often involve considerable cost uncertainty so that they can be viewed as an exchange option, i.e. a swap of an uncertain investment cost for an uncertain gross project value. Finally, the production investment can be realized at any time before the maturity date, after that the effects of R&D disappear. Consequently, an R&D project can be considered as a compound American exchange option. In this context, the Least Squares Monte Carlo method is a powerful and flexible tool for capital budgeting decisions and for valuing American-type options. But, using the simulated values as “targets”, the implementation of a neural network allows to extend the results for any R&D valuation and to abate the waiting time of Least Squares Monte Carlo simulation.

Keywords Monte Carlo simulation · Neural network · Real options · R&D investments

JEL Classification C15 · C45 · G13 · O32

✉ Giovanni Villani
giovanni.villani@zuniba.it

¹ Department of Economics and Finance, University of Bari, Largo Abbazia S. Scolastica, 53, 70124 Bari, Italy

1 Introduction

R&D investments are considered an important driving force for the growth of the modern economy. For the analysts, is very important to value these investments considering their uncertainty. As it is well-know, R&D projects are characterized by the sequentiality of their investments and by the flexibility to realize the production investment at any time before the expiration time of R&D innovation. In this scenario, the real option approach can capture these aspects, unlike the Net Present Value (NPV) and the Internal Rate of Return (IRR) that underestimate R&D projects. In particular, the R&D projects can be considered as compound American exchange option (CAEO) in which both the gross project value and the investment cost are uncertain. Papers that deal with exchange option valuation are Margrabe (1978), McDonald and Siegel (1985), Carr (1988), Carr (1995), Armada et al. (2007) and so on. In particular way, McDonald and Siegel (1985) value a simple European exchange option, Carr (1988) develops a model to price a compound European exchange option while Armada et al. (2007) propose a Richardson extrapolation in order to value a simple American exchange option.

These models consider that assets distribute "dividends" that, in real options context, are the opportunity costs if an investment project is postponed Myers (1977).

However, the analytical computation of CAEO is more difficult and it is convenient to implement a numerical method. Numerical approximation is therefore an important task as witnessed by the contributions of Tilley (1993), Barraquand and Martineau (1995), Broadie and Glasserman (1997).

The first goal of our paper is to implement a Monte Carlo methodology in order to value a CAEO applied in the context of R&D investment. To realize this objective, based on Cortelezzi and Villani (2009) and Villani (2014), we present the Least Square Monte Carlo (LSM) proposed by Longstaff and Schwartz (2001) in order to value the CAEO. Despite this approach is valid in term of accuracy, the time required for the simulation of this kind of option is very long. Consequently, the second aim of this paper is to build a neural network architecture based on Back Propagation (BP) system using the simulation results as "targets" in the learning phase. As there is not a market valuation of CAEO, the advantages of this approach are first of all the speed and the accuracy in the computations and, after that, the possibility to extend the trained neural network to value any R&D investment project. To highlight our method, we compare BP approach with Radial Basis Function (RBF) and General Regression (GRNN).

The computing power has allowed nonlinear methods to become applicable to modeling and forecasting a host of economic and nancial relationships. Neural networks, in particular, have been applied to many of these empirical cases. For instance, Aminian et al. (2006) compare the predictive power of the linear regression model against the fully generalized nonlinear neural network, with the improvement exposing the degree of nonlinearity present in the relationship investigated. Their study uses neural networks as an efcient nonlinear regression technique to assess the validity of linear regression in modeling nancial data.

Andreou et al. (2006) show that the artificial neural network models with the use of the Huber function outperform the ones optimized with least squares. Eskiizmirli et al. (2020) approximate the unknown function of the option value using a trial function, which depends on a neural network solution and satisfies the given boundary conditions of the Black–Scholes equation. Arin and Ozbayoglu (2020) develop hybrid deep learning based options pricing models to achieve better pricing compared to Black–Scholes. The results indicate that the proposed models can generate more accurate prices for all option classes. RBF method as a meshless technique is suggested to solve time fractional Black–Scholes model for European option pricing problem Golbabai et al. (2019).

The literature that studies the real option in the neural network context is not very extensive. For instance, Ma (2016) based on real options method to construct a petroleum exploration and development projects, select the appropriate option pricing method and instance data analyzed by gas exploration and point out that the application of real option method can effectively improve the investment project evaluation. Moreover, Taubes et al. (1998) propose to use neural networks to value options approximating the value function of the dynamic program showing for each mode of operation the current state as input and yielding the mode to be chosen as output.

The paper is organized in this fashion. Section 2 presents the structure of an R&D investment and its evaluation in term of real option while, Sect. 3, illustrates the valuation of CAEO using the LSM approach. Moreover, the implementation of the neural network architecture is realized in Sect. 4 and some numerical applications are proposed in Sect. 5. Finally, Sect. 6 concludes.

2 R&D Structure as Real Option

In this section, we present a two-stage R&D investment which structure is the following: R is the research investment spent at initial time $t_0 = 0$; IT is the investment technology to develop innovation payed at time t_1 ; D is the production investment in order to obtain the R&D project's value and V is the R&D project value. Let assume that $IT = qD$ is a proportion q of asset D , so it follows the same stochastic process of D and the production investment D can be realized between t_1 and T . In particular way, investing R at time t_0 , the firm obtains a first investment opportunity that can be value as a CAEO denoted by $C(S_k, IT, t_1)$. This option allows to realize the investment technology IT at time t_1 and to obtain, as underlying asset, the option to realize the market launch. Let denote by $S_k(V, D, T - t_1)$ this option value at time t_1 , with maturity date $T - t_1$ and exercisable k times. In detail, during the market launch, the firm has got another investment opportunity to invest D between t_1 and T and to receive the R&D project value V . Specifically, using the LSM approach, the firm must decide whether to invest D or to wait at any discrete time $\tau_k = t_1 + k\Delta t$, for $k = 0, 1, 2, \dots, h$ with $\Delta t = \frac{T-t_1}{h}$ and h is the number of discretizations. In this way we capture the managerial flexibility to invest D before the maturity T and so to realize the R&D cash flows. Figure 1 depicts the R&D investment structure.

Fig. 1 R&D structure evaluated as CAEO

We assume that V and D have the following geometric Brownian motion:

$$\frac{dV_t}{V_t} = (\mu_v - \delta_v)dt + \sigma_v dZ_t^v \quad (1)$$

$$\frac{dD_t}{D_t} = (\mu_d - \delta_d)dt + \sigma_d dZ_t^d \quad (2)$$

$$\text{cov}\left(\frac{dV_t}{V_t}, \frac{dD_t}{D_t}\right) = \rho_{vd} \sigma_v \sigma_d dt \quad (3)$$

where μ_v and μ_d are the expected rates of return, δ_v and δ_d are the corresponding dividend yields, σ_v^2 and σ_d^2 are the respective variance rates, ρ_{vd} is the correlation between changes in V and D , $(Z_t^v)_{t \in [0, T]}$ and $(Z_t^d)_{t \in [0, T]}$ are two Brownian processes defined on a filtered probability space $(\Omega, \mathcal{A}, \{\mathcal{F}_t\}_{t \geq 0}, \mathbb{P})$, where Ω is the space of all possible outcomes, $\mathcal{A}$ is a sigma-algebra, $\mathbb{P}$ is the probability measure and $\{\mathcal{F}_t\}_{t \geq 0}$ is a filtration with respect to Ω space. Assuming that the firm keeps a portfolio of activities which allows it to value activities in a risk-neutral way, the dynamics of the assets V and D under the risk-neutral martingale measure $\mathbb{Q}$ are given by:

$$\frac{dV_t}{V_t} = (r - \delta_v)dt + \sigma_v dZ_t^{*v} \quad (4)$$

$$\frac{dD_t}{D_t} = (r - \delta_d)dt + \sigma_d dZ_t^{*d} \quad (5)$$

$$\text{Cov}(dZ_t^{*v}, dZ_t^{*d}) = \rho_{vd} dt \quad (6)$$

where r is the risk-free interest rate, Z_t^{*v} and Z_t^{*d} are two Brownian standard motions under the probability $\mathbb{Q}$ with correlation coefficient ρ_{vd} . After some manipulation, we get the equations for the price ratio $P = \frac{V}{D}$ and D_T under the probability $\mathbb{Q}$:

$$\frac{dP_t}{P_t} = (-\delta + \sigma_d^2 - \sigma_v \sigma_d \rho_{vd}) dt + \sigma_v dZ_t^{*v} - \sigma_d dZ_t^{*d} \quad (7)$$

$$D_T = D_0 \exp\{(r - \delta_d)T\} \cdot \exp\left(-\frac{\sigma_d^2}{2} T + \sigma_d Z_T^{*d}\right) \quad (8)$$

where D_0 is the value of asset D at initial time.

We can observe that $U \equiv (-\frac{\sigma_d^2}{2}T + \sigma_d Z_T^{*d}) \sim N(-\frac{\sigma_d^2}{2}T, \sigma_d \sqrt{T})$ and therefore $\exp(U)$ is log-normal distributed whose expectation value $E_{\mathbb{Q}}[\exp(U)] = 1$. By Girsanov's theorem, we define a new probability measure $\tilde{\mathbb{Q}}$ equivalent to $\mathbb{Q}$ whose Radon-Nikodym derivative is:

$$\frac{d\tilde{\mathbb{Q}}}{d\mathbb{Q}} = \exp\left(-\frac{\sigma_d^2}{2}T + \sigma_d Z_T^{*d}\right) \quad (9)$$

Hence, substituting in (8) we can write:

$$D_T = D_0 e^{(r-\delta_d)T} \cdot \frac{d\tilde{\mathbb{Q}}}{d\mathbb{Q}} \quad (10)$$

By the Girsanov's theorem, the evolution of processes:

$$d\hat{Z}_t^d = dZ_t^{*d} - \sigma_d dt \quad (11)$$

$$d\hat{Z}_t^v = \rho_{vd} d\hat{Z}_t^d + \sqrt{1 - \rho_{vd}^2} dZ_t' \quad (12)$$

are two Brownian motions under the risk-neutral probability space $(\Omega, \mathcal{A}, \mathcal{F}, \tilde{\mathbb{Q}})$ and Z' is a Brownian motion under $\tilde{\mathbb{Q}}$ independent of $\hat{Z}_d$. By using Eqs. (11) and (12), we can now obtain the risk-neutral price ratio P :

$$P_t = P_0 \exp\left\{\left(\delta_d - \delta_v - \frac{\sigma^2}{2}\right)t + \sigma Z_t^p\right\} \quad (13)$$

where $(\sigma_v d\hat{Z}_t^v - \sigma_d d\hat{Z}_t^d) \sim \mathcal{N}(0, \sigma \sqrt{dt}) = \sigma dZ_t^p$ and $\sigma = \sqrt{\sigma_v^2 + \sigma_d^2 - 2\sigma_v \sigma_d \rho_{vd}}$ and Z_t^p is a Brownian motion under $\tilde{\mathbb{Q}}$.

3 Valuation of CAEO Using LSM Method

The value of CAEO can be determined as the expectation value of discounted cash-flows under the risk-neutral probability $\mathbb{Q}$:

$$C(S_k, IT, t_1) = e^{-rt_1} E_{\mathbb{Q}}[\max(S_k(V_{t_1}, D_{t_1}, T - t_1) - IT, 0)] \quad (14)$$

Assuming the asset D as numeraire and using Eq.(10) we obtain:

$$C(S_k, IT, t_1) = D_0 e^{-\delta_d t_1} E_{\tilde{\mathbb{Q}}}[\max(S_k(P_{t_1}, 1, T - t_1) - q, 0)] \quad (15)$$

where $IT = q D_{t_1}$.

The market launch phase $S_k(P_{t_1}, 1, T - t_1)$ can be analyzed using the LSM method. Like in any American option valuation, the optimal exercise decision at any point in time is obtained as the maximum between immediate exercise value and expected continuation value. The LSM method allows us to estimate the conditional

expectation function for each exercise date and so to have a complete specification of the optimal exercise strategic along each path. The method starts by simulating n price paths of asset P_{t_1} defined by Eq. (13) with $\delta = \delta_v - \delta_d$. Let $\hat{P}_{t_1}^i, i = 1, \dots, n$ the simulated prices. Starting from each i th simulated-path, we begin by simulating a discretization of Eq. (13) for $k = 1, \dots, h$. The process is repeated m times over a time horizon T . Starting with the last j th price $\hat{P}_T^{i,j}$, for $j = 1 \dots m$, the option value in T can be computed as $S_0(\hat{P}_T^{i,j}, 1, 0) = \max(\hat{P}_T^{i,j} - 1, 0)$. Working backward, at time τ_{h-1} , the process is repeated for each j th path. In this case, the expected continuation value may be computed using the analytic expression for an European option $S_1(\hat{P}_{\tau_{h-1}}^{i,j}, 1, \Delta t)$. Moving backwards, at time τ_{h-1} , the management must decide whether to invest or not. The value of the option is maximized if the immediate exercise exceeds the continuation value, i.e.:

$$\hat{P}_{\tau_{h-1}}^{i,j} - 1 \geq S_1(\hat{P}_{\tau_{h-1}}^{i,j}, 1, \Delta t). \quad (16)$$

We can find the critical ratio $P_{\tau_{h-1}}^*$ that solve the inequality (16):

$$P_{\tau_{h-1}}^* - 1 = S_1(P_{\tau_{h-1}}^*, 1, \Delta t)$$

and so the condition (16) is satisfied if $\hat{P}_{\tau_{h-1}}^{i,j} \geq P_{\tau_{h-1}}^*$. But it is very heavy to compute the expected continuation value for all previous time and so to determine the critical price $P_{\tau_k}^*, k = 1, \dots, h-2$, as it is shown in Carr (1995).

The main contribution of the LSM method is to determine the expected continuation values by regressing the subsequent discounted cash flows on a set of basis functions of current state variables. As described in Abramowitz and Stegun (1970), a common choice of basis functions are the weighted Power, Laguerre, Hermite, Legendre, Chebyshev, Gegenbauer and Jacobi polynomials. In our paper we consider as basis function a three weighted Power polynomial. Let be L^w the basis of functional forms of the state variable $\hat{P}_{\tau_k}^{i,j}$ that we use as regressors. We assume that $w = 1, 2, 3$. At time τ_{h-1} , the least square regression is equivalent to solve the following problem:

$$\min_{\mathbf{a}} \sum_{j=1}^m \left[S_0(\hat{P}_T^{i,j}, 1, 0) e^{-r\Delta t} - \sum_{w=1}^3 a_w L^w(\hat{P}_{\tau_{h-1}}^{i,j}) \right]^2 \quad (17)$$

The optimal $\hat{\mathbf{a}} = (\hat{a}_1, \hat{a}_2, \hat{a}_3)$ is then used to estimate the expected continuation value along each path $\hat{P}_{\tau_{h-1}}^{i,j}, j = 1, \dots, m$:

$$\hat{S}_1^j(\hat{P}_{\tau_{h-1}}^{i,j}, 1, \Delta t) = \sum_{w=1}^3 \hat{a}_w L^w(\hat{P}_{\tau_{h-1}}^{i,j})$$

After that, the optimal decision for each price path is to choose the maximum between the immediate exercise and the expected continuation value.

Proceeding recursively until time t_1 , we have a final vector of continuation values for each price-path $\hat{P}_{\tau_k}^{i,j}$ that allows us to build a stopping rule matrix in Matlab that

maximizes the value of American option. As consequence, the i th option value approximation $\hat{S}_k^i(\hat{P}_{t_1}^i, 1, T - t_1)$ can be determined by averaging all discounted cash flows generated by option at each date over all paths $j = 1, \dots, m$. Finally, it is possible to implement Monte Carlo simulation to approximate the CAEO:

$$C(S_k, IT, t_1) \approx D_0 e^{-\delta_d t_1} \left(\sum_{i=1}^n \frac{\max(\hat{S}_k^i(\hat{P}_{t_1}^i, 1, T - t_1) - q, 0)}{n} \right) \quad (18)$$

“Appendix A” illustrates the complete Matlab algorithm to value CAEO. We conclude that, applying real option methodology, the R&D project will be realized at time t_0 if $C(S_k, IT, t_1) - R$ is positive, otherwise the investment will be rejected.

4 Feed-Forward Neural Networks to Value CAEO

In this section we describe the neural network architecture in order to value the CAEO and in particular a BP system in which the input layer is composed by $n = 10$ nodes, one for each variable:

Fig. 2 Structure of the general feed-forward single hidden layer perceptron

$$CAEO = f(V, D, IT, \delta_d, \delta_v, t_1, T, \rho, \sigma_v, \sigma_d)$$

and is composed by one hidden layer with $p = 6$ nodes, as it shown in Fig. 2. Following Eskiizmirli et al. (2020), we describe the BP neural network. The yellow circles are the ten input parameters above described, the blue ones are the six nodes in the hidden layer and the pink node denotes the CAEO output given by BP structure. Moreover, the red and green lines denote a negative inhibitory and a positive excitatory respectively, depending on the weights connecting the nodes. Obviously, the thickness represents the intensity of link.

For the learning phase, the network is parameterized on a sufficient large number of targets given by the previous Monte Carlo LSM estimation of CAEO, summarized in Tables 4 and 5. The idea is to use the Monte Carlo values, whose time simulation is very long, as targets in the training phase in order to extend, with the BP neural network, the value for any input vector. This approach allows a drastical reduction of time simulation. About the Monte Carlo approach, we have used as number of discretizations $x = 100$, the number of American simulation $m = 50,000$ and the paths of Compound option $n = 30,000$. We recall that for each path $j = 1, \dots, n$ there are $m = 50,000$ trajectories to simulate the American exchange option. This leads to increase time simulation of CAEO instead of a better accuracy.

We propose a logistic activation function between the input-hidden layers. In this scenario, its property as approximator is well defined (see White 1990). In our model, we have assumed one hidden layer with six nodes and one output layer, i.e. the neural value of CAEO, with a pureline activation function. Each node performs computation and transformation operations. In particular, in the hidden layer, the aggregation function used is the sum function as:

$$a_j = \sum_{i=1}^{10} w_{ij}x_i - b_j \quad (19)$$

where $j = 1, \dots, 6$ are the nodes in hidden layer, x_i are the input values for $i = 1, \dots, 10$ and b_j is a threshold value named bias, (for more detail see Hecht-Nielsen 1990). As seen in Fig. 2, a feed forward neural network model including a single hidden layer, which takes inputs from the input layer and produces the weighted sum of inputs added onto some bias values as outputs, is preferred to solve the problem effectively. The output produced by each node of hidden layer is obtained by the logistic activation function:

$$x'_j = f(a_j) = \frac{1}{1 + e^{-a_j}}$$

and so this output become input for the output layer. In the same fashion:

$$y' = g\left(\sum_{j=1}^6 w'_j x'_j - b'\right)$$

is the output that the network produces at the end of the first cycle of learning, in which g is the pureline activation function, w'_j and b'_j are the weights and the bias, respectively.

Moreover, the BP networks are a learning algorithm based on conventional method of reduction gradient, in which the couples of input-output are introduced iteratively in the network by an opportune update and modification of weights in order to reach the minimum value of squared error (MSE) function:

$$E = \frac{1}{2K} \sum_{k=1}^K [y'_k - y_k]^2 \quad (20)$$

where K is the number of input-output data with LSM simulation, y_k is the real output (target) associated with vector input k and y'_k is the neural value.

For the numerical solution of the minimization problem defined above, the "Gradient Descent Method" is considered. Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function. To find a local minimum of a function using a gradient descent, one takes steps proportional to the negative of the gradient (or approximate gradient) of the function at the current point. In particular, the updating of the weights is obtained by retropropagating appropriately the value assumed by the function E from the output layer, through the hidden layer, up to the initial state, as:

$$w_{ij}(q+1) = w_{ij}(q) - \eta \frac{\partial E}{\partial w_{ij}}(q+1) \quad (21)$$

$$w'_j(q+1) = w'_j(q) - \eta \frac{\partial E}{\partial w'_j}(q+1) \quad (22)$$

where $w(q)$ is the value of weights at q iteration and η is a learning rate that we assume $\eta = 0.60$.

Moreover, the choice of the learning rate is important, as descent parameter η , and it plays a vital role for converging of the algorithm to the solution in gradient descent. A lower η value causes a long running time for the algorithm, which becomes computationally expensive. In contrast, large η values imply divergence from the solution in general.

4.1 Other Methods: Radial Basis Function (RBF) and General Regression Neural Network (GRNN)

As we have analyzed, BP network is one of the most widely used neural networks. It is a multi-layer network which includes at least one hidden layer. First the input is propagated forward through the network to get the response of the output layer. Then, the sensitivities are propagated backward to reduce the error. During this

process, weights in all hidden layers are modified. As the propagation continues, the weights are continuously adjusted and the precision of the output is improved.

Radial Basis Function (RBF) networks typically have three layers: an input layer, a hidden layer with a non-linear RBF activation function and a linear output layer.

The RBF network is a three-layer feed-forward neural network, between the input and the output layers there is a hidden layer. When training, vectors are input to the first layer and fanned out to the hidden layer. In the latter, a cluster of RBF functions turn the input to output, adjusting the weight of the input to the hidden layer. Then, under the target vector's supervising, the weight of the output vector of the hidden layer is adjusted. When clustering texts, the Euclidean distance between the input vectors and the weight vectors, which have been adjusted by training process, is calculated. Each input sample is sorted to a class. Then the output layer collects samples belonging to same classes and organizes an output vector, the final clustering. The most common form of basis function is the Gaussian. By contrast, the hidden units of the RBF network are formed by the distance between the prototype vector and the input vector, transformed by a non-linear basis function. The basic structure of an RBF neural network includes an n dimension input layer, a fairly larger dimension hidden layer ($p > n$) and the output layer. Typical radial basis function is Gaussian function,

$$x'_j = \exp\left(\frac{-\|a_j - c_j\|}{2\sigma_j^2}\right)$$

where p is the number of number of neurons in the hidden layer, $\|\cdot\|$ is the Euclidean norm, c_j and σ_j are the center and width of the hidden neuron j , respectively and a_j is given by formula (19). The output is given by the following linear transformation:

$$y'_{RD} = \sum_{j=1}^p w'_j x'_j$$

Generalized Regression Neural Network (GRNN), suggested by Specht (1991), belongs to RBF networks with the assumption that the number of neurons in the hidden layer is the same as the sample size of training data and the center of the i -th neuron is just the i -th sample x_i . We remark that GRNN directly produces a predicted value without training process.

5 Numerical Results

Figure 3a shows the $K = 100$ LSM data used in training and in the particular their values have been normalized between 0 and 1. The lowest value corresponds to CAEO value equals to 1.850 while the highest one is 32.250. In the same picture, the red line depicts the average value of LSM simulation. It is interesting to analyze Fig. 3b that illustrates the evolution of standard error in the training phase over the

Fig. 3 Back Propagation training results

learning cycles. It shows how the average training error goes down in order to reach the level 0.0034.

To illustrate our results, we simulate by the neural network the CAEO value starting from these initial parameter value: $V = 100$; $D = 80$; $IT = 40$; $\delta_v = 0.15$; $\delta_d = 0$; $t_1 = 1$; $T = 2$; $\rho = 0.06$; $\sigma_v = 0.90$; $\sigma_d = 0.30$ and we change (once for time) all variables. The results are summarized in Table 1. The neural network simulates the CAEO respecting the sensitivity of several variables. In particular, it increases when asset V and the volatilities σ_v and σ_d raise too. While, the CAEO decreases when the investment costs D and IT enlarge. The advantage to have a neural network to simulate a CAEO is, first of all, the time in order to receive the

Table 1 Neural Network values with the BP method

V	CAEO	D	CAEO	IT	CAEO	δ_v	CAEO	t_1	CAEO	T	CAEO
75	12.33	60	23.96	20	25.12	0.10	23.95	1	22.17	2	22.17
80	14.48	65	23.55	25	24.49	0.13	22.93	1.1	22.27	2.5	23.13
85	16.63	70	23.11	30	23.80	0.15	22.17	1.2	22.37	3	23.98
90	18.68	75	22.66	35	23.03	0.17	21.35	1.3	22.47	3.5	24.74
95	20.54	80	22.17	40	22.17	0.19	20.46	1.4	22.56	4	25.41
100	22.17	85	21.66	45	21.23	0.21	19.51	1.5	22.66	4.5	25.99
V	CAEO	D	CAEO	IT	CAEO	ρ	CAEO	σ_v	CAEO	σ_d	CAEO
105	23.57	90	21.13	50	20.21	- 0.20	23.22	0.40	6.09	0.10	17.66
110	24.75	95	20.57	55	19.12	- 0.15	23.04	0.50	8.65	0.20	20.09
115	25.72	100	19.98	60	17.95	- 0.10	22.84	0.60	11.99	0.30	22.17
120	26.52	105	19.37	65	16.73	- 0.05	22.64	0.70	15.69	0.40	23.89
125	27.17	110	18.74	70	15.47	0	22.43	0.80	19.22	0.50	25.26
130	27.71	115	18.09	75	14.19	0.05	22.22	0.90	22.17	0.60	26.34

Table 2 Sensitivity parameter on CAEO value

Parameter		Sensitivity
Volatility	σ_v	0.6357
Project value	V	0.5643
Volatility	σ_d	0.5275
Information Tecnology	IT	0.3984
Dividend yield	δ_v	0.3336
Development cost	D	0.2544
Maturity time	T	0.2341
Dividend yield	δ_d	0.2044
Time	t_1	0.1385
Correlation	ρ	0.1115

simulated output with a low standard error. We remark that the LSM Monte Carlo is accuracy with an average standard error 0.0094 but the time simulation is very long. Another advantage about the neural network is to describe the influence that all variables produce on the CAEO value. As referred in Table 2, the most important parameter is the volatility of gross project σ_v , the gross project value V and the volatility σ_d . It is also possible to verify, as shown in Fig. 4, that most of the results are accurate. They are in fact almost all arranged on the fit line. In fact, the correlation coefcient (R) to the linear t ($y = ax$) is 0.999 giving an almost perfect t, something of course expected since it was this data set used for the training of the network. The very good tting values indicate that the training was done very well.

Fig. 4 Regression based on BP Neural Network: training, validation, test and overall result

Table 3 Comparison among BP, RBF and GRNN methodology when the asset value V changes

V	BP CAEO	RBF CAEO	GRNN CAEO
75	12.33	12.14	12.01
80	14.48	14.33	14.21
85	16.63	16.32	16.35
90	18.68	18.23	18.45
95	20.54	20.11	20.52
100	22.17	21.96	21.93
MSE	0.0034	0.0056	0.0060
MAPE	0.0053	0.0077	0.0082

Table 4 Monte Carlo simulation of CAEO

	V	D	IT	δ_V	δ_D	t_1	T	ρ	σ_V	σ_D	CAEO
1	100	80	40	0.15	0	1	2	0.06	0.9	0.3	21.97
2	100	70	40	0.15	0	1	2	0.06	0.9	0.3	23.91
3	100	60	40	0.15	0	1	2	0.06	0.9	0.3	26.30
4	100	90	40	0.15	0	1	2	0.06	0.9	0.3	20.13
5	100	100	40	0.15	0	1	2	0.06	0.9	0.3	18.46
6	100	110	40	0.15	0	1	2	0.06	0.9	0.3	16.75
7	100	120	40	0.15	0	1	2	0.06	0.9	0.3	15.65
8	100	130	40	0.15	0	1	2	0.06	0.9	0.3	14.75
9	70	100	40	0.15	0	1	2	0.06	0.9	0.3	7.92
10	80	100	40	0.15	0	1	2	0.06	0.9	0.3	10.59
11	90	100	40	0.15	0	1	2	0.06	0.9	0.3	14.31
12	110	100	40	0.15	0	1	2	0.06	0.9	0.3	22.71
13	120	100	40	0.15	0	1	2	0.06	0.9	0.3	27.42
14	130	100	40	0.15	0	1	2	0.06	0.9	0.3	32.15
15	100	100	10	0.15	0	1	2	0.06	0.9	0.3	26.71
16	100	100	20	0.15	0	1	2	0.06	0.9	0.3	23.25
17	100	100	30	0.15	0	1	2	0.06	0.9	0.3	20.39
18	100	100	50	0.15	0	1	2	0.06	0.9	0.3	16.25
19	100	100	60	0.15	0	1	2	0.06	0.9	0.3	14.95
20	100	100	70	0.15	0	1	2	0.06	0.9	0.3	13.72
21	100	100	80	0.15	0	1	2	0.06	0.9	0.3	12.42
22	100	100	40	0.05	0	1	2	0.06	0.9	0.3	24.85
23	100	100	40	0.1	0	1	2	0.06	0.9	0.3	21.26
24	100	100	40	0.2	0	1	2	0.06	0.9	0.3	15.55
25	100	100	40	0.25	0	1	2	0.06	0.9	0.3	13.87
26	100	100	40	0.30	0	1	2	0.06	0.9	0.3	12.44
27	100	100	40	0.15	0.05	1	2	0.06	0.9	0.3	20.81
28	100	100	40	0.15	0.10	1	2	0.06	0.9	0.3	22.66
29	100	100	40	0.15	0.15	1	2	0.06	0.9	0.3	22.21
30	100	100	40	0.15	0.20	1	2	0.06	0.9	0.3	23.63
31	100	100	40	0.15	0.25	1	2	0.06	0.9	0.3	25.15
32	100	100	40	0.15	0	1	2.5	0.06	0.9	0.3	18.72
33	100	100	40	0.15	0	1	2.75	0.06	0.9	0.3	19.32
34	100	100	40	0.15	0	1	3	0.06	0.9	0.3	19.52
35	100	100	40	0.15	0	1	3.25	0.06	0.9	0.3	19.88
36	100	100	40	0.15	0	1	3.5	0.06	0.9	0.3	19.71
37	100	100	40	0.15	0	1	3.75	0.06	0.9	0.3	20.21
38	100	100	40	0.15	0	1	4	0.06	0.9	0.3	20.28
39	100	100	40	0.15	0	1	4.25	0.06	0.9	0.3	20.54
40	100	100	40	0.15	0	1	4.5	0.06	0.9	0.3	20.58
41	100	100	40	0.15	0	1	4.75	0.06	0.9	0.3	20.81
42	100	100	40	0.15	0	1	5	0.06	0.9	0.3	20.53

Table 4 continued

	V	D	IT	δ_V	δ_D	t_1	T	ρ	σ_V	σ_D	CAEO
43	100	100	40	0.15	0	1.25	5	0.06	0.9	0.3	21.89
44	100	100	40	0.15	0	1.5	5	0.06	0.9	0.3	22.44
45	100	100	40	0.15	0	1.75	5	0.06	0.9	0.3	23.25
46	100	100	40	0.15	0	2	5	0.06	0.9	0.3	23.73
47	100	100	40	0.15	0	2.25	5	0.06	0.9	0.3	23.98
48	100	100	40	0.15	0	2.5	5	0.06	0.9	0.3	24.33
49	100	100	40	0.15	0	2.75	5	0.06	0.9	0.3	24.29
50	100	100	40	0.15	0	3	5	0.06	0.9	0.3	24.33

Table 5 Monte Carlo simulation of CAEO

	V	D	IT	δ_V	δ_D	t_1	T	ρ	σ_V	σ_D	CAEO
51	100	100	40	0.15	0	3.25	5	0.06	0.9	0.3	24.31
52	100	100	40	0.15	0	3.5	5	0.06	0.9	0.3	24.22
53	100	100	40	0.15	0	3.75	5	0.06	0.9	0.3	24.31
54	100	100	40	0.15	0	4	5	0.06	0.9	0.3	24.23
55	100	100	40	0.15	0	4.25	5	0.06	0.9	0.3	24.15
56	100	100	40	0.15	0	4.5	5	0.06	0.9	0.3	24.13
57	100	100	40	0.15	0	1	2	0.06	0.85	0.3	16.37
58	100	100	40	0.15	0	1	2	0.06	0.80	0.3	14.62
59	100	100	40	0.15	0	1	2	0.06	0.75	0.3	13.32
60	100	100	40	0.15	0	1	2	0.06	0.70	0.3	11.87
61	100	100	40	0.15	0	1	2	0.06	0.65	0.3	10.29
62	100	100	40	0.15	0	1	2	0.06	0.60	0.3	8.86
63	100	100	40	0.15	0	1	2	0.06	0.55	0.3	7.53
64	100	100	40	0.15	0	1	2	0.06	0.50	0.3	6.36
65	100	100	40	0.15	0	1	2	0.06	0.45	0.3	5.24
66	100	100	40	0.15	0	1	2	0.06	0.40	0.3	4.33
67	100	100	40	0.15	0	1	2	0.06	0.35	0.3	3.67
68	100	100	40	0.15	0	1	2	0.06	0.30	0.3	3.05
69	100	100	40	0.15	0	1	2	0.06	0.25	0.3	2.59
70	100	100	40	0.15	0	1	2	0.06	0.20	0.3	2.21
71	100	100	40	0.15	0	1	2	0.06	0.15	0.3	1.98
72	100	100	40	0.15	0	1	2	0.06	0.10	0.3	1.84
73	100	100	40	0.15	0	1	2	0.06	0.90	0.05	16.78
74	100	100	40	0.15	0	1	2	0.06	0.90	0.10	17.04
75	100	100	40	0.15	0	1	2	0.06	0.90	0.15	17.25
76	100	100	40	0.15	0	1	2	0.06	0.90	0.20	17.47
77	100	100	40	0.15	0	1	2	0.06	0.90	0.25	17.79
78	100	100	40	0.15	0	1	2	0.06	0.90	0.30	18.13

Table 5 continued

	V	D	IT	δ_V	δ_D	t_1	T	ρ	σ_V	σ_D	CAEO
79	100	100	40	0.15	0	1	2	0.06	0.90	0.35	18.61
80	100	100	40	0.15	0	1	2	0.06	0.90	0.40	19.11
81	100	100	40	0.15	0	1	2	0.06	0.90	0.45	19.99
82	100	100	40	0.15	0	1	2	0.06	0.90	0.50	20.74
83	100	100	40	0.15	0	1	2	0.06	0.90	0.55	21.62
84	100	100	40	0.15	0	1	2	0.06	0.90	0.60	22.57
85	100	100	40	0.15	0	1	2	0.06	0.90	0.65	23.43
86	100	100	40	0.15	0	1	2	0.06	0.90	0.70	24.54
87	100	100	40	0.15	0	1	2	0.06	0.90	0.75	25.52
88	100	100	40	0.15	0	1	2	0.06	0.90	0.80	26.75
89	100	100	40	0.15	0	1	2	0.06	0.90	0.85	27.81
90	100	100	40	0.15	0	1	2	0.06	0.90	0.90	29.12
91	100	100	40	0.15	0	1	2	-0.50	0.90	0.30	23.53
92	100	100	40	0.15	0	1	2	-0.45	0.90	0.30	23.23
93	100	100	40	0.15	0	1	2	-0.40	0.90	0.30	22.71
94	100	100	40	0.15	0	1	2	-0.35	0.90	0.30	22.29
95	100	100	40	0.15	0	1	2	-0.30	0.90	0.30	21.73
96	100	100	40	0.15	0	1	2	-0.25	0.90	0.30	21.25
97	100	100	40	0.15	0	1	2	-0.20	0.90	0.30	20.82
98	100	100	40	0.15	0	1	2	-0.15	0.90	0.30	20.31
99	100	100	40	0.15	0	1	2	-0.10	0.90	0.30	19.85
100	100	100	40	0.15	0	1	2	-0.5	0.90	0.30	19.34

These results varied the success of BP neural networks to recognize the implicit relationships between input and output variables.

Finally, to appreciate the goodness of BP method, we will compare it with the RBF and general regression GRNN. Some significant results are summarized in Table 3. To evaluate the goodness of the network, the MSE (see Eq. 20) and the Mean Absolute Percentage Error (MAPE) defined as:

$$MAPE = \frac{1}{K} \sum_{k=1}^K \left| \frac{y'_k - y_k}{y_k} \right|$$

are proposed. As we can see from the results in Table 3, the RBF and the GRNN seem to underestimate the CAEO value with respect to BP. However, the three methods analyzed have a good predictive power since, even if the MSE and MAPE are slightly higher in RBF and GRNN than BP. It is evident that the BP network provides much better predictions than the other types of neural networks. As regards the latter, however, it is difficult to establish which one has the best behavior since the accuracy of the predictions is fairly uniform. However, what we can say is that the GRNN type network is the one that behaves in the worst way.

6 Conclusions

In this paper, we have shown how neural network methodology, joint with the LSM, can be used to evaluate R&D projects. In particular way, an R&D opportunity is a sequential investment and therefore can be considered as a compound option. We have assumed the managerial flexibility to realize the production investment D before the maturity T in order to benefit of R&D cash flows. So an R&D project can be view as a Compound American Exchange option (CAEO), that allows us to couple both the sequential frame and the managerial flexibility of an R&D investment. We have analyzed two main contributions. The first is that the LSM method permits to determine the expected continuation value by regressing the discounted cash flows on the simple powers of variable P , and so to overcome the effort to compute the critical prices $P_{t_k}^*$, $k = 1 \dots h - 2$. But this approach requires long time in order to value a CAEO. The second contribution analyzed is the construction of a neural network based on BP architecture using the LSM simulation results as “targets” in the learning phase. As there is not a market evaluation of CAEO, we have seen that the advantage of this approach is the speed and the accuracy in the computations and, moreover, the possibility to extend the trained neural network to value any R&D investment project. Finally, we have compared the BP results with those obtained from the RBF and GRNN neural approach. Based on MSE and MAPE, the BP provides much better predictions.

Appendix: Matlab algorithm

In this appendix we present first of all the Matlab algorithm of LSM method:

```

function LMSCAEO= LMSCAEO(V,D,q,sigv,sigd,rho,...
dV,dD,t1,T,x,m,n);
sig=sqrt(sigv^2+sigd^2-2*sigv*sigd*rho);
P0=V/D;
d=dV-dD;
dt =1/x; %Lenght of the interval of time
t=T-t1;
h=(t/dt); %Number of periods to simulate the price
for i=1:n
Pt1=P0*exp(norminv(rand,-d*t1-sig^2*t1/2,sig*sqrt(t1)));
dBt=sqrt(dt)*randn(m,h); %Brownian motion
Pt=zeros(m,h); %Initialize matrix
Pt(:,1)=Pt1*ones(m,1); %Vector of initial stock price
for k=2:h;
Pt(:,k)=Pt(:,k-1).*exp((-d-0.5*sig^2)*dt+sig*dBt(:,k));
end
PPit=Pt; %Change the name
Work Backwards; Initialize CashFlow Matrix
S=NaN*ones(m,h);
S(:,h)=max(PPit(:,h)-1,0);
for j=h:-1:3;
    Step 1: Select the path in the money at time j-1
    I=find(PPit(:,j-1)-1>0);
    ISize=length(I);
    Step 2: Project CashFlow at time j onto basis...
    function at time j-1
    if j==h;
    YY=(ones(ISize,1)*exp(-dD*[1:h-j+1]*dt)).*S(I,j:h);
    else
    YY=sum(((ones(ISize,1)*exp(-dD*[1:h-j+1]*dt)).*S(I,j:h)))';
    end
    PPb=PPit(I,j-1);
    XX=[ones(ISize,1),PPb,PPb.^2,PPb.^3];
    BB=pinv(XX'*XX)*XX'*YY;
    PPb2=PPit(:,j-1);
    XX2=[ones(m,1),PPb2,PPb2.^2,PPb2.^3];
    Find when the option is exercised:
    IStop=find(PPit(:,j-1)-1>=max(XX2*BB,0));
    Find when the option is not exercised:
    ICon=setdiff([1:m],IStop);
    Replace the payoff function with the option value:
    S(IStop,j-1)=PPit(IStop,j-1)-1;
    S(IStop,j:h)=zeros(length(IStop),h-j+1);
    S(ICon,j-1)=zeros(length(ICon),1);
    end
    YY=sum(((ones(m,1)*exp(-dD*[1:h-1]*dt)).*S(:,2:h)))';
    AEOSim(i)=mean(YY);
    PAYOFF(i)=max(AEOSim(i)-q,0);
end
CAEO=D*exp(-dD*t1)*mean(PAYOFF)

```

Acknowledgements Many thanks to the anonymous referees for their contribution to the improvement of this manuscript. This paper is dedicated to my friend Mimmo d'Angelo and his desire for research.

Funding Open access funding provided by Università degli Studi di Bari Aldo Moro within the CRUI-CARE Agreement.

Compliance with Ethical Standards

Conflict of interest The author declares that he has no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Abramowitz, M., & Stegun, I. A. (1970). *Handbook of Mathematical Functions*. New York: Dover Publications.
- Aminian, F., Suarez, E.D., Aminian, M. & Walz, D.T. (2006) .*Forecasting Economic Data with Neural Networks*, Computational Economics, Vol. 28, 71–88
- Andreou, P. C., Charalambous, C., & Martzoukos, S. H. (2006). Robust Artificial Neural Networks for Pricing of European Options. *Computational Economics*, 27, 329–351.
- Arin, E., & Ozbayoglu, A. M. (2020). Deep Learning Based Hybrid Computational Intelligence Models for Options Pricing. *Computational Economics*.
- Armada, M.R., Kryzanowsky, L. & Pereira, P.J. (2007). *A Modified Finite-Lived American Exchange Option Methodology Applied to Real Options Valuation.*, Global Finance Journal, Vol. 17, Issue 3, 419–438
- Barraquand, J., & Martineau, D. (1995). Numerical Valuation of High Dimensional Multivariate American Securities. *J. Financial Quant. Anal.*, 30, 383–405.
- Broadie, M., & Glasserman, P. (1997). Pricing American-style securities using simulation. *Journal of Economic Dynamics and Control*, 21, 1323–1352.
- Carr, P. (1988). The valuation of sequential exchange opportunities. *The Journal of Finance*, 43(5), 1235–1256.
- Carr, P. (1995). The valuation of American exchange options with application to real options. In Lenos Trigeorgis & Westport Connecticut (Eds.), *Real options in capital investment: Models, strategies and applications*. London: Praeger.
- Cortelezzi, F., & Villani, G. (2009). Valuation of R&D sequential exchange options using Monte Carlo approach. *Computational Economics*, 33(3), 209–236.
- Eskiizmirli, S., Günel, K., & Polat, R. (2020) .*On the solution of the Black–Scholes equation using feed-forward neural networks*. Computational Economics .
- Golbabai, A., Nikan, O., & Nikazad, T. (2019). Numerical analysis of time fractional Black–Scholes European option pricing model arising in financial market. *Computational and Applied Mathematics*, 38(173)
- Hecht-Nielsen, Neurocomputing. (1990). 433 pagg. University of Michigan: Addison-Wesley Publishing Company.
- Longstaff, F. A., & Schwartz, E. S. (2001). Valuing American options by simulation: A simple least-squares approach. *The Review of Financial Studies*, 14(1), 113–147.

- Margrabe, W. (1978). The value of an exchange option to exchange one asset for another. *The Journal of Finance*, 33(1), 177–186.
- Ma, M. (2016). Exploration decision-making on energy based on improved real option model and BP neural. *Networks International Journal of Control and Automation*, 9(10), 393–402.
- Myers, S. C. (1977). Determinants of corporate borrowing. *Journal of Financial Economics*, 5(2), 147–175.
- McDonald, R. L., & Siegel, D. R. (1985). Investment and the valuation of firms when there is an option to shut down. *International Economic Review*, 28(2), 331–349.
- Specht, D. F. D. F. (1991). A general regression neural network. *IEEE Transactions on Neural Networks*, 2(6), 568–576.
- Taudes, A., Natter, M., & Trcka, M. (1998). Real option valuation with neural networks. *International Journal of Intelligent Systems in Accounting, Finance & Management*, 7, 43–52.
- Tilley, J. (1993). Valuing American options in a path simulation model. *Transactions of the Society of Actuaries*, 45, 83–104.
- Villani, G. (2014). *Valuation of R&D investment opportunities using the least-square monte carlo method*. In: Corazza, M., Pizzi, C. (Eds.), *Mathematical and Statistical Methods for Actuarial Sciences and Finance*, pp. 287–299. Springer
- White, H. (1990). Connectionist nonparametric regression: Multilayer feedforward networks can learn arbitrary mappings'. *Neural Networks*, 3(5), 535–549.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.