Int. J. Nonlinear Anal. Appl. In Press, 1–10
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2024.34754.5198

Integrated high frequency RF circuit design using deep reinforcement learning via proximity policy optimization method

Ali Khakshoor Shandiz, Abbas Golmakani*, Amin Noori

Faculty of Electrical Engineering and Medical Engineering, Sajjad University, Mashhad, Iran

(Communicated by Seyed Hossein Siadati)

Abstract

The automatic design of analogue circuits is a challenging task due to the high complexity of the design, which is caused by the search space and sometimes conflicting parameters. In the article, a trial-and-error-based approach that combines reinforcement learning and deep neural networks is used to determine the values of circuit elements have been used. In methods based on reinforcement learning, the agent tries to act like an expert designer and maybe better than that by trial and error and using the information it gets from the environment. In this article, one of the latest methods of deep reinforcement learning called approximate policy optimization (PPO) is used. To show the efficiency of the above method, a cascaded LNA circuit is considered. The voltages are determined by the learning agent to optimize the circuit design requirements such as gain, noise figure and power consumption. To train the learning agent in the reward function, two categories of adverbs have been included in such a way that the main goal is to optimize the gain and noise figure, and the secondary goal is to focus on other requirements, such as power consumption. The environment, which is the amplifier circuit, is simulated in the Hspice software in 0.18 micrometre technology from the TSMC company at the frequency of 5.7 GHz and the learning agent is also defined in the MATLAB environment, which has been able to design the values of the circuit elements by interacting with the environment.

Keywords: integrated circuit design, RF, deep reinforcement learning, PPO, LNA
2020 MSC: 68T05, 94Cxx

1 Introduction

The design of analogue circuits is directly dependent on technology and functional requirements. The shrinking of the dimensions of the component reduces the supply voltage of the integrated circuits, which makes it possible to build digital and analogue circuits on one chip, but on the other hand, it causes many design issues that were not important in the design before. Such trends require that the analysis and design of circuits be accompanied by a deep understanding of the limitations imposed by the new technology [15].

Designers of these circuits in conventional (traditional) methods have a limited number of explicit patterns or in other words definitive rules to follow in their designs, so day by day the need for tools such as mechanized design of

*Corresponding author
Email addresses: ali.khakhsoor@sadjad.ac.ir (Ali Khakshoor Shandiz), golmakani@sadjad.ac.ir (Abbas Golmakani), amin.noori@sadjad.ac.ir (Amin Noori)

Received: June 2024 *Accepted:* August 2024

electronic circuits is felt more and more. Therefore, Machine Learning tools have been able to play a significant role in this field and cause ease and significant reduction of design time. Reinforcement Learning is one of the subsets of machine learning and has been the focus of researchers in this field in recent years.

On the other hand, the advances made in recent years in the semiconductor industry have led to the widespread use of integrated circuits, which has increased the design challenges of these circuits even more. The design of analogue integrated circuits has challenges such as inherent defects of the transistor, reduction of supply voltage, power consumption, circuit complications and also design corners (Process-Voltage-Temperature) [15]. Even though research on the design and implementation of integrated circuits High frequency has been done for decades, but it is still challenging [16]. In addition to the general challenges mentioned for analogue integrated circuits, the design of high-frequency integrated circuits alone also has many compromises between parameters that the designer must consider these requirements, as well as new challenges every day due to the demand for designs with High performance, low cost and more efficiency are created [16]. This problem has made design using developed artificial intelligence tools more needed than in the past.

1.1 Research background

In reference [1], the authors have used artificial neural networks to optimize the design of the two-stage operational amplifier circuit. The structure of the network used was a forward neural network with a hidden layer, which was used to train this network from a previously collected data set. In reference [14], artificial neural networks have been used in the automatic design of a two-stage operational amplifier in CMOS (Complementary Metal Oxide Semiconductor). In this article, the sizing of the circuit elements, including the dimensions of the transistors and the capacitor, as well as the bias current, have been considered. In reference [27], the authors have discussed the automatic sizing of analogue circuits. The method used by the authors is based on reinforcement learning, which has a neural network model that plays the role of supervisor in learning. In [10], Li et al. have discussed the automatic sizing of analogue integrated circuits and presented a method based on an artificial neural network with a genetic algorithm searcher to design and optimize these circuits. In reference [12], the authors have used reinforcement learning (RL) in Chip Placement, which is a time-consuming task, and have tried to minimize power, performance and area by the proposed method. This article claims to have accelerated the design time of several weeks which was required by experts and they were able to complete this design process within 6 hours. In reference [26], RL has been used along with other methods to predict and optimize clock tree synthesis and they state that their proposed method has brought significant improvement in various cases including error. The topic discussed in reference [8] is logical synthesis. To reduce the dependence on human intervention to optimize this field, the authors have benefited from reinforcement learning, the proposed method of the article is to train the RL agent with the Actor-Critic method, and they have announced that their proposed method improves the quality of the results by about 13% in the benchmarks. has been investigated. Reference [13] has presented a method that encompasses the optimization of segment placement for TensorFlow computational graphs. The presented method is based on reinforcement learning that the trained model has learned to optimize the design time and surpass the experts in this field. Reference [11] has investigated the use of automatic methods of choosing the most appropriate branching rule in each of the nodes of the search tree, which has benefited from reinforcement learning. In this article, he discusses the maximization of the learning rate in the topic of Satisfiability (SAT) solvers with the help of RL. Reference [6] discusses the logical optimization algorithm. In this regard, the authors using Deep Reinforcement Learning (DRL) have created a mechanism for logical optimizations that does not require the intervention of designers in the design process.

In reference [2], the authors have discussed the optimization of the placement of parameters in VLSI (Very Large Scale Integration). Because this process is time-consuming due to the vastness of the design space, and also the designers of this field have a lot of dependence on their artificial and meta-innovative methods. Therefore, by presenting a self-learning method based on deep reinforcement learning, the authors reduced the design time and did not rely on the human designer, and also improved the parameters of power, performance and area, and significantly reduced the length of the wire.

Mr. Lederman and colleagues in reference [4] have studied an efficient heuristic system for automatic reasoning algorithms for satisfiability problems (SAT), which is a learning system based on deep reinforcement learning. The authors in this article believe that they have greatly reduced the overall execution time with the help of their proposed method. In reference [24], the Deep Deterministic Policy Gradient (DDPG) method has been used to design integrated circuits and it has been stated that the proposed method has provided better performance than the design of experts. In reference [23], the authors have been able to perform the dimensional design of integrated circuit transistors well by combining the methods of neural networks and reinforcement learning. In reference [19], they presented a method

based on deep reinforcement learning, which was able to reach convergence in the design of the integrated circuit at a high speed.

As it has been said before, analogue circuit design has many complexities and it means that the accuracy of the design is of great importance, due to which the values of the elements in these designs are generally continuous, but in many machine learning methods, the values They are discrete, which reduces accuracy. In the proposed method of this work, which is based on deep reinforcement learning, special attention has been paid to the continuity of values to increase design accuracy. In addition to this, another innovation that has been done in this work is a newer method of deep reinforcement learning called Proximity Policy Optimization (PPO), which has higher convergence and speed than previous methods.

2 Basic concepts

In the following, the basic concepts, problem design and proposed solution will be discussed.

2.1 Basic concepts

In this section, an overview of the basic and basic concepts of the problem such as reinforcement learning, deep reinforcement learning and its types is discussed.

2.1.1 Reinforcement learning

In fact, RL is a type of learning that is the result of the interaction of the learning agent with the surrounding environment in order to reach the goal over time, without telling the agent how to act in the chosen environment. And only the learning agent receives a signal of punishment or reward for his action. In addition to the received numerical signal may be the immediate effect of the performed action, the future received signals may also be affected by the performed action of the verb [21].

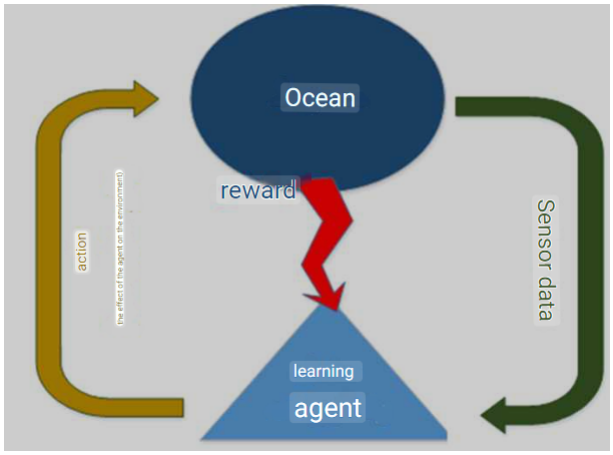

Figure 1: Overview of reinforcement learning

One of the prominent features of RL is the lack of need for an environment model in this learning model. Because the agent is directly interacting with the environment. This feature distinguishes this model from other learning models. Because in some cases it is impossible or very expensive to achieve the model. Among all the mentioned models, RL is the closest model to the human learning model and other organisms. In addition to its main elements such as agent and environment, RL has other elements such as policy, reward signal, value function and environment model. Reinforcement learning is a set of state, action and successive rewards [21]. The goal of the reinforcement learning problem is to maximize the value function for all states of the system. In this direction, when we start taking steps from state S under the π policy. We seek to maximize the total reward received under the series of actions taken by the agent. Based on the π policy, the value of a state is expressed as equation (2.1):

$$V_{(s)} = E_\pi \{R_t | S_t = S\} = E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = S \right\}. \tag{2.1}$$

In the above relationship, r is the instant reward, γ is the forgetting rate, and E is the mathematical hope. Deep learning is one of the approaches of artificial intelligence for machine learning, which enables the machine to extract high-level complex concepts from simple and basic concepts and teach them by using a series of concepts [9]. Deep learning has shown favorable results in displaying information hidden in high-dimensional data, which can be used in different applications such as scientific, commercial and political (government) [25].

2.1.2 Deep reinforcement learning

Deep reinforcement learning is actually a combination or combination of deep neural networks (deep learning) and reinforcement learning.

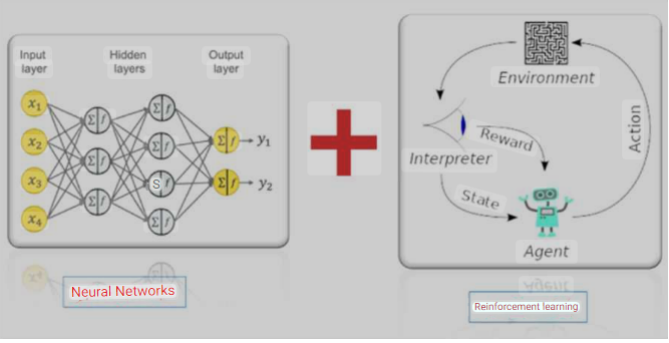

Figure 2: Overview of deep reinforcement learning

Neural networks are used to express the concepts of reinforcement learning. In this method, the state and action space are continuous. Deep neural networks are mostly used to connect the space of actions, to obtain the value of state and action, and to obtain the value of actions. Therefore, the greatest impact of neural networks will be on the agent. In continuous reinforcement learning, states and actions are no longer discrete. This means that the selection of status and action is not limited to the selection from the status and action table. And they can take a certain amount within their acceptable suffering. However, in discrete learning, the number of states and actions is limited, and increasing them will cause the learning to not be done properly and will suffer from a concept called the curse of dimensions. Continuity of state and action space has two main advantages: increasing accuracy and speed to reach the desired goal.

In this regard, various methods such as critic-actor method [20], Deep Q Network (DQN) [7], Double Deep Q Network [21], DDPG [22], Double Delayed Deep Deterministic Policy Gradient [3], Soft Actor-Critic (SAC [5] is presented).

Trust Region Policy Optimization (TRPO): This method is an algorithm of gradient descent policies of reinforcement learning that does not need a model and is online with On-Policy. This method alternates between sampling data obtained as a result of interaction with the environment. Updating policy parameters is done by solving a constrained optimization problem. This method is one of the most robust methods in deterministic and dynamic environments, but it has high computational complexity [17].

Approximate policy optimization (PPO): The PPO algorithm is a simplified version of the TRPO method, which, like it, is based on gradient descent and does not require a model. This algorithm, like TRPO, alternately between sampling the data obtained by interacting with the environment and The optimization of the limited substituted objective function is done using stochastic gradient descent. This substituted objective function improves the stability of training by limiting the size of the policy changes in each step. Unlike the TRPO method, this method has a lower computational cost and also shows more resistance in non-dynamic environments with high uncertainty. The PPO algorithm can have discrete and continuous states, and on the other hand, actions can be both discrete and continuous, which makes the scope of this method wide [18].

2.2 Proposing the problem and proposed solution

Circuit simulation is slow, especially for complex circuits. This makes a random search or exhaustive search impractical. To speed up the design process and reduce time to market, it is very important to identify time-consuming methods and replace them with methods such as process automation. Machine learning can be a promising tool for automating this process. However, the use of supervised learning in this field is faced with the problem of a lack

of training data. Therefore, to solve the problem of learning with the supervisor, the use of reinforcement learning is suggested because this method is learned through interaction, which effectively generates new circuit data and optimizes circuits.

Due to the capabilities and capabilities of deep reinforcement learning, this method can be used in the design of a wide range of analogue integrated circuits, especially

The high-frequency telecommunication complex shows good performance due to the high complexity of these circuits. In the following, a low noise amplifier circuit is examined as an example.

Figure 4 shows the overview of the proposed method, at first, the initial states (basic parameters of the circuit such as gain, etc.) are given as input to deep reinforcement learning networks. The final output of these networks or actions, which are the circuit variables, is given as input to the circuit simulator, which is the Hspice software. After the simulation, the simulation results are read by MATLAB software. From the performed call, parameters such as the next stage of the circuit and reward are calculated. Then the weights of the mentioned networks are updated. The update can be done in different ways depending on the conditions of the problem.

The following is the pseudo-code of the proposed method:

Pseudo code of PPO

- Initialize networks parameters (π_s, b_ω).
- Loop : for $k = 0, 1, 2, 3, \cdots$, do
  - Collect an episode $s_0, a_0, r_1, s_1, \cdots$ under π_s.
  - Calculate reward R_t.
  - Calculate advantage estimate :
  - $A_t = G_t - b_\omega(s_t)$, $G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$.
  - Calculate objective function:
  - $L^{CLIP}(\theta) =$.
  - $E_\pi\left[\min(r(\theta)A_t, clip(r(\theta), 1-t, 1+t)A_t)\right]$.
  - $L_\omega = -\frac{1}{T}\sum_{t=0}^{\gamma-1}(G_t - V_\omega(s_t))^2$.
  - Update networks using Adam method.
- End of Loop.

3 Simulation and results

For example, the design of the Low Noise Amplifier (LNA) circuit with the Cascode structure seen in Figure 3 is addressed in 0.18 μ technology by deep reinforcement learning, while addressing the issue of optimization in important parameters such as maximizing the amount of gain. (S_21), minimizing the input and output mismatch (S_22, S_11), noise figure (NF), power consumption is taken into consideration.

In this design, the input parameters include the number of turns and the diameter of the inductors L_d, L_g, L_s, the number of fingers of the transistors M_1 and M_2, which have the dimensions $W/L = 8\mu$ m/0.18 μ m, the values of the gate bias voltage (V_b) of M_2 and the input bias voltage (V_{bias}). As was said before, to solve problems using reinforcement learning, it is necessary to clarify the key parameters of this method. Therefore, in the following, we will deal with matching the problem of electronic circuit design with reinforcement learning and specifying the parameters of this method. The mentioned environment in reinforcement learning, in this case, is the design space of electronic circuits, which is simulated by H-Spice software. It is possible to consider the reinforcement learning modes as the parameters of the circuit design objectives. Parameters such as S_{21} gain, input and output mismatch S_{11}, S_{22}, noise figure, and chip power consumption, which are important parameters of the circuit, were taken into the attention of the reinforcement learning agent or receiver.

The selectable actions or actions of the agent can be considered parameters (variables) that can be designed in the circuit. Each circuit can have a different number of parameters in the design according to the number of its elements. For example, as mentioned before, in the design of LNA in the Cascode structure, the number of turns and the diameter of the coil of the inductors (L_d, L_g, L_s) and the number of circuit transistors (M_1 and M_2) as well as the

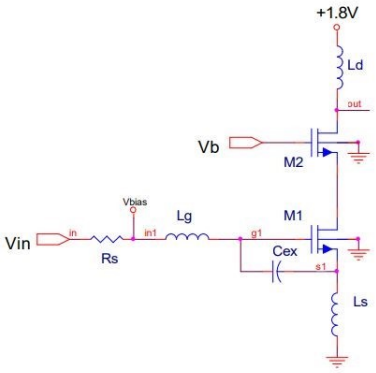

Figure 3: LNA circuit with Cascode structure

gate bias voltage of the transistor M_2 and input bias voltage (V_{bias}) were considered as changeable parameters or the choice of action. One of the most important parameters mentioned in reinforcement learning is the reward signal or punishment of the implemented action of the agent, and the correct selection of this vital element is very effective in the manner and speed of learning as well as in reaching the agent's goal. The choice of reward for each problem should be done in such a way that it helps the agent to reach the goal of the problem. In the mentioned LNA example, the goal is to maximize the gain, and minimize input and output mismatch, signal noise and chip area, the reward signal should It should be chosen in a way that is a function of the mentioned parameters and balance them in the desired direction of the problem. The reward function $R(x)$ is defined by equation (3.1):

$$R(x) = \begin{cases} \alpha * |T_1(x) + T_2(x)| & \text{if high and low priorites satisfied} \\ \beta * T_1(x) - |T_2(x)| & \text{if only high priority satisfied} \\ -|T_1(x)| & \text{other} \end{cases} \tag{3.1}$$

In relation (3.1), the expression $T_1(x)$ is the sum of the adverbs with high priority and the expression $T_2(x)$ is the sum of the adverbs with low priority, in this problem, the high priorities are attributed to S_12 and NF parameters due to their great importance in the design and also α and β are coefficients with fixed values.

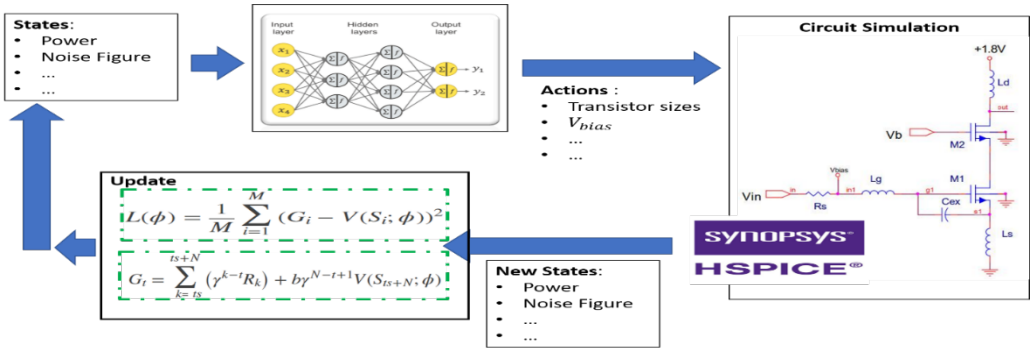

Figure 4: overview of the proposed method

The convergence diagram of the learner's reward function can be seen in Figure 5.

A set of results obtained for circuit element values from the proposed method is shown in Table 1 at 5.7 GHz frequency.

Applying the values of Table 1 to the circuit, the basic parameters of the circuit are as described in Table 2, which are all in the optimal design conditions, which include $S21 > 12db$ and $NF < 3db$ at 5.7 GHz frequency.

The values in Table 2 indicate that the proposed method has provided a wide range of acceptable circuit design values, which, as previously stated, in addition to meeting the design requirements, can satisfy the circuit performance requirements in different applications. For example, for devices that can be implanted in the body, the values of row 7 of table 1 can be used. It has a low power consumption, which naturally has a lower percentage than row 2 of the

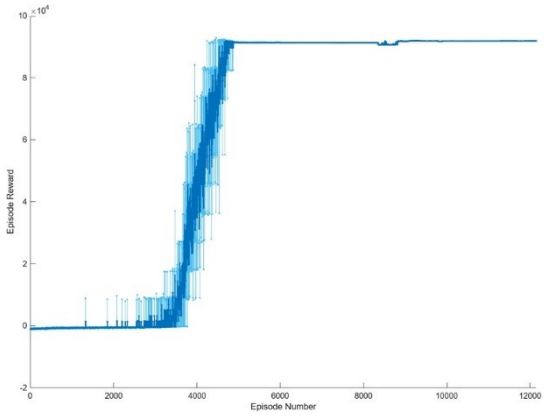

Figure 5: convergence diagram of reward function

Table 1: results obtained from the proposed method

	N-Ld	R-Ld(um)	N-Lg	R-Lg(um)	N-Ls	R-Ls(um)	Vin(v)	Vb(v)	NF-M1	NF-M2
1	**4.50**	**42.0**	**3.25**	**40.7**	**2.50**	**40.9**	**0.68**	**1.50**	**18**	**35**
2	**5.00**	**42.7**	**3.50**	**41.0**	**1.25**	**40.1**	**0.83**	**1.43**	**18**	**18**
3	**3.75**	**78.8**	**3.00**	**60.0**	**1.00**	**43.6**	**0.76**	**1.45**	**23**	**20**
4	**4.00**	**73.0**	**2.50**	**45.1**	**1.25**	**48.1**	**0.66**	**1.45**	**23**	**30**
5	**3.50**	**74.1**	**3.00**	**60.0**	**1.00**	**43.6**	**0.75**	**1.62**	**23**	**30**
6	**4.00**	**40.1**	**2.75**	**40.7**	**1.00**	**61.0**	**0.59**	**1.48**	**27**	**29**
7	**4.75**	**41.0**	**2.75**	**41.1**	**1.25**	**66.0**	**0.58**	**1.42**	**30**	**24**

table. To check the correctness of the values obtained from the proposed method, the values in Table 3 as an example have been further investigated in the design corners.

In Table 4, the effect of process-voltage-temperature design corners on the design of the circuit parameters at 5.7 GHz frequency has been investigated and it shows the correctness of the design done in the proposed method because in all the corners the basic parameters of the circuit are in the acceptable range. are located

In the following, the effect of temperature in the design carried out by the proposed method under the typical-typical process has been investigated. It can be seen in Figure 6 that at the frequency of 5.7 GHz in the temperature range of -20 to 120 degrees Celsius, the changes of S21 are around 0.6 dB.

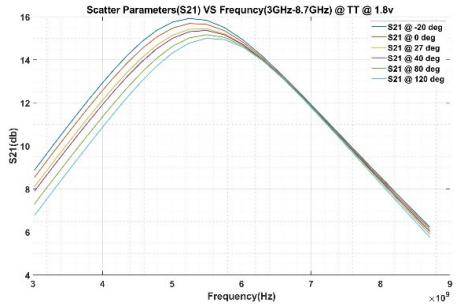

Figure 6: examining the effect of temperature on S21

It can be seen in Figure 7 that at the frequency of 5.7 GHz in the temperature range of -20 to 120 degrees Celsius, the changes of S11 are about 1.3 dB. It can be seen in Figure 8 that at the frequency of 5.7 GHz in the temperature range of -20 to 120 degrees Celsius, the changes of S22 are around 7 dB.

As can be seen in figure 9, at the frequency of 5.7 GHz in the temperature range of -20 to 120 degrees Celsius, the NF changes are about 0.9 dB.

As seen in Figure 6 to Figure 9, the resistance of the design to temperature is evident in the temperature range of

Table 2: basic circuit parameters corresponding to the obtained results

	S21(db)	S11(db)	S22(db)	NF(db)	POWER(mw)
1	**15.30**	**-11.36**	**-17.87**	**2.19**	**10.88**
2	**16.12**	**-14.48**	**-20.51**	**2.29**	**24.22**
3	**15.49**	**-16.88**	**-19.76**	**2.11**	**20.61**
4	**13.93**	**-10.22**	**-15.24**	**2.02**	**12.72**
5	**15.37**	**-21.30**	**-11.54**	**2.07**	**21.44**
6	**12.62**	**-10.68**	**-13.07**	**2.10**	**6.78**
7	**12.60**	**-10.64**	**-22.85**	**2.10**	**6.61**

Table 3: sample result for checking the corners

	N-Ld	R-Ld(um)	N-Lg	R-Lg(um)	N-Ls	R-Ls(um)	Vin(v)	Vb(v)	NF-M1	NF-M2
1	**4.50**	**42.0**	**3.25**	**40.7**	**2.50**	**40.9**	**0.68**	**1.50**	**18**	**35**

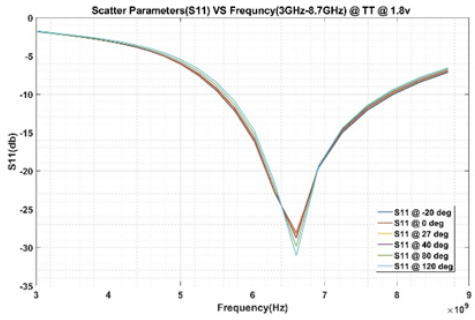

Figure 7: examining the effect of temperature on S11

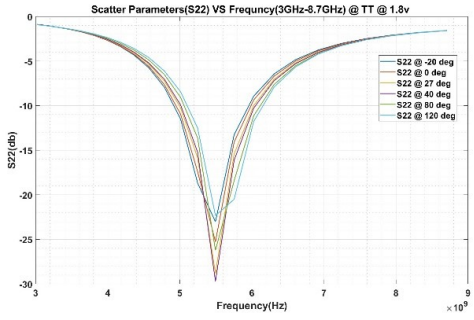

Figure 8: examining the effect of temperature on S22

$-20°C$ to $120°C$ at the frequency of 5.7 GHz.

4 Conclusion

In this article, a method based on machine learning called deep reinforcement learning is presented for the design of high-frequency RF circuits, which performs a wide and effective search within the limits of design with learning capability, and one of the prominent advantages of this method is that it does not require data for Education pointed out. In addition to this, another advantage that this method has is the consistency of the values of the basic elements and parameters of the circuit. In this regard, at each stage, after the decision of the agent in choosing the values, the circuit is simulated with Hspice and the results of the simulation are checked again by the learning agent until the desired goal of the process continues. As is evident in the design results, the answers provided have a high variety, each of which is optimal from different perspectives, which gives the designer many options. Also, from the training

Table 4: checking the effect of the design corners on the basic parameters of the circuit

Power (mw)	Noise Figure(db)	S22 (db)	S11 (db)	S21 (db)	Temperature	voltage	process	
14.32	2.50	-20.32	-9.79	15.91	120	1.44	FF	1
14.06	1.71	-15.93	-11.05	16.63	-20	1.44	FF	2
21.62	2.49	-14.36	-8.25	16.33	120	2.16	FF	3
21.23	1.70	-21.31	-9.64	17.09	-20	2.16	FF	4
5.43	3.19	-9.36	-13.42	11.97	120	1.44	SS	5
4.22	2.10	-7.00	-13.45	12.24	-20	1.44	SS	6
8.18	3.16	-18.18	-11.97	12.90	120	2.16	SS	7
6.36	2.08	-10.96	-12.30	13.20	-20	2.16	SS	8

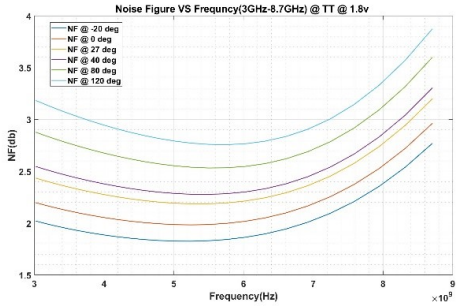

Figure 9: examining the effect of temperature on the noise figure

that the agent has learned It can be used in other designs.

References

[1] M.V. Harsha and B.P. Harish, *Artificial neural network model for design optimization of 2-stage op-amp*, 24th Int. Symp. VLSI Design Test (VDAT), Bhubaneswar, India, 2020.

[2] A. Anthony, C. Kyungwook, and L. Sung Kyu, *VLSI placement parameter optimization using deep*, 39th Int. Conf. Comput.-Aided Design, 2020.

[3] S. Fujimoto, H. Hoof, and D. Meger, *Addressing function approximation error in actor-critic methods*, 5th Int. Conf. Machine Learn., 2018.

[4] L. Gil, N.R. Marku, A.L. Edward, and A.S. Sanjit, *Lwarning heuristics for quantified boolean*, arXiv:1807.08058, Cornell University, 2019, `https://doi.org/10.48550/arXiv.1807.08058`.

[5] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, *Soft actor-critic algorithms and applications*, 2019, `https://arxiv.org/abs/1812.05905`.

[6] W. Haaswijk, E. Collins, B. Seguin, M. Soeken, F. Kaplan, S. Süsstrunk, and G.D. Micheli, *Deep learning for logic optimization algorithms*, IEEE Int. Symp. Circuits Syst. (ISCAS), Florence, 2018.

[7] H. Dong, Z. Ding, and S. Zhang, *Deep Reinforcement Learning*, Springer Singapore, Singapore, 2020.

[8] A. Hosny, S. Hashemi, M. Shalan, and S. Reda, *DRiLLS: Deep reinforcement learning for logic synthesis*, 25th Asia South Pacific Design Autom. Conf. (ASP-DAC), Beijing, 2020.

[9] G. Ian, B. Yoshua, and C. Aaron, *Deep Learning*, MIT Press, 2016.

[10] Y. Li, Y. Wang, Y. Li, R. Zhou, and Z. Lin, *An artificial neural network assisted optimization system for analog design space exploration*, IEEE Trans. Comput.-Aided Design Integ. Circuits Syst. **39** (2020), no. 10, 2640–2653.

[11] G.L. Michail and L.L. Michael, *Learning to select branching rules in the DPLL procedure for satisfiability*, Electronic Notes Discrete Math. **9** (2001), 344–359.

[12] A. Mirhoseini, A. Goldie, M. Yazgan, J. Jiang, E.M. Songhori, S. Wang, and Y.J. Lee, *Chip placement with deep reinforcement learning*, 2020, `https://doi.org/10.48550/arXiv.2004.10746`.

[13] A. Mirhoseini, H. Pham, Q.V. Le, B. Steiner, Y. Zhou, N. Kumar, M. Norouzi, S. Bengio, and J. Dean, *Device placement optimization with reinforcement learning*, 34th Int. Conf. Machine Learn., Sydney, 2017.

[14] S.D. Murphy and K.G. McCarthy, *Automated design of CMOS operational amplifier using a neural network*, 32nd Irish Signals and Systems Conference (ISSC), Athlone, Ireland, 2021.

[15] B. Razavi, *Design of Analog CMOS*, McGraw-Hill Education (2017).

[16] B. Razavi, *RF Microelectronics*, Prentice Hall (1997).

[17] J. Schulman, S. Levine, P. Abbeel, M. Jordan and P. Moritz, *Trust region policy optimization*, 32nd Int. Conf. Machine Learn., 2015.

[18] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, *Proximal policy optimization algorithms*, 2017, `https://arxiv.org/abs/1707.06347`.

[19] K. Settaluri, A. Haj-Ali, Q. Huang, K. Hakhamaneshi, and B. Nikolic, *AutoCkt: Deep reinforcement learning of analog circuit designs*, Design Autom. Test Eur. Conf. Exhib. (DATE), Grenoble, 2020.

[20] M. Sewak, *Deep Reinforcement Learning-Frontiers of Artificial Intelligence*, Springer, 2019.

[21] R.S. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1992.

[22] P.L. Timothy, J.H. Jonathan, P. Alexander, H. Nicolas, E. Tom, T. Yuval, S. David, and W. Daan, *Continuous control with deep reinforcement learning*, 4th. Int. ICLR (2016).

[23] H. Wang, K. Wang, J. Yang, L. Shen, N. Sun, H.S. Lee, and S. Han, *GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning*, 57th ACM/IEEE Design Autom. Conf. (DAC), San Francisco, 2020.

[24] H. Wang, J. Yang, H.S. Lee, and S. Han, *Learning to design circuits*, Cornell University, 2020, `https://doi.org/10.48550/arXiv.1812.02734`.

[25] L. Yann, B. Yoshua, and H. Geoffrey, *Deep learning-review*, Nature **521** (2015), 436–444.

[26] L. Yi-Chen, L. Jeehyun, A. Anthony, S. Kambiz, and L. Sung Kyu, *A generative adversarial framework for clock tree prediction and optimization*, ACM Int. Conf. Comput.-Aided Design (ICCAD'19), Westminster, 2019.

[27] Z. Zhao and L. Zhang, *Deep reinforcement learning for analog circuit sizing*, IEEE Int. Symp. Circuits Syst., Seville, 2020.