ANN based Model Designing and Synthesis of On Chip Square Spiral Inductor for RFICs using ANN and PSO based Algorithms

Amarpreet Singh[#1], Amarjeet kaur[*2]
[#]*Department of Electrical Engg,*
Baba Banda Singh Bahadur Engg College, Fatehgarh Sahib,Punjab
Sirsa,Haryana,India
[1]amarpreet.sran@gmail.com
[*]*Department of Electrical Engg,*
Baba Banda Singh Bahadur Engg College,Fatehgarh Sahib,Punjab
Fatehgarh Sahib,Punjab,India
[2]amarjeet.kaur@bbsbec.ac.in

***Abstract*— In this paper on-chip square spiral inductors are designed using ANN modeling techniques. Layout geometries form the input of the ANN model and electrical quantities forms the output . The dependency of inductor performances as inductance (L), quality factor (Q) and self-resonance frequency (SRF) on geometric dimensions and process technology parameters are described. Spirals of wide range of RF applications are studied. In our ANN based synthesis approach on-chip spiral inductor layout parameters such as spiral outer diameter(D), width of metal trace(W), number of turns in spiral(N), spacing between the adjutants metal traces(S) are taken as input and Inductance ,Q-factor and Self resonance frequency are the output of our model. Further a PSO based searching algorithm is applied with ANN model for optimization of layout parameters for the electrical parameters. We present several synthesis results which show good accuracy with respect to full-wave electromagnetic (EM) simulations. Since the proposed procedure does not require an time consuming EM simulation in the synthesis loop, it substantially reduces the cycle time in RF-circuit design optimization.**

***Keywords*— Artificial neural networks (ANNs), On-chip inductor, MLP, Spiral inductor optimization, Inductance, Q-factor, Self Resonance frequency, electromagnetic simulation , PSO**

I. Introduction

In today's radio frequency integrated circuits (RFICs), on chip spiral inductor represents one of the major components of the RF ICs that dominates circuit performance and most frequently used passive devices in modern RFICs. Silicon substrates have been widely used to fabricate this passive device(Si/SiGe) using CMOS and BiCMOS technologies. To aid in the modeling and design of spiral inductors, many researchers have proposed various techniques and equivalent circuit models. Wireless communication systems are on rapid growth and has stimulated research in low-cost, low-power, and high-performance CMOS RF integrated-circuit (IC) components for system-on-chip solutions. In an RF IC, the operating frequency of on-chip inductors is much lower than the self-resonance frequency (SRF). For example, the requirement of an inductor used in VCO operating at 3 GHz , needs a high Q-factor with self resonance frequency greater than or equal to 6.5 GHz.. Long running EM simulation is required to fulfill this type of high-SRF requirements in the RF IC design. Previous techniques such as numerical approaches that include solution of large number algebraic and differential equations are computationally expensive and time consuming. Time efficient techniques like empirical and analytical techniques are not sufficiently accurate. Artificial neural network (ANN) is a new technique and efficient alternative to above mentioned conventional modeling techniques. It is popular due to its capability of learning any arbitrary nonlinear input–output relationship from corresponding data and also because it produces smooth approximation results from discrete data. ANN model parameters such as weights and biases remain fixed for once trained neural network. Hence, the I/O relationship of the model make a closed form expression, and due to the low latency trained network gives an almost instant output . Neural models are, therefore, much faster than physics/EM models and have a higher accuracy than analytical and empirical models.

Aim of optimization depend upon the application for designing spiral inductors design model .For deciding the optimal inductor-layout geometries that give maximum quality factor at a particular operating frequency and inductance value within a predefined design space long running simulations and large no of efforts are required. The most commonly used approach is the enumeration technique [3] which uses discrete design parameters for simulations and selects the geometry parameters corresponding to the highest *Q* value for design. This technique becomes inefficient when the number of design variables increases because the complexity is exponential. Geometric programming technique for inductor-optimization problem in [10] requires a unique model formulation based on curve fitting. Sequential-quadratic-programming-based optimization technique[7] has been developed which improves the speed over enumeration. But, it has a limitation of getting trapped in the local minima. Optimization technique in[14] is Q–contour is time consuming and technology parameter dependent.Search methods for the optimization that search over all the geometry parameters satisfying a set of constraints are binary search [1] incremental search [2], and genetic algorithm [17] .We have developed a multilayer perceptron (MLP) based neural model of on-chip square spiral inductors and apply the particle-swarm optimization (PSO) algorithm [13],[11],[4][18] to search the layout space for optimization. In exploration, the ANN model is used to compute the inductance (*L*), *Q*, and SRF of each spiral. The synthesis based model procedure provides number of sets of layout parameters for a given inductance value within acceptable error limits. Synthesis results facilitate the designer with more freedom for tradeoff analysis between objectives, such as area, *Q*, and SRF for inductors. The rest of the paper is organized as follows: Section II presents neural network structures. In Section III the spiral- inductor-synthesis procedure. It also gives a brief overview of ANN modeling and PSO. In Section IV modeling and synthesis results are discussed. Finally, the conclusions are drawn in Section V.

II. NEURAL-NETWORK STRUCTURES

For understand what neural networks are and why they have the ability to represent RF and microwave component behaviors neural-network structural issues are described. We study neural networks from the external input–output processing point-of-view. Modeling accuracy dependence on structural issues are discussed. MLP which is popularly used neural-network structure is described in detail.

A. Neural network-Basic Components

A typical neural-network structure has two types of basic components, namely, the processing elements and the interconnections between them. The processing elements are called neurons and the connections between the neurons are known as links or synapses. Each synapses has a corresponding parameter associated with it known as weight. Each neuron receives stimulus from other neurons connected to it, processes the information, and produces an output. Neurons that receive stimuli from outside the network are called input neurons, while neurons whose outputs are externally used are called output neurons. Neurons that receive stimuli from other neurons and whose outputs are stimuli for other neurons in the network are known as hidden neurons. Different neural-network structures can be constructed by using different types of neurons and by connecting them differently.

B. Concept of a Neural-Network Model

Suppose 'm' and 'n' represent the number of input neurons and number of output neurons respectively of a neural network . Suppose 'X' be a n-vector containing the external inputs to the neural network, and 'Y' be a m-vector containing the outputs from the output neurons, and 'W' be a vector containing all the weight parameters representing various interconnections in the neural network. The definition of W, and the manner in which Y is computed from X and W, determine the structure of the neural network. This concepts is used in our model for deciding the structure for the neural model.

C. Neural Network Modeling Versus Conventional Modeling

The neural-network approach can be compared with conventional approaches for a better understanding. The first approach is the detailed modeling approach (e.g., electromagnetic (EM)-based models for passive components and physics-based models for active devices), where the model is defined by a well-established theory. The detailed models are accurate, but could be computationally expensive. The second approach is an approximate modeling approach, which uses either empirical or equivalent-circuit-based models for passive and active components. These models are developed using a mixture of simplified component theory, heuristic interpretation and representation, and/or fitting of experimental data. Evaluation of approximate models is much faster than that of the detailed models. However, the models are limited in terms of accuracy and input parameter range over which they can be accurate. The neural-network approach is a new type of modeling approach where the model can be developed by learning from detailed (accurate) data of the RF component. After training, the neural network becomes a fast and accurate model representing the original component behaviors.

D. MLP Neural Network

1) Structure and Notation :

MLP is a popularly used neural network structure. In the MLP neural network, the neurons are grouped into layers. The first and the last layers are called input and output layers, respectively, and the remaining layers are called hidden layers. Typically, an MLP neural network consists of an input layer, one or more hidden layers, and an output layer, as shown in Fig. (1). For example, an MLP neural network with an input layer, one hidden layer, and an output layer, is referred to as three-layer MLP (or MLP3).Suppose the total number of layers is L. The first layer is the input layer, and the Lth layer is the output layer, and layers 2 to (L-1) are hidden layers. Let the number of neurons in the l^{th} layer be N_l, l= 1, 2, 3…..L . Let W_{ij}^{l} represent the weight of the link between the j^{th} neuron of the $(l\text{-}1)^{th}$ layer and the i^{th} neuron of the l^{th} layer. Let X_i represent the i^{th} external input to the MLP Z_i^{l} and be the output of the i^{th} neuron of the l^{th} layer. There is an additional weight parameter for each neuron (W_{i0}^{l}) representing the bias for the i^{th} neuron of the l^{th} layer. As such, **W** of the MLP includes W_{ij}^{l} j=0,1,2,3…. ,N_{l-1} i=0,1,2,3…. ,N_l and l=2,3,…..,L , i.e **W**=$[W_{10}^{2}\ \ W_{11}^{2}\ \ W_{12}^{2} \ldots\ldots W_{N_L N_{L-1}}^{L}]^T$ .The parameters in the weight vector are real numbers, which are initialized before MLP training. During training, they are changed (updated) iteratively in a systematic manner [8]. Once the neural-network training is completed, the vector

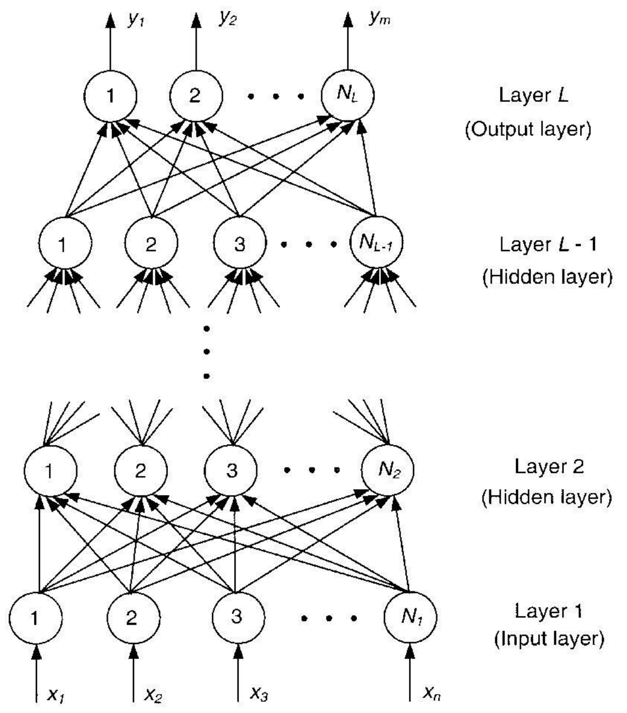

Fig 1 MLP neural-network structure. Typically, an MLP network consists of an input layer, one or more hidden layers, and an output layer.

remains fixed throughout the usage of the neural network as a model

2) Framework of Neurons:

In the MLP network, each neuron processes the stimuli (inputs) received from other neurons. The process is done through a function called the activation function in the neuron, and the processed information becomes the output of the neuron. For example, every neuron in the lth layer receives stimuli from the neurons of the (l-1)th layer, ie $Z_1^{l-1}, Z_2^{l-1}, Z_3^{l-1}, \ldots Z_{N_{l-1}}^{l-1}$. A typical ith neuron in the lth layer processes this information in two steps. Firstly, each of the inputs is multiplied by the corresponding weight parameter and the products are added to produce a weighted sum α_i^l, ie $\alpha_i^l = \sum_{j=0}^{N_{l-1}} W_{ij}^l Z_j^{l-1}$. In order to create the effect of bias parameter (W_{i0}^l) , we assume a fictitious neuron in the (l-1)th layer, whose output is $Z_0^{l-1}=1$. Secondly, the weighted sum in () is used to activate the neuron's activation function {σ(.)} to produce the final output of the neuron $Z_i^l = \sigma(\alpha_i^l)$. This output can, in turn, become the stimulus to neurons in the ($l+1$)th layer. The most commonly used hidden neuron activation function is the sigmoid function given by

$$\sigma(\alpha)=1/(1+\exp^{-\alpha}) \qquad (1)$$

Other functions that can also be used are the arc-tangent ,hyperbolic -tangent function, etc. All these are smooth switch functions that are bounded, continuous, monotonic, and continuously differentiable. Input neurons use a relay activation function and simply relay the external stimuli to the hidden layer neurons, i.e, $Z_i^1=X_i$, i=1,2,3,…..n . In the case of neural networks for RF design, where the purpose is to model continuous electrical parameters, a linear activation function can be used for output neurons. An output neuro computation is given by

$$\sigma(\alpha_i^L)=\alpha_i^L = \{ \sum_{j=0}^{N_{L-1}} W_{ij}^L Z_j^{L-1} \} \qquad . (2)$$

3) Feedforward structure working out :

Given the input vector $\mathbf{X}=[X_1,X_2,X_3,X_4\ldots\ldots X_n]^T$and the weight vector **W** , neural network feedforward computation is a process used to compute the output vector $\mathbf{Y}=[Y_1,Y_2,Y_3,Y_4,\ldots. Y_m]^T$. Feedforward computation is useful not only during neural-network training, but also during the usage of the trained neural model. The external inputs are first fed to the input neurons (i.e., first layer) and the outputs from the input neurons are fed to the hidden neurons of the second layer. Continuing this way, the outputs of the L-1th layer neurons are fed to the output layer neurons (i.e., the Lth layer). During feed forward computation, neural-network weights **W** remain fixed. The computation is given by

$$Z_i^1=X_i \quad i=1,2,3\ldots\ldots N_1; \quad n=N_1 \qquad (3)$$

$$Z_i^l = \sigma \{(\sum_{j=0}^{N_{l-1}} W_{ij}^l Z_j^{l-1})\}, \; ; \qquad (4)$$

$$i = 1,2,3,\ldots N_l \; ; \; l = 2,3,4,\ldots L$$

$$Y_i = Z_i^L \quad i=1,2,3,\ldots N_L \quad m=N_L \; . \qquad (5)$$

4) Vital Features:

It may be noted that the simple formulas in (3)–(5) are now intended for use as RF component models. It is evident that these formulas are much easier to compute than numerically solving theoretical EM or physics equations. This is the reason why neural-networkmodels are much faster than detailed numerical models of RF components. for employing neural networks to approximate RF behaviors, which can be functions of physical/geometrical/bias parameters. MLP neural networks are distributed models, i.e., no single neuron can produce the overall **X**–**Y** relationship. For a given **X** , some neurons are switched on, some are off, and others are in transition. It is this combination of neuron switching states that enables the MLP to represent a given nonlinear input–output mapping. During training process, the MLP's weight parameters are adjusted and, at the end of training, they encode the component information from the corresponding **X**–**Y** training data.

E. Network Size and Layers

For the neural network to be an accurate model of the problem to be learned, a suitable number of hidden neurons are needed. The number of hidden neurons depends upon the degree of nonlinearity of **F** and the dimensionality of **X** and **Y** (i.e., values of **n** and **m**). Highly nonlinear components need more neurons and smoother items need fewer neurons. However, the universal approximation theorem does not specify as to what should be the size of the MLP network. The precise number of hidden neurons required for a given modeling task remains an open question. Users can use either experience or a trial-and-error process to judge the number of hidden neurons. The appropriate number of neurons can also be determined through adaptive processes, which add/delete neurons during training [16], [5]. The number of layers in the MLP can reflect the degree of hierarchical information in the original modeling problem. In general, the MLPs with one or two hidden layers [15] (i.e., three- or four-layer MLPs) are commonly used for RF applications.

III. SPIRAL-INDUCTOR SYNTHESIS

A . Development- ANN Model

We use the above explained Multilayer perceptron (MLP) feedforward network is one of the most effective neural network structures .We consider four inductor-layout parameters, namely, outer diameter (*d*), number of turns (*N*), metal width (*W*), and spacing between metal traces (*s*), forms input of our neural model. As for a given fabrication process designer cannot control the technology parameters hence the are not included in the input parameters of the neural model. The output layer of NN model represent electrical attributes of the inductor which are *L*, *Q*, and SRF. Number of hidden layers and neurons in each hidden layer are decided on the platform of best performance of and Quality of the neural model. Wide combinations of hidden layers with neurons in each hidden layer are employed and the best combination is chosen out. We used hyperbolic-tangent activation function for hidden layers and linear activation function for output neurons. For the generation of training and testing data sets, planar square spiral inductors were constructed in the range of geometric dimensions shown in Table(1) for the CMOS technology . This range covers the layout Geometries for the typical wireless-communication applications. On the basis of uniform grid distribution sampling planning, each input parameter is sampled at an equal interval using the step sizes given in Table I. In nonlinear problems, more refinement or but this costs in increase in the number of spirals to be simulated, thus requiring higher training-time investment. less step size of the data samples is desired to improve the input–output mapping accuracy

Table I
Range Distribution Input

	Minimum	Maximum	Step Size
Outer-Diameter d(um)	100	340	20
Width W(um)	4	32	4
No. of turns N(um)	2.5	8.5	2
Spacing S(um)	1	5	2

Out of all the theoretically possible combinations, we have considered a large number of inductors (400 realizable spirals) have been designed and simulated using commercially available high frequency structure simulator tool Ansoft (HFSS v11.0). The Q-factor and Inductance

$$L = \frac{\mathrm{Im}\,(1/Y_{11})}{2\pi^{*}\mathrm{freq}} \quad , \quad Q = \frac{\mathrm{Im}(1/Y_{11})}{\mathrm{Re}(1/Y_{11})} \qquad (6)$$

Here, Y_{11} is the input admittance of the two-port *Y* parameters. As at the SRF is the reactance becomes zero. Hence at SRF inductance becomes zero and the *Q*-value becomes zero. Thus SRF was measured from the *Q* plot at the frequency point where the *Q*-value becomes zero. Although various methodologies for neural-network-training ,we use the hold-out method for building the neural model to reduce the total cycle time. In this there is no perceptible loss in accuracy. Out of 400 spirals, 80% were used for training, and the remaining 20% were used for testing the neural network. Due to the wide range of the input-parameters for building the neural model the corresponding output-parameter values of the inductors are also quite different. We used a preprocessing step, in which input and output data were normalized to [−1 1] with respect to the minimum and the maximum of the data range. This is done by linear scaling. During neural-network training, the weight and bias values are adjusted to minimize the training error which is a measure of the correlation between the ANN-model output and the training data. We have used the Levenberg–Marquardt method as the training algorithm in MATLAB's neural-network tool for our Model [12]. We set the learning rate as 0.01 which is found adequate, setting it too large leads to oscillations, and setting it too small value results in longer training time for reaching the level of accuracy. The training error goal was set to 0.001. Further lowering of the error limit reduces the generalization capability of the model.

On the other hand, setting it too high would lead to lower mapping accuracy.

B. Particle-Swarm Optimization

In this subsection, we give a brief introduction to PSO. Fig.(2) shows the basic flowchart of PSO. It is an evolutionary technique based on the social behavior, movement, and intelligence of swarms searching for an optimal location. PSO works on a population of potential solution candidates referred to as particles. Each particle in a swarm is represented by a position and velocity vector. Like other evolutionary algorithms, PSO uses a fitness function to search for the best position. Each particle is initialized with a random position and velocity. In every simulation run, the fitness function is evaluated by taking the current position of the particle in the solution space. The particles keep track of two best values.

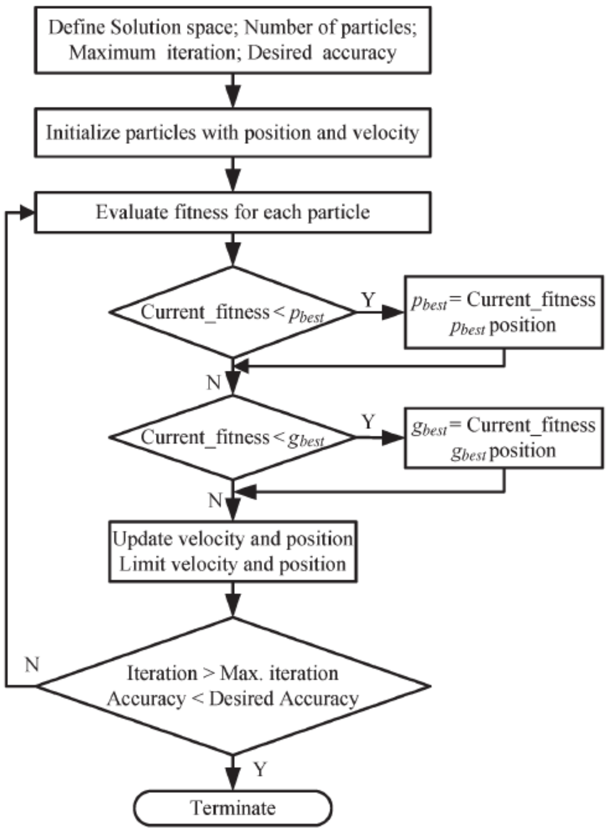
Define Solution space; Number of particles; Maximum iteration; Desired accuracy
Initialize particles with position and velocity
Evaluate fitness for each particle
Current_fitness < p_{best}
Y
p_{best} = Current_fitness
p_{best} position
N
Current_fitness < g_{best}
Y
g_{best} = Current_fitness
g_{best} position
N
Update velocity and position
Limit velocity and position
N
Iteration > Max. iteration
Accuracy < Desired Accuracy
Y
Terminate

Fig.2 Flowchart PSO seach Algorithm

The first one is the best fitness value obtained so far by the particle, the corresponding position being termed as personal best (*p*best). The other is the best fitness value achieved so far considering all the particles in the swarm. The location of the best fitness value in a whole swarm is called global best (*g*best). At each run, there is only one *g*best, and all the particles are attracted toward *g*best. In an iteration, particle velocity and position are updated based on *p*best and *g*best positions as :-

$$V^{n+1}_{id} = W*V^{n}_{id} + C_1\text{rand1}(\,) * (p^{n}_{best} - X^{n}_{id}) + C_2\text{rand2}(\,)*(g^{n}_{best} - X^{n}_{id}) \quad (7)$$

$$X^{n+1}_{id} = X^{n}_{id} + V^{n+1}_{id} \quad (8)$$

Here, $d = 1, 2, \ldots, D$; $i = 1, 2, \ldots, Z$; D being the number of design parameters, Z the swarm size, and n the iteration number. The acceleration factors C_1 and C_2 in (2) indicate the relative attraction toward *p*best and *g*best, respectively. The functions rand1() and rand2() generate random numbers that are uniformly distributed between zero and one. To assign equal weight to the relative pulls of *p*best and *g*best, each of C_1rand1() and C_2rand2() was constructed to have an average value of one by making C_1=C_2=2. The inertia-weight parameter *w* controls the tradeoff between the global and local search capabitity of the swarm. We start with a large inertia weight of 1.0 for an initial bias toward the global search and decrease it linearly to a minimum value of 0.4 through different iterations to facilitate more local explorations [6].

Another important PSO parameter is the maximum/minimum limit on particle velocity. Without any such limit, particles can go out of the solution space. Since there is no actual mechanism to control the velocity of a particle, an external condition is imposed. Ten percent of a particle's position value is set as the limit of its velocity. A position bound of the particles is also similarly imposed. The number of particles in PSO is less critical than in other population-based algorithms. The number typically varies from 10 to 60 depending on the number of design variables and the complexity of the optimization problem. For our problem, it was found that 50 particles were enough to have good convergence. It may be noted that we use the global best version of PSO. This was considered instead of other variants like local best version to improve the speed of processing. It has been shown in [9] that the PSO formulation of (2) is sufficient for avoiding being trapped in the local minima.

C. Synthesis Methodology

On-chip spiral-inductor synthesis is the process of determining the layout geometric parameters from electrical specifications. As it is obvious, a target-inductance value can be realized by many different combinations of layout parameters. Out of these, only the set of inductor-layout parameters that meet all the design constraints is considered. We have developed a spiral-inductor-synthesis procedure that helps the designer to make a tradeoff analysis between the competing objectives, namely, *Q*, SRF, and outer diameter, for a given *L*. Our synthesis procedure uses ANN and PSO. The PSO optimizer generates a swarm of particles, each representing a combination of layout parameters in the given design space. The ANN takes each combination of layout parameters and produces *L*, *Q*, and SRF as output. Cost function is computed using these electrical parameter values. Particles of the optimizer are then updated according to the minimum cost. This process continues until a desired cost function objective is achieved or the maximum number of iterations executed. Usually, the spiral-inductor-design-optimization problem is formulated to maximize the Q value for a target inductance subject to certain constraints. Since, in this synthesis procedure, our aim is to find a set of layout parameters which will give the desired inductance value with in acceptable error, the cost function is to

$$\text{Minimize } L_T - L_{ANN}$$

$$\begin{aligned} \text{subject to } & N\min \le N \le N\max \\ & D\min \le D \le D\max \\ & W\min \le W \le W\max \\ & S\min \le S \le S\max \end{aligned}$$

Here, L_T, L_{ANN}, are the target inductance, the inductance computed from the trained ANN, and the given minimum SRF, respectively. Nmin, Nmax, Dmin ,Dmax, Wmin, Wmax, Smin, and Smax are the minimum and the maximum bounds of the corresponding optimization variables. PSO algorithm provides multiple solutions of layout parameters for a target-inductance value due to the random initialization of particles and the random variables associated with the velocity and position-update process during our synthesis. The search process is terminated if the objective function is less than an acceptable error value or if the number of iterations reaches the maximum. For the synthesis of our spiral inductors, we set the error value is set to 10% and the maximum number of iterations is taken to be 1000.

IV. RESULTS AND DISCUSSION

A. ANN-Model Accuracy

In this paper, we used 400 inductor geometries for training and 100 inductors for testing the neural network. To verify the accuracy of the neural models, statistical measures, such as the average relative error and the correlation coefficient between the outputs and targets were calculated for each output parameter. The average relative error and the correlation coefficients are calculated as follows:

Average Relative Error

$$= \Sigma_1^n (x-y)/ny \qquad (9)$$

Correlation Coefficient

$$= \{(n\Sigma xy - \Sigma x\, \Sigma y)/\{[n\, \Sigma x^2 - (\Sigma x)^2][n\, \Sigma y^2 - (\Sigma y^2)]\}^{0.5}\} \quad (10)$$

Here, n, x and y are the number of samples in the data set, the ANN-model output, and the corresponding EM simulated value, respectively.

Table II
ANN Model Accuracy and Quality

ANN Model-Operating Frequency	1 GHz	2GHz	2.5GHz
Training Epochs	27	42	51
Time of Each Epoch in sec.	1.3315	1.5342	1.7935
Data set	Training	Training	Training
Percentage Average Relative Error in 'L'	0.9837	1.5342	1.5751
Percentage Average Relative Error in 'Q'	5.2433	4.3121	6.4543
Percentage Average Relative Error in 'SRF'	2.4533	3.7123	4.2341

Table III
ANN Model Accuracy and Quality

ANN Model-Operating Frequency	1 GHz	2GHz	2.5GHz
Training Epochs	27	42	51
Time of Each Epoch sec	1.3315	1.5342	1.7935
Data set	Testing	Testing	Testing
Percentage Average Relative Error in 'L'	1.5628	1.9813	1.5554
Percentage Average Relative Error in 'Q'	4.8977	6.6613	6.2334
Percentage Average Relative Error in 'SRF'	4.9398	4.7888	5.4366

Table IV
ANN Model Accuracy and Quality

ANN Model-Operating Frequency	1 GHz	2GHz	2.5GHz
Training Epochs sec	27	42	51
Time of Each Epoch	1.3315	1.5342	1.7935
Data set	Training	Training	Training
Percentage Correlation Coefficient 'L'	0.9999	0.9999	0.9999
Percentage Correlation Coefficient 'Q'	0.9899	0.9991	0.9299
Percentage Correlation Coefficient 'SRF'	0.9999	9.8988	0.9967

Table V
ANN Model Accuracy and Quality

ANN Model-Operating Frequency	1 GHz	2GHz	2.5GHz
Training Epochs	27	42	51
Time of Each Epoch sec	1.3315	1.5342	1.7935
Data set	Testing	Testing	Testing
Percentage Correlation Coefficients'L'	0.9999	0.9991	0.9989
Percentage Correlation Coefficients'Q'	.98991	0.98888	0.9291
Percentage Correlation Cofficient'SRF'	0.9991	0.9962	0.9991

The relative error signifies the closeness of the ANN outputs to the EM simulated values. The correlation coefficient is a measure of how closely the neural output fits with the target values. If this number is equal to 1.0, then there is a perfect fit between the targets and the outputs. Table II and III shows the percentage average error and Table IV and V shows the correlation coefficient of each neural-model output with respect to the EM simulated value. The average relative errors of *L*, *Q*, and SRF were found to be less than 7%. This indicates good accuracy of the trained neural network. In our examples, correlation coefficients are very close to 1.0, which indicates a good fit.

Table VI
SYNTHESIS RESULTS FOR 2.5-nH SPIRAL INDUCTORS AT 2 GHz WITH DESIGN SPECIFICATIONS: SRF >10 GHz, d = 100−300 μm, W = 8−24 μm, N = 2.5−6.5, AND s = 1−4 μm respectively.

L(nH)	D(um)	W(um)	N	S(um)	Q	SRF(GHz)
2.6756	178	8.0	3.1	3.4	11.7416	11.0639
2.5554	207	8.2	2.2	3.3	9.8965	11.9841
2.3442	222	16.3	4.2	2.6	7.6198	13.3314
2.5501	191	9.0	2.5	2.7	9.3523	12.7831

B. *Inductor Synthesis*

During the synthesis process, the objective of optimization is to find inductor structures for a target-inductance value within the desired accuracy level. Table VI shows the layout geometries of the inductors as synthesized by the proposed approach for a desired inductance value of 2.5 nH at 2-GHz operating frequency. It is seen that the PSO-search process generates multiple sets of layout parameters with different *Q* and SRF values. In this example, 4 sets of layout parameters are shown for a target inductance of 2.5 nH within±0.3 nH accuracy. This helps the designer to make a tradeoff between *Q*, area (outer diameter), and SRF. Similarly in Table VII, VIII, IX synthesized layout geometries of spiral inductors for inductance 3-nH at 2.5 GHz,6-nH at 2.5GHz,5.5-nH at 1GHz are shown respectively. It should be noted that due to designing protocols of a particular process it may not be feasible to fabricate all the inductor geometries synthesized by this approach. In this case, the design values may be rounded off to the nearest grid point when doing the layout.

Table VII
SYNTHESIS RESULTS FOR 3-nH SPIRAL INDUCTORS AT 2.5 GHz WITH DESIGN SPECIFICATIONS: SRF > 8 GHz, d = 100−300 μm, W = 8−24 μm, N = 2.5−6.5, AND s = 1−4 μm

L(nH)	D(um)	W(um)	N	S(um)	Q	SRF(GHz)
3.2531	182	9.8	3.6	1.0	7.0264	11.1666
3.1337	222	1.6	4.7	2.6	7.15	9.6262
3.2445	191	9.0	3.0	2.7	11.4182	11.8033

Table VIII
SYNTHESIS RESULTS FOR 6-nH SPIRAL INDUCTORS AT 2.5 GHz WITH DESIGN SPECIFICATIONS: SRF > 6 GHz, $d = 100-300\ \mu$m, $W = 8-24\ \mu$m, $N = 2.5-6.5$, AND $s = 1-4\ \mu$

L(nH)	D(um)	W(um)	N	S(um)	Q	SRF(GHz)
5.9881	289	9.9	3.7	3.8	11.0937	6.8041
6.2301	234	11	5.3	2.5	8.4181	7.7816
5.8134	229	10.5	4.8	3.5	9.4672	7.8349

Table IX
SYNTHESIS RESULTS FOR 5.5-nH SPIRAL INDUCTORS AT 1 GHz WITH DESIGN SPECIFICATIONS: SRF >7 GHz, $d = 100-300\ \mu$m, $W = 8-24\ \mu$m, $N = 2.5-6.5$, AND $s = 1-4\ \mu$

L(nH)	D(um)	W(um)	N	S(um)	Q	SRF(GHz)
4.7638	229	10.6	4.8	3.5	7.9145	8.7636
5.6276	206	9.8	5.9	2.4	7.2818	8.4506
5.5626	240	10.1	4.3	1.6	7.9797	7.0746

V. CONCLUSION

We have proposed fast and efficient layout synthesis system for RF on-chip spiral inductors. A four-layer MLP neural model has been developed. All the output parameters of the neural model show good matching when compared with the data generated by an EM simulator. The synthesis procedure is based on a PSO technique that evaluates the electrical parameters from the geometric parameters using the neural model.. No EM simulation is required during the synthesis procedure thus making the process efficient. The synthesis procedure provides multiple solutions for a given design specification that helps the designer in making a tradeoff between the competing objectives. Several design examples have been presented using the proposed approach. The synthesized inductors were re simulated using the Ansoft HFSS (v11.0) EM solver. The results obtained by our synthesis approach show good agreement with the EM simulation results.

REFERENCES

[1] J. E. Post, “Optimizing the design of spiral inductors on silicon,” *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47, no. 1, 15–17, Jan. 2000.

[2] N. A. Talwalkar, C. P. Yue, and S. S. Wong, “Analysis and synthesis of on-chip spiral inductors,” *IEEE Trans. Electron Devices*, vol. 52, no. 2, 176–182, Feb. 2005

[3] A. M. Niknejad and R. G. Meyer, “Analysis, design, and optimization of spiral inductors and transformers for Si RF IC’s,” *IEEE J. Solid-State Circuits*, vol. 33, no. 10, pp. 1470–1481, Oct. 1998.

[4] R. C. Eberhart and Y. Shi, “Particle swarm optimization: Developments, applications and resources,” in *Proc. Congr. Evol. Comput.*, 2001, vol. 1

[5] T. Y. Kwok and D. Y. Yeung, “Constructive algorithms for structure learning in feedforward neural networks for regression problems,” *IEEE Trans. Neural Networks*, vol. 8, pp. 630–645, May 1997.

[6] Y. Shi and R. C. Eberhart, “Parameter selection in particle swarm optimization,” in *Proc. 7th EP*, pp. 591–600.

[7] Y. Zhan and S. Sapatneker, “Optimization of integrated spiral inductors using sequential quadratic programming,” in *Proc. Des., Autom. Test Eur. Conf. Exhib.*, Feb. 2004, vol. 1, pp. 622–627.

[8] F. Wang, V. K. Devabhaktuni, C. Xi, and Q. J. Zhang, “Neural network structures and training algorithms for microwave applications,” *Int. J. RF Microwave Computer-Aided Eng.*, vol. 9, pp. 216–240, 1999.

[9] J. Park, K. Choi, and D. J. Allstot, “Parasitic-aware RF circuit design and optimization,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 10, 1953–1966, Oct. 2004.

[10] S. S. Mohan, M. M. Hershenson, S. P. Boyd, and T. H. Lee, “Simple accurate expressions for planar spiral inductances,” *IEEE J. Solid-State Circuits*, vol. 34, no. 10, pp. 1419–1424, Oct. 1999.

[11] S. K. Mandal, S. Sural, and A. Patra, “Broadband scalable model for Si-RF on-chip spiral inductors with substrate eddy current effect,” *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 17, no. 6, pp. 560–573, Nov. 2007.

[12] H. Demuth and M. Beale, *Neural Network Toolbox for use with MATLAB*. Natick, MA: MathWorks Inc.

[13] J. Ababneh, M. Khodier, and N. Dib, “Synthesis of interdigital capacitors based on particle swarm optimization and artificial neural network,” *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 16, no. 4, pp. 322– 330, Jul. 2006

[14] C. P. Yue and S. S. Wong, “Physical modeling of spiral inductors on silicon,” *IEEE Trans. Electron Devices*, vol. 47, no. 3, pp. 560–568, Mar. 2000.

[15] J. de Villiers and E. Barnard, “Backpropagation neural nets with one and two hidden layers,” *IEEE Trans. Neural Networks*, vol. 4, pp. 136–141, Jan. 1992.

[16] V. K. Devabhaktuni, M. Yagoub, and Q. J. Zhang, “A robust algorithm for automatic development of neural-network models for microwave applications,” *IEEE Trans. Microwave Theory Tech.*, vol. 49, pp. 2282–2291, Dec. 2001.

[17] T. Wang, Y. Wang, and K. Chen, “A global genetic algorithm based optimization technique for spiral inductor on silicon design,” in *Proc. 5th World Congr. Intell. Control Autom.*, Hangzhou, China, Jun. 15–19, 2004, vol. 3, pp. 2095–2098.

[18] Y. Rahmat-Samii, “Genetic algorithm (GA) and particle swarm optimization (PSO) in engineering electromagnetics,” in *Proc. ICECom*, Dubrovnik, Croatia, Oct. 1–3, 2003, pp. 1–5.