MPPT 算法在光伏阵列上的应用*

陈照民　　谢敏**

广西大学计算机与电子信息学院，南宁，530004

摘　要　分析光伏电池的物理模型，并利用 Simulink 对光伏阵列进行搭建和仿真，采集数据，仿真光伏阵列在不同光照下的输出特性曲线，根据仿真结果分析了出现功率点多峰的原因。然后利用粒子群算法（PSO）、梯度下降法、模拟退火算法对光伏阵列进行最大功率点跟踪（MPPT），并比较各类算法在最大功率点跟踪（MPPT）上应用的不同点。

关键字　光伏阵列，PSO 算法，梯度下降法，MPPT 算法

Application on MPPT Algorithm of Photovoltaic Array

Chen　Zhaomin　　Xie Min

School of Computer, Electronics & Information Guangxi University
Nanning 530004,China
gxxiemin@163.com

Abstract—The main research content of this topic is to analyze the physical model of photovoltaic cell, and use Simulink to build and simulate the photovoltaic array, collect data, and simulate the output characteristic curve of photovoltaic array under different lighting conditions. According to the simulation results, the reason of multiple peaks of power points is analyzed. Then, the particle swarm optimization algorithm (PSO), gradient descent method and simulated annealing algorithm are used to track the maximum power point (MPPT) of photovoltaic array, and the differences of various algorithms in the application of MPPT are compared.

Key words—Photovoltaic array, PSO algorithm, Gradient descent method, MPPT algorithm

1　引　言

在光伏发电系统中，光伏电池是最基础也是最重要的核心部件之一，它能够将光能转换为电能，但是由于单个光伏电池的工作电压和工作电流的范围比较小，其输出的功率很小，并不能满足发电需求，所以为了实现我们想要的合适的电压和电流，就需要将光伏电池进行串并联形成光伏组件，进而形成光伏阵列。

在现实的光伏阵列应用中，光伏阵列的输出功率常常会受到多种因素的影响，而出现光伏阵列输出功率的损失现象，局部遮阴条件下的光伏组件的最大输出功率更是难以确定。外界环境如云层、尘埃、树木和建筑物等对光伏阵列的遮挡会使光伏阵列的输出功率发生改变。在局部遮阴的条件下，光伏阵列的 P-U 输出特性曲线和均匀条件下的光伏阵列的 P-U 输出特性曲线最大的区别就是[1]：均匀条件下，光伏阵列的 P-U 输出特性曲线是呈现单峰特性的，且在峰值点电压的左边功率是逐渐升高的，在功率点电压的右边功率是逐渐降低的。而在局部遮阴的条件下光伏阵列因为组成光伏组件所受到的光照强度不一样就会造成受到光照强度不一样的光伏组件的输出功率会不一致，从而导致了光伏阵列的 P-U 输出特性曲线出现多峰值的现象，这就给利用传统的 MPPT 算法来实现光伏阵列的最大功率点跟踪造成了困难，所以，就必须探寻更好的方法来实现具有优秀性能的 MPPT 算法来跟踪最大功率点。

*基金资助：本文得到广西科技重大专项(桂科 AA21077007)资助.

**通讯作者：谢敏，博士，副教授，gxxiemin@163.com

2　粒子群算法（PSO）

2．1　基本原理

粒子群算法（Particle Swarm Optimization，PSO）是 1995 年 J.Kennedy 和 R.C.Eberhart 等人开发的一种新型进化算法[2]，是经过对人类和鸟群的生活方式的观察和启发总结出的。

粒子群算法中 “粒子”是每一个需要被优化的问题的解所抽象的，每个粒子都有一个适应值，还有一个速度来判断这个群体中粒子的飞行方向和距离，最后将每个个体在当前时间的最优适应值记录下来，作为个体的飞行经验，最后所有的个体都知道这个群体中的最优适应值，并将其坐标作为社会经验。

进行运算时，粒子 i 的第 d 维速度更新公式为：

$$V_{id}^{k} = wv_{id}^{k-1} + c_1 r_1 (pbest_{id} - x_{id}^{k-1}) + c_2 r_2 (gbest_d - x_{id}^{k-1}) \quad (1)$$

粒子 i 的第 d 维位置更新公式：

$$x_{id}^{k} = x_{id}^{k-1} + v_{id}^{k-1} \quad (2)$$

其中：V_{id}^{k} 为第 k 次迭代粒子 i 飞行速度矢量的第 d 维分量；X_{id}^{k} -第 k 次迭代粒子 i 位置矢量的第 d 维分量；c_1 和 c_2 为加速度常数；r_1 和 r_2 为两个随机函数，取值范围[0,1]；w 为惯性权重；$pbest_i$ 为粒子 i 经过的最好位置，$gbest_i$ 为群体经历的最好位置。

粒子群算法的流程图如图 1 所示。

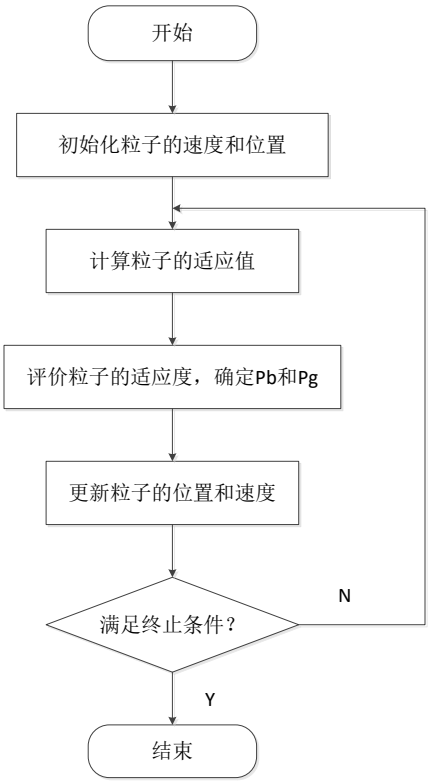

图 1　粒子群算法的流程图

在粒子群算法的 MPPT 应用中，对于算法中参数的调整特别重要，这直接影响了最后的优化结果[3]。在 MPPT 中将会影响我们是否能够达到全局最大值（最大功率点），而不陷入局部最优解，目前对于粒子群算法的参数设置上，主要还是依靠经验来设置。

粒子群算法的构成要素有群体大小 n、权重因子 w、粒子的最大速度 V_{max}、学习因子 c_1 和 c_2。群体大小 n 影响计算的收敛时间，群体值太小，计算时间少，但容易陷入局部最优，群体值太大，收敛时间长，且当群体值大到一定程度，效果就不明显。权重因子 w 能够调节算法整体的计算能力，w 比较大，群体的整体寻优能力强，但是收敛的能力变差，w 比较小则局部寻优强，但是易陷入局部最优。

2．2　粒子群算法的 MPPT 仿真

本文利用Matlab/Simulink进行仿真，仿真的目的是获取光伏阵列在不同光照下的输出特性曲线，然后根据仿真结果分析了出现功率点多峰的原因。仿真的参数如表1所示。

表 1　PSO 算法参数表

种群规模(N)	学习因子	最大迭代次数	权重范围
9	$c1$=$c2$=2	100	0.0--0.9

图2和图3分别给出了在均匀条件下，光伏阵列输出的P-U特性和PSO算法最优个体值。表 2 给出了在均匀条件下，PSO算法计算数据与实际数据对比表。

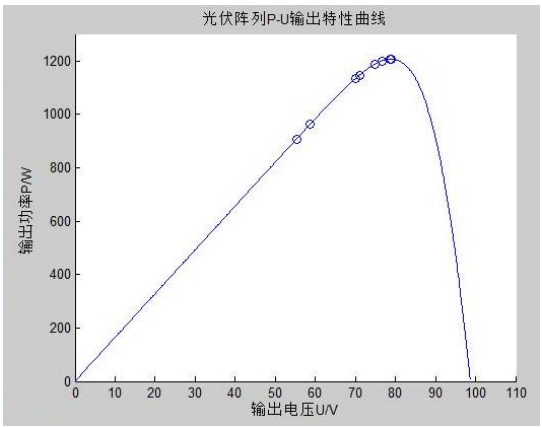

图 2　均匀条件下，光伏阵列输出的 P-U 特性曲线

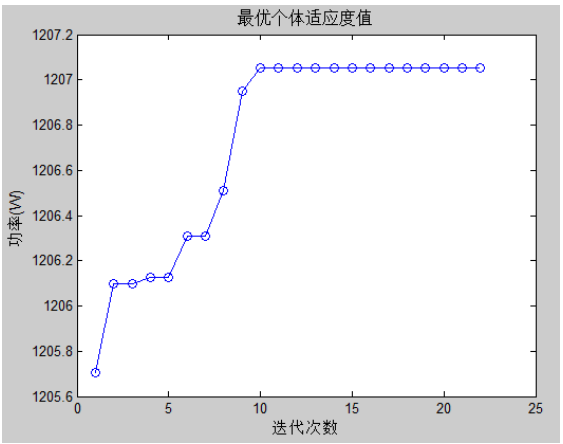

图 3　均匀条件下，PSO 算法最优个体值

表 2　均匀条件下 PSO 算法计算数据与实际数据对比表

算法数据		实际数据	
PSO_maxP(W)	PSO_voltage(V)	MaxP(W)	Voltage(V)
1207.05	78.92	1207.05	78.94

由图2和图3的仿真结果可以看出，在均匀条件下，粒子群算法虽然可以追踪到光伏阵列的最大功率点，但是并不能保证其所有粒子有效收敛。

图4和图5分别给出了在局部阴影条件下，光伏阵

列P-U输出特性和PSO算法最优个体值。表3给出了在局部阴影条件下，PSO算法计算数据与实际数据对比表。由仿真结果可以看出，在局部遮阴（S1=500W/m2，S2=S3=S4=S5=S6=1000W/m²）的条件下，粒子群算法的迭代次数明显增多，虽然可以追踪到最大值点，但是有些点会陷入到局部最优的情况，而达不到全局最大功率点，所以PSO的种群数的设置一定要适当，不然就会造成跟踪点不在功率最大值上，导致功率光伏阵列的功率损失。

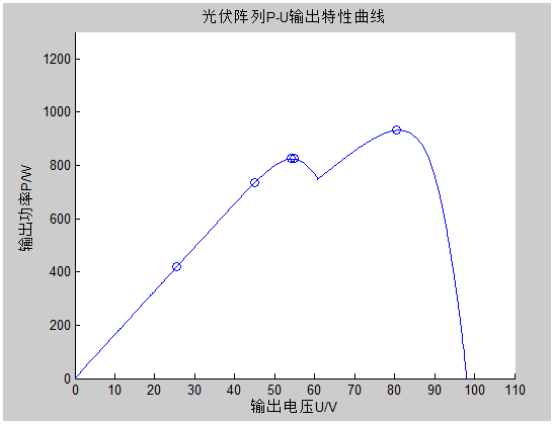

图 4 局部阴影条件下，光伏阵列输出的 P-U 特性曲线

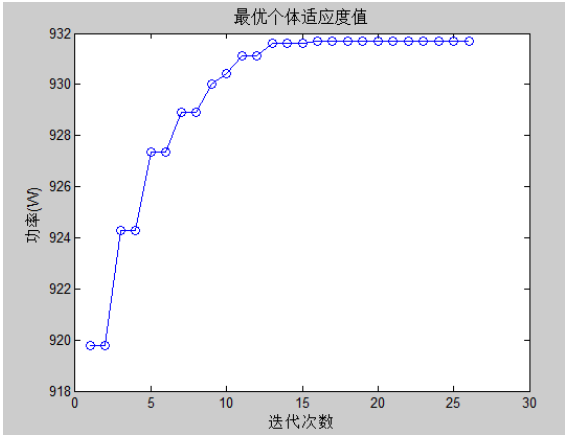

图 5 局部阴影条件下，PSO 算法最优个体值

表 3 局部阴影条件下 PSO 算法计算数据与实际数据对比表

算法数据		实际数据	
PSO_maxP(W)	PSO_voltage(V)	MaxP(W)	Voltage(V)
931.66	54.33	931.9	81.10

3 模拟退火算法

3．1 模拟退火算法基本原理

模拟退火算法（Simulated Annealing，SA）是 Metropolis 等人基于固体的退火过程与组合优化问题两者的相似性而提出的一种算法，其基本思想是在将固体设置在某一高温下，再缓缓地将固体从初始的高温下降温，在固体从高温到低温的缓慢降温过程中，根据 Metropolis 准则[4]，粒子在某一温度 T 时其趋于平衡的概率为 exp(-ΔE/(kT))，其中 E 表示粒子在温度 T 时的内能，ΔE 为粒子自身的内能变化量，k 为 Boltzmann 常数。并且在迭代的过程中反复进行“生成新解-计算函数-以概率进行变化-按准则看是否满足标准”[5]的过程，知道得出优化的目标或者是降温结束，等到算法运行结束时获得近似的最优解。

算法的运算程序如下：

（1） 先对模拟退火过程的温度 Ta 进行初始化，然后在该温度 Ta 下随机生成函数对应的解 M。

（2）2 在当前温度 Ta 的条件下进行如下过程直至在 Ta 的温度下达到平衡的状态为止。

$$M' = M + \varepsilon\xi \tag{3}$$

其中，式中，ε 为幅值的扰动参数，ξ 为随机变量，该变量服从正态分布。即

$$P = \min\{1, e^{-\frac{\Delta f}{T_a}}\} \geq rand(0,1) \tag{4}$$

上式为概率公式，就是说会以该概率来接受新解 M’，如果符合接受准则，则输出该新解 M’，如果不符合接受准则则保留原解 M。

$$T_{a+1} = \delta \times T_a \quad \delta \in (0,1) \tag{5}$$

（3） 最后按照该式执行退火，如果执行退火后满足收敛条件则直接结束运算，如果不收敛则再重新生成新解。

3．2 模拟退火算法的 MPPT 仿真

同样，利用 Matlab/Simulink 进行仿真。仿真的参数如表 4 所示。图 6 给出了均匀条件下 SA 算法最优个体值，表 5 给出了均匀条件下 SA 算法计算数据与实际数据对比表。

表 2 SA 算法参数表

粒子数量	初始温度	迭代次数	温度位移系数	温度概率系数	温度降低速率
10	20	200	0.1	0.05	0.99

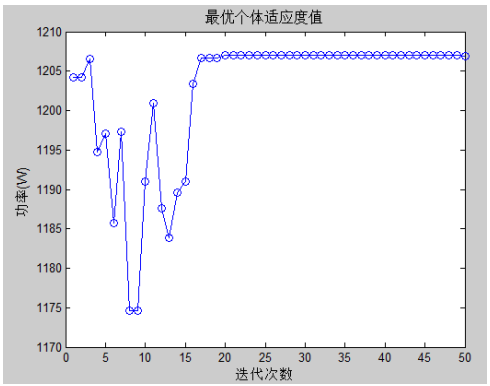

图 6 均匀条件下，SA 算法最优个体值

图7给出了在局部阴影条件下SA算法最优个体值。表5给出了在局部阴影条件下SA算法计算数据与实际数据对比表。

表 4 均匀条件下PSO算法计算数据与实际数据对比表

算法数据		实际数据	
PSO_maxP(W)	PSO_voltage(V)	MaxP(W)	Voltage(V)
1207.05	78.92	1207.05	78.94

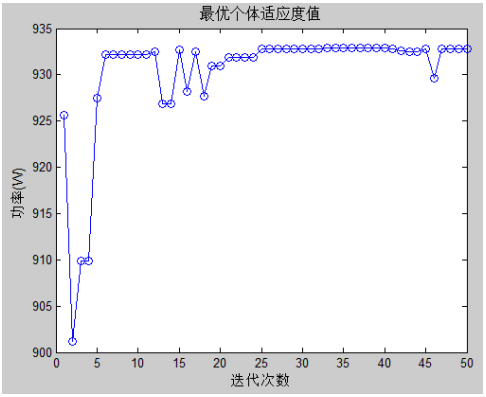

图 7 局部阴影条件下，SA 算法最优个体值

由以上两个仿真结果可以看出，在均匀条件下，模拟退火算法可以追踪到光伏阵列的最大功率点。在局部阴影的条件下（S1=500W/m2，S2=S3=S4=S5=S6=1000W/m²），模拟退火算法也可以找到最大功率点，且更易收敛，但是在局部遮阴的条件下也会存在陷入局部最优解的情况。

表 5 局部阴影条件下 SA 算法计算数据与实际数据对比表

算法数据		实际数据	
SA_maxP(W)	SA_voltage(V)	MaxP(W)	Voltage(V)
932.85	80.96	931.9	81.10

4 BP 神经网络

4．1 BP 神经网络的基本原理

BP（Back Propagation）神经网络是一种基于误差反向传播算法（BP算法）的多层前馈神经网络[7]。网络中层与层间采用的是全互连方式，而同层内的神经元之间不相连接，因此BP算法是一种梯度下降的学习算法。算法的基本思想是外界的输入信息需要经过第一层的输入节点，然后再到中间的隐层节点，最后再到输出节点，最终完成神经网络的前向传播，最后的输出值如果不是期望的输出值，那接下来算法会把实际值和期望值的误差经网络回传，修正各层的权值，多次地训练后，误差就能达到我们想要的期望值[7],使实际输出量和期望输出的量近似相等。

在BP神经网络中，常用的是三层前馈神经网络，输入层、隐含层和输出层。BP神经网络的隐层传递函数为非线性函数，常用的非线性函数有sigmoid函数。

$$f(x)=\frac{1}{1+e^{-\alpha x}} \tag{6}$$

该函数反映了神经元的饱和特性，取值范围为0到1，α为神经元的非线性参数，一般取 1。

由于BP神经网络是非线性系统[7]，初始权值对于能否找到最优解、能否收敛和训练时间的长短有着很大的关系。如果初始值的偏差很大，这会使加权后的输入量落入激活函数的饱和区，从而导致在该点的导数偏小，而在权值的自动修正公式中，由于δ与$f(x)$成正比关系，当 $f(x)$的导数趋向于0时，则有δ趋向于0，最后ω趋向于0，会使得神经网络的调节过程停滞下来。

为了保证神经网络中每个神经元的权值都能够在S函数变化的最大值上，就需要把经过初始加权处理过后的神经元的输出量近似到零，所以连接权的初始值一般是取随机数，且权值要比较小，权值一般取初始值为-1到1之间的一个随机数。对训练样本的输入量要进行归一化，这样可以让那些输入量比较大的值也可以落入激活函数中函数梯度比较大的范围内，如果神经网络的初始权值的范围选择不恰当的话，那么神经网络在学习过程中就有可能进入饱和现象，甚至是陷入局部最优解，而寻不到全局最优解，这样就会带来发电的功率损失。

隐层数对于神经网络的性能也有很大的影响，一般来说隐含层的数量增大误差会减少，但网络也更复杂，在同一训练样本数下，网络的训练时间过大会出现“过度拟合”的趋向错误!未找到引用源。，而Hornik等人证明了只有一个隐含层的多层前馈网络就能够逼近任何函数，同时还可以保证拟合的函数具有很高的精度。所以在设计BP网络时就可以考虑生成一个三层的BP神经网络。隐含层节点的数量是在训练样本时出现过度拟合现象的直接原因。隐层节点数与输入、输出的节点数，激活函数以及样本数据的类型、特性等有关。

一般来说，隐层节点数必须满足的条件如下，隐层节点数必须要小于N-1（N为神经网络的训练样本数）；训练样本数要比网络模型的连接权数多。隐层节点数太少，神经网络的性能就可能很差，甚至出现很大的误差。如果隐层节点数过多，虽然神经网络的系统误差可以训练到很小，但是会让网络的训练时间延长且易陷入局部最优解而得不出全局最优解。为了保证算法能够收敛和神经网络系统在学习过程中的稳定性，合适数值的学习率是必要的，数值比较大的学习率可能会让神经网络的权值修正过度，从而导致权值在反复的修正过程中因为某个误差而出现不收敛的情况；学习率比较小可以保证网络收敛于极小值，但

是神经网络的学习时间会变长。

系统的收敛误差界值可以根据神经网络的收敛时间和样本学习精度来灵活调整，收敛误差界值小，神经网络的学习效果好，但是收敛的速度会比较慢，且需要的训练次数会比较多，当收敛误差值取得比较大的时候收敛速度会比较快，训练次数减少，但是输出的精度会降低。所以这个收敛误差界值可以根据应用的需要灵活选取。

记神经元j为输出层数，神经网络的实际计算输出O_{ij}为y_i，通过将网络的实际输出y_i，和期望输出值d_j之间的误差反向传播来修改神经网络中的各层的权值。以上的作用可以分别用数学式表达出来。

误差定义：

$$e_j = d_j - y_j \tag{7}$$

将期望输出值d_j和实际输出值y_i的方差作为误差测度[7]：

$$E = \frac{1}{2}\sum_j (d_j - y_j)^2 \tag{8}$$

由式(8)可得，E在最小时输出误差ej可取极小值，并由此有，BP算法的目的是想将网络的实际输出无限靠近理想输出，就是要将输出误差控制到尽可能小，按照梯度下降法，要求E的极小值点，则神经网络的权值应该在函数梯度下降的方向下停止：

$$\Delta\omega_{ijk} = -\eta\frac{\partial E}{\partial \omega_{ijk}} \tag{9}$$

其中，η 为学习率，取值范围为 0 到 1。$\Delta\omega_{ijk}$ 与神经元的输出关系为：

$$\Delta\omega_{ijk} = -\eta\frac{\partial E}{\partial \omega_{ijk}} = -\eta\frac{\partial E}{\partial net_{(i+1)k}}\frac{\partial net_{(i+1)k}}{\partial \omega_{ijk}} \tag{10}$$

因为 $\delta_{ik} = -\dfrac{\partial E}{\partial net_{(i+1)k}}$，所以有：

$$\Delta\omega_{ijk} = \eta\delta_{ik}\frac{\partial net_{(i+1)k}}{\partial \omega_{ijk}} \tag{11}$$

首先，求解 $\dfrac{\partial net_{(i+1)k}}{\partial \omega_{ijk}}$。由 $net_{(i+1)k} = \sum_{h=1}^{N_1} o_{ih}\omega_{ijk}$，有：

$$\frac{\partial net_{(i+1)k}}{\partial \omega_{ijk}} = \frac{\partial}{\partial \omega_{ijk}}\left(\sum_{h=1}^{N_1} o_{jh}\omega_{ijk}\right) = O_{ij} \tag{12}$$

$$\Delta\omega_{ijk} = \eta\delta_{ik}O_{ij} \tag{13}$$

然后，使用以下公式计算δ_{ik}。

$$\delta_{ik} = -\frac{\partial E}{\partial net_{(i+1)k}} = -\frac{\partial E}{\partial o_{(i+1)k}}\frac{\partial o_{(i+1)k}}{\partial net_{(i+1)k}} \tag{14}$$

$$\frac{\partial o_{(i+1)k}}{\partial net_{(i+1)k}} = f(net_{(i+1)k}) \tag{15}$$

当$o_{(i+1)k}$为神经网络的输出层神经元时，函数E和$o_{(i+1)k}$有直接的关系，即

$$\frac{\partial E}{\partial o_{(i+1)k}} = y_k - d_k \tag{16}$$

$$\delta_{ik} = -\frac{\partial E}{\partial net_{(i+1)k}} = -\frac{\partial E}{\partial o_{(i+1)k}}\frac{\partial o_{(i+1)k}}{\partial net_{(i+1)k}} = (y_k - d_k)f(net_{(i+1)k}) \tag{17}$$

当$o_{(i+1)k}$是隐含层的神经元时，函数E和$o_{(i+1)k}$就不再是直接关系，运算复杂度就会增大，所以就需要按照$o_{(i+1)k}$是隐含层的神经元还是输出层的神经元来进行处理。此时，要对其进行有效的变换来进行间接的求解。又因为$o_{(i+1)k}$是隐含层，而函数E包含的是神经网络输出层的神经元输出，所以有如下步骤：

$$\frac{\partial E}{\partial o_{(i+1)k}} = \sum_{h=1}^{N_{i+2}}\frac{\partial E}{\partial net_{(i+2)h}}\frac{\partial net_{(i+2)h}}{\partial o_{(i+1)k}} = -\sum_{h=1}^{N_{i+2}}\delta_{(i+1)k}\omega_{(i+1)kh} \tag{18}$$

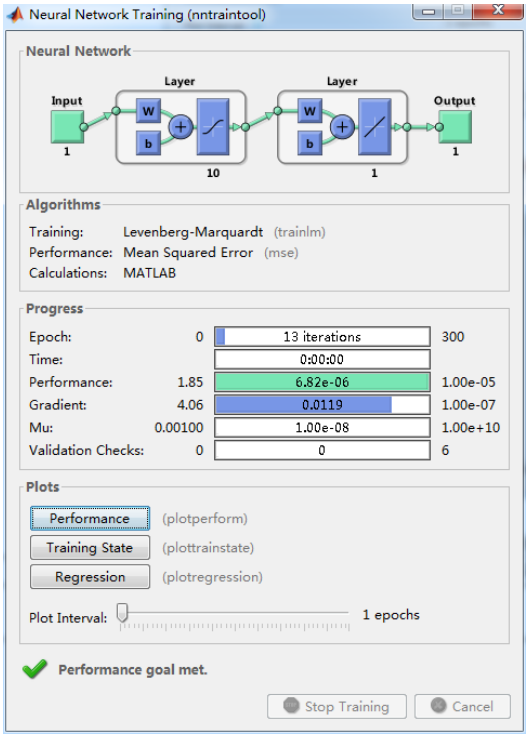

图 8 均匀条件下，BP 神经网络训练结果图

当$o_{(i+1)k}$为隐含层的神经元时，只是知道在该神经节点上的实际输出，但是无法提前知道该节点的正确输出，只知道总的误差和隐含层的输出有关联，同时，隐含层的神经元输出必定会影响下一隐含层的神经节点输入，所以有：

$$\delta_{ik} = f(net_{(i+1)k})(\sum_{h=1}^{N_{i+2}} \delta_{(i+1)k}\omega_{(i+1)kh}) \tag{19}$$

综合以上的分析，BP神经网络的权值调整公式为：

$$\Delta\omega_{ijk} = \begin{cases} \eta(d_k - y_k) f(net_{(i+1)k}) O_{ij} & \text{当i+1层为输出层} \\ \eta f(net_{(i+1)k})(\sum_{h=1}^{N_{i+2}} \delta_{(i+1)k}\omega_{(i+1)kh}) O_{ij} & \text{当i+1层为隐含层} \end{cases} \tag{20}$$

4．2　BP 神经网络的 MPPT 仿真

同样，利用 Matlab/Simulink 进行仿真。图 8 是均匀条件下 BP 神经网络的训练效果图。

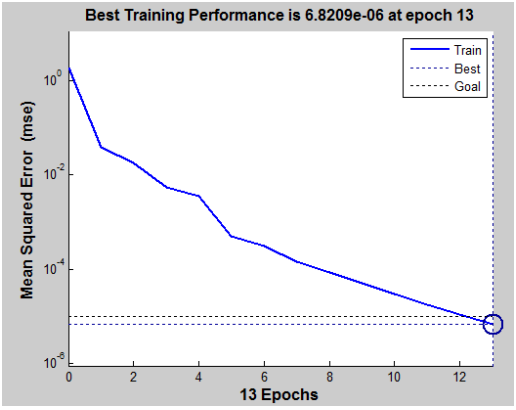

图 9　均匀条件下 BP 神经网络的最优个体值

图 9 给出了均匀条件下 BP 神经网络的最优个体值，表 6 给出了均匀条件下 BP 神经网络计算数据与实际数据对比表。

可以看出，在均匀条件下，经过训练的 BP 神经网络能够以较小的误差输出最大功率点和对应的最大点电压。

表 6　均匀条件下 BP 神经网络计算数据与实际数据对比表

算法数据		实际数据	
PSO_maxP(W)	PSO_voltage(V)	MaxP(W)	Voltage(V)
1204.86	77.10	1207.05	78.94

图10给出了在局部阴影条件下BP神经网络的训练效果图。

图11给出了在局部阴影条件下BP神经网络的最优个体值。表7给出了在局部阴影条件下BP神经网络计算数据与实际数据对比表。从结果可看出，在局部阴影条件（S1=500W/m2，S2=S3=S4=S5=S6=1000W/m²）下，BP神经网络可以比较准确的计算出最大功率点和最大功率点电压。

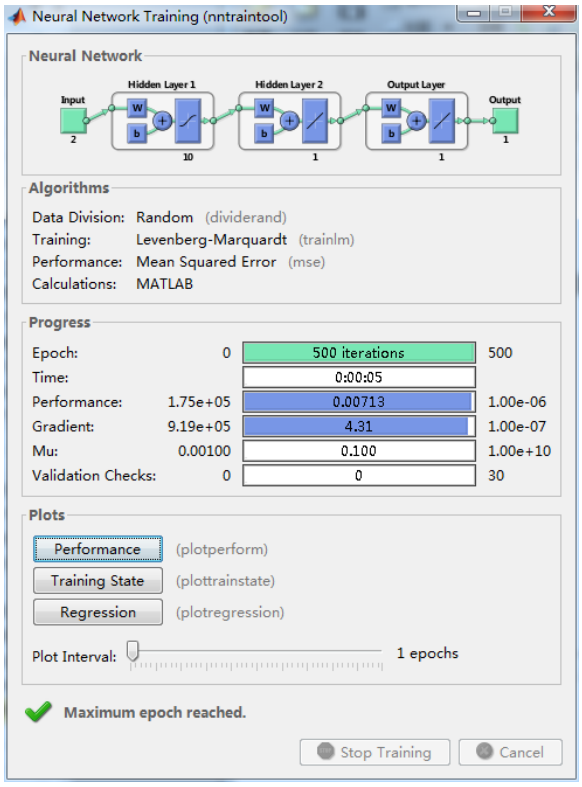

图 10　局部阴影条件下 BP 神经网络的训练效果图

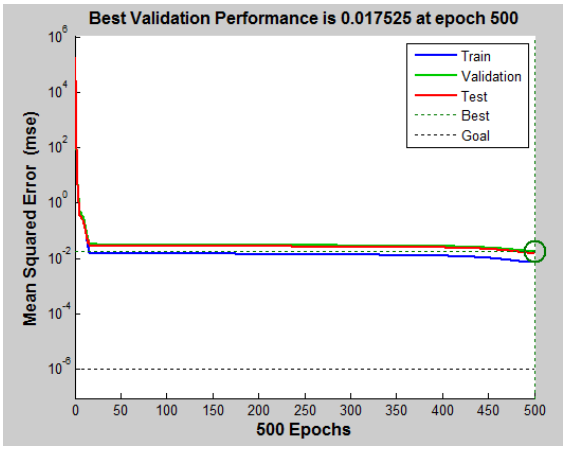

图 11 局部阴影条件下，BP 神经网络的最优个体值

表 7　局部阴影条件下 BP 神经网络计算数据与实际数据对比表

算法数据		实际数据	
SA_maxP(W)	SA_voltage(V)	MaxP(W)	Voltage(V)
911.10	86.06	931.9	81.10

5　多峰值MPPT算法的分析

本文对粒子群算法、模拟退火算法和BP神经网络算法进行了研究，对比了它们互相之间的优缺点，分

析结论如表8所示。

表 8 退火算法和 BP 神经网络算法优缺点

算法名称	优点分析	缺点分析
粒子群算法（PSO）	算法比较精准，收敛的次数也比较少	算法的参数难调整，且容易陷入局部最优解。
模拟退火算法（SA）	算法比较准确，不易陷入局部最优	算法性能与初始值有关、算法参数敏感，收敛性比较差。
BP神经网络算法	只要有足够的多的样本数，就可以得到一个比较准确MPPT预测模型	隐含层的神经元数量比较难确定，且训练参数设置不合适可能会陷入“过拟合”的现象

6 结束语

本文介绍了目前常用的光伏阵列算法及其原理，并指导光伏阵列受外界环境的影响下受到不同的光照时，会出现多峰的输出特性，而传统的单峰值算法很难实现跟踪到最大功率点，所以就需要利用多峰值的MPPT 算法来实现光伏阵列的最大功率点跟踪，并且利用 matlab 将这些常用的多峰算法实现了光伏阵列的最大功率点跟踪，并且将其跟踪的结果进行了对比和分析，得出了粒子群算法（PSO）可以比较精准，收敛的次数也比较少，但是其算法的参数难调整，且容易陷入局部最优解。

模拟退火算法（SA）比较准确，不易陷入局部最优，但是该算法性能与初始值有关、算法参数敏感，收敛性比较差。BP 神经网络算法只要有足够的多的样本数，就可以得到一个比较准确 MPPT 预测模型，但是隐含层的神经元数量比较难确定，且训练参数设置不合适可能会陷入“过拟合”的现象。

参 考 文 献

[1] 吴启琴, 贾学林, 赵俊霞, et al. 光伏阵列输出特性的研究与分析[J]. 科技创新与应用, 2019,(01): 12-16.

[2] 卞楠. 基于粒子群算法的独立光伏发电系统的研究[D]. 天津：天津理工大学, 2016.

[3] 郭洪宾, 于惠钧, 李云钢, 等. 基于自适应粒子群算法在光伏阵列多峰值系统 MPPT 的控制[J]. 湖南文理学院学报(自然科学版), 2019, (2): 45-48+59.

[4] Ishaque K, Salam Z, Amjad M, et al. An Improved Particle Swarm Optimization (PSO) – Based MPPT for PV With Reduced Steady-State Oscillation[J]. IEEE Transactions on Power Electronics, 2012, 27(8): 3627-3638.

[5] 闵轩, 江智军, 杨晓辉. 基于模拟退火-混合蛙跳法的多峰值 MPPT 算法[J]. 水电能源科学, 2018, 36(10): 215-218.

[6] 张瑞成, 何丹阳. 基于模拟退火粒子群算法的最大风能跟踪方法[J]. 工业控制计算机,2018,31(10): 40-41+49.

[7] Hassoun M H., Intrator N, Mckay S, et al. Fundamentals of Artificial Neural Networks[J].Proceedings of the IEEE, 2002, 84(6): 906.