Voc, *Isc*, *T*, *kB*, and *q* are the open circuit voltage, the short circuit current, the temperature, the Boltzmann constant and the electron charge. Assume these parameters have obtained their values in the main Matlab program. If *I* and *V* are the experimental data, we can write an M-file named myfun.m according to Eq. (6b)

```
Function y=myfun(b)
global V, I, Voc, Isc, T, kB, q;
Ical=-(-q*V+(-lambertw(q*b(2)*(Isc-(Voc-b(2)*Isc)/b(3))*exp(-q*Voc/(b(1) *kB*T))*b(3)/(b(2)*b(1)*kB
    *T+b(3)*b(1)*kB*T)*exp(b(3)*q*(b(2)*(Isc+b(2)*Isc/b(3))+V)/b(1)/kB/T/(b(3)+b(2))))+b(3)*q*(b(2)
    *(Isc+b(2)*Isc/b(3))+V)/b(1)/kB/T/(b(3)+b(2)))*b(1)*kB*T)/q/b(2);
y=Ical-I;
```

In above myfun.m file, *Ical* is the calculated current according to Eq. (6b). If we have input the experimental data and set the initial values for n_0, R_{s0}, and R_{sh0}, it is very easy to extract the parameters by invoking myfun.m in the main program just by using two lines of code as the following,

```
b0=[n0, Rs0, Rsh0];
b=lsqnonlin('myfun',b0);
```

Then, b(1) is n , b(2) is R_s and b(2) is R_{sh} .

From above discussion, it is shown that only a few lines of code are required to extract the device parameters. Our proposed method is very simple. The fitting process can be manipulated by setting the parameters of *lsnonlin* function. It is also easy to write Myfun.m according to Eq. (6a).