**高阶微分方程的近似求解算法如下：**

两边经过移项和同乘dt等操作可得(在迭代时，y[i]为0)：

从而得：

其中表示对y的n阶差分，得以递归地得到：

when n>0

上面算法的Java实现如下所示：

public static double d(int n,double[] y,int i)

{

if(n==0)return y[i];

else return d(n-1,y,i)-d(n-1,y,i-1);

}

public double[] output(double[] a,double[] input,double t0,double t1,double[] b)

{

int length=input.length;

double[] x=input,y=new double[length];

double dT=(t1-t0)/(length-1);

double[] dt=new double[a.length>b.length?a.length:b.length];

dt[0]=Math.pow(dT,a.length-1);

for(int i=1;i<dt.length;i++)dt[i]=dt[i-1]/dT;

double a0=a[0];

for(int i=1;i<a.length;i++){a0\*=dT;a0+=a[i];}

for(int i=0;i<length;i++)

{

double yi=0;y[i]=0;

for(int j=0;j<a.length&&i-j>=0;j++)yi-=a[j]\*d(j,y,i)\*dt[j];

for(int j=0;j<b.length&&i-j>=0;j++)yi+=b[j]\*d(j,x,i)\*dt[j];

y[i]=yi/a0;

}

return y;

}