## 电路仿真程序设计

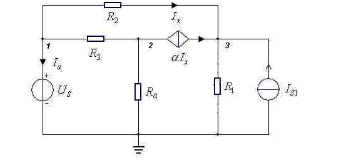
**陈诗元 傅沁宜**

## 任务分工

陈诗元：图形界面、控件和用户操作的实现，数据结构设计与搭建，保存与读取的实现；

傅沁宜：调用matlab进行电路矩阵运算和绘制信号曲线的实现。

## 基本原理

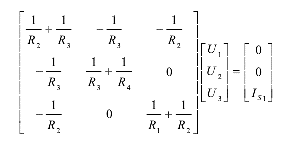


3

0

2

1

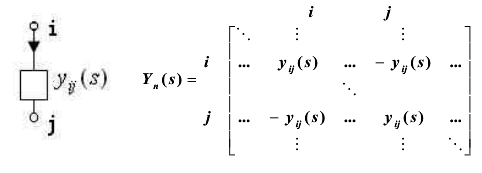
我们用节点电位法求解电路方程。图示电路中，共存在0，1,2，3四个节点。把其中3个节点电位当作变量，另一个节点电位置0，由电流守恒定律可列方程如下：

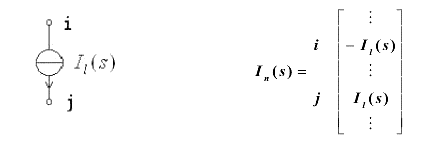
可见对角线上的是某节点相连的电导之和，非对角线上的A(I,j)是节点i与j 间电导的大小

类似的，对于无源二端元件，电流源，电压源，我们的处理是

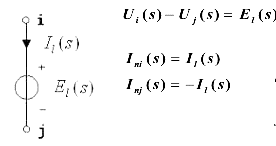
### 电路矩阵：

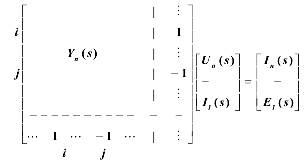
1. 无源二端元件



1. 独立电流源

矩阵方程右边加上电流源流入流出端口分别加上正的电流源项和负的电流源项。

1. 独立电压源

增设求解变量Il(s)取代i端电压，列出如右边矩阵所示方程。

1. 叠加原理

存在多个电流源电压源的情况，电路状态相当于它们独立存在时的情形的叠加。

### 复数计算

对于频率为f 的正弦波，在

电导矩阵中，若为（I,j）节点间为电阻*R*，

若为电容C， (i为虚数单位)

若为电感L,

电流源电压源的信号可由复数表示

### 傅里叶分解

由快速傅里叶变换可以把周期信号（如正弦波，锯齿波，方波）分解成不同频率的正弦波。然后分别带入复数矩阵中计算电位。再用傅里叶反变换叠加即得最终结果。

## 程序特色：

优化了图形界面，提供了便于操作的图形界面，包括可以进行鼠标中键拖动画布，元件的连线、缩放与拖动，示波器的删除以及元件属性的修改等等。

提供了电路的保存和读取。

对屏幕进行网格化处理，同时连线始终正交，使得界面更加干净整洁。

程序采用bitmap的方式进行绘图：先在内存中的某块区域进行绘图，再将其中的一部分复制到显示器上，从而尽量减少屏幕的重绘，因此在用户拖动、拉伸窗口时避免冗余的计算以及屏幕的闪动，大大提高反应效率。

## 操作说明：

打开程序，在工具栏中点击“新建”按钮，可以新建一个项目，此时工具栏的其他选项变为可选。

点击工具栏中的图标，可以进行电阻、电感、电容等元件的创建，将鼠标移到指定地方单击左键，可将该元件放置该区域。在弹出对话框内填入相应参数，即完成元件的创建。

鼠标左键屏幕上已有的元件，可以选中该元件，此时保持左键拖动鼠标可以实现元件的拖动，松开左键即完成拖动。

滑动鼠标滚轮可以对元件进行缩放，同时按住中键并拖动可以对工作区进行整体拖动。

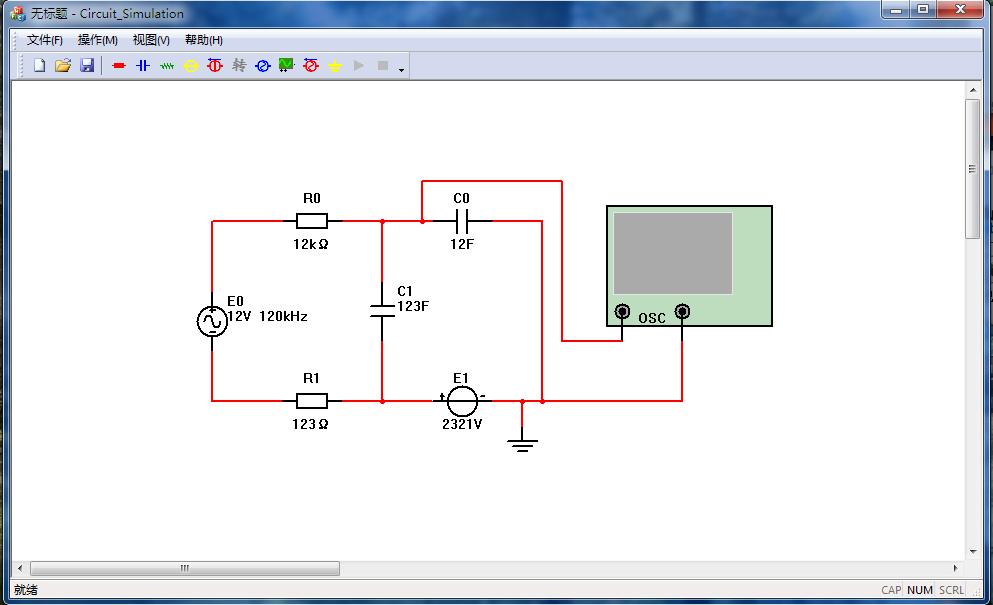
在创建元件或者选中元件后，可以点击工具栏中的“转”或右键菜单选择旋转元件，实现元件的选择。

双击元件，或者选中元件在右键菜单中选择编辑参数，可以对元件参数进行修改。同时还可以更改有源器件的正负极。

将鼠标移至元件节点，单击左键，会进入连线模式，程序会自动使连线垂直，并且会捕捉到最近的网格。在连线过程中可以单击鼠标右键进行取消。

在连线完毕后，单击运行按钮进行仿真，同时双击示波器，在弹出对话框内选择完显示参数后，会弹出波形显示界面。

单击“新建”按钮清空工作区。



## 程序设计：

电路仿真通过mfc调用了matlab软件进行矩阵计算和波形输出。调用通过matlab配套的软件matcom来把matlab语言文件翻译成C语言。因此在程序中加载定义头文件’matlab.h’和lib文件就可以实现matlab的调用。

### Simu类

以上仿真通过Simu类来完成，simu类定义如下：

class Simu

{

public:

// data：

Simu(void){reU.resize(nodes.size(),0);}//初始化各点电位

~Simu(void);

//Mm 类是记录写入写出matlab的数据,其中含有记录矩阵元素的成员变量r(),可以记录int, \*char等类型的数据。默认为double型矩阵。Mex\_d记录的为电导矩阵。

Mm mex\_d;

Mm cacu\_d;//系数矩阵

int nodesnum;//节点总数

vector < vector< complex<double> > > mexv\_d;//电导矩阵

vector < vector< complex<double> > > cacuv\_d;//系数矩阵

vector< complex<double> >reU;//返回电位

//vector:各点仿真电压

//注意： 在其他电源看来，某电压源两端算作短路，两个node算作一个node,需要预先做处理（每次输入的电导矩阵都是不同的），返回时也需要将计算出的点电压分给两个node

//因此电路仿真选择对每个电源做一次计算然后叠加

//functions:

//void mexdInit(SOURCETYPE type,int sourceno);

void mexdInitI(int simusourceno);

void mexdInitE(int simusourceno);

Mm isource(Mm N, Mm d, Mm I, Mm Ip, Mm In, Mm &U);

//计算某电流源对各node点电压的贡献，返回电压值复数

Mm usource(Mm N, Mm d, Mm U, Mm Up, Mm Un, Mm &Uout);

//计算某电压源对各node点电压的贡献，返回电压值复数

Mm sinfigure(Mm X, Mm xpoint, Mm CH1, Mm CH2, Mm ch1on, Mm ch2on, Mm ch1char, Mm ch2char, Mm timechar);

//画波形图

}

### Matlab源文件

其中计算电流源电压源的m文件代码如下：

Isource.m:

function [is]= isource(N,d,I,Ip,In,U)  
%生成系数矩阵  
a=zeros(N-1);  
for iter=1:N-1  
 for j=1:N-1  
 a(iter,j)=a(iter,j)-d(iter+1,j+1);%node0 É¾È¥ÁË  
 end  
end  
  
for kk=1:N-1  
for k=1:N  
a(kk,kk)=d(kk+1,k)+a(kk,kk);  
end  
end   
%生成右边的列矩阵  
b=zeros(1,N-1);  
if Ip>0  
 b(1,Ip)=I;  
end  
if In>0  
 b(1,In)=-I;  
end  
b=conj(b');  
%计算输出  
U=a\b;

Usource.m

function [us]= usource(N,d,U,Up,Un,Uout)  
a=zeros(N-1);  
for kk=1:N-1  
for k=1:N  
a(kk,kk)=d(kk+1,k)+a(kk,kk);  
end  
end  
  
for iter=1:N-1;  
 for j=1:N-1  
 a(iter,j)=a(iter,j)-d(iter+1,j+1);%node0 É¾È¥ÁË  
 end  
end  
b=zeros(1,N-1);  
for iter=1:N-1  
 a(N-1,iter)=0;  
end  
if Up>0  
 a(N-1,Up)=1;  
end  
if Un>0  
 a(N-1,Un)=-1;  
end

%生成右边的列矩阵  
b(N-1)=U;  
b=conj(b');  
b=conj(b');  
%计算输出  
Uout=a\b;

### 仿真波形图

## 设计心得

这次的程序引入了matlab。由于matlab的版本与vs不同，所以通过dll调用失败了。但因为有第三方软件matcom实现了m文件向c语言的转化。这样调用的好处是可以即时调试对m文件内容进行修改。Matlab作为数学计算是一个理想的软件，它语言简单绘图方便，本身带有电路仿真的模块。本程序只是用了它矩阵功能和绘图功能。

在图形界面设计方面，由于时间的关系，很多原本设想的功能未能实现，包括元件的删除等。最麻烦的要属于对于连线操作的设计，在这方面花了较多的时间。

就数据结构而言，我设计了Component类实现对元件属性的封装，Point类实现对元件端点的实现，而主要的Node类则存放着拓扑结构中的节点信息。我构建了必要的相互指向的指针，以便于对其拓扑结构进行操作，但这样带来的一个缺点是保存和读取的函数较为繁琐。

这次大作业由于是时间的限制，做的比较仓促，希望后面能进一步改进。