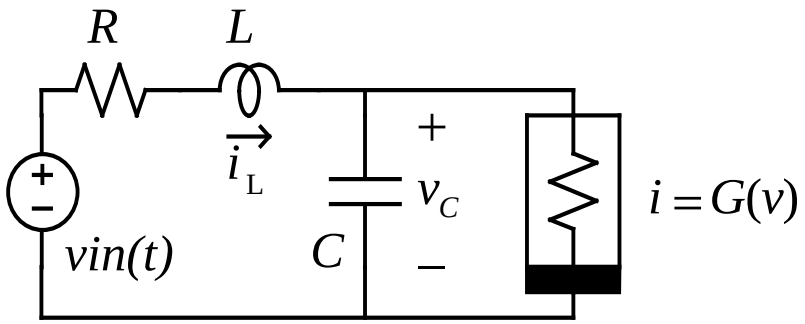
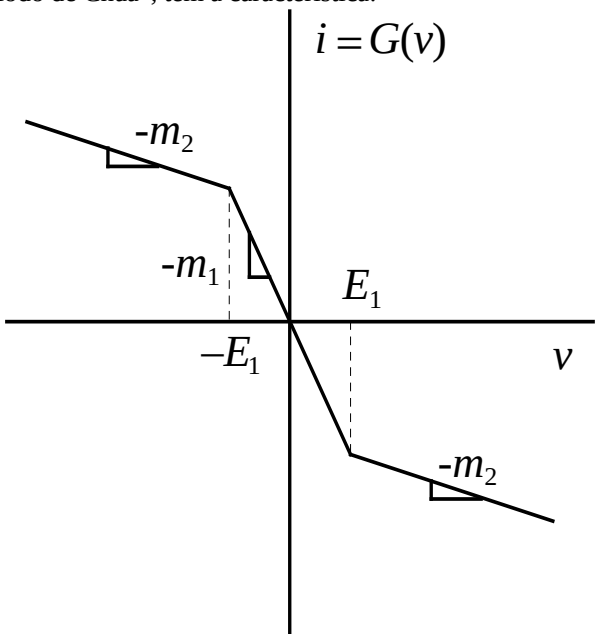
Circuitos Elétricos I – 1º. semestre de 2006 – Trabalho

Escrever um programa que simule o circuito não autônomo de Chua usando equações de estado.

O "circuito não autônomo de Chua" é um dos mais simples circuitos de segunda ordem que exibem comportamento caótico:



O resistor não linear, o "diodo de Chua", tem a característica:



As equações de estado que descrevem o circuito são:

$$\frac{di_L}{dt} = \frac{1}{L}\left(vin(t) - Ri_L(t) - v_C(t)\right)$$

$$\frac{dv_C}{dt} = \frac{1}{C}\left(i_L(t) - G(v_C(t))\right)$$

Onde o resistor não linear é definido por:

$$G(v) = \begin{cases} -m_2 v - m_2 E_1 + m_1 E_1 & \text{se } v < -E_1 \\ -m_1 v & \text{se } -E_1 \le v \le E_1 \\ -m_2 v + m_2 E_1 - m_1 E_1 & \text{se } v > E_1 \end{cases}$$

Com parâmetros adequadamente escolhidos e $vin(t) = F\cos(\omega t)$, o comportamento do circuito produz formas de onda caóticas.

O programa deve operar em uma interface gráfica, calculando e plotando a solução do circuito no tempo, podendo plotar quaisquer das variáveis t, $i_L(t)$, $v_C(t)$, $vin(t)$ umas contra as outras. Todos os parâmetros devem poder ser ajustados antes do início da plotagem.

O cálculo pode ser feito pelo método explícito de Euler:

$$\vec{x}(t_0 + \Delta t) \approx \vec{x}(t_0) + \Delta t \frac{d\vec{x}}{dt}(t_0)$$

$$\vec{x}(t) = \begin{bmatrix} i_L(t) \\ v_C(t) \end{bmatrix}$$

Observando-se que como esse método não é muito preciso, o passo deve ser pequeno. Recomenda-se mais de 100 pontos por ciclo, ao menos. Esse número deve ser ajustável. Note-se que um ciclo da entrada demora $2\pi/\omega$ segundos, o que pode ser usado como estimativa do tempo para um ciclo. O número de ciclos a calcular deve ser ajustável também.

O programa deve plotar as curvas à medida em que faz o cálculo, tão rapidamente quanto possível. Não use linguagens de programação interpretadas, ou que exijam bibliotecas enormes instaladas. O programa executável, mais código fonte, exemplos e tudo o mais que for necessário para executar o programa, deve caber em menos de 1 MB. O programa deve rodar em Windows.

Possibilidades:

É interessante fazer um gráfico de $G(v)$ junto com um de v/R. Os dois gráficos devem se interceptar, na condição em que o circuito gera caos.

Note-se que o circuito permite simular outros comportamentos, como:
Um ressonador LC, com R pequeno, $F = 0$ e $G(v) = 0$. Útil para "debugar" o programa.
Um oscilador com limitador não linear, fazendo m_2 negativo , R pequeno e $F = 0$. Idêntico ao que está em um exemplo do livro de Desoer e Kuh.

É possível usar o método dos trapézios, muito mais preciso:

$$\vec{x}(t_0 + \Delta t) \approx \vec{x}(t_0) + \frac{\Delta t}{2}\left(\frac{d\vec{x}}{dt}(t_0) + \frac{d\vec{x}}{dt}(t_0 + \Delta t)\right)$$

Notando que $G(v)$ é linear por partes, dentro de um segmento qualquer pode-se colocar as equações de estado na forma linear:

$$\frac{d\vec{x}}{dt} = [A(t)]\vec{x}(t) + \vec{B}(t)u(t)$$

E assim:

$$\vec{x}(t_0 + \Delta t) \approx$$

$$\approx \left[[I] - \frac{\Delta t}{2}[A(t_0 + \Delta t)]\right]^{-1}\left[\vec{x}(t_0) + \frac{\Delta t}{2}[A(t_0)] + \frac{\Delta t}{2}\vec{B}(t_0)u(t_0) + \frac{\Delta t}{2}\vec{B}(t_0 + \Delta t)u(t_0 + \Delta t)\right]$$

A matriz $[A]$ e o vetor B podem ser usados com seus valores no segmento atual, e $u(t)$=1. Para maior precisão, pode-se conferir se o segmento se manteve, e se mudou, refazer o cálculo com $[A(t_o+\Delta t)]$ e $B(t_o+\Delta t)$ no novo segmento. Não deve fazer muita diferença.

É também possível resolver o circuito exatamente, dentro de cada segmento de $G(v)$, usando a fórmula para a solução das equações de estado. Da mesma forma, a matriz $[A]$ e o vetor B são constantes dentro

de um segmento. A integral de convolução pode ser simplificada assumindo *vin*(*t*) constante dentro do intervalo Δt (use o valor no centro do intervalo), ou resolvida exatamente. Notar que a entrada tem *vin*(*t*) e uma fonte constante devida à forma de *G*(*v*). Assume-se aí que durante um intervalo de tempo o circuito opera dentro de um único segmento. Se isso não acontecer, há um pequeno erro, que depende do tamanho do intervalo de tempo usado. Pode-se, com alguma complicação, localizar os instantes onde há mudança de segmento e eliminar esse erro.

É útil poder salvar os resultados da última análise como referência, para comprovar a alta sensibilidade de um sistema caótico a qualquer variação de seus parâmetros. Mínimas variações geram grandes mudanças nas formas de onda em poucos ciclos.

Sobre como fazer gráficos no Windows ou outros sistemas com interface gráfica orientada a objeto:

Em sistemas como o Delphi e o C++ da Borland (recomendo Delphi 3), existe um painel gráfico que pode ser colocado em uma janela. O painel gráfico tem uma rotina de redesenho que o programador tem que escrever. Todo o código de cálculo e desenho pode ficar nesta rotina. A rotina pegaria os parâmetros de caixas de texto colocadas nesta mesma janela ou em outra. Entre estes parâmetros estariam os limites do gráfico a plotar. Sejam estes *x*min, *x*max, *y*min e *y*max. O gráfico seria plotado dentro de uma janela entre os limites *h*min, *h*max, *v*min e *v*max, em pixels. Estes parâmetros podem ser lidos dos parâmetros da janela. Um ponto (*x*,*y*) corresponde a um pixel na tela (*h*,*v*) que pode ser calculado como:

$$h = \text{round}(a_x x + b_x)$$
$$v = \text{round}(a_y x + b_y)$$

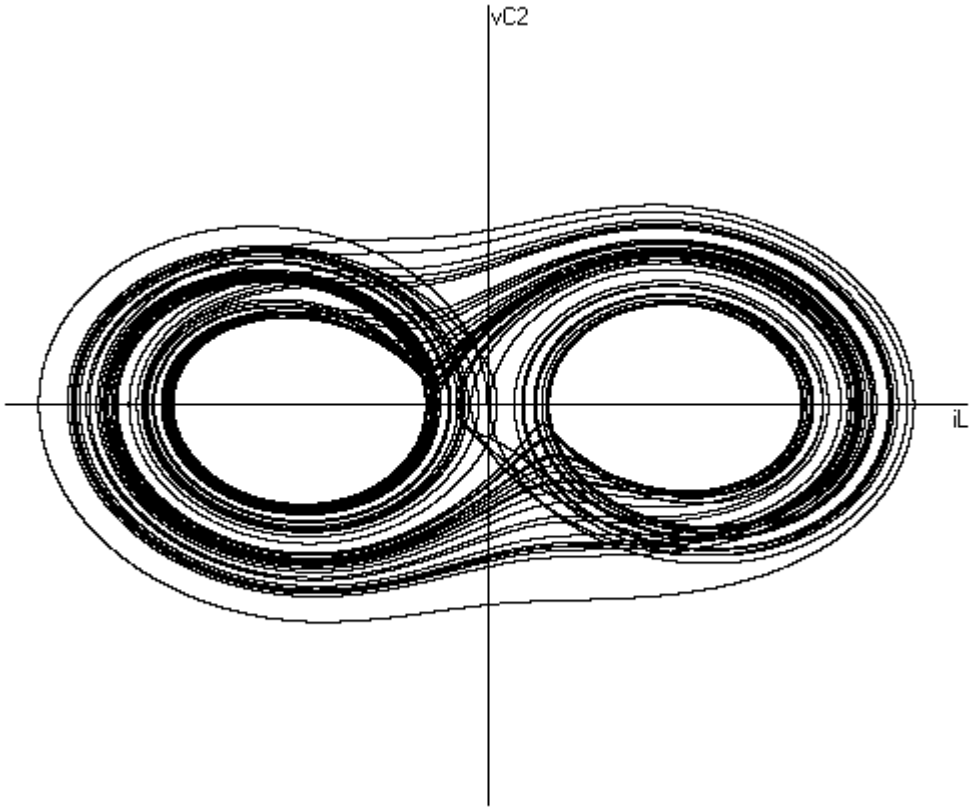
(É necessário limitar os valores aplicados à função de arredondamento (round) de forma a que não excedam os limites para números inteiros.)
As constantes de mapeamento a_x, a_y, b_x e b_y podem ser calculadas sabendo-se que:

$$a_x x\min + b_x = h\min$$
$$a_x x\max + b_x = h\max$$
$$a_y y\min + b_y = v\max$$
$$a_y y\max + b_y = v\min$$

De onde vem:

$$a_x = \frac{h\max - h\min}{x\max - x\min}$$
$$b_x = h\min - a_x x\min$$
$$a_y = \frac{v\min - v\max}{y\max - y\min}$$
$$b_y = h\min - a_y x\min$$

Os gráficos são então feitos com a função “lineto(h,v)”. O primeiro ponto é marcado com a função “moveto(h,v)”. As mesmas funções servem para plotar eixos. Para fazer o cálculo e o gráfico, basta mandar redesenhar a janela gráfica (com o nome “Janelagrafica”, no caso), tipicamente com os comandos:
Janelagrafica.Invalidate;
Janelagrafica.Show;
Estes comandos seriam colocados em uma rotina acionada por um botão.



A figura mostra uma forma de onda caótica similar à que esse circuito deve gerar. Mas essa figura foi obtida com outra estrutura de circuito (o circuito autônomo de Chua).

Última atualização: 15/5/2006
Antonio Carlos M. de Queiroz