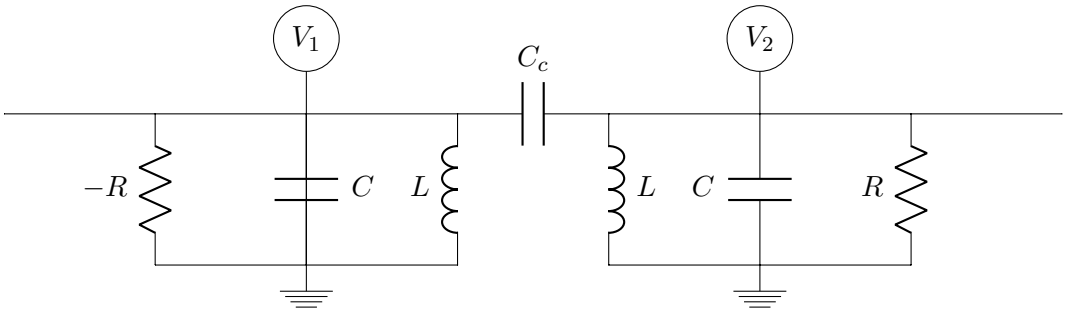
Research Summary

Cesar Eduardo Garza

July 10, 2019

In our research into Topological Electronic Lattices, we have looked into RLC circuits with the following layout:

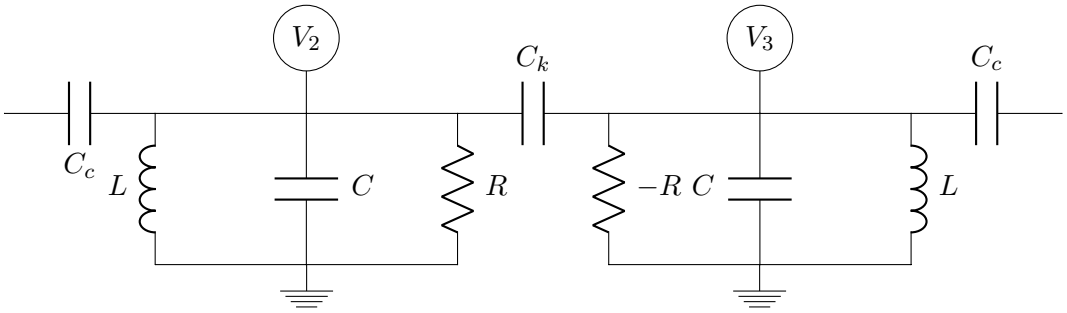


The two loops separated by the Capacitor C_c will henceforth be referred to as "subcircuits". We then applied Kirchhoff's current and voltage laws to obtain the following matrix:

$$\begin{pmatrix} -i\gamma + \frac{1}{\omega} - \omega(1+\kappa) & \omega\kappa \\ \omega\kappa & i\gamma + \frac{1}{\omega} - \omega(1+\kappa) \end{pmatrix} \begin{pmatrix} V_1 \\ V_2 \end{pmatrix}$$

where $\kappa = \frac{C_c}{C}$, $\gamma = \frac{1}{R}\sqrt{\frac{L}{C}}$, and $\omega = \omega'\sqrt{LC}$

We then coupled an arbitrary amount of these circuits using a capacitor C_k as shown below



This coupling gave us the following system of equations:

$$\begin{pmatrix} a^* & b & 0 & 0 & \dots & 0 \\ b & d & c & 0 & \dots & 0 \\ 0 & c & d^* & b & \dots & 0 \\ 0 & 0 & b & d & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & b & a \end{pmatrix} \begin{pmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ \vdots \\ V_{2n} \end{pmatrix}$$

Where $a = i\gamma + \frac{1}{\omega} - \omega(1+\kappa)$, $b = \omega\kappa$, $c = \omega\kappa'$, $d = i\gamma + \frac{1}{\omega} - \omega(1+\kappa+\kappa')$, $\kappa' = \frac{C_k}{C}$, and x^* refers to the complex conjugate of x.

If we refer to the matrix as A, then it is easy to see that $A^T = A$, and the matrix is symmetric. Additionally, it is easy to see that $\mathrm{Tr}\,(A) \in \mathbb{R}$ which, while it is not definitive proof, is a strong indication that eigenvalues of A will be found in complex conjugate pairs.

To evaluate the eigenvalues of A symbolically, sympy was used. The following is the code used to generate the matrices for 4 circuits coupled together:

```
1 from sympy import *
2 init_printing(use_unicode = True)
3
4 w, k, kappa, g = symbols("w k k' g")
5
```

```
6
7  a, b, c, d = symbols('a b c d')
8  firstLine = [-I*g+1/w-w*(1+k),w*k]
9  evenLine = [w*k, I*g+1/w-w*(1+k+kappa), w*kappa]
10 oddLine = [w*kappa, -I*g+1/w-w*(1+k+kappa), w*k]
11 lastLine = [w*k, I*g+1/w-w*(1+k)]
12
13
14 def generateMatrix(size):
15     mat = []
16     mat.append([*firstLine, *[0]*2*(size-1)])
17     for i in range((size - 1) * 2):
18         if i % 2 is 0:
19             mat.append([*[0]*i, *evenLine, *[0]*(2*(size-1)-i - 1)])
20         else:
21             mat.append([*[0]*i, *oddLine, *[0]*(2*(size - 1) - i - 1)])
22     mat.append([*[0]*2*(size - 1), *lastLine])
23     return Matrix(mat)
24
25 fourCase = generateMatrix(4)
26 fourEigen = fourCase.eigenvals()
27 pprint(simplify(fourEigen))
```

From running the above code on cases for $n = 2, 3, 4, 5, 6$, we determined that eigenvalues follow the following order: First, due to A being tridiagonal, there will always be the two trivial eigenvalues that follow for all couplings of the circuits, even for $n = 1$:

$$\lambda = -\omega(1 + \kappa) + \frac{1}{\omega} \pm \sqrt{-\gamma^2 + \omega^2\kappa^2}$$

For cases $n > 1$, there are the following additional eigenvalues:

$$\lambda = -\omega(1 + \kappa + \kappa') + \frac{1}{\omega} \pm \sqrt{-\gamma^2 + \omega^2(\kappa^2 + \lambda'\kappa\kappa' + \kappa'^2)}$$

where λ' is an as-of-yet undetermined variable. Using the code above, we determined the following values of λ' for the following values n:

n	λ'
2	0
3	± 1
4	$0, \pm\sqrt{2}$
5	$\frac{\pm 1 \pm \sqrt{5}}{2}$
6	$0, \pm 1, \pm\sqrt{3}$

The appearance of the golden ratio ϕ for the case $n = 5$ is especially intriguing, as it may point towards a geometric explanation. Additionally, cases of even n will always include $\lambda' = 0$ and cases of odd n will never include $\lambda' = 0$.

It is interesting that λ is a quadratic function of ω^2, meaning that if we were to set $\lambda = 0$, we will always be able to explicitly solve for the normal modes of the coupled circuits. Indeed, using the following code, I solved for normal modes given λ':

```
from sympy import *
init_printing(use_unicode = True)
w, k, kappa, g = symbols("w k k' g")
a, b, c, d = symbols('a b c d')

trivialCase = [-w*(1+k) +sqrt(-g**2+k**2 * w**2)+1/w, \
 -w*(1+k) - sqrt(-g ** 2+k ** 2*w ** 2)+1/w]

def solver(li):
    output = set()
    for i in li:
        x = solve(i, w)
        output.update(x)

    return(output)

def pprinter(li):
    for i in li:
        pprint(simplify(i))

def generateEigenValues(li):
    mat = [*trivialCase]
    base = -w * (1+k) + 1/w
```

```
    for i in li:
        mat.append(base + sqrt(-(g ** 2) + \
        (w ** 2) * (k ** 2 + i * k * kappa + kappa ** 2) ))
        mat.append(base - sqrt(-(g ** 2) + \
       (w ** 2) * (k ** 2 + i * k * kappa + kappa ** 2) ))

    return mat

genCase = [a, b, c]
genEig = generateEigenValues(genCase)
pprinter(solver(genEig))
```

This resulted in generating the normal modes for A. The two trivial cases for λ generate the following four normal modes:

$$\omega = \pm \frac{\sqrt{(1 + \kappa + \frac{1}{2}\gamma^2) \pm \sqrt{\frac{1}{4}\gamma^4 - \gamma^2\kappa - \gamma^2 + \kappa^2}}}{2\sqrt{2\kappa + 1}}$$

Other cases for λ generate the following normal modes:

$$\omega = \pm \frac{\sqrt{(1 + \kappa + \frac{1}{2}\gamma^2) \pm \sqrt{\frac{1}{4}\gamma^4 - \gamma^2\kappa - \gamma^2 + \kappa'^2 - \lambda'\kappa\kappa'}}}{2\sqrt{2\kappa + 1 + \kappa'^2 + \lambda'\kappa\kappa'}}$$

Currently, we are attempting to find more values of λ' for larger and larger n to attempt to find a pattern and determine λ' from n. This would be invaluable in finding all normal modes of the circuit for arbitrary n, as well as taking the limit as $n \to \infty$. Additionally, we are in the process of obtaining the necessary materials to create this circuit and test it. I have also written a numerical solver for the eigenvalues for use when testing, and can use this to compare with Fatemeh's results.

Thank you for your time.