UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA MECÁNICA

ALGORITMOS NUMÉRICOS DE RECONSTRUCCIÓN DE CONDUCTANCIAS

Memoria

Autora: Fàtima el Baghdadi

Directora: Ángeles Carmona

Departamento MAT

Co-Director: Andrés Encinas

Convocatoria: Junio 2019

Algoritmos numéricos de reconstrucción de conductancias

TRABAJO DE FIN DE GRADO

Junio del 2019

Autora: Fàtima el Baghdadi

Directores: Ángeles Carmona y Andrés Encinas

Grado: Ingeniería mecánica

Convocatoria: Junio del 2019

Escola d'Enginyeria Est de Barcelona

Índex

Índex de figures

1 Resumen

La idea de este trabajo surge de la investigación que están llevando a cabo mis tutores del departamento de matemáticas de la EEBE. Es una investigación que estudia los problemas inversos y deduce algoritmos para la recuperación de las características interiores de un elemento a partir de aplicarle una entrada y obtener la respuesta tal como se visualiza en la siguiente figura:

Figura 1: Metodología del problema inverso. Elaboración propia

Este trabajo trata de implementar algunos de éstos algoritmos en Matlab y demostrar su veracidad además de visualizar el potencial que pueden llegar a tener, ya que se pueden aplicar en múltiples ámbitos. Este proyecto se puede ampliar en múltiples áreas ya que es el principio de una potente herramienta para el uso industrial, médico, astronómico,... En esta memoria se recoge los fundamentos teóricos del trabajo como los algoritmos implementados. Estos algoritmos proceden de los artículos citados en las referencias. Para finalizar, se ofrecen una serie de conclusiones sobre el desarrollo de este trabajo y el cumplimiento de los objetivos marcados.

Palabras claves: algoritmo, red Spider, recuperación de conductancias, problema inverso.

1.1 Resum

La idea d'aquest treball sorgeix de la investigació que els meus tutors estan duent a terme al departament de matemàtiques de la EEBE. Es tracta d'una investigació que estudia els problemes inversos i dedueix els algoritmes per a la recuperació de les característiques interiors d'un element a partir d'una aplicació d'una entrada i l'obtenció de la resposta com es pot observar a la figura següent:

Figura 2: Metodologia del problema invers. Elaboració pròpia

Aquest treball implementa alguns d'aquests algorismes en MATLAB i demostra la seva veracitat, a més a més s'adverteix del potencial que tenen, ja que es poden aplicar en diversos camps. Aquest projecte es pot ampliar en diferents àrees, ja que és el principi d'una poderosa eina per a l'ús industrial, mèdic, astronòmic,... En aquest informe es recullen els fonaments teòrics del treball com els algoritmes implementats. Aquests algoritmes provenen dels articles citats a les referències. Finalment, s'ofereixen una sèrie de conclusions sobre el desenvolupament d'aquest treball i el compliment dels objectius marcats.

Paraules claus: algorisme, xarxa Spider, recuperació de conductàncies, problema invers.

1.2 Abstract

The idea of this paper arises from the research that my tutors are carrying out at the department of mathematics of the EEBE. It is a research that studies the inverse problems and demonstrates algorithms for the recovery of the internal characteristics of an element from an input and get the answer as visualized in the following figure:

Figura 3: Inverse problem methodology. Source: prepared by the author

This paper implement some of these algorithms in Matlab and demonstrate their veracity. Moreover, it shows the potential that they can have because they can be applied in multiple fields. This project can be expanded in multiple areas since it is the principle a powerful tool for industrial, medical, astronomical, ... In this end-of-degree project, the theoretical foundations of the work are collected as the algorithms implemented. These algorithms come from the articles

cited in the references. Finally, a series of conclusions are offered on the development of this work and the fulfillment of the objectives set.

Keywords: algorithm, Spider network, recovery of conductancy, inverse problem.

2 Agradecimientos

Me gustaría agradecer el esfuerzo de mis tutores por la ayuda recibida y material proporcionado para poder realizar este trabajo. También me gustaría darles las gracias a las personas que me han apoyado, en especial, a mi querida hermana, la Anna Juanola, Alba Bartrés y Pau Saldaña.

3 Introducción

El 19 de abril de 2019 los medios de comunicación anunciaron un avance muy importante: la primera fotografia de la historia de un agujero negro. Con lo que se demuestra la existencia de los agujeros negros y todas sus respectivas consecuencias. Esto ha sido posible gracias a un algoritmo para la reconstrucción de las imágenes de la sombra del agujero negro. Podemos deducir que gracias a este algoritmo ha habido un avance muy importante de la ciencia y esto muestra el poder que tienen estos algoritmos de recuperación, ya que hay diferentes tipos de algoritmos y pueden ayudar a la sociedad a realizar avances muy relevantes en todos los ámbitos. Se puede utilizar para las predicciones meteorológicas, construcción de imágenes que pueden ser de tomografías, como de astronomía, recuperación de propiedades de un material, estimación de flujos, problemas geofísicos, ingeniería petrolera,... Con lo cual este Trabajo Fin de Grado trata sobre el "Algoritmos numéricos de reconstrucción de conductancias", en específico, sobre la programación en MATLAB del algoritmo de recuperación de conductancias para redes Spiders (o arañas). Este algoritmo como otros algoritmos para retículas de 2D o 3D están muy desarrollados a nivel teórico, pero falta mucho camino para llegar a implementarlos y un primer paso a realizar es la implementación de algoritmo para una red Spider en código. Este documento contiene el proceso realizado para implementar este algoritmo.

3.1 Motivación

Mi principal motivación para este TFG es el hecho de querer profundizar más en algún área de las matemáticas. Las matemáticas son una ciencia exacta en la que se parte de demostraciones lógicas y como consecuencia, uno tiene la certeza de que aquello es correcto, además que es una ciencia abstracta. También me gusta la programación ya que es un procedimiento lógico y a la vez creativo porque puedes conseguir una meta empleando diferentes caminos con lo que no hay unas normas que determinen cómo tienes que resolver un problema. Si combinamos las matemáticas con la programación y se le añade el "detalle" que se puede aplicar en cualquier campo y que tiene un potencial muy elevado entonces, es un buen trabajo de fin de grado para dedicar el tiempo necesario para mostrar a los demás este potencial. A priori, sabía que quería hacer algo relacionado con las matemáticas ya que es un campo que me apasiona, pero al entender lo que querían que hiciera Ángeles y Andrés, supe que era mi TFG.

3.2 Objetivos

Para llevar a cabo este TFG ha sido necesario proponerse unos objetivos para marcar la metodología de éste mismo:

- Entender los conceptos básicos de física y matemáticas que son necesarios para el desarrollo del problema directo.
- Entender qué es el problema directo y sus aplicaciones.
- Aplicar el problema directo en ejemplos y comprender la resolución de éstos.
- Entender qué es el problema inverso y sus aplicaciones.

- Asimilar los artículos recomendados por los tutores.
- Aprender a programar con Matlab y elaborar el algoritmo en MATLAB. Además, aplicar este algoritmo en diferentes casos y verificar su funcionamiento.
- Conocer el concepto de la función de Green y el algoritmo de la obtención de ésta. Seguidamente, programar y verificar el programa con un ejemplo.
- Finalmente obtener una visión global del problema directo-inverso y cómo se relacionan con la función de Green y el Laplaciano.

3.3 Alcance del proyecto

El alcance de este proyecto está acotado mediante los objetivos propuestos, es decir, la implementación de los algoritmos teóricos para la recuperación de conductancias para una Spider y la función de Green de un cilindro y una Spider. No obstante, se ha podido ampliar el proyecto y introducir la Aplicación Dirichlet y su programación en Código.

Este recorrido está desarrollado a lo largo de este documento.

4 El método directo

4.1 El método directo en un entorno continuo

El método directo[**15**] es un procedimiento por el que conociendo las características de un objeto o su funcionamiento y aplicándoles unas condiciones de entrada se obtiene unos resultados llamados valores de salida.

Para desarrollar este método se le atribuirá un significado físico para entender los conceptos que representan sus formulaciones matemáticas respectivas.

Por ello se considera un cuerpo formado por cierto material que ocupa un dominio acotado en tres dimensiones, cuyo interior es Ω y cuya frontera es Γ. Entonces se entiende el calor como una forma de energía asociada a la agitación térmica de las moléculas del material. El calor es protagonista de tres procesos: producción, almacenamiento y transporte y están ligados por el principio de conservación:

$$Q_P = Q_A + Q_T \tag{1}$$

- Q_P: el calor total producido en el volumen V en $[t_1, t_2]$ viene dado por:

$$Q_P = \int_{t_1}^{t_2} \int_V f(x, y, z, t)\, dxdydzdt \tag{2}$$

  Donde $f(x, y, z, t)$ representa el calor neto producido por unidad de volumen y unidad de tiempo.

- Q_A: el calor almacenado en un volumen se expresa de la siguiente forma:

$$Q_A = \int_V [e(x, y, z, t_2) - e(x, y, z, t_1)]\, dxdydz = \int_{t_1}^{t_2} \int_V \frac{\partial e}{\partial t}(x, y, z, t)\, dxdydzdt \tag{3}$$

  Donde $e(x, y, z, t)$ es una función que representa la densidad de calor instantánea en cada punto.

- Q_T: la cantidad de calor que atraviesa una superficie S en $[t_1, t_2]$ es:

$$Q_T = \int_{t_1}^{t_2} \int_S < \phi, n > \, dsdt = \int_{t_1}^{t_2} \int_V \operatorname{div} \phi \, dxdydzdt \tag{4}$$

  ϕ es el vector flujo que es el campo vectorial cuyas componentes en la base cartesiana $\{ i, j, k\}$ son:

$$\phi(x, y, z, t) = \phi_1(x, y, z, t)i + \phi_2(x, y, z, t)j + \phi_3(x, y, z, t)k = \alpha n + \beta_1 t_1 + \beta_2 t_2 \tag{5}$$

Una vez desarrollados los diferentes tipos de calor se obtienen la siguiente ecuación:

$$\frac{\partial e}{\partial t}(x, y, z, t) + \operatorname{div} \phi(x, y, z, t) = f(x, y, z, t) \tag{6}$$

Donde e y ϕ son funciones desconocidas y se reducirán a una sola incógnita a través de la temperatura:

- Almacenamiento de calor (en relación con temperatura) : El calor almacenado de un cuerpo depende del tipo de material, de la distribución de su masa y de la temperatura.

$$\frac{\partial e}{\partial t}(x,y,z,t) = \rho(x,y,z)c(x,y,z)\frac{\partial u}{\partial t}(x,y,z,t) \tag{7}$$

  ρ > 0 es la densidad del material y se estima que no depende de la temperatura.
  c > 0 es el calor específico (capacidad de acumular calor) y se estima que no depende de la temperatura u.

- Transporte de calor mediante conducción (con relación a la temperatura) : Los dos mecanismos del transporte del calor son la conducción (el calor que se transfiere por el contacto entre moléculas vecinas) y la convección (el calor que se transporta por el movimiento de las moléculas). Se considera el transporte únicamente por conducción mediante la ley de Fourier que relaciona el vector flujo con la temperatura:

$$\phi(x,y,z,t) = -K(x,y,z)\nabla(u)(x,y,z,t) \tag{8}$$

  K: matriz de conductividad térmica simétrica y positiva
  $\nabla(u)$: es el vector temperatura.
  Con el desarrollo de las incógnitas iniciales en relación a la temperatura, la ecuación final abreviada es:

$$\rho c\frac{\partial u}{\partial t} - \text{div } \left[K\nabla(u)\right] = f \tag{9}$$

4.2 Las condiciones de contorno

Las ecuaciones desarrolladas anteriormente son generales para diferentes tipos de problemas y para distinguir el problema que se quiere estudiar de otro es necesario establecer un dominio acotado. Por ello se estudiará este tipo de condiciones de contorno. Cabe destacar que no se consideran las condiciones iniciales (condiciones en relación al tiempo) porque se supone que estamos en un régimen estacionario.

- Condición de contorno de Dirichlet: establece la temperatura de los puntos de la frontera en unos valores dados o impone desplazamientos en el contorno en el caso de una estructura.

$$u(x,y,z,t) = h(x,y,z,t) \text{ donde } (x,y,z)\ \epsilon\ \Gamma \tag{10}$$

- Condición de contorno de Neumann: se considera que se regula el flujo en los puntos de la frontera y se mantienen en unos valores deseados o se considera que el contorno tiene una tensión constante:

$$\frac{\partial u}{\partial \mathsf{n}_K}(x,y,z,t) = h_0(x,y,z,t) \quad \text{donde } (x,y,z)\ \epsilon\ \Gamma \tag{11}$$

Si $h_0 = 0$ y $h = 0$ significa que no hay flujo a través del contorno porque el dominio está aislado térmicamente, o en el caso de una cuerda, que sus extremos son libres.

- Condición de contorno de Robin (combinación lineal de las condiciones de Dirichlet y Neumann): Si se considera que estamos en un problema de un proceso ondulatorio, se supone que la cuerda está conectada a una masa que está unida a una base a través de un resorte de constante elástica.

  Si se considera que estamos en un problema de transferencia de calor, se supone que la frontera está en contacto con el medio con el que existe una transferencia de calor.

$$\frac{\partial u}{\partial \mathsf{n}_K}(x,y,z,t) = \alpha(x,y,z)\left[u(x,y,z,t) - u_{ext}(x,y,z,t)\right] \tag{12}$$

  U_{ext}:función que describe la temperatura de los puntos en contacto con el medio donde hay esta transferencia (en el caso de un problema térmico).

  Si se define: $h(x,y,z,t) = \alpha(x,y,z)\, u_{ext}(x,y,z,t)$, entonces se obtiene la siguiente condición de contorno:

$$\alpha(x,y,z)u(x,y,z,t) + \frac{\partial u}{\partial \mathsf{n}_K}(x,y,z,t) = h(x,y,z,t) \tag{13}$$

En conclusión, los problemas de evolución estacionarios corresponden a la siguiente formulación:

$$\begin{aligned} -\text{div}\,\left[K\nabla(u)\right] + qu &= F \quad en\,\Omega, \\ hu - \tfrac{\partial u}{\partial \mathsf{n}_K} &= g_1 \quad en\,\Gamma_1, \\ u &= g_0 \quad en\,\Gamma_0, \end{aligned} \tag{14}$$

donde:

g_0 es una función definida sobre Γ_0

h y g_1 son funciones definidas sobre Γ_1

Se consideran que los conjuntos Γ_0 y Γ_1 tales que $\Gamma_0 \cap \Gamma_1$ =0 y $\Gamma_0 \cup \Gamma_1 = \Gamma$

4.3 La noción del problema bien planteado

Se dice que un problema definido por una ecuación diferencial en derivadas parciales y unas condiciones de contorno está bien planeado si se cumplen los siguientes requisitos:

- Existencia de solución para cualesquiera datos del problema suficientemente regulares.

- Unicidad de solución.
- Dependencia continua de la solución con respecto a los datos del problema.

Los datos del problema son:

- La función F que modela acciones aplicadas en el interior del sistema.
- La función g representa acciones impuestas en el contorno del sistema.
- La función f representa el estado inicial concreto del sistema.

Dadas las funciones F, f y g con ciertas condiciones de regularidad se quiere garantizar una función u y sólo una que satisfaga la ecuación diferencial y las condiciones de contorno.

5 La noción de solución

Se busca la existencia de soluciones del tipo de problemas planteados en (15) de evolución estacionarios. Por ello, se desarrolla un procedimiento para poder hallar el valor de esta solución.

$$\begin{array}{cc} -\text{div}\left[K\nabla(u)\right] + qu = F & en\,\Omega, \\ hu + < K\nabla(u), N >= g_1 & en\,\Gamma_1, \\ u = g_0 & en\,\Gamma_0, \end{array} \tag{15}$$

Si $v\,\epsilon\,C^2\,(\Omega)$ y el valor de v en $\Gamma_0 = 0$, entonces:

$$\begin{array}{cc} -\text{div}\left[K\nabla(u)\right] v + quv = Fv & en\,\Omega, \\ huv - \frac{\partial u}{\partial \mathsf{n}_K} v = g_1 v & en\,\Gamma_1, \\ uv = g_0 v & en\,\Gamma_0, \end{array} \tag{16}$$

A la vez se puede integrar en ambos lados y la ecuación seguiría siendo válida:

$$\int_\Omega Fv = -\int_\Omega \text{div}\,(K\nabla(u))v + \int_\Omega quv \tag{17}$$

Para poder desglosar la equación (17), es necesario hacer un paréntesis con conceptos previos:

$$\text{div}\,(vF) = v\text{div}\,(F) + < \nabla(v), F > \tag{18}$$

$$-\,\text{div}\,(K\nabla(u))v = -\text{div}\,(vK\nabla(u)) + < \nabla(v), K\nabla(u) > \tag{19}$$

$$\int_\Omega -\text{div}\,(K\nabla(u))v = -\int_\Omega \text{div}\,(vK\nabla(u)) + \int_\Omega < \nabla(v), K\nabla(u) > \tag{20}$$

$$\int_\Omega -\text{div}\,(K\nabla(u))v = -\int_{\Gamma_0\cup\Gamma_1} < (vK\nabla(u)), N > + \int_\Omega < \nabla(v), K\nabla(u) > \tag{21}$$

$$\int_\Omega -\text{div}\,(K\nabla(u))v = \int_\Omega < \nabla(v), K\nabla(u) > - \int_{\Gamma_0\cup\Gamma_1} \frac{\partial u}{\partial \mathsf{n}_K} v \tag{22}$$

Seguidamente, se procederá a sustituir la ecuación (22) en la (17):

$$\int_\Omega fv = \int_\Omega < \nabla(v), K\nabla(u) > - \int_{\Gamma_0} \frac{\partial u}{\partial \mathsf{n}_K} v - \int_{\Gamma_1} \frac{\partial u}{\partial \mathsf{n}_K} v + \int_\Omega quv \tag{23}$$

Como el valor de v en Γ_0 es 0, se elimina su respectiva integral. También se sabe que: $huv - \frac{\partial u}{\partial \mathsf{n}_K} v = g_1 v$ de la ecuación (16) con la que se podrá relacionar con la ecuación (23):

$$\int_\Omega fv = \int_\Omega < \nabla(v), K\nabla(u) > - \int_{\Gamma_1} huv - \int_{\Gamma_1} g_1 v + \int_\Omega quv \tag{24}$$

Finalmente, se obtiene la siguiente formulación:

$$\int_\Omega < \nabla(v), K\nabla(u) > + \int_\Omega quv + \int_{\Gamma_1} huv = \int_\Omega fv + \int_{\Gamma_1} g_1 v \text{ para todo } v = 0 \text{ en } \Gamma_0 \qquad (25)$$

Esta expresión se denomina Formulación débil del problema de evolución inicial y por lo tanto una función $u \epsilon C^1(I \times \overline{\Omega})$ será denominada solución débil del problema si satisface la identidad (25) para cada $v \epsilon C^1$ $(\overline{\Omega})$ tal que $v = 0$ en Γ_0.
Nota: $\overline{\Omega} = \Omega \cup \Gamma$

Si u es una solución débil tal que además $u \epsilon C^2(I \times \bar{\Omega})$, entonces es una solución clásica.

Cuando la matriz de rigidez es definida positiva a partir de la función cuadrática se introduce el funcional de energía:

$$\int_\Omega < \nabla(u), K\nabla(u) > + \int_\Omega qu^2 + \int_{\Gamma_1} hu^2 - 2\int_\Omega fv - 2\int_{\Gamma_1} g_1 u = 0 \text{ para todo } v = 0 \text{ en } \Gamma_0 \quad (26)$$

La solución de este funcional se basa en obtener u con la mínima energía posible. Por ello, se introducirá la minimización de Funcionales cuadráticos.

6 Funcionales cuadráticos

En este capítulo se desarrollará la teoría relativa a la minización de Funcionales cuadráticos donde se trabajará con espacios vectoriales reales, es decir donde el cuerpo base es $\mathbb{R}$.

Un subconjunto $C \subset V$ se denomina *convexo* si para cada $u, v \in C$ se tiene que $tv + (1-t)u \in C$ para cada $t \in [0,1]$. (Se puede observar que $tv + (1-t)u$, $t \in [0,1]$ describe el segmento que une los vectores u y v). Un tipo especial de conjuntos convexos en un espacio vectorial son los *subespacios afines* que son los conjuntos de la forma:

$$C = u_0 + W = \Big\{ u_0 + v : v \in W \Big\}, \quad \text{donde } W \subset V \text{ subespacio vectorial.} \tag{27}$$

En esta situación, W se denomina variedad lineal, o subespacio vectorial, de C.

Se puede observar que si $C = u_0 + W$, entonces para cada $u \in C$ se tiene que $C = u + W$. En particular $C = W$ (y es por tanto un subespacio vectorial) si y sólo si $u_0 \in W$ y esto ocurre si y sólo si $0 \in C$. Por tanto, los subespacios afines de V que pasan por 0 son los subespacios vectoriales de V. El interés de este trabajo estará centrado en los subespacios afines de un espacio vectorial.

Si V es un espacio vectorial real, se considerará los siguientes espacios de funciones lineales asociados con él:

1) V^*, su *espacio Dual*; es decir al espacio vectorial de las *formas lineales* $\ell : V \to \mathbb{R}$ sobre V.

2) $\mathscr{L}(V)$, el espacio de endomorfismos en V; es decir el espacio vectorial de las *aplicaciones lineales* $F : V \to V$.

3) $\mathscr{B}(V)$, el conjunto de *aplicaciones bilineales* sobre V

Se sabe que $b \in \mathscr{B}(V)$ si y sólo si $b : V \times V \to \mathbb{R}$ satisface que es lineal en ambas variables, o equivalentemente que

$$\begin{aligned} b(\alpha u + \beta v, \gamma z + \delta w) &= \alpha\gamma b(u,z) + \beta\gamma b(v,z) + \alpha\delta b(u,w) + \beta\delta b(v,w) \\ &= \begin{bmatrix} \alpha & \beta \end{bmatrix} \begin{bmatrix} b(u,z) & b(u,w) \\ b(v,z) & b(v,w) \end{bmatrix} \begin{bmatrix} \gamma \\ \delta \end{bmatrix} \end{aligned} \tag{28}$$

para cada $u, v, z, w \in V$ y cada $\alpha, \beta, \gamma, \delta \in \mathbb{R}$. Si $b \in \mathscr{B}(V)$, entonces para cada $u \in V$ se tiene que $b(u,\cdot), b(\cdot,u) \in V^*$.

Si $b \in \mathscr{B}(V)$, se dirá que b es *simétrica* si $b(v,w) = b(w,v)$, para cada $v, w \in V$; mientras que se dirá que b es *antisimétrica* o *alternante* si $b(v,w) = -b(w,v)$ para cada $v, w \in V$. Es claro que una forma bilineal no tiene porqué ser ni simétrica ni antisimétrica, pero cada forma bilineal $b \in \mathscr{B}(V)$ se puede expresar de manera única como suma de una forma bilineal simétrica b_s y otra antisimétrica, b_a:

$$b(u,v) = b_s(u,v) + b_a(u,v), \quad \text{donde } b_s = \frac{1}{2}\big(b(u,v) + b(v,u)\big) \text{ y } b_a = \frac{1}{2}\big(b(u,v) - b(v,u)\big) \tag{29}$$

El objetivo es el estudio de las denominadas *Formas cuadráticas en* V. Una aplicación $q : V \to \mathbb{R}$ se denomina *forma cuadrática sobre* V si satisface que

1) $q(tu) = t^2 q(u)$, para cada $t \in \mathbb{R}$ y cada $u \in V$.

2) Existe $b \in \mathscr{B}(V)$ tal que $q(u) = b(u, u)$, para cada $u \in V$.

La segunda condición implica la primera. Sin embargo es necesario introducirla, porque existen aplicaciones que satisfacen la primera y no provienen de una forma bilineal. Por ejemplo, si para cada $p > 0$ se considerará $q_p : \mathbb{R}^2 \to \mathbb{R}$ definida como

$$q_p(x, y) = \left(|x|^p + |y|^p\right)^{\frac{2}{p}} \tag{30}$$

entonces $q_p(tx, ty) = t^2 q_p(x, y)$, pero se verá posteriormente que sólo es una forma cuadrática cuando $p = 2$.

Se observa que la forma cuadrática asociada a cada forma bilineal antisimétrica es nula, puesto que si $b \in \mathscr{B}(V)$ es antisimétrica, entonces $q(u) = b(u, u) = -b(u, u) = -q(u)$ y necesariamente $q(u) = 0$. Por tanto, si $b \in \mathscr{B}(V)$ es una bilineal arbitraria, entonces la forma cuadrática asociada a b coincide con la forma cuadrática asociada a su parte simétrica y por tanto, debe satisfacer la denominada *Identidad del Paralelogramo:*

$$q(v + w) + q(v - w) = 2q(v) + 2q(w), \quad \text{para cada} \quad x, y \in V. \tag{31}$$

Resulta entonces que $q : V \to R$ es una forma cuadrática si y sólo si satisface la identidad de polarización, y entonces, la única forma bilineal simétrica $b \in \mathscr{B}(V)$ que satisface que $q(u) = b(u, u)$, para cada $u \in V$, es la determinada por las denominadas *Identidades de Polarización:*

$$b(v, w) = \frac{1}{2}\big[q(v + w) - q(v) - q(w)\big] = \frac{1}{4}\big[q(v + w) - q(v - w)\big].$$

Por ello a esa única bilineal simétrica determinada por q se la denomina *forma polar asociada a* q.

Nota: Si se toma $u = (1, 0)$ y $v = (0, 1)$, entonces para cada $p > 0$ se tiene que

$$q_p(u) = q_p(v) = 1, \quad q_p(u + v) = q_p(u - v) = 2^{\frac{2}{p}}$$

y por tanto q_p satisface la identidad del paralelogramo sólo si $2 = 2^{\frac{2}{p}}$; es decir sólo si $p = 2$. Es sencillo comprobar que en este caso, $p = 2$ se satisface la identidad.

Si $b \in \mathscr{B}(V)$ es una forma bilineal simétrica y q es su forma cuadrática asociada, se denominan *isótropos* de b, o de q, a los $v \in V$ tales que $q(v) = 0$. Es claro que $v = 0$ es isótropo para cada forma cuadrática. Se considera que b, o q, es semidefinida positiva si $q(v) \geq 0$ para cada $v \in V$ y semidefinida negativa si $q(v) \leq 0$ para cada $v \in V$. En particular, q es definida positiva; respectivamente definida negativa, si es semidefinida positiva, respectivamente negativa, y su único elemento isótropo es el cero; es decir $q(v) > 0$ para cada $v \neq 0$, respectivamente $q(v) < 0$ para cada $v \neq 0$.

Si q es semidefinida positiva, y b es su forma polar, entonces se satisface la *desigualdad de Cauchy-Schwarz*:

$$|b(u,v)| \leq q(u)q(v), \quad \text{para cada } u, v \in V.$$

Por tanto, $u \in V$ es isótropo si y sólo si $b(u,v) = 0$ para cada $v \in V$ y como consecuencia el conjunto de vectores isótropos es un subespacio vectorial de V.

Observar que $b \in \mathscr{B}(V)$ es semidefinida negativa si y sólo si $-b$ es semidefinida positiva, de manera que la desigualdad de Cauchy-Schwarz y sus consecuencias son asimismo válidas para formas bilineales semidefinidas negativas.

6.1 Minimización de Funcionales cuadráticos

Un *Funcional cuadrático* en un espacio vectorial real V, es una aplicación de la forma $\mathcal{J} : V \to R$ definida por

$$\mathcal{J}(u) = q(u) - 2\ell(u), \quad u \in V \tag{32}$$

donde q es una forma cuadrática y $\ell \in V^*$ es un funcional lineal.

Observar que para cada $u, v \in V$ tenemos que

$$\begin{gathered}\mathcal{J}(u+v) = q(u+v) - 2\ell(u+v) = q(u) + q(v) + 2b(u,v) - 2\ell(u) - 2\ell(v) = \\ \mathcal{J}(u) + \mathcal{J}(v) + 2b(u,v) = \mathcal{J}(u) + \widehat{\mathcal{J}}(v),\end{gathered} \tag{33}$$

donde $\widehat{\mathcal{J}}(v) = q(v) - 2\hat{\ell}(v)$ y $\hat{\ell}(v) = \ell(v) - b(u,v)$.

La intención es plantear el siguiente problema de optimización:

Dados $\mathcal{J}$ un funcional cuadrático y C un subespacio afín de variedad lineal W, hallar

$$\min_{u \in C} \Big\{ \mathcal{J}(u) \Big\}$$

Como C es un subespacio afín de V y W es su variedad lineal, se sabe que existe $u_0 \in V$ tal que $C = u_0 + W$ y por tanto

$$\min_{u \in C} \Big\{ \mathcal{J}(u) \Big\} = \min_{v \in W} \Big\{ \mathcal{J}(u_0 + v) \Big\} = \mathcal{J}(u_0) + \min_{v \in W} \Big\{ \widehat{\mathcal{J}}(v) \Big\} \tag{34}$$

así que la realidad el problema de optimización se produce sobre el subespacio W.

Lo primero que se deduce del problema anterior es que para que tenga solución q debe ser semidefinida positiva sobre W: Se supone que existe $v \in W$ tal que $q(v) < 0$. Entonces

$$\mathcal{J}(u_0 + tv) = \mathcal{J}(u_0) + t^2 q(v) - 2t\big[\ell(v) - b(u_0, v)\big] \tag{35}$$

y claramente, $\lim_{t \to \pm\infty} \mathcal{J}(u_0 + tv) = -\infty$, de manera que $\mathcal{J}$ no tiene mínimo sobre C. Así pues, para que el problema de minimización planteado tenga sentido, es necesario que $\mathcal{J}$ sea semidefinida positiva sobre W.

El resultado fundamental en esta sección es el siguiente:

Si la bilineal b, o la forma cuadrática q, es semidefinida positiva sobre W, entonces $u \in C$ minimiza el funcional cuadrátrico $\mathcal{J}$ sobre C si y sólo si satisface la *Ecuación de Euler*:

$$b(u, v) = \ell(v), \quad \text{para cada } v \in W. \tag{36}$$

Se supone que $u \in C$ minimiza $\mathcal{J}$ sobre C. Entonces para cada $v \in W$ se tiene que la función φ_v definida como

$$\varphi_v(t) = \mathcal{J}(u + tv) = \mathcal{J}(u) + t^2 q(v) - 2t\big[\ell(v) - b(u, v)\big] \tag{37}$$

tiene un mínimo en $t = 0$ y por tanto $0 = \varphi_v'(0) = \Big[2tq(v)) - 2\big[\ell(v) - b(u, v)\big]\Big]_{t=0}$, de manera que u satisface que $b(u, v) = \ell(v)$.

Recíprocamente, si u satisface la Ecuación de Euler; es decir, $b(u, v) = \ell(v)$ para cada $v \in W$, entonces si $v \in W$ resulta que

$$\mathcal{J}(u + v) = \mathcal{J}(u) + q(v) \underbrace{-2\ell(v) + 2b(u, v)}_{0} = \mathcal{J}(u) + \underbrace{q(v)}_{\geq 0} \geq \mathcal{J}(u). \tag{38}$$

Como $C = u + W$, la anterior desigualdad implica que u es un mínimo de $\mathcal{J}$ sobre C.

Observar también que si $u \in C$ minimiza $\mathcal{J}$ sobre W, entonces dado $v \in V$ se tiene $\mathcal{J}(u + v) = \mathcal{J}(u)$; es decir que $u+v$ también minimiza $\mathcal{J}$, si y sólo $q(v) = 0$; es decir si y sólo si v es isótropo. Por tanto la unicidad de mínimo sólo es posible si no existe más elemento isótropo en W que el 0; es decir si y sólo si q es definida positiva sobre W.

Se advierte que a la misma conclusión se puede llegar desde la ecuación de Euler: Si $u \in C$ satisface que $b(u, v) = \ell(v)$, para cada $v \in W$ y $z \in W$ es isótropo, entonces $b(z, v) = 0$ para cada $v \in V$ y por tanto

$$b(u + z, v) = b(u, v) + b(z, v) = b(u, v) = \ell(v), \quad \text{para cada } v \in V \tag{39}$$

de manera que $u + z$ también satisface la identidad de Euler, y por tanto minimiza $\mathcal{J}$ sobre W.

Se puede observar que se ha establecido la equivalencia entre minimizar el funcional cuadrático $\mathcal{J}$ y satisfacer la ecuación de Euler correspondiente. Sin embargo, aún falta constituir las condiciones para la existencia de mínimo, lo que en general deberá relacionarse por métodos alternativos, relativos a propiedades de sistemas lineales, en el caso de que el espacio ambiente sea de dimensión finita o a la aplicación de propiedades provenientes del análisi funcional en el caso infinito dimensional. No obstante, lo que sí se asegura es que la ecuación de Euler instaura condiciones sobre el funcional lineal ℓ para que exista solución: Si existe mínimo de $\mathcal{J}$ sobre W, cada mínimo debe satisfacer la identidad de Euler $b(u, v) = \ell(v)$. Como $C = u_0 + W$, necesariamente $u = u_0 + v_0$ donde $v_0 \in V$ y por tanto la ecuación de Euler puede expresarse como

$$b(u_0, v) + b(v_0, v) = \ell(v), \quad \text{para cada } v \in W. \tag{40}$$

En particular, si z es un vector isótropo de W, debe satisfacerse que

$$\ell(z) = b(u_0, z) + \underbrace{b(v_0, z)}_{0} = b(u_0, z). \tag{41}$$

Por tanto, una condición necesaria para que exista mínimo de $\mathcal{J}$ sobre W es que se satisfaga la denominada *Hipótesis de compatibilidad*

$$b(u_0, z) = \ell(z), \quad \text{para cada } z \in W \text{ tal que } q(z) = 0. \tag{42}$$

Observar que la condición de compatibilidad se reduce a una identidad trivial si $z = 0$. Así pues, la presencia de vectores isótropos establecen condiciones necesarias que ha de satisfacer el funcional lineal ℓ para que exista solución. Demostrar que estas condiciones de compatibilidad, que están relacionadas con la *Alternativa de Fredholm* son suficientes para asegurar la existencia de mínimo requerirán un análisis más profundo.

6.2 El caso de dimensión finita

Se supone ahora que $\dim W = m$ y se considera $\{w_1, \ldots, w_m\}$ una base de W. Entonces

$$C = u_0 + W = \left\{u_0 + x_1 w_1 + \cdots x_m w_m : x_1, \ldots, x_m \in \mathbb{R}\right\}. \tag{43}$$

Si para cada $j = 1, \ldots, m$ se define $b_j = \ell(w_j) - b(u_0, w_j)$ y para cada $i, j = 1, \ldots, m$, $a_{ij} = b(w_i, w_j)$, entonces $u = u_0 + x_1 w_1 + \cdots + x_m w_m$ satisface las ecuaciones de Euler si y sólo si

$$\begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \tag{44}$$

sistema de ecuaciones que reciben el nombre de *Ecuaciones de Euler-Lagrange*. La matriz de coeficientes del sistema anterior es simétrica y semidefinida positiva. Por otra parte, $v = y_1 w_1 + \cdots + y_m w_m \in W$ es isótropo si y sólo si

$$\begin{bmatrix} y_1 & \cdots y_m \end{bmatrix} \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mm} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = 0 \tag{45}$$

de manera que las condiciones de compatibilidad se expresan como

$$y_1 b_1 + \cdots + y_m b_m = 0, \quad \text{para cada } (y_1, \ldots, y_m) \text{ isótropo.}$$

7 El problema inverso

Los problemas inversos intentan describir las propiedades o la composición del interior de un objeto mediante un estímulo en la parte exterior del objeto. Un ejemplo claro de este problema en la vida cotidiana es cuando se toca la sandía o se pica para ver si la sandía está buena o no. El procedimiento que se emplea es que a partir de picar la sandía (estímulo externo) y escuchar el ruido producido (la respuesta) se predice cómo está la sandía (característica interior de la fruta).

El origen de la formulación de estos problemas cabe situarlo en la década de los 50 del pasado siglo, cuando el matemático argentino-estadounidense A. P. Calderón trabajaba como ingeniero civil en YPF y quería saber si había yacimientos petrolíferos bajo el mar por medio de mediciones superficiales. No obstante, habría que esperar hasta 1980 para que el problema apareciera por primera vez formulado en una publicación científica, [**3**]. En honor a Calderón, en la actualidad el problema inverso descrito es conocido como Problema de Calderón.

Desde el punto de vista matemático, la clave para la resolución del problema de Calderón es la denominada, aplicación Dirichlet-a-Neumann que es el operador que permite obtener la información en el interior del cuerpo a partir de las mediciones en su frontera. Este tipo de problemas se han estudio en medios discretos [**6**, **7**, **8**, **9**, **10**] y por ello se trabaja con redes que contienen la información y que se mostraran más adelante.
Curtis, Ingerman y Morrow fueron quienes estudiaron las condiciones de la reconstrucción sobre redes circulares planares [**11**, **12**, **13**].

Hoy en día, este tipo de problemas se están estudiando y avanzando en su desarrollo en el departamento de Matemáticas de la EEBE por diferentes profesores incluyendo mis tutores para poder aplicarlo a nivel real.
Una aplicación destacable es la Tomografía de Impedancia Eléctrica (TIE) que es una prueba que permite observar el interior del organismo de un paciente a partir de la información proporcionada por electrodos aplicados sobre la piel del mismo.

Para poder profundizar sobre cómo funciona el problema inverso, se tratará el artículo ***Overdetermined partial boundary value problems on finite networks*** [**1**] de los autores: C. Araúz, A. Carmona y A.M. Encinas. Con este artículo, se introducirán los conceptos previos y la metodología que se emplea para recuperar las conductancias de una red Spider.

7.1 Preliminares

Sea $\Gamma = (V, c)$ una red finita con los vértices en V y E el conjunto de aristas de la red de Γ donde cada arista (x, y) tiene asignado un número positivo denominado conductancia $c(x, y)$. Esta denominación proviene de la analogía de una red con un circuito constituido únicamente por resistencias. La conductancia tiene el significado físico de inverso de la resistencia de cada arista. Así, podemos identificar esta asignación de valores como una aplicación donde $c : V \times V \to [0, +\infty)$. Además, $c(y, x) = c(x, y)$ y $c(x, y) = 0$ si $(x, y) \notin E$. También $x, y \in V$ es adyacente $x \sim y$ si $c(x, y) > 0$.
Si se considera un subconjunto $F \subset V$, entonces su frontera $\delta(F)$ está dada por los vértices de

$V \setminus F$ que son adyacentes a al menos un vértice de F. Los vértices de $\delta(F)$ son llamados los vértices del contorno.

Una función $\omega \in C^+(V)$ con supp(ω) = $\bar{F}$ (conjunto de puntos donde la función no es nula) tal que $\int_{\bar{F}} \omega^2(x)dx = 1$ se denomina peso en $\bar{F}$. El conjunto de pesos es dotado por $\Omega(\bar{F})$.

Se indica $K_F \in C(\delta(F))$ como la función $K_F(x) = \sum_{y \in F} c(x, y)$.

La derivada normal de $u \in C(\bar{F})$ en F es la función en $C(\delta(F))$ que está dada por:

$$\frac{\partial u}{\partial \mathsf{n}_F}(x) = \int_F c(x, y)(u(x) - u(y))dy \quad \text{por cualquier } x \in \delta(F). \tag{46}$$

Cualquier función $K \in C(F \times F)$ es llamada núcleo en F. Además, una función $u \in C(F)$ se puede identificar con el núcleo $K(x, x) = u(x)$ y $K(x, y) = 0$, y la matriz diagonal se escribe como D_u.

El Laplaciano combinatorio de Γ es el operador lineal $\mathscr{L} : C(V) \to C(V)$ que asigna a cada $u \in C(V)$ la función definida para todo $x \in V$ como

$$\mathscr{L}(u)(x) = \int_V c(x, y)(u(x) - u(y))dy. \tag{47}$$

Dado $q \in C(V)$, el operador de Schrödinger en Γ con potencial q es el operador lineal $\mathscr{L} : C(V) \to C(V)$ que asigna para cada $u \in C(V)$, la función $\mathscr{L}_q(u) = \mathscr{L}(u) + qu$. Mientras q sea real, el operador de Schrödinger es autoadjunto.

La relación entre los valores del operador Schrödinger con potencial q en un conjunto conexo $F \subseteq V$ y los valores de la derivada normal en $\delta(F)$ está dada por la Primera Identidad de Green:

$$\begin{aligned} \int_F v(x)\mathscr{L}_q(u)(x)dx = \frac{1}{2}\int_{\bar{F}}\int_{\bar{F}} c_F(x, y)(u(x) - u(y))(v(x) - v(y))dxdy + \\ \int_F q(x)u(x)v(x)dx - \int_{\delta(F)} v(x)\frac{\partial u}{\partial \mathsf{n}_F}(x)dx. \end{aligned} \tag{48}$$

donde $u, v \in C(\bar{F})$ y $c_F = c \cdot \mathcal{X}_{\bar{F}\times\bar{F}\setminus(\delta(F)\times\delta(F))}$

Una consecuencia directa de la primera identidad es la Segunda Identidad de Green para cada $u, v \in C(\bar{F})$:

$$\int_F (v(x)\mathscr{L}_q(u)(x) - (u(x)\mathscr{L}_q(v)(x))dx = \int_{\delta(F)} (u(x)\frac{\partial(v)}{\partial \mathsf{n}_F} - v(x)\frac{\partial u}{\partial \mathsf{n}_F}(x))dx. \tag{49}$$

Se define la energía asociada a F y q como la forma bilineal simétrica $\mathcal{E}_q^F : C(\bar{F}) \times C(\bar{F}) \to \mathbb{R}$ que es dada por cualquier $u, v \in C(\bar{F})$ por:

$$\mathcal{E}_q^F(u,v) = \frac{1}{2}\int_{\bar{F}}\int_{\bar{F}} c_F(x,y)(u(x)-u(y))(v(x)-v(y))dxdy + \int_{\bar{F}} q(x)u(x)v(x)dx. \tag{50}$$

A partir de la Primera Identidad de Green, por cualquier $u, v \in C(\bar{F})$, se obtiene que:

$$\mathcal{E}_q^F(u,v) = \int_F v(x)\mathscr{L}_q(u)(x)dx + \int_{\delta(F)} v(x)\left[\frac{\partial u}{\partial \mathsf{n}_F}(x) + q(x)u(x)\right]dx = 0. \tag{51}$$

Por cualquier peso $\sigma \in \Omega(\bar{F})$, el potencial asociado con σ es la función en $C(\bar{F})$ definida como $q_\sigma = -\sigma^{-1}\mathscr{L}(\sigma)$ en F, $q_\sigma = -\sigma^{-1}\frac{\partial(\sigma)}{\partial \mathsf{n}_F}$ en $\delta(F)$. Este tipo de potenciales fueron introducidos también en el caso continuo por Calderón **[3]**.
Asimismo, de ahora en adelante supondremos que se satisface $q = q_\sigma + \lambda\mathcal{X}_{\delta(\bar{F})}$ con $\lambda \geq 0$.

La energía es semidefinida positiva en $C(\bar{F})$ si $\lambda \geq 0$ y existe $\sigma \in \Omega(\bar{F})$ tal que $q = q_\sigma + \lambda\mathcal{X}_{\delta(\bar{F})}$. En este caso, es definida positiva si $\lambda > 0$. De esta forma, por cualquier $g \in C(\delta(F))$ el problema de Dirichlet: $\mathscr{L}_q(u) = 0$ en F y $u = g$ en $\delta(F)$ tiene una única solución u_g.

La aplicación $\Lambda_q : C(\delta(F)) \to C(\delta(F))$ que asigna a cualquier función $g \in C(\delta(F))$, la función $\Lambda_q(g) = \frac{\partial u_g}{\partial \mathsf{n}_F} + qg$ es llamada aplicación de Dirichlet-to-Robin. Esta aplicación es autoadjunta y semidefinida positiva cuya forma cuadrática asociada está dada por:

$$\int \delta(F)\, g(x)\Lambda_q(g)(x)dx = \mathcal{E}_q^F(u_g, u_g) \tag{52}$$

Además, λ es el valor propio más pequeño de Λ_q y sus funciones propias asociadas son múltiplos de σ. Asimismo, si $N_q \in C(\delta(F))$ y $C(\delta(F)) \to \mathbb{R}$ es el núcleo de Λ_q, su asociada matriz N_q es una matriz simétrica e irreductible.
En general, N_q es llamada la matriz de respuesta de la red. Dados $A, B \in \delta(F)$, un par de subconjuntos disjuntos (subconjuntos con intersección vacía), se considera la submatriz de la matriz respuesta $N_q(A;B) = N_q(x,y)_{(x,y)\in A\times B}$.

La aplicación Dirichlet-to-Robin se basa en un método donde a un problema se le aplica las condiciones de contorno de Dirichlet y se encuentra una única solución. A esta solución se le aplica la derivada normal.

7.2 Problemas de contornos sobredeterminados

Se considera un subconjunto conexo $F \subset V$ y A, $B \subset \delta(F)$ subconjuntos no vacíos tal que $A \cap B = 0$. Además, se denota por $R = \delta(F) \setminus (A \cup B)$, así que $\delta(F) = A \cup B \cup R$ es una partición de $\delta(F)$.
Se considera un nuevo tipo de problema de contorno donde los valores de las funciones y de sus derivadas normales se conocen en la misma parte del contorno, lo que representa un problema sobredeterminado, mientras en otras partes del contorno no se sabe ningún dato. El caso límite es cuando $B = R = 0$, donde los valores de la función en el contorno es nula y el valor de la derivada normal es constante. Este problema se le conoce con el nombre de problema de Serrin

[4].
Para cualquier $f \in C(F), g \in C(A \cup R)$ y $h \in C(A)$, el problema de contorno parcial de Dirichlet – Neumann en F con datos f, g, h consiste en encontrar $u \in C(F)$ tal que

$$\mathscr{L}_q(u) = f \quad \text{en} \quad F, \quad \frac{\partial u}{\partial \mathsf{n}_F} = h \quad \text{en} \quad A \quad \text{y} \quad u = g \quad \text{en} \quad A \cup R.$$

Dado que los valores de u se conocen en A, la condición de contorno $\frac{\partial u}{\partial \mathsf{n}_F} = h$ es equivalente a $\frac{\partial u}{\partial \mathsf{n}_F} + qu = h + qg$.

El problema de contorno parcial Dirichlet – Neumann homogéneo en F consiste en encontrar $u \in C(\bar{F})$ tal que:

$$\mathscr{L}_q(u) = 0 \quad \text{en} \quad F, \quad \frac{\partial u}{\partial \mathsf{n}_F} = u = 0 \quad \text{en} \quad A \quad \text{y} \quad u = 0 \quad \text{en} \quad R.$$

Está claro que el conjunto de soluciones del problema del contorno homogéneo es un subespacio de $C(F \cup B)$, que se denota por ν_B. Asimismo, si el problema tiene solución y u es una solución particular, entonces $u + \nu_B$ describe el conjunto de las soluciones. Además, si u es una solución del problema, entonces por cualquier $x \in A$, se tiene que:

$$\int_F c(x,y)u(y)dy = g(x)K_F(x) - h(x) \tag{53}$$

Entonces, si u es una solución del problema homogéneo por cualquier $x \in A$, se tiene que:

$$\int_F c(x,y)u(y)dy = 0 \tag{54}$$

El problema adjunto del problema de contorno parcial Dirichlet – Neumann sobredeterminado en F viene dado por:

$$\mathscr{L}_q(v) = 0 \quad \text{en } F, \ \frac{\partial v}{\partial \mathsf{n}_F} = v = 0 \quad \text{en } B \text{ y } \ v = 0 \quad \text{en } R. \tag{55}$$

El subespacio de las soluciones al problema anterior es denotado por el ν_A. Está claro que $\nu_A \in C(F \cup A)$. Entonces, la Segunda Identidad de Green lleva al siguiente resultado

- **Proposición 3.1** Los problemas (54) y (55) son mútuamente adjuntos

$$\int_F v(x)(L)(u)(x)dx = \int_F u(x)(L)(v)(x)dx \tag{56}$$

  por cualquier $u, v \in C(\bar{F})$ tal que $\frac{\partial u}{\partial \mathsf{n}_F} = u = 0$ en A; $\frac{\partial (v)}{\partial \mathsf{n}_F} = v = 0$ en B y $u = v = 0$ en R.

- **Proposición 3.2** La alternativa de Fredholm consiste en que dado $f \in C(F)$, $g \in C(A \cup R)$, $h \in C(A)$, el problema de contorno es

$$\mathscr{L}_q(u) = f \quad \text{en } F, \frac{\partial u}{\partial \mathsf{n}_F} = h \quad \text{en } A \text{ y } \ u = g \ \text{ en } A \cup R \tag{57}$$

  tiene solución únicamente si

$$\int_F f(x)v(x)dx + \int_F h(x)v(x)dx = \int_{A \cup B} g(x)\frac{\partial u}{\partial \mathsf{n}_F}(x)dx \quad \text{por cada } v \in \nu_A \tag{58}$$

Además, cuando la condición anterior se cumple, existe una solución única del problema de contorno en $\nu_B^{\perp}$ tal que:

$$\int_{F\cup B} u(x)z(x)dx = 0 \quad \text{por cada } z \in \nu_B. \tag{59}$$

- **Observación 3.3** La alternativa Fredholm establece la siguiente formula donde se relacionan las dimensiones de los subconjuntos:

$$\dim \nu_A - \dim \nu_B = |A| - |B| \tag{60}$$

Además, la existencia de una solución para cualquier dato equivale a $\nu_A = 0$, es decir, $|B| - |A| = \dim \nu_B \geq 0$. La unicidad de las soluciones equivale a $|A| - |B| = \dim \nu_A \geq 0$. En particular, si $|A| = |B|$, la existencia de una solución al problema (55) para cualquier dato f, g y h es equivalente a la unicidad de la solución y por lo tanto es equivalente al hecho de que el problema homogéneo tenga $v = 0$ como su solución única.

Finalmente, se define el operador $\wp$: Sea $\wp$ un operador lineal $\mathcal{C}(\bar{F}) \to \mathcal{C}(F \setminus \{x_{00}\})$ que está dado por los valores

$$\wp(z)(x_{lk} = c(x_{lk}, x_{lk+1})z(x_{lk+1}) + c(x_{lk}, x_{l+1k})z(x_{l+1k}) + c(x_{lk}, x_{l-1k})z(x_{l-1k})$$

por cualquier $z \in \mathcal{C}(\bar{F})$ y $x_{lk} \in F \setminus \{x_{00}\}$. Este operador se empleará en el algoritmo y se explicará detalladamente en la sección del algoritmo en código.

7.3 Ejemplos de determinación de solución a problemas de contorno sobredeterminados

Para poder aplicar algunos conceptos explicados anteriormente se resolverán algunos ejemplos **[5]** propuestos en la figura 4:

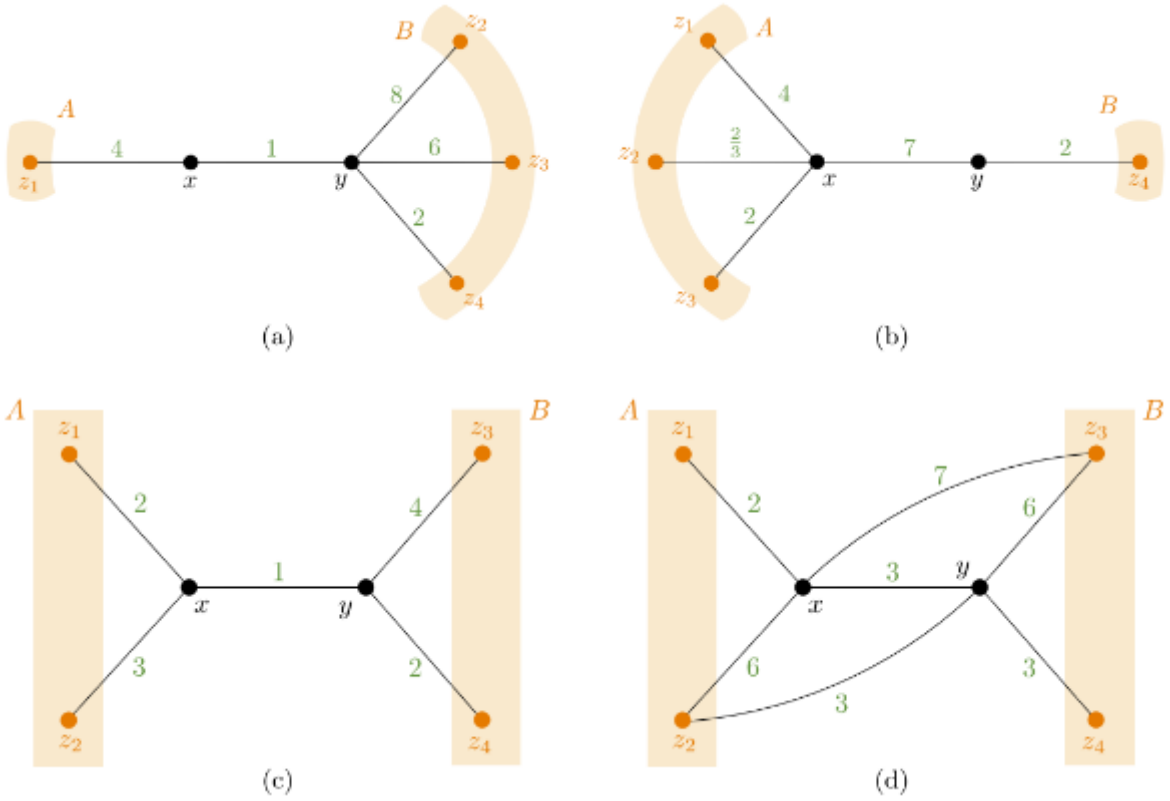


Figura 4: Ejemplos de diferentes redes con diferentes contornos. Fuente: **[1]**

Se resolverán los diferentes tipos de redes presentes en la figura 4:

- Ejemplo (a): Primeramente se construirá el Laplaciano de la red:

$$\mathscr{L}_a = \begin{bmatrix} 4 & 0 & 0 & 0 & -4 & 0 \\ 0 & 8 & 0 & 0 & 0 & -8 \\ 0 & 0 & 6 & 0 & 0 & -6 \\ 0 & 0 & 0 & 2 & 0 & -2 \\ -4 & 0 & 0 & 0 & 5 & -1 \\ 0 & -8 & -6 & -2 & -1 & 17 \end{bmatrix} \tag{61}$$

  Seguidamente, se constuirá el vector $q_{\sigma a}$

$$q_{\sigma_a} = -\mathscr{L}(\sigma_a)/\sigma_a \tag{62}$$

donde $\mathscr{L}_{\sigma a}$ se calcula como la conductancia por el diferencial de potencial en los nodos adyacentes como se ha explicado en (47) pero considerando que estamos en un entorno discreto la integral se convierte en sumatorio: $\sum_{y\in V} c(x,y)(\sigma(x)-\sigma(y))$. Por lo tanto, el vector es:

$$q_{\sigma a} = -\mathscr{L}(\sigma_a)/\sigma_a = \frac{-1}{\sigma_a}\begin{bmatrix}-2\\-4\\-3\\-1\\2\\8\end{bmatrix} = \begin{bmatrix}12\\24\\18\\6\\-3\\-12\end{bmatrix} \tag{63}$$

Una vez construido el vector del potencial asociado al peso, se calculará la matriz del Laplaciano teniendo en cuenta el peso con la equación $q = q_\sigma + 2\chi_{\delta(F)}$ mencionada anteriormente donde el valor λ es un valor que hemos fijado.

$$\mathscr{L}_{q_{\sigma_a}} = \begin{bmatrix}18&0&0&0&-4&0\\0&34&0&0&0&-8\\0&0&26&0&0&-6\\0&0&0&10&0&-2\\-4&0&0&0&2&-1\\0&-8&-6&-2&-1&5\end{bmatrix} \tag{64}$$

Una vez calculada la matriz de conductancias con su respectivo peso, el problema a resolver es:

$$\mathscr{L}_q(u_a) = (0,2) \quad \text{en } F, \quad \frac{\partial u_a}{\partial \mathsf{n}_F} = 1 \quad \text{en } A \quad u_a = 3 \quad \text{en } A\cup B \tag{65}$$

Por lo tanto, se deberá buscar las soluciones que cumplan las condiciones en (65): En primer lugar, se buscará los valores en el interior es decir en F y por ello , se selecciona la información en de las filas en x e y respectivamente y se podrá relacionar las incógnitas con los valores f dado por el enunciado:

$$\begin{bmatrix}-4&0&0&0&2&-1\\0&-8&-6&-2&-1&5\end{bmatrix}\begin{bmatrix}3\\u(z_2)\\u(z_3)\\u(z_4)\\u(x)\\u(y)\end{bmatrix} = \begin{bmatrix}0\\2\end{bmatrix} \tag{66}$$

Además hay una segunda condición a tener en cuenta en el contorno A:

$$\frac{\partial u_a}{\partial \mathsf{n}}(u(z_1)) = 4(u(z_1)-u(x)) = 4(3-u(x)) = 1 \to u(x) = \frac{11}{4} \tag{67}$$

Con el valor de $u(x)$ conocido se buscará el valor de $u(y)$, $u(z_2)$, $u(z_3)$:

$$-12 + 2u(x) - u(y) = 0 \tag{68}$$

$$-8u(z_2) - 6u(z_3) - 2u(z_4) - u(x) + 5u(y) = 2 \tag{69}$$

Se obtiene que el valor de $u(y) = \frac{-13}{2}$ y el valor de $u(z_4) = \frac{149}{8} - 4u(z_2) - 6u(z_3)$ donde los valores de $u(z_2)$ y $u(z_3) \in \mathbb{R}$. Como se puede observar la diferencia de cardinales de los dos tipos de contornos es 2 que equivale a los dos grados de libertad del sistema y significa que hay una infinidad de soluciones para este problema en concreto.

- Ejemplo (b): Primeramente se construirá el Laplaciano de la red:

$$\mathscr{L}_b = \begin{bmatrix} 4 & 0 & 0 & 0 & -4 & 0 \\ 0 & \frac{2}{3} & 0 & 0 & -\frac{2}{3} & 0 \\ 0 & 0 & 2 & 0 & -2 & 0 \\ 0 & 0 & 0 & 2 & 0 & -2 \\ -4 & -\frac{2}{3} & -2 & 0 & \frac{41}{3} & -7 \\ 0 & 0 & 0 & -2 & -7 & 9 \end{bmatrix} \tag{70}$$

Seguidamente, se constuirá el vector $q_{\sigma b}$

$$q_{\sigma_b} = -\mathscr{L}(\sigma_b)/\sigma_b = \frac{-1}{\sigma_b} \begin{bmatrix} -2 \\ -\frac{1}{3} \\ -1 \\ -1 \\ \frac{10}{3} \\ 1 \end{bmatrix} = \begin{bmatrix} 12 \\ 2 \\ 6 \\ 6 \\ -5 \\ -\frac{3}{2} \end{bmatrix} \tag{71}$$

Una vez construido el vector del potencial asociado al peso, se calculará la matriz del Laplaciano teniendo en cuenta el peso con $\lambda = 2$:

$$\mathscr{L}_{q_{\sigma_b}} = \begin{bmatrix} 18 & 0 & 0 & 0 & -4 & 0 \\ 0 & \frac{14}{3} & 0 & 0 & -\frac{2}{3} & 0 \\ 0 & 0 & 10 & 0 & -2 & 0 \\ 0 & 0 & 0 & 10 & 0 & -2 \\ -4 & -\frac{2}{3} & -2 & 0 & \frac{26}{3} & -7 \\ 0 & 0 & 0 & -2 & -7 & \frac{15}{2} \end{bmatrix} \tag{72}$$

Una vez calculada la matriz de conductancias con su respectivo peso, el problema a resolver es:

$$\mathscr{L}_q(u_b) = (4, 3) \quad \text{en } F, \qquad \frac{\partial u_b}{\partial \mathsf{n}_F} = (2, 1, -1) \quad \text{en } A \qquad u_b = (0, 1, -1) \quad \text{en } A \cup B \tag{73}$$

Por lo tanto, se deberá buscar las soluciones que cumplan las condiciones en (73). En primer lugar, nos interesará buscar los valores en el interior es decir en F y por ello , se selecciona la información en de las filas en x e y respectivamente y se podrá relacionar las incógnitas con los valores f dado por el enunciado:

$$\begin{bmatrix} -4 & -\frac{2}{3} & -2 & 0 & \frac{26}{3} & -7 \\ 0 & 0 & 0 & -2 & -7 & \frac{15}{2} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ -1 \\ u(z_4) \\ u(x) \\ u(y) \end{bmatrix} = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \tag{74}$$

Además hay una segunda condición a tener en cuenta en el contorno A que da el mismo valor de $u(x)$ empleando cualquier de las 3 condiciones de Neumann:

$$\frac{\partial u_b}{\partial \mathsf{n}}(u(z_1)) = 4(u(z_1) - u(x)) = 4(3 - u(x)) = 2 \rightarrow u(x) = \frac{-1}{2} \tag{75}$$

Con el valor de $u(x)$ conocido se buscará el valor de $u(y)$ y $u(z_4)$:

$$-\frac{2}{3} + 2 + \frac{26}{3}u(x) - 7u(y) = 4 \tag{76}$$

$$-2u(z_4) - 7u(x) + \frac{15}{2}u(y) = 3 \tag{77}$$

Se obtiene que el valor de $u(y) = -1$ y el valor de $u(z_4) = -3.5$. Como se puede observar, el problema tiene solución y es única.

- Ejemplo (b) tiene otras condiciones de contorno para observar otro tipo de solución: El procedimiento a realizar para obtener el laplaciano considerando el peso es el mismo que en el otro apartado, por lo tanto:

$$\mathscr{L}_{q_{\sigma_{b'}}} = \begin{bmatrix} 18 & 0 & 0 & 0 & -4 & 0 \\ 0 & \frac{14}{3} & 0 & 0 & -\frac{2}{3} & 0 \\ 0 & 0 & 10 & 0 & -2 & 0 \\ 0 & 0 & 0 & 10 & 0 & -2 \\ -4 & -\frac{2}{3} & -2 & 0 & \frac{26}{3} & -7 \\ 0 & 0 & 0 & -2 & -7 & \frac{15}{2} \end{bmatrix} \tag{78}$$

El problema a resolver es:

$$\mathscr{L}_q(u_{b'}) = (3,1) \quad \text{en } F, \quad \frac{\partial u_{b'}}{\partial \mathsf{n}_F} = (0,1,1) \quad \text{en } A \quad u_{b'} = (3,0,2) \quad \text{en } A \cup B \tag{79}$$

En primer lugar, imponiendo la condición de Neumann se puede observar que cada ecuación da un valor de $u(x)$ diferente por lo tanto el problema no tiene solución :

$$\frac{\partial u_{b'}}{\partial \mathsf{n}}(u(z_1)) = 4(u(z_1) - u(x)) = 0 \rightarrow u(x) = 3 \tag{80}$$

$$\frac{\partial u_{b'}}{\partial \mathsf{n}}(u(z_2)) = \frac{2}{3}(u(z_2) - u(x)) = 1 \rightarrow u(x) = -1.5 \tag{81}$$

$$\frac{\partial u_{b'}}{\partial \mathsf{n}}(u(z_3)) = 2(u(z_3) - u(x)) = 1 \rightarrow u(x) = 1.5 \tag{82}$$

- Ejemplo (c): Primeramente se construirá el Laplaciano de la red:

$$\mathscr{L}_c = \begin{bmatrix} 2 & 0 & 0 & 0 & -2 & 0 \\ 0 & 3 & 0 & 0 & -3 & 0 \\ 0 & 0 & 4 & 0 & 0 & -4 \\ 0 & 0 & 0 & 2 & 0 & -2 \\ -2 & -3 & 0 & 0 & 6 & -1 \\ 0 & 0 & -4 & -2 & -1 & 7 \end{bmatrix} \tag{83}$$

Seguidamente, se constuirá el vector $q_{\sigma c}$

$$q_{\sigma_c} = -\mathscr{L}(\sigma_c)/\sigma_c = \frac{-1}{\sigma_c} \begin{bmatrix} 0 \\ \frac{3}{2} \\ 2 \\ 0 \\ -\frac{3}{2} \\ -2 \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{9}{4} \\ -3 \\ 0 \\ 9 \\ 12 \end{bmatrix} \tag{84}$$

Una vez construido el vector del potencial asociado al peso, se calculará la matriz del Laplaciano teniendo en cuenta el peso con $\lambda = 2$:

$$\mathscr{L}_{q_{\sigma_c}} = \begin{bmatrix} 4 & 0 & 0 & 0 & -2 & 0 \\ 0 & \frac{11}{4} & 0 & 0 & -3 & 0 \\ 0 & 0 & 3 & 0 & 0 & -4 \\ 0 & 0 & 0 & 4 & 0 & -2 \\ -2 & -3 & 0 & 0 & 15 & -1 \\ 0 & 0 & -4 & -2 & -1 & 19 \end{bmatrix} \tag{85}$$

Una vez calculada la matriz de conductancias con su respectivo peso, el problema a resolver es:

$$\mathscr{L}_q(u_c) = (0,1) \quad \text{en } F, \qquad \frac{\partial u_c}{\partial \mathsf{n}_F} = (16,54) \quad \text{en } A \quad u_c = (-8,2) \quad \text{en } A \cup B \tag{86}$$

Por lo tanto, se deberá buscar las soluciones que cumplan las condiciones en (86): En primer lugar, nos interesará buscar los valores en el interior es decir en F y por ello , se

selecciona la información en de las filas en x e y respectivamente y se podrá relacionar las incógnitas con los valores f dado por el enunciado:

$$\begin{bmatrix} -2 & -3 & 0 & 0 & 15 & -1 \\ 0 & 0 & -4 & -2 & -1 & 19 \end{bmatrix} \begin{bmatrix} -8 \\ 2 \\ u(z_3) \\ u(z_4) \\ u(x) \\ u(y) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{87}$$

Además hay una segunda condición a tener en cuenta en el contorno A que da el mismo valor de $u(x)$ empleando cualquier de las 2 condiciones de Neumann:

$$\frac{\partial u_c}{\partial \mathsf{n}}(u(z_1)) = 2(u(z_1) - u(x)) = 2(-8 - u(x)) = 16 \rightarrow u(x) = -16 \tag{88}$$

Con el valor de $u(x)$ conocido se buscará el valor de $u(y)$, $u(z_3)$ y $u(z_4)$:

$$-2(-8) - 3 \cdot 2 + 15u(x) - u(y) = 0 \tag{89}$$
$$-4u(z_3) - 2u(z_4) - u(x) + 19u(y) = 1 \tag{90}$$

Se obtiene que el valor de $u(y) = -230$ y el valor de $u(z_4) = -2u(z_3) - \frac{-4355}{2}$ donde $u(z_3) \in \mathbb{R}$. Como se puede observar, el problema tiene un grado de libertad y por lo tanto, hay una infinidad de soluciones.

- Ejemplo (d): Primeramente se construirá el Laplaciano de la red:

$$\mathscr{L}_d = \begin{bmatrix} 2 & 0 & 0 & 0 & -2 & 0 \\ 0 & 9 & 0 & 0 & -6 & -3 \\ 0 & 0 & 13 & 0 & -7 & -6 \\ 0 & 0 & 0 & 3 & 0 & -3 \\ -2 & -6 & -7 & 0 & 18 & -3 \\ 0 & -3 & -6 & -3 & -3 & 15 \end{bmatrix} \tag{91}$$

Seguidamente, se constuirá el vector $q_{\sigma d}$

$$q_{\sigma_d} = -\mathscr{L}(\sigma_d)/\sigma_d = \frac{-1}{\sigma_d} \begin{bmatrix} -1 \\ -\frac{9}{2} \\ -\frac{13}{2} \\ -\frac{3}{2} \\ \frac{11}{2} \\ 6 \end{bmatrix} = \begin{bmatrix} 6 \\ 27 \\ 39 \\ 9 \\ -\frac{11}{2} \\ -9 \end{bmatrix} \tag{92}$$

Una vez construido el vector del potencial asociado al peso, se calculará la matriz del Laplaciano teniendo en cuenta el peso con $\lambda = 2$:

$$\mathscr{L}_{q_{\sigma_d}} = \begin{bmatrix} 10 & 0 & 0 & 0 & -2 & 0 \\ 0 & 38 & 0 & 0 & -6 & -3 \\ 0 & 0 & 54 & 0 & -7 & -6 \\ 0 & 0 & 0 & 14 & 0 & -3 \\ -2 & -6 & -7 & 0 & \frac{25}{2} & -3 \\ 0 & -3 & -6 & -3 & -3 & 6 \end{bmatrix} \tag{93}$$

Una vez calculada la matriz de conductancias con su respectivo peso, el problema a resolver es:

$$\mathscr{L}_{qd}(u) = (7, -2) \quad \text{en } F, \quad \frac{\partial u_d}{\partial \mathsf{n}_F} = (2, 0) \quad \text{en } A \quad u_d = (1, -2) \quad \text{en } A \cup B \tag{94}$$

Por lo tanto, se deberá buscar las soluciones que cumplan las condiciones en (94): En primer lugar, nos interesará buscar los valores en el interior es decir en F y por ello , se selecciona la información en de las filas en x e y respectivamente y se podrá relacionar las incógnitas con los valores f dado por el enunciado:

$$\begin{bmatrix} -2 & -6 & -7 & 0 & \frac{25}{2} & -3 \\ 0 & -3 & -6 & -3 & -3 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ u(z_3) \\ u(z_4) \\ u(x) \\ u(y) \end{bmatrix} = \begin{bmatrix} 7 \\ -21 \end{bmatrix} \tag{95}$$

Además hay una segunda condición a tener en cuenta en el contorno A que da el mismo valor de $u(x)$ empleando cualquier de las 2 condiciones de Neumann:

$$\frac{\partial u}{\partial \mathsf{n}}(u(z_1)) = 2(u(z_1) - u(x)) = 2(1 - u(x)) = 2 \to u(x) = 0 \tag{96}$$

Una vez identificado el valor $u(x)$ se empleará la segunda condición de Neumann en el segundo nodo para determinar el valor de $u(y)$:

$$\frac{\partial u}{\partial \mathsf{n}}(u(z_2)) = 6(u(z_2) - u(x)) + 6(u(z_2) - u(y)) = 0 \to u(y) = -6 \tag{97}$$

Con los valores de $u(x)$ y $u(y)$ conocidos se buscarán los valores $u(z_3)$ y $u(z_4)$:

$$-2 + 12 - 7u(z_3) + \frac{25}{2}u(x) - 3u(y) = 7 \tag{98}$$

$$6 - 6u(z_3) - 3u(z_4) - 3u(x) + 6u(y) = -21 \tag{99}$$

Se obtiene que el valor de $u(z_3) = 3$ y el valor de $u(z_4) = -9$. Como se puede observar, el problema tiene solución y es única.

7.4 Aplicación Dirichlet-to-Neumann parcial

Se considera el siguiente problema de Dirichlet:

$$\mathscr{L}_q(u_g) = 0 \quad \text{en } F, \quad u_g = g \quad \text{en } \delta(F) \tag{100}$$

Además, se impone que $\sigma \in \Omega(\bar{F})$ y existe $\lambda \geq 0$ tal que $q = q_\sigma + \lambda \mathcal{X}_{\delta(F)}$. Entonces el problema tiene una única solución por cualquier $g \in \mathcal{C}(\delta(F))$.
Se define la Aplicación de Dirichlet-to-Neumann parcial como un operador lineal $\Lambda_{A,B} : \mathcal{C}(A) \to \mathcal{C}(B)$, que asigna a cada $v \in \mathcal{C}(A)$ la función:

$$\Lambda_{A,B}(v) = \frac{\partial u_v}{\partial \mathsf{n}_F} \mathcal{X}_B. \tag{101}$$

Entonces, $\Lambda_{B,A}$ se define de manera análoga a $\Lambda_{A,B}$. Se puede observar que $u_v = 0$ en $B \cup R$ y por lo tanto $\Lambda_{A,B}(v) = \Lambda_q(v)\mathcal{X}_B$ Con ello se obtiene las siguientes proposiciones:

- **Proposición 3.4** $\Lambda^*_{A,B} = \Lambda_{B,A}$, Y además, $ker\Lambda_{A,B} = \nu_A \cdot \mathcal{X}_A$, y $ker\Lambda_{B,A} = \nu_B \cdot \mathcal{X}_B$. Las matrices asociadas a $\Lambda_{A,B}$ y $\Lambda_{B,A}$ son $N_q \cdot \mathcal{X}_{B\times A}$ y $N_q \cdot \mathcal{X}_{A\times B}$, y por lo tanto, las matrices asociadas son: $N_q(B; A)$ y $N_q(A; B)$, respectivamente.

- **Corolario 3.5** El problema $\frac{\partial u}{\partial \mathsf{n}_F} = h$ en A y $u = g$ en $A \cup R$ tiene una solución por cualquier dato si y solo si $\Lambda_{A,B}$ tiene el rango máximo lo que también ocurre cuando $\Lambda_{B,A}$ tiene rango máximo. En particular, cuando $|A| = |B|$, $\Lambda_{A,B}$ es no singular si y sólo si $\Lambda_{B,A}$ es no singular, y en este caso el problema tiene una solución única por cualquier dato.

- **Proposición 3.6** se supone que $\Lambda_{A,B}$ tiene un rango máximo y $g \in \mathcal{C}(A \cup R)$ y $h \in \mathcal{C}(A)$. Si u es una solución del problema

$$\mathscr{L}_q(u) = 0 \quad \text{en } F, \qquad u = g \quad \text{en } A \cup R \qquad \text{y} \quad \frac{\partial u}{\partial \mathsf{n}_F} = h \quad \text{en } A. \tag{102}$$

  Entonces, los valores de u en B están determinados por la identidad:

$$\Lambda_{B,A}(u) = h - \Lambda_{A\cup R,A(g)}.$$

  Si además $|A| = |B|$, entonces:

$$u = \Lambda^{-1}_{B,A}(h) - \Lambda^{-1}_{B,A} \circ \Lambda_{A\cup R,A}(g). \tag{103}$$

  Curtis y Morrow **[13, Corolario 3.14]** demostraron que para una red plana circular, $\Lambda_{A,B}$ es no singular si y sólo si el par (A; B) está conectado a través de Γ. En particular, esto ocurre si Γ está bien conectado. En el mismo estudio, identificaron cuatro tipos de redes planas básicas bien conectadas y describieron un algoritmo para recuperar las conductancias. Una de las características clave de este algoritmo es la fórmula de punta de contorno que permite la recuperación de la conductancia en una arista de punta del contorno. La prueba de este resultado se puede adaptar fácilmente al caso con los operadores de Schrödinger y las Aplicaciones de Dirichlet a Neumann.

- **Lemma 3.7** Sea $\Gamma = (\bar{F}, c)$ una red plana y circular conexa. Se supone que Γ tiene una punta de contorno xy con $x \in \delta(F)$ y $y \in F$. Si la contracción xy a un único vértice del contorno rompe la conexión a través de Γ entre un par circular (A, B) , entonces:

$$c(x,y) = \frac{\omega(x)}{\omega(y)}(N_q(x;x) - N_q(x;B) \cdot N_q(A;B)^{-1} \cdot N_q(A;x) - \lambda). \tag{104}$$

Todos los resultados de esta sección son ciertos también para redes generales, no necesariamente planares.

8 Algoritmo para la recuperación de las conductancias en redes Spider

El objetivo de esta sección es la recuperación completa de la función de conductividad de una familia de redes utilizando únicamente la información proporcionada por la Aplicación Dirichlet-to-Robin. Este algoritmo considera el Laplaciano así como los operadores de Schrödinger. Se asume $q = q_\sigma + \lambda_{\mathcal{X}\delta(F)}$ donde $\sigma \in \Omega(\bar{F})$ y $\lambda \geq 0$.
Se considera un disco en el plano y ∂D su frontera. Una red circular se representa en el círculo coincidiendo los nodos frontera con la frontera del disco. Una red araña bien conectada se caracteriza por tener $n \equiv 3(Mod4)$ radios y $m = \frac{n-3}{4}$ círculos. Entonces, $\Gamma = (V, c)$ tiene n vértices de contorno dados por $\delta(F) = \{v1, ..., v_n\}$ que se colocan en el orden circular proporcionado por ∂D.

Los vértices en F se distribuyen de la siguiente manera:

- Se coloca un vértice x_{00} en el centro del círculo de contorno ∂D y se dibuja una línea recta de x_{00} a cada v_j: esta línea se llama radio j.
- Se dibuja m diferentes circunferencias concéntricas con centro x_{00} de tal manera que todos se encuentran en $\dot{D}$. Cada uno de estos círculos serán los círculos i, donde los círculos están etiquetados desde el menor diámetro al diámetro más alto.
- Por último, se coloca un vértice $x_{j,i}$ en la intersección de cada círculo i y radio j.
  Entonces, $F = \left\{ x_{ji} \right\}_{i=1,...,m, j=1,...,n} \cup \{x_{00}\}$. Sea $x_{j0} = x_{00}$ y $x_{jm+1} = v_j$ para todo $j = 1, ..., n$. Además, se considera la notación $x_{ji} = x_{j-ni}$ para cualquier $j \geq n$. Para cada $j = 1, ..., n$ los conjuntos de contorno son $A_j = \{v_{1+j}, ..., v_{\frac{n11}{2}+j}\} \subset \delta(F)$, $B_j = \{v_{\frac{n+1}{2}} + j, ..., v_{n-1+j}\} \subset \delta(F)$ y $R_j = \{v_j\} \subset \delta(F)$.

Por otra parte, estas configuraciones de frontera en una red araña bien conectada garantizan que A_j y B_j siempre están conectados a través de Γ, y de hecho el conjunto formado por los vértices en los caminos que conectan A_j y B_j coincide con $V \backslash v_j$.

Dado un índice $i \in \{0, ..., m + 1\}$, las capas circulares de vértices son $D_i = \{x_{li} \in V : l = 1, ..., n\} \subset V$. En particular, $D_0 = x_{00}$ y $D_{m+1} = \delta(F)$.
La recuperación de las conductancias en una red araña bien conectada es un proceso iterativo porque no somos capaces de dar fórmulas explícitas para todas las conductancias al mismo tiempo, por lo que proporcionamos un algoritmo de recuperación en su lugar. Por lo tanto, describimos el algoritmo en pasos, cada uno de los cuales requiere la información obtenida en el paso anterior.

En primer lugar, se considera N_q una matriz de respuesta con $\lambda \geq 0$ como menor autovalor y $\sigma \in \Omega(\delta(F))$ es el autovector asociado a λ. Además, seleccionamos $\omega \in \Omega(\bar{F})$ de tal manera que $\omega = k\sigma$ en $\delta(F)$, 0 < k < 1.

Las entradas del algoritmo de recuperación están dadas por la información conocida, como

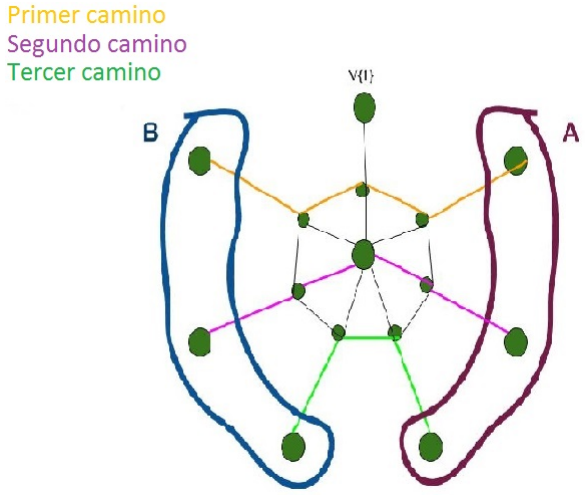


Figura 5: Ejemplo de dos subconjuntos bien conectados. Elaboración propia

N_q, λ, y ω, y las salidas son las conductancias c tales que Λ_q es la Aplicación Dirichlet-to-Robin asociada con el operador de Schrödinger con potencial $q = q_\omega + \lambda \mathcal{X}_{\delta(F)}$ y $c_\omega = \frac{c}{w \otimes w}$. Además, es importante tener en cuenta que este algoritmo de recuperación equivale a recuperar q. Las matrices en el último conjunto son singulares y débilmente dominante diagonalmente.

Los pasos a realizar para reconstruir las conductancias son:

0 Para recuperar las conductancias es necesario fijar los rayos y considerar el problema de contorno parcial sobredeterminado que implica encontrar $u_j \in \mathcal{C}(\bar{F})$ tal que

$$\mathscr{L}_q(u_j) = 0 \quad \text{en } F, \qquad u_j = \varepsilon_{vj} \quad \text{en } A_j \cup R_j \quad \text{y} \quad \frac{\partial u_j}{\partial \mathsf{n}_F} = 0 \quad \text{en } A_j. \tag{105}$$

Existe un gran conjunto de vértices en la red araña bien conectada Γ en el que $u_j = 0$. Se denota este conjunto por:

$$Z(u_j) = \{x \in \bar{F} : u_j(x) = 0\} = \bar{F} \backslash supp(u) \tag{106}$$

Claramente, $A_j \subseteq Z(uj)$. Sin embargo, $Z(uj)$ es mucho más grande que A_j.

- **Proposición 4.1** Se satisface que:

$$Z(u_j) = \{x_{li} \in V : i = 0, ..., m+1, l = i + j - m, ..., 3m + 2 + j - i\}. \tag{107}$$

1 Se fija el índice $j \in \{1, ..., n\}$ y se considera la solución única $u_j \in \mathcal{C}(\bar{F})$ al problema (105). Se sabe que $u_j = 0$ en A_j y $u_j = 1$ en R_j. Además, los valores de u_j en B_j están dados por

$$u_{Bj} = -Nq(A_j; B_j)^{-1} \cdot N_q(A_j; v_j). \tag{108}$$

Como consecuencia, se obtiene u_j en $\delta(F)$.

2 Aplicando la fórmula de la punta del contorno dada en el Lemma 3.7 y considerando que todas las aristas de contorno son puntas, se conocen los valores de las conductancias de todas las aristas que unen los vértices desde D_{m+1} y D_m.

3 Se vuelve a fijar el índice $j \in \{1, ..., n\}$ y se considera la solución única $u_j \in \mathcal{C}(\bar{F})$ al problema (105). Entonces, se obtiene los valores de u_j en D_m mediante el Lemma 4.2.

- **Lemma 4.2** Los valores de u_j en D_m están dados por

$$u_j(x_{km}) = \frac{1}{c(v_k, x_{km})}(\lambda u_j(v_k) - N_q(v_k; v_j) - N_q(v_k; B_j) \cdot u_{Bj}) + \frac{\omega(x_{km})}{\omega(v_k)} u_j(v_k) \quad (109)$$

por todo $k = 1, ..., n$.

4 Se recuperan las conductancias de todas las aristas con ambos extremos en D_m.

- **Proposición 4.3** Sea $i \in \{0, ..., m-1\}$ y $j = 1, ..., n$. Se supone que se conocen los valores de u_j en D_{i+2} y D_{i+1} y las conductancias de todas las aristas que unen vértices de D_{i+2} y D_{i+1}. Luego, las conductancias $c(x_{i+j-m+1i+1}, x_{i+j-mi+1})$ están dadas por

$$c(x_{i+j-m+li+1}, x_{i+j-mi+1}) = \frac{-u_j(x_{i+j-m+1i+2})}{u_j(x_{i+j-mi+1)}} c(x_{i+j-m+1i+1}, x_{i+j-mi+2}). \quad (110)$$

5 Se recuperan las conductancias de todas las aristas que unen los vértices de D_m y D_{m-1}.

- **Proposición 4.4** Sea $i \in \{0, ..., m-1\}$ y $j = 1, ..., n$. Una vez conocidos los valores de u_j en D_{i+2} y D_{i+1}, las conductancias de todas las aristas que unen los vértices de D_{i+2} y D_{i+1} y las conductancias que unen los nodos en D_{i+1}, las conductancias $c(x_{i+j-mi}, x_{i+j-mi+1})$ están dadas por

$$c(x_{i+j-mi}, x_{i+j-mi+1}) = \left(\frac{\wp(u_j)(x_{i+j-mi+1})}{u_j(x_{i+j-mi+1})} - \frac{\wp(\omega)(x_{i+j-mi+1})}{\omega(x_{i+j-mi+1})} \right) \cdot \frac{\omega(x_{i+j-mi+1})}{\omega(x_{i+j-mi})} \quad (111)$$

6 En este paso, se calculan los valores de u_j en D_{m-1} para todos $j = 1, ..., n$.

- **Proposición 4.5** Sea $i \in \{0, ..., m-1\}$ y $j = 1, ..., n$. Los valores de u_j en D_{i+2} y D_{i+1}, las conductancias radiales como adyacentes se conocen, entonces los valores de u_j en D_i están dados por

$$u_j(x_{ki}) = \frac{u_j(x_{ki+1})\wp(\omega)(x_{ki+1})}{\omega(x_{ki+1})c(x_{ki+1}, x_{ki})} - \frac{\wp(u_j)(x_{ki+1})}{c(x_{ki+1}, x_{ki})} + \frac{\omega(x_{ki})}{\omega(x_{ki+1})} u_j(x_{ki+1}) \quad (112)$$

por todo $k = 1, .., n$.

7 Finalmente, se repite el mismo proceso para recuperar las conductancias aplicando la Porposición 4.3 del cuarto paso, seguido de la Proposición 4.4 del paso 5, y luego la Proposición 4.5 del paso 6 por cada $i = m$ - 2, ..., 0. El proceso se finaliza para $i = 0$ (cuando se han recorrido todos los círculos). El valor $u_j\ (x_{00}) = 0$ para todo $j = 1, ..., n$, que se conoce ya porque $x_{00} \in Z(u_j)$. Este es el paso final del proceso porque todas las conductancias son conocidas en este punto.

9 Algoritmo de recuperación de conductancias programado en Matlab

9.1 Algoritmo aplicado a una Spider de 11 nodos

En esta sección se explicará el proceso que se emplea para recuperar las conductancias. El ejemplo que se resolverá es de una red araña con 11 nodos. En este caso, cada círculo tiene 11 nodos ($n = 11$) distribuidos en 11 radios.

Los nodos se enumeran en sentido horario desde fuera hacia dentro para este algoritmo.

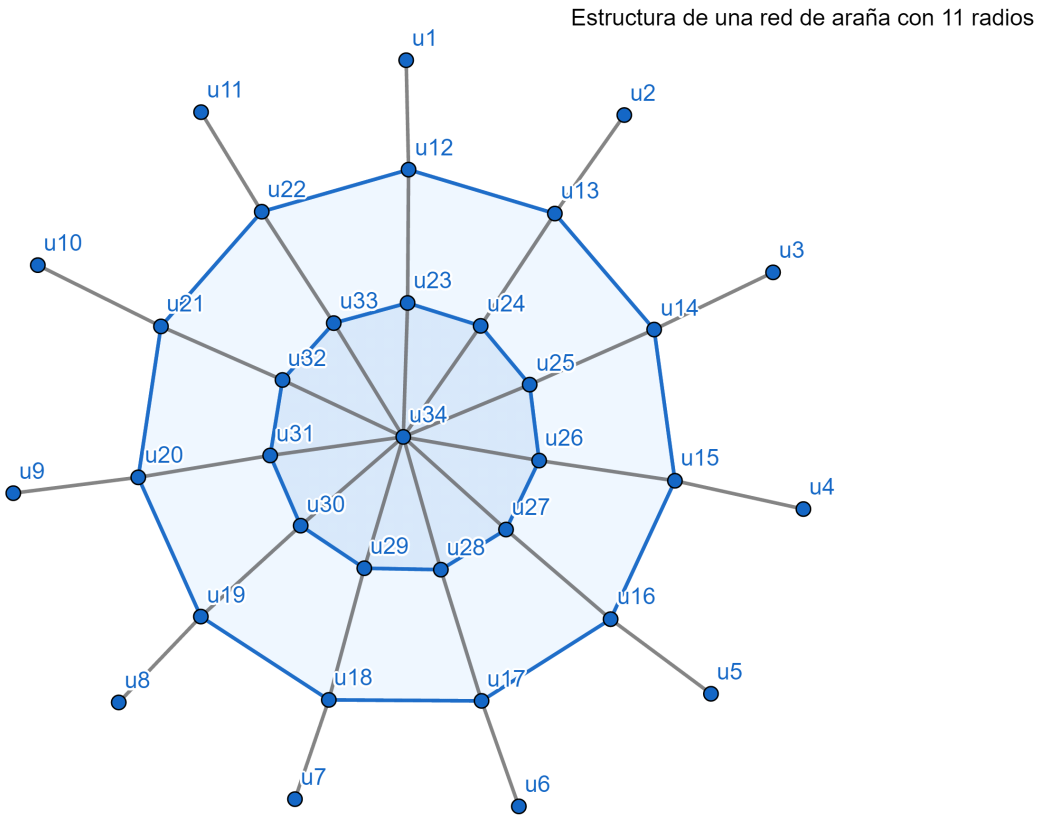


Figura 6: Spider de 11 rayos. Elaboración propia

Al ejecutar el programa, primeramente nos pide el número de rayos que tendrá la Spider

```
n = input('INTRODUCE EL NÚMERO DE LOS RADIOS n:');
```

E introducimos el valor de 11. Entonces, calcula los círculos que tendrá la Spider. En este caso son 2 círculos y lo imprimirá para que lo vea el usuario.

```
circulo = (n-3)/4;% El valor de círculos que tendrá la Spider

disp(['EL NÚMERO DE CÍRCULOS ES:',num2str(circulo)])
```

También nos pedirá el valor de la Lambda: Introduce el valor de Lambda.

Y le introduciremos el valor de 3.

Le introducimos la matriz de conductancias:

L= xlsread('Algoritmo_11_nodos.xlsx'); % Se importa la matriz de conductancia que será el enunciado, esta matriz será importada del Excel.

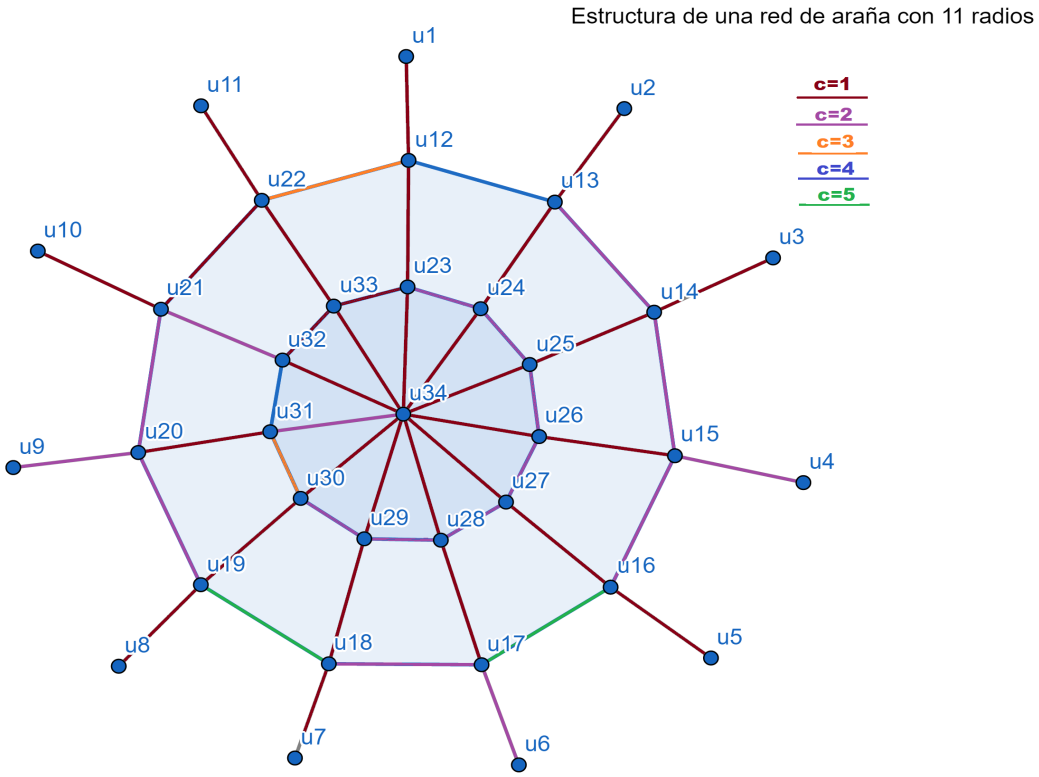


Figura 7: Spider de 11 nodos introducida en el MATLAB con las conductancias. Elaboración propia

Creamos una matriz que contendrá las conductancias que recuperará:

c = zeros((n∗ circulo+1)+n,(n∗ circulo+1)+n);% La matriz que contendrá las conductancias de las ramas de la Spider.

También creamos otra matriz con los primeros n nodos de la diagonal con el valor de la Lambda por el hecho de ser los nodos del contorno.

Lambda_diagonal = zeros((n∗ circulo+1)+n,(n∗ circulo+1)+n); % Se crea una matriz diagonal, donde en los primeros n nodos(que son los del contorno) habrá el peso de estos.

for i= 1: n % Se crea una matriz diagonal para contemplar el efecto de la Lambda en los nodos del contorno. Este valor de Lambda es el mismo para todos los nodos.

Lambda_diagonal(i,i)= Lambda;

end

Seguidamente, se introduce el vector Peso que contendrá los pesos de cada nodo que estarán en columna y se normaliza para emplear un vector con norma 1 aunque no tiene ningún efectos a nivel de cálculos

```
w = Peso'; % El vector columna de Peso lo transforma en fila para el cálculo posterior. Se realiza este cambio para que sea más ágil la escritura del algoritmo.
```

Además, se crea una matriz diagonal con los pesos en la diagonal:

```
Diag_peso = diag(Peso); % Se transforma el vector de peso en una matriz diagonal para los posteriores cálculos para coincidan las dimensiones.
```

Se calcula el Laplaciano de la Spider con los pesos declarados anteriormente.

```
Q_W=-diag(L* Peso)* inv(Diag_peso);

L_q= L+ Q_W + Lambda_diagonal; % Se calcula el Laplaciano teniendo en cuenta los pesos como el valor de la Lambda.
```

A posteriori, se calcula la matriz de respuesta utilizando el Complemento de Schur. Este complemento identifica 4 submatrices de la matriz del Laplaciano L_q y con ellas calcula la matriz de respuesta:

```
% Complemento de Schur

D_F = L_q(1:n,1:n);

C0 = L_q(1:n,n+1:n* circulo+n+1);

C0_t = L_q(n+1:n* circulo+n+1, 1:n);

L_F = L_q(n+1:n* circulo+n+1,n+1:n* circulo+n+1);

Nq= D_F - C0* inv(L_F) * C0_t; % Se calcula la matriz de respuesta mediante el Complemento de Schur
```

Una vez obtenida la matriz de respuesta se procederá al algoritmo para recuperar las conductancias. Es decir, el algoritmo real empieza desde aquí, ya que los cálculos anteriores eran para obtener la matriz de respuesta que se introduce al algoritmo real.

Además, para asegurar que la matriz de respuesta es simétrica se comprueba la simetría y lo imprime para que el usuario vea que la matriz principalmente cumple con este requisito:

```
 A= 0;

 for i= 1:n % Se comprueba que la matriz de respuesta es simétrica

    for j = 1:n

       if Nq(i,j)==Nq(j,i)

          A=A+0;

       else

          A= A+1;

       end
```

```
    end
end
if A==0
    disp('La matriz de respuesta es simétrica')
else
    disp('La matriz de respuesta no es simétrica')
end
```

A continuación, se calcula un parámetro que ofrece el Matlab para identificar la sensibilidad de la matriz, es decir, identificar si es una matriz que al generarle la inversa se pierde mucha precisión o no.

El número de condición de inversión de una matriz mide la sensibilidad de la solución de un sistema de ecuaciones lineales a errores en los datos. Da una indicación de la exactitud de los resultados de la inversión de la matriz y la solución de la ecuación lineal, busca la relación entre los valores mayor y menor singulares. En este caso, el valor obtenido es de 166,2306.

```
Ccondicion= cond(L_F); % devuelve el 2-norma para la inversión de los números, igual a la relación entre el mayor valor singular de L_F para los más pequeños. Si el número de condición de L_F es mucho mayor que 1, la matriz es sensible a los cálculos inversos.

disp(Ccondicion);
```

La sucesión de esta parte del código resuelve el problema inverso en sí. El primer paso, es recuperar el valor de la lambda y su autovector asociado que será el menor, ya que como se ha expuesto en los apartados anteriores, se escoge aquel valor que produce 'la mínima energía posible'.

Como se puede observar el valor de Lambda que se obtiene sustituirá al valor de Lambda inicial, pero éste será el mismo. En cambio, el vector peso que se obtiene, sólo recupera los valores del contorno normalizados, en cambio los valores de los nodos interiores dependiendo del valor de peso que se imponga se recuperan unos valores de conductancias u otros.
Debido a este factor, los cálculos de recuperación se realizan con los valores del vector peso inicial (w). Este vector w no está normalizado, ya que en los cálculos posteriores donde aparece el valor de w está dividiendo respecto otro valor de w y por lo tanto, normalizarlo o no en este caso no afecta a los cálculos.

```
[eig_vector, eig_value]=eig(Nq); % Se obtiene la matriz con todos los autovalores y autovectores de la matriz de respuesta.

Lambda= eig_value(1,1); % Se escoge el autovalor más pequeño, ya que se ordena de menor a mayor.

disp(Lambda)

ww = eig_vector(:,1); % Se escoge el autovector asociado a este autovalor.

for i= 1: n
  if ww(i,1)<0
     ww(i,1) = - ww(i,1);
```

```
  end
end
disp(ww)
% VALORES DE uj EN LOS CONTORNO R Y A DE LA SPIDER:
vj = zeros(1+ (n-1)/2,1);
```

El valor de Lambda obtenido es de 3, efectivamente y su autovector asociado es:

```
3.0000

0.3015
0.3015
0.3015
0.3015
0.3015
0.3015
0.3015
0.3015
0.3015
0.3015
0.3015
```

Figura 8: Autovalor y autovector de la Spider de 11 rayos. Elaboración propia

Se impone que el primer valor es 1 donde se ubicará el contorno R y los nodos $(n-1)/2$ se les impone un valor de 0, ya que es la condición de Dirichlet.

```
vj(1,1) = 1; % El primer valor corresponde al valor del contorno R que es 1.
for q= 2:(n-1)/2 % La primera mitad de los nodos del contorno seguidos del nodo donde se encuentra el nodo que pertenece a R, se impone que la función valorada en estos es 0.
  vj(q,1) = 0;
end
```

Se crea dos submatrices, una que contendrá los valores de uj en el contorno B y otra que contendrá los valor uj del contorno R, A y B.

```
% SUBMATRICES DE Nq: R(contorno que rompe la conexión donde uj=1), A(contorno donde uj= 0) y B(contorno donde los valores son desconocidos) que variarán a lo largo de j
uj= []; % Se declara la matriz que contendrá todos los valores de ujde los nodos de la Spider.
UB_j_t = []; % Se declara la matriz que contendrá los valores uj del contorno B.
```

Las condiciones de contorno que se impone en este tipo de problemas son tres contornos R, A y B.

- **En el contorno R, la función valorada en su respectivo nodo es 1.**

R_j = [j]

- **En el contorno A, la función valorada en la primera mitad de nodos seguidos del nodo que pertenece al contorno R.**

  A_j = [(1+j): ((n-1)/2+j)]

- **Finalmente, en el contorno B, se desconoce la solución de la función valorada en la otra mitad de los nodos.**

  B_j = [((n+1)/2+j):(n-1+j)]

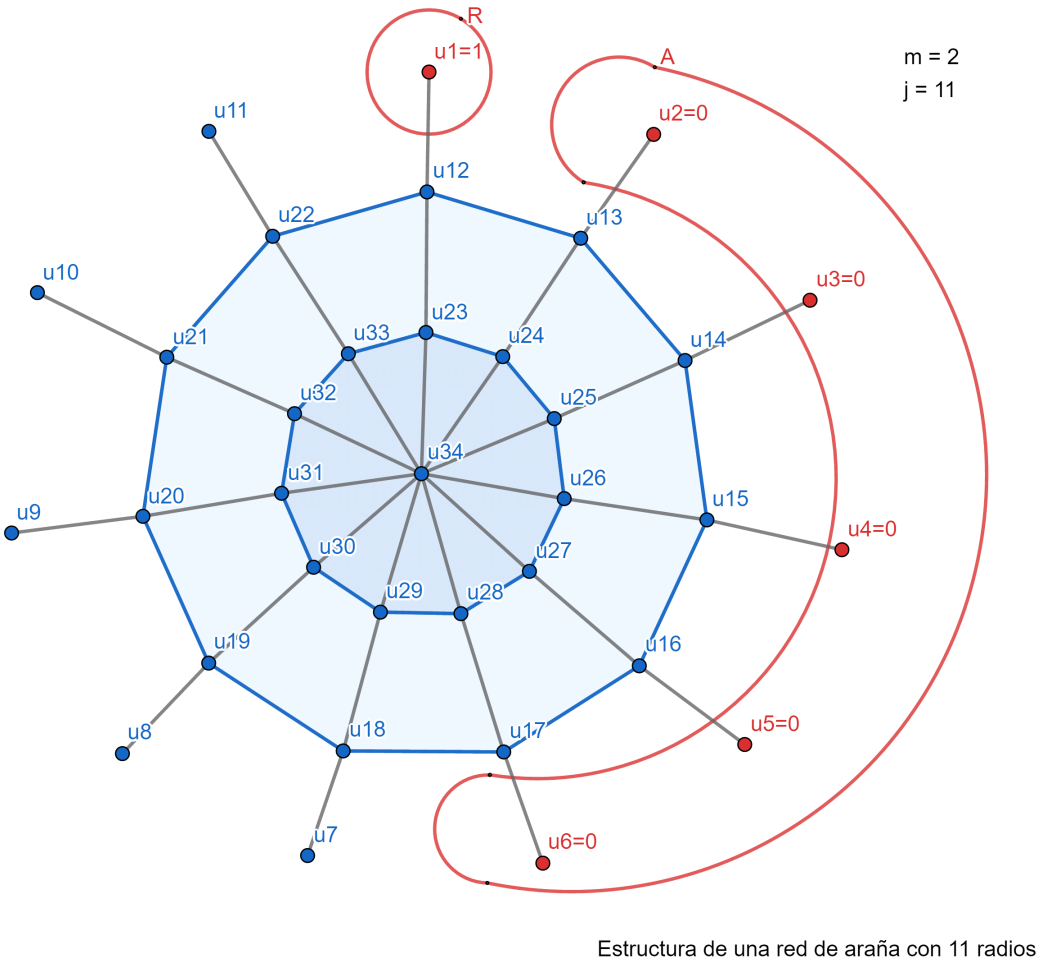


Figura 9: Condiciones de contorno de la Spider de 11 rayos. Elaboración propia

El primer paso a realizar es la obtención de los valores de la función valorada en los nodos del contorno B mediante la Proposición 3.6:

UB_j = -inv(Nq(A_j,B_j))∗ Nq(A_j,j)

Este primer paso que se ha realizado teniendo en cuenta que el nodo $n = 1$ forma del contorno R se debe repetir pero cambiando los nodos de contorno desde el primer nodo hasta $n =$11.

El problema a nivel de programación es que cuando el contorno R se encuentra en el nodo n =2, los nodos que forman parte del contorno A son: 3, 4, 5, 6 y 7. En cambio, los nodos que forman parte del contorno B son: 8, 9, 10, 11, 1 y aquí se puede observar que se tiene que reinicializar la cuenta. Este efecto se contempla restando n cuando se supera el índice n:

Como se podrá observar, j es la variable auxiliar que irá variando a lo largo de los nodos.

```
A_j = [(1+j): ((n-1)/2+j)]; % Se define el rango de índices que pertenecen al contorno A que serán la mitad de los nodos a partir del nodo donde se encuentra el contorno R.

 if (1+j)<=n & & ((n-1)/2+j) > n % En el caso que los nodos del contorno superen la totalidad de nodos, habrá que volver a reinicializar el recorrido de nodos.

    % En este caso, si suponemos que estamos en una Spider de 11 nodos y el contorno R se encuentra en el nodo 7, entonces los nodos que

   % forman parte del contorno A son: 8, 9, 10, 11, 1 y por lo tanto en el caso del último nodo que superaría los nodos

    % totales se le resta la cantidad de nodos n =11 para que vuelva a

    % iniciarse el recorrido.

   A1 = ((1+j):n); % Como se puede observar la submatriz A1 contendría desde el nodo 8 al 11.

   A2 = (1: ((n-1)/2+j)-n); % La submatriz A2 contendría los nodos de 1 a 1 para este ejemplo.

   A_j = [A1 A2]; % Se unen las dos submatrices A1 y A2 para construir el rango de los nodos del contorno A.

 elseif ((n-1)/2+j) > n & & (1+j)>n % En el caso que tanto la primera parte del primer rango como la segunda sean superiores, se reinicializa la cuenta en las dos partes del rango.

    % Un ejemplo de este caso es si el contorno R está en el nodo 11, entonces el rango de nodos que forman parte del contorno A son 1, .2, 3, 4, 5

   A_j = [(1+j)-n: ((n-1)/2+j)-n];

 end


 B_j = [((n+1)/2+j):(n-1+j)]; % Se define el rango de índices que pertenecen al contorno B que serán la segunda mitad de los nodos.

 if (n-1+j) > n & & ((n+1)/2+j)<=n % Se reinicializa los nodos cuando se excede el número de nodos totales siguiendo el mismo procedimiento que para los nodos del contorno A.

    B1 = (((n+1)/2+j):n);

    B2 = (1: (n-1+j)-n);

    B_j = [B1 B2];

 elseif(n-1+j) > n & & ((n+1)/2+j)>n
```

```
    B_j = [((n+1)/2+j)-n: (n-1+j)-n];

  end
```

En esta parte de código se calcula los valores uj en el contorno B:

```
UB_j = -inv(Nq(A_j,B_j))* Nq(A_j,j); % Se emplea la proposición 3.6 para obtener los valores de uj en los nodos del contorno B.

   vj_f_cons = [vj; UB_j]; % Con los valores de UB_j encontrados, se les añade éstos a la matriz inicial que contendrá los valores de la función en los nodos de la Spider por cada contorno R.

   UB_j_t = [ UB_j_t, UB_j]; % Matriz con todos los valores uj del contorno B por cada contorno R posicionado en un nodo.
```

Los valores de uj en el contorno B en cada contorno R ubicado en un nodo son los siguientes:

-2.5000	-1.0000	-0.1250	-1.0000	-1.5000	-2.4000	-1.0000	-0.4000	-1.0000	-1.6667	-1.3333
10.5000	2.3125	2.5000	18.0000	12.5000	9.6000	7.0000	1.5000	29.0000	2.1111	60.5000
-12.5833	-4.8750	-16.3750	-54.5000	-21.7500	-20.1000	-9.3750	-11.5000	-18.8333	-20.0000	-101.5000
15.6667	5.6875	25.0000	49.5000	16.7500	11.2500	24.2500	4.3667	48.5000	21.0000	37.2500
-3.6667	-4.5000	-12.0000	-14.0000	-3.7500	-10.2000	-5.1667	-3.4000	-29.0000	-3.2500	-22.0000

Figura 10: Recuperación de los valores de uj en el contorno B. Elaboración propia

Además, una vez obtenidos los resultados se deben ir cambiando de posición dependiendo en qué nodo esté el contorno R. Esto se contempla con el siguiente:

```
if j == 1

   vj_f_canv = vj_f_cons;% los valores del contorno corresponden a la matriz inicial

   end

   if j> 1

   vj_f_canv = [vj_f_cons((n-j+2): (n)); vj_f_cons(1:n-j+1)];% Cambio orden de las de los valores del contorno en función de la j que va variando, se van rotando las filas.

   end
```

1.0000	-4.5000	25.0000	-54.5000	12.5000	-2.4000	0	0	0	0	0
0	1.0000	-12.0000	49.5000	-21.7500	9.6000	-1.0000	0	0	0	0
0	0	1.0000	-14.0000	16.7500	-20.1000	7.0000	-0.4000	0	0	0
0	0	0	1.0000	-3.7500	11.2500	-9.3750	1.5000	-1.0000	0	0
0	0	0	0	1.0000	-10.2000	24.2500	-11.5000	29.0000	-1.6667	0
0	0	0	0	0	1.0000	-5.1667	4.3667	-18.8333	2.1111	-1.3333
-2.5000	0	0	0	0	0	1.0000	-3.4000	48.5000	-20.0000	60.5000
10.5000	-1.0000	0	0	0	0	0	1.0000	-29.0000	21.0000	-101.5000
-12.5833	2.3125	-0.1250	0	0	0	0	0	1.0000	-3.2500	37.2500
15.6667	-4.8750	2.5000	-1.0000	0	0	0	0	0	1.0000	-22.0000
-3.6667	5.6875	-16.3750	18.0000	-1.5000	0	0	0	0	0	1.0000

Figura 11: Recuperación de los valores de uj en el primer círculo. Elaboración propia

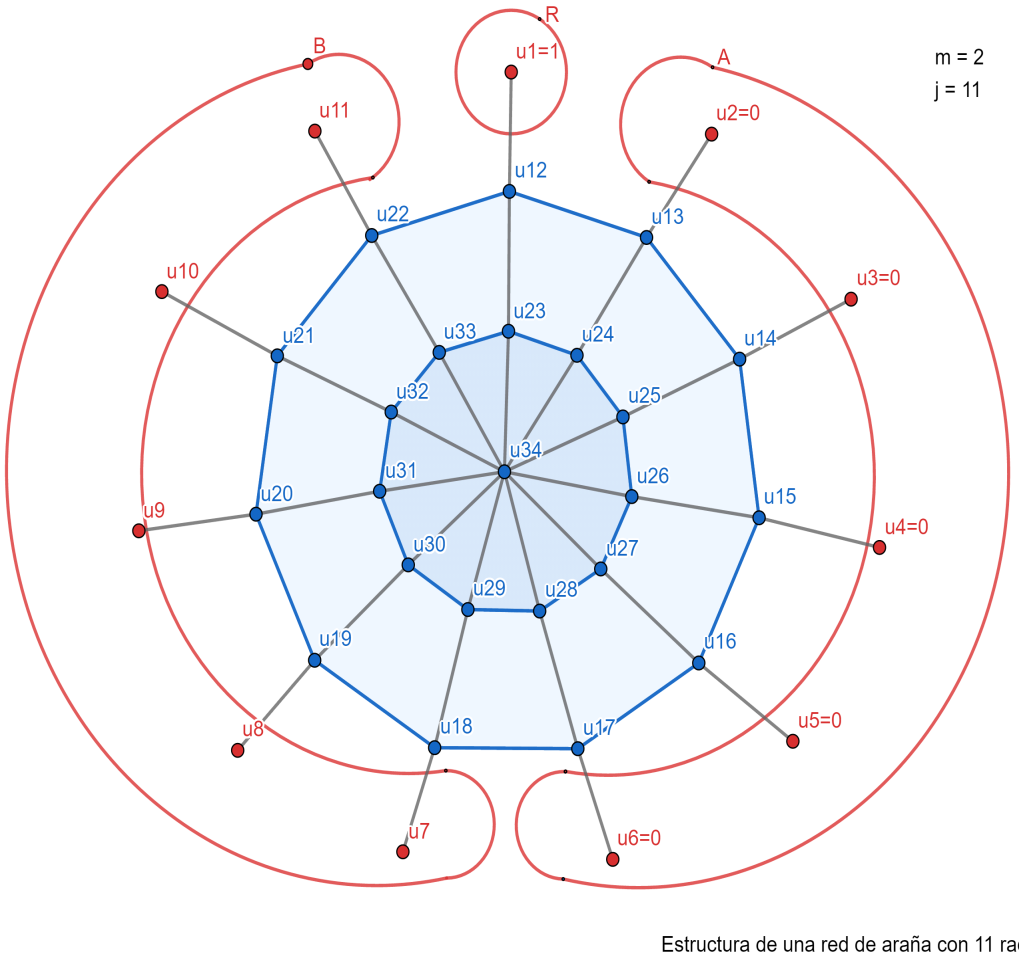


Figura 12: Recuperación de los valores de uj en el primer círculo de la Spider de 11 rayos. Elaboración propia

Una vez determinados todos los valores en los nodos del contorno, se procede a calcular las conductancias externas que conectan los nodos del contorno con el primer círculo mediante el Lema 3.7:

c(j,j+n) = w(j)/w(j+n)∗ (Nq(j,j)-Nq(j,B_j)∗ inv(Nq(A_j,B_j))∗ Nq(A_j,j)-Lambda)

Se debe ir cambiando la j como en el primer caso para determinar todas las conductancias. Una vez encontrada estas conductancias, se sabe que la conductancia que va del nodo 1 al 12 es la misma que de la de 12 a 1, por ello se impone la simetría de la matriz donde se guarda las conductancias:

c(j+n,j) = c(j,j+n)

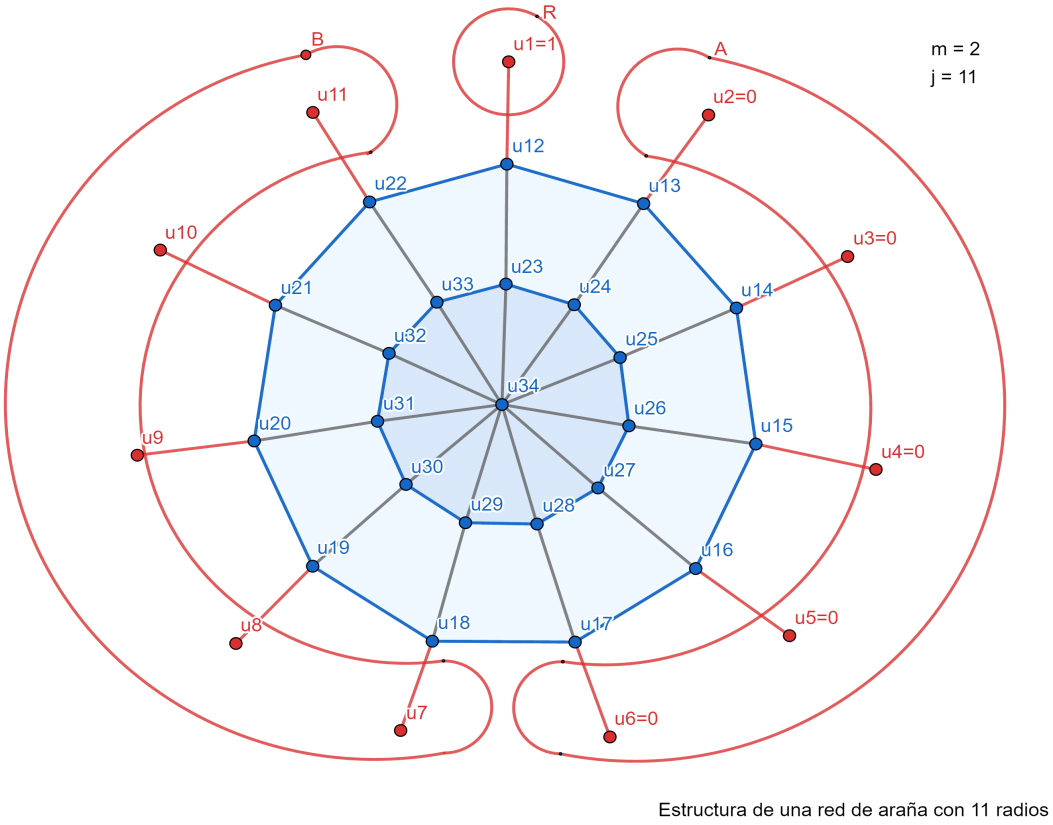


Figura 13: Recuperación de las conductancias radiales de la Spider de 11 rayos. Elaboración propia

El tercer paso a realizar, es encontrar la solución (uj) de la función valorada en los nodos del primer círculo (m=1). Esto es posible mediante el Lema 4.2 que se calcula mediante el valor del contorno y conductancia de los pasos anteriores:

```
% ESTUDIO DE VALORES DE uj EN EL PRIMER CÍRCULO.

m =1; % Se busca los valores uj en el primer círculo.

for j= 1:n

  B_j = [((n+1)/2+j):(n-1+j)]; %  Índices que pertenecen a las columnas de la matriz respuesta.

  if (n-1+j) > n & & ((n+1)/2+j)<=n % Si se supera los nodos del círculo, se reinicializará tal como se ha explicado anteriormente.

    B1 = (((n+1)/2+j):n);

    B2 = (1: (n-1+j)-n);

    B_j = [B1 B2];

  elseif(n-1+j) > n & & ((n+1)/2+j)>n

    B_j = [((n+1)/2+j)-n: (n-1+j)-n];

  end
```

```
  for k=1:n % Se busca los valores de uj en cada contorno R y por ello por cada contorno R, es decir
j, se estudiará todos los valores del círculo utilizando la variable auxiliar k que recorrerá todos los
nodos, es decir de 1 hasta n.

    uj(n* m+k,j)= 1/c(n* m+k, n* m+k-n)* (Lambda* uj(n* m+k-n,j)-Nq(k,j)-Nq(k,B_j)* UB_j_t(1:end,
j)) + w(n* m+k)/w(n* m+k-n)* uj(n* m+k-n,j);

  end

end
```

El procedimiento es ir moviendo la variable k que recorrerá los nodos del círculo (m=1) donde es constante el índice j que marca donde está contorno R(por lo tanto, el contorno A y B). Una vez recorridos todos los nodos del primer círculo, se mueve el contorno R, es decir la j, y se vuelve a realizar el mismo procedimiento de la variable k hasta que el contorno R haya recorrido todos los nodos del contorno.

Los resultados obtenidos son:

-0.0000	-0.2500	2.0000	-4.0000	0.5000	-0.0000	0.0000	0.0000	0.0000	-0.0000	-0.0000
-0.0000	-0.0000	-0.5000	4.0000	-2.0000	0.6000	0.0000	0.0000	0.0000	-0.0000	-0.0000
-0.0000	0.0000	-0.0000	-1.0000	2.0000	-2.4000	0.5000	0.0000	0.0000	-0.0000	-0.0000
-0.0000	0.0000	-0.0000	0.0000	-0.5000	2.4000	-2.0000	0.2000	0.0000	-0.0000	-0.0000
0.0000	0.0000	0.0000	0.0000	-0.0000	-0.6000	2.0000	-0.8000	1.0000	-0.0000	-0.0000
0.0000	0.0000	0.0000	0.0000	-0.0000	-0.0000	-0.5000	0.8000	-4.0000	0.3333	-0.0000
0.0000	0.0000	0.0000	0.0000	-0.0000	-0.0000	-0.0000	-0.2000	4.0000	-1.3333	2.0000
0.5000	0.0000	0.0000	0.0000	-0.0000	-0.0000	0.0000	-0.0000	-1.0000	1.5000	-8.5000
-3.0000	0.5000	0.0000	0.0000	-0.0000	-0.0000	0.0000	0.0000	-0.0000	-0.5000	8.0000
1.6667	-0.5625	0.1250	0.0000	-0.0000	-0.0000	0.0000	0.0000	0.0000	-0.0000	-1.0000
-0.3333	0.7500	-1.7500	1.0000	-0.0000	-0.0000	0.0000	0.0000	0.0000	-0.0000	-0.0000

Figura 14: Recuperación de los valores uj en el segundo círculo. Elaboración propia

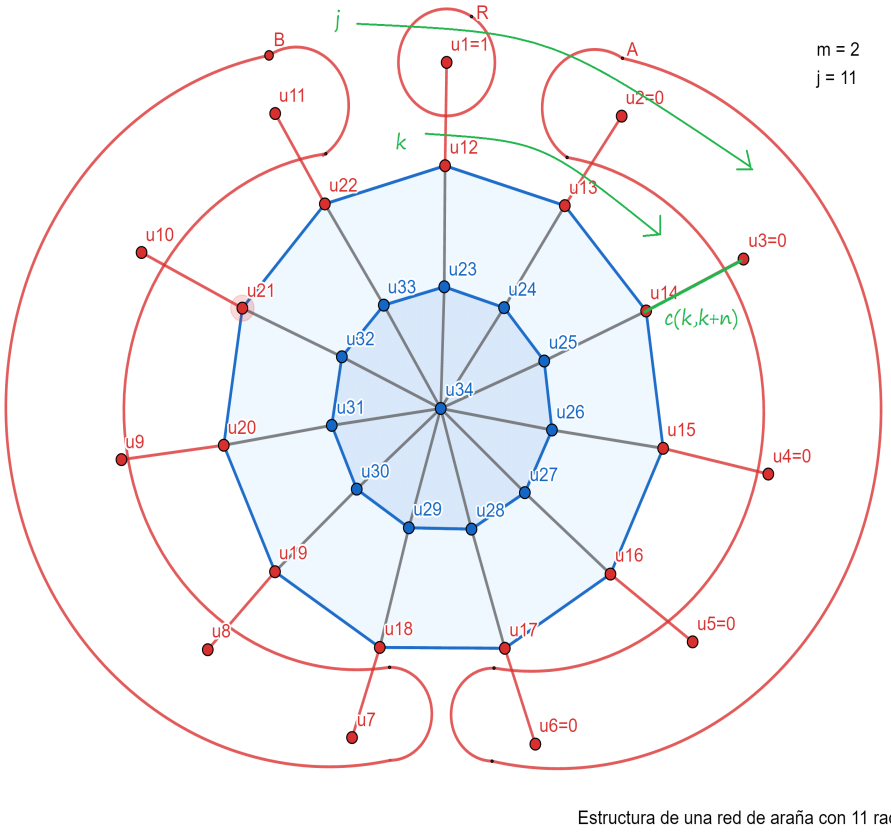


Figura 15: Recuperación de valores uj en el segundo círculo de la Spider de 11 rayos. Elaboración propia

El cuarto paso se basa en determinar los valores de las conductancias adyacentes que unen los nodos del primer círculo mediante la proposición 4.3. De esta proposición depende en qué nodo estamos se utiliza una expresión u otra:

% RECUPERACIÓN DE LAS CONDUCTANCIAS Y DE LOS VALORES uj EN LOS CÍRCULOS:

for m = 1: circulo % Se recorrerán todos los círculo que hay en el Spider

for j = 1:n % Se recorrerán todos los nodos con el contorno R

if m ==1 % En el caso que estemos en el primer círculo

if j== 1 % Cuando estamos en el primer nodo: la conductancia adyacente entre el primer nodo y el anterior que es el último del círculo.

c(n∗ m+j, n∗ m+n)= -uj(n∗ m-n+j,j)/uj(n∗ m+n,j)∗ c(n∗ m-n+j, n∗ m+j); % Se calcula la conductancia aplicando la Proposición 4.3.

c(n∗ m+n, n∗ m+j)= c(n∗ m+j, n∗ m+n); % Se impone la simetria para obtener la matriz de conductancias simétrica.

elseif(j>=2 & & j<=n) % Cuando estamos en otro nodo, es decir entre el segundo y el último, el cálculo de las conductancias es empleando la posición j para el nodo seleccionado y j-1 para el anterior.

```
        c(j+n* m,j+n* m-1)= -uj(n* m-n+j,j)/uj(j+n* m-1,j)* c(n* m-n+j,j+n* m);

        c(j+n* m-1, j+n* m)= c(j+n* m,j+n* m-1); % Se impone la simetría

    end

end
```

Este paso se debe realizar moviéndose por todos los nodos el primer círculo (i = 1:n) manteniendo constante el contorno R en un nodo y una vez estudiados todos los nodos se realiza el mismo considerado todos los contornos de R en todos los nodos (j=1:n).

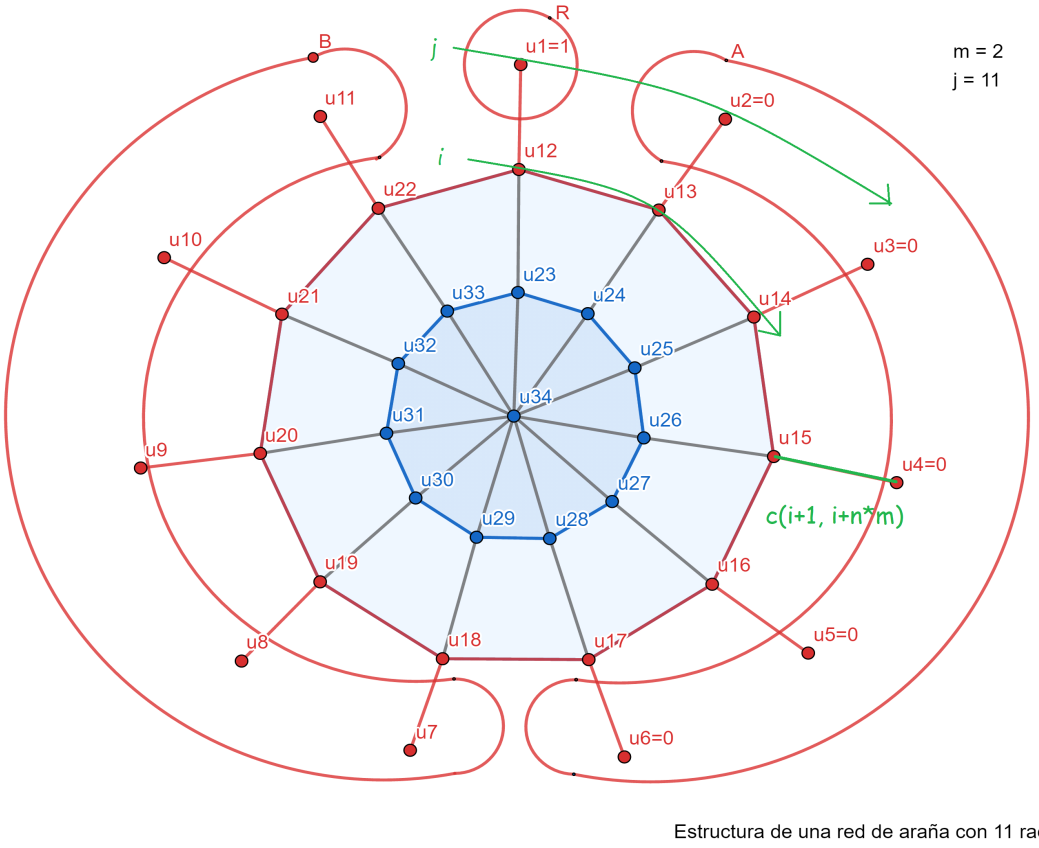


Figura 16: Recuperación de las conductancias adyacentes de la Spider de 11 rayos. Elaboración propia

El quinto paso a realizar es determinar el valor de las conductancias radiales que conectan el círculo m=1 y m=2:

Para ello, primero se explicará el operador lineal "p" :

Este operador calcula la suma de productos de las conductancias adyacentes c(A,B) y c(A,C) por el valor de la función valorada en los nodos A y C más la conductancia radial c(D,A) por la función valorada en el nodo D:

p(z)(xa) = c(A,B)* z(B) + c(A,C)* z(C) + c(A,D)* z(D)

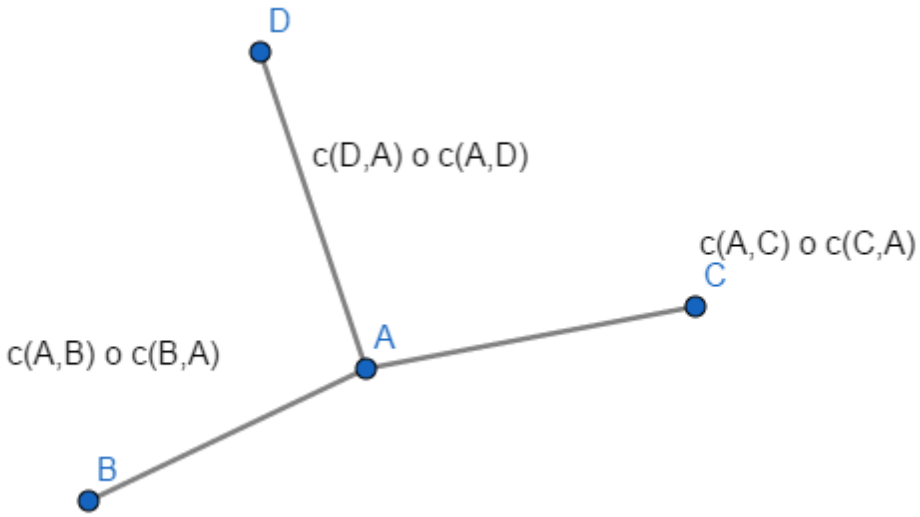


Figura 17: Representación gráfica de 3 nodos adyacentes con un nodo con conexión radial. Elaboración propia

Una vez explicado el operador "p", se calcula la conductancia radial mediante la proposición 4.4 que dependiendo en qué círculo estamos estudiando y depende de qué nodo, sus respectivas fórmulas varían:

- **En el caso que nos encontremos que estamos buscando una conductancias entre dos círculos donde el círculo interior no es el centro de la Spider:**

% RECUPERACIÓN DE LA CONDUCTANCIA RADIAL:

for t = 1:n % Se estudiará el recorrido del contorno R en todos los nodos n del contorno.

if m ∼ = circulo % Cuando nos encontramos en un círculo que no es el más interno, ya que el último círculo, las conductancias de los nodos de este círculo respecto el nodo central se calcula de otra manera que se especifica posteriormente.

if m ==1 % Cuando estamos en el primer círculo

if t==1 % Si estamos en el primer nodo del círculo, se estudia el nodo anterior que equivale al del último nodo del círculo. Al aplicar el operador "p" se valora la función en el nodo posterior, en el anterior y en el círculo anterior del mismo rayo. En la valoración en el nodo posterior, éste equivale al primer nodo del círculo, por ello se tiene que imponer n∗ m+1.

% Veamos un ejemplo más claro si suponemos que el contorno R está en el nodo 1:

% c(22,33)=((c(22, 21)∗ uj(21,1) + c(22, 12)∗ uj(12,1) + c(22, 11)∗ uj(11,1))/uj(22,1) - (c(22, 21)∗ w(21) + c(22, 12)∗ w(12) + c(22,11)∗ w(11))/w(22))∗ w(22)/w(33)

c(n∗ m+n,n∗ m+2∗ n)=((c(n∗ m+n, n∗ m+n-1)∗ uj(n∗ m+n-1,t) + c(n∗ m+n, n∗ m+1)∗ uj(n∗ m+1,t) + c(n∗ m+n, n∗ m)∗ uj(n∗ m,t))/uj(n∗ m+n,t) - (c(n∗ m+n, n∗ m+n-1)∗ w(n∗ m+n-1) + c(n∗ m+n, n∗ m+1)∗ w(n∗ m+1) + c(n∗ m+n, n∗ m)∗ w(n∗ m))/w(n∗ m+n))∗ w(n∗ m+n)/w(n∗ m+2∗ n);

c(n∗ m+2∗ n, n∗ m+n)= c(n∗ m+n,n∗ m+2∗ n); % Se impone la simetría.

```
elseif t ==2 % Si estamos en el segundo nodo del círculo, se estudia el nodo anterior que equivale al primer nodo del círculo. Al aplicar el operador "p" se valora la función en el nodo posterior, en el anterior y en el círculo anterior del mismo rayo. En la valoración en el nodo anterior, éste equivale al último nodo del círculo, por ello se tiene que imponer n* m+n.

% Veamos un ejemplo más claro si suponemos que el contorno R está en el nodo 1:

% c(12,23)=((c(12, 22)* uj(22,1) + c(12, 13)* uj(13,1) + c(12, 1)* uj(1,1))/uj(12,1) - (c(12, 22)* w(22) + c(12, 13)* w(13) + c(12,1)* w(1))/w(12))* w(12)/w(23)

c(n* m+1,n* m+1+n)=((c(n* m+1, n* m+n)* uj(n* m+n,t) + c(n* m+1, n* m+2)* uj(n* m+2,t) + c(n* m+1, n* m+1-n)* uj(n* m+1-n,t))/uj(n* m+1,t) - (c(n* m+1, n* m+n)* w(n* m+n) + c(n* m+1, n* m+2)* w(n* m+2) + c(n* m+1, n* m+1-n)* w(n* m+1-n))/w(n* m+1))* w(n* m+1)/w(n* m+1+n);

c(n* m+1+n, n* m+1)= c(n* m+1,n* m+1+n); % Se impone la simetría.


elseif (3 <=t) & & (t<=n) % Si estamos en el nodo t del círculo (que es entre el tercero y el último), al aplicar el operador "p" se valora la función en el nodo posterior(t+1), en el anterior(t-1) y en el círculo anterior del mismo rayo(t-n).

% Suponemos que el contorno R está en el nodo 1 y estamos tratando el nodo t=4 del círculo:

% c(15,26)=((c(15, 14)* va(14,1) + c(15, 16)* va(16,1) + c(15, 4)* va(4,1))/va(15,1) -(c(15, 14)* w(14) + c(15,16)* w(16) + c(15,4)* w(4))/w(15))* w(26)/w(15)

c(n* m+t-1,n* m+t-1+n)=((c(n* m+t-1, n* m+t-2)* uj(n* m+t-2,t) + c(n* m+t-1, n* m+t)* uj(n* m+t,t) + c(n* m+t-1, n* m+t-n-1)* uj(n* m+t-n-1,t))/uj(n* m+t-1,t) -(c(n* m+t-1, n* m+t-2)* w(n* m+t-2) + c(n* m+t-1, n* m+t)* w(n* m+t) + c(n* m+t-1, n* m+t-n-1)* w(n* m+t-n-1))/w(n* m+t-1))* w(n* m+t-1)/w(n* m+t-1+n); % Se calcula la conductancia radial aplicando la proposición 4.4.

c(n* m+t-1+n, n* m+t-1)= c(n* m+t-1,n* m+t-1+n); % Se impone la simetría.

    end

  end

end
```

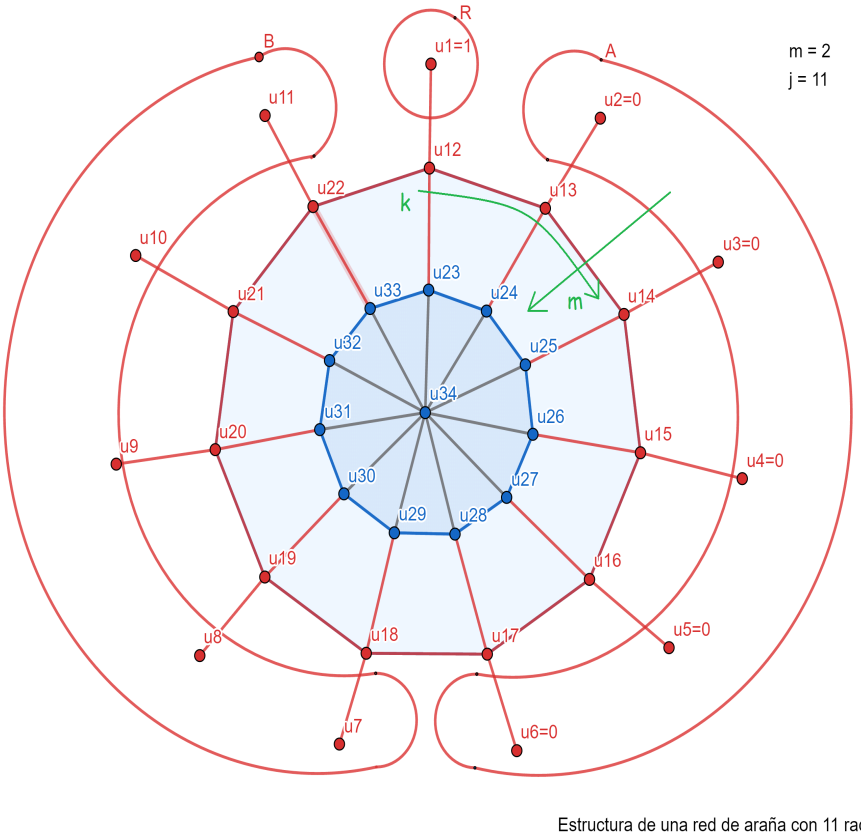


Figura 18: Recuperación de las conductancias radiales entre el primer círculo y el segundo. Elaboración propia

El sexto paso a realizar es valorar la función en los nodos del segundo círculo y para ello se empleará la proposición 4.5:

% PASO 6: SOLUCIÓN EN LOS NODOS INTERIORES

for r = 1: n % Se recorrerán todos los nodos con el contorno R

for s= 1:n % Se busca los valores de uj en cada contorno R y por ello por cada contorno R, es decir r, se estudiará todos los valores del círculo utilizando la variable auxiliar s que recorrerá todos los nodos, es decir de 1 hasta n.

if m ∼ =circulo % Se impone la condición que el círculo no sea el más interno, ya que se calcula los valores de uj de cada círculo desde el círculo anterior, por el cual el valor de uj en el círculo final es el valor uj del centro y no lo necesitamos para el cálculo de las conductancias.

if s==1 % En el caso que estemos en el primer nodo, al aplicar el operador "p" al nodo del mismo rayo pero en el círculo anterior, se valora la función en el nodo posterior, en el anterior y en el círculo anterior del mismo rayo. En la valoración en el nodo anterior, éste equivale al último nodo del círculo, por ello se tiene que imponer n∗ m+n.

% Veamos un ejemplo considerando que el contorno r se encuentra en el primer nodo: uj(22,1) =+(uj(12,1)∗ (c(12,22)∗ w(22) + c(12,13)∗ w(13) +c(12,1)∗ w(1)))/(w(12)∗ c(12,23))-(c(12, 22)∗ uj(22,1) + c(12, 13)∗ uj(13,1) + c(12,1)∗ uj(1,1))/c(12,23)+w(23)/w(12)∗ uj(12,1)

uj(n∗ m+1+n,r) = (uj(n∗ m+1,r)∗ (c(n∗ m+1, n∗ m+n)∗ w(n∗ m+n) + c(n∗ m+1, n∗ m+2)∗ w(n∗

```
m+2) + c(n* m+1, n* m+1-n)* w(n* m+1-n)))/(w(n* m+1)* c(n* m+1,n* m+1+n))-(c(n* m+1, n* m+n)*
uj(n* m+n,r) + c(n* m+1, n* m+2)* uj(n* m+2,r) + c(n* m+1, n* m+1-n)* uj(n* m+1-n,r))/c(n* m+1,n*
m+1+n)+w(n* m+1+n)/w(n* m+1)* uj(n* m+1,r); % Cálculo del valor uj mediante la proposición 4.5.

                if abs(uj(n* m+1+n,r))<0.0001

                    uj(n* m+1+n,r)= 0;

                end

            elseif s==n % En el caso que estemos en el último nodo, al aplicar el operador "p" al nodo
del mismo rayo pero en el círculo anterior, se valora la función en el nodo posterior, en el anterior y en el
círculo anterior del mismo rayo. En la valoración en el nodo posterior, éste equivale al primer nodo del
círculo, por ello se tiene que imponer n* m+1.

                % Veamos un ejemplo considerando que el contorno r se encuentra en el primer nodo: uj(33,1)
=+(uj(22,1)* (c(22,21)* w(21) + c(22,12)* w(12) +c(22,11)* w(11)))/(w(22)* c(22,33))-(c(22, 21)* uj(21,1) +
c(22, 12)* uj(12,1) + c(22, 11)* uj(11,1))/c(22,33)+w(33)/w(22)* uj(22,1)

                uj(n* m+2* n,r) =(uj(n* m+n,r)* (c(n* m+n, n* m+n-1)* w(n* m+n-1) + c(n* m+n, n* m+1)*
w(n* m+1) + c(n* m+n, n* m)* w(n* m)))/(w(n* m+n)* c(n* m+n,n* m+2* n))-(c(n* m+n, n* m+n-1)*
uj(n* m+n-1,r) + c(n* m+n, n* m+1)* uj(n* m+1,r) + c(n* m+n, n* m)* uj(n* m,r))/(c(n* m+n,n* m+2*
n))+w(n* m+2* n)/w(n* m+n)* uj(n* m+n,r); % Cálculo del valor uj mediante la proposición 4.5.

                if abs(uj(n* m+2* n,r))<0.0001

                    uj(n* m+2* n,r)= 0;

                end

            elseif (1 <s) & & (s<n) % Si estamos en el nodo s del círculo (que es entre el primero y el
último), al aplicar el operador "p" al nodo del mismo rayo pero en el círculo anterior, se valora la función
en el nodo posterior(s+1), en el anterior(s-1) y en el círculo anterior del mismo rayo(s-n).

                % Suponemos que el contorno R está en el nodo 1 y estamos tratando el nodo t=4 del
círculo: uj(26,1)=+uj(15,1)* (c(15, 14)* w(14) + c(15, 16)* w(16) + c(15, 4)* w(4))/(w(15)* c(15, 26))- (c(15,
14)* uj(14,1) + c(15, 16)* uj(16,1) + c(15, 4)* uj(4,1))/c(15, 26)+w(26)/w(15)* uj(15,1);

                uj(n* m+s+n,r)=uj(n* m+s,r)* (c(n* m+s, n* m+s-1)* w(n* m+s-1) + c(n* m+s, n* m+s+1)* w(n*
m+s+1) + c(n* m+s, n* m+s-n)* w(n* m+s-n))/(w(n* m+s)* c(n* m+s,n* m+s+n))- (c(n* m+s, n* m+s-1)*
uj(n* m+s-1,r) + c(n* m+s, n* m+s+1)* uj(n* m+s+1,r) + c(n* m+s, n* m+s-n)* uj(n* m+s-n,r))/c(n* m+s,
n* m+s+n)+w(n* m+s+n)/w(n* m+s)* uj(n* m+s,r); % Cálculo del valor uj mediante la proposición 4.5.


                if abs(uj(n* m+s+n,r))<0.0001

                    uj(n* m+s+n,r)= 0;

                end

            end

        end

    end
```

Los resultados obtenidos son:

```
         0         0    0.2500   -0.5000         0         0         0         0         0         0         0
         0         0         0    0.5000   -0.2500         0         0         0         0         0         0
         0         0         0         0    0.2500   -0.3000         0         0         0         0         0
         0         0         0         0         0    0.3000   -0.2500         0         0         0         0
         0         0         0         0         0         0    0.2500   -0.1000         0         0         0
         0         0         0         0         0         0         0    0.1000   -0.5000         0         0
         0         0         0         0         0         0         0         0    0.5000   -0.1667         0
         0         0         0         0         0         0         0         0         0    0.1667   -1.0000
   -0.1667         0         0         0         0         0         0         0         0         0    0.5000
    0.3333   -0.1250         0         0         0         0         0         0         0         0         0
         0    0.1250   -0.2500         0         0         0         0         0         0         0         0
```

Figura 19: Recuperación de los valores uj en el segundo círculo. Elaboración propia

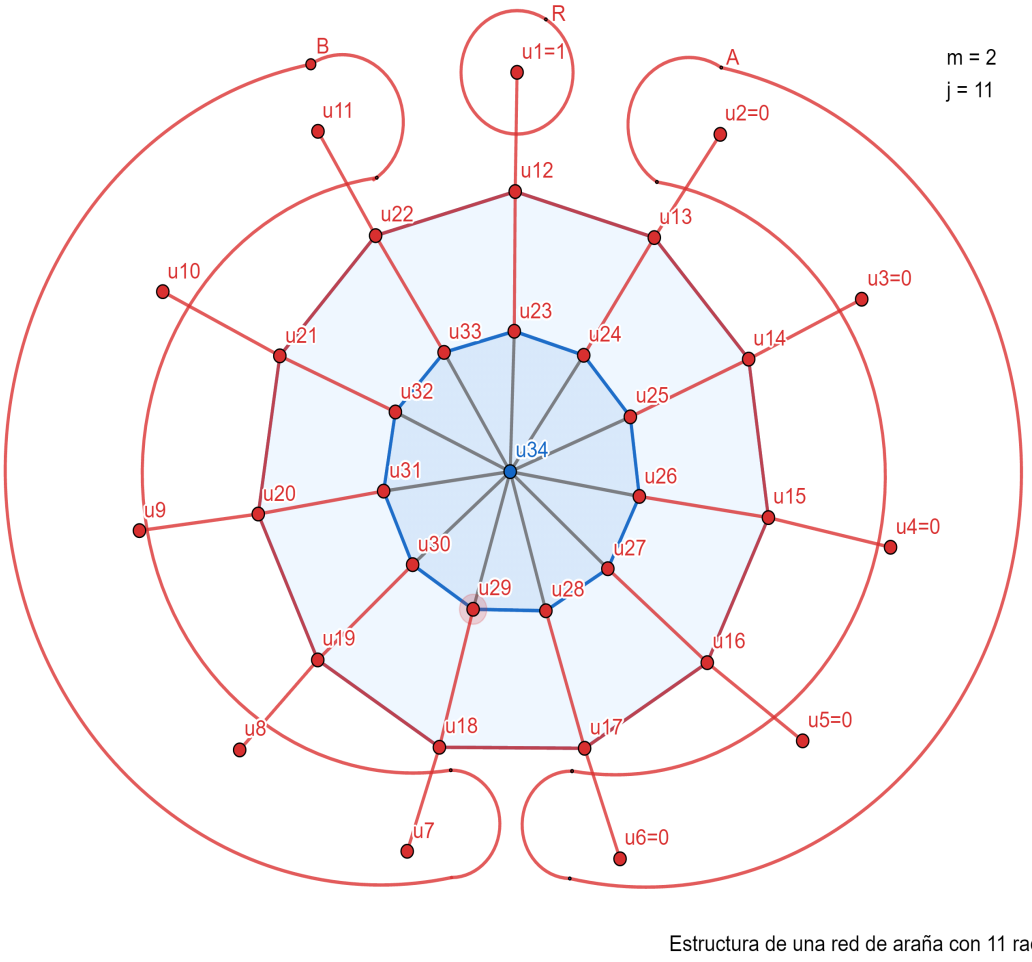


Figura 20: Recuperación de los valores uj en el segundo círculo de la Spider de 11 rayos. Elaboración propia

Una vez recuperadas las conductancias adyacentes, radiales y los valores de uj, se vuelve a empezar el mismo bucle pero para el círculo 2:

Se calcula las conductancias adyacentes para el segundo círculo que es superior a 1, por lo tanto, nos encontramos en el segundo condicional.

```
if m>1 % Cuando estamos en un círculo superior a 1 hay que ir retorciendo una posición en cada círculo. Este efecto se contempla restando (m-1), es decir, si estamos en el segundo círculo se retrocede una posición, pero si estamos en el tercero círculo, se retrocede dos posición y así sucesivamente.


    A= j+(n* m)-(m-1); % A será la variable que representará el nodo que se estudia que depende del círculo en que estemos y del contorno R.

     if A< (m)* n+1 %  Como que se tiene que retroceder una posición o las posiciones necesaria que indica m-1. En el caso que el nodo anterior sea inferior al primer nodo del círculo, se le suma los nodos n para volver al círculo.

       A= j+(n* m)-(m-1)+n;

    end

  B= A-1; % B es el nodo anterior a A, por ello se le resta una posición. En el caso que el nodo adyacente inferior al primer nodo del círculo, se impone el valor del último nodo del mismo círculo como en el caso anterior. Por ejemplo, si estamos en el nodo 24 (j=2) y estamos en el segundo círculo, el nodo A es 23 y el nodo B debería ser el 33, por ello hay que imponer el valor del nodo B.

    if B< (m)* n+1

       B= A-1+n;

    end

      c(A,B)= -uj(A-n,j)/uj(B,j)* c(A-n,A); % Se aplica la Proposición 4.3 para calcular el valor de la conductancia adyacente.

    c(B,A)=c(A,B);% Se impone la simetría.

  end
```

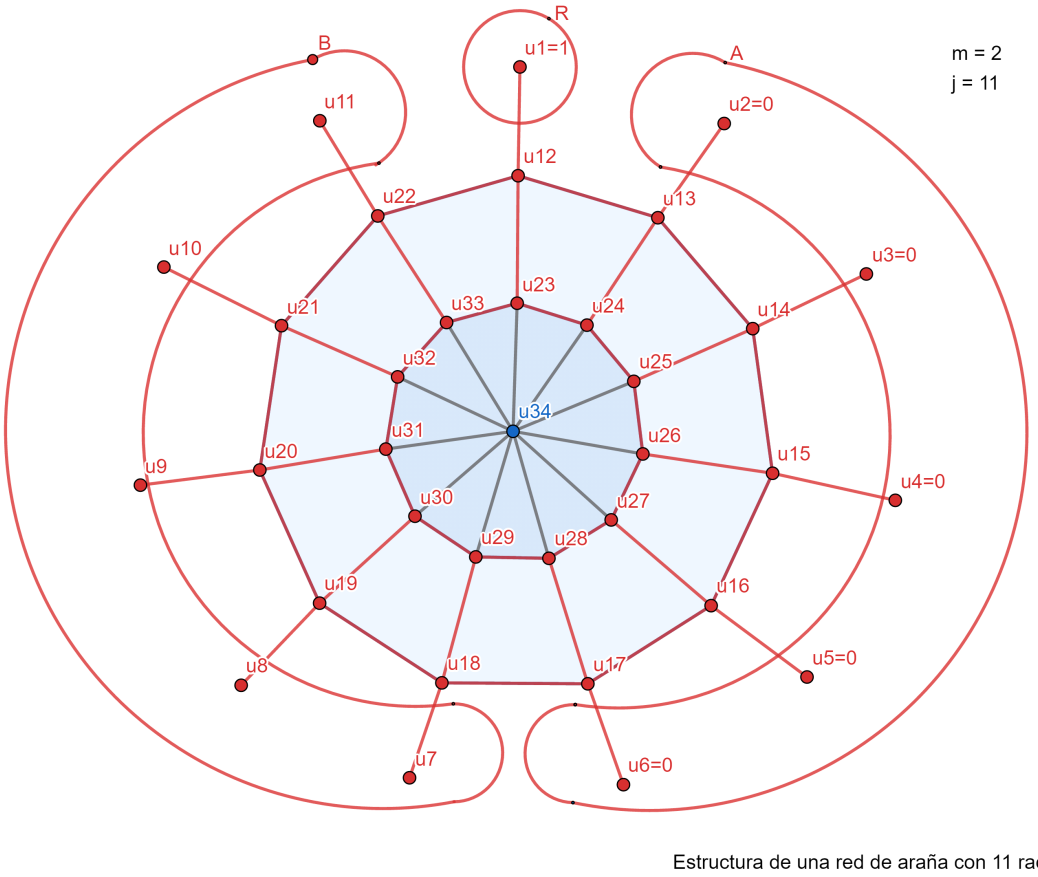


Figura 21: Recuperación de las conductancias adyacentes del segundo círculo de la Spider de 11 rayos. Elaboración propia

A continuación, calculamos las conductancias radiales: en este caso que nos encontremos que estamos buscando una conductancias entre dos círculos donde el círculo interior coincide con el centro de la Spider:

```
if m== circulo % Cuando estamos en el último círculo, es decir, el más interior, el nodo del siguiente círculo es común para todos los todos.

    Q = n* m + t-1 -(m-1); % El nodo a estudiar es Q, a medida que se avanza de círculo se retrocede un nodo sucesivamente.

    if Q< n* m +1 % Si al retroceder el número del nodo es inferior al primer nodo del círculo, se le suma n nodos para volver a estar en el mismo círculo.

        Q = n* m + t-1 -(m-1) +n;

    end

    Z= Q-1; % Z es el nodo anterior al nodo que se le aplica el operador "p". Si el nodo es inferior al primer nodo del círculo se impone que vuelva al mismo círculo del rayo correspondiente.

    if Z< n* m +1

        Z = Q-1+n;
```

```
        end

            G= Q+1; % G es el nodo posterior al nodo que se le aplica el operador "p". Si este nodo
supera el último nodo del círculo, se impone que es el primer nodo ddel mismo círculo.

        if G> (m+1)* n

            G = Q+1-n;

        end

            c(Q,n* circulo+n+1)=((c(Q,Z)* uj(Z,t) + c(Q, G)* uj(G,t) + c(Q, Q-n)* uj(Q-n,t))/uj(Q,t)
-(c(Q, Z)* w(Z) + c(Q,G)* w(G) + c(Q, Q-n)* w(Q-n))/w(Q))* w(Q)/w(n* circulo+n+1); % Se calcula la
conductancia radial aplicando la proposición 4.4.

            c(n* circulo+n+1, Q)= c(Q,n* circulo+n+1); % Se impone la simetría.

    end
```

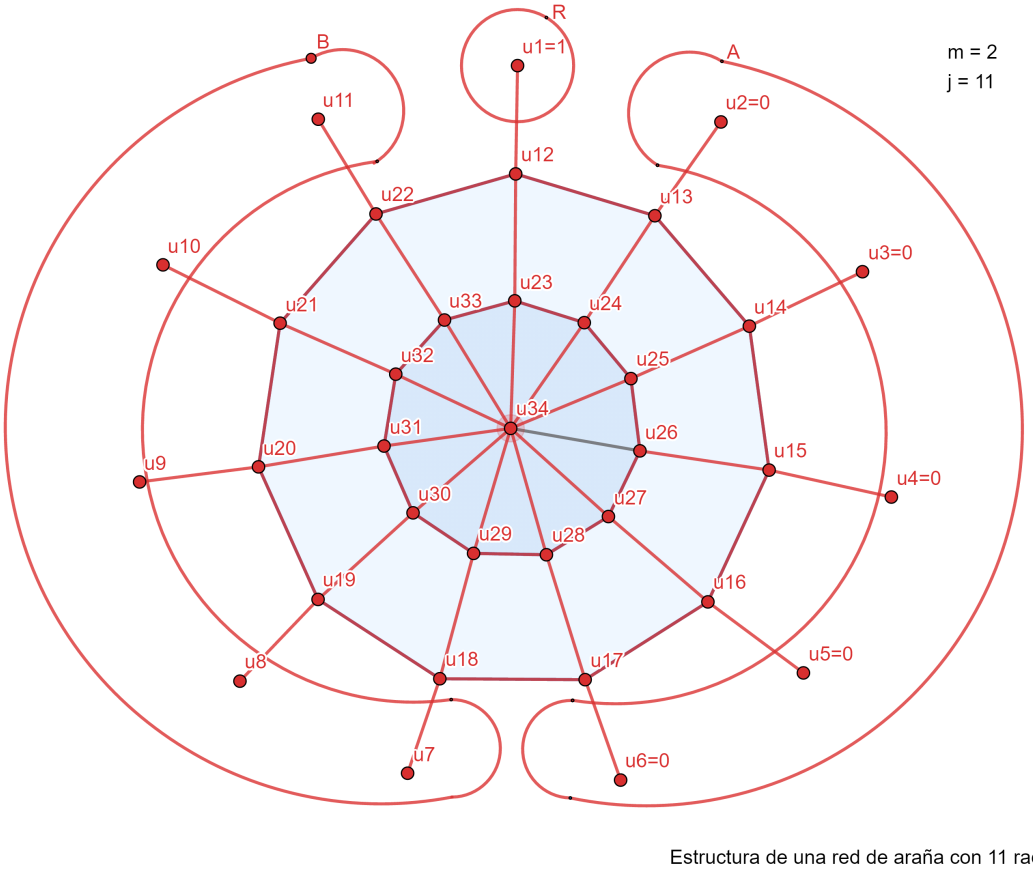


Figura 22: Recuperación de las conductancias radiales más internas de la Spider de 11 rayos. Elaboración propia

Finalmente, se impone que la matriz de conductancias sea simétrica y en la diagonal se ubicaran conductancias totales:

```
% Matriz simétrica con los nodos de la diagonal con la conductancia total
 for i= (1: n* circulo+n+1)
    sumafila= 0;
    for j = 1:n* circulo+n+1
       sumafila= sumafila + c(i,j);
    end
    for t =1:n* circulo+1+n
       if i==t
          c(i,t)= sumafila;
       end
    end
 end
 for i= (1: n* circulo+n+1) % Valores de las conductancias fuera de la diagonal negativas para que la matriz de las conductancias cumpla sus características.
    for j = 1:n* circulo+n+1
     if i ~= j
        c(i,j)= -c(i,j);
     end
    end
 end
```

El resultado es el siguiente:

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 8 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 7 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 9 & -5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5 & 11 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 9 & -5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5 & 9 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 7 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 6 & -2 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 7 & -3 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 10 & -2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 8 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

Figura 23: Matriz de conductancias recuperada. Elaboración propia

Estos valores obtenidos son exactos y se puede observar que se recuperan hasta las conductancias internas con exactitud:

-2.0000	0	0	0	0	0	0	0	0	0	0	0	0	0
0	-1.0000	0	0	0	0	0	0	0	0	0	0	0	0
0	0	-1.0000	0	0	0	0	0	0	0	0	0	0	0
0	0	-3.0000	-1.0000	0	0	0	0	0	0	0	0	0	0
0	0	0	0	-1.0000	0	0	0	0	0	0	0	0	0
0	0	0	0	0	-1.0000	0	0	0	0	0	0	0	0
0	0	0	0	0	0	-1.0000	0	0	0	0	0	0	0
0	0	0	0	0	0	0	-1.0000	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1.0000	0	0	0	0	0
0	0	0	0	0	0	0	0	0	-1.0000	0	0	0	0
-2.0000	0	0	0	0	0	0	0	0	0	-1.0000	0	0	0
7.0000	-2.0000	0	0	0	0	0	0	0	0	0	-1.0000	0	0
-2.0000	6.0000	-1.0000	0	0	0	0	0	0	0	0	0	-2.0000	0
0	-1.0000	6.0000	0	0	0	0	0	0	0	0	0	0	-1.0000
0	0	0	6.0000	-2.0000	0	0	0	0	0	0	0	-2.0000	-1.0000
0	0	0	-2.0000	6.0000	-2.0000	0	0	0	0	0	0	0	-1.0000
0	0	0	0	-2.0000	6.0000	-2.0000	0	0	0	0	0	0	-1.0000
0	0	0	0	0	-2.0000	6.0000	-2.0000	0	0	0	0	0	-1.0000
0	0	0	0	0	0	-2.0000	6.0000	-2.0000	0	0	0	0	-1.0000
0	0	0	0	0	0	0	-2.0000	6.0000	-2.0000	0	0	0	-1.0000
0	0	0	0	0	0	0	0	-2.0000	6.0000	-2.0000	0	0	-1.0000
0	0	0	0	0	0	0	0	0	-2.0000	7.0000	-3.0000	0	-1.0000
-1.0000	0	0	0	0	0	0	0	0	0	-3.0000	10.0000	-4.0000	0	-2.0000
0	-2.0000	0	0	0	0	0	0	0	0	0	-4.0000	8.0000	-1.0000	-1.0000
0	0	-1.0000	-2.0000	0	0	0	0	0	0	0	0	-1.0000	5.0000	-1.0000
0	0	0	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	-2.0000	-1.0000	-1.0000	12.0000

Figura 24: Matriz de conductancias recuperada con decimales de los últimos 14 nodos. Elaboración propia

9.2 Algoritmo aplicado a una Spider de 15 nodos

Aplicamos este mismo algoritmo a una Spider de 15 nodos con las siguientes características:

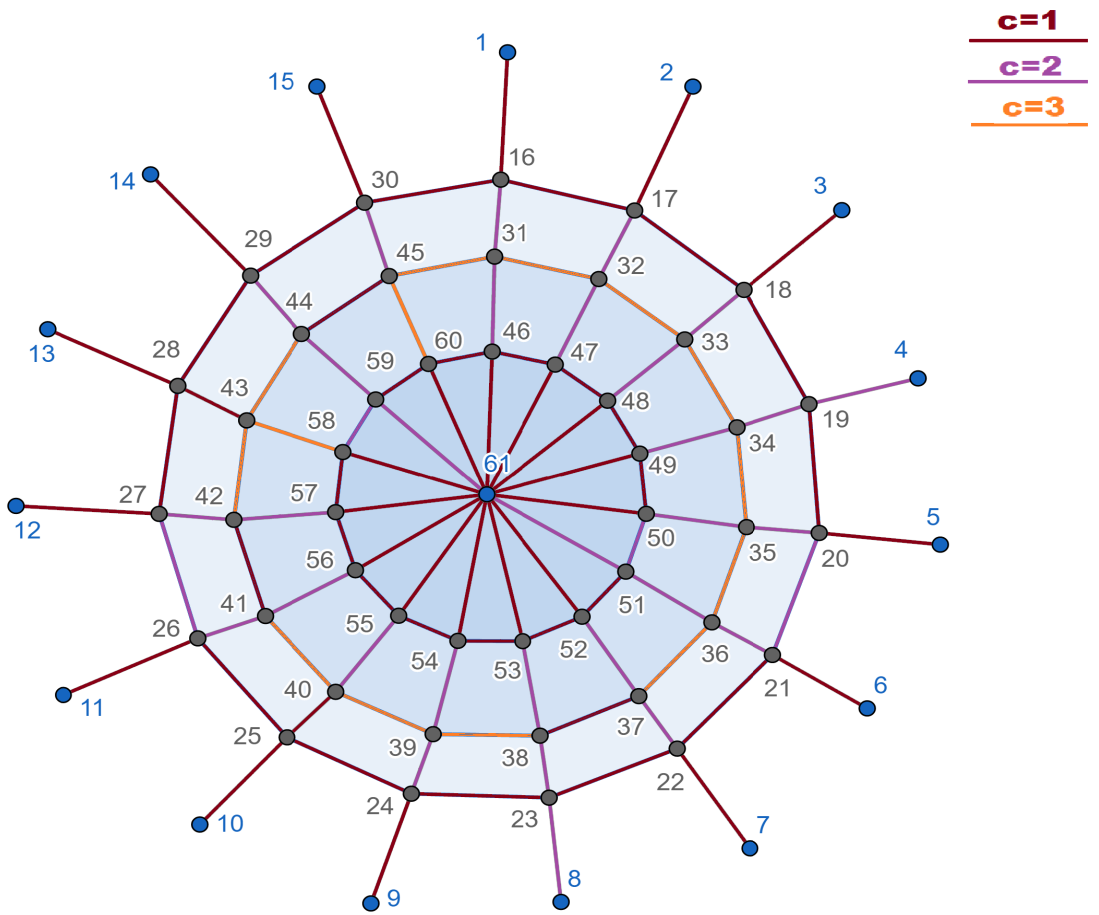


Figura 25: Spider de 15 nodos introducida en el MATLAB con las conductancias. Elaboración propia

El vector de peso es el siguiente:

```
Columns 1 through 27

   1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     2     2     2     2     2     2     2     2     2     2     2     2

Columns 28 through 54

   2     2     2     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1     2     2     2     2     2     2     2     2     2

Columns 55 through 61

   2     2     2     2     2     2     5
```

Figura 26: Vector peso para la Spider de 15 nodos. Elaboración propia

El valor de la Lambda es 3. Entonces la matriz de respuesta de la Spider es:

```
La matriz de respuesta es simétrica
    4.4999   -0.2369   -0.1297   -0.1519   -0.0660   -0.0625   -0.0570   -0.0979   -0.0546   -0.0550   -0.0638   -0.0698   -0.0861   -0.1306   -0.2380
   -0.2369    4.5040   -0.2281   -0.2195   -0.0788   -0.0694   -0.0591   -0.0972   -0.0529   -0.0520   -0.0583   -0.0613   -0.0672   -0.0886   -0.1347
   -0.1297   -0.2281    4.5208   -0.3816   -0.1085   -0.0858   -0.0647   -0.0983   -0.0513   -0.0494   -0.0541   -0.0557   -0.0570   -0.0683   -0.0885
   -0.1519   -0.2195   -0.3816    5.4124   -0.3557   -0.2460   -0.1541   -0.2011   -0.0965   -0.0894   -0.0953   -0.0965   -0.0949   -0.1070   -0.1230
   -0.0660   -0.0788   -0.1085   -0.3557    4.5645   -0.2689   -0.1401   -0.1502   -0.0624   -0.0540   -0.0550   -0.0548   -0.0524   -0.0570   -0.0608
   -0.0625   -0.0694   -0.0858   -0.2460   -0.2689    4.5565   -0.2083   -0.1975   -0.0733   -0.0593   -0.0577   -0.0567   -0.0535   -0.0575   -0.0599
   -0.0570   -0.0591   -0.0647   -0.1541   -0.1401   -0.2083    4.5256   -0.3706   -0.1089   -0.0745   -0.0629   -0.0590   -0.0534   -0.0559   -0.0568
   -0.0979   -0.0972   -0.0983   -0.2011   -0.1502   -0.1975   -0.3706    5.3898   -0.3863   -0.2170   -0.1454   -0.1238   -0.1027   -0.1016   -0.1002
   -0.0546   -0.0529   -0.0513   -0.0965   -0.0624   -0.0733   -0.1089   -0.3863    4.5193   -0.2315   -0.1229   -0.0929   -0.0676   -0.0610   -0.0573
   -0.0550   -0.0520   -0.0494   -0.0894   -0.0540   -0.0593   -0.0745   -0.2170   -0.2315    4.4774   -0.2238   -0.1493   -0.0915   -0.0704   -0.0603
   -0.0638   -0.0583   -0.0541   -0.0953   -0.0550   -0.0577   -0.0629   -0.1454   -0.1229   -0.2238    4.5438   -0.2809   -0.1508   -0.0984   -0.0746
   -0.0698   -0.0613   -0.0557   -0.0965   -0.0548   -0.0567   -0.0590   -0.1238   -0.0929   -0.1493   -0.2809    4.5435   -0.2256   -0.1297   -0.0876
   -0.0861   -0.0672   -0.0570   -0.0949   -0.0524   -0.0535   -0.0534   -0.1027   -0.0676   -0.0915   -0.1508   -0.2256    4.4742   -0.2418   -0.1297
   -0.1306   -0.0886   -0.0683   -0.1070   -0.0570   -0.0575   -0.0559   -0.1016   -0.0610   -0.0704   -0.0984   -0.1297   -0.2418    4.4991   -0.2313
   -0.2380   -0.1347   -0.0885   -0.1230   -0.0608   -0.0599   -0.0568   -0.1002   -0.0573   -0.0603   -0.0746   -0.0876   -0.1297   -0.2313    4.5026
```

Figura 27: Matriz de respuesta de la Spider de 15 nodos. Elaboración propia

El valor que determina la sensibilidad es de 121,1993.

Una vez ejecutado el algoritmo se obtiene los siguientes valores uj: (Estos valores están expresados $\cdot 10^3$):

- **En los nodos del contorno por cada contorno R ubicado en un nodo de los del contorno:**

0.0010	-0.0147	0.0522	-0.3009	0.1640	-0.0376	0.0147	-0.0010	0	0	0	0	0	0	0
0	0.0010	-0.0125	0.1468	-0.1664	0.0834	-0.0752	0.0147	-0.0060	0	0	0	0	0	0
0	0	0.0010	-0.0293	0.0753	-0.0834	0.1667	-0.0762	0.0940	-0.0010	0	0	0	0	0
0	0	0	0.0010	-0.0076	0.0202	-0.0925	0.0973	-0.2685	0.0071	-0.0002	0	0	0	0
0	0	0	0	0.0010	-0.0078	0.0874	-0.2509	1.5765	-0.1094	0.0134	-0.0010	0	0	0
0	0	0	0	0	0.0010	-0.0268	0.1594	-1.5100	0.1855	-0.0394	0.0053	-0.0010	0	0
0	0	0	0	0	0	0.0010	-0.0238	0.4015	-0.1297	0.0639	-0.0264	0.0350	-0.0005	0
0	0	0	0	0	0	0	0.0010	-0.0275	0.0261	-0.0354	0.0400	-0.1163	0.0038	-0.0007
-0.0030	0	0	0	0	0	0	0	0.0010	-0.0095	0.0338	-0.0858	0.5818	-0.0515	0.0367
0.0183	-0.0010	0	0	0	0	0	0	0	0.0010	-0.0086	0.0499	-0.7070	0.1157	-0.1367
-0.1336	0.0605	-0.0013	0	0	0	0	0	0	0	0.0010	-0.0127	0.3328	-0.1049	0.2889
0.3044	-0.3063	0.0202	-0.0040	0	0	0	0	0	0	0	0.0010	-0.0675	0.0457	-0.2850
-0.3427	0.6567	-0.0753	0.0231	-0.0002	0	0	0	0	0	0	0	0.0010	-0.0061	0.1281
0.1612	-0.4907	0.1130	-0.0934	0.0104	-0.0005	0	0	0	0	0	0	0	0.0010	-0.0323
-0.0235	0.1311	-0.1004	0.2543	-0.0691	0.0073	-0.0010	0	0	0	0	0	0	0	0.0010

Figura 28: Valores de uj en el contorno. Elaboración propia

- **En los nodos del primer círculo por cada contorno R ubicado en un nodo de los del contorno:**

0	-0.0010	0.0090	-0.0693	0.0362	-0.0056	0.0010	0	0	0	0	0	0	0	0
0	0	-0.0010	0.0223	-0.0365	0.0182	-0.0112	0.0010	0	0	0	0	0	0	0
0	0	0	-0.0020	0.0112	-0.0183	0.0365	-0.0112	0.0060	0	0	0	0	0	0
0	0	0	0	-0.0010	0.0056	-0.0365	0.0375	-0.0730	0.0010	0	0	0	0	0
0	0	0	0	0	-0.0005	0.0112	-0.0422	0.2470	-0.0102	0.0005	0	0	0	0
0	0	0	0	0	0	-0.0010	0.0168	-0.2260	0.0320	-0.0056	0.0005	0	0	0
0	0	0	0	0	0	0	-0.0020	0.0510	-0.0285	0.0139	-0.0031	0.0010	0	0
0	0	0	0	0	0	0	0	-0.0010	0.0060	-0.0139	0.0158	-0.0315	0.0005	0
0	0	0	0	0	0	0	0	0	-0.0010	0.0056	-0.0182	0.1095	-0.0056	0.0015
0.0030	0	0	0	0	0	0	0	0	0	-0.0010	0.0105	-0.1930	0.0322	-0.0315
-0.0093	0.0010	0	0	0	0	0	0	0	0	0	-0.0005	0.0315	-0.0158	0.0418
0.0455	-0.0280	0.0007	0	0	0	0	0	0	0	0	0	-0.0010	0.0031	-0.0412
-0.0942	0.1823	-0.0172	0.0040	0	0	0	0	0	0	0	0	0	-0.0010	0.0288
0.0200	-0.0923	0.0234	-0.0111	0.0002	0	0	0	0	0	0	0	0	0	-0.0010

Figura 29: Valores de uj en el primer círculo. Elaboración propia

- **En los nodos del segundo círculo por cada contorno R ubicado en un nodo de los del contorno:**

-0.0010	0.0112	-0.0211	0.0532	-0.0096	0.0005	0	0	0	0	0	0	0	0	0
0	0	0.0007	-0.0087	0.0043	-0.0003	0	0	0	0	0	0	0	0	0
0	0	0	0.0013	-0.0043	0.0022	-0.0007	0	0	0	0	0	0	0	0
0	0	0	0	0.0007	-0.0022	0.0043	-0.0007	0	0	0	0	0	0	0
0	0	0	0	0	0.0003	-0.0043	0.0043	-0.0040	0	0	0	0	0	0
0	0	0	0	0	0	0.0007	-0.0050	0.0300	-0.0007	0	0	0	0	0
0	0	0	0	0	0	0	0.0013	-0.0260	0.0037	-0.0003	0	0	0	0
0	0	0	0	0	0	0	0	0.0020	-0.0040	0.0022	-0.0003	0	0	0
0	0	0	0	0	0	0	0	0	0.0007	-0.0022	0.0022	-0.0020	0	0
0	0	0	0	0	0	0	0	0	0	0.0003	-0.0022	0.0130	-0.0003	0
0	0	0	0	0	0	0	0	0	0	0	0.0003	-0.0130	0.0022	-0.0010
-0.0010	0	0	0	0	0	0	0	0	0	0	0	0.0020	-0.0022	0.0065
0.0065	-0.0020	0	0	0	0	0	0	0	0	0	0	0	0.0003	-0.0065
-0.0053	0.0107	-0.0004	0	0	0	0	0	0	0	0	0	0	0	0.0007
0.0020	-0.0130	0.0037	-0.0013	0	0	0	0	0	0	0	0	0	0	0

Figura 30: Valores de uj en el segundo círculo. Elaboración propia

- **En los nodos del tercer círculo por cada contorno R ubicado en un nodo de los del contorno:**

0	0.0007	-0.0029	0.0062	-0.0004	0	0	0	0	0	0	0	0	0	0
0	0	0	-0.0027	0.0013	0	0	0	0	0	0	0	0	0	0
0	0	0	0	-0.0013	0.0007	0	0	0	0	0	0	0	0	0
0	0	0	0	0	-0.0007	0.0013	0	0	0	0	0	0	0	0
0	0	0	0	0	0	-0.0013	0.0013	0	0	0	0	0	0	0
0	0	0	0	0	0	0	-0.0013	0.0080	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-0.0040	0.0007	0	0	0	0	0
0	0	0	0	0	0	0	0	0	-0.0013	0.0007	0	0	0	0
0	0	0	0	0	0	0	0	0	0	-0.0007	0.0007	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-0.0007	0.0040	0	0
0	0	0	0	0	0	0	0	0	0	0	0	-0.0040	0.0007	0
0	0	0	0	0	0	0	0	0	0	0	0	0	-0.0007	0.0020
0.0020	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.0020
-0.0020	0.0040	0	0	0	0	0	0	0	0	0	0	0	0	0
0	-0.0020	0.0007	0	0	0	0	0	0	0	0	0	0	0	0
0	0	-0.0013	0.0027	0	0	0	0	0	0	0	0	0	0	0

Figura 31: Valores de uj en el tercer círculo. Elaboración propia

Finalmente, la matriz de conductancias recuperada redondeada es la siguiente:

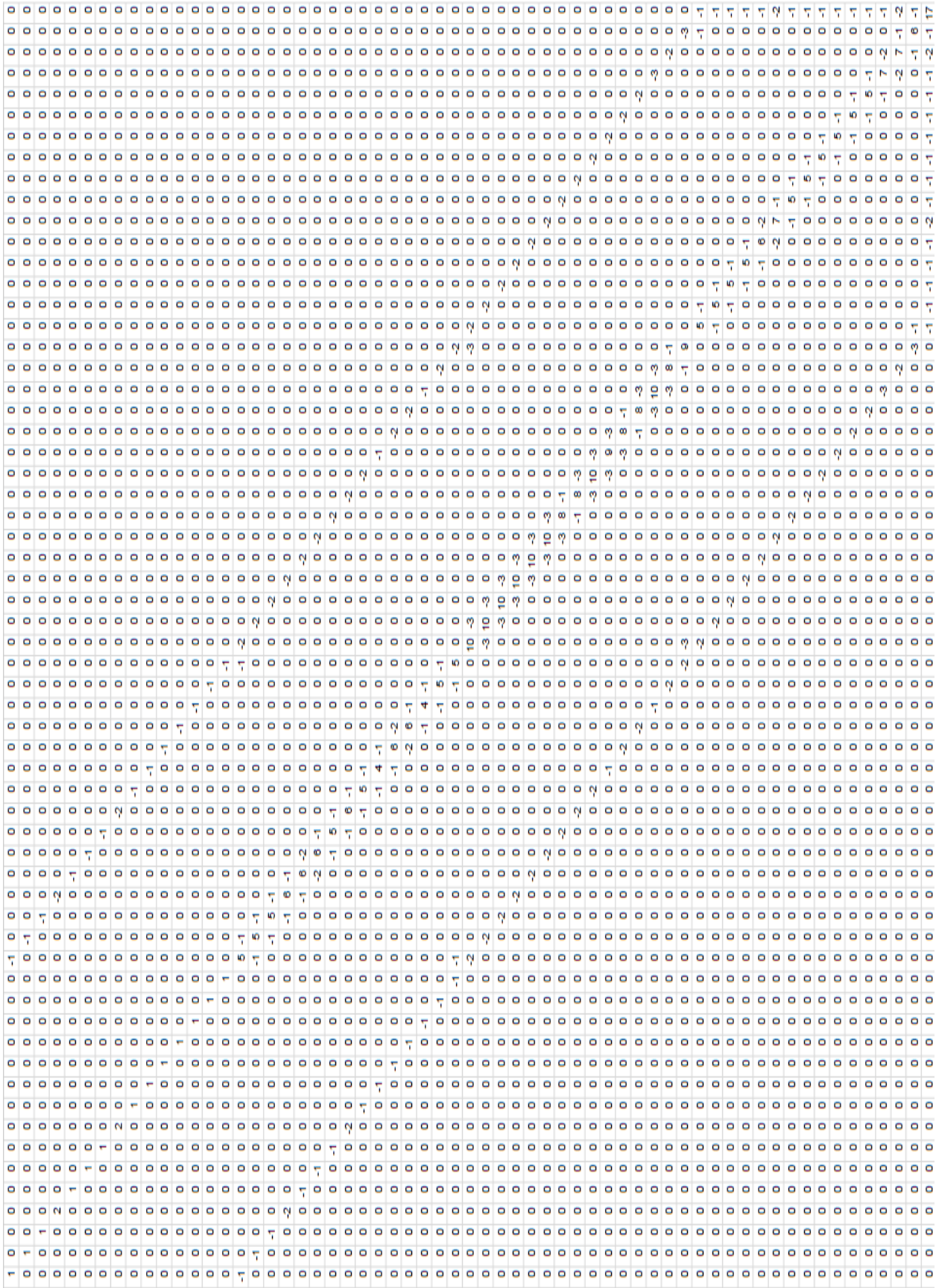


Figura 32: matriz de conductancias recuperada. Elaboración propia.

Un detalle relevante a destacar es que con las dimensiones de esta Spider en el último círculo las conductancias no se recuperan exactamente debido a la imprecisión al calcular la inversa en el complemento de Schur.

0	0	-2.0001	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	-2.0000	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	-2.0000	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	-2.0000	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	-2.0000	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	-2.0000	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	-1.9999	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	-2.0000	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	-2.0000	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-3.0000	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	-1.9999	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	-3.0000	0
-1.0000	0	0	0	0	0	0	0	0	0	0	0	0	-1.0000	-1.0000
4.9999	-0.9999	0	0	0	0	0	0	0	0	0	0	0	0	-1.0000
-0.9999	5.0002	-1.0002	0	0	0	0	0	0	0	0	0	0	0	-1.0001
0	-1.0002	4.9999	-0.9998	0	0	0	0	0	0	0	0	0	0	-0.9998
0	0	-0.9998	6.0000	-2.0001	0	0	0	0	0	0	0	0	0	-1.0001
0	0	0	-2.0001	7.0001	-1.0000	0	0	0	0	0	0	0	0	-2.0000
0	0	0	0	-1.0000	5.0001	-1.0001	0	0	0	0	0	0	0	-1.0001
0	0	0	0	0	-1.0001	5.0000	-1.0000	0	0	0	0	0	0	-1.0000
0	0	0	0	0	0	-1.0000	4.9999	-0.9999	0	0	0	0	0	-0.9999
0	0	0	0	0	0	0	-0.9999	5.0001	-1.0001	0	0	0	0	-1.0001
0	0	0	0	0	0	0	0	-1.0001	5.0000	-1.0000	0	0	0	-0.9999
0	0	0	0	0	0	0	0	0	-1.0000	5.0000	-1.0000	0	0	-1.0000
0	0	0	0	0	0	0	0	0	0	-1.0000	6.9998	-1.9998	0	-0.9999
0	0	0	0	0	0	0	0	0	0	0	-1.9998	7.0003	-1.0001	-2.0003
0	0	0	0	0	0	0	0	0	0	0	0	-1.0001	6.0001	-1.0000
-1.0000	-1.0001	-0.9998	-1.0001	-2.0000	-1.0001	-1.0000	-0.9999	-1.0001	-0.9999	-1.0000	-0.9999	-2.0003	-1.0000	17.0003

Figura 33: Conductancias recuperadas con 4 decimales de los últimos 15 nodos de la Spider. Elaboración propia

9.3 Algoritmo aplicado a una Spider de 19 nodos

Volvemos a aplicar el algoritmo a una Spider de 19 nodos con las siguientes características:

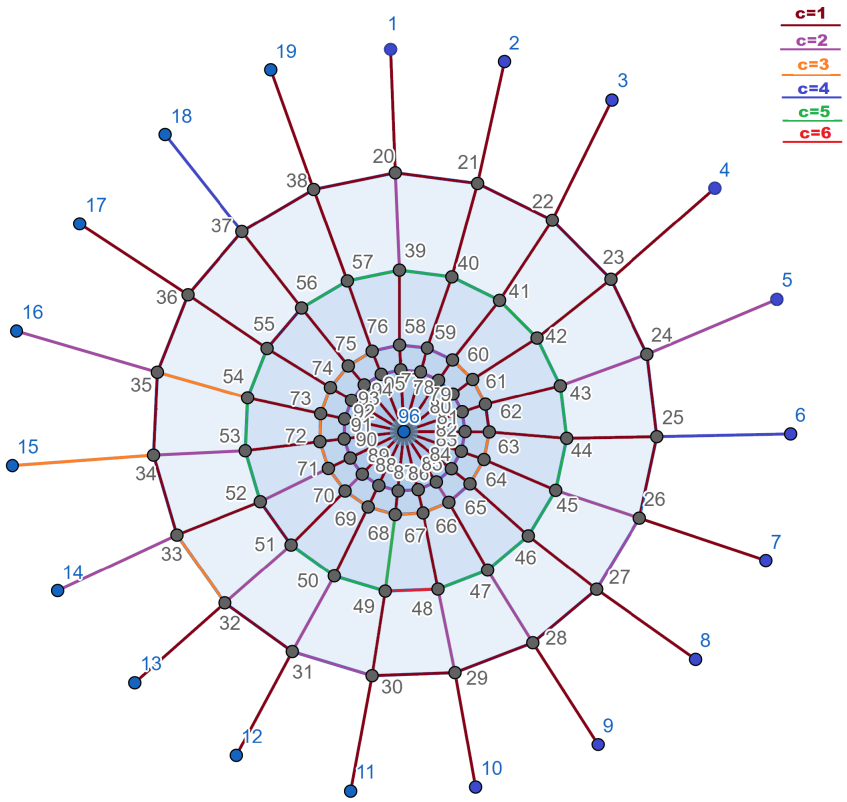


Figura 34: Spider de 19 nodos introducida en el MATLAB con las conductancias. Elaboración propia

El vector de peso es el siguiente:

```
Peso= [ 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1;
        1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1;
        1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 7; 2; 7; 4;
        2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 8;
        2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 2; 9; 10;];
```

Figura 35: Vector peso para la Spider de 19 nodos. Elaboración propia

Se impone que el valor de la Lambda es 1 y entonces la matriz de respuesta de la Spider es:

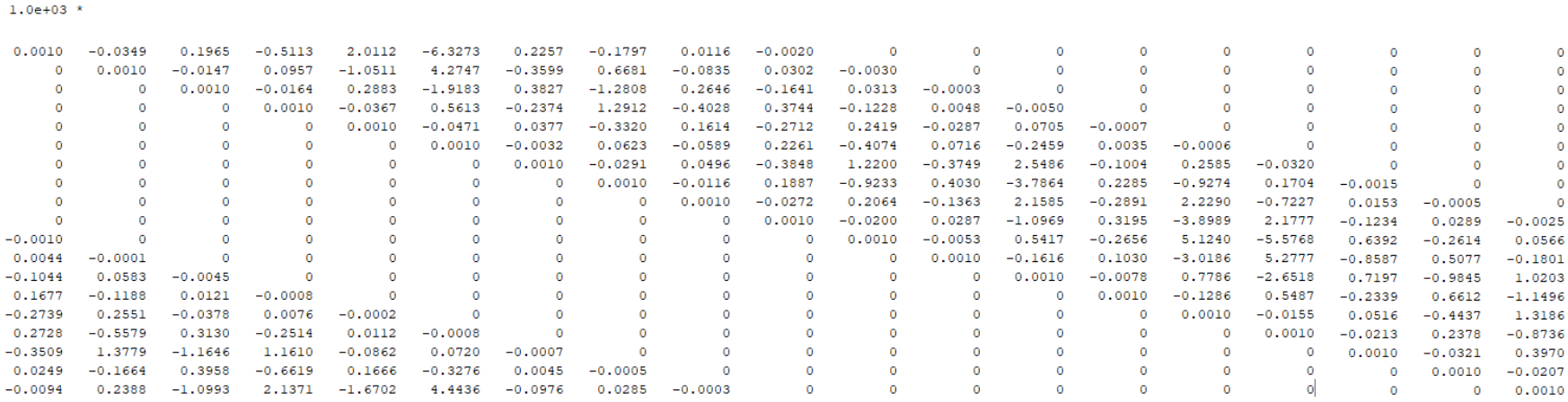
1.7420	-0.0888	-0.0399	-0.0240	-0.0334	-0.0372	-0.0172	-0.0169	-0.0196	-0.0202	-0.0186	-0.0197	-0.0182	-0.0309	-0.0535	-0.0689	-0.0297	-0.1350	-0.0704
-0.0888	1.6841	-0.1042	-0.0422	-0.0400	-0.0366	-0.0150	-0.0143	-0.0160	-0.0162	-0.0147	-0.0154	-0.0141	-0.0237	-0.0401	-0.0513	-0.0217	-0.0932	-0.0366
-0.0399	-0.1042	1.6793	-0.1033	-0.0669	-0.0468	-0.0158	-0.0144	-0.0152	-0.0149	-0.0133	-0.0139	-0.0125	-0.0208	-0.0344	-0.0437	-0.0183	-0.0756	-0.0254
-0.0240	-0.0422	-0.1033	1.6871	-0.1485	-0.0760	-0.0187	-0.0159	-0.0155	-0.0146	-0.0127	-0.0130	-0.0116	-0.0190	-0.0309	-0.0390	-0.0162	-0.0656	-0.0205
-0.0334	-0.0400	-0.0669	-0.1485	2.1610	-0.3246	-0.0498	-0.0378	-0.0322	-0.0283	-0.0237	-0.0240	-0.0210	-0.0341	-0.0542	-0.0682	-0.0281	-0.1123	-0.0340
-0.0372	-0.0366	-0.0468	-0.0760	-0.3246	2.4215	-0.1875	-0.1074	-0.0612	-0.0443	-0.0334	-0.0326	-0.0274	-0.0435	-0.0673	-0.0841	-0.0345	-0.1363	-0.0407
-0.0172	-0.0150	-0.0158	-0.0187	-0.0498	-0.1875	1.7562	-0.1213	-0.0503	-0.0292	-0.0194	-0.0179	-0.0144	-0.0223	-0.0333	-0.0413	-0.0169	-0.0662	-0.0196
-0.0169	-0.0143	-0.0144	-0.0159	-0.0378	-0.1074	-0.1213	1.7179	-0.0896	-0.0408	-0.0234	-0.0202	-0.0154	-0.0233	-0.0336	-0.0414	-0.0169	-0.0659	-0.0195
-0.0196	-0.0160	-0.0152	-0.0155	-0.0322	-0.0612	-0.0503	-0.0896	1.7247	-0.0918	-0.0409	-0.0308	-0.0209	-0.0302	-0.0406	-0.0492	-0.0200	-0.0776	-0.0229
-0.0202	-0.0162	-0.0149	-0.0146	-0.0283	-0.0443	-0.0292	-0.0408	-0.0918	1.7265	-0.0884	-0.0516	-0.0277	-0.0372	-0.0441	-0.0519	-0.0210	-0.0806	-0.0237
-0.0186	-0.0147	-0.0133	-0.0127	-0.0237	-0.0334	-0.0194	-0.0234	-0.0409	-0.0884	1.7195	-0.1234	-0.0441	-0.0516	-0.0457	-0.0494	-0.0198	-0.0749	-0.0220
-0.0197	-0.0154	-0.0139	-0.0130	-0.0240	-0.0326	-0.0179	-0.0202	-0.0308	-0.0516	-0.1234	1.7493	-0.0739	-0.0797	-0.0551	-0.0539	-0.0214	-0.0795	-0.0233
-0.0182	-0.0141	-0.0125	-0.0116	-0.0210	-0.0274	-0.0144	-0.0154	-0.0209	-0.0277	-0.0441	-0.0739	1.7747	-0.2147	-0.0857	-0.0553	-0.0213	-0.0747	-0.0217
-0.0309	-0.0237	-0.0208	-0.0190	-0.0341	-0.0435	-0.0223	-0.0233	-0.0302	-0.0372	-0.0516	-0.0797	-0.2147	2.1923	-0.2501	-0.1059	-0.0392	-0.1291	-0.0370
-0.0535	-0.0401	-0.0344	-0.0309	-0.0542	-0.0673	-0.0333	-0.0336	-0.0406	-0.0441	-0.0457	-0.0551	-0.0857	-0.2501	2.4924	-0.2458	-0.0809	-0.2322	-0.0650
-0.0689	-0.0513	-0.0437	-0.0390	-0.0682	-0.0841	-0.0413	-0.0414	-0.0492	-0.0519	-0.0494	-0.0539	-0.0553	-0.1059	-0.2458	2.5573	-0.1192	-0.3049	-0.0840
-0.0297	-0.0217	-0.0183	-0.0162	-0.0281	-0.0345	-0.0169	-0.0169	-0.0200	-0.0210	-0.0198	-0.0214	-0.0213	-0.0392	-0.0809	-0.1192	1.7544	-0.1912	-0.0381
-0.1350	-0.0932	-0.0756	-0.0656	-0.1123	-0.1363	-0.0662	-0.0659	-0.0776	-0.0806	-0.0749	-0.0795	-0.0747	-0.1291	-0.2322	-0.3049	-0.1912	3.1952	-0.2004
-0.0704	-0.0366	-0.0254	-0.0205	-0.0340	-0.0407	-0.0196	-0.0195	-0.0229	-0.0237	-0.0220	-0.0233	-0.0217	-0.0370	-0.0650	-0.0840	-0.0381	-0.2004	1.8047

Figura 36: Matriz de respuesta de la Spider de 19 nodos. Elaboración propia

El valor que determina la sensibilidad es de 1,7134·10^3.

Una vez ejecutado el algoritmo se obtiene los siguientes valores uj: (Estos valores están expresados · 10^3):

- **En los nodos del contorno por cada contorno R ubicado en un nodo de los del contorno:**

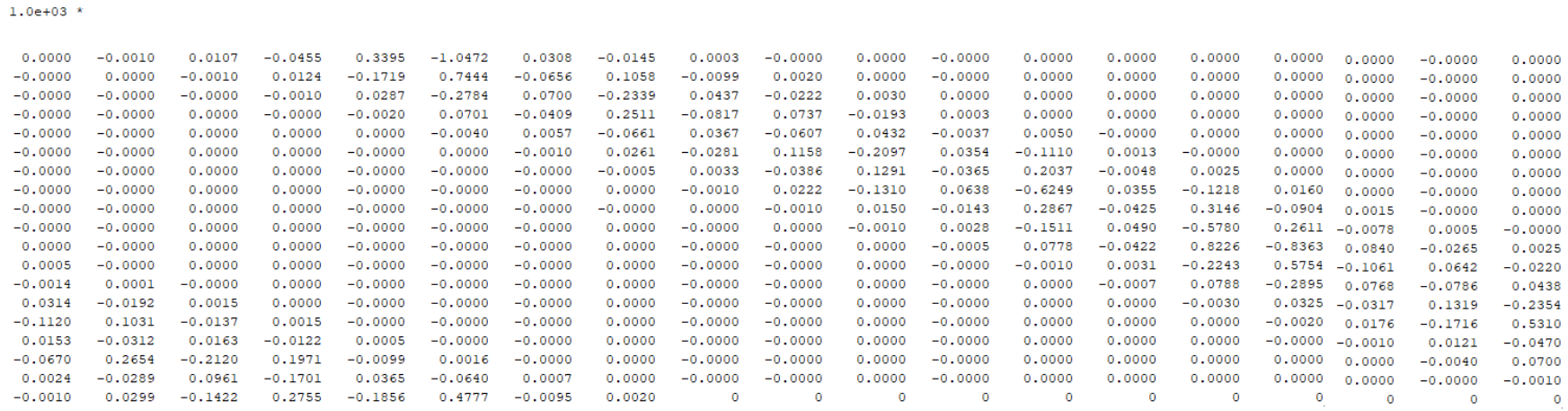


```
1.0e+03 *

 0.0010   -0.0349    0.1965   -0.5113    2.0112   -6.3273    0.2257   -0.1797    0.0116   -0.0020         0         0         0         0         0         0         0         0         0
      0    0.0010   -0.0147    0.0957   -1.0511    4.2747   -0.3599    0.6681   -0.0835    0.0302   -0.0030         0         0         0         0         0         0         0         0
      0         0    0.0010   -0.0164    0.2883   -1.9183    0.3827   -1.2808    0.2646   -0.1641    0.0313   -0.0003         0         0         0         0         0         0         0
      0         0         0    0.0010   -0.0367    0.5613   -0.2374    1.2912   -0.4028    0.3744   -0.1228    0.0048   -0.0050         0         0         0         0         0         0
      0         0         0         0    0.0010   -0.0471    0.0377   -0.3320    0.1614   -0.2712    0.2419   -0.0287    0.0705   -0.0007         0         0         0         0         0
      0         0         0         0         0    0.0010   -0.0032    0.0623   -0.0589    0.2261   -0.4074    0.0716   -0.2459    0.0035   -0.0006         0         0         0         0
      0         0         0         0         0         0    0.0010   -0.0291    0.0496   -0.3848    1.2200   -0.3749    2.5486   -0.1004    0.2585   -0.0320         0         0         0
      0         0         0         0         0         0         0    0.0010   -0.0116    0.1887   -0.9233    0.4030   -3.7864    0.2285   -0.9274    0.1704   -0.0015         0         0
      0         0         0         0         0         0         0         0    0.0010   -0.0272    0.2064   -0.1363    2.1585   -0.2891    2.2290   -0.7227    0.0153   -0.0005         0
      0         0         0         0         0         0         0         0         0    0.0010   -0.0200    0.0287   -1.0969    0.3195   -3.8989    2.1777   -0.1234    0.0289   -0.0025
-0.0010         0         0         0         0         0         0         0         0         0    0.0010   -0.0053    0.5417   -0.2656    5.1240   -5.5768    0.6392   -0.2614    0.0566
 0.0044   -0.0001         0         0         0         0         0         0         0         0         0    0.0010   -0.1616    0.1030   -3.0186    5.2777   -0.8587    0.5077   -0.1801
-0.1044    0.0583   -0.0045         0         0         0         0         0         0         0         0         0    0.0010   -0.0078    0.7786   -2.6518    0.7197   -0.9845    1.0203
 0.1677   -0.1188    0.0121   -0.0008         0         0         0         0         0         0         0         0         0    0.0010   -0.1286    0.5487   -0.2339    0.6612   -1.1496
-0.2739    0.2551   -0.0378    0.0076   -0.0002         0         0         0         0         0         0         0         0         0    0.0010   -0.0155    0.0516   -0.4437    1.3186
 0.2728   -0.5579    0.3130   -0.2514    0.0112   -0.0008         0         0         0         0         0         0         0         0         0    0.0010   -0.0213    0.2378   -0.8736
-0.3509    1.3779   -1.1646    1.1610   -0.0862    0.0720   -0.0007         0         0         0         0         0         0         0         0         0    0.0010   -0.0321    0.3970
 0.0249   -0.1664    0.3958   -0.6619    0.1666   -0.3276    0.0045   -0.0005         0         0         0         0         0         0         0         0         0    0.0010   -0.0207
-0.0094    0.2388   -1.0993    2.1371   -1.6702    4.4436   -0.0976    0.0285   -0.0003         0         0         0         0         0         0         0         0         0    0.0010
```

Figura 37: Valores de uj en el contorno. Elaboración propia

- **En los nodos del primer círculo por cada contorno R ubicado en un nodo de los del contorno:**



```
1.0e+03 *

  0.0000   -0.0010    0.0107   -0.0455    0.3395   -1.0472    0.0308   -0.0145    0.0003   -0.0000    0.0000   -0.0000    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0000    0.0000
 -0.0000    0.0000   -0.0010    0.0124   -0.1719    0.7444   -0.0656    0.1058   -0.0099    0.0020    0.0000   -0.0000    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0000    0.0000
 -0.0000   -0.0000   -0.0000   -0.0010    0.0287   -0.2784    0.0700   -0.2339    0.0437   -0.0222    0.0030    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0000    0.0000
 -0.0000   -0.0000    0.0000   -0.0000   -0.0020    0.0701   -0.0409    0.2511   -0.0817    0.0737   -0.0193    0.0003    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0000    0.0000
 -0.0000   -0.0000    0.0000    0.0000    0.0000   -0.0040    0.0057   -0.0661    0.0367   -0.0607    0.0432   -0.0037    0.0050   -0.0000    0.0000    0.0000    0.0000   -0.0000    0.0000
 -0.0000   -0.0000    0.0000    0.0000   -0.0000    0.0000   -0.0010    0.0261   -0.0281    0.1158   -0.2097    0.0354   -0.1110    0.0013   -0.0000    0.0000    0.0000   -0.0000    0.0000
 -0.0000   -0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000   -0.0005    0.0033   -0.0386    0.1291   -0.0365    0.2037   -0.0048    0.0025    0.0000    0.0000   -0.0000    0.0000
 -0.0000   -0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0010    0.0222   -0.1310    0.0638   -0.6249    0.0355   -0.1218    0.0160    0.0000   -0.0000    0.0000
 -0.0000   -0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0010    0.0150   -0.0143    0.2867   -0.0425    0.3146   -0.0904    0.0015   -0.0000    0.0000
 -0.0000   -0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0010    0.0028   -0.1511    0.0490   -0.5780    0.2611   -0.0078    0.0005   -0.0000
  0.0000   -0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0005    0.0778   -0.0422    0.8226   -0.8363    0.0840   -0.0265    0.0025
  0.0005   -0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000   -0.0010    0.0031   -0.2243    0.5754   -0.1061    0.0642   -0.0220
 -0.0014    0.0001   -0.0000    0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0007    0.0788   -0.2895    0.0768   -0.0786    0.0438
  0.0314   -0.0192    0.0015    0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000    0.0000   -0.0030    0.0325   -0.0317    0.1319   -0.2354
 -0.1120    0.1031   -0.0137    0.0015   -0.0000   -0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000    0.0000    0.0000   -0.0020    0.0176   -0.1716    0.5310
  0.0153   -0.0312    0.0163   -0.0122    0.0005   -0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000    0.0000    0.0000   -0.0000   -0.0010    0.0121   -0.0470
 -0.0670    0.2654   -0.2120    0.1971   -0.0099    0.0016   -0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0040    0.0700
  0.0024   -0.0289    0.0961   -0.1701    0.0365   -0.0640    0.0007    0.0000   -0.0000   -0.0000    0.0000   -0.0000    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0000   -0.0010
 -0.0010    0.0299   -0.1422    0.2755   -0.1856    0.4777   -0.0095    0.0020         0         0         0         0         0         0         0         0         0         0         0
```

Figura 38: Valores de uj en el primer círculo. Elaboración propia

- **En los nodos del segundo círculo por cada contorno R ubicado en un nodo de los del contorno:**

1.0e+03 *

0	0	0.0002	-0.0019	0.0218	-0.0654	0.0016	-0.0004	-0.0000	0	0	0	0	0	0	0	0	0	0
0	0	0	0.0002	-0.0047	0.0286	-0.0031	0.0035	-0.0001	-0.0000	0.0000	0	0	0	0	0	0	0	0
0	0	0	0	0.0004	-0.0099	0.0036	-0.0118	0.0016	-0.0004	0.0000	0	0	0	0	0	0	0	0
0	0	0	0	0	0.0016	-0.0017	0.0131	-0.0044	0.0034	-0.0006	0.0000	-0.0000	0	0	0	0	0	0
0	0	0	0	0	0	0.0002	-0.0047	0.0034	-0.0055	0.0023	-0.0001	-0.0000	0	0	0	0	0	0
0	0	0	0	0	0	0	0.0002	-0.0008	0.0055	-0.0103	0.0013	-0.0020	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0.0002	-0.0034	0.0131	-0.0036	0.0170	-0.0003	0.0000	0	0	0	0
0	0	0	0	0	0	0	0	0	0.0004	-0.0049	0.0032	-0.0320	0.0012	-0.0010	0.0000	0	0	0
0	0	0	0	0	0	0	0	0	0	0.0004	-0.0009	0.0256	-0.0039	0.0217	-0.0032	-0.0000	0	0
0	0	0	0	0	0	0	0	0	0	0	0.0001	-0.0115	0.0051	-0.0641	0.0272	-0.0006	-0.0000	0.0000
0	0	0	0	0	0	0	0	0	0	0	0	0.0004	-0.0005	0.0156	-0.0167	0.0011	-0.0002	0.0000
-0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0003	-0.0253	0.0677	-0.0113	0.0045	-0.0005
-0.0002	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0030	-0.0237	0.0095	-0.0127	0.0074
0.0009	-0.0002	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0008	-0.0018	0.0080	-0.0111
-0.0044	0.0034	-0.0003	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0006	-0.0068	0.0219
0.0056	-0.0112	0.0029	-0.0006	0.0000	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0.0008	-0.0095
-0.0017	0.0091	-0.0079	0.0070	-0.0003	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0010
0.0002	-0.0050	0.0206	-0.0369	0.0032	-0.0016	0.0000	0	0	0	0	0	0	0	0	0	0	0	0
0	0.0004	-0.0031	0.0067	-0.0050	0.0116	-0.0001	0	0	0	0	0	0	0	0	0	0	0	0

Figura 39: Valores de uj en el segundo círculo. Elaboración propia

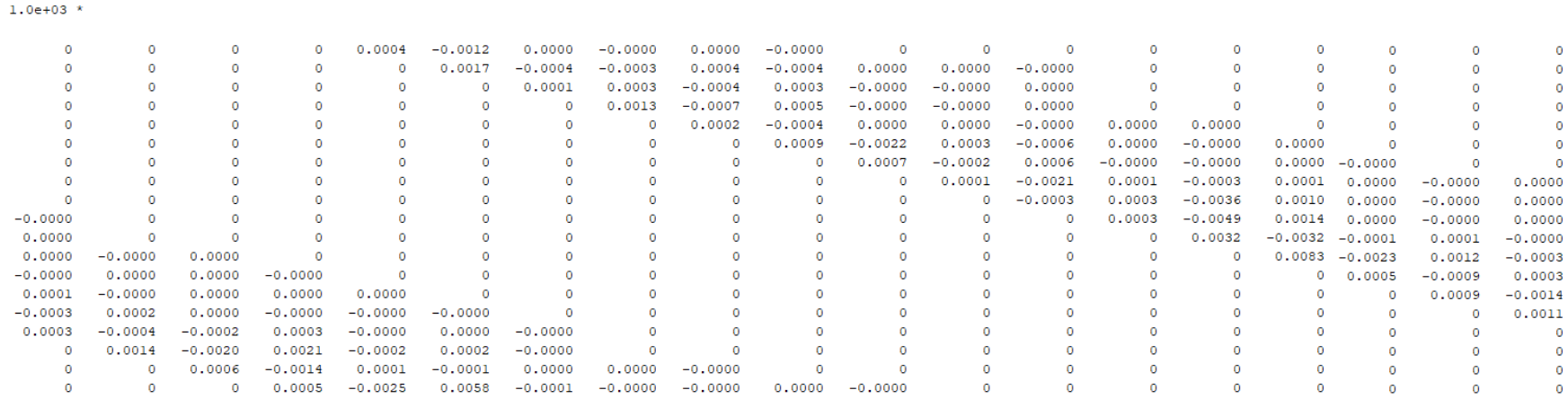
- **En los nodos del tercer círculo por cada contorno R ubicado en un nodo de los del contorno:**

1.0e+03 *

0	0	0	-0.0001	0.0014	-0.0041	0.0001	-0.0000	-0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0
0	0	0	0	-0.0001	0.0021	-0.0002	0.0001	-0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0
0	0	0	0	0	-0.0004	0.0003	-0.0006	-0.0001	0.0001	-0.0000	-0.0000	0.0000	0	0	0	0	0	0
0	0	0	0	0	0	-0.0002	0.0023	-0.0008	0.0005	-0.0000	-0.0000	0.0000	0	0	0	0	0	0
0	0	0	0	0	0	0	-0.0002	0.0004	-0.0009	0.0004	-0.0000	0.0001	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0.0001	-0.0005	0.0015	-0.0003	0.0007	-0.0000	-0.0000	0	0	0	0
0	0	0	0	0	0	0	0	0	-0.0002	0.0015	-0.0004	0.0011	-0.0000	0.0000	-0.0000	0	0	0
0	0	0	0	0	0	0	0	0	0	-0.0002	0.0004	-0.0042	0.0001	0.0001	-0.0000	0.0000	0	0
0	0	0	0	0	0	0	0	0	0	0	-0.0000	0.0026	-0.0004	0.0014	-0.0003	-0.0000	0.0000	-0.0000
0	0	0	0	0	0	0	0	0	0	0	0	-0.0001	0.0002	-0.0022	-0.0003	-0.0000	0.0000	-0.0000
0.0000	0	0	0	0	0	0	0	0	0	0	0	0	-0.0001	0.0052	-0.0046	-0.0001	0.0001	-0.0000
0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0	-0.0022	0.0113	-0.0022	0.0008	-0.0001
0.0000	-0.0000	0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0001	0.0002	-0.0002	-0.0003
0.0001	-0.0000	-0.0000	0.0000	0	0	0	0	0	0	0	0	0	0	0	0	-0.0002	0.0017	-0.0022
-0.0003	0.0001	-0.0000	0.0000	-0.0000	0.0000	0	0	0	0	0	0	0	0	0	0	0	-0.0002	0.0016
0.0004	-0.0007	0.0000	0.0001	-0.0000	0.0000	0	0	0	0	0	0	0	0	0	0	0	0	-0.0003
-0.0001	0.0011	-0.0009	0.0006	-0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0
0	-0.0001	0.0008	-0.0014	0.0000	0.0002	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0
0	0	-0.0002	0.0006	-0.0009	0.0017	-0.0000	-0.0000	0.0000	0	0	0	0	0	0	0	0	0	0

Figura 40: Valores de uj en el tercer círculo. Elaboración propia

- **En los nodos del cuarto círculo por cada contorno R ubicado en un nodo de los del contorno:**



1.0e+03 *

0	0	0	0	0.0004	-0.0012	0.0000	-0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0
0	0	0	0	0	0.0017	-0.0004	-0.0003	0.0004	-0.0004	0.0000	0.0000	-0.0000	0	0	0	0	0	0
0	0	0	0	0	0	0.0001	0.0003	-0.0004	0.0003	-0.0000	-0.0000	0.0000	0	0	0	0	0	0
0	0	0	0	0	0	0	0.0013	-0.0007	0.0005	-0.0000	-0.0000	0.0000	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0.0002	-0.0004	0.0000	0.0000	-0.0000	0.0000	0.0000	0	0	0	0
0	0	0	0	0	0	0	0	0	0.0009	-0.0022	0.0003	-0.0006	0.0000	-0.0000	0.0000	0	0	0
0	0	0	0	0	0	0	0	0	0	0.0007	-0.0002	0.0006	-0.0000	-0.0000	0.0000	-0.0000	0	0
0	0	0	0	0	0	0	0	0	0	0	0.0001	-0.0021	0.0001	-0.0003	0.0001	0.0000	-0.0000	0.0000
0	0	0	0	0	0	0	0	0	0	0	0	-0.0003	0.0003	-0.0036	0.0010	0.0000	-0.0000	0.0000
-0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0003	-0.0049	0.0014	0.0000	-0.0000	0.0000
0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0032	-0.0032	-0.0001	0.0001	-0.0000
0.0000	-0.0000	0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0083	-0.0023	0.0012	-0.0003
-0.0000	0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0005	-0.0009	0.0003
0.0001	-0.0000	0.0000	0.0000	0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0009	-0.0014
-0.0003	0.0002	0.0000	-0.0000	-0.0000	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0.0011
0.0003	-0.0004	-0.0002	0.0003	-0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0
0	0.0014	-0.0020	0.0021	-0.0002	0.0002	-0.0000	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0.0006	-0.0014	0.0001	-0.0001	0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0	0	0
0	0	0	0.0005	-0.0025	0.0058	-0.0001	-0.0000	-0.0000	0.0000	-0.0000	0	0	0	0	0	0	0	0

Figura 41: Valores de uj en el cuarto círculo. Elaboración propia

Finalmente, la matriz de conductancias recuperada redondeada es la siguiente:

Figura 42: Primera parte(filas:1-59 columnas:1-96) de la matriz recuperada. Elaboración propia

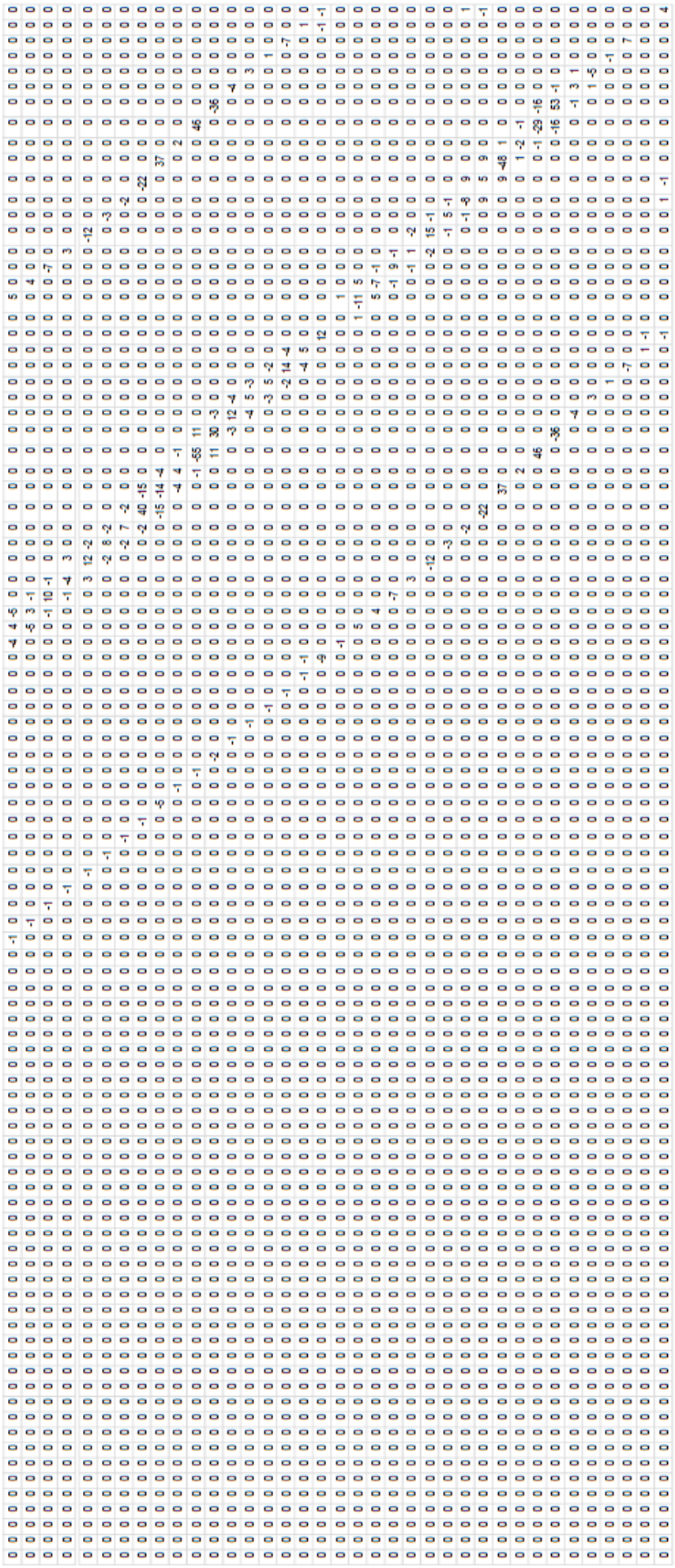


Figura 43: Segunda parte(filas:60-96 columnas:1-96) de la matriz recuperada. Elaboración propia

En este caso, la recuperación exacta de las conductancias falla en el primer círculo y al emplear éstas en los cálculos posteriores, las conductancias más interiores no tienen que ver con las conductancias iniciales.

Este hecho de recuperación inexacta de las conductancias también lo destaca el artículo "***Resistor network approaches to electrical impedance tomography***" **[1] de Liliana Borcea, Vladimir Druskin, Fernando Guevara Vasquez y Alexander V. Mamonov.** El artículo afirma que debido al mal acondicionamiento de estas matrices, la recuperación de las matrices de conductancias puede ser negativa.
No obstante, en principio para recuperar las conductancias se parte de una matriz de respuesta conocida, es decir, que no se tiene que calcular. En estos casos, las calculamos porque partimos de conductancias conocidas y calculamos la matriz de respuesta para recuperar estas conductancias y comprobar la funcionalidad del algoritmo. Es un problema a nivel numérico pero que se genera al calcular la matriz de respuesta, esto no debería ocurrir, ya que la matriz de respuesta se construye a partir de medidas experimentales en el laboratorio.

10 Algoritmo para la aplicación Dirichlet en redes Spider

Los objetivos de esta sección son la descripción de la aplicación Dirichlet-to-Neumann en una Spider cuyas conductancias radiales y circulares son ambas constantes, y su implementación mediante un código en Matlab. Los resultados teóricos se encuentran en el trabajo *Boundary value problems on layered product networks* ([14]), mientras que el algoritmo forma parte de los nuevos resultados obtenidos en este Trabajo Fin de Grado.

La obtención explícita de la matriz de respuesta en una red general está ligada al cálculo del complemento de Schur de una cierta submatriz del operador de Schrödinger. Como he comprobado en los diferentes ejemplos testados, incluso para tamaños moderados de redes las matrices involucradas están mal condicionadas, lo que conduce a graves problemas de inestabilidad numérica. Sin embargo, cuando la red tiene suficiente estructura, como en el caso de Spider con conductancias constantes, es posible obtener explícitamente la expresión de la aplicación Dirichlet-to-Neumann. Desde el punto de vista del problema inverso, la situación planteada puede parecer sin gran interés ya que solamente se han de recuperar dos valores de la conductancia, pero es un buen test para comprobar fortalezas y debilidades de nuestro algoritmo.

Para obtener la expresión de la aplicación es necesario conocer primero la función de Green para el problema de Dirichlet en una red Spider. En el artículo mencionado, se obtiene dicha expresión en términos de la función de Green de un cilindro que puede verse como el producto de un camino y un ciclo, ver la Figura 44. Los autovalores y autofunciones del camino se conocen solo cuando dicho camino tiene conductancias constantes lo que nos da una idea de la dificultad del problema, y lo mismo ocurre con la función de Green de un ciclo.

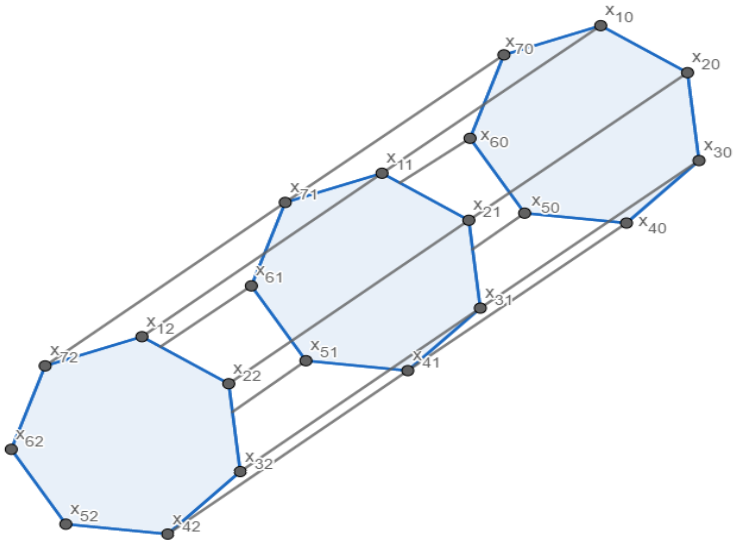


Figura 44: El cilindro formado por caminos y círculo en 3D. Elaboración propia

En el caso de redes cilíndricas, se considera el producto de las redes $\Gamma_1 \times \Gamma_2 = (V_c, c_c)$.
El problema de Dirichlet a resolver es

$$\left.\begin{array}{rrr} \mathcal{L}_\beta^S(u) = f & en & F_S \\ u = 0 & en & \delta(F_S) \end{array}\right\} \tag{113}$$

donde $\beta \geq 0$. Este problema es equivalente al de Dirichlet en un cilindro

$$\left.\begin{array}{rrr} \mathcal{L}_\beta(u_C) = f & en & F_S \\ u_C = \alpha \mathcal{X}_{H_0} & en & \delta(F_S) \end{array}\right\} \tag{114}$$

con la condición

$$\alpha = \frac{f(x_{00}^S) + \sum_{t=1}^{n} c(x_{t0}, x_{t1}) u_C(x_{t1})}{\beta + \sum_{t=1}^{n} c(x_{t0}, x_{t1})}. \tag{115}$$

La única solución al problema (114) está dada por

$$u_C = \mathcal{G}_\beta(f\mathcal{X}_F) + \alpha \sum_{t=1}^{n} c(x_{t0}, x_{t1}) \mathcal{G}_\beta(\epsilon_{x_{t1}}) + \alpha \mathcal{X}_{H_0}. \tag{116}$$

en $\bar{F}$, donde $\mathcal{G}_\beta$ es la función de Green para el problema de Dirichlet en un cilindro. Además el valor de α en este caso es

$$\alpha = \frac{f(x_{00}^S) + \sum_{t=1}^{n} c(x_{t0}, x_{t1}) \mathcal{G}_\beta(f\mathcal{X}_F)(x_{t1})}{\beta + \sum_{t=1}^{n} c(x_{t0}, x_{t1}) \left[1 - \sum_{r=1}^{n} c(x_{r0}, x_{r1}) \mathcal{G}_\beta(x_{r0}, x_{r1})\right]}. \tag{117}$$

Entonces, el problema (114) es equivalente a resolver el siguiente problema semi-homogéneo

$$\left.\begin{array}{rrr} \mathcal{L}_\beta(v) = f - \mathcal{L}_\beta(\alpha \mathcal{X}_{H_0}) & en & F_S, \\ v = 0 & en & \delta(F_S), \end{array}\right\} \tag{118}$$

entonces $u_c = v + \alpha \mathcal{X}_{H_1}$ en $\bar{F}$ es la solución del problema (114). Esta técnica se le llama 'levantar el dato'.

- **Proposición 1.1** La función de Green de $\Gamma_1 \times \Gamma_2 = (V_c, c_c)$ respecto los pesos $w_1 \otimes w_2$ y con un valor de $\beta > 0$ está dado por

$$G_{qc}(x_{ij},x_{lk}) = \frac{2}{m+1}\sum_{r=1}^{m}\frac{sin(\frac{j\pi r}{m+1})\ sin(\frac{k\pi r}{m+1})[U_{n-1-|i-l|}(1+\frac{\beta+\lambda_r}{2c_1}) + U_{|i-l|-1}(1+\frac{\beta+\lambda_r}{2c_1})}{[2c_1T_n(1+\frac{\beta+\lambda_r}{2c1})-1]} \quad (119)$$

donde $l = 1,...,n, k = 1,...,m, i = 1,...,n, j = 0,...,m+1$, $\lambda_r = 2c_2(1-cos(\frac{\pi r}{m+1}))$ y U_k, T_k son los polinomios de Chebyshev de segunda y primera especie, respectivamente. Como se puede observar, en los límites (cuando $j = 0$ y $j = m+1$) la función de Green es 0.

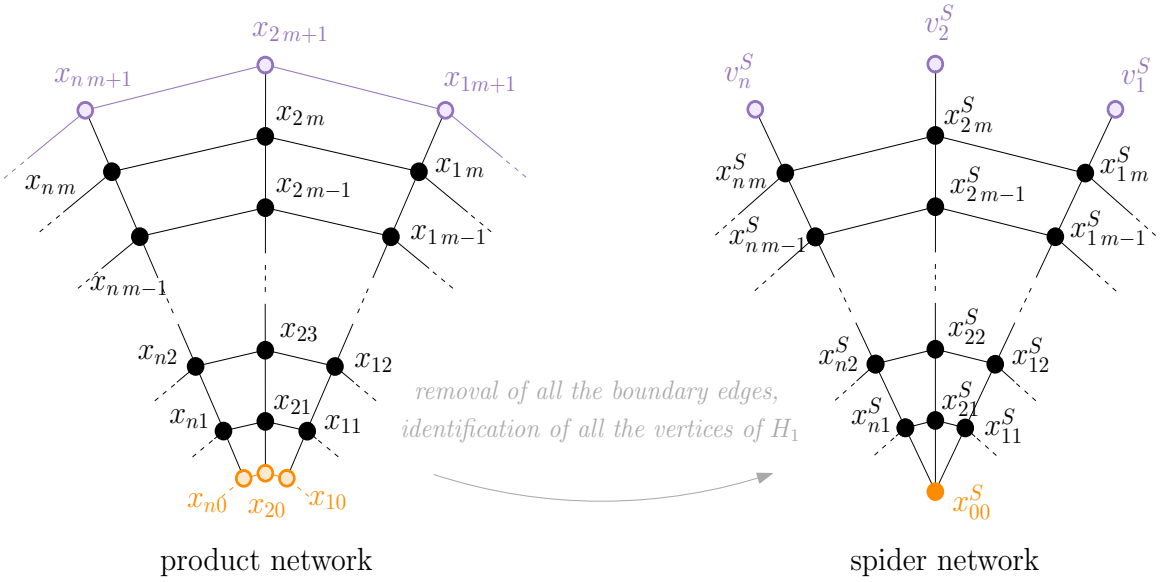


Figura 45: Equivalente de una Spider en Cilindro (círculos x caminos). Fuente: **[14]**

El siguiente resultado muestra la expresión de la función de Green de una Spider con conductancias arbitrárias en términos de la función de Green de un cilindro.

Proposición 1.2 La función de Green de la red de una Spider, G^s_{qs}, es dada por las siguientes fórmulas

- Si $x^s_{lk} \neq x^s_{00}$, entonces

$$G^s_{qs}(x^s_{00},x^s_{lk}) = \frac{\sum_{r=1}^{n} c(x_{r0},x_{r1})G_{qc}(x_{r1},x_{kl})}{\lambda + \sum_{t=1}^{n} c(x_{t0},x_{t1})(1-\sum_{r=1}^{n} c(x_{r0},x_{r1})G_{qc}(x_{r1},x_{t1}))} \quad (120)$$

$$G^s_{qs}(x^s_{ij},x^s_{lk}) = G_{qc}(x_{ij},x_{lk})G^s_{qs}(x^s_{00},x^s_{lk})\sum_{t=1}^{n} c(x_{t0},x_{t1})G_{qc}(x_{ij},x_{t1}) \quad \text{por todo } x^s_{ij} \in F_s \setminus \{x^s_{00}\} \quad (121)$$

$$G^s_{qs}(v^s_j,x^s_{lk}) = 0 \ \text{ por todo } v^s_j \in \delta(F_s) \quad (122)$$

- Si $x^s_{lk} = x^s_{00}$ entonces:

$$G^s_{qs}(x^s_{00}, x^s_{00}) = \frac{1}{\lambda + \sum\limits_{t=1}^{n} c(x_{t0}, x_{t1})(1 - \sum_{r=1}^{n} c(x_{r0}, x_{r1})G_{qc}(x_{t1}, x_{r1}))} \tag{123}$$

$$G^s_{qs}(x^s_{ij}, x^s_{00}) = G^s_{qs}(x^s_{00}, x^s_{00}) \sum_{t=1}^{n} c(x_{t0}, x_{t1})G_{qc}(x_{ij}, x_{t1}) \text{ por todo } x^s_{ij} \in F_s \setminus \{x^s_{00}\} \tag{124}$$

$$G^s_{qs}(v^s_j, x^s_{00}) = 0 \text{ por todo } v^s_j \in \delta(F_s) \tag{125}$$

Una vez presentadas las formulaciones para la obtención de la función de Green Spider se procede a presentar la Aplicación Dirichlet-to-Neumann de una Spider que ha sido obtenida en el desarrollo de este proyecto:

- **Proposición 1.3** aplicación Dirichlet-to-Neumann de una Spider con conductancias $c1$ y $c2$ está dada por

$$\Lambda(v_i, v_j) = -c_2^2 G^s_{qs}(x_{im}, x_{jm}) + c_2\epsilon_{v_j}(v_i) \tag{126}$$

  Para verificar la obtención de la función de Green de la Spider, se programaran estas ecuaciones y se aplicará a la Spider. Una vez obtenidas las funciones de Green, se aplicará la Aplicación Dirichlet y se buscará la matriz de respuesta y se comparará con la matriz de respuesta empleando el complemento de Schur.

  La limitación de este procedimiento es que las conductancias del camino y del círculo son constantes y no pueden variar entre ellas. Además no se contempla los pesos de los nodos.

11 Código de la aplicación Dirichlet-to-Neumann en Matlab

El primer detalle a tener en cuenta es que la Spider vista desde el punto de vista de un cilindro todos los nodos del círculo 0 corresponden al nodo x_{00}. Los círculos como se pude observar en la Figura 46 se enumeran desde el interior hasta el exterior y en el algoritmo se empleará la misma metodología.

11.1 Algoritmo aplicado a una Spider de 7 nodos

Se aplica el algoritmo teórico explicado en la sección anterior en una Spider de 7 nodos.

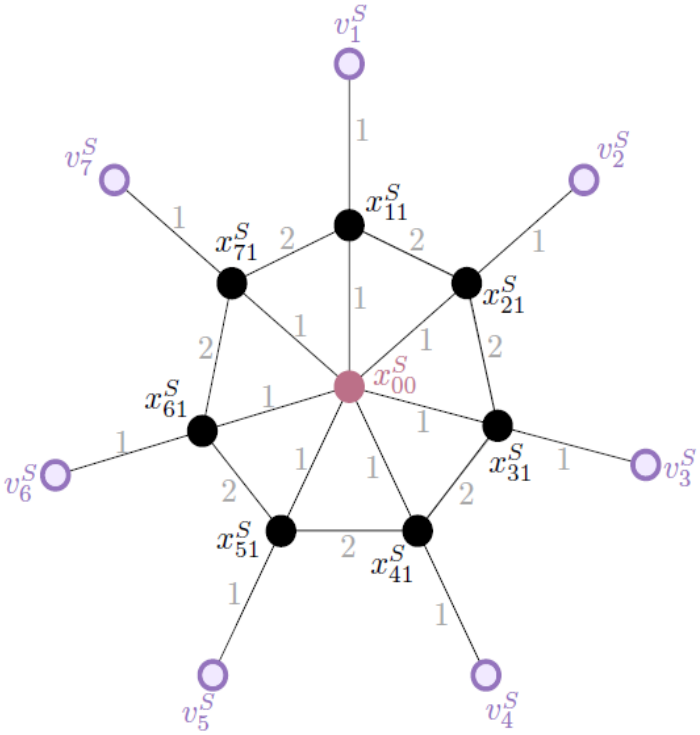


Figura 46: Características de la Spider de 7 nodos a la cual se le busca la función de Green. Fuente: **[14]**

El algoritmo es el siguiente:

```
% DIMENSIONES DE LA SPIDER (se supone que cada nodo tiene un único radio)

n = input('INTRODUCE EL NÚMERO DE LOS RADIOS n:');

if mod(n-3,4)~ =0

  disp('ERROR AL INTRODUCIR DIMENSIONES DE SPIDER')% Cuando el usuario introduce un valor que no es múltiplo de 4 indicará que no es correcto.
```

```
end

circulo = (n-3)/4;% El valor de círculos que tendrá la Spider

disp(['EL NÚMERO DE CÍRCULOS ES:',num2str(circulo)])

Lambda= 0; % Es el caso del Laplaciano combinatorio

c1=2; % Conductancia del círculo

c2=1; % Conductancia del camino

m= circulo; % los círculos totales que tiene el cilindro son los círculos de la Spider más el círculo que corresponde al nodo x00 y el círculo donde se encuentran los nodos del controno.

for i= m:-1:1 % Se recorren los círculos desde el exterior hasta el interior sin tener en cuenta el círculo de x00 y el del contorno

    for j=1:n % Se recorren todos los nodos del círculo

        for k= m:-1:1 % Se recorren los círculos desde el exterior hasta el interior sin tener en cuenta el círculo de x00 y el del contorno

            for l=1:n % Se recorren todos los nodos del círculo

                Sum1=0; % Se inicia el valor de la variable que irá acumulando los cálculos de Lambda y que depende R

                for R= 1:circulo % Se calcula tantos autovalores como círculos tiene la Spider (no el cilindro)

                    Lambda_r(R,1) = 2* c2* (1-cos(pi* R/(m+1)));% El valor de las conductancias es constante en el camino

                Sum1 = Sum1 + (sin(i* pi* R/(m+1))* sin(k* pi* R/(m+1))* (chebyshevU(n-1-abs(j-l),1+ (Lambda + Lambda_r(R,1))/(2* c1))+chebyshevU(abs(j-l)-1,1+ (Lambda + Lambda_r(R,1))/(2* c1)))/(2* c1* chebyshevT(n,1+ (Lambda + Lambda_r(R,1))/(2* c1))-1); % Se aplica la Proposición 1.1. Los chebyshev se han programado a parte con una función ya que el Matlab no permite este cálculo.

                end


                    G_q((m+1-i)* n+j,(m+1-k)* n+l)= 2/(m+1)* Sum1; % Se calcula la función de Green del cilindro y el valor de Sum1 se calcula anteriormente


            end


        end

    end

end
```

13/58	5/58	1/29	1/58	1/58	1/29	5/58
5/58	13/58	5/58	1/29	1/58	1/58	1/29
1/29	5/58	13/58	5/58	1/29	1/58	1/58
1/58	1/29	5/58	13/58	5/58	1/29	1/58
1/58	1/58	1/29	5/58	13/58	5/58	1/29
1/29	1/58	1/58	1/29	5/58	13/58	5/58
5/58	1/29	1/58	1/58	1/29	5/58	13/58

Figura 47: Resultado Algoritmo función de Green aplicado a Spider de 7 nodos. Elaboración propia

Como se puede observar, los resultados suman el mismo valor en todas las filas ya que es la misma fila pero desplazada una posición. Estos valores corresponden a los nodos de los círculos internos que tiene la Spider, es decir, los nodos de los círculos sin tener en cuenta el vértice x_{00} y los valores del contorno ya que son 0 por la condición Dirichlet.

El algoritmo programado para calcular la función de Green de la Spider es el siguiente:

```
% FUNCIÓN DE GREEN PARA LA SPIDER EMPLEANDO LA PROPOSICIÓN 1.2

% Cálculo del valor de función de Green en el nodo x00 ya que se empleará

% para el cálculos de la función de Green en otros nodos.

G_qt=0; % Se parte con la variable igual a 0 que acumulará los valores a medida que se va avanzando de nodo.

for t= 1:n % Se recorre todos los nodos de cada círculo

  G_qr_t= 0; % Se parte con la variable igual a 0 que acumulará los valores a medida que se va avanzando de nodo.

  for r= 1:n % Se recorre todos los nodos de cada círculo

     G_qr_t= G_qr_t + c2* G_q(circulo* n+t, circulo* n+r); % Se aplica la fórmula de la proposición 1.2

  end

  G_qt= G_qt + c2* (1-G_qr_t); % Se aplica la fórmula de la proposición 1.2

end

G_qs(n* circulo+n+1, n* circulo+n+1) = 1/(Lambda + G_qt); % Se aplica la fórmula de la proposición 1.2


% G_qs(x00,xlk)


for k= m:-1:1 % Se recorren los círculos desde el exterior hasta el interior sin tener en cuenta el círculo de x00 y el del contorno
```

```
   for l=1:n % Se recorre todos los nodos de cada círculo

      G_qr= 0;

    for r= 1:n % Se recorre todos los nodos de cada círculo

       G_qr= G_qr + c2* G_q(n* circulo+r, (m+1-k)* n+l); % Se aplica la fórmula de la proposición 1.2

    end

    G_qs(n* circulo+n+1, (m+1-k)* n+l) = G_qs(n* circulo+n+1, n* circulo+n+1)* G_qr; % Se aplica la
fórmula de la proposición 1.2

  end

end

for i= m:-1:1 % Se recorren los círculos desde el exterior hasta el interior sin tener en cuenta el círculo de
x00 y el del contorno

  for j=1:n % Se recorre todos los nodos de cada círculo

    for k= m:-1:1

      for l=1:n % Se recorre todos los nodos de cada círculo

          G_q_t=0; % Se parte con la variable igual a 0 que acumulará los valores a medida que se va
avanzando de nodo.

        for t=1:n % Se recorre todos los nodos de cada círculo

        G_q_t= G_q_t + c2* G_q((m+1-i)* n+j, n* circulo+t); % Se impone que en los nodos del contorno
la función de Green es 0.



        end



          G_qs((m+1-i)* n+j,(m+1-k)* n+l)= G_q((m+1-i)* n+j,(m+1-k)* n+l) + G_qs(n* circulo+n+1,
(m+1-k)* n+l)* G_q_t; % Se impone que en los nodos del contorno la función de Green es 0.

      end

    end

  end

end



for j=1:n % Se recorre todos los nodos de cada círculo
```

```
  for k= m:-1:1 % Se recorren los círculos desde el interior hasta el exterior sin tener en cuenta el círculo de x00 y el del contorno

    for l=1:n % Se recorre todos los nodos de cada círculo

      G_qs(j, (m+1-k)* n+l)=0; % Se impone que en los nodos del contorno la función de Green es 0.

    end

  end

end


for i= m:-1:1 % Se recorren los círculos desde el interior hasta el exterior sin tener en cuenta el círculo de x00 y el del contorno

  for j=1:n % Se recorre todos los nodos de cada círculo

        G_q_t=0; % Se parte con la variable igual a 0 que acumulará los valores a medida que se va avanzando de nodo.

      for t=1:n % Se recorre todos los nodos de cada círculo

        G_q_t= G_q_t + c2* G_q((m+1-i)* n+j, n* circulo+t); % Se aplica la fórmula de la proposición 1.2

      end

   G_qs((m+1-i)* n+j, n* circulo+n+1)= G_qs(n* circulo+n+1, n* circulo+n+1)* G_q_t; % Se aplica la fórmula de la proposición 1.2

  end

end


for j=1:n % Se recorre todos los nodos de cada círculo

  G_qs(j, n* circulo+n+1)=0; % Se impone que en los nodos del contorno la función de Green es 0.

end

 disp(G_qs); % Se imprime la matriz de la función de Green.

 % Se calcula la inversa del Laplaciano de los nodos interiores y sin considerar los pesos para poder verificar la coincidencia de la de Green de la Spider sin considerar los pesos.

 A= L(n+1:end, n+1:end); % Se seleccionan los nodos interiores que son a partir del número de contornos tanto en filas como en columnas

 B= inv(A); % Se realiza la inversa de la Submatriz del laplaciano A

 D_F = L(1:n,1:n);
```

```
% COMPLEMENTO DE SCHUR

 % Se calcula el complemento de Schur para comparar la matriz de respuesta

 % con la de la aplicación Dirichlet

Lambda_dia = zeros(n* circulo+1+n,n* circulo+1+n); % Se crea una matriz diagonal, donde en los primeros n nodos( que son los del contorno) habrá el peso de estos

for i= 1:  n % Se crea una matriz diagonal para contemplar el efecto de la Lambda en los nodos del contorno. Este valor de Lambda es el mismo para todos los nodos.

 Lambda_dia(i,i)= Lambda;

end

L_q= L; % Se calcula el Laplaciano teniendo el valor de la Lambda, no se considera los pesos ya que al calcular la aplicación Dirichlet los pesos no se han tenido en cuenta.

D_F = L_q(1:n,1:n);

C0 = L_q(1:n,n+1:n* circulo+n+1);

C0_t = L_q(n+1:n* circulo+n+1, 1:n);

L_F = L_q(n+1:n* circulo+n+1,n+1:n* circulo+n+1);

Nq= D_F - C0* inv(L_F) * C0_t; % Se calcula la matriz de respuesta mediante el Complemento de Schur

disp(Nq) % Se imprime la matriz de respuesta a partir de la Aplicación Dirichlet de una Spider
```

Para comprobar este resultado, se escoge la submatriz del Laplaciano de los nodos interiores de la Spider y se invierte esta submatriz, ya que $[\mathscr{L}]\,[G_q] = [I]$. Efectivamente, la submatriz invertida del Laplaciano y la función de Green da lo mismo.

```
 6    -2     0     0     0     0    -2    -1
-2     6    -2     0     0     0     0    -1
 0    -2     6    -2     0     0     0    -1
 0     0    -2     6    -2     0     0    -1
 0     0     0    -2     6    -2     0    -1
 0     0     0     0    -2     6    -2    -1
-2     0     0     0     0    -2     6    -1
-1    -1    -1    -1    -1    -1    -1     7
```

Figura 48: Submatriz del Laplaciano con los nodos interiores y la inversa de esta submatriz. Elaboración propia

60/203	32/203	43/406	18/203	18/203	43/406	32/203	1/7
32/203	60/203	32/203	43/406	18/203	18/203	43/406	1/7
43/406	32/203	60/203	32/203	43/406	18/203	18/203	1/7
18/203	43/406	32/203	60/203	32/203	43/406	18/203	1/7
18/203	18/203	43/406	32/203	60/203	32/203	43/406	1/7
43/406	18/203	18/203	43/406	32/203	60/203	32/203	1/7
32/203	43/406	18/203	18/203	43/406	32/203	60/203	1/7
1/7	1/7	1/7	1/7	1/7	1/7	1/7	2/7

Figura 49: Inversa de la submatriz del Laplaciano que coincide con la función de Green de la Spider Elaboración propia

Finalmente, se calcula la matriz de respuesta a partir de la aplicación Dirichlet-to-Neumann que coincide con la matriz de respuesta calculada a partir del complemento de Schur:

```
% APLICACIÓN DIRICHLET DE UNA SPIDER:

Eps= eye(n); % Se crea la matriz identidad que representaría la función Epsilon que vale 1 cuando los subíndices x=y y 0 cuando los subíndices son diferentes.

for i=1:n % Se recorren todos los nodos del primer círculo que son los que tienen la información del contorno.

  for j=1:n % Se recorren todos los nodos del primer círculo que son los que tienen la información del contorno.

    Nq_Dir(i,j)= -c2^ 2 * G_qs(n+i, n+j) + c2*Eps(i,j); % Se aplica la proposición 1.3

  end

end

disp(Nq_Dir)% Se imprime la matriz de respuesta a partir de la Aplicación Dirichlet de una Spider
```

Para calcular los Chebyshevs de primer y de segundo tipo, se han tenido que programar a parte ya que Matlab no los calcula:

```
function [C_T]= chebyshevT(g,x)
  T =zeros(g,1);
  if g==0
    C_T = 1; % Polinomio de grado 0
  elseif g==1
    C_T = x; % Polinomio de grado 1
  elseif g==2
    C_T = 2* x^ 2-1; % Polinomio de grado 2
  elseif g>2 % En el caso que sea inferior a dos que coja los siguientes polinomios en la posición indicada en la matriz para respetar posición de matriz según grado
    T(1,1)= 1;
    T(2,1)= x;
    for n=1: (g-1) % Grado en que se encuentra
      T(n+2,1)=2* x* T(n+1,1)-T(n,1); % Se aplica la fórmula para el ChebyshevT (que es el de primer
```

```
tipo)
    end
    C_T= 2* x* T(n+1,1)-T(n,1); % Se guarda en una variable para evitar errores de matrices
  else
    C_T = 0; % cuando -1, no se considera otros valores negativos, ya que al haber un valor absoluto, el
caso más crítico es de -1
  end
end
function [C_U]= chebyshevU(g,x)
  U =zeros(g,1);
  if g==0
    C_U = 1; % Polinomio de grado 0
  elseif g==1
    C_U = 2* x; % Polinomio de grado 1
  elseif g==2
    C_U = 4* x^ 2-1; % Polinomio de grado 2
  elseif g>2 % En el caso que sea inferior a dos que coja los siguientes polinomios en la posición indicada
en la matriz para respetar posición de matriz según grado

    U(1,1)= 1;
    U(2,1)= 2* x;
    for n=1: (g-1) % Grado en que se encuentra
      U(n+2,1)=2* x* U(n+1,1)-U(n,1); % Se aplica la fórmula para el ChebyshevT (que es el de segundo
tipo)
    end
    C_U= 2* x* U(n+1,1)-U(n,1); % Se guarda en una variable para evitar errores de matrices
  else
    C_U = 0; % cuando -1, no se considera otros valores negativos, ya que al haber un valor absoluto, el
caso más crítico es de -1
  end
end
```

11.2 Algoritmo aplicado a una Spider de 11 nodos

Se aplica el algoritmo programado a una Spider de 11 nodos con las conductancias radiales con un valor de 1 y las conductancias adyacentes 2 como el primer ejemplo. Las matrices de respuesta obtenidas empleando el complemento de Schur y la aplicación Dirichlet son las siguientes:

0.7158	-0.1412	-0.0809	-0.0545	-0.0429	-0.0385	-0.0385	-0.0429	-0.0545	-0.0809	-0.1412
-0.1412	0.7158	-0.1412	-0.0809	-0.0545	-0.0429	-0.0385	-0.0385	-0.0429	-0.0545	-0.0809
-0.0809	-0.1412	0.7158	-0.1412	-0.0809	-0.0545	-0.0429	-0.0385	-0.0385	-0.0429	-0.0545
-0.0545	-0.0809	-0.1412	0.7158	-0.1412	-0.0809	-0.0545	-0.0429	-0.0385	-0.0385	-0.0429
-0.0429	-0.0545	-0.0809	-0.1412	0.7158	-0.1412	-0.0809	-0.0545	-0.0429	-0.0385	-0.0385
-0.0385	-0.0429	-0.0545	-0.0809	-0.1412	0.7158	-0.1412	-0.0809	-0.0545	-0.0429	-0.0385
-0.0385	-0.0385	-0.0429	-0.0545	-0.0809	-0.1412	0.7158	-0.1412	-0.0809	-0.0545	-0.0429
-0.0429	-0.0385	-0.0385	-0.0429	-0.0545	-0.0809	-0.1412	0.7158	-0.1412	-0.0809	-0.0545
-0.0545	-0.0429	-0.0385	-0.0385	-0.0429	-0.0545	-0.0809	-0.1412	0.7158	-0.1412	-0.0809
-0.0809	-0.0545	-0.0429	-0.0385	-0.0385	-0.0429	-0.0545	-0.0809	-0.1412	0.7158	-0.1412
-0.1412	-0.0809	-0.0545	-0.0429	-0.0385	-0.0385	-0.0429	-0.0545	-0.0809	-0.1412	0.7158

Figura 50: Matriz de respuesta de 11 rayos empleando el complemento de Schur. Elaboración propia

0.7161	-0.1410	-0.0808	-0.0544	-0.0428	-0.0384	-0.0384	-0.0428	-0.0544	-0.0808	-0.1410
-0.1410	0.7161	-0.1410	-0.0808	-0.0544	-0.0428	-0.0384	-0.0384	-0.0428	-0.0544	-0.0808
-0.0808	-0.1410	0.7161	-0.1410	-0.0808	-0.0544	-0.0428	-0.0384	-0.0384	-0.0428	-0.0544
-0.0544	-0.0808	-0.1410	0.7161	-0.1410	-0.0808	-0.0544	-0.0428	-0.0384	-0.0384	-0.0428
-0.0428	-0.0544	-0.0808	-0.1410	0.7161	-0.1410	-0.0808	-0.0544	-0.0428	-0.0384	-0.0384
-0.0384	-0.0428	-0.0544	-0.0808	-0.1410	0.7161	-0.1410	-0.0808	-0.0544	-0.0428	-0.0384
-0.0384	-0.0384	-0.0428	-0.0544	-0.0808	-0.1410	0.7161	-0.1410	-0.0808	-0.0544	-0.0428
-0.0428	-0.0384	-0.0384	-0.0428	-0.0544	-0.0808	-0.1410	0.7161	-0.1410	-0.0808	-0.0544
-0.0544	-0.0428	-0.0384	-0.0384	-0.0428	-0.0544	-0.0808	-0.1410	0.7161	-0.1410	-0.0808
-0.0808	-0.0544	-0.0428	-0.0384	-0.0384	-0.0428	-0.0544	-0.0808	-0.1410	0.7161	-0.1410
-0.1410	-0.0808	-0.0544	-0.0428	-0.0384	-0.0384	-0.0428	-0.0544	-0.0808	-0.1410	0.7161

Figura 51: Matriz de respuesta de 11 rayos empleando la Aplicación Dirichlet. Elaboración propia.

Como se puede observar no son exactamente idénticas, sino que difieren de algunos decimales.

11.3 Algoritmo aplicado a una Spider de 19 nodos

Se aplica el algoritmo programado a una Spider de 19 nodos con las conductancias radiales con un valor de 1 y las conductancias adyacentes 2 como los ejemplos anteriores. Las matrices de respuesta obtenidas empleando el complemento de Schur y la aplicación Dirichlet son las siguientes:

0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345
-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724
-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436
-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292
-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216
-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174
-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151
-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138
-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132
-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132
-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138
-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151
-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174
-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216
-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292
-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436
-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724
-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345
-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0151	-0.0138	-0.0132	-0.0132	-0.0138	-0.0151	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218

Figura 52: Matriz de respuesta de 19 rayos empleando el complemento de Schur. Elaboración propia

0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345
-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724
-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436
-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292
-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216
-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174
-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150
-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137
-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132
-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132
-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137
-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150
-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174
-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292	-0.0216
-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436	-0.0292
-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724	-0.0436
-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345	-0.0724
-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218	-0.1345
-0.1345	-0.0724	-0.0436	-0.0292	-0.0216	-0.0174	-0.0150	-0.0137	-0.0132	-0.0132	-0.0137	-0.0150	-0.0174	-0.0216	-0.0292	-0.0436	-0.0724	-0.1345	0.7218

Figura 53: Matriz de respuesta de 19 rayos empleando la aplicación Dirichlet. Elaboración propia

En este caso, las matrices son casi idénticas, exceptuando el valor de la octava fila y primera columna que difiere de 0.0001.

12 Análisis impacto ambiental

Dado que se trata de un trabajo estrictamente teórico, de su realización no se deriva ningún impacto ambiental reseñable.

13 Metodología y dificultades

Según los objetivos marcados anteriormente, se ha seguido el siguiente procedimiento para presentar el TFG:
La primera semana (18/02/2019-25/02/2019) se ha dedicado a la introducción del tema.

Las tres semanas siguientes (25/02/2019-18/03/2019) me he dedicado a formarme en relación al problema directo/inverso. Las dificultades en este paso son debidas a la escasa formación que tenía tanto de la formulación matemática de los problemas como de su interpretación física. Aunque, gracias a la ayuda de mis tutores he podido llevar a cabo el trabajo.

Las posteriores 5 semanas (18/02/2019-22/04/2019) me he concentrado en la programación del algoritmo de recuperación de conductancias en una Spider y su verificación. El hecho de haber programado poco con Matlab ha hecho que tenga que formarme por Internet. Además, he tenido dificultades ya que tenía problemas con partes del código que inicialmente había programado mal porque no había entendido bien el algoritmo matemático del artículo con lo que era constantemente pararse a entender bien cada paso e implementarlo.
Asimismo, he tenido bastantes dificultades con el código, ya que el hecho de girar una matriz a la vez que va girando el nodo del contorno, aunque a simple vista nuestra mente lo hace rápidamente la implementación es complicada y me ha llevado largas noches dar con la solución.
Otra dificultad se presenta cuando se va avanzando de círculo y se buscan las conductancias adyacentes y las radiales, ya que se debe calcular entre el nodo estudiado y el anterior a éste (se puede ver en la explicación del código). Esto me ha llevado tiempo ya que cuando el nodo anterior es anterior al primer nodo del círculo hay que hacer que vuelva al mismo círculo.
También se ha tenido en cuenta que para las conductancias adyacentes, radiales y los valores uj se debe diferenciar si estamos en el primer círculo, en círculos interiores o en el círculo más interno ya que sus condiciones varían y se deben contemplar. Todos estos detalles me han dificultado la implementación del algoritmo teórico ya que éstos aparecen haciendo ejemplos y verificando caso por caso y paso por paso.

La dificultad más importante se ha presentado en el paso 6 del algoritmo. En las primeras implementaciones no recuperaba las conductancias finales y estuve con los tutores buscando el error en el programa que aunque parecía estar bien las conductancias más internas no se recuperaban correctamente. Entonces resolví el sistema de 34 ecuaciones a mano de una Spider de 11 nodos (ya que Matlab no lo resolvía al haber incógnitas en el vector solución) y encontré que los valores de uj en el círculo más interno no daban iguales a los del programa y significaba que en el sexto paso había algo que hacía que no diera lo mismo y localicé un error de signos en el artículo.

Una vez finalizado el algoritmo de la Spider, he implementado los algoritmos de obtención de Función de Green del cilindro y de la Spider para poderlos utilizar en la Aplicación Dirichlet. El tiempo invertido ha sido de unas tres semanas. Los algoritmos teóricos forman parte del artículo que aún no está finalizado y se han tenido que retocar para poderlos implementar correctamente en el Matlab. Además, mis tutores dedujeron la proposición de la Aplicación Dirichlet de una Spider para poderla presentar en este TFG.

Finalmente, me he dedicado a escribir y redactar el trabajo y perfeccionar detalles para la pre-

sentación de este documento.

14 Conclusiones

Para finalizar este trabajo hay cuestionarse si se ha cumplido con los objetivos del proyecto, si se han adquirido nuevos conocimientos y qué puntos deja esta investigación abiertos a trabajos o proyectos futuros.

En primer lugar, creo que he cumplido con los objetivos propuestos aunque la implementación de los algoritmos ha sido más costosa de lo que inicialmente pensaba.

En segundo lugar, han sido numerosas los nuevos conocimientos adquiridos: desde aprender a programar con Matlab, profundizar en fundamentos físicos y matemáticos, hasta emplear todo tipo de herramientas y soluciones rápidas para poder resolver las dificultades surgidas. Es importante destacar que todo esto ha sido gracias al apoyo de los tutores que me han guiado en todo momento.

He conseguido programar el algoritmo de recuperación de las conductancias en una red Spider que era el objetivo fundamental del trabajo. Para testar el algoritmo, se ha partido de redes Spider con conductancias conocidas, se han calculado las correspondientes matrices de respuestas y entonces se ha aplicado el algoritmo para recuperar las conductancias. Hemos mostrado que en problemas de pequeño tamaño, el algoritmo recuperaba exactamente las conductancias iniciales. Sin embargo, también he observado que incluso en redes de tamaño moderado, el algoritmo no alcanza a recuperar exactamente las conductacias de las que se partía. Este hecho es debido, esencialmente, a que el proceso de determinación de la matriz de respuestas requiere el cálculo de la inversa de una matriz mal condicionada. No obstante, ha de tenerse en cuenta que en problemas de aplicación real, el dato de partida es precisamente la matriz de respuestas por lo que no se requerirá el cálculo de ninguna inversa. En todo caso, sí se ha de tener en cuenta que el problema inverso, incluso en el caso continuo, está mal condicionado y que por tanto, matrices de respuesta muy similares pueden conducir a valores muy diferentes en las conductancias.

La segunda parte de este trabajo consistía en programar la función de Green de un cilindro, de una Spider y presentar la expresión explícita de la Aplicación Dirichlet teóricamente y programada para una Spider. Estos propósitos se han realizado correctamente, pero contienen una limitación ya que se han obtenido a partir de un productos de círculos por caminos y por lo tanto, las conductancias de los caminos son iguales entre ellas y ocurre lo mismo con las de los círculos. Con la Aplicación Dirichlet se ha podido observar que se obtiene la matriz de respuesta que es prácticamente la misma que la que se obtiene con el complemento de Schur.

Es importante destacar que aunque los problemas directos son problemas lineales, que en el continuo se plantean mediante problemas de contorno para operadores de Schrödinger, los problemas inversos son problemas altamente no lineales. El operador de Schrödinger contiene la información interna de la red a través de las conductancias, mientras que la función de Green nos proporciona la respuesta de la red ante una fuerza aplicada en el interior con condiciones nulas en el contorno. Por otro lado, la aplicación Dirchlet-to-Neumann representa la respuesta del sistema ante una actuación en la frontera del dominio, y por tanto se pueden medir sin el conocimiento previo del interior.

14.1 Extensión TFG

Este Trabajo Fin de Grado se puede ampliar de diferentes maneras: se puede aplicar a situaciones reales y estudiar su utilidad, se puede programar la recuperación de conductancias en diferentes geometrías como buscar la manera de obtener la matriz Nq sin perder exactitud. También se podría intentar desarrollar la Aplicación Dirichlet para una Spider de diferentes conductancias, ya que sería algo que se podría generalizar y sería una herramienta muy potente.

15 Análisis económico

15.1 Costes de Ingeniería

Concepto		Horas	Precio/Hora	Precio parcial
Formación	Formación previa de matemáticas y física	50	20€/h	1.000€
	Problemas directos	60	20€/h	1.200€
	Problemas inversos	120	20€/h	2.400€
Aplicación	Algoritmo de la recuperación de conductancias de una Spider	200	30€/h	6.000€
	Algoritmo de la Función Green del cilindro y de la Spider	80	30€/h	2.400€
Tutoria	Horas invertidas por los tutores para la formación y apoyo en la realización de algoritmos	180	40€/h	7200€
Redacción de la memoria		120	25€/h	3.000€
			Subtotal 1	23.200euros

15.2 Licencias

Software	Precio
Matlab	700€
Subtotal 2	700€

15.3 Material

Material informático	Precio
Portátil	450€
Subtotal 3	450€

16 Bibliografia

[1] Arauz, C., Carmona, A., Encinas, A. Overdetermined partial boundary value problems on finite networks. "Journal of mathematical analysis and applications", 01 Març 2015, vol. 423, núm. 1.
[2] C. Arauz, A. Carmona, A. M. Encinas, M. Mitjana, Green functions on product networks, Discrete Appl. Math., https://doi.org/10.1016/j.dam.2018.10.004
[3] A.P. Calderón, On an inverse boundary value problem., in Seminar on Numerical Analysis and its Applications to Continuum Physics (Rio de Janeiro, 1980), pp. 65-73, Soc. Brasil. Mat., Rio de Janeiro, 1980. Reprinted in Comput. Appl. Math. 25 (2006), 133-138.
[4] C. Araúz, A. Carmona, A.M. Encinas, Discrete Serrin's problem, Linear Algebra Appl. (2014), http://dx.doi.org/10.1016/j.laa.2014.01.038.
[5]Arauz Lombardía, C. "The inverse problem on finite networks". Tesi doctoral, UPC, Departament de Matemàtica Aplicada III, 2014. Disponible en: <http://hdl.handle.net/2117/95431>
[6] L. Borcea, F. Guevara Vasquez, A.V. Mamonov, A discrete Liouville identity for numerical reconstruction of Schrödinger potentials, (2016) arXiv:1601.07603.
[7] L. Borcea, V. Druskin, F. Guevara Vasquez, A.V. Mamonov, Resistor network approaches to electrical impedance tomography, in Inverse problems and Applications: Inside Out II, 55-118. Math. Sci. Res. Inst. Publ., 60, Cambridge Univ. Press, 2013.
[8] L. Borcea, A.V. Mamonov, V. Druskin, Circular resistor networks for electrical impedance tomography with partial boundary measurements, Inverse Problems 26 (2010), 045010-045039.
[9] L. Borcea, V. Druskin, F. Guevara Vasquez, Electrical impedance tomography with resistor networks, Inverse Problems 24 (2008), 035013-035044.
[10] Boyer, J., Garzella, J.J., Guevara Vasquez, F., On the solvability of the discrete conductivity and Schrödinger inverse problems, SIAM J. Appl. Math. 76 (2016) 1053-1075.
[11] E.B. Curtis, E. Mooers, J.A. Morrow, Finding the conductors in circular networks from boundary measurements, RAIRO Modél. Math. Anal. Numér. 28 (1994), 781-814.
[12] E. B. Curtis, D. Ingerman, J. A. Morrow: Circular planar graphs and resistor networks, Linear Algebra Appl. 283 (1998), 115-150.
[13] E. Curtis, J. Morrow: Inverse Problems for Electrical Networks. Series on Applied Mathematics, World Scientific Publishing 13, 2000.
[14] A. Carmona, A.M. Encinas, M. Mitjana, Boundary value problems on layered product networks, Unpublished manuscript. [15] Encinas, A.M. Rodellar, J., Curso de Ecuaciones Diferenciales en Derivadas Parciales, UPC 2005

17 Anexo

17.1 Algoritmo completo de recuperación de conductancias de una Spider

```
clear all
% DIMENSIONES DE LA SPIDER (se supone que cada nodo tiene un único radio)
n = input('INTRODUCE EL NÚMERO DE LOS RADIOS n:');
circulo = (n-3)/4;% El valor de círculos que tendrá la Spider
disp(['EL NÚMERO DE CÍRCULOS ES:',num2str(circulo)])
Lambda = input('Introduce el valor de Lambda');% El valor de la Lambda

c = zeros((n* circulo+1)+n,(n* circulo+1)+n);% La matriz que contendrá las conductancias de las ramas de la Spider
L= xlsread('TFG_ALGORITMO.xlsx'); % Se importa la matriz de conductancia que será el enunciado, esta matriz será importada del Excel

Lambda_diagonal = zeros((n* circulo+1)+n,(n* circulo+1)+n); % Se cre una matriz diagonal, donde en los primeros n nodos( que son los del contorno) habrá el peso de estos

for i= 1:  n % Se crea una matriz diagonal para contemplar el efecto de la Lambda en los nodos del contorno. Este valor de Lambda es el mismo para todos los nodos.
 Lambda_diagonal(i,i)= Lambda;
end

Peso= [ ]; % Se define la matriz de pesos que se emplará para resolver el ejemplo de la Spider.
w = Peso'; % El vector columna de Peso lo transforma en fila para el cálculo posterior.  Se realiza este cambio para que sea más ágil la escritura del algoritmo.
Diag_peso = diag(Peso); % Se tranforma el vector de peso en una matriz diagonal para los posteriores cálculos para coincidan las dimensiones.
Q_W=-diag(L* Peso)* inv(Diag_peso);
L_q= L+ Q_W + Lambda_diagonal; % Se calcula el Laplaciano teniendo en cuenta los pesos como el valor de la Lambda.

% Complemento de Schur
D_F = L_q(1:n,1:n);
C0 = L_q(1:n,n+1:n* circulo+n+1);
C0_t = L_q(n+1:n* circulo+n+1, 1:n);
L_F = L_q(n+1:n* circulo+n+1,n+1:n* circulo+n+1);
Matriz_Nq= inv(L_F)* L_F;
disp(Matriz_Nq);
I= eye(n* circulo+n+1-(n));
Identidad= Matriz_Nq- I;
disp(Identidad);
Rcondicion= rcond(L_F); % devuelve una estimación de la condición recíproca de L_F en 1-norm. Si bien está condicionada L_F , rcond(L_F) es cerca de 1.0. Si mal está condicionada L_F , rcond(L_F) es cerca de 0.
disp(Rcondicion);
Ccondicion= cond(L_F); % devuelve el 2-norma para la inversión de los números, igual a la relación entre el mayor valor singular de L_F para los más pequeños. Si el número de condición de L_F es mucho mayor que 1, la matriz es sensible a los cálculos inversos.
disp(Ccondicion);
Nq= D_F - C0* inv(L_F)* C0_t; % Se calcula la matriz de respuesta mediante el Complemento de Schur

[eig_vector, eig_value]=eig(Nq);
LambdaL= eig_value(1,1);
```

```
ww = eig_vector(:,1);
for i= 1: n
  if ww(i,1)<0
     ww(i,1) = - ww(i,1);
  end
end
disp(LambdaL)
disp(ww)
A= 0;
 for i= 1:n % Se comprueba que la matriz de respuesta es simétrica
    for j = 1:n
       if Nq(i,j)==Nq(j,i)
          A=A+0;
       else
          A= A+1;
       end
    end
 end
 if A==0
    disp('La matriz de respuesta es simétrica')
 else
    disp('La matriz de respuesta no es simétrica')
 end

% VALORES DE uj EN LOS CONTORNO R Y A DE LA SPIDER:
vj = zeros(1+ (n-1)/2,1); % Se crea una matriz que contendrá los valores en los nodos del contorno R y A que son las condiciones que se imponen para resolver estos tipos de problemas.
vj(1,1) = 1; % El primer valor correponde al valor del contorno R que es 1.
for q= 2:(n-1)/2 % La primera mitad de los nodos del contorno seguidos del nodo donde se encuentra el nodo que pertence a R, se impone que la función valorada en estos es 0.
  vj(q,1) = 0;
end

% SUBMATRICES DE Nq: R(contorno que rompe la conexión donde uj=1), A(contorno donde uj= 0) y B(contorno donde los valores son desconocidos) que variarán a lo largo de j
uj = []; % Se declara la matriz que contendrá todos los valores de uj de los nodos de la Spider.
UB_j_t = []; % Se declara la matriz que contendrá los valores uj del contorno B.

% RECUPERACIÓN DE LAS CONDUCTANCIAS Y VALORES uj EN EL CONTORNO:
for j = 1:n
  R_j = [j]; % Posición donde está el contorno R. Este contorno variará de 1 hasta n (recorrerá todos los rayos).
   A_j = [(1+j): ((n-1)/2+j)]; % Se define el rango de índices que pertenecen al contorno A que serán la mitad de los nodos a partir del nodo donde se encuentra el contorno R.
  if (1+j)<=n & & ((n-1)/2+j) > n % En el caso que los nodos del contorno superen la totalidad de nodos, habrá que volver a reinicializar el recorrido de nodos.
      % En este caso, si suponemos que estamos en una Spider de 11 nodos y el contorno R se encuentra en el nodo 7, entonces los nodos que
       % forman parte del ontorno A son: 8, 9, 10, 11, 1 y por lo tanto en el caso del último nodo que superaría los nodos

      % iniciarse el recorrido.
    A1 = ((1+j):n); % Como se puede observar la submatriz A1 contendría desde el nodo 8 al 11.
    A2 = (1: ((n-1)/2+j)-n); % La submatriz A2 contendría los nodos de 1 a 1 para este ejemplo.
   A_j = [A1 A2]; % Se unen las dos submatrices A1 y A2 para construir el rango de los nodos del contorno
```

```
A.
  elseif ((n-1)/2+j) > n & & (1+j)>n % En el caso que tanto la primera parte del primer rango como la segunda sean superiores, se reinicializa la cuenta en las dos partes del rango.
      % Un ejemplo de este caso es si el contorno R está en el nodo 11, entonces el rango de nodos que forman parte del contorno A son 1, .2, 3, 4, 5
    A_j = [(1+j)-n: ((n-1)/2+j)-n];
  end

  B_j = [((n+1)/2+j):(n-1+j)]; % Se define el rango de índices que pertenecen al contorno B que serán la segunda mitad de los nodos.
  if (n-1+j) > n & & ((n+1)/2+j)<=n % Se reinicializa los nodos cuando se excede el número de nodos totales siguiendo el mismo procedimiento que para los nodos del contorno A.
     B1 = (((n+1)/2+j):n);
     B2 = (1: (n-1+j)-n);
     B_j = [B1 B2];
  elseif(n-1+j) > n & & ((n+1)/2+j)>n
     B_j = [((n+1)/2+j)-n: (n-1+j)-n];
  end

   UB_j = -inv(Nq(A_j,B_j))* Nq(A_j,j); % Se emplea la proposición 3.6 para obtener los valores de uj en los nodos del contorno B.
  vj_f_cons = [vj; UB_j]; % Con los valores de UB_j encontrados, se les añade éstos a la matriz incial que contendrá los valores de la función en los nodos de la Spider por cada contorno R.
  UB_j_t = [ UB_j_t, UB_j]; % Matriz con todos los valores uj del contorno B por cada contorno R posicionado en un nodo.
  if j == 1 % Cuando estamos en el caso que el contorno R se encuentra en el en el nodo 1, se mantiene la matriz que contiene las soluciones tal como está, ya que coinciden el valor de la j con la posición de las soluciones.
     vj_f_canv = vj_f_cons;
  end

   if j> 1 % En el caso que el contorno R se encuentra en otro nodo, hay que alinear la posición de los valores de uj con sus nodos respectivos. Por ello se irá girando los valores de la matriz en función donde se encuentre j (contorno R ubicado).
      vj_f_canv = [vj_f_cons((n-j+2): (n)); vj_f_cons(1:n-j+1)];% Cambio orden de las de los valores del contorno en función de la j que va variando.
  end

   uj = [uj, vj_f_canv]; % Se guarda en la matriz uj todos los valores de uj por cada cada contorno R, ya que será la matriz que contendrá todos los valores de los nodos de la Spider en todos los contornos R.
  c(j,j+n) = w(j)/w(j+n)* (Nq(j,j)-Nq(j,B_j)* inv(Nq(A_j,B_j))* Nq(A_j,j)-Lambda); % Se recupera la conductancias rediales que conectan los nodos del contorno con los nodos del primer círculo aplicando el Lemma 3.7.
   c(j+n,j) = c(j,j+n);% Se impone la simetría, ya que la conductancia es la misma de un nodo al otro o viceversa.

end

% RECUPERACIÓN DE VALORES DE uj EN EL PRIMER CÍRCULO.
m =1; % Se busca los valores uj en el primer círculo.
for j= 1:n
  B_j = [((n+1)/2+j):(n-1+j)]; %  Índices que pertenecen a las columnas de la matriz respuesta.
  if (n-1+j) > n & & ((n+1)/2+j)<=n % Si se supera los nodos del círculo, se reinicializará tal como se ha explicado enteriormente.
     B1 = (((n+1)/2+j):n);
```

```
    B2 = (1: (n-1+j)-n);
    B_j = [B1 B2];
  elseif(n-1+j) > n & & ((n+1)/2+j)>n
    B_j = [((n+1)/2+j)-n: (n-1+j)-n];
  end

  for k=1:n % Se busca los valores de uj en cada contorno R y por ello por cada contorno R, es decir j, se estudiará todos los valores del círculo utilizando la variable auxiliar k que recorrerá todos los nodos, es decir de 1 hasta n.
    uj(n* m+k,j)= 1/c(n* m+k, n* m+k-n)* (Lambda* uj(n* m+k-n,j)-Nq(k,j)-Nq(k,B_j)* UB_j_t(1:end, j)) + w(n* m+k)/w(n* m+k-n)* uj(n* m+k-n,j);
  end


end

% RECUPERACIÓN DE LAS CONDUCTANCIAS Y DE LOS VALORES uj EN LOS CÍRCULOS:
for m = 1: circulo % Se recorrerán todos los círculo que hay en el Spider
  for j = 1:n % Se recorrerán todos los nodos con el contorno R
    if m ==1 % En el caso que estemos en el primer círculo
      if j== 1 % Cuando estamos en el primer nodo: la conductancia adyacente entre el primer nodo y el anterior que es el último del círculo.
        c(n* m+j, n* m+n)= -uj(n* m-n+j,j)/uj(n* m+n,j)* c(n* m-n+j, n* m+j); % Se calcula la conductancia aplicando la Proposición 4.3.
          c(n* m+n, n* m+j)= c(n* m+j, n* m+n); % Se impone la simetria para obtener la matriz de conductancias simétrica.
       elseif(j>=2 & & j<=n) % Cuando estamos en otro nodo, es decir entre el segundo y el último, el cálculo de las conductancias es empleando la posición j para el nodo seleccionado y j-1 para el anterior.
          c(j+n* m,j+n* m-1)= -uj(n* m-n+j,j)/uj(j+n* m-1,j)* c(n* m-n+j,j+n* m);
          c(j+n* m-1, j+n* m)= c(j+n* m,j+n* m-1); % Se impone la simetría
      end
    end
    if m>1 % Cuando estamos en un círculo superior a 1 hay que ir retrociendo una posición en cada círculo, este efecto se contempla restando (m-1), es decir, si estamos en el segundo círculo se retrocede una posción, pero si estamos en el tercero círculo, se retrocede dos posición y así succesivamente.

      A= j+(n* m)-(m-1); % A será la variable que representará el nodo que se estudia que depende del círculo en que estemos y del contorno R.
       if A< (m)* n+1 % Como que se tiene que retroceder una posición o las posiciones necesaria que indica m-1. En el caso que el nodo anterior sea inferior al primer nodo del círculo, se le suma los nodos n para volver al círculo.
         A= j+(n* m)-(m-1)+n;
      end

     B= A-1; % B es el nodo anterior a A, por ello se le resta una posición. En el caso que el nodo adyacente inferior al primer nodo del círculo, se impone el valor del último nodo del mismo círculo como en el caso anterior. Por ejemplo, si estamos en el nodo 24 (j=2) y estamos en el segundo círculo, el nodo A es 23 y el nodo B debería ser el 33, por ello hay que imponer el valor del nodo B.
      if B< (m)* n+1
          B= A-1+n;
      end

        c(A,B)= -uj(A-n,j)/uj(B,j)* c(A-n,A); % Se aplica la Proposición 4.3 para calcular el valor de la conductancia adyacente.
      c(B,A)=c(A,B);% Se impone la simetría.
```

```
    end
  end

  % RECUPERACIÓN DE LA CONDUCTANCIA RADIAL:
  for t = 1:n % Se estudiará el recorrido del contorno R en todos los nodos n del contorno.
        if m ~ = circulo % Cuando nos encontramos en un círculo que no es el más interno, ya que el último círculo, las conductancias de los nodos de este círculo respecto el nodo central se calcula de otra manera que se especifica posteriorente.
          if m ==1 & & circulo ~ = 1 % Cuando estamos en el primer círculo y nos encontramos en una Spider de 7 nodos ya que sólo tiene un círculo y las conductancias radiales todas estan conectadas con X00
              if t==1 % Si estamos en el primer nodo del círculo, se estudia el nodo anterior que equivale al del último nodo del círculo. Al aplicar el operador "p" se valora la función en el nodo posterior, en el anterior y en el círculo anterior del mismo rayo. En la valoración en el nodo posterior, éste equivale al primer nodo del círculo, por ello se tiene que imponer n* m+1.
                % Veamos un ejemplo más claro si suponemos que el contorno R está en el nodo 1:
                 % c(22,33)=((c(22, 21)* uj(21,1) + c(22, 12)* uj(12,1) + c(22, 11)* uj(11,1))/uj(22,1) - (c(22, 21)* w(21) + c(22, 12)* w(12) + c(22,11)* w(11))/w(22))* w(22)/w(33)
                c(n* m+n,n* m+2* n)=((c(n* m+n, n* m+n-1)* uj(n* m+n-1,t) + c(n* m+n, n* m+1)* uj(n* m+1,t) + c(n* m+n, n* m)* uj(n* m,t))/uj(n* m+n,t) - (c(n* m+n, n* m+n-1)* w(n* m+n-1) + c(n* m+n, n* m+1)* w(n* m+1) + c(n* m+n, n* m)* w(n* m))/w(n* m+n))* w(n* m+n)/w(n* m+2* n);
                c(n* m+2* n, n* m+n)= c(n* m+n,n* m+2* n); % Se impone la simetría.

              elseif t ==2 % Si estamos en el segundo nodo del círculo, se estudia el nodo anterior que equivale al primer nodo del círculo. Al aplicar el operador "p" se valora la función en el nodo posterior, en el anterior y en el círculo anterior del mismo rayo. En la valoración en el nodo anterior, éste equivale al último nodo del círculo, por ello se tiene que imponer n* m+n.
                % Veamos un ejemplo más claro si suponemos que el contorno R está en el nodo 1:
                % c(12,23)=((c(12, 22)* uj(22,1) + c(12, 13)* uj(13,1) + c(12, 1)* uj(1,1))/uj(12,1) - (c(12, 22)* w(22) + c(12, 13)* w(13) + c(12,1)* w(1))/w(12))* w(12)/w(23)
               c(n* m+1,n* m+1+n)=((c(n* m+1, n* m+n)* uj(n* m+n,t) + c(n* m+1, n* m+2)* uj(n* m+2,t) + c(n* m+1, n* m+1-n)* uj(n* m+1-n,t))/uj(n* m+1,t) - (c(n* m+1, n* m+n)* w(n* m+n) + c(n* m+1, n* m+2)* w(n* m+2) + c(n* m+1, n* m+1-n)* w(n* m+1-n))/w(n* m+1))* w(n* m+1)/w(n* m+1+n);
                c(n* m+1+n, n* m+1)= c(n* m+1,n* m+1+n); % Se impone la simetría.

              elseif (3 <=t) & & (t<=n) % Si estamos en el nodo t del círculo (que es entre el tercero y el último), al aplicar el operador "p" se valora la función en el nodo posterior(t+1), en el anterior(t-1) y en el círculo anterior del mismo rayo(t-n).
                 % Suponemos que el contorno R está en el nodo 1 y estamos tratando el nodo t=4 del círculo:
                % c(15,26)=((c(15, 14)* va(14,1) + c(15, 16)* va(16,1) + c(15, 4)* va(4,1))/va(15,1) -(c(15, 14)* w(14) + c(15,16)* w(16) + c(15,4)* w(4))/w(15))* w(26)/w(15)
                 c(n* m+t-1,n* m+t-1+n)=((c(n* m+t-1, n* m+t-2)* uj(n* m+t-2,t) + c(n* m+t-1, n* m+t)* uj(n* m+t,t) + c(n* m+t-1, n* m+t-n-1)* uj(n* m+t-n-1,t))/uj(n* m+t-1,t) -(c(n* m+t-1, n* m+t-2)* w(n* m+t-2) + c(n* m+t-1, n* m+t)* w(n* m+t) + c(n* m+t-1, n* m+t-n-1)* w(n* m+t-n-1))/w(n* m+t-1))* w(n* m+t-1)/w(n* m+t-1+n); % Se calcula la conductancia radial aplicando la proposición 4.4.
                c(n* m+t-1+n, n* m+t-1)= c(n* m+t-1,n* m+t-1+n); % Se impone la simetría.
              end
          end


        end

        if m> 1 & & m ~ = circulo % En el caso que estamos en un círculo entre el primer círculo y el último. En este caso, se declara los valores Q, Z, G que representaran los valores del nodo actual, anterior
```

```
y posterior.
                Q = n* m + t-1 -(m-1); % El nodo a estudiar es Q, a medida que se avanza de círculo se
retrocede un nodo succesivamente.
                if Q< n* m +1 % Si al retroceder el número del nodo es inferior al primer nodo del círculo,
se le suma n nodos para volver a estar en el mismo círculo.
                  Q = n* m + t-1 -(m-1) +n;
                end

              Z= Q-1; % Z es el nodo anterior al nodo que se le aplica el operador "p". Si el nodo es inferior
al primer nodo del círculo se impone que vuelva al mismo círculo del rayo correspondiente.
                if Z< n* m +1
                  Z = Q-1+n;
                end
                  G= Q+1; % G es el nodo posterior al nodo que se le aplica el operador "p". Si este nodo
supera el último nodo del círculo, se impone que es el primer nodo ddel mismo círculo.
                if G> (m+1)* n
                  G = Q+1-n;
                end
                  c(Q,Q+n)=((c(Q,Z)* uj(Z,t) + c(Q, G)* uj(G,t) + c(Q, Q-n)* uj(Q-n,t))/uj(Q,t) -(c(Q, Z)*
w(Z) + c(Q, G)* w(G) + c(Q, Q-n)* w(Q-n))/w(Q))* w(Q)/w(Q+n); % Se calcula la conductancia radial
aplicando la proposición 4.4.
                c(Q+n, Q)= c(Q,Q+n); % Se impone la simetría.

        end

          if m== circulo % Cuando estamos en el último círculo, es decir, el más interior, el nodo del
siguiente círculo es común para todos los todos.
                Q = n* m + t-1 -(m-1); % El nodo a estudiar es Q, a medida que se avanza de círculo se
retrocede un nodo succesivamente.
                if Q< n* m +1 % Si al retroceder el número del nodo es inferior al primer nodo del círculo,
se le suma n nodos para volver a estar en el mismo círculo.
                  Q = n* m + t-1 -(m-1) +n;
                end

              Z= Q-1; % Z es el nodo anterior al nodo que se le aplica el operador "p". Si el nodo es inferior
al primer nodo del círculo se impone que vuelva al mismo círculo del rayo correspondiente.
                if Z< n* m +1
                  Z = Q-1+n;
                end
                  G= Q+1; % G es el nodo posterior al nodo que se le aplica el operador "p". Si este nodo
supera el último nodo del círculo, se impone que es el primer nodo ddel mismo círculo.
                if G> (m+1)* n
                  G = Q+1-n;
                end
                  c(Q,n* circulo+n+1)=((c(Q,Z)* uj(Z,t) + c(Q, G)* uj(G,t) + c(Q, Q-n)* uj(Q-n,t))/uj(Q,t)
-(c(Q, Z)* w(Z) + c(Q,G)* w(G) + c(Q, Q-n)* w(Q-n))/w(Q))* w(Q)/w(n* circulo+n+1); % Se calcula la
conductancia radial aplicando la proposición 4.4.
                c(n* circulo+n+1, Q)= c(Q,n* circulo+n+1); % Se impone la simetría.



        end

    end
    % PASO 6: SOLUCIÓN EN LOS NODOS INTERIORES
```

```
for r = 1: n % Se recorrerán todos los nodos con el contorno R
    for s= 1:n % Se busca los valores de uj en cada contorno R y por ello por cada contorno R, es decir r, se estudiará todos los valores del círculo utilizando la variable auxiliar s que recorrerá todos los nodos, es decir de 1 hasta n.
        if m ~ =circulo % Se impone la confición que el círculo no sea e más interno, ya que se calcula los valores de uj de cada círculo desde el círculo anterior, por el cual el valor de uj en el círculo fínal es el valor uj del centro y no lo necesitamos para el cálculo de las conductancias.
            if s==1 % En el caso que estemos en el primer nodo, al aplicar el operador "p" al nodo del mismo rayo pero en el círculo anterior, se valora la función en el nodo posterior, en el anterior y en el círculo anterior del mismo rayo. En la valoración en el nodo anterior, éste equivale al último nodo del círculo, por ello se tiene que imponer n* m+n.
                % Veamos un ejemplo considerando que el contorno r se encuentra en el primer nodo: uj(22,1) =+(uj(12,1)* (c(12,22)* w(22) + c(12,13)* w(13) +c(12,1)* w(1)))/(w(12)* c(12,23))-(c(12, 22)* uj(22,1) + c(12, 13)* uj(13,1) + c(12,1)* uj(1,1))/c(12,23)+w(23)/w(12)* uj(12,1)
                uj(n* m+1+n,r) = (uj(n* m+1,r)* (c(n* m+1, n* m+n)* w(n* m+n) + c(n* m+1, n* m+2)* w(n* m+2) + c(n* m+1, n* m+1-n)* w(n* m+1-n)))/(w(n* m+1)* c(n* m+1,n* m+1+n))-(c(n* m+1, n* m+n)* uj(n* m+n,r) + c(n* m+1, n* m+2)* uj(n* m+2,r) + c(n* m+1, n* m+1-n)* uj(n* m+1-n,r))/c(n* m+1,n* m+1+n)+w(n* m+1+n)/w(n* m+1)* uj(n* m+1,r); % Cálculo del valor uj mediante la proposición 4.5.

            elseif s==n % En el caso que estemos en el último nodo, al aplicar el operador "p" al nodo del mismo rayo pero en el círculo anterior, se valora la función en el nodo posterior, en el anterior y en el círculo anterior del mismo rayo. En la valoración en el nodo posterior, éste equivale al primer nodo del círculo, por ello se tiene que imponer n* m+1.
                % Veamos un ejemplo considerando que el contorno r se encuentra en el primer nodo: uj(33,1) =+(uj(22,1)* (c(22,21)* w(21) + c(22,12)* w(12) +c(22,11)* w(11)))/(w(22)* c(22,33))-(c(22, 21)* uj(21,1) + c(22, 12)* uj(12,1) + c(22, 11)* uj(11,1))/c(22,33)+w(33)/w(22)* uj(22,1)
                uj(n* m+2* n,r) =(uj(n* m+n,r)* (c(n* m+n, n* m+n-1)* w(n* m+n-1) + c(n* m+n, n* m+1)* w(n* m+1) + c(n* m+n, n* m)* w(n* m)))/(w(n* m+n)* c(n* m+n,n* m+2* n))-(c(n* m+n, n* m+n-1)* uj(n* m+n-1,r) + c(n* m+n, n* m+1)* uj(n* m+1,r) + c(n* m+n, n* m)* uj(n* m,r))/(c(n* m+n,n* m+2* n))+w(n* m+2* n)/w(n* m+n)* uj(n* m+n,r); % Cálculo del valor uj mediante la proposición 4.5.

            elseif (1 <s) & & (s<n) % Si estamos en el nodo s del círculo (que es entre el primero y el último), al aplicar el operador "p" al nodo del mismo rayo pero en el círculo anterior, se valora la función en el nodo posterior(s+1), en el anterior(s-1) y en el círculo anterior del mismo rayo(s-n).
                % Suponemos que el contorno R está en el nodo 1 y estamos tratando el nodo t=4 del círculo: uj(26,1)=+uj(15,1)* (c(15, 14)* w(14) + c(15, 16)* w(16) + c(15, 4)* w(4))/(w(15)* c(15, 26))- (c(15, 14)* uj(14,1) + c(15, 16)* uj(16,1) + c(15, 4)* uj(4,1))/c(15, 26)+w(26)/w(15)* uj(15,1);
                uj(n* m+s+n,r)=uj(n* m+s,r)* (c(n* m+s, n* m+s-1)* w(n* m+s-1) + c(n* m+s, n* m+s+1)* w(n* m+s+1) + c(n* m+s, n* m+s-n)* w(n* m+s-n))/(w(n* m+s)* c(n* m+s,n* m+s+n))- (c(n* m+s, n* m+s-1)* uj(n* m+s-1,r) + c(n* m+s, n* m+s+1)* uj(n* m+s+1,r) + c(n* m+s, n* m+s-n)* uj(n* m+s-n,r))/c(n* m+s, n* m+s+n)+w(n* m+s+n)/w(n* m+s)* uj(n* m+s,r); % Cálculo del valor uj mediante la proposición 4.5.


            end


        end
    end
end
disp(c) % imprime la matriz de conductancias
```