Métodos numéricos para matrices

Curso de Física Computacional

M. en C. Gustavo Contreras Mayén

Facultad de Ciencias - UNAM

19 de abril de 2018

Primer ejemplo: Vibraciones en una molécula

Supongamos que queremos estudiar el espectro de vibraciones de una molécula con n grados de libertad.

Primera aproximación

La primera aproximación consiste en investigar las oscilaciones armónicas del sistema, expandiendo la energía potencial hasta el segundo orden en las coordenadas generalizadas alrededor de las posiciones de equilibrio:

Primera aproximación

La primera aproximación consiste en investigar las oscilaciones armónicas del sistema, expandiendo la energía potencial hasta el segundo orden en las coordenadas generalizadas alrededor de las posiciones de equilibrio:

$$U(q_1, q_2, \ldots, q_n) \simeq \frac{1}{2} \sum_{i,j=1} A_{ij}\, q_i\, q_j$$

Energía potencial

$$U(q_1, q_2, \ldots, q_n) \simeq \frac{1}{2} \sum_{i,j=1} A_{ij}\, q_i\, q_j$$

donde q_i son las coordenadas generalizadas y A_{jk} son parámetros del potencial que usualmente pueden obtenerse a partir de un cálculo de química cuántica.

Energía cinética

La energía cinética puede escribirse en términos de las velocidades generalizadas.

$$T(\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n) \simeq \frac{1}{2} \sum_{i,j=1} M_{ij}\, \dot{q}_i\, \dot{q}_j$$

donde las $\dot{q}_i = dq_i/dt$ son las velocidades generalizadas, las M_{ij} son los elementos de masa generalizado en la matriz, cuyos valores dependen en particular de la molécula.

El Lagrangiano

Aplicando la ecuación de Lagrange

$$\frac{\partial \mathcal{L}}{\partial q_i} - \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_i} = 0$$

donde el lagrangiano del sistema es: $\mathcal{L} = T - U$, por tanto, tenemos que:

$$\sum_{j=1}^{n} \left(A_{ij}\, q_j + M_{ij}\, \ddot{q}_j \right) = 0, \qquad i = 1, 2, \ldots, n$$

Dependencia del tiempo de tipo oscilatorio

Si suponemos que la dependencia del tiempo en las coordenadas generalizadas es de tipo oscilatorio, con

$$q_j = x_j \, e^{-i\omega t}$$

Se obtiene el siguiente sistema de ecuaciones homogéneo:

$$\sum_{j=1}^{n} \left(A_{ij} - \omega^2 \, M_{ij} \right) x_j = 0, \qquad i = 1, 2, \ldots, n$$

Expresión matricial

La ecuación anterior se puede re-escribir en forma matricial, como.

$$\begin{pmatrix} A_{11} & \dots & A_{1n} \\ \vdots & \vdots & \vdots \\ A_{n1} & \dots & A_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \lambda \begin{pmatrix} M_{11} & \dots & M_{1n} \\ \vdots & \vdots & \vdots \\ M_{n1} & \dots & M_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

Expresión reducida

De manera equivalente

$$\mathbf{A}\,\mathbf{x} = \lambda\,\mathbf{M}\,\mathbf{x}$$

donde $\lambda = \omega^2$ es el valor propio y $\mathbf{x}$ es el correspondiente vector propio de la ecuación de valores propios.

Solución no trivial del sistema

Este es un sistema de ecuaciones lineales homogéneas; con el fin de contar con una solución no trivial del conjunto de ecuaciones, el determinante de la matriz de coeficientes debe anularse, esto es

$$det|\mathbf{A} - \lambda\, \mathbf{M}| = 0$$

Raíces de la ecuación matricial

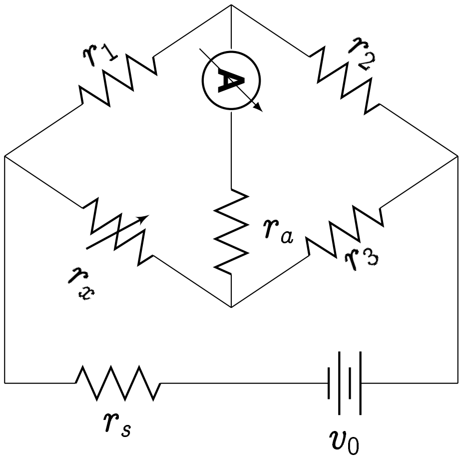
$$det\left|\mathbf{A} - \lambda\,\mathbf{M}\right| = 0$$

Las raíces de esta ecuación λ_k con $k = 1, 2, \ldots, n$ proporcionan todas las frecuencias angulares vibracionales de la molécula

$$\omega_k = \sqrt{\lambda_k}$$

Segundo ejemplo: Circuito eléctrico

Consideremos el siguiente circuito eléctrico:



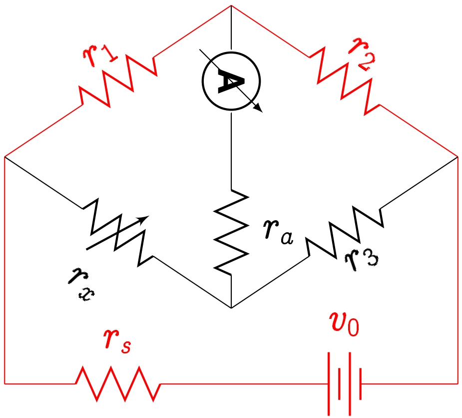
Solución para el circuito

Podemos aplicar las leyes de Kirchhoff para obtener un conjunto de ecuaciones que relacionan los voltajes y las corrientes del circuito, luego entonces, resolverlo y encontrar las incógnitas.

Solución para el circuito

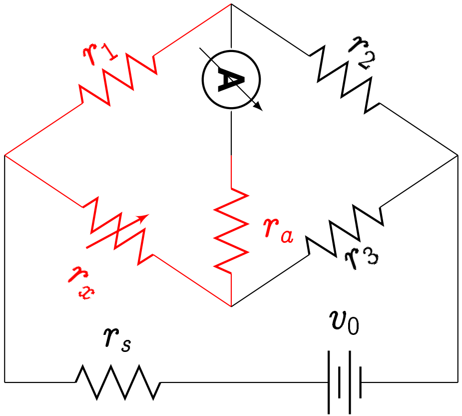
Pero ahora veamos el caso del puente desbalanceado de Wheastone: tenemos tres circuitos (o mallas) independientes:

Circuito/Malla independiente 1



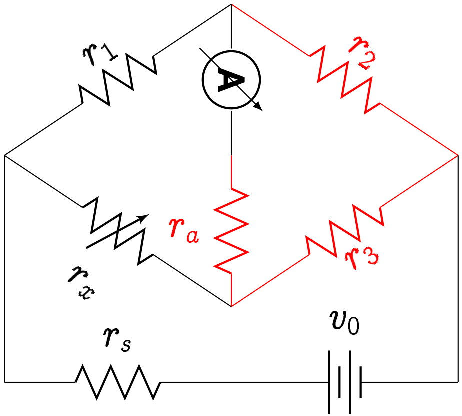
$$r_s\, i_1 + r_1\, i_2 + r_2\, i_3 = v_0$$

Circuito/Malla independiente 2



$$-r_x\, i_1 + (r_1 + r_x + r_a)\, i_2 + r_a\, i_3 = 0$$

Circuito/Malla independiente 3



$$-r_3\, i_1 - r_a\, i_2 + (r_2 + r_3 + r_a)\, i_3 = 0$$

Sistema de ecuaciones

Entonces el sistema de ecuaciones resulta ser:

$$\begin{aligned} r_s\, i_1 + r_1\, i_2 + r_2\, i_3 &= v_0 \\ -r_x\, i_1 + (r_1 + r_x + r_a)\, i_2 + r_a\, i_3 &= 0 \\ -r_3\, i_1 - r_a\, i_2 + (r_2 + r_3 + r_a)\, i_3 &= 0 \end{aligned}$$

Sistema de ecuaciones

Entonces el sistema de ecuaciones resulta ser:

$$\begin{aligned} r_s\, i_1 + r_1\, i_2 + r_2\, i_3 &= v_0 \\ -r_x\, i_1 + (r_1 + r_x + r_a)\, i_2 + r_a\, i_3 &= 0 \\ -r_3\, i_1 - r_a\, i_2 + (r_2 + r_3 + r_a)\, i_3 &= 0 \end{aligned}$$

El sistema anterior se puede escribir como

$$\mathbf{R}\,\mathbf{i} = \mathbf{v}$$

Descripción del sistema

$$\mathbf{R}\,\mathbf{i} = \mathbf{v}$$

donde la matriz de coeficientes de las resistencias es

$$\mathbf{R} = \begin{pmatrix} r_s & r_1 & r_2 \\ -r_x & r_1 + r_x + r_a & -r_a \\ -r_3 & -r_a & r_2 + r_3 + r_a \end{pmatrix}$$

Descripción del sistema

Los vectores columna de corrientes y voltajes son:

$$\mathbf{i} = \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} \qquad \mathbf{v} = \begin{pmatrix} v_0 \\ 0 \\ 0 \end{pmatrix}$$

Solución del sistema matricial

Multiplicando en ambos lados por $\mathbf{R}^{-1}$ (la matriz inversa de $\mathbf{R}$), la ecuación

$$\mathbf{R}\mathbf{i} = \mathbf{v}$$

resulta

$$\mathbf{i} = \mathbf{R}^{-1}\mathbf{v}$$

pero se requiere que conozcamos $\mathbf{R}^{-1}$.

Solución del sistema matricial

Multiplicando en ambos lados por $\mathbf{R}^{-1}$ (la matriz inversa de $\mathbf{R}$), la ecuación

$$\mathbf{R}\mathbf{i} = \mathbf{v}$$

resulta

$$\mathbf{i} = \mathbf{R}^{-1}\mathbf{v}$$

pero se requiere que conozcamos $\mathbf{R}^{-1}$.

¿Será necesario conocer siempre la matriz inversa?

Sistemas lineales algebraicos

En este tema analizamos la solución de sistemas de ecuaciones lineales algebraicos, con n incógnitas.

Sistemas lineales algebraicos

Cuando tomamos un problema físico, los conjuntos de ecuaciones a menudo son muy grandes, por lo que consumen una gran cantidad de recursos computacionales.

Uso de propiedades de la matriz

Por lo general, se reducen los requerimientos de almacenamiento y el tiempo de ejecución mediante el aprovechamiento de propiedades especiales de la matriz de coeficientes, tales como poca densidad (la mayoría de los elementos de un matriz dispersa son cero -las llamadas matrices *sparse*-)

Algoritmos específicos de solución

Existen muchos algoritmos dedicados a la solución de grandes conjuntos de ecuaciones, cada uno de ellos está adaptado a una determinada forma del coeficiente de la matriz (simétrica, de bandas, escaso, etc.)

Librerías de algoritmos

En internet se encuentran disponibles librerías para resolver cierto tipo de problemas, al usarlas se requiere conocer la manera en que se proporcionan los argumentos y las variables de resultado que proporcionan, se obliga la revisión de la documentación de cada uno de los códigos.

Definición de sistema algebraico

Un sistema algebraico tiene la forma

$$\begin{aligned} A_{11}x_1 + A_{12}x_2 + \ldots + A_{1n}x_n &= b_1 \\ A_{21}x_1 + A_{22}x_2 + \ldots + A_{2n}x_n &= b_2 \\ &= \vdots \\ A_{n1}x_1 + A_{n2}x_2 + \ldots + A_{nn}x_n &= b_n \end{aligned}$$

o sencillamente

$$\mathbf{A}\,\mathrm{x} = \mathrm{b}$$

Matriz aumentada

Una representación particularmente útil de las ecuaciones para los propósitos computacionales es la *matriz de coeficientes aumentada*, que se obtiene al juntar el vector constante b a la matriz de coeficientes A de la siguiente manera:

Matriz aumentada

$$[A|b] = \left[\begin{array}{cccc|c} A_{11} & A_{12} & \dots & A_{1n} & b_1 \\ A_{21} & A_{22} & \dots & A_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} & b_n \end{array}\right]$$

Unicidad de la solución

Un sistema de n ecuaciones lineales con n incógnitas tiene una solución única, siempre que el determinante de la matriz de coeficientes sea no singular, es decir, $|A| = 0$.

Unicidad de la solución

Las filas y columnas de una matriz no singular son linealmente independientes en el sentido de que no hay ninguna fila (o columna) que sea una combinación lineal de las otras filas (o columnas)

Soluciones infinitas

Si la matriz de coeficientes es singular, las ecuaciones pueden tener un número infinito de soluciones o no soluciones en absoluto, dependiendo del vector constante.

Caso 1

Como ejemplo veamos las ecuaciones

$$\begin{aligned} 2x + y &= 3 \\ 4x + 2y &= 6 \end{aligned}$$

Caso 1

Como ejemplo veamos las ecuaciones

$$\begin{aligned} 2x + y &= 3 \\ 4x + 2y &= 6 \end{aligned}$$

La segunda ecuación puede obtenerse multiplicando la primera ecuación por dos.

Caso 1

Cualquier combinación de x e y que satisface la primera ecuación es también una solución de la segunda.

El número de tales combinaciones es infinito.

Las dos rectas se superponen

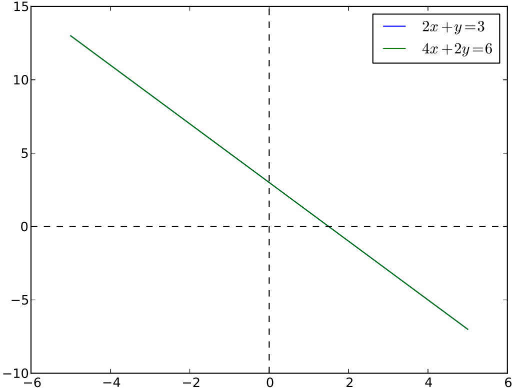


Figura 1: *Representación gráfica cuando la matriz de coeficientes es singular.*

Caso 2

En otro caso, las ecuaciones

$$\begin{aligned} 2\,x + y &= 3 \\ 4\,x + 2\,y &= 0 \end{aligned}$$

No tienen solución, ya que la ecuación equivalente $2\,x + y = 0$, contradice la primera.

Caso 2

En otro caso, las ecuaciones

$$
\begin{aligned}
2\,x + y &= 3\\
4\,x + 2\,y &= 0
\end{aligned}
$$

No tienen solución, ya que la ecuación equivalente $2\,x + y = 0$, contradice la primera.

Por tanto, cualquier solución que satisface la primera, no puede satisfacer la segunda.

Las dos rectas son paralelas

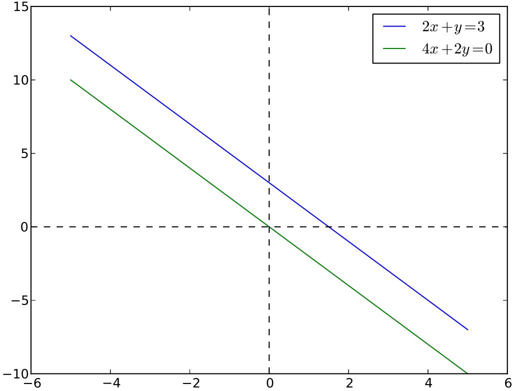


Figura 2: *Representación gráfica de un sistema algebraico sin solución.*

Problemas mal condicionados

Una pregunta obvia es: ¿qué sucede cuando la matriz de coeficientes es casi singular? es decir, si $|A|$ es muy pequeño?

Problemas mal condicionados

Con el fin de concluir si el determinante de la matriz es pequeño, necesitamos una referencia contra la cual el determinante se pueda medir.

Norma de una matriz

Esta referencia se denomina la norma de la matriz y se denota por $\parallel A \parallel$.

Entonces podemos decir que el determinante es pequeño si

$$|A| << \parallel A \parallel$$

Definición de la norma de una matriz

Existen diferentes maneras de calcular la norma de una matriz, tales como

$$\| A \| = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij}^2}$$

$$\| A \| = \max_{1 \leq i \leq j} \sum_{j=1}^{n} |A_{ij}|$$

Número de condición de la matriz

Una medida formal de condicionamiento está dada por el número de condición de la matriz, definido por:

$$\mathrm{cond}(A) = \| A \| \| A^{-1} \|$$

Si este número es cercano a la unidad, la matriz está bien condicionada.

Número de condición de la matriz

El número de condición aumenta con el grado de mal acondicionamiento, tendiendo a infinito para una matriz singular.

Toma en cuenta que el número de condición no es único, sino que depende de la elección de la norma de la matriz.

Número de condición de la matriz

Desafortunadamente, el número de condición es costoso de calcular para grandes matrices.

En la mayoría de los casos es suficiente para medir el condicionamiento comparando el determinante con las magnitudes de los elementos en la matriz.

Efectos del mal condicionamiento

Si las ecuaciones están mal condicionadas, pequeños cambios en la matriz de coeficientes dan como resultado grandes cambios en la solución.

Efectos del mal condicionamiento

A modo de ejemplo, consideremos las ecuaciones:

$$\begin{aligned} 2\,x + y &= 3 \\ 2\,x + 1.001\,y &= 0 \end{aligned}$$

que tiene la solución $x = 1501.5$, $y = -3000$

La norma es menor que los coeficientes

Dado que

$$|A| = 2\,(1.001) - 2\,(1) = 0.002$$

es mucho menor que los coeficientes, las ecuaciones están mal condicionadas.

La norma es menor que los coeficientes

Dado que

$$|A| = 2\,(1.001) - 2\,(1) = 0.002$$

es mucho menor que los coeficientes, las ecuaciones están mal condicionadas.

Usemos arreglos y las funciones de `python` para explorar este sistema.

Representación de las matrices

El sistema de ecuaciones algebraicas

$$
\begin{aligned}
2\,x + y &= 3 \\
2\,x + 1.001\,y &= 0
\end{aligned}
$$

Se va a representar de la siguiente forma:

Matrices $\mathbf{A}\,\mathbf{x} = \mathbf{b}$

$$\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 2 & 1.001 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} 3 \\ 0 \end{pmatrix}$$

El objeto `array`

El objeto que debemos de utilizar es el **`array`**, ya que es la manera con la que podremos aprovechar las funciones de `python`.

El objeto `array`

Nótese que la manera en que se ingresan los elementos del arreglo en `python`, es por renglones, en otros lenguajes de programación o programas, los elementos se pueden ingresar por columnas, siendo necesario revisar la documentación respectiva.

El objeto `array`

$$\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 2 & 1.001 \end{pmatrix}$$

```
A = array([[2., 1.], [2, 1.001]])
```

Algunas funciones en `linalg`

En `numpy.linalg` se tiene un conjunto de funciones útiles para las operaciones entre arreglos (podemos expresar el término *matrices* sin caer en complicaciones), entre otras podemos mencionar:

1. Producto entre matrices y vectores

Función	Descripción
`dot(a, b)`	Producto punto de dos arreglos.
`vdot(a, b)`	Producto punto de dos vectores.

2. Norma y otros valores

Función	Descripción
`norm(x)`	La norma de un arreglo o vector.
`cond(x)`	Calcula el número de condición de una matriz.
`det(a)`	Calcula el determinante de un arreglo.
`matrix_rank(M)`	Devuelve el rango de un arreglo.
`trace(a)`	Devuelve la suma a lo largo de la(s) diagonal(es) de un arreglo.

3. Solución de ecuaciones e inversa

Función	Descripción
`solve(a, b)`	Resuelve una ecuación matricial lineal o un sistema de ecuaciones escalares.
`inv(a)`	Calcula la inversa de una matriz.

Haciendo operaciones

Para las siguientes operaciones es posible utilizar una terminal de **`qtConsole`** o si tienes abierto **`Spyder`**, escribe las instrucciones en la misma ventana de la terminal:

Calculando el determinante

```
>>> from numpy import array, linalg
>>> a = array([[2., 1.], [2, 1.001]])
>>> b = array([3., 0.])
```

Calculando el determinante

```
>>> from numpy import array, linalg
>>> a = array([[2., 1.], [2, 1.001]])
>>> b = array([3., 0.])
```

Calculamos el valor del determinante

```
>>> print (linalg.det(a))
0.002
```

Número de condición

Ahora calculamos el valor del número de condición

```
>>> print (linalg.cond(a))
5001.00030004
```

Solución del sistema

Resolvemos el sistema algebraico

```
>>> print (linalg.solve(a,b))
[ 1501.5 -3000. ]
```

El sistema está mal condicionado

El efecto de los malos acondicionamientos se puede verificar mediante un cambio en la segunda ecuación:

$$\begin{aligned} 2x + y &= 3 \\ 2x + 1.002y &= 0 \end{aligned}$$

El sistema está mal condicionado

Creamos un nuevo arreglo c:

```
>>> c = array([[2.,1.],[2.,1.002]])
```

El sistema está mal condicionado

Creamos un nuevo arreglo c:

```
>>> c = array([[2.,1.],[2.,1.002]])
```

Calculamos el número de condición:

```
>>> print (linalg.cond(c))
2501.00060016
```

Resolvemos el sistema

Usando la función **`solve(a, b)`**, resolvemos el sistema

```
>>> print (linalg.solve(c,b))
[  751.5 -1500. ]
```

Un pequeño cambio provoca un enorme impacto

El resultado es $x = 751.5$, $y = -1500$.

Nótese que un cambio del $0.1\,\%$ en el coeficiente de y produce un cambio de $100\,\%$ en la solución!

Sistemas algebraicos lineales

Las ecuaciones algebraicas lineales, se presentan en casi todas las ramas del análisis numérico.

Su aplicación más visible es en el análisis de sistemas lineales: cualquier sistema cuya respuesta es proporcional a la entrada se considera que es lineal.

Sistemas algebraicos lineales

Los sistemas lineales incluyen estructuras, sólidos, elásticos, flujo de calor, filtraciones de líquidos, campos electromagnéticos, los circuitos eléctricos, etc.

Si el sistema es discreto, tal como una viga o un circuito eléctrico, entonces su análisis conduce directamente a ecuaciones algebraicas lineales.

Sistemas algebraicos discretos

Si el sistema es discreto, como en los esfuerzos en una viga o circuito eléctrico, entonces su análisis conduce directamente a ecuaciones algebraicas lineales.

Sistemas algebraicos discretos

En el caso de una viga estática, por ejemplo, las ecuaciones surgen cuando se describen las condiciones de equilibrio de los soportes.

Las incógnitas $x_1, x_2, \ldots, x_n$ representas las fuerzas en las uniones y las reacciones de soporte, mientras que las constantes $b_1, b_2, \ldots, b_n$ son las cargas externas.

Cambio a un sistema discreto

El comportamiento de los sistemas continuos se describen por ecuaciones diferenciales, en lugar de ecuaciones algebraicas.

Sin embargo, como el análisis numérico sólo puede ocuparse de variables discretas, es primero necesario aproximar una ecuación diferencial con un sistema de ecuaciones algebraicas.

Métodos conocidos de solución

Son conocidos los métodos de diferencias finitas, elementos finitos y métodos de contorno del elemento para el análisis de esta manera.

Métodos conocidos de solución

Estos métodos utilizan diferentes aproximaciones para conseguir una *discretización*, pero en cada caso la tarea final es el mismo: resolver un sistema (a menudo un sistema muy grande) de ecuaciones lineales algebraicas.

Expresiones de los sistemas algebraicos

El modelado de sistemas lineales invariablemente da lugar a las ecuaciones de la forma $\mathbf{A}\,\mathbf{x} = \mathbf{b}$, donde $\mathbf{b}$ es la entrada y $\mathbf{x}$ representa la respuesta del sistema.

El coeficiente de la matriz $\mathbf{A}$, refleja las características del sistema, es independiente de la entrada.

Expresiones de los sistemas algebraicos

En otras palabras, si se cambia la entrada, las ecuaciones tienen que ser resueltas de nuevo con una $\mathbf{b}$ diferente, pero el mismo $\mathbf{A}$.

Por lo tanto, es deseable tener un algoritmo que resuelva a ecuación la solución de algoritmo que puede gestionar cualquier número de vectores constantes con un mínimo esfuerzo computacional.

Métodos de solución

Hay dos clases de métodos para resolver sistemas de ecuaciones lineales algebraicas:

1. Métodos directos.

Métodos de solución

Hay dos clases de métodos para resolver sistemas de ecuaciones lineales algebraicas:

1. Métodos directos.
2. Métodos indirectos.

Métodos directos

La característica común de los métodos directos es que transforman las ecuaciones originales en ecuaciones equivalentes (ecuaciones que tienen la misma solución) que se pueden resolver más fácilmente.

Operaciones elementales

La transformación se lleva a cabo por la aplicación de las tres operaciones citadas a continuación.

Estas *operaciones elementales* no cambian la solución, pero pueden afectar el determinante de la matriz de coeficientes como se indica en el paréntesis.

Operaciones elementales

1. Intercambiar dos ecuaciones (cambia el signo de $|A|$)

Operaciones elementales

1. Intercambiar dos ecuaciones (cambia el signo de $|A|$)
2. Multiplicar una ecuación por una constante distinta de cero (se multiplica $|A|$ por la misma constante)

Operaciones elementales

1. Intercambiar dos ecuaciones (cambia el signo de $|A|$)
2. Multiplicar una ecuación por una constante distinta de cero (se multiplica $|A|$ por la misma constante)
3. Multiplicar una ecuación por una constante distinta de cero y luego restarlo de otra ecuación (deja a $|A|$ sin cambios)

Métodos indirectos

Los métodos iterativos, o métodos indirectos, comienzan con una suposición de la solución x, y luego repetidamente refinan la solución hasta alcanzar un criterio de convergencia.

Métodos indirectos

Los métodos iterativos son generalmente menos eficientes que sus homólogos directos debido al gran número de iteraciones necesarias.

Pero sí tienen ventajas computacionales importantes si la matriz de coeficientes es muy grande y escasamente pobladas (la mayoría de los coeficientes son cero).

Tres métodos directos

A continuación se enumeran los tres métodos directos más usados, cada uno de los cuales utiliza operaciones elementales para producir su propia manera de resolver ecuaciones.

Tres métodos directos

Método	Forma inicial	Forma final
Eliminación de Gauss	$A\,x = b$	$U\,x = c$
Descomposición LU	$A\,x = b$	$L\,U\,x = b$
Eliminación Gauss-Jordan	$A\,x = b$	$I\,x = c$

Donde $\mathbf{U}$ representa una matriz triangular superior, $\mathbf{L}$ es una matriz triangular inferior e $\mathbf{I}$ es la matriz identidad.

Tres métodos directos

Una matriz cuadrada se llama triangular si contiene sólo elementos cero en un lado de la diagonal principal.

Matriz U

Una matriz triangular superior U de 3×3 tiene la forma

$$U = \begin{bmatrix} U_{11} & U_{12} & U_{13} \\ 0 & U_{22} & U_{23} \\ 0 & 0 & U_{33} \end{bmatrix}$$

Matrices L

Mientras que una matriz triangular inferior L de 3×3 es del tipo:

$$L = \begin{bmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{bmatrix}$$

Matrices triangulares

Las matrices triangulares juegan un papel importante en el álgebra lineal, ya que simplifican muchos cálculos.

Matrices triangulares

Por ejemplo, consideremos la ecuación $L\,x = c$:

$$
\begin{aligned}
L_{11}\,x_1 &= c_1 \\
L_{21}\,x_1 + L_{22}\,x_2 &= c_2 \\
L_{31}\,x_1 + L_{32}\,x_2 + L_{33}\,x_3 &= c_3 \\
&\vdots
\end{aligned}
$$

Matrices triangulares

Si se resuelven las ecuaciones hacia delante, comenzando con la primera ecuación, los cálculos son muy fáciles, ya que cada ecuación contiene sólo una incógnita a la vez.

Sustitución hacia adelante

La solución se obtiene haciendo:

$$
\begin{aligned}
x_1 &= \frac{c_1}{L_{11}} \\
x_2 &= \frac{(c_2 - L_{21}\, x_1)}{L_{22}} \\
x_3 &= \frac{(c_3 - L_{31}\, x_1 - L_{32}\, x_2)}{L_{33}} \\
&\dots
\end{aligned}
$$

Este procedimiento se conoce como *sustitución hacia adelante*.

Sustitución hacia atrás

De una manera similar la operación $U\,x = c$, se encuentra en el proceso de eliminación de Gauss, y puede resolverse fácilmente por sustitución hacia atrás, que se inicia con la última ecuación y va retrocediendo a través de las ecuaciones.

Sustitución hacia atrás

Las ecuaciones $L\,U\,x = b$, que están asociadas con la descomposición $L\,U$, también pueden resolverse rápidamente si las sustituimos con dos conjuntos de ecuaciones equivalentes:
$L\,y = b$ y $U\,x = y$.

Sustitución hacia atrás

Ahora $L\,y = b$ se puede resolverse para y por sustitución hacia delante, seguido de la solución de $U\,x = y$ por medio de la sustitución hacia atrás.

Operaciones con la matriz identidad

Las ecuaciones $I\,x = c$, que se generan en la eliminación Gauss-Jordan, son equivalentes a $x = c$ (recordemos la identidad $I\,x = x$), de modo que c es la solución.

Ejercicio 1

Determinar si la siguiente matriz es singular:

$$A = \begin{pmatrix} 2.1 & -0.6 & 1.1 \\ 3.2 & 4.7 & -0.8 \\ 3.1 & -6.5 & 4.1 \end{pmatrix}$$

Solución al ejercicio 1

Usando lo que hemos visto respecto a las funciones de **`numpy.linalg`**, declaramos en una variable el arreglo $\mathbf{A}$:

```
>>> A = array([[2.1, -0.6, 1.1], \
[3.2, 4.7, -0.8], \
[3.1, -6.5, 4.1]])
```

Solución al ejercicio 1

Calculamos el valor del determinante

```
>>> print (linalg.det(A))
0.0
```

Resultado

$$A = \begin{bmatrix} 2.1 & -0.6 & 1.1 \\ 3.2 & 4.7 & -0.8 \\ 3.1 & -6.5 & 4.1 \end{bmatrix}$$

La matriz A es singular, ya que el determinante vale cero.

Ejercicio 2

Resolver la ecuación $A\,x = b$, donde

$$A = \begin{bmatrix} 8 & -6 & 2 \\ -4 & 11 & -7 \\ 4 & -7 & 6 \end{bmatrix} \qquad b = \begin{bmatrix} 28 \\ -40 \\ 33 \end{bmatrix}$$

El módulo `scipy.linalg`

El módulo `scipy.linalg` extiende las funciones que hemos mencionado anteriormente de `numpy.linalg`.

Incluye un conjunto de funciones para resolver problemas matriciales de una manera en particular.

Funciones básicas en `scipy.linalg`

Función	Descripción
`inv(a)`	Calcula la inversa de una matriz.
`solve(a, b)`	Resuelve el sistema $ax = b$, para una matriz a cuadrada.
`solve_banded`	Resuelve el sistema $a\,x = b$, donde x es una matriz en banda.
`det(a)`	Calcula el determinante de una matriz.
`norm(a)`	Calcula la norma de una matriz.

Factorizaciones en `scipy.linalg`

Función	Descripción
`lu(a)`	Factoriza LU de una matriz a, devuleve la matriz pivo P, la matriz triangular inferior L y la matriz triangular superior U.
`lu_factor(a)`	Factoriza $L\,U$ de una matriz a, devuelve la matriz pivote y las matrices $L\,U$ superpuestas.
`lu_solve()`	Resuelve el sistema $a\,x = b$, dada una factorización $L\,U$.

Abrimos un archivo nuevo en Spyder I

Obtenemos ahora con la librería `scipy.linalg`, las matrices L y U:

Código 1: Código para obtener las matrices LU

```
from scipy import array, linalg

A = array([[8., -6. , 2.],[-4., 11., 
    -7.],[4., -7, 6.]])

P, L, U = linalg.lu(A)

print ('P')
print (P)
```

Abrimos un archivo nuevo en Spyder II

```python

print ('L')
print (L)

print ('U')
print (U)
```

¿Qué devuelve la función

Lo que nos devuelve la función son tres arreglos: P, L y U, que son las matrices pivote, triangular inferior y triangular superior, respectivamente.

Resultado de la factorización

La descomposición $L\,U$ de la matriz de coeficientes A es:

$$A = LU = \begin{bmatrix} 1. & 0. & 0. \\ -0.5 & 1. & 0. \\ 0.5 & -0.5 & 1 \end{bmatrix} \begin{bmatrix} 8. & -6. & 2. \\ 0. & 8. & 6. \\ 0. & 0. & 2. \end{bmatrix}$$

Recuperando la matriz inicial A

Para verificar que efectivamente el producto de las matrices L y U nos regresa la matriz A, basta con que realicemos el producto de matrices:

Recuperando la matriz inicial A

Si usamos el operador multiplicación:

```
>>> print (L*U)
array([[  0. ,   -6. ,    4. ],
       [ -0. ,    8. , -12. ],
       [  0. ,  -0.5,    4. ]])
```

Recuperando la matriz inicial A

Si usamos el operador multiplicación:

```
>>> print (L*U)
array([[  0. ,   -6. ,    4. ],
       [ -0. ,    8. ,  -12. ],
       [  0. ,   -0.5,    4. ]])
```

Vemos que el resultado no es la matriz inicial A, ¿por qué?

Una operación que no devuelve lo esperado

Lo que obtenemos es una matriz que es el producto "entrada" por "entrada" de las matrices L y U, por lo que no es nuestra matriz inicial A.

Una operación que no devuelve lo esperado

La manera de recuperar la matriz A, es mediante el correcto uso del producto de dos arreglos, para esto debemos de utilizar la función `dot`, recordemos que se puede usar hasta de tres maneras distintas:

`dot` con escalares y matrices

1. Cuando multiplicamos un escalar por un escalar, nos devuelve un escalar.

```
>>> dot(3, 4)
>>>12
```

dot con escalares y matrices

2. Cuando multiplicamos un escalar por un arreglo, nos devuelve un arreglo.

```
>>> dot(-1, L)
>>> array([[-1. ,  0. ,  0. ],
           [ 0.5, -1. ,  0. ],
           [-0.5,  0.5, -1. ]])
```

`dot` con escalares y matrices

3. Cuando multiplicamos un arreglo por un arreglo, nos devuelve un arreglo.

```
>>> dot(L, U)
```

Uso de la función dot

Entonces la manera en que debemos de usar la función `dot` es la siguiente:

```
>>> dot(L,U)
array([[  8.,  -6.,   2.],
       [ -4.,  11.,  -7.],
       [  4.,  -7.,   6.]])
```

Que corresponde a la matriz inicial A.

Solución mediante sustituciones

Una vez conocidas las matrices L y U, podemos realizar los procedimientos de sustutición hacia adelante para L, y luego la sustitución hacia atrás para U, así conoceremos la solución del problema.

Solución para L

Resolvemos primero $L\,y = b$ con sustitución hacia adelante:

$$
\begin{aligned}
y_1 &= 28 \\
-\frac{y_1}{2} + y_2 &= -\,40 \quad \rightarrow \quad y_2 = -40 + \frac{y_1}{2} = -26 \\
\frac{y_1}{2} - \frac{y_2}{2} + y_3 &= 33 \quad \rightarrow \quad y_3 = 33 + \frac{y_2}{2} + \frac{y_1}{2} = 6
\end{aligned}
$$

Solución para U

La solución x ahora se obtiene de $Ux = y$ por sustitución hacia atrás:

$$
\begin{aligned}
2x_3 &= y_3 && \rightarrow && x_3 = y_3/2 = 3 \\
8x_2 - 6x_3 &= y_2 && \rightarrow && x_2 = \frac{y_2 + 6x_3}{8} = -1 \\
8x_1 - 6x_2 + 2x_3 &= y_1 && \rightarrow && x_1 = \frac{y_1 + 6x_2 - 2x_3}{8} = 2
\end{aligned}
$$

Solución para U

La solución x ahora se obtiene de $Ux = y$ por sustitución hacia atrás:

$$
\begin{aligned}
2x_3 = &y_3 \quad \rightarrow \quad x_3 = y_3/2 = 3 \\
8x_2 - 6x_3 = &y_2 \quad \rightarrow \quad x_2 = \frac{y_2 + 6x_3}{8} = -1 \\
8x_1 - 6x_2 + 2x_3 = &y_1 \quad \rightarrow \quad x_1 = \frac{y_1 + 6x_2 - 2x_3}{8} = 2
\end{aligned}
$$

Por tanto la solución es $x = [2, -1, 3]^T$