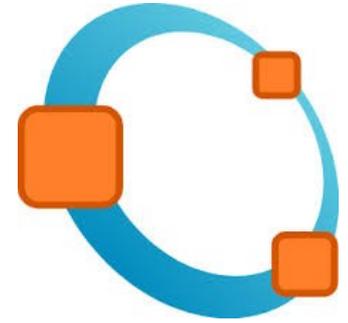


Introducción a Octave

Unidad 2



Daniel Millán

San Rafael, Argentina Marzo-Abril 2019



Departamento de
Ingeniería Mecánica



UNCUYO
UNIVERSIDAD
NACIONAL DE CUYO



FACULTAD DE
**CIENCIAS APLICADAS
A LA INDUSTRIA**



Operaciones con vectores y matrices

1. Vectores/matrices en la ventana de órdenes.
2. Operaciones con vectores/matrices.
3. Tipos de datos. Números reales: `float`, `double`.
4. Variables y expresiones matriciales.
5. Tipos de matrices predefinidos.
6. Operador `(:)`. Matriz vacía `[]`. Borrado filas/columnas.
7. Guión de órdenes (script): creación, edición y ejecución.
8. V/M a partir de un fichero y mediante funciones y declaraciones.
9. Operadores relacionales. Operadores lógicos.



1. Vectores/matrices en la ventana de órdenes

- Los vectores se pueden definir directamente introduciendo los elementos que lo componen, p.ej.
`>> u=[1,2,3] % vector fila`
- Observamos que **u** es un vector fila.
- En el caso de desear un vector columna utilizamos ;
`>> v=[1;2;3] % vector columna`

Ejercicio: ¿Es posible calcular la suma de **u** y **v**?

- Para transformar un vector fila en columna se debe transponer dicho vector mediante la orden ' (comilla).
- Octave genera por defecto vectores fila:
`>> w(1)=1; w(2)=4; w(3)=9;`
`>> w`



1. Vectores/matrices en la ventana de órdenes

- Las matrices se crean introduciendo los elementos
>> **A=[1 4 -3; 2 1 5; -2 5 3]**
- Adicionalmente es posible crear matrices mediante vectores filas/columnas
>> **f1=[1 4 -3]; f2=[2 1 5]; f3=[2 5 -3];**
>> **Af=[f1;f2;f3]**
>> **c1=[1 2 2]; c2=[4 1 5]; c3=[-3 5 -3];**
>> **Ac=[c1,c2,c3]**

Ejercicio: Calcular la transpuesta de **A := A^T**.

Ejercicio: Calcular y comprobar la inversa: **A⁻¹ = inv(A)**.



2. Operaciones con vectores/matrices

- Operadores aritméticos:
 - + adición o suma
 - sustracción o resta
 - * multiplicación
 - ' traspuesta
 - ^ potenciación
 - \ división-izquierda
 - / división-derecha
 - . * producto elemento a elemento
 - ./ y .\ división elemento a elemento
 - .^ elevar a una potencia elemento a elemento



2. Operaciones con vectores/matrices

- Operadores matriciales elemento a elemento (*, ^, \ y /). Para ello basta precederlos por un punto (.)

```
>> [1 2 3 4].^2
```

```
>> [1 2 3 4].*[1 -1 1 -1]
```

```
>> [1 -1;1 -1].^3
```

```
>> [1 2;3 4]*[1 -1;1 -1]
```

```
>> [1 2;3 4].*[1 -1;1 -1]
```

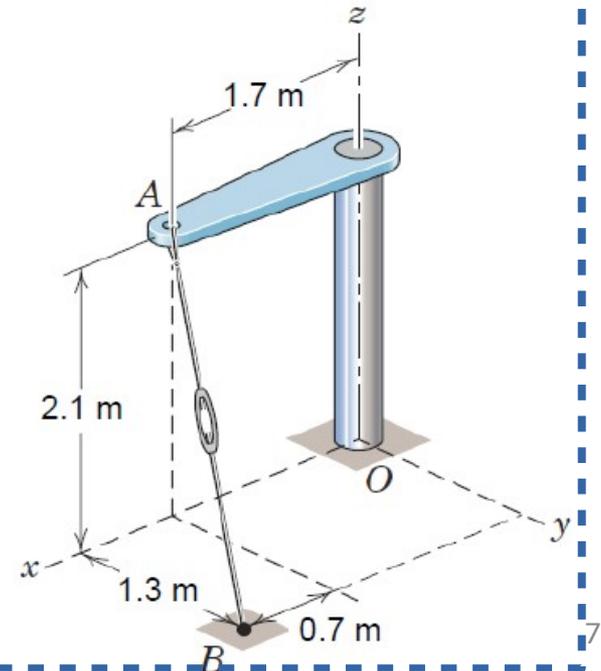
```
>> [1 2;3 4]./[1 -1;1 -1]
```

```
>> [1 2;3 4].\[1 -1;1 -1]
```

2. Operaciones con vectores/matrices

- Producto escalar, vectorial y tensorial
 - >> `u=[1,2,3]; v=[1,1,1];`
 - >> `dot(u,v)` %producto escalar o punto ($u \cdot v'$)
 - >> `cross(u,v)` %producto vectorial o cruz
 - >> `u'*v` %producto tensorial o abierto

Ejercicio: El tensor de la figura, se ajusta hasta que la tensión del cable AB es de 2.5kN. Determinar, con OCTAVE, el momento $M = r \times T$ respecto al punto O de la tensión del cable, que actúa en el punto A, y la magnitud de ese momento [¿Unidades?].





2. Operaciones con Vectores/Matrices

- Resolución de sistemas lineales

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

donde \mathbf{A} es una matriz invertible de $N \times N$ (filas x columnas), \mathbf{x} y \mathbf{b} son vectores columna de $N \times 1$ no nulos.

- La resolución de este sistema de ecuaciones se puede realizar de 2 formas diferentes

$$\mathbf{x} = \text{inv}(\mathbf{A}) * \mathbf{b}$$

$$\mathbf{x} = \mathbf{A} \backslash \mathbf{b}$$

Ejercicio: Dada la matriz $\mathbf{A} = [1 \ 4 \ -3; 2 \ 1 \ 5; -2 \ 5 \ 3]$ y el vector $\mathbf{b} = [1; 2; 3]$ obtenga el vector solución \mathbf{x} mediante **2 formas diferentes** y compruebe los resultados obtenidos (resto).



3. Tipos de datos. Números reales: float, double

- Octave es un programa preparado para trabajar con vectores y matrices. Como caso particular emplea variables escalares (matrices de dimensión 1).
- Octave trabaja siempre en **doble precisión**, es decir guardando cada dato en 8 bytes, con unas 15 cifras decimales exactas.
- También puede trabajar con cadenas de caracteres (*strings*) y con otros tipos de datos: **Tensores de más de dos dimensiones, matrices esparsas, vectores y matrices de celdas, estructuras**, etc.
- Octave utiliza **Inf** para aquellos números más grandes que lo que es capaz de representar y **NaN** (Not a Number) para resultados que no están definidos como un número.

```
>> 1/0 >> 0/0 >> 1/Inf >> Inf/Inf
```



4. Variables y expresiones matriciales

- Una **variable** es un nombre que se da a una entidad numérica, que puede ser una matriz, un vector o un escalar.
- El valor de una variable se modifica mediante el **operador de asignación** (=).

- Una **expresión** puede tener las dos formas siguientes: primero, asignando su resultado a una variable,

variable = expresión

y segundo evaluando simplemente el resultado del siguiente modo,

expresión

en cuyo caso el resultado se asigna automáticamente a una variable interna llamada **ans** (de *answer*) que almacena el último resultado obtenido.



5. Tipos de matrices predefinidos

- **eye(4)** forma la matriz unidad de tamaño (4×4)
- **zeros(3,5)** forma una matriz de *ceros* de tamaño (3×5)
- **zeros(4)** ídem de tamaño (4×4)
- **ones(2,3)** forma una matriz de *unos* de tamaño (2×3)
- **linspace(0,1,7)** genera un vector con 7 valores igualmente espaciados entre 0 y 1
- **rand(3)** crea una matriz de números aleatorios entre 0 y 1, con distribución uniforme, de tamaño (3×3)
- **randn(4)** matriz de números aleatorios de 4×4, con distribución normal, de valor medio 0 y varianza 1.
- **magic(4)** crea una matriz (4×4) con los números 1, 2, ... 4*4, tal que todas las filas y columnas suman lo mismo.
- **hilb(5)** matriz de Hilbert de 5×5. El elemento (i,j) está dado por $(1/(i+j-1))$. Esta matriz produce grandes errores numéricos al resolver sistemas lineales $\mathbf{A} \mathbf{x} = \mathbf{b}$.



5. Tipos de matrices predefinidos

Formación de un matriz a partir de Otras ya definidas

- recibiendo alguna de sus propiedades (por ejemplo el tamaño),
- por composición de varias submatrices más pequeñas,
- modificándola de alguna forma.

Ejercicio: Dada la matriz $A=\text{rand}(3)$ y el vector $x=[1;2;3]$, comprobar:

- ✓ $[m, n]=\text{size}(A)$ devuelve el número de filas y de columnas de la matriz A .
 $n=\text{length}(x)$ calcula el número de elementos de un vector x .
- ✓ $\text{zeros}(\text{size}(A))$ forma una matriz de *ceros* del mismo tamaño que una matriz A
 $\text{ones}(\text{size}(A))$ ídem con *unos*
- ✓ $A=\text{diag}(x)$ forma una matriz diagonal A cuyos elementos diagonales son los elementos del vector x .
- ✓ $x=\text{diag}(A)$ forma un vector x a partir de los elementos de la diagonal de A .
- ✓ $\text{diag}(\text{diag}(A))$ crea una matriz diagonal a partir de la diagonal de la matriz A .
 $\gg B=\text{diag}(\text{diag}(A))$
 $\gg C=[A, \text{eye}(3); \text{zeros}(3), B]$

6. Operador (:). Matriz vacía []. Borrado filas/columnas

- El operador “:” es muy importante y puede usarse de varias formas.

¡Probar y jugar, es la mejor forma de aprender!.

Arreglo de números o vector

```
>> x=1:10
>> x=1:2:10
>> x=1:1.5:10
>> x=10:-1:1

>> x=[0.0:pi/50:2*pi]';
>> y=sin(x); z=cos(x);
>> [x y z]
```

Matrices

```
>> A=magic(5)
>> A(2,3)
>> A(5,1:4)
>> A(3,:)
>> A(end,:)
>> A(3:5,:)
>> A([1 2 5],:)
>> A(:) % vector columna
>> B=eye(size(A));
>> B([2 4 5],:)=A(1:3,:)
```

6. Operador (:). Matriz vacía []. Borrado filas/columnas

- Una matriz definida sin ningún elemento entre los corchetes es una matriz que existe, pero que está vacía, o lo que es lo mismo que tiene *dimensión cero*.

```
>> A=magic(3)
```

```
>> B=[]
```

```
>> exist("B"), exist("C")
```

```
>> isempty(A), isempty(B)
```

```
>> A(:,3)=[]
```

- Las funciones *exist()* e *isempty()* permiten chequear si una variable existe y si está vacía.
- En el último ejemplo se ha eliminado la 3^a columna de A asignándole la matriz vacía.



7. Archivos *.m que poseen un guión de órdenes (*scripts*).

- Los archivos con extensión .m son ficheros de texto sin formato (ficheros ASCII) que constituyen el centro de la programación en Octave.
- Estos archivos se crean y modifican con un editor de textos cualquiera.
- Octave 4 en adelante provee su propio editor de textos, que es también *Debugger*.
- Existen dos tipos de archivos *.m, los *ficheros de guiones de comandos* (llamados *scripts* en inglés) y las *funciones*.
- Los *scripts* contienen un conjunto de comandos (programa) que se ejecutan secuencialmente cuando se teclea el nombre del archivo en la ventana de órdenes o se incluye dicho nombre en otro fichero *.m.

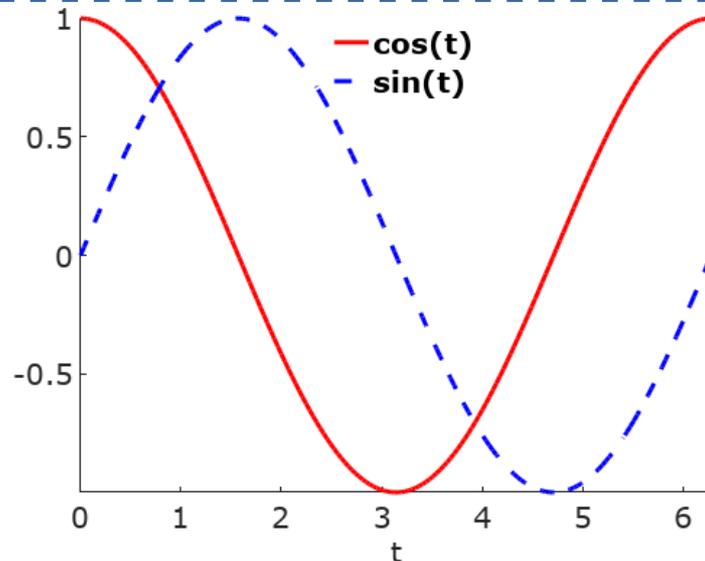


7. Creación, edición, ejecución y depuración de *scripts*.

- El *script* se puede llamar con **F5** desde el editor o desde la línea de órdenes.
- El *script* crea variables que pertenecen al **espacio de trabajo base** de Octave, y permanecen en él cuando se termina la ejecución de dicho programa.
- Un *script* puede llamar a otros *scripts*.

Ejercicio: graficar las funciones $\cos(t)$ y $\sin(t)$ en $[0, 2\pi]$.

Para graficar emplear `ezplot`.





8. V/M desde: archivo, funciones y declaraciones

- Es posible emplear un archivo ***nombre.m*** (extensión ***.m***) que contiene instrucciones, vectores, matrices y/o funciones.
- Un archivo ****.m*** puede llamar a otros ficheros ****.m***, e incluso puede llamarse a sí mismo (funciones recursivas).
 - Las variables definidas dentro de un archivo de órdenes ****.m*** (**script**), o guión de órdenes, que se ejecuta desde la línea de órdenes, son variables del **espacio de trabajo** y pueden ser accedidas desde fuera de dicho archivo.
 - Si un archivo de órdenes se llama desde una **function**, las variables que se crean pertenecen al espacio de trabajo de dicha función.
- **Estos dos tipos de archivos de órdenes constituyen aspectos relevantes que serán vistos en más detalle en futuras unidades.**

Ejercicio: crear un fichero llamado ***unidad.m*** que construya una matriz unidad de tamaño 3×3 llamada ***U33***.



8. V/M desde: archivo, funciones y declaraciones

- Se pueden definir las matrices y vectores por medio de *funciones de librería* y de *funciones programadas por el usuario* (que se verán más adelante).



9. Operadores relacionales. Operadores lógicos.

Operadores relacionales, se aplican a vectores y matrices:

< menor que

> mayor que

<= menor o igual que

>= mayor o igual que

== igual que

~= distinto que

- La comparación se realiza elemento a elemento, y el resultado es otra matriz de unos y ceros del mismo tamaño

```
>> A=[1 2;0 3]; B=[4 2;1 5];
```

```
>> A==B
```

```
>> A~=B
```



9. Operadores relacionales.

Operadores lógicos.

Operadores lógicos:

& *and* (función equivalente: **and(A,B)**). Se evalúan siempre ambos operandos, y el resultado es **true** sólo si ambos son **true**.

&& *and* breve: si el primer operando es **false** ya no se evalúa el segundo, pues el resultado final será **false**.

| *or* (función equivalente: **or(A,B)**). Se evalúan ambos operandos, y el resultado es **false** sólo si ambos son **false**.

|| *or* breve: si el primer operando es **true** no se evalúa el segundo, pues el resultado final no puede ser más que **true**.

~ *negación lógica* (función equivalente: **not(A)**)

xor(A,B) realiza un "or exclusivo", es decir, devuelve 0 en el caso en que ambos sean 1 ó ambos sean 0.

- Los operadores lógicos se combinan con los relacionales para poder comprobar el cumplimiento de condiciones múltiples.²⁰