Numpy

Funciones Universales y reglas "Broadcasting"



Francisco José Madrid Cuevas

• Doctor en Informática por la Universidad Politécnica de Madrid en 2003. Desde 1996 he sido profesor a tiempo completo de la Universidad de Córdoba impartiendo docencia en Informática en varias titulaciones de Ingeniería.





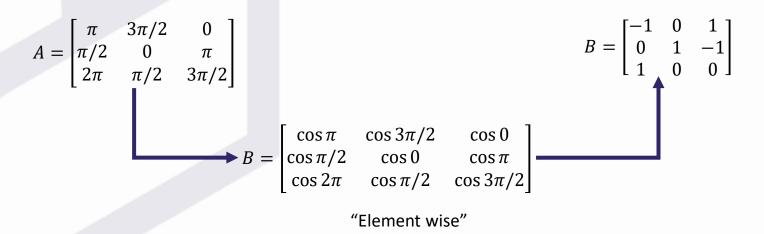
Contenidos

- Qué es una función universal.
 - Vectorización del código.
- Reglas de Broadcasting.



¿Qué es una función universal?

• Calcular el coseno de los valores de un arreglo.





¿Qué es una función universal?

• Uso de funciones universales para vectorizar el código.

```
B = \begin{bmatrix} \cos \pi & \cos 3\pi/2 & \cos 0\\ \cos \pi/2 & \cos 0 & \cos \pi\\ \cos 2\pi & \cos \pi/2 & \cos 3\pi/2 \end{bmatrix}
```

Código sin vectorizar

Código vectorizado

ufunc



¿Qué es una función universal?

- Funciones universales en Numpy.
 - Funciones universales disponibles:
 - Operaciones aritméticas y funciones matemáticas.
 - Funciones trigonométricas.
 - Funciones de manipulación de bits.
 - <u>Funciones de comparación.</u>
 - <u>Funciones para punto flotante.</u>



• Funciones universales con dos argumentos.

C = np.add(A, B)

$$1+1=2$$

$$A\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} + B\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = C\begin{bmatrix} 2 & 3 & 4 \\ 5 & 6 & 7 \\ 8 & 9 & 10 \end{bmatrix}$$
Forma: (3,3) (3,3) (3,3)

¿Qué ocurre cuando las formas de los argumentos no coinciden?



- Reglas a aplicar antes de realizar una ufunc con X_1 , X_2 ,... argumentos:
 - 1. Todo argumento X_i con X_i .ndim < max_dim = max $\{X_1$.ndim, X_2 .ndim, ... $\}$ preañaden dimensiones con tamaño 1 hasta alcanzar max_dim dimensiones.
 - 2. La salida tendrá max_dim dimensiones, cada una con el máximo tamaño encontrado.
 - 3. Un argumento X_i será válido si el tamaño de cada dimensión es 1 o igual al tamaño máximo de todos los argumentos para esa dimensión.
 - 4. Si un argumento X_i tiene tamaño 1 para una dimensión, el valor de ese elemento se utiliza para todas las operaciones en esa dimensión.



Ejemplo 1: C = np.add(A, B)

$$A\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} + B[1 & 1 & 1]$$

Forma (3,3)

$$\begin{bmatrix}
2 & 3 & 4 \\
5 & 6 & 7 \\
8 & 9 & 10
\end{bmatrix}$$
Forma (3,3)

- 1. $max_dim = max\{A.ndim=2, B.ndim=1\} = 2$
- 2. La forma de B.shape=(3,) pasa a B'.shape = (1, 3)
- 3. La forma de A se mantiene (3, 3).
- 4. La salida C tendrá forma (3, 3) (máximo tamaño en cada dimensión).
- 5. A y B' tienen tamaños 1 o 3 en las dos dimensiones.
- 6. El elemento B'[1] = [1 1 1] se usa para sumar en la primera dimensión



Ejemplo 2: C = np.add(A, B)

$$A\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} + B\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$
 Forma (3,1)
Forma (3,3)

- 1. max_dim = max{A.ndim=2, B.ndim=2} = 2
- 2. La forma de B.shape=(3,1) se mantiene.
- 3. La forma de A.shape=(3, 3) se mantiene.
- 4. La salida C tendrá forma (3, 3) (máximo tamaño en cada dimensión).
- 5. A y B tienen tamaños 1 o 3 en las dos dimensiones.
- El elemento B[_,1] = [1] se usa para sumar en la segunda dimensión.



• Ejemplo 3: C = np.add(A, B)

$$A\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} + B[1 \quad 1]$$
Forma (2,)

- 1. $max_dim = max\{A.ndim=2, B.ndim=1\} = 2$
- 2. La forma de B.shape=(1,) pasa a B'.shape = (1, 2)
- 3. La forma de A.shape=(3, 3) se mantiene.
- 1. La salida C tendrá forma (3, 3) (máximo tamaño en cada dimensión).
- 5. B tiene tamaño 2 en la segunda dimensión que no es ni 1 ni 3.
- 6. Se genera un ERROR y no se calcula C.



