BIOL 4505/6505

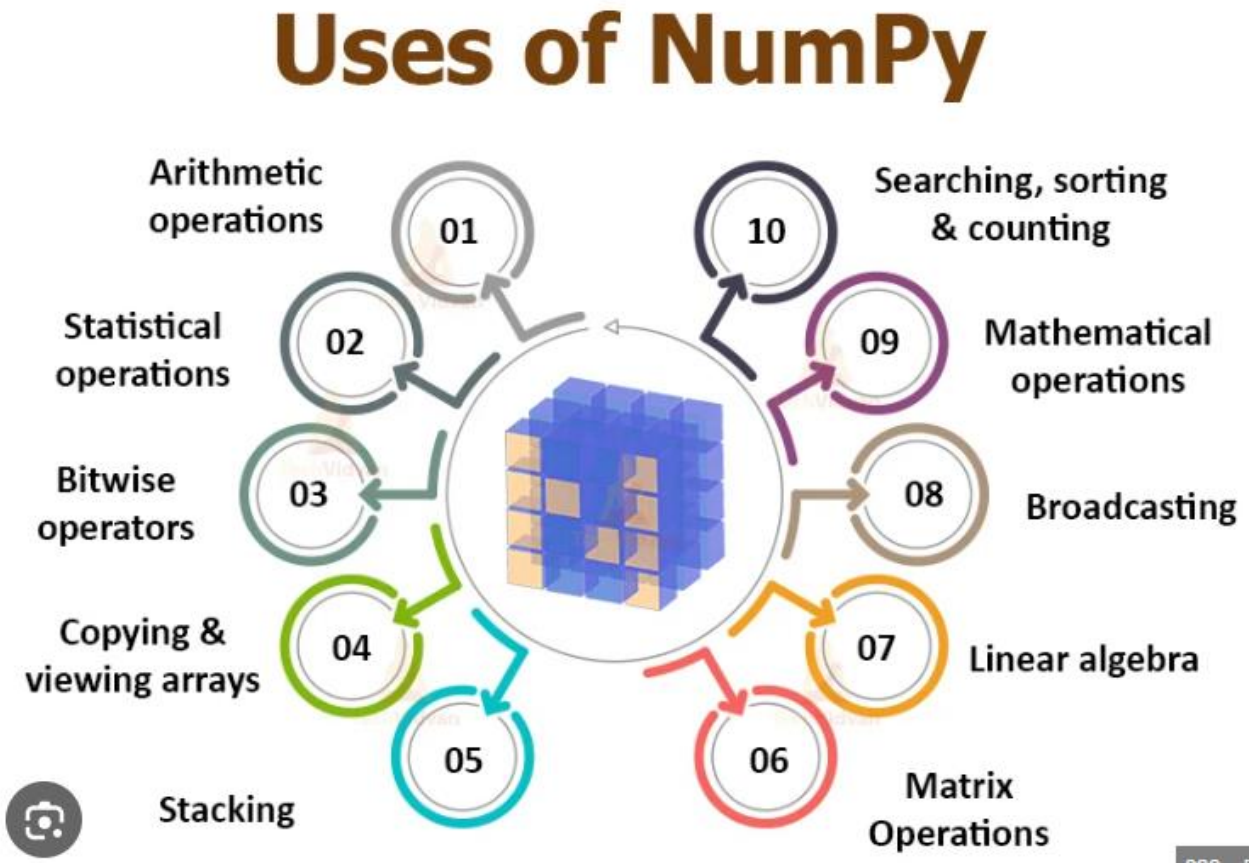
Programming in Biological and Health Sciences

Patrick McGrath

October 20, 2025

Georgia Institute of Technology

NumPy (Numerical Python) is extremely powerful for mathematical applications

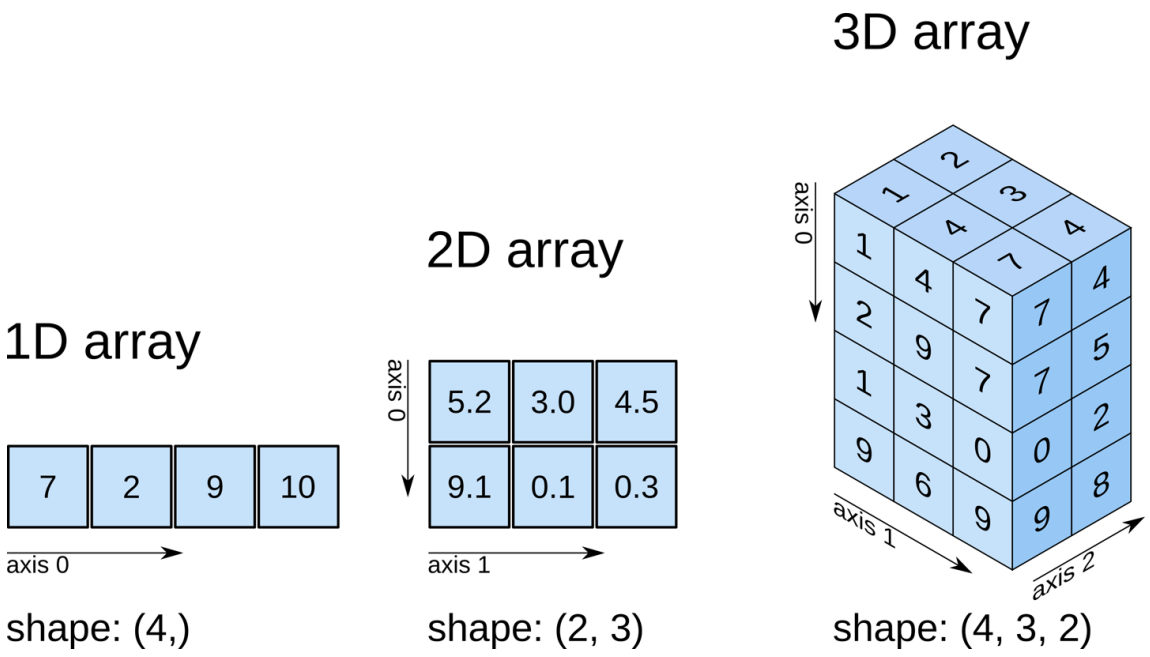


- Memory
- Speed
- Used by many other important modules

Things we will cover today

- Arrays -> how data is held by numpy
  - Dimension
  - Shape
  - Data type
- Examples
- Accessing data in arrays
  - Accessing individual elements
  - Filtering data

Each array has dimensionality, shape, and data type



Data Type	Description
bool_	Boolean (True or False) stored as a byte
int	Default integer type
intc	Identical to C inte.g int32 in 64
intp	Integer used for indexing
int8	Byte (-128 to 127)
int16	Integer (-32768 to 32767)
int32	Integer (-2147483648 to 2147483647)
int64	Integer (-9223372036854775808 to 9223372036854775807)
uint8	Unsigned integer (0 to 255)
uint16	Unsigned integer (0 to 65535)
uint32	Unsigned integer (0 to 4294967295)
uint64	Unsigned integer (0 to 18446744073709551615)
float 16	Half precision float: sign bit, 5 bits exponent, 10 bits mantissa
float32	Single precision float: sign bit, 8 bits exponent, 23 bits mantissa
float64	Double precision float: sign bit, 11 bits exponent, 52 bits mantissa
complex64	Complex number, represented by two 32-bit floats (real and imaginary components)
complex 128	Complex number, represented by two 64-bit floats (real and imaginary components)

```
(Pdb) g
<GenotypeArray shape=(1675692, 3, 2) dtype=int8>
0|1 0|1 0|0
0|0 1|0 0|1
0|0 1|0 0|1
...
0|1 0|0 0|0
0|1 0|1 0|0
0|1 0|0 0|0

(Pdb) g.ndim
3
(Pdb) g.shape
(1675692, 3, 2)
(Pdb) g.dtype
dtype('int8')
```

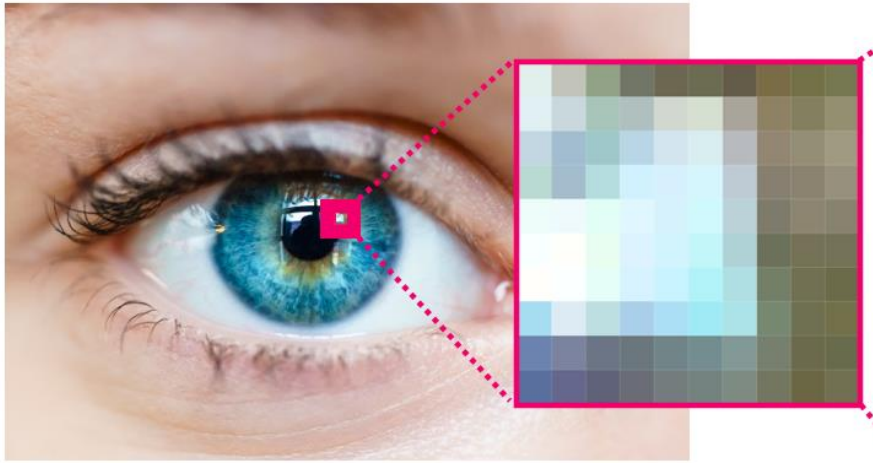
Differences between lists and arrays

	Python List	NumPy Array
Data Types	Different data types	Single type
Size	Dynamic	Fixed
Performance	Slower	Faster
Functionality	Basic operations (append, insert, etc.)	Advanced mathematical operations
Use Case	General-purpose, mixed data types	Numerical computations, large datasets

CodeCut

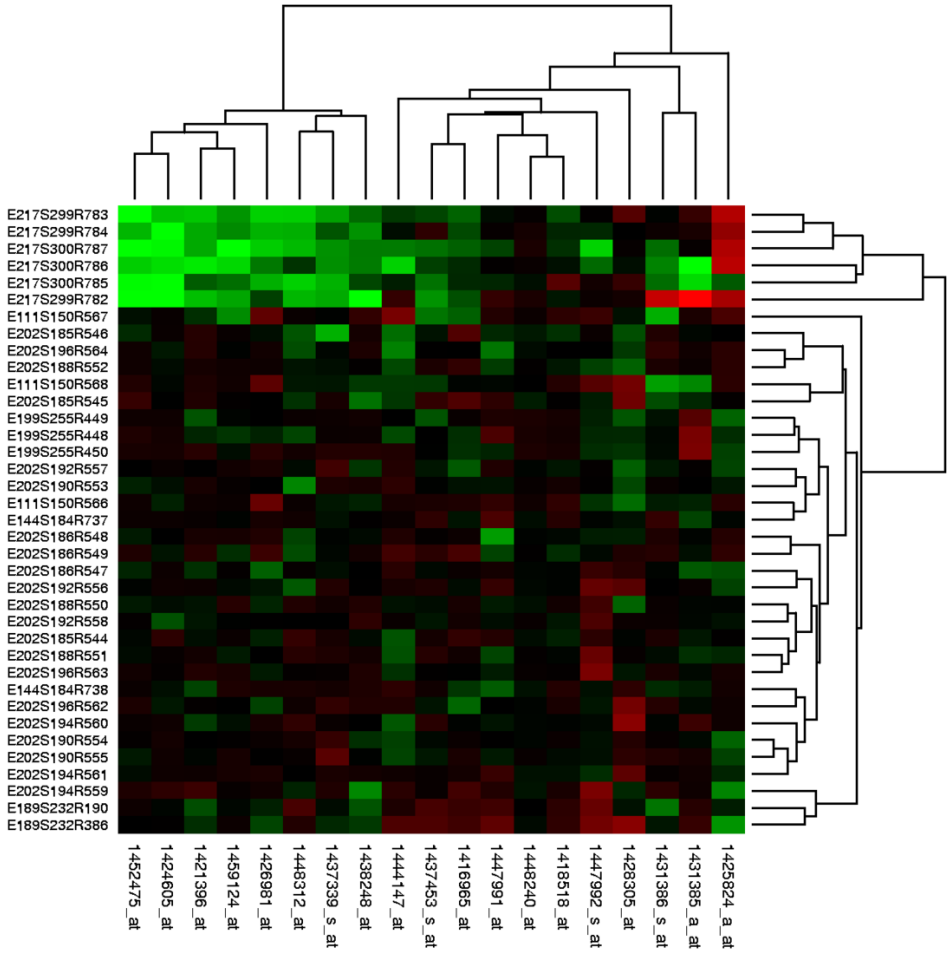
Image data often stored as numpy arrays

- 3D – 2 dimensions for x and y, 1 dimension for color

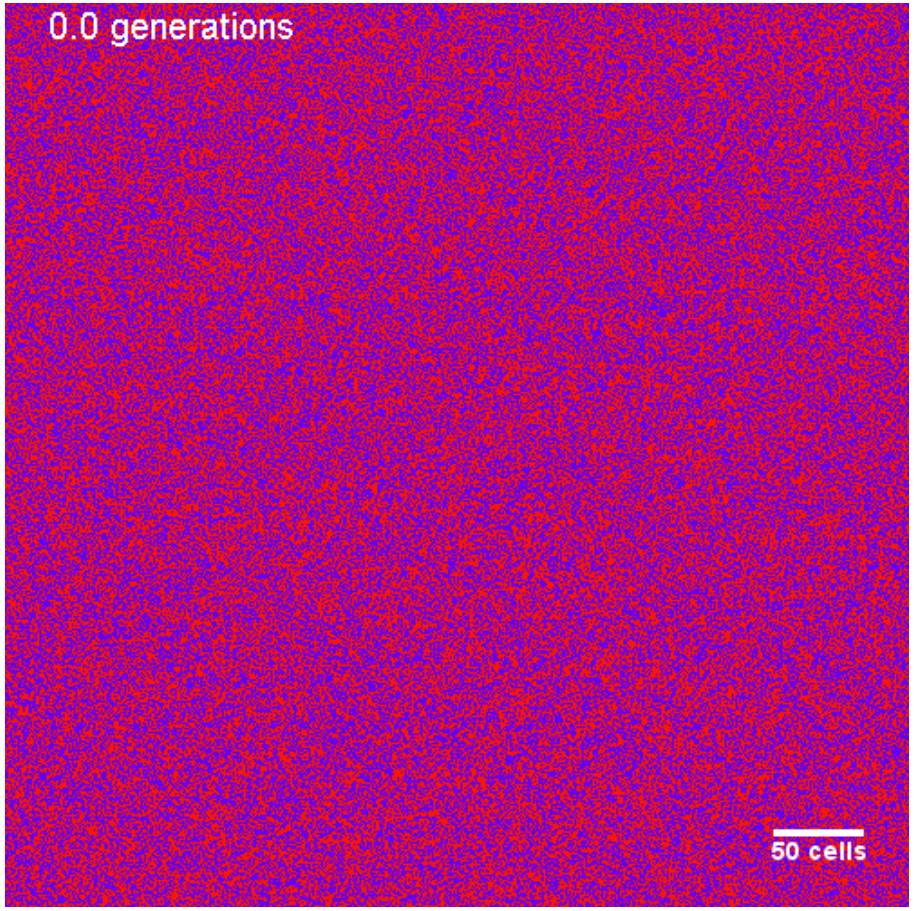


			233	188	137	96	90	95	63	73	73	82
	237	202	159	120	105	110	88	107	112	121	109
226	191	147	110	101	112	98	123	110	119	142	131
221	191	176	182	203	214	169	144	133	145	155	122
185	160	161	184	205	223	186	137	147	161	140	115
181	174	189	207	206	215	194	136	142	151	133	87
246	237	237	231	208	206	192	122	143	144	111	74
254	254	241	224	199	192	181	99	122	117	107	74
239	248	232	207	187	182	184	110	114	110	113	74
193	215	193	167	158	164	181	114	112	111	105	82
113	119	110	111	113	123	135	120	108	106	113	
93	97	91	103	107	111	122	112	104	114		

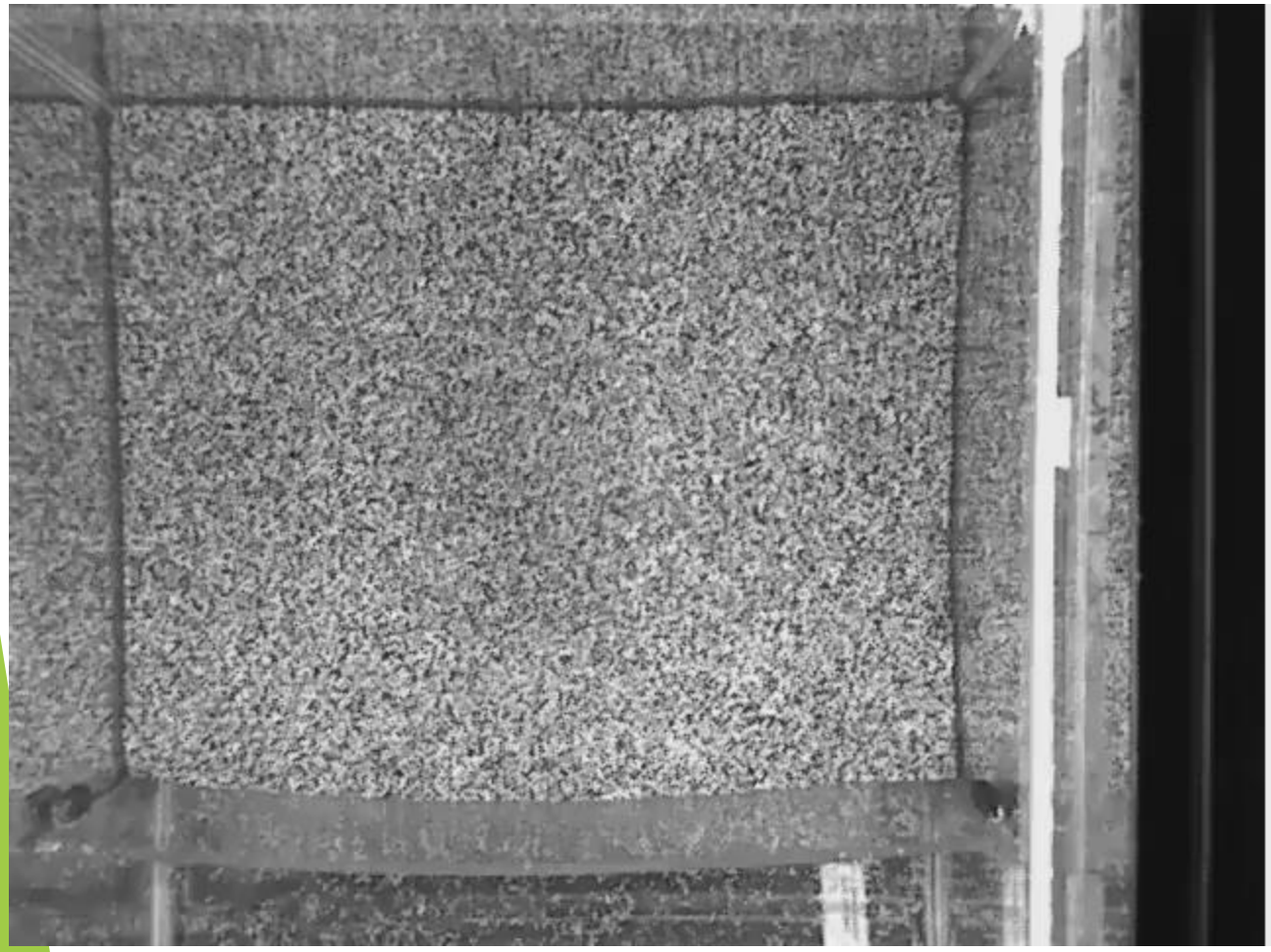
Gene expression data naturally fits as an array

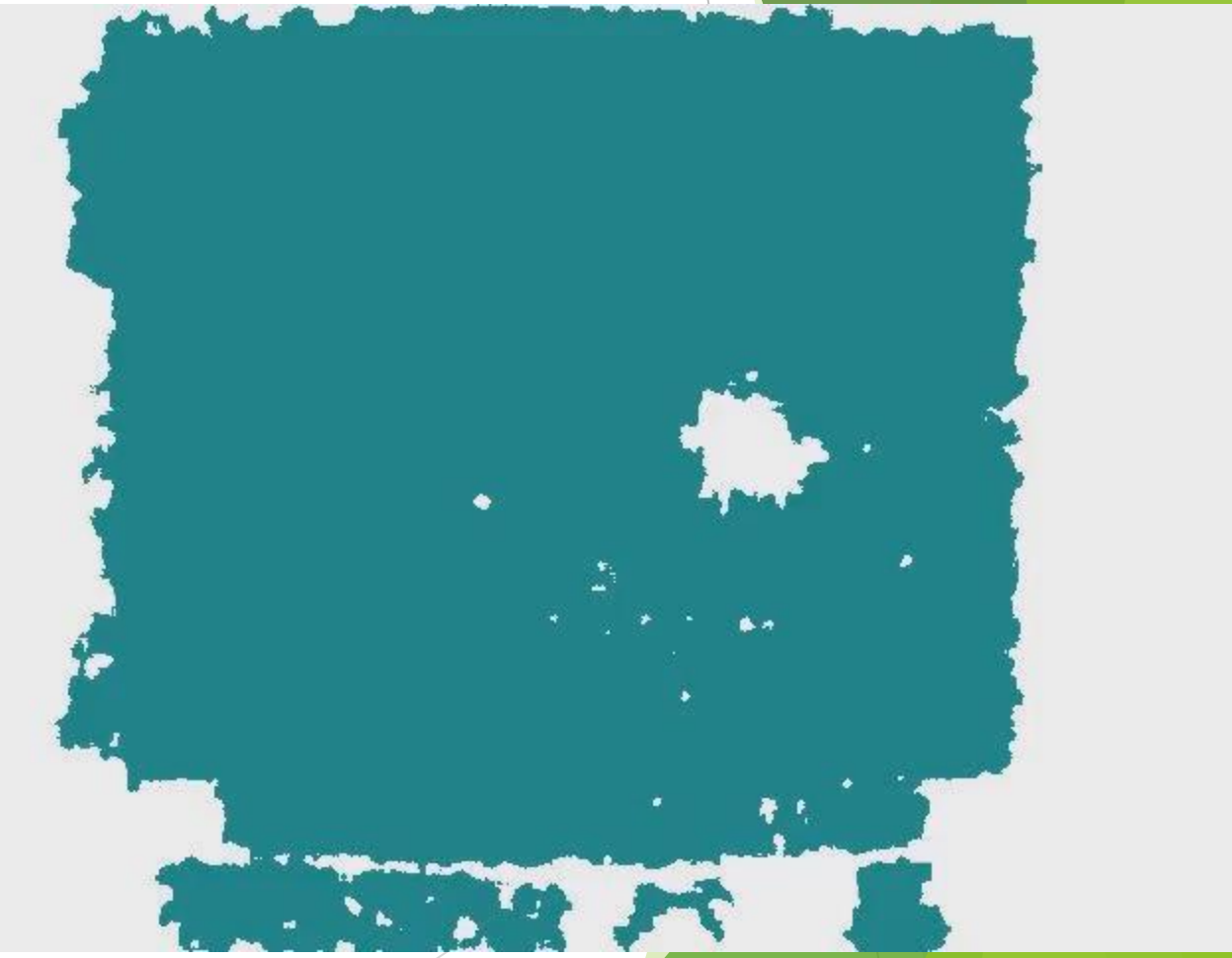


Mutual killers create spatial patterning



Depth Data provides real time view of current bower structure





How to access data.

```
>>> a = numpy.fromfunction(lambda i,j: 10*i+j,(6,6))
```

```
>>> a[0, 3:5]
array([3,4])

>>> a[4:, 4:]
array([[44, 45],
       [54, 55]])

>>> a[:, 2]
array([2, 12, 22,
      32, 42, 52])

>>> a[2::2, ::2]
array([[20, 22, 24],
       [40, 42, 44]])
```

0	1	2	3	4	5
10	11	12	13	14	15
20	21	22	23	24	25
30	31	32	33	34	35
40	41	42	43	44	45
50	51	52	53	54	55

All arrays can take in Booleans to filter data

```
import numpy as np
arr=np.array([1,2,3,4])
print("Original array:",arr)
a=[True,False,False,True]
b=arr[a]
print("Filtered array:",b)
```

```
Original array: [1 2 3 4]

Filtered array: [1 4]
```

Booleans arrays are created by conditionals

filter.py > ...

```python
import numpy as np

list_1 = [1, 2, 3, 4, 5, 6, 7, 8, 9]
numbers = np.array(list_1)
evens = numbers[numbers % 2 == 0]

print(evens)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

```
PS E:\Filter functions> python filter.py
[2 4 6 8]
PS E:\Filter functions>
```

Missing Data is encoded as np.nan

- np.nan (not a number) is a float type value used as a placeholder in arrays for undefined or missing data
- NaN values affect mathematical operations and need to be worked around
- You can see what values are nan using np.isnan()
- NaN values can be filtered with Booleans

```
>>> import numpy as np
>>> arr = np.array([1, np.nan, 3, 4, 5, 6, np.nan])
>>> arr.sum()
np.float64(nan)
>>> np.nansum(arr)
np.float64(19.0)
>>> np.isnan(arr)
array([False,  True, False, False, False, False,  True])
>>> mask = np.isnan(arr)
>>> mask
array([False,  True, False, False, False, False,  True])
>>> ~mask
array([ True, False,  True,  True,  True,  True, False])
>>> arr[~mask]
array([1., 3., 4., 5., 6.])
>>>
```

Today’s Assignment

- Read in a numpy array of depth data generated from a bower building male cichlid in the McGrath Lab
- Use numpy to summarize the attributes of the array
- Use numpy and matplotlib to visualize the change in sand depth between multiple days of bower building