## CSCI2202: Lecture 9 Randomness and Probability

Finlay Maguire (<u>finlay.maguire@dal.ca</u>) TA: Ehsan Baratnezhad (<u>ethan.b@dal.ca</u>) TA: Precious Osadebamwen (<u>precious.osadebamwen@dal.ca</u>)

### Overview

- Numpy basics and gotcha
- Randomness
- Pseudorandom Number Generators
- Numpy PRNG functions
  - Using PRNGs to estimate values (Monte Carlo)
- Probability Distributions
  - Estimating outcomes by sampling from distributions
- Optimisation
  - Grid-search vs Random-search
- Simulation
  - Calculating p-values by simulating Null distribution
  - Bootstrapping to estimate confidence intervals

### Numpy

NumPy (**Num**erical **Py**thon): numerical/array library that forms basis of most of scientific python

Core ndarray object that supports arrays with arbitrary numbers of dimensions/axes.

Operates similar to nested lists (index, slicing, nested dimensions) but with many special numerical methods.

Limitations for speed: elements only 1 type, semi-mutable (change values/shape but not number of entries), rectangular (i.e., each row must have same number of columns)



```
https://numpy.org/doc/stable/user/absolute_beginners.html
```

```
import numpy as np
np.zeros(3) # array([0., 0., 0.])
np.ones(2) # arrav([1..1.])
a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print(a)
arrav([[1. 2. 3].
      [4, 5, 6],
      [7, 8, 9]])
a[0,1] # 2
a[:, 2] # arrav([2, 5, 8])
a.shape # (3, 3)
a.reshape(1, 9) # array([[1, 2, 3, 4, 5, 6, 7, 8,
9]])
a.sum() # np.int64(45)
```

### Numpy/Pandas: Views vs Copies

- Space efficiency: NumPy tries to avoid making copies of data
- Many NumPy operations e.g., subset of an array create a VIEW
- Like alias but to just a portion of the original array
- Changing a view will change part of the original array
- If want a **COPY** then make an explicit copy (slow): ndarray.copy
- As Pandas is built on numpy it often does the same thing (one of the main sources of "warnings" in Pandas).



```
a = np.array([1, 2, 3, 4, 5, 6])
а
array([1, 2, 3, 4, 5, 6])
b = a[3:]
b
array([4, 5, 6])
b[0] = 40
а
array([ 1, 2, 3, 40, 5, 6])
```

### Randomness

### Randomness is an essential tool in science

Almost every area of science makes use of randomness:

**Sampling:** Random subset of group can be used to create a smaller unbiased representation of group

**Optimisation:** Efficiently sampling possible values to find "best" combination of parameters (e.g., model fitting)



https://www.scribbr.com/methodology/random-assignment/,

https://engineering.rappi.com/bayesian-optimization-how-to-calibrate-the-hyper-parameters-of-computing-expensive-models-c3ca54b39dbb,

### Randomness is an essential tool in science

Almost every area of science makes use of randomness:

**Simulation:** Model what different scenarios could look like to understand a system.

**Estimation:** Calculate something you can't directly measure through random sampling and simulation

(Many of these are actually the same thing!)

https://f1000research.com/articles/9-1198, https://www.geeksforgeeks.org/bootstrap-method.



### (Pseudo)random Number Generation

TOUR OF ACCOUNTING OVER HERE WE HAVE OUR RANDOM NUMBER GENERATOR. OVER HERE WINE NINE SURE COMPACTOR 

- True random numbers only from random information (e.g., hardware random physical processes secrets module)
- Usually "pseudorandom" numbers using pseudorandom number generators (PRNGs).
- PRNGs DETERMINISTICALLY generate a series of random-appearing numbers based on a seed parameter
- Statistical tests to work out if random enough
- Linear Congruential Generator (LCG) one of simpler methods but has issues
- Numpy uses a Permutation Congruential Generator which does an additional permutation step

def lcg\_prng(seed, size):

```
state = seed
```

multiplier = 6364136223846793005 # big prime numbers

increment = 1442695040888963407 # big prime numbers

modulo = 2 \*\* 64 # normalise to integer

for i in range(size): # incrementally update seed/state

state = (multiplier \* state + increment) % modulo

#### yield state

list(lcg\_prng(42,2)) #[10481999410520546993,4159066171780167020]

list(lcg\_prng(42,2)) #[10481999410520546993,4159066171780167020]

list(lcg\_prng(10,2)) #[9743825058228238609,3298011952073619532]

list(lcg\_prng(10,4)) # [9743825058228238609,3298011952073619532, 658254279741938347,4015636148853147230]

### Reproducible research = setting a random seed

• For reproducibility: explicitly set this random seed (or python will choose a true random number as seed).



- Built-in but relatively slow
- Seed set: random.seed
- np.random
  - fast, optimised, numerical library
  - o np.random.default\_rng(seed)
  - np.random.seed(seed) # deprecated



### Numpy random PRNG functions

Create a PRGN generator with a specific seed (<u>ideal</u>: big positive integer, <u>practice</u>: 42...).

Could implement choice with integers

Random can be scaled to arbitrary ranges:

(high-low) \* random(0 to 1) + low

```
rng = np.random.default_rng(42)
rng.integers(low=0, high=8, size=3)
array([6, 2, 7])
rng.choice(['A', 'B', 'C'])
'B'
```

rng.uniform(low=3.0, high=7.0, size=2)
array([5.57546048, 6.29104645])

```
rng.random(size=2) # ==uniform: 0-1.0
array([0.37079802, 0.92676499])
```

```
(7-3) * rng.random(size=2) + 3 # scaling
array([5.21833915, 3.25526902]) #3->7
```

```
a = np.arange(4) # array([0,1,2,3])
rng.shuffle(a)
a
array([2, 3, 0, 1])
```

# Estimating hard to measure values with random numbers

### Monte Carlo Estimation/Simulation

- Odds of winning roulette always playing red?
- Analytical probability calculation:
  - p(win|red) = red\_slots / all\_slots
  - p(win|red) = 18 / (18 \* 2 + 2)
  - p(win|red) = 0.473
- What if the roulette wheel was hidden?
- What if you didn't know the rules of how to calculate probability directly?
- Alternative: just play red a lot and see how often you win or lose.
- More times you play the better an estimate of the true p(win|red)
- Developed formally by Stanislaw Ulam for the Manhattan Project



### Estimating $\pi$ with random samples

area of circle / total area =  $\pi/4$ 

- area ~= number of random samples inside
- $\pi$  = (points inside / total points) \* 4

More samples = more accurate estimate



Figure 2: Squares with zero, 250, 2500 and 25000 points.

def estimate\_pi(num\_points, rng):

```
# Generate random points in a 2x2 square centered at origin
```

```
points = rng.uniform(-1, 1, (num_points, 2))
```

# Calculate distance from origin for each point

```
distances = np.sum(points**2, axis=1)
```

# Count points inside the unit circle (distance < 1)</pre>

```
inside_circle = np.sum(distances <= 1)</pre>
```

```
# Pi estimate: (points in circle / total points) * 4
```

```
pi_estimate = (inside_circle / num_points) * 4
```

return pi\_estimate

estimate\_pi(1, rng) # 4.0

estimate\_pi(250, rng) # 3.152

estimate\_pi(1\_000\_000, rng) # 3.142028

### Sampling from probability distributions

### **Probability Distributions**

**Random Variable**: A variable whose value is determined by a random process. It can be:

- **Discrete**: Can only take specific values (like the number of heads in coin tosses) -
- **Continuous**: Can take any value within a range (like height or temperature)

**Probability Mass Function (PMF)**: For discrete random variables, assigns probabilities to each possible value.

#### Probability Density Function (PDF): For

continuous random variables, describes the relative likelihood of different values.

## **Cumulative Distribution Function (CDF)**: Gives the probability that a random variable is less than or equal to a certain value.



### Sampling from specific distributions

• **Normal/Gaussian**: sum of random variables tends to normal (central limit theorem), stay normal after many operations  $(x-\mu)^2$ 



· Binomial: binarv successes of independent trials



```
rng = np.random.default_rng(42)
rng.normal(loc=10, scale=4, size=3)
array([11.21886832, 5.84006358, 13.00180478]
```

rng.binomial(n=100, p=0.5) # 51
rng.binomial(n=100, p=0.5) # 56
rng.binomial(n=100, p=0.5) # 48

### Sampling from specific distributions

• Poisson: events occurring in a fixed interval



• Exponential: time between events



```
rng = np.random.default_rng(42)
rng.poisson(lam=4, size=3) #array([7, 3, 0])
```

```
rng.poisson(lam=4, size=3) # array([6, 4, 4])
```

rng.exponential(scale=2, size=4)

array([1.91320557, 1.24276617, 3.01592275, 4.05600316])

rng.exponential(scale=2, size=4)

array([0.66413763, 0.09950144, 1.84964447, 5.1068388 ])

### Many distributions!

np.random has ~36 built-in distributions

scipy.stats has > 120

Fun less-common distributions:

**Cauchy** - ratio of 2 normal random variables and has no mean or variance

**Gumbel** - used to calculate the probability of maximum values of random draws from other distributions



		100	
		100	Research and a second second
	Does untiples have a first discrimination	8	Entering and some
	Draw semples have a binamial	1000	
	disclosion.	100	
	Contraction of the second second	1005	
	Door complete from a site option	2	Arity and resident radia and the
	and the state of t	and a	Read of Arrival and Arrival
	Draw wavegles, from the Dechlet	100	
	dardenter.	and .	A real field of the state and the
		-	A Day concernation with
	Door complete from an experimental	-	
		Local .	1
	Dises samples have an F-shareholis a	100400	A Property of the Application of
	Property and the Rest of Party and	ing .	Real control of the second sec
	daugh dau	Links	Paintened where where and
		1000	Approximation of the second se
	Disas astrophy. It are the gramatice	1993.00	140000 PV-00000 PV-000
	Exclusion.	-	Approximation and a solid sector of the
		-	Effect approach a local action of the station of the state of the stat
	Draw samples have a Darstort	-	Approximation of the state of the state
	Restletter.	an other	Apartical grant action into anti-
ind rearying week	Dasa any des hiers a Pipergermann	and the second	Electric Construction and a second se
	disa de atori	1000	Elegen ) to add add on an odd with
		Television of the local division of the loca	
	Downsomphis frees the Laplace ar	Married Voters	Professional and the set
	daubh represented das but an with	1000	Parameter and the state
	specified incommentation and a set same	ine a	Ny salah
		-	A new process of the state of the
	Draw unreplete from a logation	lanat.	A new West of the series and
	distribution.	ALC: N	R Tel M Refer for a first set of the
		-	114-1 V - 11-11-11
und .	these are give it on a log worked	-	Figure Constantion and the second
		-	Free Species Antonio Sciences Sectors on other and the
	Diam antiphics have a laparithmic.	-	Compte Arrest in card of a data and
	series distribution.	basen.	Language and the stand of the growth matrix stations and the
		Dogo -	
(F)	Dean samples have a real-montal	benegen	Repaired and concerning one
	Kordedian.	in .	End Annulas, and an analy
oris holters essential	Description surfaces have a multipaciate	in an	Even who wanted a first state
	homogeneous des destant	lens.	Report of an adv with
		Longe and	Register with the same
in you, loss -1	Dates receivers samples from a	insist .	
	makes the second database.	<u> </u>	Final stars also and
. Train	Data searches have a spectrum	-	Mark States and a state of the state
	International structures	-	
		2	Reading and the state of the
nonel sand	Does assigns have a monormal doi.	8-	Real and Party in the state parts
	repara dia dan kalime	-	Dealers have shown and and
	Para and a barrier and a state	10.00	Party a dress and a well.
Tanks Termina and			
im, eurol, size 3	And Anger.	And a second	International and the state
ing moral words	dardene.	-	Kanan baya and a strain as the second Kanan second second sector and
ier, marač skotiji	diardentes Done racellare samples from a roomal	11.1	Nyana Nyana Santana akia akia Nyana akia akia akia Nyany akia kutoka akia akia Nyany akia kutoka akia
in, sund, dar f	Erne angen i sur de anne erne erne erne erne erne erne ern	11.1	A part i que a l'acteur en la cardin la part card cardina acteur acteur la finale de la parte i acteur acteur M'arge acteurs acteur acteur la desen de l'éga carde acteur la desen de la parte acteur
in, sond, song	enter anogar e ren de consenter e d'arrèction. Doar e coller samples fran a correl. Dar sour el debilisation. Dans amales han a Porte I UKA	hala	Egen light of the set
in und uns	Enter angeles are an esta and an esta de la constante esta de la constante sur españa finance a constante esta de la constante	111111111	Example of contrast and contrast. Reserved contrast and contrast and contrast. Marine and contrast and contrast and contrast. Marine for Representation and and and and and and and and and an
in und uns	Enter angeles Part de las antenios e de la datas. Done carellare samples from a correll Ele avante debathacien Done complex from a Partie EUAR Locate de databación de lo quedent		A see Aquest intervention and a select Selection and a selection and a Selection of a generative intervention Advances for the selection and Advances for the selection and Advances intervention Advances and an advances Advances and an advances Advances and a selection Advances and advances and advances Advances advances Advances Advances advanc
in wast work	Ende standarde er er er er annanderer Balandersten Desse vereinter særspilse frære a vormal Ellersvend skeldelsader Desse særspilse forer a Farrie FARA Lerenal skeldestadere refe sperefied skage	halalli.	Anni lynn diamenalaine Tarray antaramenalaine Mala tarlayna tarlay anna an adama Mala tarlayna tarlay anna an Mala tarlayna tarlay anna Mala tarlayna tarlay anna an Mala tarlay an
in, soul de J	Look anger or er of an annexes at the locals. Does receiver samply provide a roomal Discover of dealitations. Does angeles here a Pareta FLARA shape. Does samples here a Pareta		Am Syndroiten advantage Am Syndroiten advanta
in, soul_def	A solar sensitive samples from a management of the fractions are regular from a management December of destinations December of destinations December of the sensitive of the sensitive shape. December of the sensitive of the sensitive shape.		Amin Yanni Amin Anima Anima Miria Marian Anima Miria Marian Anima Maria Marian Maria Mar
in, unul_ung	Des segue ren de sanderer alakatusten Des ender samples from a normal Des sontet debilitation Des sontet debilitation des sontet de sanderer des sontet de sanderer des sontet de sanderer des sontet de sanderer de sontet de sanderer Des sontet de sanderer		Neniperate and an office office and an office of the second office of the An office of the second office of the An office of the second of the second of the An office of the second of the second of the An office of the second of the second of the An office of the second of the second of the second of the An office of the second of the second of the second of the second
in, soul_sing	Dece stepping with the international advantation. Dece method in perspace from external Decession of advantations Decession of advantations Decession of the international decession Deces ourselptic with a dvantation decession. Decession operation of the international decession of the international dvantation decession of the international dvantation		Non-Spannersberger           All-All-All-All-All-All-All-All-All-All
in, unal_ung	Deter stergingen sind von der Konstenden die Ankanzen Desse ereichen zwengele Anzei er sonsett Desse wengelen kennen ander gestelltet die space. Desse wengelen beite an Prinzers die die dasse. Desses wengelen ein (2, 1) fehren is periore die die dasse welft werden zu (2, 1) fehren is periore die die dasse welft gestelltet ergenerend. S.		Nenipersention of the Self
in, unu	Use UKP (in the second	Maria Maria	An Insurancia of Angel A
in, and, un S	Links strapping that has a functional distribution. Descent control of the second second proceeds of additional proceeds of additional process strapping to the set of the second distribution. Descent second second second second distributions Descent second second second second distributions. Descent second second second second distributions.		Amaramatana ang ang ang ang ang ang ang ang ang
in, and, un S	Index angular, the true is an antibility of effectives. None requires samply, there a reveal is a second of ability of the second constrained in the second second second constrained interfaces and the second constrained interfaces and the second constrained interfaces and the second constrained interfaces and the second constrained in the second second second second second second second second second second second second second second second secon		No. 1999           Amangan and and and and and and and and and a
in, and, un f	International and the second s		National State       State   <
in, and, un S	In the second se		Inclusion and any and any
in, end, un j	An extension of the intervention of the interv		An element and and an element and and an element and
er, end, welf	In a second seco		An Jone Series (Construction) An Annual (Co
er, depe, vertied, and	An example of the second secon		All values           Al
er, dege, welled, and ar, dege, welled, and	Anatom and an		An Jone and an and an
ος, φορο, καθος, καψ ος, φορο, καθος, καψ	Anatomic and an analysis of a second an		An element and an element and an element an
er, ang, unj er, depe, rating, anj er, depe, rating, anj er, depe, rating	Anderen and an anti- and an anti-anti-anti-anti-anti- land and anti-anti-anti-anti- land anti-anti-anti-anti- anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti- anti-anti-anti-anti-anti-anti-anti- anti-anti-anti-anti-anti-anti-anti-anti-		All values and all
ur, dage, entrol ( and ar, dage, entrol ( and ar, dage, and ar, dage, and ar, dage, and	Extension and an extension and and and and and and and and and an		An June 2019 An
or, days, welled, and in days, welled, and graved	Extension of the sector of the		An June 2019 An
ος, άρχη, τολος, άκξ αν, άρχη, τολος, άκξο αν, άρχη, ποζ	Ander Angebacker		An example of a part of
en here (der) en here verten (der) en here verten (der) en here verten (der)	Handson and an and an		An energy of the sector of the
or, hope, earlied, and are, hope, earlied, and are, hope, earlied	Externational Sector Se		An unit of the sector of the s
n, nord, darij ne dogo, netkol, neđ ne dogo, netkol, neđ ne dogo, neđ neda	Analas and an		Annual and a second a
ος, φορς, καθ ος, φορς, καθος, κοβ ος, φορς, καθ	Handback Bandba		Acutoria de la companya de la compan
en dege perfect and an dege perfect and an dege med an dege med an dege med	Ander and a set of the		Balance         Balance           Balance         <
or, hang, well, darj ar, daga, welland, darj ar, daga, melj apa, melj	Analas an		Balance         Balance <t< th=""></t<>
ος δους αυχός τουδος τουδος αυχόρος τουδος αυχόρος τουδος αυχόρος τουδος αυχόρος αυχό αυχό αυχό αυχό αυχό αυχό αυχό αυχό	Ander and an and an		Balance         Balance           Balance         <
ος, δους ος, δημη, καθος αυζι ος, δημη, καθος ος δους ος, δους ος ος ος ος ος ος ος ος ος ος ος ος ος	Ander and an and an		Anne and a second a s
هم محمد ( هدر ) مع هوی بعد العرب ( هدر ) مع هوی بعد العرب ( هدر ) مع هوی بعد ( مع محمد ) مع هوی العرب ( مع محمد ) مع محمد )	Analos a a a a a a a a a a a a a a a a a a a		Control         Control <t< th=""></t<>
ου, πους (κατβ ου, πορος, τουθική, ποβ ου, πορος, τουθική, ποβ ου, πομος, πουβ ους ους ους ους ους ους ους ους	Enderson and an and an		Actions of the section of the sectio
un, same ( dar) un, shape, and and an, shape, and a gap shaft and and and and and and and and and and	Ander and an and an		Balance         Balance <t< th=""></t<>
en, handi, dan ji an, daga, tarihi di, andi an, daga, tarihi di, andi an, daga, andi ang dagaji	Anarres and a series of a seri		Action of the sector of the se
en, merel (and) en, deger, walket, socio en, deger, walket, socio en, deger, met ger g	Handback Answingenes bestel Sensemptice best and sensemptice Sensemptice best and sensemptice Sensemptic		Control of the sector
en, handi dariji an, hapen, endenet, kadj an, hapen, end gen, nadj gen	Ander and an and an		Control         Control <t< th=""></t<>
هم ( هدی ( هدی ( هم ( هدی ( هم ( هدی ( هم ( هدی ( هم ( هدی ( ) هم ( هدی ( ) ) ) ) ) ) ) ) ) ) ) ) )	Ander and an and an		Control         Control <t< th=""></t<>
ος, δους ( do 1 ) ος, δορο, κοθος ( do 2 ) ος, δορο, κοθος ος, δορο, κοθ ορο, καθ φ	Ander and an and an		Contention         Contention           Contention         Contention <t< th=""></t<>
ος, δους ζους β ος, δημης, αναθυτοίς διαξί ος, δημης, αναθ δηξι αικαθ β	Ander and an		Control         Control <t< th=""></t<>

Andrew Andr

Distributions

### Understanding outcomes by sampling from a distribution

Let's say we developed a vaccine

Clinical Trial showed that the vaccine prevented 70% of people exposed from getting infected.

If we gave this vaccine to 1000 people in the community and they were each exposed to the disease:

- How many of them would we expect to be protected?
- How variable would this be?

Can calculate this by randomly sampling from a binomial distribution!

```
vaccine_protection = 0.7 # 70%
num people = 1000 # Population size
num simulations = 50 # Number of times to run the experiment
protected counts = rng.binomial(n=num people,
                                p=vaccine effectiveness.
                                size=num simulations)
median protected = np.median(protected counts)
std protected = np.std(protected counts)
print(f"{median protected} +/- {std protected}")
693.5 +/- 16.569
```

### Optimisation

### Optimising functions is common in science & engineering

Scientific problems often involve finding the most efficient path, optimal configuration, or best parameters:

- Reducing drag in the design of a new boat propellor
- Finding the best evolutionary tree to explain DNA data
- Working out how to allocate vaccines to maximum effect

Optimisation without randomness means you can get stuck in local maxima/minima!



### **Functional optimisation**

**Problem**: finding x and y with that maximise the value of a function (often unobservable directly)



def multi\_peak\_function(x, y):
 peak1 = 3 \* np.exp(-0.2 \* ((x - 2)\*\*2 + (y - 2)\*\*2))
 peak2 = 2 \* np.exp(-0.3 \* ((x + 2)\*\*2 + (y + 2)\*\*2))
 peak3 = 1.8 \* np.exp(-0.5 \* ((x - 2)\*\*2 + (y + 1)\*\*2))
 peak4 = 1.5 \* np.exp(-0.5 \* ((x + 1)\*\*2 + (y - 3)\*\*2))
 peak5 = 1.2 \* np.exp(-0.7 \* ((x - 0)\*\*2 + (y - 0)\*\*2))
 # Combine all peaks

return peak1 + peak2 + peak3 + peak4 + peak5



### **Grid-Search**

**Problem**: finding x and y with that maximise the value of a function (often unobservable directly)

Solution 1: try every possible x and y

- Infinite number of possibilities

**Solution 2**: systematically search x and y

- Many combinations but should be close



```
num_points = 25
```

(2.0833, 2.0833) = 3.0185

```
x grid = np.linspace(-5, 5, num points)
y grid = np.linspace(-5, 5, num points)
max value = float('-inf')
max\_coords = (0, 0)
all points = []
# Evaluate function at each grid point
for x in x grid:
      for y in y grid:
            z = multi_peak_function(x, y)
            all points.append((x, y, z))
            if z > max value:
                max value, max coords = z, (x, y)
all_points.sort(key=lambda point: point[2], reverse=True)
print(all_points[0])
```

### **Random Search**

**Problem**: finding x and y with that maximise the value of a function (often unobservable directly)

Solution 3: randomly try a number of possible values

- Much more efficient exploration of possible parameter space
- Not guaranteed to find optimal value



#### num points = 200x\_random = np.random.uniform(-5, 5, num\_points) y random = np.random.uniform(-5, 5, num points) Max value, max coords, all points = float('-inf'), (0,0), [] # Evaluate function at each random point for i in range(num points): x, y = x\_random[i], y\_random[i] z = multi peak function(x, y)all points.append((x, y, z)) if z > max value: max value = zmax coords = (x, y)all points.sort(key=lambda point: point[2], reverse=True) print(all points[0])

(2.03, 2.04) = 3.0299

### Simulation

### Detecting the Higgs Boson with simulations!

- Smashing bundles of protons together creates many different decay paths.
- CMS/Atlas Detectors can detect signals related to different decays
- How do you know when you've detected the signal of the Higgs Boson?
- Simulate every possible decay path and combination of decay paths with and without Higgs Boson predicted signal
- Compare simulated data to collected data!





# Simulations as key part of frequentist statistics

### Evaluating if treatments work

You're an alpaca shepherd and want to work out if a new shampoos increases the wool quality of your alpaca.

Our experiment will have the following hypotheses:

 $H_{_0}: \mu_{_{treatment}} <= \mu_{_{control}}$ 

Null: shampoo doesn't change average wool quality.

 $H_A: \mu_{treatment} > \mu_{control}$ 

Alternative: new shampoo yields superior wool quality



### Evaluating if treatments work

We can randomly assign our alpacas to:

- Treatment group who gets the new shampoo
- Control group that keeps the old shampoo

Then we measure the wool quality in each alpaca and calculate the average for each group.

We can then calculate the difference in mean wool quality for each group to give us a test statistic:

*Test Statistic* =  $\mu_{\text{Treatment}}$  -  $\mu_{\text{Control}}$ 

*Test Statistic* = 1.6



treat = np.array([4.8, 7.2, 6.2, ...])
control = np.array([5.4, 4.4, 5.1, ... ])

test\_stat = np.mean(treat) - np.mean(control)

### Determining whether a test statistic is significant

Need to know what the DISTRIBUTION of test statistics would look like if null were true

Can estimate this with simulation by shuffling the alpacas between groups and recalculating our test-statistic.

Shuffled test statistic = -0.5



### Determining whether a test statistic is significant

Need to know what the DISTRIBUTION of test statistics would look like if null were true

Can estimate this with simulation by shuffling the alpacas between groups and recalculating our test-statistic.

Shuffled test statistic = -0.5

Repeat many times = test distribution



### Determining whether a test statistic is significant

Need to know what the DISTRIBUTION of test statistics would look like if null were true

Can estimate this with simulation by shuffling the alpacas between groups and recalculating our test-statistic.

Shuffled test statistic = -0.5

Repeat many times = test distribution



```
n permutations = 200
combined = np.concatenate([treat, control])
n1, n2 = len(treat), len(control)
observed diff = np.mean(treat) - np.mean(control)
null distribution = np.zeros(n permutations)
for i in range(n permutations):
       np.random.shuffle(combined)
       new treat = combined[:n1]
       new control = combined[n1:n1+n2]
       null_distribution[i] = np.mean(new_tream) - np.mean(control)
```

### Compare observed test statistics to null test-distribution

p-value is just the proportion of permutations where the sampled difference was greater than or equal to the difference we observed in our original treatment and control groups.

If we did 200 permutations and 16 had a difference >= to our observed test statistic

p-value = 16 / 200 = 0.08 or 8%



p\_value = np.mean(np.abs(null\_distribution) >= np.abs(test\_statistic)

Frequent statistics really just boils down to this procedure!



You can memorise rules/common practices to do this analytically or more efficiently but essentially you are always comparing a test statistics to the distribution of that statistic under a null

#### Calculating confidence intervals by random re-sampling

Let's say we are measuring the average growth in a set of 15 plants.

Can calculate the mean easily but how do we know what the plausible range of values would be.

A **confidence interval** gives a range of plausible values for an unknown population parameter (like a mean, proportion, or median)

**Definition:** a 95% confidence interval of [10.2, 12.8] means if you were to repeat your sampling process many times ~95% of the resulting intervals would contain the true population parameter.

# We'll make up some data but in the real-world we'd measure this in an experiment.

```
n_plants = 15
```

```
true_mean_growth = 3.25
```

```
growth_data = rng.normal(loc=true_mean_growth,
```

```
scale=2.5, size=n_plants)
```

```
# Calculate the sample mean
```

```
sample_mean = np.mean(growth_data)
```

### Bootstrapping: sampling original data with replacement



$\operatorname{outstrap} = 1000$	
tstrap_means = []	
_ in range(n_bootstrap):	

1000

n b

boo

for

# Resample with replacement

bootstrap\_sample = np.random.choice(growth\_data,

size=len(growth\_data),

replace=True)

bootstrap\_means.append(np.mean(bootstrap\_sample))

bootstrap\_means = np.array(bootstrap\_means)

### Bootstrapping: sampling original data with replacement





lower\_ci = np.percentile(bootstrap\_means, 2.5)

upper\_ci = np.percentile(bootstrap\_means, 97.5)

### Summary

- Numpy basics and gotcha
- Randomness
- Pseudorandom Number Generators
- Numpy PRNG functions
  - Using PRNGs to estimate values (Monte Carlo)
- Probability Distributions
  - Estimating outcomes by sampling from distributions
- Optimisation
  - Grid-search vs Random-search
- Simulation
  - Calculating p-values by simulating Null distribution
  - Bootstrapping to estimate confidence intervals