4. Probability II - random walks

Last time

- Random numbers
- Frequencies and relative frequencies
- Random variables
- Probability distribution

Where are we heading?

- We have introduced **stochastic** (random) models
- We have a stochastic model of recovery

Where are we heading?

- We have introduced **stochastic** (random) models
- We have a stochastic model of recovery

- Goal: Model individual people (**agents**) moving in space

Where are we heading?

- We have introduced **stochastic** (random) models
- We have a stochastic model of recovery

- Goal: Model individual people (**agents**) moving in space
- When they are in **contact** they can transmit infection

Where are we heading?

- We have introduced **stochastic** (random) models
- We have a stochastic model of recovery

- Goal: Model individual people (**agents**) moving in space
- When they are in **contact** they can transmit infection
- And they can recover

Goals for today

- Random motion: Random walks
- Characterising variability

Random walks

Brownian motion

- Watch a particle in water under microscope.
- Follows a random path: **Brownian motion**.
- Fundamental dynamical process in many domains:
  - Biology – protein inside cell
  - Chemistry – reactant in
  - Physics – particle in fluid
  - Engineering – jet noise
  - Economics – stock price
  - Environmental sciences – pollutant spreading out
  - Mathematics – fundamental random process

Modelling random motion: Random walk

- Expensive to simulate collisions of many particles
- Instead, **directly simulate** random kicks using random numbers

Modelling random motion: Random walk

- Expensive to simulate collisions of many particles
- Instead, **directly simulate** random kicks using random numbers

- Simplest model: **simple random walk**:
  - 1 particle moving on integers in 1D
  - Jumps left or right, e.g. with equal probability

Modelling random motion: Random walk

- Expensive to simulate collisions of many particles
- Instead, **directly simulate** random kicks using random numbers

- Simplest model: **simple random walk**:
  - 1 particle moving on integers in 1D
  - Jumps left or right, e.g. with equal probability

- How can we generate jumps ± 1 with uniform probability?

Generating random jumps

- One solution: `julia    jump() = rand( (-1, +1) )`

Generating random jumps

- One solution: `julia   jump() = rand( (-1, +1) )`
- Another solution: generate random Boolean (`true` / `false`) and convert:

```
r = rand(Bool)
Int(r)    # convert to integer
```

Generating random jumps

- One solution: `julia   jump() = rand( (-1, +1) )`
- Another solution: generate random Boolean (`true` / `false`) and convert:

```
r = rand(Bool)
Int(r)   # convert to integer
```

- How convert this to ± 1? Which is faster?

Simple random walk

- Random **walk**: *sequence* of random *jumps*!

Simple random walk

- Random **walk**: *sequence* of random *jumps*!
- Don't use global scope; immediately *make a function*:

```julia
function walk(N)
    x = 0       # initial position
    positions = [x]  # store the positions

    for i in 1:N
        x += jump()
        push!(positions, x)
    end

    return positions
end
```

Interactive animation of walker position

- First instinct by now: Plot data and make it interactive

Interactive animation of walker position

- First instinct by now: Plot data and make it interactive
- Key trick:
- *Pre-generate* data so don't have different randomness each time:

```
using Interact

N = 100
positions = walk(N)

@manipulate for n in 1:N
    plot(positions[1:n], xlim=(0, N), ylim=(-20, 20), m=:o,
end
```

Shape of random walk

- Plot several walks in single figure using `for`
- Since `for` returns `nothing`, evaluate graph to plot:

```
p = plot(leg=false)  # empty plot
N = 100

for i in 1:10    # number of walks
    plot!(walk(N))
end

p  # or plot!()
```

- **Exercise**: Animate position of several walkers simultaneously

Distribution of walker position

- Fix a time n, e.g. $n = 10$ and think about X_n
- Can ask same questions as before:
  - What is mean position $\langle X_n \rangle$?
  - What is spread of X_n?
  - What does probability distribution of X_n look like?

Random processes

- Notation:
  - Steps $S_i = \pm 1$
  - Position X_n at step n

Random processes

- Notation:
  - Steps $S_i = \pm 1$
  - Position X_n at step n
- $X_n = S_1 + S_2 + \cdots + S_n = \sum_{i=1}^{n} S_i$
- S_i are random variables; X_n is also random variable

Random processes

- Notation:
  - Steps $S_i = \pm 1$
  - Position X_n at step n
- $X_n = S_1 + S_2 + \cdots + S_n = \sum_{i=1}^{n} S_i$
- S_i are random variables; X_n is also random variable
- Collection $(X_n)_{n=1}^{N}$ is **random process** – i.e. random variable at each time

Dynamics of random process

- Questions:
  - What is dynamics *as a function of time*?

Dynamics of random process

- Questions:
  - What is dynamics *as a function of time*?
  - How does mean position change *as function of time*?
  - How does spread change *as function of time*?

Dynamics of random process

- Questions:
  - What is dynamics *as a function of time*?
  - How does mean position change *as function of time*?
  - How does spread change *as function of time*?
  - Number of sites visited up to time n
  - First time to reach certain position

Dynamics of random process

- Questions:
  - What is dynamics *as a function of time*?
  - How does mean position change *as function of time*?
  - How does spread change *as function of time*?
  - Number of sites visited up to time n
  - First time to reach certain position
- Last two questions cannot be answered by looking at single time n

Probability distribution of X_n

- X_n is **discrete random variable**
- Run "cloud" ("ensemble") of **independent** walkers, i.e. *don't interact with one another*
- To generate data, could use `walk(N)`, but only need final position:

```
jump() = rand( (-1, +1) )

walk_position(N) = sum(jump() for i in 1:N)
```

- Faster to generate all random numbers at once: `julia walk_position2(N) = sum(rand((-1, +1), N))`

Variability

- We have **finite sample** from ideal **population**
- If we repeat experiment, get different sample with different counts.
- How characterize this *variability*?

Shape of distribution

- See that position “clusters around” central value: **expected value**
- Values near extremes “never” occur

Shape of distribution

- See that position “clusters around” central value: **expected value**
- Values near extremes “never” occur
- Characterise using **summary statistics**: numbers that summarise aspects of **distribution**
- Simplest: **sample mean** = average value

Shape of distribution

- See that position “clusters around” central value: **expected value**
- Values near extremes “never” occur
- Characterise using **summary statistics**: numbers that summarise aspects of **distribution**
- Simplest: **sample mean** = average value

- Shape is “bell curve”: **Gaussian** or **normal** distribution

Mean

- Given outcomes x_i for $i = 1, \ldots, N$, (arithmetic) mean is

$$\bar{x} := \frac{1}{N} \sum_{i=1}^{N} x_i$$

- Calculate in Julia:

```
mean(data) = sum(data) / length(data)


m = sum(n1_data) / length(n1_data)
```

- NB: `mean` is in `Statistics` standard library (no need to install)
- Add to plot using `vline!([m])`

Centre the data

- Distribution “spreads out” from mean

Centre the data

- Distribution “spreads out” from mean
- Want to know **how far** it spreads

Centre the data

- Distribution “spreads out” from mean
- Want to know **how far** it spreads

,

- **Centre** data by subtracting mean:

```
centered_data = data .- m
```

Spread

- Measure spread as ”average *distance from mean”

Spread

- Measure spread as ”average *distance from mean”
- If just take `mean` of new data, get tiny result near 0:

```
mean(data)
```

- (`1e-14` means 1×10^{-14}, i.e. a value that is effectively 0.)

Spread

- Measure spread as ”average *distance from mean”
- If just take `mean` of new data, get tiny result near 0:

```
mean(data)
```

- (`1e-14` means 1×10^{-14}, i.e. a value that is effectively 0.)
- Why? Negative values *cancel out* positive values.
- Need to *avoid cancellation*. How?

Spread II

- Options to avoid cancellation of displacements from mean:
  - take **absolute value**
  - take **square**:

Spread II

- Options to avoid cancellation of displacements from mean:
    - take **absolute value**
    - take **square**:

```
spread = mean(abs.(centered_data))    # no standard name?

variance = mean(centered_data .^ 2)
σ = √variance

@show spread, variance
```

Spread III

- (Sample) **variance** defined by squaring
- So must take $\sqrt{}$ for “correct units” (metres vs. metres^2)

Spread III

- (Sample) **variance** defined by squaring
- So must take $\sqrt{}$ for “correct units” (metres vs. metres^2)

- σ is called **standard deviation**
- For this distribution, both measures of spread give similar result

Spread III

- Most data is in interval $[\mu - 2\sigma, \mu + 2\sigma]$
- How much? Calculate!:

```
count(-2σ .< centered_data .< 2σ) / length(centered_data)
```

Spread III

- Most data is in interval $[\mu - 2\sigma, \mu + 2\sigma]$
- How much? Calculate!:

```
count(-2σ .< centered_data .< 2σ) / length(centered_data)
```

- Approx. 95%: “universal” in many (but *not all* situations)

Summary

- Random walks: Simple model of random motion in space

Summary

- Random walks: Simple model of random motion in space
- Characterise variability using **mean** and **variance** or **standard deviation**

Summary

- Random walks: Simple model of random motion in space
- Characterise variability using **mean** and **variance** or **standard deviation**

- Most data within 2 standard deviations of mean in common distributions that are result of adding many effects up