GOV 2001/ 1002/ E-2001 Section 2[1]
Stochastic Components of Statistical Models

Mayya Komisarchik

Harvard University

February 3, 2016

[1] Heartfelt thanks to all of the Gov 2001 TFs of yesteryear; this section draws heavily upon materials created by Stephen Pettigrew and Solé Prillaman.

Outline

Replication Paper

- Read *Publication, Publication* in Perusall.
- Find a partner to coauthor with.
- Find a set of papers you would be interested in replicating.

  1. Recently published (in the last two years).
  2. From a top journal in your field.
     - Think *American Political Science Review* or *American Economic Review*, not the Nordic Council for Reindeer Husbandry Research's *Rangifer*. Unless your field is reindeer husbandry.
  3. Use methods at least as sophisticated as in this class (more than just basic OLS).

Replication Paper

- Have a classmate (somebody in your study group, preferably) approve your choice of article. They should make sure it fits the criteria listed in *Publication, Publication*.
- Each set of partners should email us (Gary, Anton, and Mayya) with a PDF of your article, a brief (2-5 sentence) explanation for why you picked it. And the name of the person who checked that it met the requirements in *Publication, Publication*.
- Begin to find the data. Some journals require authors to submit their data to the journal. Some authors you'll have to email directly for the data.

Canvas

- Everybody who is:
  1. Registered for the class through Harvard
  2. Cross-registered from MIT or a different Harvard department
  3. Registered through the Extension School
  4. Formally auditing the course through the Extension School

  should have access to Canvas by now.

- Let us know if you fall into one of these categories, or a category that isn't on this list, and still need access to Canvas!

This week's homework

- By later tonight there will be two assignments on the Quizzes section on Canvas
  1. The second problem set, which you'll complete exactly as you did last week
  2. An assessment problem, which you must complete **independently** and can only submit one time to Canvas
- Your TFs cannot help you with the assessment problem. If you have a clarifying question, email all three of us and we'll post an answer on Canvas if we think we need to clarify something about the question.

OUTLINE

What are statistical models and how do we use them?

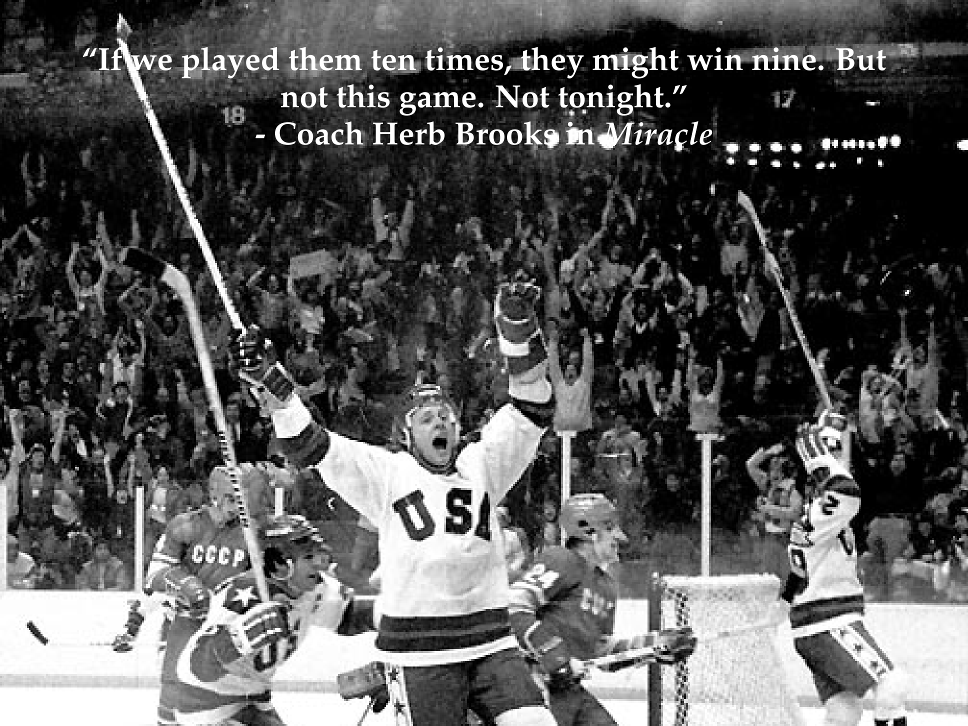
Statistical models are mathematical representations of stochastic processes in the real world.

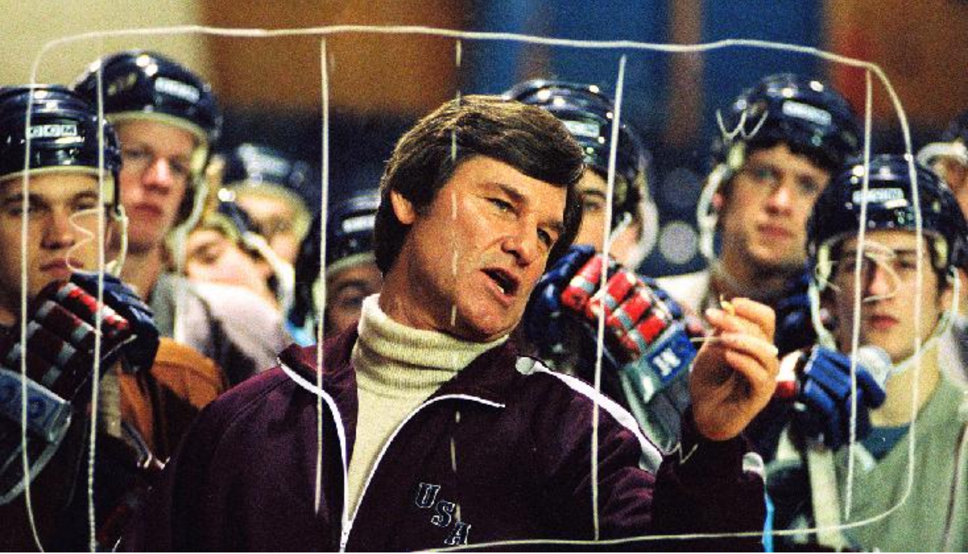
To build a statistical model, we need to know two things about the real-world process we're trying to represent:

1. What is the dependent variable?

2. What was the data generation process that created that dependent variable?

Our goal is ultimately to match our data generation process to a statistical model that behaves in a broadly similar way.


“If we played them ten times, they might win nine. But not this game. Not tonight.”
- Coach Herb Brooks in *Miracle*
18
17
USA
CCCP



"If Y is whether the Soviet team beats us, then $Y \sim Bern(0.9)$. But $y_{thisgame} \neq 1$. $y_{tonight} = 0$."

- Coach Herb Brooks in *Miracle*

"P(Miracle) =

$(0.1)^1 \cdot (1-0.1)^{(1-1)}$.

9 to 1 were good odds for Mother Russia"

What do we need to know about probability distributions in order to use them?

There are several things we need to know about probability distributions in order to use them:

- We need to know their general properties (i.e. how to apply the axioms of probability)
- We need to know the data generation process ("DGP") each probability distribution is designed to represent
- We need to know the specific properties of our probability distributions (i.e. support, parameters, etc.)
- We need to understand the assumptions we're making by picking a particular distribution (e.g. what is constant? what is variable?)

Outline

The Bernoulli Distribution

- **Notation:** $Y \sim \text{Bernoulli}(\pi)$
- **Outcome Variable:** The Bernoulli distribution models *binary* outcome variables. That is, your outcome variable Y can *only* take on one of two mutually exclusive values.
  - Coin flip
  - Vote choice in an election between two candidates
  - Outcome of a sporting event where no ties are possible
- **Notes and Assumptions:** Note that the Bernoulli distribution does *not* model repeated events! **One-time** binary outcomes are distributed Bernoulli

The Bernoulli Distribution

- **PMF:** $P(Y = y \mid \pi) = \pi^y (1 - \pi)^{1-y}$
- **Parameters:**
  - π : The probability that $Y = 1$ (probability of "success")
    - $\pi \in [0, 1]$
- **Support:** $Y \in \{0, 1\}$
- **E(Y):** π
- **Var(Y):** $\pi(1 - \pi)$
- **Drawing from a Bernoulli in R:**

```
rbinom(100, size = 1, prob = π)
```

The Binomial Distribution

- **Notation:** $Y \sim \text{Binomial}(n, \pi)$
- **Outcome Variable:** The binomial distribution models the total number of successes in n Bernoulli trials. So here, we *are* repeating a binary process many times.
  - Number of heads in 10 coin flips
  - Number of rainy days in a week
- **Notes and Assumptions:** n is a fixed number of trials, you don't generally estimate this.

The Binomial Distribution

- **PMF:** $P(Y = y \mid \pi, n) = \binom{n}{y}\pi^y(1-\pi)^{1-y}$
  - $\binom{n}{y} = \frac{n!}{y!(n-y)!}$
- **Parameters:**
  - π : The probability that $Y = 1$ (probability of "success")
    - $\pi \in [0, 1]$
  - n trials
    - $n =$ integers $\in [1, \infty)$
- **Support:** $Y \in [0, \infty)$
- **E(Y):** $n\pi$
- **Var(Y):** $n\pi(1-\pi)$
- **Drawing from a Bernoulli in R:**
  `rbinom(100, size =` n`, prob =` π`)`

The Multinomial Distribution

- **Notation:** $Y \sim \text{Multinomial}(n, \pi_1, \pi_2 ... \pi_k)$
- **Outcome Variable:** This is a generalization of the binomial. Instead of counting the number of successes in a single category, we are now counting the number of successes in each of k mutually exclusive categories.
  - Number of people who walk into a gelato shop and select chocolate, vanilla, hazelnut, etc. from each of k available flavors
  - Number of people who vote for each of the remaining Republican primary candidates
- **Notes and Assumptions:** n is *still* a fixed number of trials, so don't estimate it unless you like to get weird. Note also that this returns a vector of counts, $\boldsymbol{Y}$, that tells you the number of successes in each category.

The Multinomial Distribution

- **PMF:** $P(Y_1 ... Y_k = y_1 ... y_k \mid \pi_1 ... \pi_k, n) = \frac{n!}{y_1! y_2! ... y_k!} \pi_1^{y_1} \pi_2^{y_2} \ldots \pi_k^{y_k}$
- **Parameters:**
  - $\boldsymbol{\pi}$: This is the vector of probabilities $(\pi_1, \pi_2 ... \pi_k)$, in which each element represents the probability that $y_k = 1$, or the probability of a success in category k.
    - $\pi_j \in [0, 1]$
    - $\sum_{j=1}^{k} \pi_j = 1$
  - n trials
    - $n =$ integers $\in [1, \infty)$
- **Support:** $y_j \in [0, n]$
  - $\sum_{j=1}^{k} y_j = n$
- **E(Y):** $n\pi_j$
- **Var(Y):** $n\pi_j(1 - \pi_j)$
- **Drawing from a Multinomial in R:**

```
rmultinom(100, size = n, prob =
c(π1, π2, ...πk))
```

Poisson Distribution

What is this?

Poisson Distribution

It's a Prussian soldier getting kicked in the head by a horse!

Poisson Distribution

It's also the logo for Stata Press.

The Poisson Distribution

- **Notation:** $Y \sim \text{Poisson}(\lambda)$
- **Outcome Variable:** The Poisson distribution models counts of rare events that occur in a fixed time or space.
  - Number of raindrops that hit a 1″ x 1″ square on the sidewalk during a rainstorm
  - Number of executive orders that a president issues in a week
  - Number of Prussian soldiers who died each year by being kicked in the head by a horse (Bortkiewicz, 1898).
- **Notes and Assumptions:**
  - If we are using a Poisson, we're assuming that the number of trials is very large (think total number of raindrops that fall in a storm)
  - We also assume that the probability of success (hitting the 1″ square) for any individual trial is tiny.

The Poisson Distribution

- **PMF:** $P(Y = y \mid \lambda) = \frac{e^{-\lambda}\lambda^{y}}{y!}$
- **Parameters:**
  - λ is the rate at which successes occur
    - $\lambda > 0$
- **Support:** $Y \in [0, \infty)$
- **E(Y):** λ
- **Var(Y):** λ
- **Drawing from a Poisson in R:**
  `rpois(100, lambda =` λ`)`

Outline

The Univariate Normal Distribution

- **Notation:** $Y \sim \mathrm{N}(\mu, \sigma^2)$
- **Outcome Variable:** The normal distribution models outcome variables that can take on any real number as a value and cluster in a bell curve around the mean.
  - The heights of male students in this room
  - SAT/GRE scores
- **Notes and Assumptions:** Remember that it's difficult to think of data that is *truly* unbounded, so be careful if you're using the normal distribution to predict something like vote shares; you'll often end up predicting that a candidate gets $> 100\%$ of the vote (which, if you're studying Putin, that's probably fair, I'm fine with it. Otherwise you need to re-think your model).

THE UNIVARIATE NORMAL DISTRIBUTION

- **PDF:** $P(Y = y \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\mu)^2}{2\sigma^2}}$
- **Parameters:**
  - μ is the mean of the univariate normal distribution
    - This distribution is symmetric about the mean
    - $\mu \in \mathbb{R}$
  - σ^2 is the variance of the univariate normal distribution
    - σ^2 dictates the spread if your normal bell curve
    - $\sigma^2 > 0$
- **Support:** $Y \in \mathbb{R}$
- **E(Y):** μ
- **Var(Y):** σ^2
- **Drawing from a Univariate Normal in R:**
  `rnorm(100, mean =` μ`, sd =` $\sqrt{\sigma^2}$`)`

The Uniform Distribution

- **Notation:** $Y \sim \text{Uniform}(\alpha, \beta)$
- **Outcome Variable:** The uniform distribution models outcome variables that are equally likely to take on any real value in some defined interval.
  - The degree of longitude you're pointing to if you stop a spinning globe with your finger.
  - In the discrete version of the uniform distribution, this could be the bib number of the runner who comes in first in a race
- **Notes and Assumptions:** It's a box.

The Uniform Distribution

- **PDF:** $P(Y = y \mid \alpha, \beta) = \frac{1}{\beta - \alpha}$
- **Parameters:**
  - α The earliest point in your interval for y
    - $\alpha \in \mathbb{R}$
  - β is the last point in your interval for y
    - $\beta \in \mathbb{R}$
    - $\beta > \alpha$
- **Support:** $Y \in [\alpha, \beta]$
- **E(Y):** $\frac{\alpha+\beta}{2}$
- **Var(Y):** $\frac{(\beta-\alpha)^2}{12}$
- **Drawing from a Uniform in R:**
  `runif(100, min = α, max = β)`

The Exponential Distribution

- **Notation:** $Y \sim \text{Expo}(\lambda)$
- **Outcome Variable:** The exponential distribution models wait times between events.
    - How long you have to wait for the bus
    - Time between terrorist attacks
- **Notes and Assumptions:** The exponential distribution is "memoryless"
    - That is, your expected wait time between events is constant regardless of how much time has actually elapsed since the last event occurred.
    - e.g. $E(Y) = E(Y \mid y > 1) = E(Y \mid y > 10)$, etc.

The Exponential Distribution

- **PDF:** $P(Y = y \mid \lambda) = \lambda e^{-\lambda y}$
- **Parameters:**
  - λ is the rate at which events occur; you'll hear this called the "arrival rate"
    - $\lambda > 0$
- **Support:** $Y \in [0, \infty)$
- **E(Y):** $\frac{1}{\lambda}$
- **Var(Y):** $\frac{1}{\lambda^2}$
- **Drawing from a Uniform in R:**

  `rexp(100, rate = λ)`

Outline

Writing PMF and PDF Functions in R

- Why do I even need to learn how to do this?
- MAJOR SPOILER:
  - We're working up to maximum likelihood functions.
  - While you can often use canned functions like `rnorm()` and `rpois()` when you're just working with the normal PDF and Poisson PMF, respectively, you'll quickly see when we start programming log likelihood functions that your functions will not match any known distribution.
  - So we'll learn to code these functions up from scratch by going through an example with the univariate normal PDF

Normal PDF Function Example

- First, recall the normal PDF:
  - $P(Y = y \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\mu)^2}{2\sigma^2}}$
- Our first task is to create a function in R that takes the arguments y, μ, and σ
  - `normal <- function (y, mu, sigma){exp(-(y - mu)^ 2 / (2 * sigma^ 2)) / (sigma * sqrt(2 * pi))}`
- Now let's check that this is a valid PDF.

Normal PDF Function Example

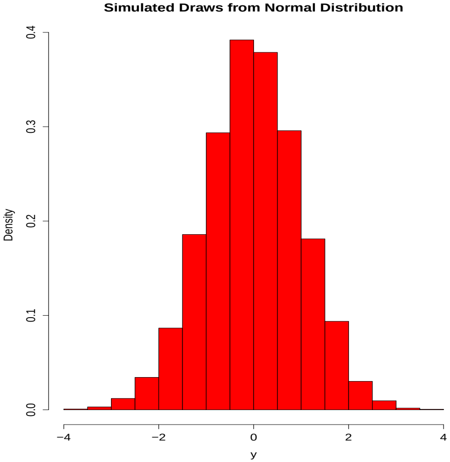
- We know that, by definition, a PDF must integrate to 1 to be a valid PDF.
- So let's integrate our normal PDF function in R!
  - `integrate(normal, lower = -1000, upper = 1000, mu = 0, #put any extra arguments needed for your function here sigma = 3)`
  - **Output:** `1 with absolute error < 9e-05`
- Note that, even though the support of Y in a normal distribution is $(-\infty, \infty)$, we don't need to integrate from negative infinity to infinity. We can just choose large numbers (though you can represent the quantity ∞ or $-\infty$ in R by typing `Inf` or `-Inf`).

Normal PDF Function Example

- Suppose we want to simulate from our PDF function and plot the results. How do we do that?
  - First, let's generate a large number of potential values for Y (keep in mind you should only generate values in the support of Y. Try to make increments between values smaller for continuous distributions).
    - `values <- seq(-10, 10, 0.001)`
  - Now let's use our function to generate some values for $P(Y = y \mid \mu, \sigma)$
    - `weights <- normal(values, mu = 0, sigma = 1)`
  - Now we can draw values from our vector of values, where the probability of drawing any given value y is weighted by the densities.
    - `draws <- sample(values, size = 10000 prob = weights, replace = T)`

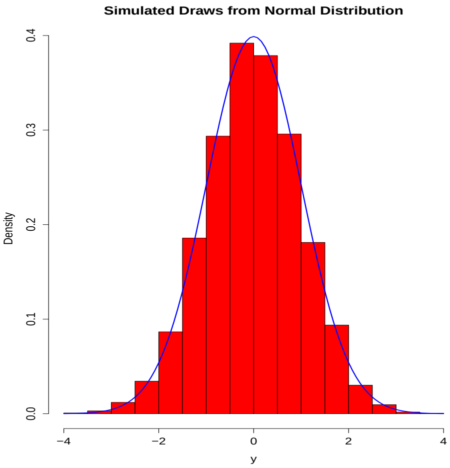
NORMAL PDF FUNCTION EXAMPLE

- Now we can plot our PDF!
  - ```
    hist(draws,
    xlim = c(-4,4),
    prob = T,
    xlab = "y",
    main = "Simulated
    Draws from Normal
    Distribution",
    col = "red")
    ```


    ```

Normal PDF Function Example

- We can overlay the bell curve density of a standard normal
  - ```
    hist(draws,
    xlim = c(-4,4),
    prob = T,
    xlab = "y",
    main = "Simulated
    Draws from Normal
    Distribution",
    col = "red")
    curve(normal(x,
    mu = 0, sigma =
    1),
    add = T,
    col = "blue",
    lwd = "2")
    ```


    ```

Normal PDF Function Example

- We can also use our draws to integrate the normal distribution.
- We can, as before, use the `integrate` function in R:
  - `integrate(normal, lower = -1.96, upper = 1.96, mu = 0, #put any extra arguments needed for your function here sigma = 1)`
  - **Output:** `0.9500042 with absolute error < 1e-15`
- Or we can use our draws from the same standard normal:
  - `mean(draws > -1.96 & draws < 1.96)`
  - `.9506`

A Note on Plotting in R

There are a couple of ways to export your R plots:

- First, you can hit the "Export" button in your RStudio window and save your plot as a file
- A better way to export your R plots is by doing this in your code:

  ```
  setwd("~/Directortypath/filename.pdf")
  pdf("filename.pdf")
  ...plot code...
  dev.off()
  ```
- This also works with .png files and other filetypes, just make sure to change your file extensions and sub out the `pdf()` command for `png()`, etc.