1 Introduction

Contents

1.1 Preamble

- The subject matter of computational statistics is that of Statistics itself, but developed via computation rather than only through mathematics.

- The goal of the course is to present essential statistical concepts.
  - Simulation is used to illustrate the concepts and to provide understanding.
  - Mathematical development provides an alternative presentation of the same ideas, when that is possible, and is used to develop a tool or get insight into a concept.

- Because simulation is the primary means to develop this understanding,
  - the statistics/estimators/tests, etc. used should be of sufficient complexity that a complete mathematical treatment would be beyond the level of this course.

- Because the statistics/estimators/tests etc. can be complex, several numerical methods are introduced. E.g. Gradient descent, Newton's method, iteratively reweighted least squares, etc.

Programming in R

- Students are expected to program.
  - The language will be R; other languages are not accepted at this point.

- The idea is to convey programming concepts alongside the statistical concepts. The purpose of the programming, like mathematics, is to illustrate the statistical concepts.
  - This means that we will take advantage of the functional programming language R to write some rather general purpose code.

- Because the code, like mathematics, is being used to convey the statistical concepts, clarity and simplicity of the code is primary.
  - That is, the code is not production-level code but instead teaching-level code.

- Where possible, programming constructs should be those which will be seen again in higher level courses. For example, the tasks **Map** and **Reduce** are used to iterate or accumulate (respectively) over some list stucture. These tasks can easily be carried out with the `apply` family of functions in R.
  - Such functions are commonplace in functional programming and will be seen a lot by those who go on to "big data" applications involving computing which is distributed over many machines.

- Example: Mapping

```r
p = 3
# Perform an iterative task with a for loop
x = list()
set.seed(341)
for(i in 1:10^4){
  x[[i]] = matrix(rnorm(p^2),p,p)
}
print(x[1:2])
```

```
## [[1]]
##            [,1]       [,2]        [,3]
## [1,] -1.0596247 -0.9927813 -0.66327646
## [2,] -0.3078065  0.8335972 -0.05148524
## [3,]  0.8662984  0.4734632  1.46211061
##
## [[2]]
##            [,1]       [,2]       [,3]
## [1,] -0.7199292  1.7825284 -1.0770761
## [2,]  0.8222773 -1.1153666  0.5601894
## [3,]  1.3379703 -0.7554148 -0.1139378
```

```r
# Perform the same iterative task with an apply function
set.seed(341)
x = mapply( function(x) {  matrix(rnorm(p^2),p,p) }, 1:10^4, SIMPLIFY = FALSE)
print(x[1:2])
```

```
## [[1]]
##            [,1]       [,2]        [,3]
## [1,] -1.0596247 -0.9927813 -0.66327646
## [2,] -0.3078065  0.8335972 -0.05148524
## [3,]  0.8662984  0.4734632  1.46211061
##
## [[2]]
##            [,1]       [,2]       [,3]
## [1,] -0.7199292  1.7825284 -1.0770761
## [2,]  0.8222773 -1.1153666  0.5601894
## [3,]  1.3379703 -0.7554148 -0.1139378
```

- Example: Reducing

```r
# Perform an accumulative task with a foor loop
z = matrix(0, p, p)
for(i in 1:10^4){
  z = z + x[[i]]
}
print(z)
```

```
##            [,1]      [,2]      [,3]
## [1,]  183.14875 -17.56777 -99.76430
## [2,]  -48.49314 -47.29259 150.54937
## [3,] -124.91438 141.62495  45.92042
```

```r
# Perform the same accumulative task with an apply function
z = apply(simplify2array(x), MARGIN = c(1,2), FUN = sum)
print(z)
```

```
##            [,1]      [,2]      [,3]
## [1,]  183.14875 -17.56777 -99.76430
## [2,]  -48.49314 -47.29259 150.54937
## [3,] -124.91438 141.62495  45.92042
```

Depth

Many important topics (computational and statistical) are covered very lightly.

- For example, floating point arithmetic is passed over quickly, with some discussion of the potential consequences, just to give exposure to the difference between computational results and mathematical results.
- Similarly, for example, robust regression methods are treated superficially as just providing another set of equations to be solved by the methods here. The intention is for the notes and lecture presentations to provide fairly generic concepts and algorithms that may be applied in a wide variety of problems.
  - Details of particular situations (e.g. estimators, etc.) can be explored in assignments as instances of the generic approach. This gives some latitude to tailor and adapt assignments over time.

Mathematical vs. Computational approach

- In contrast to the computational approach, mathematical development will necessarily deal with simpler statistics/estimators/tests.
  - Besides being more tractable, the simplicity of the mathematics should follow on from STAT 231, MATH 237 (multivariable calculus; infinite sequences and series), MATH 235 (linear algebra, matrix decompositions).
  - There is no dependence on STAT 330, 331, 332, 333, 340, but there will be occasional conceptual intersections with these courses.

- Much of the learning in the course is expected to occur through the assignments.
  - These will involve programming in R, simulation, mathematical development/proof, and should both reinforce the classroom material and stretch the students understanding of the fundamental concepts.

Road map

- Populations
  - explicit and implicit defined attributes
- Samples (finite populations)
  - all possible, selecting samples, estimation
- Inference
  - target and study populations,
  - comparing sub-populations,
  - confidence intervals,
  - resampling.
- Prediction
  - accuracy over a population,
  - bias-variance tradeoff.

1.2 The statistical narrative

- We direct attention towards populations,
  - the units which define them, and
  - variates as measures on the units.
- All genuine populations are finite and these must be taken from real populations.
  - Some populations that are used throughout the notes are presented as examples.

Population attributes

- Summaries of finite populations (i.e. population attributes)
  - Defining attributes of interest will rely on statistical concepts, and
  - Population attributes might involve considerable calculation (e.g. gradient descent, etc.), but…
  - **No probability** appears for some time because we have the entire population, consequently
  - Estimation algorithms appear before any samples are obtained.

- Examples;
  - population average

$$\frac{1}{N}\sum_{i=1}^{N} y_i$$

  - the value of θ that minimizes the quantity below, defined over the population

$$\widehat{\theta} = \underset{\theta\in\mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{N} |y_i - \theta|$$

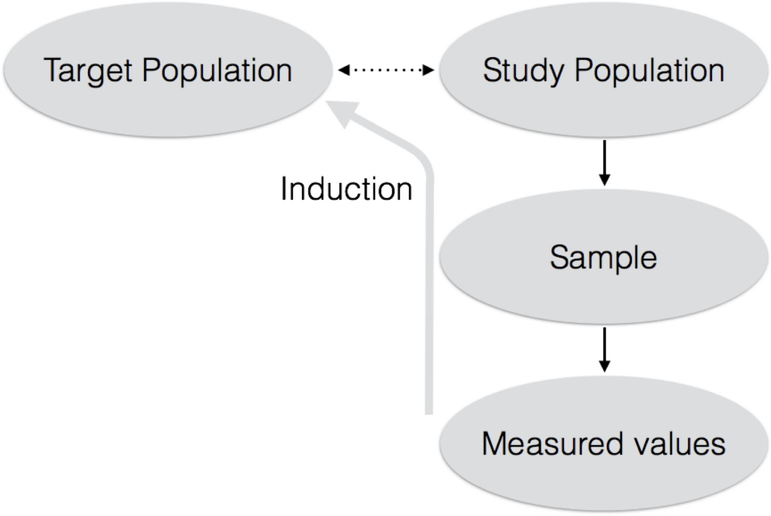
Samples

- All possible samples
  - are explored and characteristics of various population attributes examined within this context.
  - This is done without probability.
- All possible samples entail a combinatorial explosion for calculation,
  - the notion of selecting a subset of the samples is introduced.
  - Probability is used for the selection of a sample from a from all possible samples.

Computation

- The course leans heavily on computation and also on the concepts (and some language) from survey sampling
  - Note that STAT 332 is not a pre-req or co-req so the concepts used are introduced here in the computational context.
- R code is used to implement and reinforce the statistical sampling concepts.
  - Especially to illustrate the sampling distribution of any statistic and to compare different attributes, sample sizes, and sampling methods.

The inferential path of induction



The anatomy of a significance test

This is illustrated through the comparison of two sub-populations of a given population.

- Permutation tests are used for a variety of discrepancy measures.
- The problem of multiple testing.
- A summary of the concepts from statistical inference.

Resampling

- Given only a single sample,
  - the target–study–sample framework of induction can now be used to suggest that the single sample in hand could be used as if it were a study population.
- This leads to the introduction of bootstrap methods.

Prediction

- Oftentimes interest lies in **predicting** the value of a variate (the **response** variate) given the value of some **explanatory** variates.
- We build a **response model** that encodes how that prediction is to be carried out.
- How should the accuracy be measured?
  - Bias–variance tradeoff
  - training and test and/or cross-validation.