Experiment design (advanced)

Ivano Malavolta

VU VRIJE UNIVERSITEIT AMSTERDAM

LOOKING FURTHER

Roadmap

Advanced design types

Instrumentation

Advanced design types

We can have the following cases:

- 1 factor and 2 treatments (1F-2T)
- 1 factor and >2 treatments (1F-MT)
- 2 factors and 2 treatments (2F-2T)
- >2 factors, each one with >=2 treatments (MF-MT)

2 factors

- Things become more complex
- Two factors may *interact* with each other
- Interaction must be modeled:
  - τ_i : effect of treatment *i* (level of factor A)
  - β_j : effect of treatment *j* (level of factor B)
  - $(\tau\beta)_{ij}$: effect of the interaction between τ_i and β_j

Definition of effect

- Null hypothesis: no difference in means

- Typical model of an outcome (dependent variable):
    - $Y_{ij} = \mu + T_i + \text{error}$

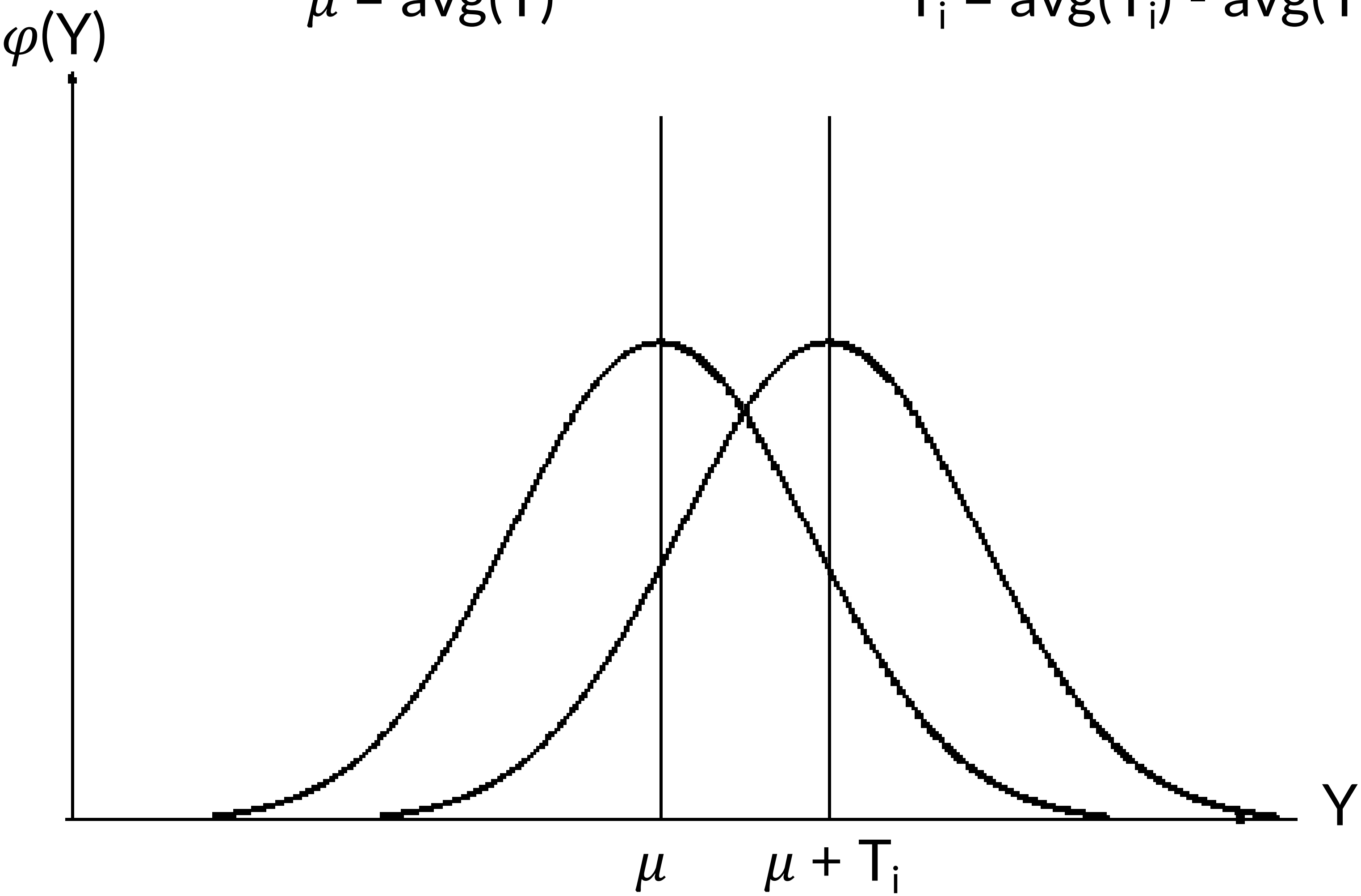
Outcome for **subject *j*** under **treatment *i***

Average of all values

Effect of treatment *i* (offset)

Definition of effect

μ = avg(Y) T_i = avg(Y_i) - avg(Y)



Interaction

Interaction: **non-additive** effects between factors

Additive model

$Y_{ij} = \mu + \tau_i + \beta_j + \text{error}$

	$B = 0$	$B = 1$
$A = 0$	6	7
$A = 1$	4	5

Non-Additive model

$Y_{ij} = \mu + \tau_i + \beta_j + (\tau\beta)_{ij} + \text{error}$

	$B = 0$	$B = 1$
$A = 0$	1	4
$A = 1$	7	6

 https://goo.gl/XUq8tp

2 factors, 2 treatments (2F-2T)

- Example:
  - **Subjects**: 8 applications
  - **Factor 1**: Connection Protocol
    - Treatment 1: HTTP
    - Treatment 2: HTTPS
  - **Factor 2**: Sorting Algorithm
    - Treatment 1: BubbleSort
    - Treatment 2: QuickSort

2F-2T: factorial design

- Consider **all possible combinations** of treatments

- Each treatment is **randomly** assigned to experimental objects
- **Balanced** design: each combination is assigned to an equal number of objects

		Factor 1: **Connection Protocol**	
		Treatment 1: **HTTP**	Treatment 2: **HTTPS**
Factor 2: **Sorting**	Treatment 1: **BubbleSort**	Application 4,6	Application 1,7
	Treatment 2: **QuickSort**	Application 2,3	Application 5,8

2F-2T: factorial design

- μ_i: mean of the dependent variable for treatment *i*
- μ_i=avg(P)
- τ_i : effect of treatment *i* (**HTTP/HTTPS**) of factor A (**Conn. Protocol**)
- β_j : effect of treatment *j* (**Bubble/Quick**) of factor B (**Sorting**)
- $(\tau\beta)_{ij}$: effect of the interaction between τ_i and β_j

2F-2T: factorial design

Null hypothesis for A: H_{0A}: $\tau_1 = \tau_2 = 0$

Null hypothesis for B: H_{0B}: $\beta_1 = \beta_2 = 0$

Null hypothesis for interaction: H_{0AB}: $(\tau\beta)_{ij} = 0 \ \forall \ i,j$

2F-2T: factorial design

Alternative hypothesis for A: $H_{1A}: \exists\, i \mid \tau_i \neq 0$

Alternative hypothesis for B: $H_{1B}: \exists\, j \mid \beta_j \neq 0$

Alternative hypothesis for interaction: $H_{1AB}: \exists\, (i,j) \mid (\tau\beta)_{ij} \neq 0$

2F-2T: 2-stage nested design

- Example:
  - **Objects**: 8 applications
  - **Factor 1**: **Interface**
    - Treatment 1: Web-based
    - Treatment 2: Client-based
  - **Factor 2**: **Programming Language**
    - Treatment 1,1: PHP
    - Treatment 1,2: ASP
    - Treatment 2,1: Java
    - Treatment 2,2: C++

2F-2T: 2-stage nested design

- One of the two factors has different treatments with respect to the other factor
- Balanced design, randomized application

Factor 1: **Interface**			
Treatment 1: **Web-based**		Treatment 2: **Client-based**	
Factor 2: **Programming Language**		Factor 2: **Programming Language**	
Treatment 1,1: **PHP**	Treatment 1,2: **ASP**	Treatment 2,1: **Java**	Treatment 1,2: **C++**
Application 1,3	Application 6,2	Application 7,8	Application 5,4

More than 2 factors

- Number of experimental groups explodes

- Total number of trials is at least k*n

  k = number of factors

  n = number of treatments per factor

- More trials = more subjects (larger sample size)

Factorial designs

FULL-COVERAGE

- every possible combination of all the alternatives of all the factors

$$N = \prod_{i=1}^{k} n_i$$

N = #trials (without considering the subjects)

k = #factors

n_i = #levels for i-th factor

Factorial designs – pros and cons

Discover the effects of each factor and its interactions with the other factors

BEWARE: combinatorial curse

SOLUTIONS?

$$N = \prod_{i=1}^{k} n_i$$

N = #trials

k = #factors

n_i = #levels for i-th factor

Latin square designs

- *Latin Square*: an *n x n* array with *n* different symbols
- Divide factors in *main factor* and *co-factors* (or *blocking factors)*
  - Levels of the main factors are the "letters" (in the cells)
  - Levels of the co-factors are rows and columns
- All levels of the main factor occur for each blocking factor

Latin square designs

- 3 factors
  - **Code Size:** Small, Medium, Large (*Main Factor*)
  - **Programming Language:** Java, C++, C (*Blocking Factor*)
  - **Operating System:** Windows, Linux, OS X (*Blocking Factor*)
- Total number of groups:
  - Full 3^3 factorial design: 27
  - Latin Square: 9

Latin square designs

		Factor 1: **Programming Language**		
		Treatment 1: **Java**	Treatment 2: **C++**	Treatment 3: **C**
Factor 2: **Operating System**	Treatment 1: **Windows**	Small	Medium	Large
	Treatment 2: **Linux**	Medium	Large	Small
	Treatment 3: **OS X**	Large	Small	Medium

Another example

- **Main factor**: object-orientation of language (A, B, C, D)
- **Co-factor 1**: size of the project (Very small to Very large)
- **Co-factor 2**: team experience (T1, T2, T3, T4)

		Team			
		T1	T2	T3	T4
Project type	Very small	A	B	C	D
	Small	D	A	B	C
	Large	C	D	A	B
	Very large	B	C	D	A

Pitfalls of Latin squares

- *Incomplete* or *partial* design
  - Key: balancing + randomization

		Factor			
		A	B	C	D
	1	X	X	X	
Blocks	2	X	X		X
	3	X		X	X
	4		X	X	X

- Assumption: factors **do not interact**
- Randomization is limited by design

VU

How to choose your design

It depends on the experiment you are executing.

First things first, identify and fix the main pillars of your experiment:

- main factor
- co-factors
- blocking factors

Think as follows: without any of them your experiment is either incomplete or trivial

Then: always start from a full factorial design as default design (since it is the most complete) and then, based on the feasibility of the experiment, decide if you can proceed with it or you need to resort to:

- lowering the number of subjects, repetitions, treatments, etc.
- changing the design of your experiment (e.g., using the Latin square method)

Instrumentation

Instrumentation

Goal: Provide a way to conduct and monitor the experiment

- Instrumentation must **not** affect our control of the experiment
- Types of instrumentation:
  - Objects (e.g. servers, apps)
  - Guidelines (checklists, documentation)
  - Measurement tools (power meters, profiling software, etc.)

What this module means to you?

- You know how to design an experiment
- Experiment design is an essential choice when doing an experiment
- Constraints on statistical methods
- **Use a simple design**
- Maximize the usage of the available subjects
    - automation will be your friend here

Readings

Chapter 8

Chapter 5

IMPORTANT - Checklist for making a good experiment design

(section 5.9)