

Session 1: Data Preparation Practical

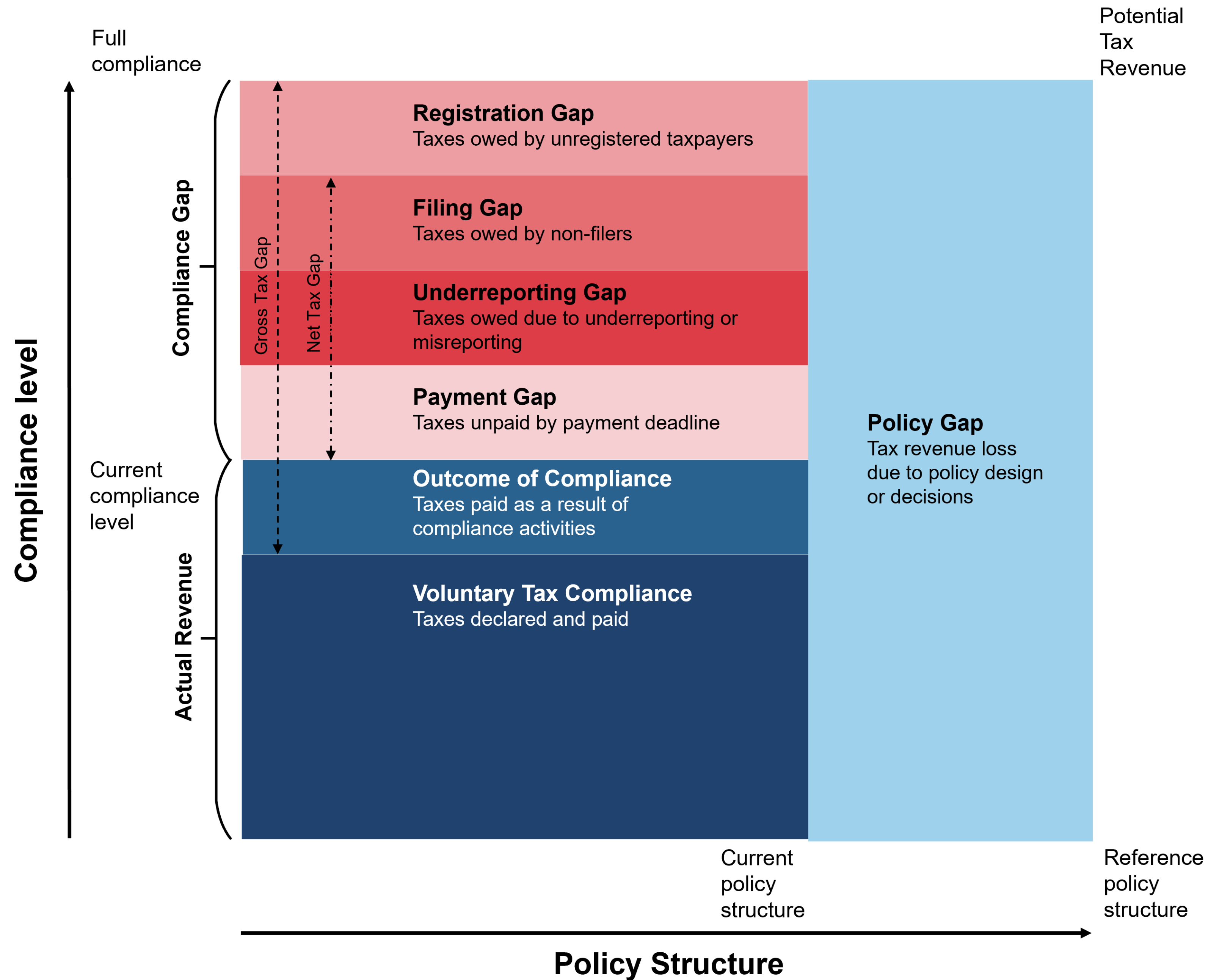


UNU
WIDER

Break out groups:

Maria Jousté + Amina Ebrahim + Sebastian Castillo

Kwabena Adu-Ababio + Karri Vuoristo



Key Learnings from the Theory videos

- ❖ Tax Gap definitions
- ❖ Benefits and limitations of the Bottom-up approach
- ❖ Data requirements for the tax gap estimation
- ❖ How to prepare data for tax gap estimation

Aim of this practical session

1. How to prepare tax return and audit data for tax gap estimation
2. Show how to execute the steps in Toolkit Part 1

We are not teaching how to use Stata, merely guiding the steps to follow.

Own implementation requires contextualizing to the tax type (VAT, CIT, PAYE), and data structure and challenges.

Exercise 1 – Stata Environment setup & data import

In this exercise, you will try to implement the First Stage of the Toolkit Part 1: Data preparation. This stage aims to ensure all necessary packages are installed and updated, prepare the STATA environment, set up directory paths, and import and check the CIT return data files.

- a) Run the code lines 23 to 28 to prepare the Stata environment.
- b) Modify line 31 to your own file directory and run line 32 to set the directory path
- c) Import the cit_return data (csv file), rename the variables region (region, isci, year) to (province, isic_code, rtn_year) respectively and save the data in Stata format with the name “cit”. Browse the data to check.

Exercise 1 – Stata Environment setup & data import

a) Run the code lines 23 to 28 to prepare the Stata environment.

```
set more off  
clear all  
clear matrix  
set linesize 200  
set matsize 800  
set seed 201807
```

b) Modify line 31 to your own file directory and run lines 32 & 33 to set the directory paths

```
global dir "<YOUR PATH>"  
global data "$dir\Data"  
global results "$dir\Results"
```

Exercise 1 – Solutions

- c) Import the cit_return data (csv file), rename the variables region (region, isci, year) to (province, isic_code, rtn_year) respectively and save the data in Stata format with the name “cit”. Browse the data to check.

```
import delimited "$data\cit_return.csv", clear
rename region province
rename isci isic_code
rename year rtn_year
browse
save "$data\cit.dta", replace
```

Exercise 2 – Data management process for CIT data

In this exercise, you will try to implement the Second Stage of the Toolkit Part 1:
Data management process for the return data

- a) Identify the number of exact duplicates in the CIT return data. Drop exact duplicates.
- b) Taxpayers sometimes make multiple revisions of their returns. In this context, a revision is a duplicate record for the same taxpayer (tpin) and year (rtn_year). Keep the latest return revision.
- c) Tax return data can be quite large. Browse the dataset and remove any unnecessary variables. Then sort, reorder, and save the cleaned dataset

Exercise 2 – Solution

- a) Identify the number of exact duplicates in the CIT return data. Drop exact duplicates.

```
use "$data\cit.dta", clear
duplicates report
duplicates drop
```

- b) Taxpayers sometimes make multiple revisions of their returns. In this context, a revision is a duplicate record for the same taxpayer (tpin) and year (rtn_year). Keep the latest return revision.

```
sort tpin rtn_year
duplicates tag tpin rtn_year, gen(dup0)
tabulate dup0
bys tpin rtn_year: egen latest = max(revision) if dup0 != 0
keep if latest==revision
```

The toolkit also assists with appending datasets and duplicates from multiple files. This is not necessary in the exercise

Exercise 2 – Solution

- c) Tax return data can be quite large. Browse the dataset and remove any unnecessary variables. Then sort, reorder, and save the cleaned dataset

```
browse  
drop revision dup0 latest  
sort tpin rtn_year  
order tpin rtn_year tax_region city province isic_code  
broad_sector //  
save "$data\cit_clean.dta", replace
```

CIT returns are typically annualised by PAYE and VAT data usually need to be collapsed to the annual level. The cleaned CIT data is ready for merging with the audit data. But first, the Audit data needs to be cleaned.

Exercise 3 – Data management process for Audit data

In this exercise, you will try to implement the Third Stage of the Toolkit Part 1: Data management process for the audit data.

- a) Import the audit data (csv file), rename total_tax to total_assessed, keep audits related to CIT for merging with CIT returns and save the data in Stata format with the name "audit". Browse the data to check.
- b) Investigate the Total Assessed variable. Check for negative assessments. Let's assume a negative assessment means the taxpayer is compliant. Replace negative assessments with zero to indicate compliance. (You may reconsider this assumption in your context)
- c) Audits can be conducted for multiple years, but often only a total assessment is reported. Allocate total assessments across the audited years to create an annual assessment amount.
- d) Remove all exact duplicates, ensuring only unique records remain and save the cleaned audit data.

Exercise 3 – Solutions

- a) Import the audit data (csv file), rename total_tax to total_assessed, keep audits related to CIT for merging with CIT returns and save the data in Stata format with the name "audit". Browse the data to check.

```
import delimited "$data\audit_data.csv", clear  
rename totaltax total_assessed  
tab tax_head  
keep if tax_head=="CIT"  
save "$data\audit.dta", replace  
browse
```

Exercise 3 – Solutions

- b) Investigate the Total Assessed variable. Check for negative assessments. Let's assume a negative assessment means the taxpayer is compliant. Replace negative assessments with zero to indicate compliance. (You may reconsider this assumption in your context)

```
codebook total_assessed  
count if total_assessed<0  
replace total_assessed = 0 if total_assessed < 0
```

Exercise 3 – Solutions

- c) Audits can be conducted for multiple years, but often only a total assessment is reported. Allocate total assessments across the audited years to create an annual assessment amount.

```
br initial_period final_period  
gen auditedyears=(final_period-initial_period)+1  
gen annual_assessed = total_assessed / auditedyears  
drop if annual_assessed==.
```

- d) Remove all exact duplicates, ensuring only unique records remain and save the cleaned audit data.

```
duplicates report  
save "$data\audit_clean.dta", replace
```

The toolkit included detailed data preparation notes for a variety of challenges in the audit data that the team have encountered. Without a consistent form, the preparation of the audit data requires careful attention and consideration.

Exercise 4 – Cleaning the date variable

In audit data, dates often appear in inconsistent formats. In this exercise, we will show an example of these date problems and how to convert them to STATA formats. In this example, you are given date_problem data with two date variables and eight observations.

- a) Import the date_problem data. Examine the assessment date variable to identify the problems
- b) Dates with missing or incorrect DD, MM or YYYY require attention to clean. Browse the incorrect dates and attempt to correct them.
- c) Convert the corrected date variable to a Stata format and save the dataset.

Exercise 4 – Solutions

- a) Import the date_problem data. Examine the assessment date variable to identify the problems

```
import delimited "$data/date_problem.csv", clear  
list assessment_date assessment_year  
br assessment_date assessment_year
```

Exercise 4 – Solutions

- b) Dates with missing or incorrect DD, MM or YYYY require attention to clean. Browse the incorrect dates and attempt to correct them.

```
browse if missing(date(assessment_date, "DMY")) &  
missing(date(assessment_date, "MDY")) & missing(date(assessment_date, "YMD"))  
& missing(date(assessment_date, "MYD"))  
replace assessment_date = "01/11/2019" if assessment_date == "01-Nov"  
replace assessment_date = "05/12/2018" if assessment_date == "05/012/2018"  
replace assessment_date = "16/12/2017" if assessment_date == "16/21207"  
replace assessment_date = "01/02/2021" if assessment_date == "Feb-21"  
replace assessment_date = "03/01/2018" if assessment_date == "3.1.12018"
```

Exercise 4 – Solutions

c) Convert the corrected date variable to a Stata format and save the dataset.

```
generate assessment_date_clean = date(assessment_date, "DMY")  
format assessment_date_clean %td  
drop assessment_date  
save "$data\date_clean.dta", replace
```

More information available online

wider.unu.edu

Follow us on LinkedIn, Facebook, and Twitter @UNUWIDER