Handbook for Writing Quantitative Academic Research Papers

Installing, Configuring, and Using Positron, Quarto, and Zotero

Dianyi Yang [†,1,*] Yuxin Fu [†,2,*]

2025-09-27

[†] These authors contributed equally to this work.

[1] Department of Politics and International Relations, University of Oxford
[2] Department of Government and Public Administration, University of Macau

[*] Correspondence: Dianyi Yang <dianyi.yang@politics.ox.ac.uk>, Yuxin Fu <yc57370@um.edu.mo>

Abstract

Work in progress.

Keywords— Work in progress.

Work in progress.

Table of contents

1 Installation

This section will guide you through the installation and setup of Positron, Quarto, and Zotero for writing quantitative academic research papers.

1.1 Install R & Positron & Quarto

First, we need to download and install R, Positron and Quarto.

1. Follow the instructions on CRAN mirror to download and install R.
2. Follow the instructions on Positron Download to download Positron.
3. Run SETUP and install Positron. Check all of the boxes when asked to.

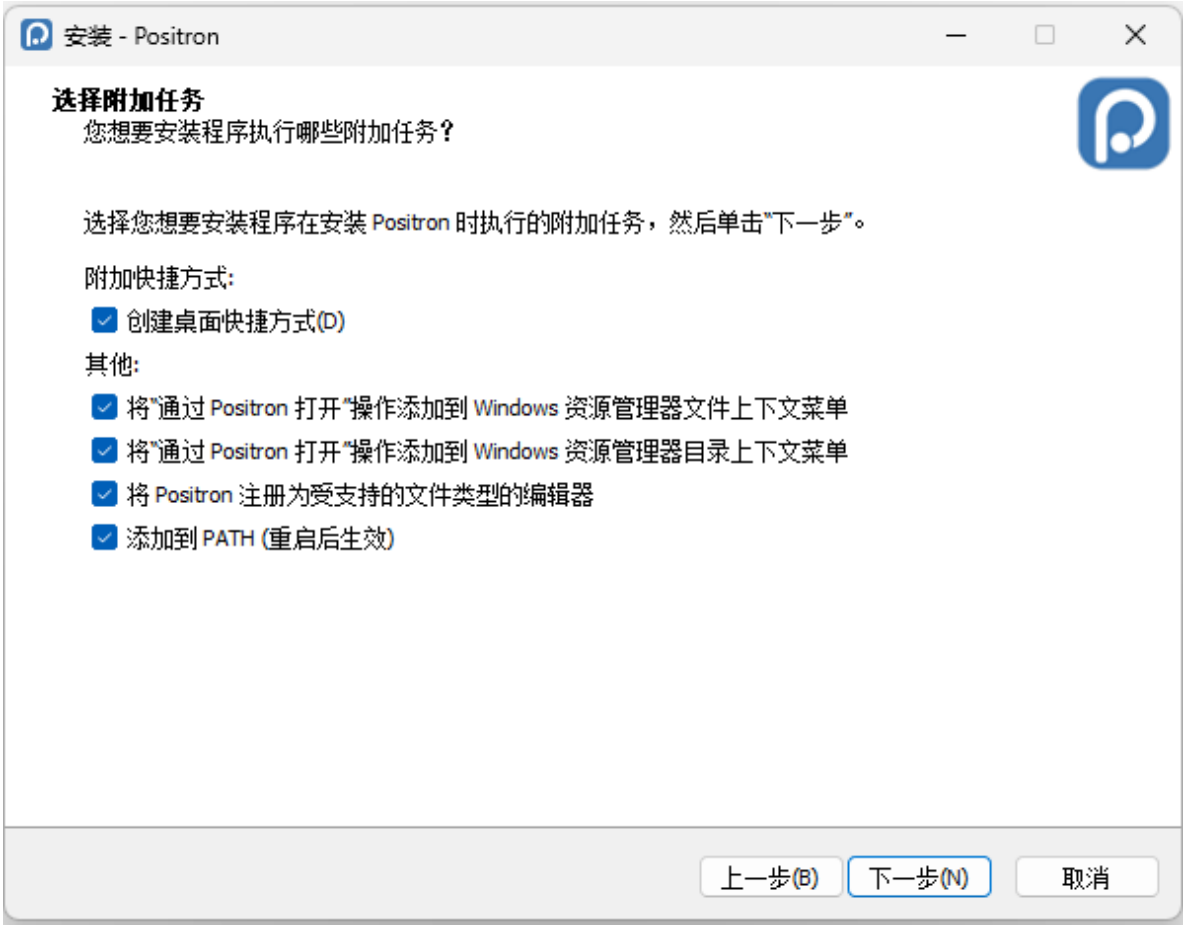

Figure 1: Check all of the boxes

4. Follow the instructions on Quarto Get Started to download and install Quarto.

1.2 Set up System Environment Variables

Next, we need to set up the system environment variables for proxys.

For Windows:

1. Open the Windows Start Menu and search for "Env".
2. Click on "Edit the system environment variables".
3. In the System Properties window, click on the "Environment Variables" button.
4. In the Environment Variables window, under the "System variables" section, click on the "New" button.
5. In the New System Variable window, enter `HTTP_PROXY` as the Variable name and `http://127.0.0.1:xxxx` as the Variable value. Click OK. (Replace `xxxx` with the port number you are using for your proxy.)
6. Repeat step 5 to create another system variable named `HTTPS_PROXY` with the same value `http://127.0.0.1:xxxx`.
7. Repeat step 5 to create another system variable named `NO_PROXY` with the value `localhost,127.0.0.1`.
8. Click OK to close the Environment Variables window. Click OK to close the System Properties window.

For Mac:

Work in progress.

1.3 Join Github Eduction

To use Github Copilot, we need to join the Github Education program.

1. Open your web browser and go to GitHub Education.
2. Click on the "Join GitHub Education" button.
3. Follow the instructions to verify your student status and complete the application process.

1.4 Update Positron and Set Up Github Copilot Chat

Now, we need to update Positron and set up Github Copilot Chat.

1. Open Positron.
2. Click on the "Manage" Icon on the bottom left corner.
3. Click on "Settings" to open the Settings tab.
4. In the Settings tab, search for "channel", and switch the "releases" to "dailies".
5. Click on the "Manage" Icon on the bottom left corner. Click on "Check for Updates" to update Positron to the latest version. After the update is complete, restart Positron.
6. Repeat step 2 & 3 to open the Settings tab again.
7. In the Settings tab, search for "assistant". Check on the box "Enable Positron Assistant".
8. Click on the "Chat" Icon on the left sidebar to open the Chat tab.
9. In the Chat tab, click on "add model provider" button.
10. In the Configure Language Model Provider window, select "GitHub Copilot". Click on "Sign in".
11. In the GitHub login page, enter your GitHub credentials to sign in.
12. After signing in, you will be redirected back to Positron. You can now use the LLMs provided by GitHub Copilot to assist with your coding.

1.5 Install R Package Manager

Next, we need to install the R Package Manager to manage R packages.

1. Open Positron. Click on the "Extensions" Icon on the left sidebar.
2. Search for "Positron R Package Manager". Click on "Install" to install the extension.
3. Once the extension is installed and activated, you can find the R Package Manager icon on the left sidebar.
4. Drag the icon to the secondary sidebar on the right.

1.6 Install Zotero

Finally, we need to download and install Zotero for managing references.

Work in progress.

2 Usage

This section will guide you through the usage of Positron, Quarto, and Zotero for writing quantitative academic research papers.

2.1 Fetch ORCID Template

Now, every time before you start writing a new paper, we could install the ORCID template for better LaTeX formatting.

1. Open Positron, create a qmd file as your transcript. Click on "TERMINAL" on the bottom sidebar.
2. In the terminal, type the following command and press Enter: "quarto add kv9898/orcid".
3. Enter "Y" when asked to trust the authors and to continue.
4. You can now work on an ORCID template. Copy and paste the example syntax from here: ORCID Template.
5. Click "Preview" on the top bar to try compile the document. Follow the instructions on the terminal to install any missing packages.

2.2 YAML header

You can make changes to the default YAML header to fit your needs. Refer to the comments on the code in the template for more information.

Work in progress.

2.3 An Example Regression Analysis

Here is an example regression analysis using the ORCID template and the mtcars dataset to show you some basic syntax when you are writing a quantitative academic research paper.

First, let's examine the relationship between car weight and fuel efficiency using a scatter plot.

Insert an r code cell by clicking the "Insert Code Cell" button on the top bar, and copy and paste the following code into the code cell:

```r
# Assign a label to the scatter plot for cross-referencing
# Labels for figures should have the prefix "fig-"
#| label: fig-scatter
# Assign a caption to the scatter plot
#| fig-cap: "Relationship between Car Weight and Fuel Efficiency"
# Set echo, warning, and message to false for the plot output only
#| echo: false
#| warning: false
#| message: false
# Install the necessary packages
install.packages(c("ggplot2", "dplyr"))
# If the packages are already installed, you can skip the above line
library(ggplot2)
library(dplyr)
# Create descriptive statistics plot
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point(aes(color = factor(cyl)), size = 3) +
  geom_smooth(method = "lm", se = TRUE, color = "blue") +
  labs(
    x = "Weight (1000 lbs)",
    y = "Miles per Gallon",
    color = "Cylinders"
  ) +
  theme_minimal()
```

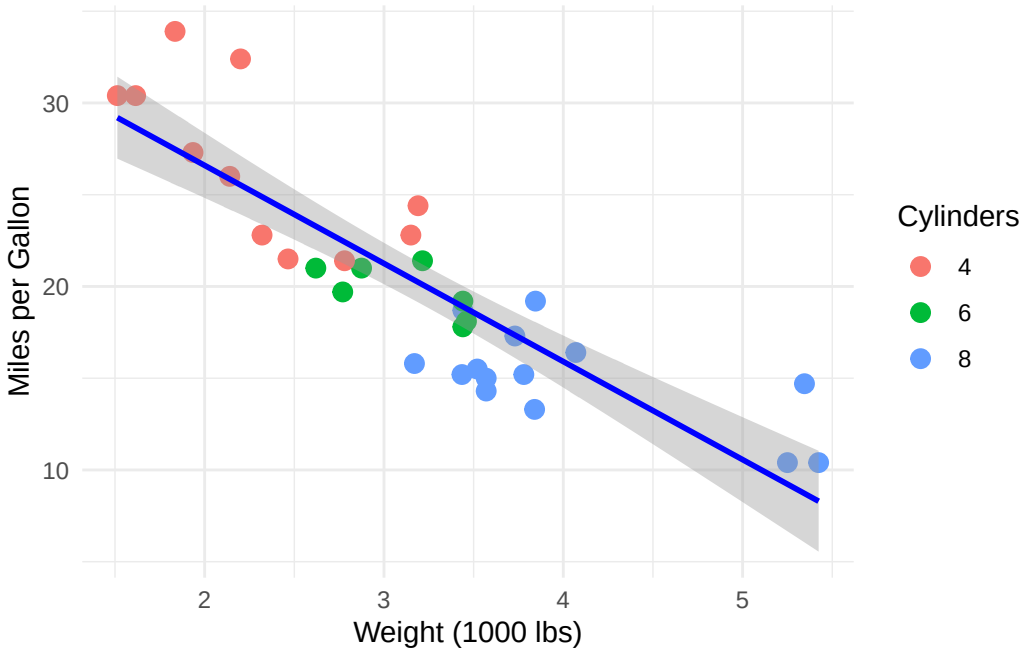

Figure 2: Relationship between Car Weight and Fuel Efficiency

Next, we'll perform a regression analysis. Equations written in LaTeX syntax should be covered by double dollar signs `$$ ... $$` and be assigned a label using `{#label}` for cross-referencing.

```
$$
mpg = \beta_0 + \beta_1 \cdot weight + \beta_2 \cdot cylinders + \epsilon
$$ {#eq-regression}  #Labels for equations should have the prefix "eq-"
```

$$mpg = \beta_0 + \beta_1 \cdot weight + \beta_2 \cdot cylinders + \epsilon \tag{1}$$

We can also use "&" to align equations. For example, if we want our equations to be aligned at the equal signs and quote them as one group, we can write:

```
$$
\begin{aligned}
1+2&=3\\
3+4&=7\\
11&=5+6
\end{aligned}
$$ {#eq-group}
```

Note that here we use `\begin{aligned} ... \end{aligned}` to align a group of equations, and we use `\\` to indicate line breaks, and `&` to indicate the alignment point.

The three equations are now quoted as one group and aligned at the equal signs:

$$\begin{aligned} 1 + 2 &= 3 \\ 3 + 4 &= 7 \\ 11 &= 5 + 6 \end{aligned} \tag{2}$$

Finally, let's create a regression table to summarize our regression results. We will use the `modelsummary` package to create a well-formatted regression table.

```
# Assign a label to the regression table for cross-referencing
# Labels for tables should have the prefix "tbl-"
#| label: tbl-regression
# Assign a caption to the regression table
#| tbl-cap: "Regression Results: Fuel Efficiency Model"
# Set echo, warning, and message to false for the plot output only
#| echo: false
#| warning: false
#| message: false
library(modelsummary)
# Fit regression models
model1 <- lm(mpg ~ wt, data = mtcars)
model2 <- lm(mpg ~ wt + cyl, data = mtcars)
model3 <- lm(mpg ~ wt + cyl + hp, data = mtcars)
# Create regression table
modelsummary(
  list("Model 1" = model1, "Model 2" = model2, "Model 3" = model3),
  stars = TRUE,
  gof_map = c("nobs", "r.squared", "adj.r.squared")
)
```

Table 1: Regression Results: Fuel Efficiency Model

	Model 1	Model 2	Model 3
(Intercept)	37.285***	39.686***	38.752***
	(1.878)	(1.715)	(1.787)
wt	−5.344***	−3.191***	−3.167***
	(0.559)	(0.757)	(0.741)
cyl		−1.508**	−0.942+
		(0.415)	(0.551)
hp			−0.018
			(0.012)
Num.Obs.	32	32	32
R2	0.753	0.830	0.843
R2 Adj.	0.745	0.819	0.826

+ p <0.1, * p <0.05, ** p <0.01, *** p <0.001

The equations, figures, and tables are automatically numbered in a correct sequence, and could be cross-referenced in the text.

To create proper cross-references in Quarto, use the following syntax: `@label` at where you want to make cross references, where *label* is the label you assigned to the figure, table, or equation. For instance, `@eq-regression` refers to the regression equation Equation 1, `@eq-group` refers to the group of equations Equation 2, `@fig-scatter` refers to the scatter plot Figure 2, and `@tbl-regression` refers to the regression table Table 1. You could refer to Quarto Cross-References Documentation for more information on cross-reference.

2.4 An Example Literature Review

Work in progress.