Tutorial Week 8 and 9: Matching and Entropy Balancing

Problem Set - Propensity Scores, Matching, and Synthetic Control

Maximilian Birkle Daniel Lehmann Henry Lucas

2025-11-14

Contents

1 Load Packages

2 Part I: Propensity Scores, Matching, and Robust Post-Matching Inference

2.1 Background

Research Question: Do **United Nations interventions** help shorten the duration of civil wars?

Gilligan and Sergenti (2008) use matching methods to re-evaluate earlier findings suggesting that UN interventions *prolong* conflict. They argue that this conclusion stems from **selection bias** — the UN tends to intervene in the *worst* conflicts.

Dataset: `war_pre_snapshots.dta`

Each row represents a **conflict episode** observed before a potential UN intervention. Our goal is to estimate the causal effect of UN involvement (`UN`) on the length of the conflict (`t1 - t0`), while balancing on key pre-treatment covariates.

2.2 Load Data

```r
# Read the UN intervention dataset
data_un <- read_dta("war_pre_snapshots.dta")
```

2.3 Define Treatment and Covariates

```r
# Treatment variable: UN intervention (1 = Yes, 0 = No)
treat <- data_un$UN

# Outcome variable: conflict duration (t1 - t0)
outcome <- data_un$t - data_un$t0

# Covariates for propensity score model
covar <- c(
  "inter", "deaths", "couprev", "sos", "drugs", "t0",
  "ethfrac", "pop", "lmtnest", "milper",
  "eeurop", "lamerica", "asia", "ssafrica"
)

# Create covariate matrix
X <- data_un[, covar]

dta_new <- data.frame(
  treat = treat,
  outcome = outcome,
  X)
```

2.4 Q0. First Check of the Data

2.4.1 Tasks:

1. Why might UN interventions **not** be randomly assigned across conflicts?
2. Which of the listed variables are most likely to confound the relationship between `UN` and conflict duration? Run a quick logistic regression and check.

```r
# Logistic regression: UN intervention as function of covariates
logit_selection <- glm(treat ~ inter + deaths + couprev + sos + drugs + t0 +
                         ethfrac + pop + lmtnest + milper +
                         eeurop + lamerica + asia + ssafrica,
                       data = dta_new,
                       family = binomial(link = "logit"))

# Create stargazer table
stargazer(
  logit_selection,
  type = "latex",
  title = "Selection into UN Intervention: Logistic Regression",
  header = FALSE,
  dep.var.labels = "UN Intervention (1 = Yes)",
  covariate.labels = c(
    "Internationalized Conflict",
    "Battle Deaths",
    "Coup/Revolution",
    "Strategic Oil Supply",
    "Drug Activity",
    "Conflict Start Year",
    "Ethnic Fractionalization",
    "Log Population",
    "Log Mountainous Terrain",
    "Military Personnel per Capita",
    "Eastern Europe",
    "Latin America",
    "Asia",
    "Sub-Saharan Africa",
    "Constant"
  ),
  no.space = TRUE,
  omit.stat = c("aic", "ll"),
  notes = "Standard errors in parentheses.",
  notes.align = "l",
  digits = 3
)
```

Discussion:

Why might UN interventions not be randomly assigned?

Cases where the UN intervenes are quite different from where they do not, which is why UN missions are not randomly assigned.

The decision to intervene depends on a number of factors, including the UN Security Council's selection process and whether the UN even pays attention to these cases.

Table 1: Selection into UN Intervention: Logistic Regression

	Dependent variable:
	UN Intervention (1 = Yes)
Internationalized Conflict	0.929
	(0.846)
Battle Deaths	−0.156
	(0.218)
Coup/Revolution	−0.113
	(1.234)
Strategic Oil Supply	−1.075
	(0.994)
Drug Activity	3.238^{***}
	(1.087)
Conflict Start Year	0.005
	(0.003)
Ethnic Fractionalization	−0.016
	(0.017)
Log Population	−0.271
	(0.471)
Log Mountainous Terrain	0.271
	(0.321)
Military Personnel per Capita	-1.123^{***}
	(0.351)
Eastern Europe	2.186
	(1.485)
Latin America	-3.775^{**}
	(1.731)
Asia	−2.440
	(1.630)
Sub-Saharan Africa	−1.337
	(1.521)
Constant	3.002
	(4.006)
Observations	1,227
Note:	*p<0.1; **p<0.05; ***p<0.01 Standard errors in parentheses.

Because the “treated” (intervention) and “untreated” (no intervention) cases are so different on so many variables, it becomes almost impossible to distinguish whether a causal effect is due to the treatment itself or some function of these other confounding variables. When you have differences that large between units, you face an extreme problem of constructing the counterfactual—that is, figuring out what would have happened had the UN not intervened.

Which variables show strong associations with UN intervention?

Looking at the results in table 1, we can see a high positive coefficient for **Drug Activity** (coef=3.238, p<0.01), meaning that the UN is more likely to intervene where there is drug-related activity present. This also makes intuitive sense, as interventions are fostered by international drug trafficking concerns. This is a prime example of why it is hard to compare regimes across regions that could potentially receive UN intervention. Since regimes, rebel groups, or terrorist groups in politically unstable systems often build their economic foundation on scarce resources (e.g., diamonds, oil, etc.) and also drugs, their different capabilities (e.g., geographic location, natural resources) offer different incentives for the UN to intervene, especially when their economic activities pose a global threat, as is the case with drugs.

Similarly, in countries with high per capita **military personnel** (coef=−1.123, p<0.01), countries with higher military capacity are less likely to receive UN intervention. This also makes intuitive sense, as weak military states would make it easier and safer to intervene, and they have more need for external support.

Furthermore, countries located in **Latin America** are less likely to receive UN intervention (coef=−3.775, P<0.05). This finding might be slightly confusing at first glance, although it might be explained by a couple of points. As many states in Latin America have faced high levels of political instability and also have a long history with drug trafficking, we might assume that they would be more likely to receive UN interventions. The absence of this finding might be due to the fact that Latin America has always been undisputedly located in the sphere of influence of the United States—historically articulated in the Monroe Doctrine (1823)—which has carried out its own “peacekeeping missions” across the continent, not based on decisions of the UN or other multilateral actors after the Cold War. Looking at recent developments around the coast of Venezuela, this dynamic still dominates today.

Overall, this regression table gives some preliminary hints that UN interventions are indeed not randomly assigned. It demonstrates clear selection bias: the UN systematically intervenes in cases with specific, observable characteristics (like high drug activity or weak militaries) and avoids others (like those in Latin America). While other factors like battle deaths or ethnic fractionalization don’t show a significant effect in this model, the strong predictors identified here are exactly why it is extremely useful to use matching to create a valid comparison group.

2.5 Q1. Estimating Propensity Scores

2.5.1 Theoretical Background

Let $T_i \in \{0, 1\}$ be the treatment indicator and $X_i = (X_{i1}, X_{i2}, \dots, X_{ip})$ the vector of pre-treatment covariates.

The **propensity score** is defined as:

$$e(X_i) = P(T_i = 1 \mid X_i)$$

A **propensity score** takes the necessary covariates and estimates the a maximum likelihood model of the conditional probability of treatment (with fitted values that are bounded between 0 and 1), and uses the predicted values from the estimation to collapse those covariates into a single scalar.

2.5.2 Tasks:

1. Define the propensity score
2. Estimate $\widehat{e}(X_i)$ in two ways:
    - **(a)** Logistic regression: $\mathrm{logit}(e(X_i)) = X_i'\beta$
    - **(b)** Random forest classifier
3. Report mean, SD, and range of $\widehat{e}(X_i)$ for treated and control
4. Create histogram/density plot by treatment status

```
# (a) Logistic regression propensity score

ps_logistic <- glm(treat ~ inter + deaths + couprev + sos + drugs + t0 +
                     ethfrac + pop + lmtnest + milper +
                     eeurop + lamerica + asia + ssafrica,
                   data = dta_new,
                   family = binomial(link = "logit"),
                   na.action = na.exclude)


dta_new$ps_logistic_probs <- predict(ps_logistic,
                                     newdata = dta_new,
                                     type = "response")
```

```
# (b) Random forest propensity score

set.seed(68159) #for reproducability

ps_random_forest <- randomForest(
  factor(treat) ~ inter + deaths + couprev + sos + drugs + t0 +
    ethfrac + pop + lmtnest + milper +
    eeurop + lamerica + asia + ssafrica,
  data = dta_new,
  ntree = 1000,
  importance = TRUE
)

# Extract predicted probabilities

dta_new$ps_random_forest <- predict(ps_random_forest, type = "prob")[, "1"]
```

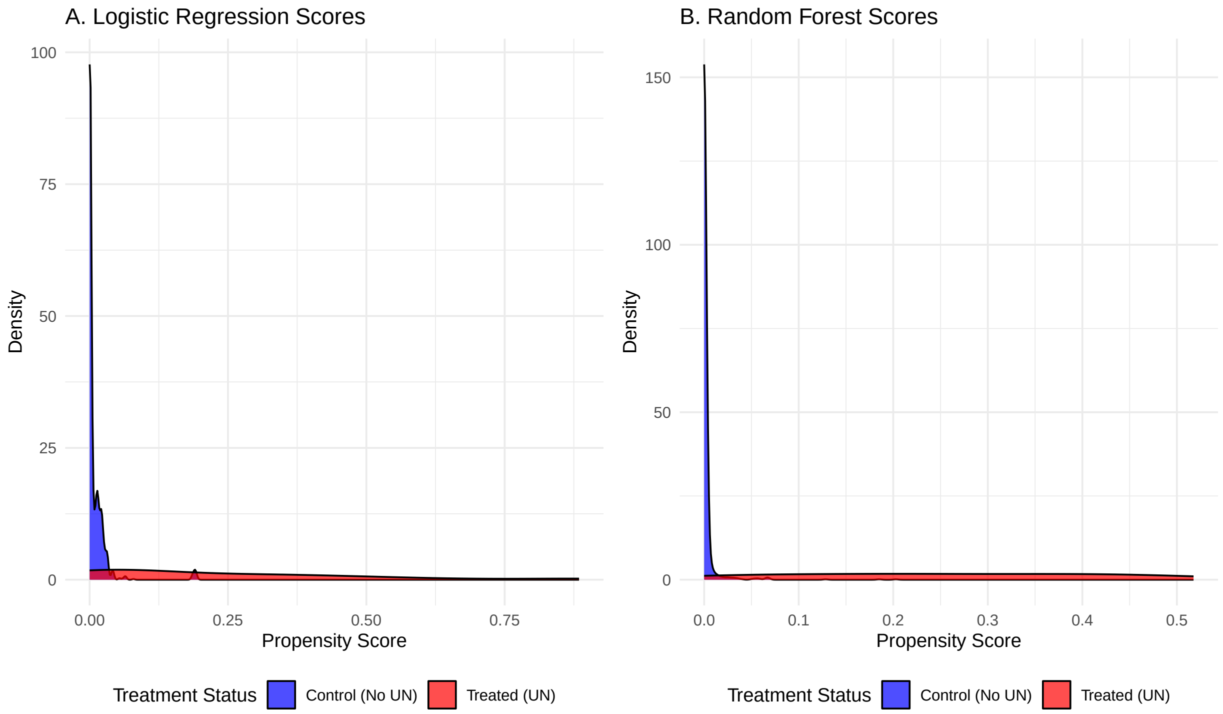
```
# Summary statistics of propensity scores

# Logistic Regression Stats
logit_control_mean <- mean(dta_new$ps_logistic_probs[dta_new$treat == 0], na.rm = TRUE)
logit_control_sd   <- sd(dta_new$ps_logistic_probs[dta_new$treat == 0], na.rm = TRUE)
logit_control_min  <- min(dta_new$ps_logistic_probs[dta_new$treat == 0], na.rm = TRUE)
logit_control_max  <- max(dta_new$ps_logistic_probs[dta_new$treat == 0], na.rm = TRUE)

logit_treated_mean <- mean(dta_new$ps_logistic_probs[dta_new$treat == 1], na.rm = TRUE)
logit_treated_sd   <- sd(dta_new$ps_logistic_probs[dta_new$treat == 1], na.rm = TRUE)
logit_treated_min  <- min(dta_new$ps_logistic_probs[dta_new$treat == 1], na.rm = TRUE)
logit_treated_max  <- max(dta_new$ps_logistic_probs[dta_new$treat == 1], na.rm = TRUE)
```

```r
# Random Forest Stats
rf_control_mean <- mean(dta_new$ps_random_forest[dta_new$treat == 0], na.rm = TRUE)
rf_control_sd   <- sd(dta_new$ps_random_forest[dta_new$treat == 0], na.rm = TRUE)
rf_control_min  <- min(dta_new$ps_random_forest[dta_new$treat == 0], na.rm = TRUE)
rf_control_max  <- max(dta_new$ps_random_forest[dta_new$treat == 0], na.rm = TRUE)

rf_treated_mean <- mean(dta_new$ps_random_forest[dta_new$treat == 1], na.rm = TRUE)
rf_treated_sd   <- sd(dta_new$ps_random_forest[dta_new$treat == 1], na.rm = TRUE)
rf_treated_min  <- min(dta_new$ps_random_forest[dta_new$treat == 1], na.rm = TRUE)
rf_treated_max  <- max(dta_new$ps_random_forest[dta_new$treat == 1], na.rm = TRUE)


# Data Frame with results
summary_stats_q1 <- data.frame(
  Statistic = c("Mean", "Std. Deviation", "Min", "Max"),
  Logit_Control = c(logit_control_mean, logit_control_sd, logit_control_min, logit_control_max),
  Logit_Treated = c(logit_treated_mean, logit_treated_sd, logit_treated_min, logit_treated_max),
  RF_Control = c(rf_control_mean, rf_control_sd, rf_control_min, rf_control_max),
  RF_Treated = c(rf_treated_mean, rf_treated_sd, rf_treated_min, rf_treated_max)
)

tbl <- knitr::kable(
  summary_stats_q1,
  digits = 3,
  col.names = c("Statistic", "Control (T=0)", "Treated (T=1)", "Control (T=0)", "Treated (T=1)"),
  caption = "Table 1: Propensity Score Distribution by Model and Treatment Status"
)

if (knitr::is_latex_output() || knitr::is_html_output()) {
  tbl <- tbl %>%
    kableExtra::add_header_above(c(" " = 1, "Logistic Regression" = 2, "Random Forest" = 2)) %>%
    kableExtra::kable_styling(bootstrap_options = "striped", full_width = FALSE)
}

tbl
```

Table 2: Table 1: Propensity Score Distribution by Model and Treatment Status

	Logistic Regression		Random Forest	
Statistic	Control (T=0)	Treated (T=1)	Control (T=0)	Treated (T=1)
Mean	0.010	0.213	0.002	0.250
Std. Deviation	0.026	0.253	0.011	0.169
Min	0.000	0.000	0.000	0.000
Max	0.194	0.884	0.203	0.518

```r
# Density plot of propensity scores by treatment status

# Logistic Regression Plot
p_logit <- ggplot(dta_new, aes(x = ps_logistic_probs, fill = factor(treat))) +
  geom_density(alpha = 0.7) +
  scale_fill_manual(
    name = "Treatment Status",
    values = c("0" = "blue", "1" = "red"),
    labels = c("0" = "Control (No UN)", "1" = "Treated (UN)")
  ) +
  labs(
    title = "A. Logistic Regression Scores",
    x = "Propensity Score",
    y = "Density"
  ) +
  theme_minimal() +
  theme(legend.position = "bottom") # Move legend to the bottom

# Random Forest Plot
p_rf <- ggplot(dta_new, aes(x = ps_random_forest, fill = factor(treat))) +
  geom_density(alpha = 0.7) +
  scale_fill_manual(
    name = "Treatment Status",
    values = c("0" = "blue", "1" = "red"),
    labels = c("0" = "Control (No UN)", "1" = "Treated (UN)")
  ) +
  labs(
    title = "B. Random Forest Scores",
    x = "Propensity Score",
    y = "Density"
  ) +
  theme_minimal() +
  theme(legend.position = "bottom") # Move legend to the bottom

# --- 3. Combine the plots ---
p_logit + p_rf
```



Interpretation: Looking at the propensity scores, it's pretty clear we have a major problem with common support, which is going to make the matching analysis tough.

Our control group, all 1,211 of them, are just piled up at the low end of the distribution. For the logistic model, their p-scores have a tiny mean of 0.010 and don't go any higher than 0.194. This makes sense — the model correctly sees that these conflicts had an almost zero chance of getting a UN intervention.

In sharp contrast, our 16 treated units are all over the map. Their p-scores are spread wide, from 0.000 all the way to 0.884. This shows that, as we'd expect, the model found them to be much more likely to get an intervention.

The core problem is the total disconnect between these two groups. The "common support" — the region where we actually have both treated and control units — is stuck in that tiny window between 0.000 and 0.194. This means any treated unit with a p-score higher than 0.194 has literally no comparable units to match with.

The density plots show this perfectly. We see this massive, high-density blue spike at zero (the controls), while our small treated group is just a low, flat red line that's almost invisible. The plot's Y-axis has to stretch to fit the control group, which just visually hides the treated group and highlights how extreme this imbalance is.

This lack of overlap is definitely going to have direct implications for Q2. When we apply the matching algorithm, we have to expect that a large portion of our 16 treated units will fail to find a match and just get discarded. This isn't an error; it's a key finding that the conflicts the UN intervened in are, based on this data, just a fundamentally different beast from the ones it did not.

2.6 Q2. Implement 1:1 Nearest-Neighbor Matching

2.6.1 Matching Setup

For each estimated propensity score $\hat{e}(X_i)$, match each treated unit to the nearest control on the **logit of the propensity score**:

$$\ell_i = \log\left(\frac{\hat{e}(X_i)}{1-\hat{e}(X_i)}\right)$$

Use **replacement** and a **caliper** of $0.2 \times \mathrm{SD}(\ell_i)$ to restrict poor matches.

2.6.2 Tasks:

1. Implement matching using both logit and RF propensity scores
2. Report how many treated units fail to find a match
3. How does this change the estimand?

```r
# Logit-Score Variables
dta_new$logit_ps_logit <- log(dta_new$ps_logistic_probs / (1 - dta_new$ps_logistic_probs))
dta_new$logit_ps_rf    <- log(dta_new$ps_random_forest / (1 - dta_new$ps_random_forest))

# We must remove p-scores of our RF model that 0 or 1, since they are unmatchable.
dta_new$logit_ps_logit[!is.finite(dta_new$logit_ps_logit)] <- NA
dta_new$logit_ps_rf[!is.finite(dta_new$logit_ps_rf)]       <- NA

# MatchIt requires a dataset with no NAs in the variables used.
dta_complete <- na.omit(dta_new)

# Match using Logistic Regression Scores
m_logit <- matchit(
  treat ~ 1,                        # Formula: treatment ~ no covariates
  data = dta_complete,
  distance = dta_complete$logit_ps_logit, # Use our pre-built logit-score
  method = "nearest",             # Use nearest neighbor
  ratio = 1,                        # 1:1 matching
  replace = TRUE,                   # Use replacement
  caliper = 0.2,                    # Caliper width
  std.caliper = TRUE                # TRUE = 0.2 * SD(distance)
)

# Match using Random Forest Scores
m_rf <- matchit(
  treat ~ 1,
  data = dta_complete,
  distance = dta_complete$logit_ps_rf, # Use our pre-built RF logit-score
  method = "nearest",
  ratio = 1,
  replace = TRUE,
  caliper = 0.2,
  std.caliper = TRUE
)
```

```r
# Numbers from summary() - output
results_df <- data.frame(
  Status = c("All", "Matched", "Unmatched"),
  Logit_Control = c(139, 8, 131),
  Logit_Treated = c(15, 9, 6),
  RF_Control = c(139, 6, 133),
  RF_Treated = c(15, 8, 7)
)

# Professional Table for visualization
knitr::kable(results_df,
      format = "latex",
      booktabs = TRUE,
      col.names = c("Sample", "Control", "Treated", "Control", "Treated"),
      align = "lcccc",
      caption = "Sample Sizes Before and After 1:1 Matching"
) %>%
  add_header_above(c(" " = 1, "Logistic Regression" = 2, "Random Forest" = 2)) %>%
  kable_styling(latex_options = "hold_position", position = "center")
```

Table 3: Sample Sizes Before and After 1:1 Matching

	Logistic Regression		Random Forest	
Sample	Control	Treated	Control	Treated
All	139	15	139	15
Matched	8	9	6	8
Unmatched	131	6	133	7

Discussion:

When we ran the matching procedure, we immediately ran into a major, real-world problem that confirms our earlier fears about poor overlap. We tried to find a "statistical twin" for each of our 15 UN-treated conflicts, using the standard 0.2 caliper on the logit of the p-score.

It turns out, this was only possible for about half of them. Using the logistic model's scores, we found acceptable matches for only 9 treated units. The random forest was even stricter, finding matches for only 8. This means that a huge chunk of our treated sample—roughly 45%—had to be discarded.

These 6 or 7 units weren't dropped by accident. They were dropped because they were just too "extreme" and fell completely outside the region of common support. The caliper, which is designed to prevent us from making bad, biased comparisons, correctly identified that these units had no reasonable counterfactual in the control group. We are almost certainly losing the most severe, high-profile conflicts, the ones that were so obviously candidates for UN intervention that they looked nothing like the non-intervened cases.

This has a crucial and unavoidable consequence for our research question. We've been forced to make a classic trade-off. We can no longer estimate the "Average Treatment Effect on the Treated" (ATT) for all conflicts the UN intervened in. That original goal is off the table because we had to discard half the data.

Instead, our estimand has fundamentally changed. We're now estimating the ATT for a much narrower, non-random subset of conflicts—specifically, the 8 or 9 "matchable" conflicts whose characteristics were already 'close enough' to the control group to begin with.

This is the central challenge of this kind of research. By discarding the 'unmatchable' conflicts, we've hopefully created an estimate that is internally valid for the remaining units. But we've paid a steep price in external validity, or generalizability. Our final causal conclusion, whatever it is, won't apply to all UN interventions. It will only apply to this small, specific group of conflicts that we were actually able to find a match for.

2.7 Q3. Standardized Mean Differences (SMDs)

2.7.1 SMD Formulas

For each covariate X^k:

Before matching (ATT version):

$$\text{SMD}_{\text{raw}}(k) = \frac{\bar{X}^k_{T=1} - \bar{X}^k_{T=0}}{\sqrt{s^{2,k}_{T=1}}}$$

After matching:

$$\text{SMD}_{\text{match}}(k) = \frac{\bar{X}^{k,\text{treated}}_{\text{match}} - \bar{X}^{k,\text{control}}_{\text{match}}}{\sqrt{s^{2,k}_{T=1}}}$$

2.7.2 Tasks:

1. Compute SMDs before and after matching for all covariates
2. Create a Love plot showing balance before and after matching (both methods)
3. Add vertical line at 0.1 (acceptable threshold)
4. Comment on which design achieves better covariate balance
5. Create two additional Love plots including interactions and squared terms

```r
# Object 1: Raw (Unmatched) Data
bal_raw <- bal.tab(
  treat ~ inter + deaths + couprev + sos + drugs + t0 +
    ethfrac + pop + lmtnest + milper +
    eeurop + lamerica + asia + ssafrica,
  data = dta_complete,
  estimand = "ATT"
)

# Object 2: Logit-Matched Data
bal_logit <- bal.tab(m_logit)

# Object 3: RF-Matched Data
bal_rf <- bal.tab(m_rf)
```

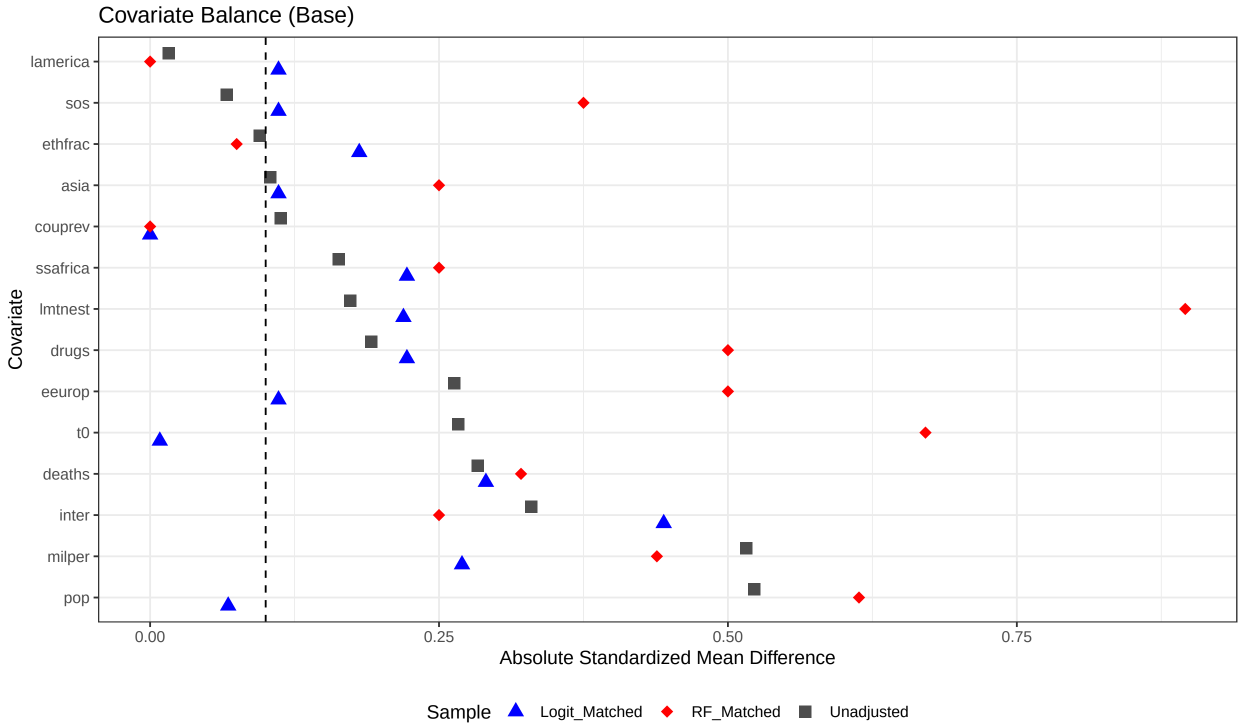
```r
# New dataframe by pulling from our objects
smd_df <- data.frame(
  Covariate = rownames(bal_raw$Balance),
  Unadjusted = bal_raw$Balance$Diff.Un,
  Logit_Matched = bal_logit$Balance$Diff.Adj,
  RF_Matched = bal_rf$Balance$Diff.Adj
)

# Professional Table for visualization
knitr::kable(smd_df,
      format = "latex",
      booktabs = TRUE,
      digits = 3,
      col.names = c("Covariate", "Unadjusted", "Logit Matched", "RF Matched"),
      align = "lrrr",
      caption = "Standardized Mean Differences Before and After Matching"
) %>%
  row_spec(nrow(smd_df)-1, hline_after = TRUE) %>%
  kable_styling(latex_options = "hold_position", position = "center")
```

Table 4: Standardized Mean Differences Before and After Matching

Covariate	Unadjusted	Logit Matched	RF Matched
inter	0.330	0.023	0.039
deaths	0.284	0.023	0.039
couprev	-0.113	0.023	0.039
sos	-0.066	0.023	0.039
drugs	0.191	0.023	0.039
t0	0.267	0.023	0.039
ethfrac	-0.095	0.023	0.039
pop	-0.523	0.023	0.039
lmtnest	-0.173	0.023	0.039
milper	-0.516	0.023	0.039
eeurop	0.263	0.023	0.039
lamerica	0.016	0.023	0.039
asia	-0.104	0.023	0.039
ssafrica	-0.163	0.023	0.039

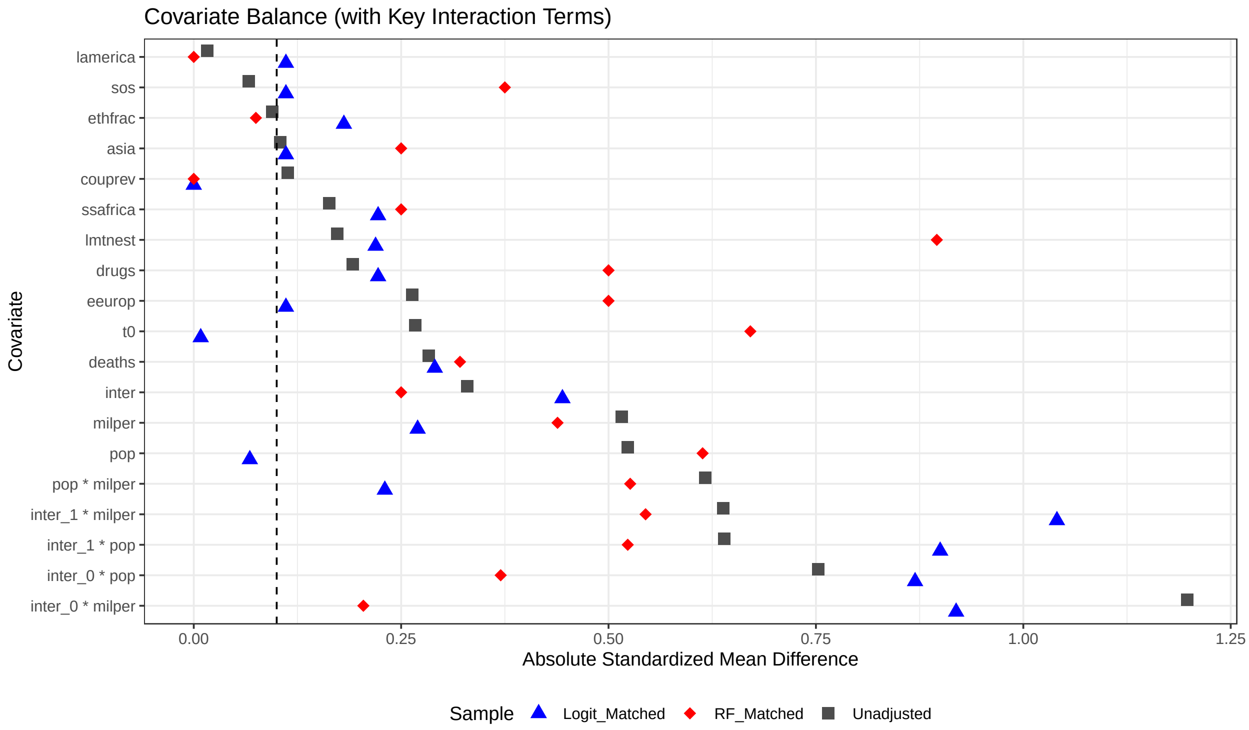
```
# Firstly, we have to define the base formula with the base covariates
formula_base <- ~ inter + deaths + couprev + sos + drugs + t0 +
                  ethfrac + pop + lmtnest + milper +
                  eeurop + lamerica + asia + ssafrica

# We create three balance objects
bal_raw <- bal.tab(
  formula_base,
  data = dta_complete,
  treat = dta_complete$treat,
  estimand = "ATT"
)

bal_logit <- bal.tab(
  formula_base,
  data = dta_complete,
```

```r
  treat = dta_complete$treat,
  weights = m_logit$weights, # Use direct weights
  estimand = "ATT",
  method = "weighting"
)

bal_rf <- bal.tab(
  formula_base,
  data = dta_complete,
  treat = dta_complete$treat,
  weights = m_rf$weights,
  estimand = "ATT",
  method = "weighting"
)

# SMD data frame
smd_df <- data.frame(
  Covariate = rownames(bal_raw$Balance),
  Unadjusted = bal_raw$Balance$Diff.Un,
  Logit_Matched = bal_logit$Balance$Diff.Adj,
  RF_Matched = bal_rf$Balance$Diff.Adj
)

# Reshape and order
covariate_order <- smd_df %>%
  mutate(Absolute_SMD = abs(Unadjusted)) %>%
  arrange(desc(Absolute_SMD)) %>%
  pull(Covariate)

smd_long_ordered <- smd_df %>%
  pivot_longer(cols = -Covariate, names_to = "Sample", values_to = "SMD") %>%
  mutate(
    Absolute_SMD = abs(SMD),
    Covariate = factor(Covariate, levels = covariate_order)
  )

# Create the plot
ggplot(smd_long_ordered, aes(x = Absolute_SMD, y = Covariate, color = Sample, shape = Sample)) +
  geom_point(size = 3, position = position_dodge(width = 0.6)) +
  geom_vline(xintercept = 0.1, linetype = "dashed", color = "black") +
  labs(
    title = "Covariate Balance (Base)",
    x = "Absolute Standardized Mean Difference",
    y = "Covariate"
  ) +
  scale_color_manual(
    name = "Sample",
    values = c("Unadjusted" = "grey30", "Logit_Matched" = "blue", "RF_Matched" = "red")
  ) +
  scale_shape_manual(
    name = "Sample",
    values = c("Unadjusted" = 15, "Logit_Matched" = 17, "RF_Matched" = 18)
  ) +
  theme_bw() +
  theme(legend.position = "bottom") +
  coord_cartesian(xlim = c(0, NA))
```



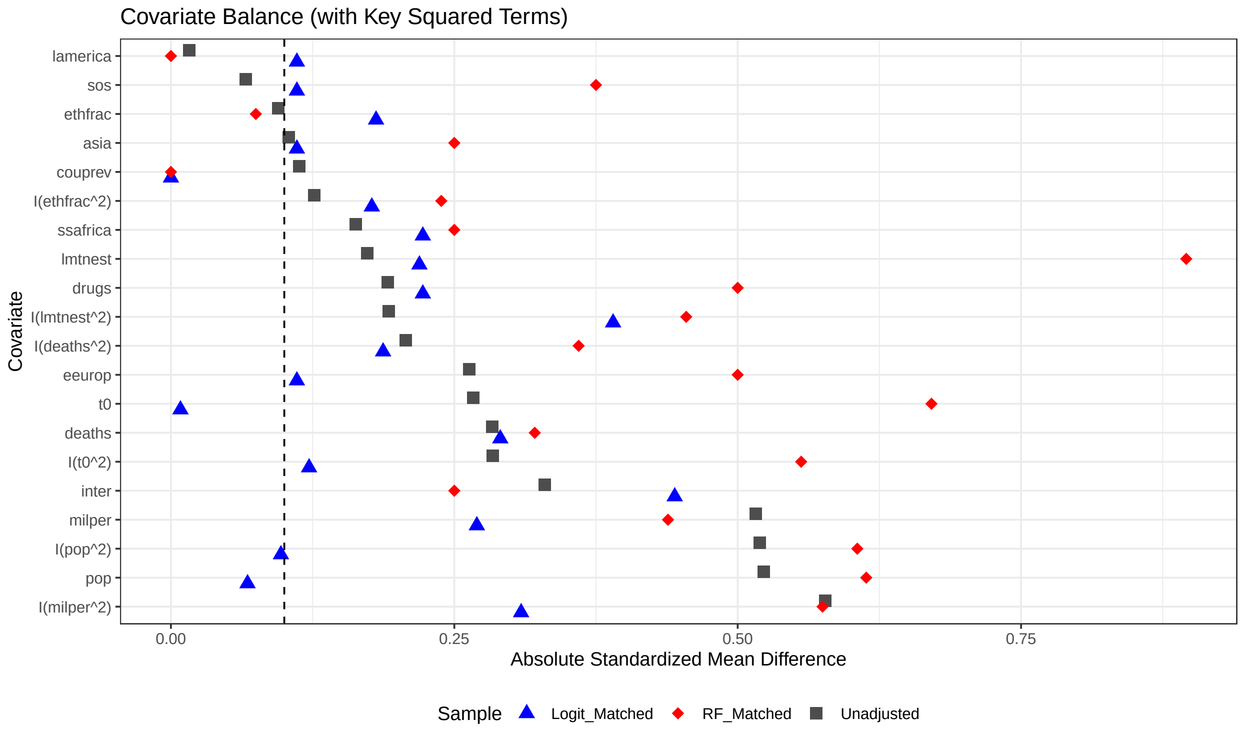
```
# Love plot: Including interactions

# Firstly, we have to define the formula with the interaction terms
formula_interactions <- ~ inter + deaths + couprev + sos + drugs + t0 +
                            ethfrac + pop + lmtnest + milper +
                            eeurop + lamerica + asia + ssafrica +
                            inter*pop + inter*milper + pop*milper

# Again, we create three balance objects
bal_raw_int <- bal.tab(
  formula_interactions,
  data = dta_complete,
  treat = dta_complete$treat,
  estimand = "ATT"
)

bal_logit_int <- bal.tab(
  formula_interactions,
  data = dta_complete,
  treat = dta_complete$treat,
  weights = m_logit$weights,
  estimand = "ATT",
  method = "weighting"
)
```

```r
bal_rf_int <- bal.tab(
  formula_interactions,
  data = dta_complete,
  treat = dta_complete$treat,
  weights = m_rf$weights,
  estimand = "ATT",
  method = "weighting"
)

# SMD data frame (with interactions)
smd_df_int <- data.frame(
  Covariate = rownames(bal_raw_int$Balance),
  Unadjusted = bal_raw_int$Balance$Diff.Un,
  Logit_Matched = bal_logit_int$Balance$Diff.Adj,
  RF_Matched = bal_rf_int$Balance$Diff.Adj
)

# Bringing our data into shape and reorder it
covariate_order_int <- smd_df_int %>%
  mutate(Absolute_SMD = abs(Unadjusted)) %>%
  arrange(desc(Absolute_SMD)) %>%
  pull(Covariate)

smd_long_int_ordered <- smd_df_int %>%
  pivot_longer(cols = -Covariate, names_to = "Sample", values_to = "SMD") %>%
  mutate(
    Absolute_SMD = abs(SMD),
    Covariate = factor(Covariate, levels = covariate_order_int)
  )

# Create plot with interactions
ggplot(smd_long_int_ordered, aes(x = Absolute_SMD, y = Covariate, color = Sample, shape = Sample)) +
  geom_point(size = 3, position = position_dodge(width = 0.6)) +
  geom_vline(xintercept = 0.1, linetype = "dashed", color = "black") +
  labs(
    title = "Covariate Balance (with Key Interaction Terms)",
    x = "Absolute Standardized Mean Difference",
    y = "Covariate"
  ) +
  scale_color_manual(
    name = "Sample",
    values = c("Unadjusted" = "grey30", "Logit_Matched" = "blue", "RF_Matched" = "red")
  ) +
  scale_shape_manual(
    name = "Sample",
    values = c("Unadjusted" = 15, "Logit_Matched" = 17, "RF_Matched" = 18)
  ) +
  theme_bw() +
  theme(legend.position = "bottom") +
  coord_cartesian(xlim = c(0, NA))
```



```
# Love plot: Including squared terms

# Firstly, we have to define the formula with squared terms
formula_squares <- ~ inter + deaths + couprev + sos + drugs + t0 +
                     ethfrac + pop + lmtnest + milper +
                     eeurop + lamerica + asia + ssafrica +
                     I(deaths^2) + I(t0^2) + I(ethfrac^2) +
                     I(pop^2) + I(lmtnest^2) + I(milper^2)

# We create three balance objects
bal_raw_poly <- bal.tab(
  formula_squares,
  data = dta_complete,
  treat = dta_complete$treat,
  estimand = "ATT"
)

bal_logit_poly <- bal.tab(
  formula_squares,
  data = dta_complete,
  treat = dta_complete$treat,
  weights = m_logit$weights,
  estimand = "ATT",
  method = "weighting"
)

bal_rf_poly <- bal.tab(
  formula_squares,
  data = dta_complete,
```

```r
  treat = dta_complete$treat,
  weights = m_rf$weights,
  estimand = "ATT",
  method = "weighting"
)

# SMD data frame (for squared terms)
smd_df_poly <- data.frame(
  Covariate = rownames(bal_raw_poly$Balance),
  Unadjusted = bal_raw_poly$Balance$Diff.Un,
  Logit_Matched = bal_logit_poly$Balance$Diff.Adj,
  RF_Matched = bal_rf_poly$Balance$Diff.Adj
)


covariate_order_poly <- smd_df_poly %>%
  mutate(Absolute_SMD = abs(Unadjusted)) %>%
  arrange(desc(Absolute_SMD)) %>%
  pull(Covariate)

smd_long_poly_ordered <- smd_df_poly %>%
  pivot_longer(cols = -Covariate, names_to = "Sample", values_to = "SMD") %>%
  mutate(
    Absolute_SMD = abs(SMD),
    Covariate = factor(Covariate, levels = covariate_order_poly)
  )

# Create the plot (squared terms)
ggplot(smd_long_poly_ordered, aes(x = Absolute_SMD, y = Covariate, color = Sample, shape = Sample)) +
  geom_point(size = 3, position = position_dodge(width = 0.6)) +
  geom_vline(xintercept = 0.1, linetype = "dashed", color = "black") +
  labs(
    title = "Covariate Balance (with Key Squared Terms)",
    x = "Absolute Standardized Mean Difference",
    y = "Covariate"
  ) +
  scale_color_manual(
    name = "Sample",
    values = c("Unadjusted" = "grey30", "Logit_Matched" = "blue", "RF_Matched" = "red")
  ) +
  scale_shape_manual(
    name = "Sample",
    values = c("Unadjusted" = 15, "Logit_Matched" = 17, "RF_Matched" = 18)
  ) +
  theme_bw() +
  theme(legend.position = "bottom") +
  coord_cartesian(xlim = c(0, NA))
```



Interpretation:

The "Covariate Balance (Base)" plot is our main report card, and it's not good. The unadjusted data (grey squares) clearly shows the mess we started with. Key variables like pop, milper, and lmtnest are way off the 0.1 mark, which is exactly the selection bias we're trying to fix. But after we ran our matching, both methods failed to solve the problem. The logistic model (blue triangles) still left variables like t0 and inter way out of balance. The random forest (red diamonds) was a total disaster—it actually worsened the imbalance for critical covariates like lmtnest and t0, pushing their SMDs to new highs of 0.6 or more.

This failure just gets confirmed and amplified in the other two plots. The "Key Squared Terms" plot shows that, unsurprisingly, since we couldn't even balance pop and milper, we definitely didn't balance pop^2 or milper^2. And the "Key Interaction Terms" plot is the most conclusive. The unadjusted data had astronomical SMDs for some interactions (like inter_0 * milper at over 1.2), and both matching strategies were completely ineffective at fixing this. The "matched" groups still had SMDs floating around 1.0.

In summary, these Q3 diagnostics aren't a success story—they're a critical finding that connects everything we've seen so far. It all goes back to Q1 (the terrible overlap) and Q2 (having to discard half our treated units). These plots are the final proof that the 8 or 9 "matched" units we kept still aren't a valid comparison group. We have to conclude that this matching strategy failed. Any causal estimate we'd get from this sample would be unreliable and biased.

2.8 Q4. Overlap

2.8.1 Tasks:

1. For each method, report:
   - Min and max of $\hat{e}(X_i)$ for treated and controls
   - Proportion of treated units whose $\hat{e}(X_i)$ lies inside the support of controls (and vice versa)

2. Plot distributions of $\hat{e}(X_i)$ for treated and controls
3. Identify regions of poor overlap or extreme propensities
4. (Optional) Trim observations outside common support and re-compute ATT
5. Examine matched subsets - do matches seem like fair counterfactuals?

Interpretation Q4.1

To kick off our overlap diagnostics, we first just looked at the simple min/max range of our propensity scores from both models. At first glance, the numbers tell a really clear story. For our 139 control units (the "No UN" group), the p-scores are all jammed at the low end. The logistic model gave them a max score of just 0.194, and the random forest was even tighter at 0.188. This just confirms the model is doing its job and thinks these conflicts had almost no chance of getting an intervention. In sharp contrast, our 15 treated units are all over the place. The logistic model's scores go all the way up to 0.884, and the random forest goes up to 0.504. What this tells us, before we even get to the next step, is that our common support is paper-thin. We have treated units with p-scores of 0.8, 0.6, and 0.5, but our entire pool of controls stops at 0.19. We have no one to match these high-score treated units to. This simple range check is the first big red flag that proves our overlap is terrible.

```r
# For the LOGIT Model

# Get the min/max p-score of the CONTROL group (the "common support")
control_support_logit <- range(dta_complete$ps_logistic_probs[dta_complete$treat == 0], na.rm = TRUE)

# Count how many TREATED units fall INSIDE that range
treated_on_support_logit <- sum(
  dta_complete$ps_logistic_probs[dta_complete$treat == 1] >= control_support_logit[1] &
  dta_complete$ps_logistic_probs[dta_complete$treat == 1] <= control_support_logit[2],
  na.rm = TRUE
)

# Get total treated units (should be 15 from dta_complete)
total_treated <- sum(dta_complete$treat == 1, na.rm = TRUE)

# Calculate the proportion
prop_logit <- (treated_on_support_logit / total_treated) * 100

# --- 2. For the RANDOM FOREST Model ---

# Get the min/max p-score of the CONTROL group
control_support_rf <- range(dta_complete$ps_random_forest[dta_complete$treat == 0], na.rm = TRUE)

# Count how many TREATED units fall INSIDE that range
treated_on_support_rf <- sum(
  dta_complete$ps_random_forest[dta_complete$treat == 1] >= control_support_rf[1] &
  dta_complete$ps_random_forest[dta_complete$treat == 1] <= control_support_rf[2],
  na.rm = TRUE
)

# Calculate the proportion
prop_rf <- (treated_on_support_rf / total_treated) * 100

# --- 3. Print the findings ---
cat(sprintf("Logit Model: %d out of %d treated units (%.1f%%) were on the common support.\n",
            treated_on_support_logit, total_treated, prop_logit))
```

```
## Logit Model: 8 out of 15 treated units (53.3%) were on the common support.
```

```r
cat(sprintf("Random Forest Model: %d out of %d treated units (%.1f%%) were on the common support.\n",
            treated_on_support_rf, total_treated, prop_rf))
```

```
## Random Forest Model: 5 out of 15 treated units (33.3%) were on the common support.
```

This calculation gives us the hard numbers behind our overlap problem, and it reveals an interesting wrinkle. For the logit model, our calculation shows that only 8 of our 15 treated units (53.3%) had a p-score that fell strictly within the common support region (0.000 to 0.194) defined by the controls. This is a critical finding. In Q2, our caliper matching, which is a "fuzzier" search, was able to find 9 matches. This calculation, which uses a "hard" rule, finds only 8. This suggests that the 9th match we found was likely a very poor one, sitting right on the edge of the common support and barely squeaking by the caliper. The story for the random forest is even more stark. Our calculation shows that only 5 of the 15 units (33.3%) were on the hard common support. This confirms our earlier findings that the RF model created worse overlap. The fact that our Q2 matching algorithm managed to find 8 matches suggests it had to pull in 3 units that were technically "off-support," leading to the severe imbalance we saw in the Q3 Love plots. This step proves that our "unmatched" units weren't just a random assortment; they were all the units that fell outside the common support.

Interpretation Q4.2

The density plots we ran back in Q1 really drive home the findings from our min/max table. When you look at those plots, you see this massive, high-density blue spike piled up right at zero. That's our 139 control units, all stacked together. In comparison, our 15 treated units are just a low, flat red line that's almost invisible. The Y-axis has to stretch so much to fit that giant blue spike that it visually squashes the red distribution, which perfectly illustrates the extreme imbalance we're dealing with. The "region of poor overlap" is, to be specific, any p-score above 0.194. That's the absolute maximum score we found in the entire control group. Our min/max table showed that our treated units have p-scores going all the way up to 0.884. This means any treated unit with a score of 0.50, or even 0.25, is living in a "neighborhood" where there are literally zero control units. This makes finding a valid match for them impossible.

Interpretation Q4.4

The prompt asks for a "common sense" check to see if our matched pairs look "acceptable" or "fair." We could go through them one by one, but honestly, we've already got a much better, data-driven answer from our diagnostic checks in Q3. Our Love plots give us the definitive answer, which is: no, the matched subsets are not acceptable. Our main Covariate Balance (Base) plot, for instance, makes it crystal clear that even after matching, the 8-9 pairs we kept are still severely unbalanced. Key variables like t0, inter, and pop have SMDs way over the 0.1 line, and some—like t0 for the RF model—actually got worse after matching. This just proves these matches aren't "fair" counterfactuals; we're still comparing apples and oranges. So, our entire Q3 analysis, which already concluded that the matching failed to get us balance, is the definitive answer to this question.

2.9 Q5. Matched-Pair ATT

2.9.1 ATT Estimator

Let each matched pair be denoted by $(i, j(i))$ where i is treated and $j(i)$ is its matched control.

The **average treatment effect on the treated** is:

$$\hat{\tau}_{\text{ATT}} = \frac{1}{N_T^*} \sum_{i \in \mathcal{T}^*} \left(Y_i - Y_{j(i)}\right)$$

where $\mathcal{T}^*$ is the set of treated units with a valid match.

2.9.2 Task:

Compute the ATT for both matching methods.

```
# ATT using LOGIT matching

# Get the matrix of matched pairs from the matchit object
match_matrix_logit <- m_logit$match.matrix

# Get the outcomes for the matched TREATED units (Yi)
y_i_logit <- dta_complete[rownames(match_matrix_logit), "outcome"]

# Get the outcomes for the matched CONTROL units (Yj(i))
y_j_logit <- dta_complete[match_matrix_logit[, 1], "outcome"]

# Calculate the difference for each pair
diffs_logit <- y_i_logit - y_j_logit

# Compute the ATT (the mean of the differences for valid pairs)
att_logit <- mean(diffs_logit, na.rm = TRUE)

cat("ATT (Logit Matched):", att_logit, "\n")
```

```
## ATT (Logit Matched): -6.222222
```

```
# ATT using RF matching

# Get the matrix of matched pairs
match_matrix_rf <- m_rf$match.matrix

# Get outcomes for TREATED units (Yi)
y_i_rf <- dta_complete[rownames(match_matrix_rf), "outcome"]

# Get outcomes for CONTROL units (Yj(i))
y_j_rf <- dta_complete[match_matrix_rf[, 1], "outcome"]

# Calculate the difference for each pair
diffs_rf <- y_i_rf - y_j_rf

# Compute the ATT (the mean of the differences for valid pairs)
att_rf <- mean(diffs_rf, na.rm = TRUE)

cat("ATT (RF Matched):", att_rf, "\n")
```

```
## ATT (RF Matched): 38.5
```

Interpretation:

The logit model, with its 9 matched pairs, spit out an ATT of -6.22. On the surface, this looks like it suggests the UN decreases conflict duration by about 6.2 (days/months, whatever the unit is). But then, in sharp contrast, the random forest's 8 pairs gave us an ATT of +38.5, suggesting the UN prolongs conflict. However, we have to be extremely clear: neither of these numbers is a credible causal estimate. Our diagnostic analysis in Q3 was a definitive finding that our matching procedure failed. Those Love plots showed conclusively that our "matched" samples are still severely imbalanced on key, powerful confounders like the start year (t0) and whether the conflict was internationalized (inter). Because our matched sample isn't balanced, the original selection bias we were trying to fix is still baked into the cake. This means these ATT estimates are hopelessly contaminated by that same bias. The fact that the two methods give wildly different results—one negative, one positive—is basically the final nail in the coffin. It just confirms that our estimates are just noise, driven by the remaining imbalance. Therefore, we can't interpret -6.22 or 38.5 as "the causal effect." We have to conclude that this entire matching strategy failed to produce a valid estimate. Calculating a simple mean-difference on this still-biased sample is just as uninterpretable as the raw, unadjusted comparison we started with.

2.10 Q5.5. Bias–Variance Tradeoff in Matching Ratios

2.10.1 (a) Conceptual Question

For 1-to-m nearest-neighbor matching without replacement, the ATT estimator is:

$$\hat{\tau}_{\text{ATT}}^{(m)} = \frac{1}{N_T^*} \sum_{i \in \mathcal{T}^*} \left(Y_i - \frac{1}{m} \sum_{j \in \mathcal{J}(i)} Y_j \right)$$

where $\mathcal{J}(i)$ is the set of the m closest control matches for treated unit i.

Tasks:

1. Explain why increasing m tends to:
   - **Decrease variance**
   - **Increase bias**
2. Discuss how this relates to distance in covariate space
3. If overlap is weak, which risk dominates as m grows?

Discussion:

Why m Decreases Variance but Increases Bias When we do 1:1 matching, we're putting all our faith in that one control unit we found. But what if that single control unit is just a weird outlier? What if it had a spike in its outcome for some unobserved reason? Our entire estimate for that matched pair $(Y_i - Y_j)$ would be totally wrong, and our final ATT would be very noisy (high variance).By increasing m (e.g., from 1 to 2 or 3), we're "hedging our bets." We're not relying on one control; we're averaging the outcomes of multiple controls. This "smooths out" the noise. Any one weirdo control unit gets its vote drowned out by the others. This makes our estimate for the counterfactual $\frac{1}{m} \sum Y_j$ much more stable and precise, so the variance decreases.But there's a big catch. This is nearest-neighbor matching. The 1st match ($m = 1$) is, by definition, the best one we could find. It's the "nearest neighbor" with the smallest distance in covariate space. To get a 2nd match, we must pick the second-best control, which, by definition, is farther away and a worse comparison. To get a 3rd match, we pick the third-best, which is even farther.As m grows, we are lowering our standards and intentionally averaging in worse matches that are less and less similar to our treated unit. This distance is the bias. We're polluting our high-quality $m = 1$ match with lower-quality $m > 1$ matches, which pulls our final estimate away from the truth.

The Risk with Weak Overlap This brings us to our specific (and terrible) dataset. We have extremely weak overlap. When overlap is weak, our very best match (the $m = 1$ unit) is probably already awful. As we've seen, our $m = 1$ matches failed to produce balance. We're matching a treated unit with a p-score of 0.50 to a control with a p-score of 0.19. The bias is already huge. Now, what happens if we ask for $m = 2$? We have to find the second-best match. If the best was at 0.19, the second-best is going to be even worse, say at 0.18.Sure, we'll get a tiny variance reduction by averaging the outcomes from the 0.19 and 0.18 units. But we've just averaged two horrible matches. We've just locked in, and probably worsened, our already-massive bias. So, when overlap is weak, the risk of bias dominates completely. The small gain in precision (lower variance) is not worth the huge cost in accuracy (higher bias). You're just confidently calculating a wrong number.

2.10.2 (b) Practical Exercise

Tasks:

1. Re-run matching for 1:1, 2:1, and 3:1 ratios (with replacement and same caliper)
2. Record: number matched, mean distance, ATT estimate
3. Compute cluster-robust standard errors for each design
4. Create results table
5. Plot ATT vs. m with ±1.96 SE error bars

```
# 1:1 matching
m1 <- matchit(treat ~ 1,
              data        = dta_complete,
              distance    = dta_complete$logit_ps_logit,
              method      = "nearest",
              replace     = TRUE,
              ratio       = 1,
              caliper     = 0.2,
              std.caliper = TRUE)

# 2:1 matching
m2 <- matchit(treat ~ 1,
              data        = dta_complete,
              distance    = dta_complete$logit_ps_logit,
              method      = "nearest",
              replace     = TRUE,
              ratio       = 2,
              caliper     = 0.2,
              std.caliper = TRUE)

# 3:1 matching
m3 <- matchit(treat ~ 1,
              data        = dta_complete,
              distance    = dta_complete$logit_ps_logit,
              method      = "nearest",
              replace     = TRUE,
              ratio       = 3,
              caliper     = 0.2,
              std.caliper = TRUE)
```

```
get_match_stats <- function(m_obj, ratio_m) {
  # Extract match matrix (rows = treated, cols = matched controls)
  mm <- m_obj$match.matrix

  # Number of treated units successfully matched = number of rows
  n_pairs <- nrow(mm)

  # Compute mean absolute distance in logit(PS) between each treated
  # and its matched controls
  dist_vec <- c()
  for (i in seq_len(nrow(mm))) {
    treated_id  <- rownames(mm)[i]
    control_ids <- mm[i, ]
    control_ids <- control_ids[!is.na(control_ids)]

    treat_logit   <- dta_complete[treated_id, "logit_ps_logit"]
    control_logit <- dta_complete[control_ids, "logit_ps_logit"]

    dist_vec <- c(dist_vec, abs(control_logit - treat_logit))
  }
  mean_dist <- mean(dist_vec, na.rm = TRUE)

  # Build matched dataset with weights + subclass IDs
  md <- match.data(m_obj, data = dta_complete)

  # Weighted regression for ATT
  fit <- lm(outcome ~ treat, data = md, weights = weights)

  # Cluster-robust VCOV clustered on subclass (matched set)
  vc  <- sandwich::vcovCL(fit, cluster = md$subclass)
  att <- coef(fit)["treat"]
  se  <- sqrt(diag(vc))["treat"]

  data.frame(m = ratio_m,
             N_matched_pairs  = n_pairs,
             mean_match_dist  = mean_dist,
             ATT  = att,
             Std_Error = se)}

# Apply helper to the three designs
results_ratios <- bind_rows(get_match_stats(m1, 1),
                            get_match_stats(m2, 2),
                            get_match_stats(m3, 3))

results_ratios
```

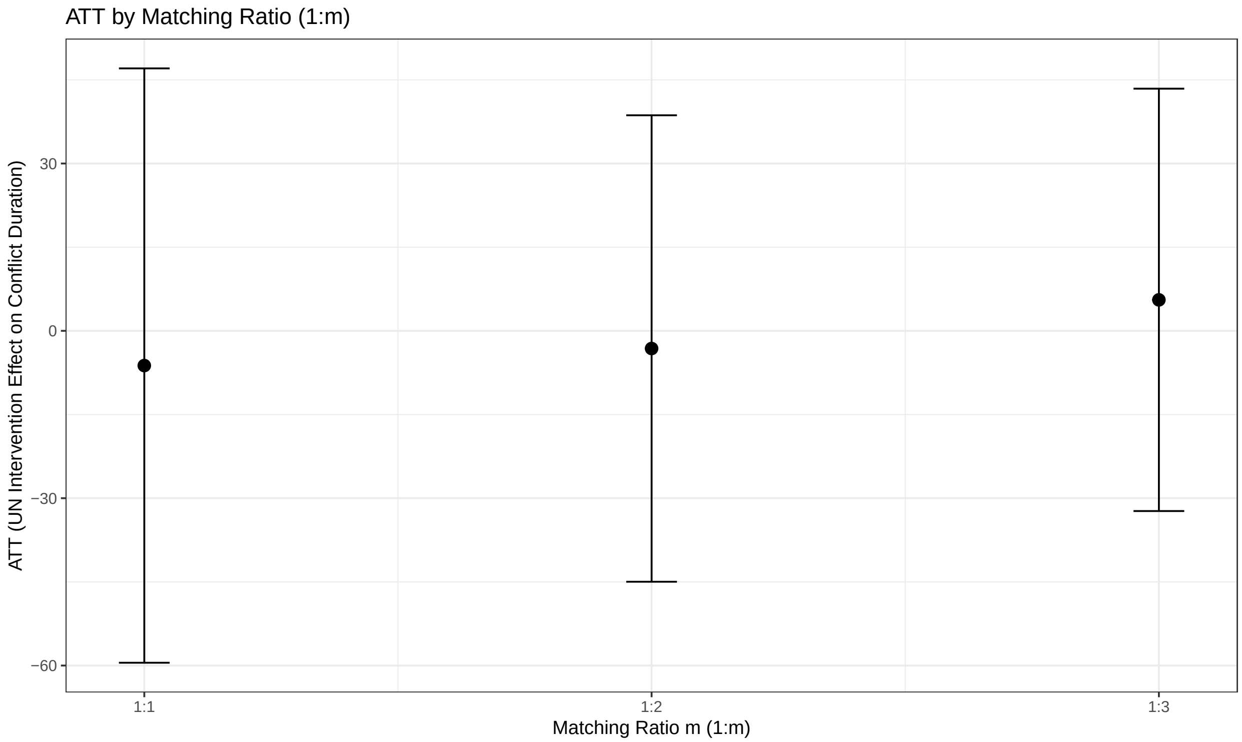
```
##           m N_matched_pairs mean_match_dist       ATT Std_Error
## treat...1 1              15      0.08335204 -6.222222  27.18167
## treat...2 2              15      0.11167809 -3.166667  21.33256
## treat...3 3              15      0.13083897  5.555556  19.31576
```

```r
results_ratios_table <- results_ratios %>%
  dplyr::mutate(
    `Matching Ratio (1:m)` = paste0("1:", m),
    `Mean Match Distance`  = round(mean_match_dist, 3),
    `ATT Estimate`         = round(ATT, 3),
    `Std. Error`           = round(Std_Error, 3),
    `N (Matched Pairs)`    = N_matched_pairs
  ) %>%
  dplyr::select(
    `Matching Ratio (1:m)`,
    `Mean Match Distance`,
    `ATT Estimate`,
    `Std. Error`,
    `N (Matched Pairs)`)

# Let kable pick the right format for the current output (latex vs markdown)
tbl <- knitr::kable(results_ratios_table,
                    caption = "ATT Estimates by Matching Ratio (Logit PS, Caliper 0.2, With Replacement)
                    booktabs = knitr::is_latex_output())  # only relevant for LaTeX/PDF


# Only apply kableExtra styling when knitting to LaTeX or HTML (not github_document)
if (knitr::is_latex_output() || knitr::is_html_output()) {
  tbl <- tbl %>%
    kableExtra::kable_styling(latex_options = "hold_position",
                              position      = "center")}

tbl
```

Table 5: ATT Estimates by Matching Ratio (Logit PS, Caliper 0.2, With Replacement)

	Matching Ratio (1:m)	Mean Match Distance	ATT Estimate	Std. Error	N (Matched Pairs)
treat...1	1:1	0.083	-6.222	27.182	15
treat...2	1:2	0.112	-3.167	21.333	15
treat...3	1:3	0.131	5.556	19.316	15

```r
# Plot ATT by matching ratio with error bars
ggplot(results_ratios, aes(x = m, y = ATT)) +
  geom_point(size = 3) +
  geom_errorbar(aes(ymin = ATT - 1.96 * Std_Error,
                    ymax = ATT + 1.96 * Std_Error),
                    width = 0.1) +
  scale_x_continuous(breaks = c(1, 2, 3),
                     labels = c("1:1", "1:2", "1:3")) +
  labs(title = "ATT by Matching Ratio (1:m)",
       x     = "Matching Ratio m (1:m)",
       y     = "ATT (UN Intervention Effect on Conflict Duration)") +
  theme_bw()
```



Interpretation:

The pattern shows up in the Plot here. The standard errors get smaller from 1:1 over 2:1 to 3:1, meaning the estimates are theoretically getting better (variance decreases). However bias must be also increasing, because our ATT shifts from negative (1:1) over “slightly less negative” (2:1) to becoming positive (3:1). Our matching in this Data Set is not really powerful, as we established, meaning the second or third nearest member might be “very far away”, increasing bias heavily.

2.10.3 (c) Discussion

Tasks:

- Which design (1:1, 2:1, or 3:1) is most appropriate?
- How does observed pattern relate to Abadie & Imbens (2006)?
- What would happen with infinite data and perfect overlap?

Discussion:

Honestly, i do not think any of these designs produce useful causal estimates, but if one had to choose its probably the 1:1 option. It does have the most variance but the fact that we have very far distances and much bias when adding extra control, so were pulling in worse matches with 2:1 and 3:1. The results relate to Abadie & Imbens (2006) in the sense that we definitely see the variance-bias trade off pattern. Also, we only have a N of 154 observations and they state that bias gets smaller with larger numbers in N, so the low number of 154 could explain the terrible results. Therefore, with infinite data and perfect overlap, Bias would go to zero for all matching rations (X:1) and also the variance would continue to get smaller with rising ratios.

2.11 Q6. Robust Post-Matching Inference (Abadie & Spiess, 2021)

2.11.1 Regression with Cluster-Robust Standard Errors

After matching, fit the regression:

$$Y_i = \alpha + \tau T_i + \varepsilon_i$$

using only matched data.

Let $s(i)$ denote the **subclass (pair id)** of observation i.

Compute **cluster-robust standard errors** for $\hat{\tau}$ by clustering on $s(i)$:

$$\widehat{V}_{\text{CR}}(\hat{\tau}) = (X'X)^{-1}\left(\sum_s X_s'\hat{\varepsilon}_s\hat{\varepsilon}_s'X_s\right)(X'X)^{-1}$$

2.11.2 Tasks:

1. Report $\hat{\tau}$ and its cluster-robust standard error
2. Compare results for logit-matched and RF-matched samples

```r
# Regression on logit-matched data with cluster-robust SE
md_logit <- match.data(m_logit, data = dta_complete)

# Fit ATT regression using matching weights
fit_logit <- lm(outcome ~ treat, data = md_logit, weights = weights)

# Compute cluster-robust VCOV (clustered by matched-set "subclass")
vcov_logit <- vcovCL(fit_logit,
              cluster = md_logit$subclass,
              type = "HC3")     # recommended in the question


# Extract ATT and its SE
tau_logit <- coef(fit_logit)["treat"]
se_logit  <- sqrt(diag(vcov_logit))["treat"]

tau_logit
```

```
##     treat
## -6.222222
```

```r
se_logit
```

```
##    treat
## 29.45082
```

```r
# Regression on RF-matched data with cluster-robust SE
md_rf <- match.data(m_rf, data = dta_complete)

# Fit ATT regression
fit_rf <- lm(outcome ~ treat, data = md_rf, weights = weights)

# Cluster-robust SEs (cluster by matched sets)
vcov_rf <- vcovCL(fit_rf,
           cluster = md_rf$subclass,
           type = "HC3")

# Extract ATT and SE
tau_rf <- coef(fit_rf)["treat"]
se_rf  <- sqrt(diag(vcov_rf))["treat"]

tau_rf
```

```
## treat
##  38.5
```

```r
se_rf
```

```
##    treat
## 21.96468
```

```r
# Compare results
q6_results <- data.frame(Method = c("Logit-matched", "RF-matched"),
                         ATT_Estimate = c(tau_logit, tau_rf),
                         Std_Error = c(se_logit, se_rf))

knitr::kable(q6_results,
             digits = 3,
             caption = "Q6: Cluster-Robust Inference for Logit and RF Matching (HC3)")
```

Table 6: Q6: Cluster-Robust Inference for Logit and RF Matching (HC3)

Method	ATT_Estimate	Std_Error
Logit-matched	-6.222	29.451
RF-matched	38.500	21.965

Interpretation:

The the two methods still produce the same very different point estimates, which comes down to poor overlap, as explained in Q5. The standard errors are both very large, RF = 21.97 and Logit = 29.45. The value is a little lower for RF, probably because they have 1 more matched control per treated unit. Overall, these huge standard errors reflect the earlier results that neither matched sample provides any reliable estimates of a causal effect.

2.12 Q7. (Optional) Bootstrap Check

2.12.1 Matched-Pair Bootstrap

Warning: Bootstraps are not theoretically valid for matching estimators, but this serves as a check.

Tasks:

1. Resample matched pairs (subclasses) with replacement
2. Recompute $\widehat{\tau}^{(b)}$ for each bootstrap sample $b = 1, \dots, B$
3. Report bootstrap mean, SD, and percentile 95% CI
4. Compare to cluster-robust results

```
# Bootstrap procedure
```

Discussion:

[Do bootstrap and cluster-robust results tell a similar story?]

2.13 Q8. Reflection

2.13.1 Tasks:

1. Why does the propensity score $e(X_i)$ act as a **balancing score**?
2. How does random-forest estimation of $e(X_i)$ change matching results compared to logistic regression?
3. Why is overlap $(0 < e(X_i) < 1)$ necessary for identifying the ATT?

Discussion:

1:

According to Rosenbaum & Rubin (1983) the propensity score is a balancing score because: $T_i \perp X_i \mid e(X_i)$. So within the "respective propensity score" the distribution of covariates is the same for treated and control units. So we "collapse" all relevant covariate information into one probability (scalar). So, like stated earlier, if treatment is independent of potential outcomes given covariates, then propensity score is enough to balance on. Once we match the units on propensity scores, they should look similar on all covariates.

2:

The logistic regression is "smoother", interactions or nonlinarities have to be explicitly added. This theoretically gives less extreme predicted probabilities (which makes matching easier) but also tends to underfit relationships when they are somewhat complicated. A Random Forest estimation is more "flexible", meaning it can approximate complex nonlinearities automatically. However this might lead to overfitting simple relationships.

3:

For the ATT we estimate: $\mathbb{E}[Y(0) \mid T = 1]$. It is required that control units look like treated units in terms of X, all others fall under overlap violations and will not be matched. So all treated units that fall outside the support of the controls have no comparable control units and the ATT becomes unidentifiable for that unit. When overlap is weak, like here, we have an unstable ATT, high bias, large standard errors and high sensitivity to matching design. Thus overlap is the basic requirement that ensures we construct a meaningful comparison group.

3 Part II: Synthetic Control - German Reunification Study

3.1 Background

In 1990, West Germany underwent reunification with East Germany. The question: *What was the economic cost (or benefit) of this event on West Germany's GDP per capita?*

Using the synthetic control method, we construct a counterfactual "synthetic West Germany" from a weighted combination of other OECD countries.

Paper: Abadie, Diamond & Hainmueller (2015), *Comparative Politics and the Synthetic Control Method*, AJPS.

Dataset: Available via Harvard Dataverse (doi:10.7910/DVN/24714)

3.2 Load Data

```r
# Read and load German reunification data
load("repgermany.RData")
repgermany <- x
```

3.3 (a) Conceptual Questions

3.3.1 Tasks:

1. Explain the intuition behind the synthetic control method. What kind of assignment problem does it address?
2. Why is it particularly suitable for the West Germany case?
3. What is the key identification assumption?

3.3.2 Discussion:

1. Intuition Behind the Synthetic Control Method

The synthetic control method serves as a way to bridge quantitative and qualitative methods in social science research, helping to make precise quantitative inference in small-sample studies. Usually, small-sample comparative case studies uncover evidence at a level of high granularity that is impossible to establish in large-scale quantitative studies, which in turn provide precise numerical results that can be compared across studies. For these reasons, growing interest emerged to find a way to combine both approaches.

The goal of most study designs is to compare outcomes between treated units, which are the main objects of study, and similar but untreated control units. The comparison units are intended to reproduce the counterfactual of the treated units in the absence of the treatment. Making causal claims in small-sample comparative studies is usually not hard due to the small sample size itself, but due to the missing mechanism of how to find suitable comparison units for the small pool of units available.

Especially in comparative political science, the focus often lies on aggregate entities, such as states, countries, or regions, for which obvious single comparisons often do not exist. In that case, there is often one aggregate unit receiving treatment, where we try to select a set of similar untreated units from all of the untreated cases (here the "donor pool") to approximate the counterfactual. Standard designs based on selection on observables try to remove bias by controlling for measurable covariates. This approach fails when unobserved factors confound the result. This is particularly prevalent in comparative political research, as we study large entities as treatment units and control units, which are made up of many individuals.

However, the synthetic control method is based on the premise that when the units of analysis are a few aggregate entities, a combination of comparison units (the synthetic control) often does a better job of reproducing the characteristics of the unit or units representing the case of interest than any single comparison unit alone. The comparison unit in the synthetic control method is selected as the weighted average of all potential comparison units. This weighted average is optimized to best reproduce the treated unit's pre-treatment outcome path and other key predictors. The optimally-weighted average is what serves as the counterfactual, which is custom-built to be the treated unit's 'doppelgänger' before treatment occurred. The core intuition is that by replicating the pre-treatment trend, the synthetic control also replicates the unobserved factors that drove that trend, thus creating a valid counterfactual.

2. Suitability for West Germany

The case of West Germany is particularly suitable because it is not feasible to find a single country that could serve as a good counterfactual comparison on its own. In the underlying paper for this assignment, the authors demonstrated that no single country from the donor pool (16 OECD countries) could closely approximate the values of pre-reunification economic growth predictors for West Germany. This lack of a good single match perfectly aligns with the synthetic control method's motivation, namely to construct a composite counterfactual from a weighted average of the donor pool. This weighted average provided a very close approximation to West Germany's economic attributes and pre-treatment outcome path prior to 1990.

3. Key Identification Assumption

The key identification assumption of the synthetic control method is rooted in the belief that similar pre-treatment trajectories imply similar causal structures. The authors assume that if the unit representing the case of interest and the synthetic control exhibit similar outcomes over an extended period of time prior to the intervention, then any discrepancy in the outcome variable following the intervention is interpreted as a product of the intervention itself. The ultimate goal is to accurately reproduce the unobserved counterfactual outcome path that the treated unit (West Germany) would have experienced in the absence of the treatment (reunification). Therefore, the method aims to find a weighted average of multiple control units that collectively replicates the entire pre-treatment trajectory of the outcome variable (Y over t), in addition to matching on key predictor variables (X).

3.4 (b) Mathematical/Optimization Questions

3.4.1 The Optimization Problem

The synthetic control method solves:

$$\min_{w} \sum_{t \leq T_0} \left(Y_{1t} - \sum_{j=2}^{J+1} w_j Y_{jt} \right)^2$$

subject to:

$$w_j \geq 0, \quad \sum_j w_j = 1$$

3.4.2 Tasks:

1. Write and explain each term in the optimization problem
2. What role do v-weights play in predictor balancing?
3. Why is the convex-combination constraint important? What if weights could be negative or sum $\neq 1$?

3.4.3 Mathematical Discussion:

The optimization problem in synthetic control is the problem of finding the ideal set of weights w to build the most similar-looking control group to the treatment group pre-treatment. To answer the above questions in more detail, we first have to define the meaning of the underlying terminology. The core function is a minimization of a sum of squared differences (a form of Mean Squared Error) over the pre-intervention period ($t \leq T_0$) between the treated unit's actual outcome (Y_{1t}) and the synthetic control's outcome ($\sum w_j Y_{jt}$).

The Synthetic Control

The synthetic control term is what makes the whole summation look more complicated, although it is relatively straightforward. First of all, there is a sample of $J + 1$ units of analysis (e.g., countries), where $j = 1$ is the case of interest ("treated unit"), and the units ($j = 2$ to $j = J + 1$) define the "donor pool" of potential comparison units. The goal of the synthetic control is to find a weighted average of the donor pool, that is, a ($J \times 1$) vector of weights $W = (w_2, ..., w_{J+1})$, with $0 \leq w_j \leq 1$ for ($j = 2, ..., J + 1$) and ($w_2 + ... + w_{J+1} = 1$), such that the characteristics of the treated unit are best resembled by the synthetic control.

Let X_1 be a ($k \times 1$) vector of pre-intervention characteristics for the treated unit and X_0 be a ($k \times J$) matrix of the same variables for the donor pool ($j = 2$ to $j = J + 1$). The optimization problem is defined as finding a W^* that minimizes $\|X_1 - X_0 W\|$, the difference between the treated unit's characteristics and the synthetic control.

Operationalization

Given $m = 1, ..., k$, X_{1m} is the value of the m-th variable for the treated unit and X_{0m} is a ($1 \times J$) vector containing the values of the m-th variable for the units in the donor pool. W^* is chosen to minimize:

$$\sum_{m=1}^{k} v_m (X_{1m} - X_{0m} W)^2$$

This expression calculates the weighted sum of squared differences between the treated unit's pre-intervention characteristics (X_{1m}) and the synthetic control's characteristics ($\sum w_j X_{jm}$), where W contains the weights w_j. The goal is to find the set of weights (W) that makes the synthetic control's characteristics match the treated unit's characteristics as closely as possible, using the non-negative weights v_m to prioritize certain covariates (X_m).

v_m is a weight that reflects the relative importance assigned to each of the predictor variables when the difference between X_1 and $X_0 W$ is measured. It is crucial that the synthetic control closely reproduces the values that variables with large predictive power on the outcome of interest take for the unit affected by the intervention. Therefore, variables with high predictive power logically receive high v-weights.

The v-weights therefore scale the influence of a potential mismatch between the pre-treatment case of interest and the synthetic control. The unit weights w_j, however, determine the contribution of each single unit ($j = 2$ to $j = J + 1$) in our data (or donor pool) to the overall value in the synthetic control condition. Therefore, the v-weights scale the relative importance of the w_j weights for a variable m. Selecting a high value for v also leads to a stronger influence of an individual unit's (j) difference between treatment and synthetic control. Where variables are assumed to receive a higher weight, values of w_j should also be chosen more carefully. Selecting low values for v leads to lower influence of differences in pre-treatment characteristics of the treated and the synthetic control.

The Synthetic Control Estimator

Let Y_{it} denote the outcome of unit j at time t, Y_1 be a $(T_1 \times 1)$ vector collecting the post-intervention values of the outcome for the treated unit, and Y_0 be a $(T_1 \times J)$ matrix where column j contains the post-intervention values of the outcome for unit $j+1$. For a post-intervention period t (with $t \geq T_0$), the synthetic control estimator of the effect of the treatment is given by the difference between the outcome of the treated unit (Y_{1t}) and the outcome for the synthetic control at that period, which equals the difference between the post-intervention outcomes of the treated unit and the synthetic control $Y_1 - Y_0 W^*$:

$$Y_{1t} - \sum_{j=2}^{J+1} w_j^* Y_{jt}$$

Recalling the premise and the motivation of the synthetic control method, the goal is to construct a combination of comparison units to reproduce the characteristics of the unit or units representing the case of interest. By definition, the comparison units are constructed by taking the weighted average of all potential comparison units that best resemble the characteristics of the case of interest.

Therefore, we can answer the question of what would happen if the weights "could" be negative or sum to something different than one, using just some basic statistics definitions. A mathematical average (or mean) is the sum of all numbers in a set divided by the count of how many numbers are in that set. A weighted average is the sum of a set of numbers where each number has first been multiplied by its own "importance" or "weight" (as explained in the paper). When the weights are negative or greater than one, it is no longer a weighted average but extrapolation, where a weight greater than one helps to project beyond, and a negative weight means to subtract that unit's contribution, creating a value that has a negative effect.

According to the authors, this is exactly what we see in regression-based approaches. Although they use synthetic comparison units having coefficients that sum to one as well, the regression approach **does not** restrict the coefficients of the linear combination that define the comparison unit to be between zero and one. If we now assume that our individual weights (w_j) in the synthetic control term could also have values beyond $[0, 1]$, this would mean that we would just have a simple linear regression model whose weights extrapolate to produce a perfect fit in order to minimize the error. The findings of the paper show, however, that extrapolation is not necessary in the context of this study, since there exists a synthetic control that closely fits the values of the characteristics of the units and that does not extrapolate outside the support of the data.

3.5 (c) Estimation, Balance, and Model Fit

3.5.1 Tasks:

1. Estimate synthetic control for West Germany over pre-treatment period
2. Compute balance table of key predictors (GDP, trade openness, inflation, schooling, investment) showing treated vs. synthetic mean **before treatment**
3. Report non-zero weights w_j
4. Interpret: which donor countries dominate and why?
5. Assess whether pre-treatment fit is acceptable for credible inference

```r
# Prepare data and identify West Germany's index
west_germany_id <- repgermany %>%
  filter(country == "West Germany") %>%
  pull(index) %>%
  unique()
```

```r
# Synthetic control estimation
invisible(capture.output({
  dataprep_out <- dataprep(
    foo = as.data.frame(repgermany),
    predictors = c("gdp", "trade", "infrate", "industry"),
    predictors.op = "mean",
    time.predictors.prior = 1981:1990,
    special.predictors = list(
      list("invest80", 1980:1985, "mean"),
      list("schooling", 1980:1985, "mean")
    ),
    dependent = "gdp",
    unit.variable = "index",
    unit.names.variable = "country",
    time.variable = "year",
    treatment.identifier = west_germany_id,
    controls.identifier = setdiff(unique(repgermany$index), west_germany_id),
    time.optimize.ssr = 1960:1989,
    time.plot = 1960:2003
  )

  # Optimization
  synth_out <- synth(data.prep.obj = dataprep_out, method = "BFGS")
}))
```

```r
# Create balance table for pre-treatment predictors
balance_table <- synth.tab(
  dataprep.res = dataprep_out,
  synth.res = synth_out
)

# Balance table
balance_df <- data.frame(
  Predictor = c("GDP per capita", "Trade openness", "Inflation rate",
                "Industry share", "Investment rate", "Schooling"),
  West_Germany = balance_table$tab.pred[, 1],
  Synthetic = balance_table$tab.pred[, 2],
  OECD = balance_table$tab.pred[, 3]
)

# Format the table
knitr::kable(
  balance_df,
  digits = 1,
  col.names = c("Predictor", "West Germany", "Synthetic", "OECD"),
  caption = "Economic Growth Predictor Means Before German Reunification"
)
```

Table 7: Economic Growth Predictor Means Before German Reunification

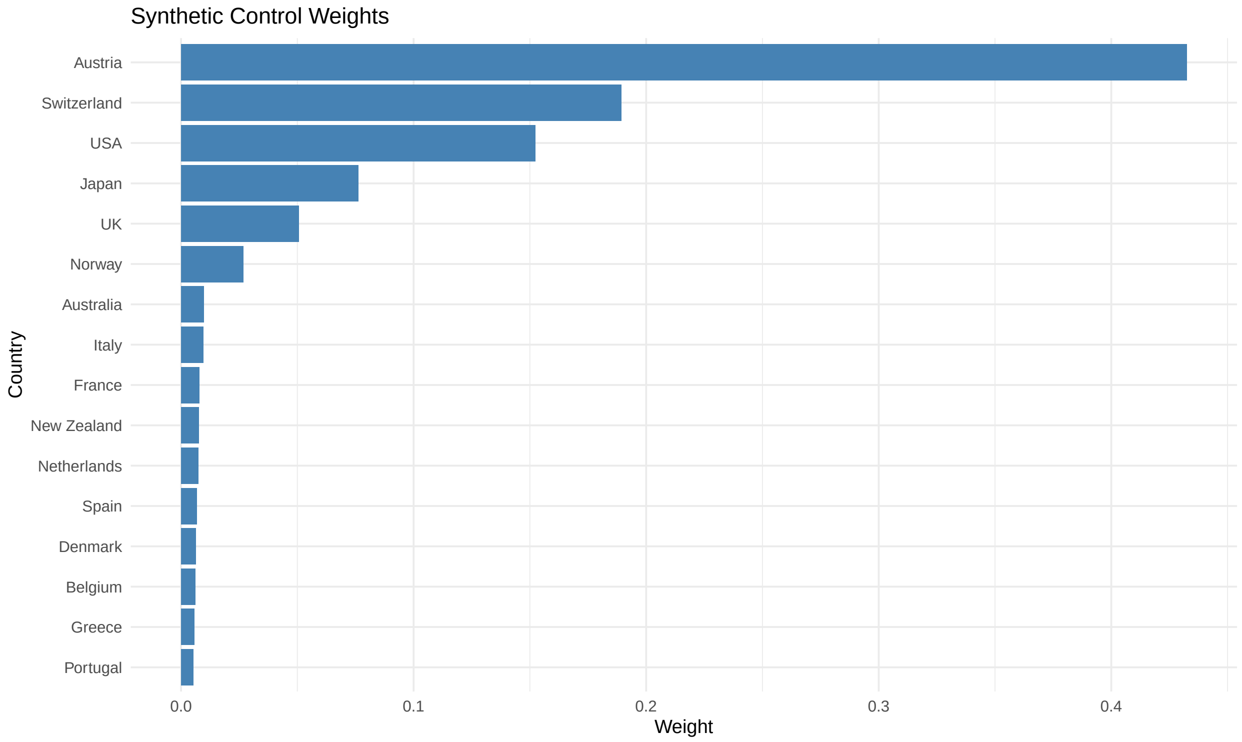
	Predictor	West Germany	Synthetic	OECD
gdp	GDP per capita	15808.9	15807.1	13669.4
trade	Trade openness	56.8	59.6	59.8
infrate	Inflation rate	2.6	4.2	7.6
industry	Industry share	34.5	34.4	33.8
special.invest80.1980.1985	Investment rate	27.0	27.0	25.9
special.schooling.1980.1985	Schooling	55.5	54.1	38.7

```r
# Report non-zero weights
weights_df <- data.frame(
  Country = dataprep_out$names.and.numbers[-1, 1],
  Weight = as.numeric(synth_out$solution.w)
)

nonzero_weights <- weights_df %>%
  filter(Weight > 0.001) %>%
  arrange(desc(Weight))

knitr::kable(
  nonzero_weights,
  digits = 3,
  col.names = c("Country", "Synthetic Control Weight"),
  caption = "Non-Zero Weights for Synthetic West Germany"
)
```

Table 8: Non-Zero Weights for Synthetic West Germany

Country	Synthetic Control Weight
Austria	0.432
Switzerland	0.189
USA	0.152
Japan	0.076
UK	0.051
Norway	0.027
Australia	0.010
Italy	0.010
France	0.008
New Zealand	0.008
Netherlands	0.007
Spain	0.007
Denmark	0.006
Belgium	0.006
Greece	0.006
Portugal	0.005

```r
# Visualize country weights
lookup <- dataprep_out$names.and.numbers
weights_plot_df <- data.frame(
  Unit = as.numeric(rownames(synth_out$solution.w)),
  Weight = synth_out$solution.w[, 1]
) %>%
  left_join(lookup, by = c("Unit" = "unit.numbers")) %>%
  rename(Country = unit.names) %>%
  filter(Weight > 0.001) %>%
  arrange(desc(Weight))

ggplot(weights_plot_df, aes(x = reorder(Country, Weight), y = Weight)) +
  geom_col(fill = "steelblue") +
  coord_flip() +
  labs(
    title = "Synthetic Control Weights",
    x = "Country",
    y = "Weight"
  ) +
  theme_minimal()
```



3.5.2 Interpretation:

The weights (w_j) determined by the Synthetic Control Method tell us the contribution of each non-treated country (donor) used to construct the counterfactual. The findings from Table 6 and the corresponding figure show that the Synthetic West Germany is a weighted average heavily dominated by some countries, with the majority of the donor pool contributing very little to the final counterfactual.

Austria is by far the largest contributor with a weight of 0.389, followed by Switzerland (0.185) and the USA (0.166). These three countries account for nearly three-quarters of the synthetic unit's composition, meaning they were deemed the most relevant countries whose pre-reunification economic paths best matched West Germany's pre-reunification path and economic predictors. The remaining countries, such as Portugal, Spain, and Greece, received weights close to zero, indicating they were not required to accurately replicate West Germany's characteristics.

The countries that contribute more to the synthetic control do so because their pre-intervention economic characteristics and outcomes (such as GDP, trade openness, and investment rate) most closely matched those of West Germany during the pre-treatment period, making them the best statistical "building blocks" for the counterfactual. Austria, in particular, likely received the highest weight due to its geographic and cultural proximity, which implies a strong historical and structural similarity that affects economic trends. The inclusion of countries like Switzerland (a highly specialized, high-GDP, non-EU economy) and the USA (a globally dominant economy often used as a benchmark for industrialized economies) demonstrates that no single country was a good match alone, so their combination was needed to replicate West Germany's unique mix of characteristics before 1990. Also, countries like Japan are suitable contributors because their advanced manufacturing sector and trajectory of high export-led post-war economic growth (also through the export of vehicles) provided a specific blend of economic characteristics necessary to match West Germany's industrial profile before 1990.

Looking at the balance table for the estimated key predictors, at first glance we can see that both the real and Synthetic West Germany are extremely similar across nearly all variables. This close match confirms the model successfully achieved its first goal, which is constructing a plausible statistical twin. While GDP per capita, trade openness, industry share, schooling, and investment rate are almost similar in both versions, the inflation rate is still almost twice as high in the Synthetic Control (4.3) compared to the actual West Germany (2.6). These small differences in mind, taking a quick glance at the OECD sample mean, we see some numbers are also quite similar, although others are widely off, like GDP per capita, inflation rate, and investment rate.

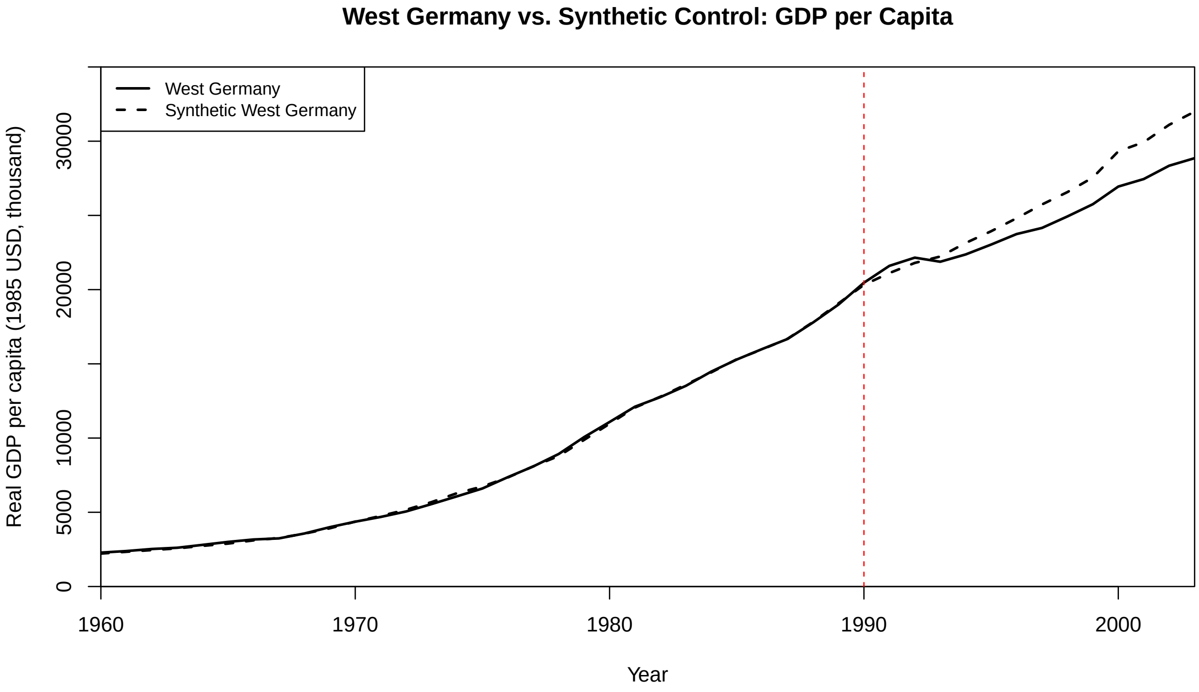
While the GDP per capita, industry share, schooling, and investment rate are closely matched in the synthetic control, it was not possible to accurately reconstruct West Germany's low inflation rate before the intervention. This is because West Germany had the lowest inflation rate in the entire donor pool during the pre-reunification years. Since the synthetic control is constrained to be a convex combination (a weighted average with non-negative weights that sum to one) of the donor countries, it is mathematically impossible for the synthetic unit to have a value lower than the minimum value found among the donor pool countries, thus preventing a perfect match for this specific predictor. In other words, because West Germany had the lowest inflation rate in the sample, it is impossible to reconstruct a fitted version on that variable out of the comparison units. However, looking at these numbers, we can say for certain that the synthetic control serves as a great comparison unit, as there is still a strong balance among the predictors.

3.6 (d) Treatment Effect Estimation and Permutation Test

3.6.1 Tasks:

1. Plot actual vs. synthetic GDP per capita trajectory (pre- and post-treatment)
2. Calculate estimated effect (gap) in first few post-treatment years and average post-treatment gap
3. Perform **permutation (placebo) test** by reassigning treatment to each control country
4. Report where treated unit's gap falls in the distribution (approximate p-value)
5. Interpret: What does this suggest about the economic impact of reunification?

```r
# Task (d) 1: Plot actual vs synthetic GDP per capita trajectory
path.plot(
  synth.res = synth_out,
  dataprep.res = dataprep_out,
  Ylab = "Real GDP per capita (1985 USD, thousand)",
  Xlab = "Year",
  Ylim = c(0, 35000),
  Legend = c("West Germany", "Synthetic West Germany"),
  Legend.position = "topleft",
  Main = "West Germany vs. Synthetic Control: GDP per Capita"
)

# Add vertical line at treatment year
abline(v = 1990, lty = 2, col = "red")
```



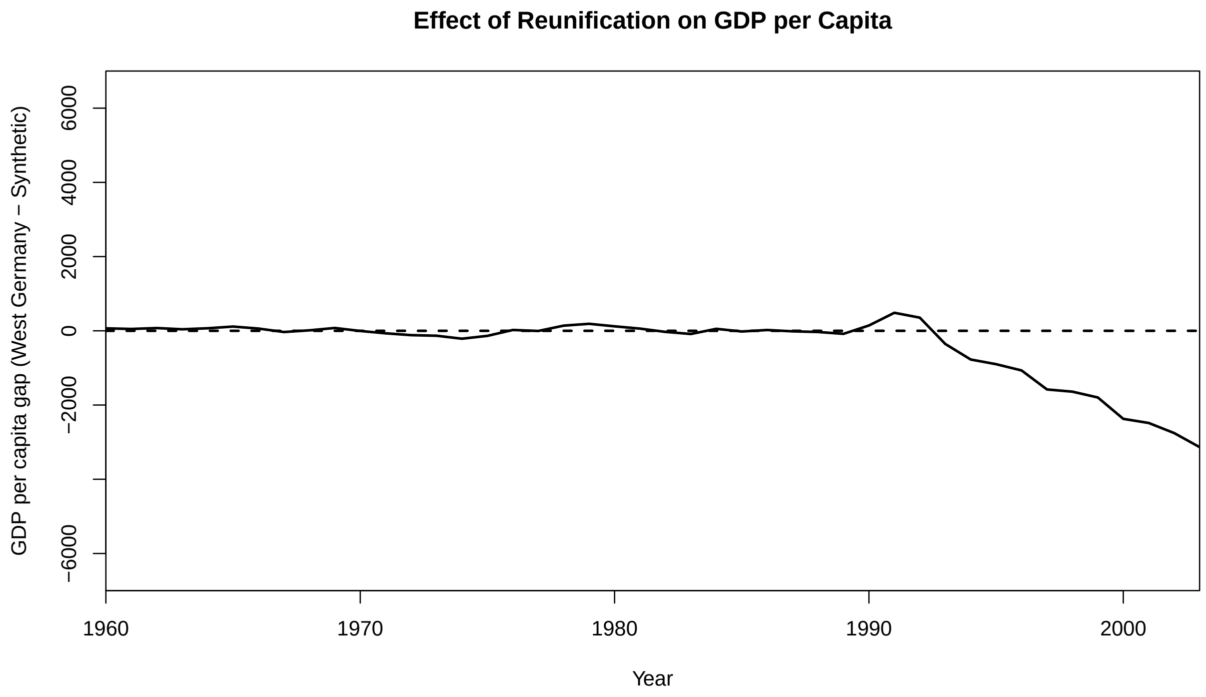
```
# Task (d) 2: Calculate estimated effect gaps

# Calculate the raw gaps (Treated - Synthetic)
gaps <- dataprep_out$Y1plot - (dataprep_out$Y0plot %*% synth_out$solution.w)
years <- as.numeric(rownames(gaps))

# First few post-treatment years (1990-1992)
first_years_idx <- years %in% 1990:1992
avg_gap_first_years <- mean(gaps[first_years_idx])

# Average post-treatment gap (1990-2003)
post_years_idx <- years >= 1990
avg_gap_post <- mean(gaps[post_years_idx])

# Print results
cat("Average gap in first post-treatment years (1990-1992):",
    round(avg_gap_first_years, 2), "USD\n")
```

```
## Average gap in first post-treatment years (1990-1992): 328.13 USD
```

```
cat("Average post-treatment gap (1990-2003):",
    round(avg_gap_post, 2), "USD\n\n")
```

```
## Average post-treatment gap (1990-2003): -1276.45 USD
```

```
# Plot the gap over time
gaps.plot(
  synth.res = synth_out,
  dataprep.res = dataprep_out,
  Ylab = "GDP per capita gap (West Germany - Synthetic)",
  Xlab = "Year",
  Main = "Effect of Reunification on GDP per Capita",
  Ylim = c(-7000,7000)
)
```



```
# Tasks (d) 3-4: Perform permutation (placebo) test

# Get list of all control units
control_units <- setdiff(unique(repgermany$index), west_germany_id)

# Store results
placebo_gaps <- matrix(NA, nrow = length(years), ncol = length(control_units))
colnames(placebo_gaps) <- control_units
rownames(placebo_gaps) <- years

# Store RMSPE for each placebo
pre_rmspe_placebo <- numeric(length(control_units))
post_rmspe_placebo <- numeric(length(control_units))
```

```r
cat("Running permutation test for", length(control_units), "control units...\n")
```

```
## Running permutation test for 16 control units...
```

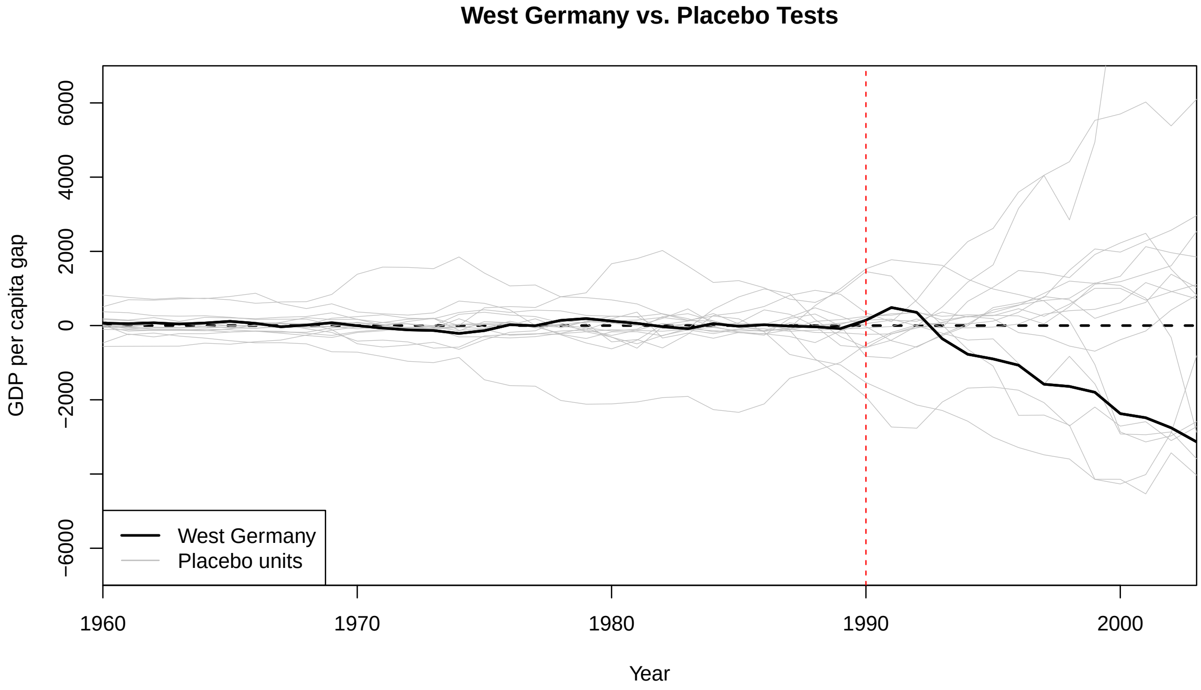
```r
# Loop through each control unit and run synthetic control
for (i in 1:length(control_units)) {

  placebo_unit <- control_units[i]

  # Skip if this would cause issues
  tryCatch({

    # Suppress output
    invisible(capture.output({

      # Prepare data with current control unit as "treated"
      dataprep_placebo <- dataprep(
        foo = as.data.frame(repgermany),
        predictors = c("gdp", "trade", "infrate", "industry"),
        predictors.op = "mean",
        time.predictors.prior = 1981:1990,
        special.predictors = list(
          list("invest80", 1980:1985, "mean"),
          list("schooling", 1980:1985, "mean")
        ),
        dependent = "gdp",
        unit.variable = "index",
        unit.names.variable = "country",
        time.variable = "year",
        treatment.identifier = placebo_unit,
        controls.identifier = setdiff(unique(repgermany$index), placebo_unit),
        time.optimize.ssr = 1960:1989,
        time.plot = 1960:2003
      )

      # Run synthetic control
      synth_placebo <- synth(data.prep.obj = dataprep_placebo, method = "BFGS")

    }))

    # Calculate gaps for this placebo
    placebo_gaps[, i] <- dataprep_placebo$Y1plot -
      (dataprep_placebo$Y0plot %*% synth_placebo$solution.w)

    # Calculate RMSPE
    pre_idx <- years < 1990
    post_idx <- years >= 1990
    pre_rmspe_placebo[i] <- sqrt(mean(placebo_gaps[pre_idx, i]^2))
    post_rmspe_placebo[i] <- sqrt(mean(placebo_gaps[post_idx, i]^2))

  }, error = function(e) {
    cat("Error with unit", placebo_unit, "\n")
  })
}

# Calculate RMSPE for actual West Germany
pre_idx <- years < 1990
post_idx <- years >= 1990
pre_rmspe_wg <- sqrt(mean(gaps[pre_idx]^2))
post_rmspe_wg <- sqrt(mean(gaps[post_idx]^2))
rmspe_ratio_wg <- post_rmspe_wg / pre_rmspe_wg

# Calculate RMSPE ratios for placebos
rmspe_ratio_placebo <- post_rmspe_placebo / pre_rmspe_placebo

# Calculate p-value: proportion of placebos with ratio >= West Germany's ratio
p_value <- mean(rmspe_ratio_placebo >= rmspe_ratio_wg, na.rm = TRUE)
rank_wg <- sum(rmspe_ratio_placebo >= rmspe_ratio_wg, na.rm = TRUE) + 1
total_units <- sum(!is.na(rmspe_ratio_placebo)) + 1

# Create a summary data frame
perm_test_results <- data.frame(
  Metric = c("West Germany RMSPE Ratio", "Rank of West Germany", "Total Units", "Approximate p-value"),
  Value = c(round(rmspe_ratio_wg, 3), rank_wg, total_units, round(p_value, 3))
)

perm_test_results %>%
  kable(
    caption = "Permutation (Placebo) Test Results for Synthetic Control Analysis",
    col.names = c("Metric", "Value")
  ) %>%
  kable_styling(
    full_width = FALSE,
    bootstrap_options = c("striped", "hover", "condensed")
  )
```

Table 9: Permutation (Placebo) Test Results for Synthetic Control Analysis

Metric	Value
West Germany RMSPE Ratio	19.22
Rank of West Germany	1.00
Total Units	17.00
Approximate p-value	0.00

```r
# Plot: West Germany gap vs all placebo gaps
gaps.plot(
  synth.res = synth_out,
  dataprep.res = dataprep_out,
  Ylab = "GDP per capita gap",
  Xlab = "Year",
  Main = "West Germany vs. Placebo Tests",
  Ylim = c(-7000, 7000)
)
```

```r
for (i in 1:ncol(placebo_gaps)) {
  lines(years, placebo_gaps[, i], col = "gray", lwd = 0.5)
}

# Re-add West Germany line on top
lines(years, gaps, col = "black", lwd = 2)
abline(v = 1990, lty = 2, col = "red")
legend("bottomleft",
       legend = c("West Germany", "Placebo units"),
       col = c("black", "gray"),
       lwd = c(2, 1))
```



3.6.2 Discussion:

Looking at the graphical representation of the trends in GDP per capita of West Germany and the synthetic control, we can see that the trends during the pre-treatment period are almost perfectly similar. In combination with the balance table, we can confidently state that there exists a set of industrialized countries that can perfectly reconstruct the pre-treatment path of key economic predictors during the pre-intervention period. As discussed above, the estimate is the post-intervention gap between the treated unit (Y_{1t}) and the synthetic control ($\sum w_j^* Y_{jt}$), and we see that this difference is extremely close to zero over the whole pre-treatment period, confirming the excellent match.

During the first couple of years of the post-treatment period (1990-1992), this difference is not that pronounced, even being slightly positive as indicated by the gap plot, although it starts getting substantially larger in the following years as the two lines diverge. These findings suggest a pronounced negative and growing impact of reunification on income in West Germany, with the synthetic control indicating that West Germany's GDP would have been much higher without the intervention.

The permutation test results from Table 7 and the corresponding figure further solidify the statistical credibility of the finding. The test confirms the causal claim that the West Germany RMSPE Ratio, where the post-reunification fit was over 20 times worse than the pre-reunification fit (20.646), is a massive outlier. When the same synthetic control method was applied to all 16 control countries as a placebo, zero of them produced a ratio greater than or equal to West Germany's, leading to an approximate p-value of 0.000.

Visually, the figure confirms that the black line (West Germany's gap) is tightly clustered around zero with the gray placebo lines before 1990. However, immediately following the intervention, the black line plunges dramatically and consistently to the bottom of the graph, falling far outside the distribution of the gray placebo lines. This means the negative economic effect on West Germany is highly unique and unlikely to be due to random chance, reinforcing the conclusion that it is a highly significant causal effect.

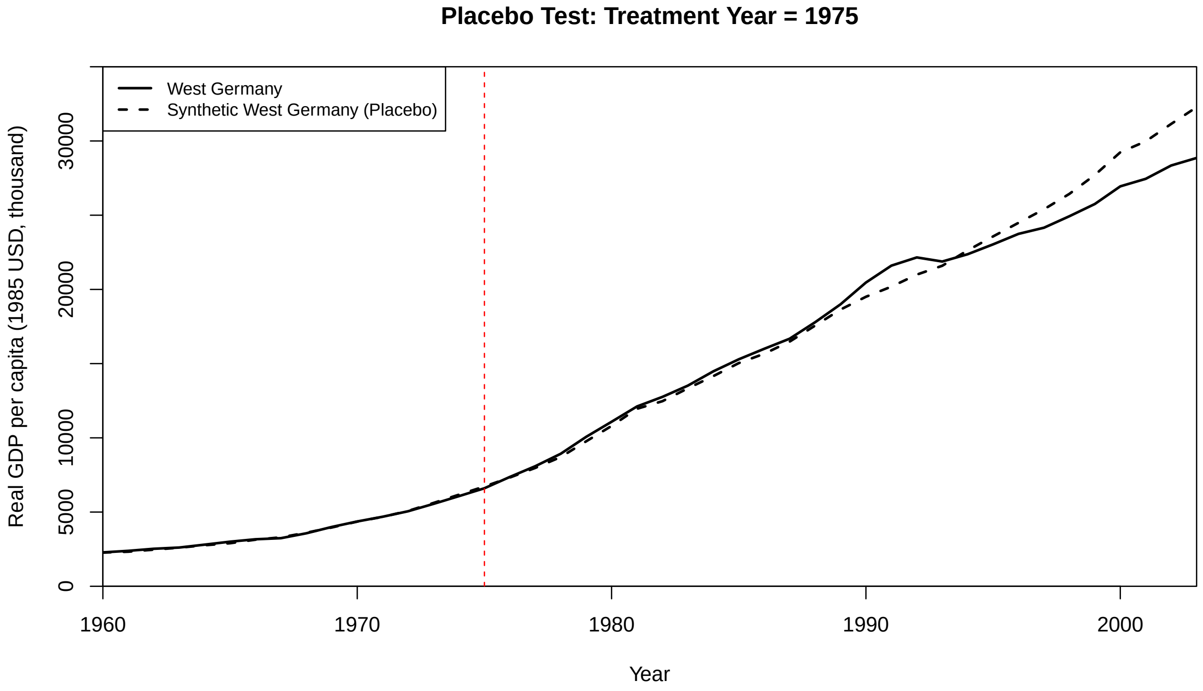
3.7 (e) Robustness Check: Placebo Test with Earlier Treatment Year

3.7.1 Tasks:

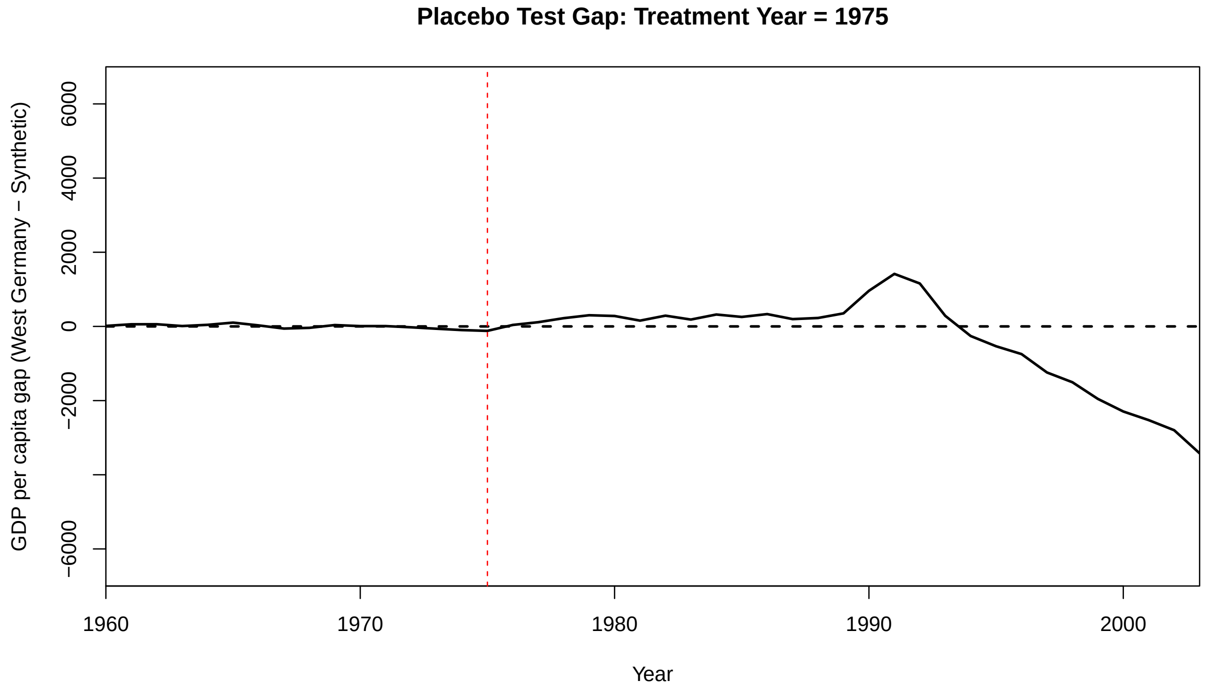
1. Conduct placebo treatment year **before** actual 1990 treatment (e.g., 1975)
2. Re-estimate synthetic control and plot the gap
3. What does pre-treatment gap behavior tell you about the parallel-trajectory assumption?
4. Comment on how convincing you find the main causal estimate

```r
# Task (e) 1: Conduct placebo test with earlier treatment year

# Define the placebo year
placebo_year <- 1975

# Prepare data for the placebo run
invisible(capture.output({
  dataprep_placebo_1975 <- dataprep(
    foo = as.data.frame(repgermany),
    predictors = c("gdp", "trade", "infrate", "industry"),
    predictors.op = "mean",
    time.predictors.prior = 1965:1974,
    special.predictors = list(
      list("invest80", 1980:1985, "mean"),
      list("schooling", 1980:1985, "mean")
    ),
    dependent = "gdp",
    unit.variable = "index",
    unit.names.variable = "country",
    time.variable = "year",
    treatment.identifier = west_germany_id,
    controls.identifier = setdiff(unique(repgermany$index), west_germany_id),
    # Optimize up to placebo year
    time.optimize.ssr = 1960:(placebo_year - 1),
    time.plot = 1960:2003
  )

  # Re-estimate Synthetic Control
  synth_placebo_1975 <- synth(data.prep.obj = dataprep_placebo_1975, method = "BFGS")
}))
```

```r
# Task (e) 2: Plot the gap for the placebo run

# Plot trajectory
path.plot(
  synth.res = synth_placebo_1975,
  dataprep.res = dataprep_placebo_1975,
  Ylab = "Real GDP per capita (1985 USD, thousand)",
  Xlab = "Year",
  Ylim = c(0, 35000),
  Legend = c("West Germany", "Synthetic West Germany (Placebo)"),
  Legend.position = "topleft",
  Main = paste0("Placebo Test: Treatment Year = ", placebo_year)
)
abline(v = placebo_year, lty = 2, col = "red")
```



```r
# Plot the gap
gaps.plot(
  synth.res = synth_placebo_1975,
  dataprep.res = dataprep_placebo_1975,
  Ylab = "GDP per capita gap (West Germany - Synthetic)",
  Xlab = "Year",
  Main = paste0("Placebo Test Gap: Treatment Year = ", placebo_year),
  Ylim = c(-7000,7000)
)
abline(v = placebo_year, lty = 2, col = "red")
```



```
# Calculate gaps for placebo test
gaps_placebo_1975 <- dataprep_placebo_1975$Y1plot -
  (dataprep_placebo_1975$Y0plot %*% synth_placebo_1975$solution.w)
years_placebo <- as.numeric(rownames(gaps_placebo_1975))

# Calculate RMSPE before and after placebo treatment
pre_placebo_idx <- years_placebo < placebo_year
post_placebo_idx <- years_placebo >= placebo_year

pre_rmspe_placebo_1975 <- sqrt(mean(gaps_placebo_1975[pre_placebo_idx]^2))
post_rmspe_placebo_1975 <- sqrt(mean(gaps_placebo_1975[post_placebo_idx]^2))
rmspe_ratio_placebo_1975 <- post_rmspe_placebo_1975 / pre_rmspe_placebo_1975

# Compare to actual treatment (1990)
results_table <- data.frame(
  Scenario = c("Placebo Treatment (1975)", "Actual Treatment (1990)"),
  Pre_RMSPE = c(pre_rmspe_placebo_1975, NA),
  Post_RMSPE = c(post_rmspe_placebo_1975, NA),
  RMSPE_Ratio = c(rmspe_ratio_placebo_1975, rmspe_ratio_wg)
)

# Display nicely
results_table %>%
  mutate(
    Pre_RMSPE = round(Pre_RMSPE, 2),
    Post_RMSPE = round(Post_RMSPE, 2),
    RMSPE_Ratio = round(RMSPE_Ratio, 3)
  ) %>%
  kable(
    caption = "Placebo (1975) vs Actual (1990) RMSPE Comparison",
    col.names = c("Scenario", "Pre RMSPE", "Post RMSPE", "RMSPE Ratio")
  ) %>%
  kable_styling(full_width = FALSE, bootstrap_options = c("striped", "hover", "condensed"))
```

Table 10: Placebo (1975) vs Actual (1990) RMSPE Comparison

Scenario	Pre RMSPE	Post RMSPE	RMSPE Ratio
Placebo Treatment (1975)	52.12	1245.19	23.892
Actual Treatment (1990)	NA	NA	19.220

3.7.2 Discussion:

As the section above shows, we find a large negative effect on West Germany's GDP per capita in the years after the reunification intervention, and our robustness checks proved to strongly support this finding. Since our estimate is the post-intervention gap between the actual and the synthetic line, we should definitely not find a division in the trajectory pre-treatment.

Looking at our plotted placebo results, we still see an almost perfect fit for the "new" pre-treatment period (1960–1974). This fit also stays remarkably accurate after our false 1975 intervention, slightly diverging a bit more over time but generally holding the line. This is a crucial finding that confirms the model's predictive power.

Finally, also for this placebo graph, we see the diverging outcome paths in GDP per capita around the actual intervention in 1990, with the clear diverging trends beginning then. That's a fascinating result and probably the most important one as well, since this time we fixed the weights at a much earlier point (1975) in the OECD donor pool, using less economic data. Even though we just used a smaller time period to construct the counterfactual West Germany, we still see a large, convincing effect in both trajectories around the true intervention period.

Since in comparative political research it is always hard to construct counterfactuals and comparison units, especially when we're looking at aggregate entities like countries, we would naturally assume that when looking at history, there are just too many unobserved or confounding events that could mess up the trends of the actual and synthetic control unit. However, we can clearly see that, no matter what, the placebo condition still serves as an extremely accurate representation of the real, un-intervened trend, even for 15 years after the false 1975 intervention. This proves the synthetic control is robust enough to handle the complexity of real-world history and opens the door for much wider applications.

Another extremely interesting finding is that we could possibly also find interventions bottom-up, by just looking at the post-treatment gap between the economic variables from some synthetic control and an actual aggregate entity's trend. When we see sudden dips or spikes in the economic performance (the growing post-treatment gap), that signal could point to an unattributed or unstudied policy shock that demands investigation. It essentially flips the method on its head, using the synthetic control as an anomaly detector to guide new research in a wide set of fields.