

Performing Propensity Score-Integrated Comparative Analysis in External Control Arm Studies Using R

March 7, 2023

Lixiao Chen*, Lorraine Fang*, Diana Tai*, Marc De Benedetti**, Lihua Yue**

Bristol Myers Squibb, Berkeley Heights, New Jersey, USA

*RWE Programmer **RWE Biostatistician

Abstract

Generating an external control arm (ECA) using real-world data (RWD) in supporting regulatory submissions is becoming an industry trend. This project explored how to use R to support a regulatory submission in an ECA study.

Working with RWD introduces unique challenges that are not present in randomized clinical trials, such as confounding, selection bias, and missing data. This project has explored the novel initiative of converting these more advanced aspects of the statistical analysis from SAS to R, and contribute to the limited R documentation related to ECA studies.

At time of writing, many aspects of an ECA study using propensity score methodologies have been successfully implemented in R, such as propensity score generation, **multiple imputations, and handling IPTW weights in outcome models**. Contributions to the existing R methodology will be presented, along with **outlining some challenges** and their solutions as well as potential future work.

Real World Data (RWD) and External Control Arm (ECA)

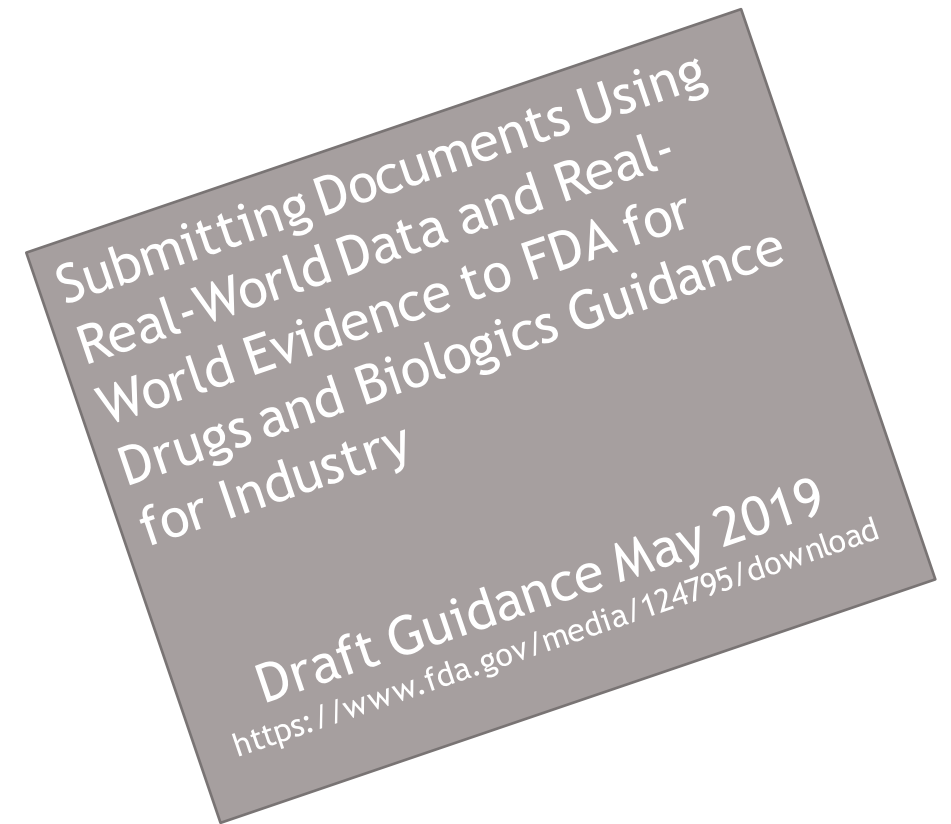
- **RWD**: The FDA defines **real-world data** as healthcare information derived from multiple sources **outside of typical clinical research settings**. This data is collected, de-identified, and stored in a variety of sources to be later analyzed
- **ECA**: A cohort created from existing patient-level data, either old clinical trial or RWD to be used as a comparator arm

Background & Motivation of Using External Control Arm

- Randomized Controlled Trials (RCTs) are industry gold standard
 - Used to evaluate efficacy and safety of treatments
 - Good randomization corrects for confounders and biases
- Barriers to randomization
 - Disease rarity
 - Scarcity of patients
 - Ethical & feasibility considerations
 - Temporal & financial costs

When to Use External Control Arm

- **Single-arm Studies**
 - Uncontrolled studies require context to interpret outcomes
 - Clinical settings with no standard of care treatment, limited options
- **Orphan & rare diseases**
 - Difficult recruitment of patients
- **Consider for**
 - Diseases with highly predictable courses and mortality
 - Studies in which the drug effect is self-evident
 - Situations when the effect of treatment is substantial



Real-World ECA Study Challenges

- Randomization does not exist between external control arm and the investigational treatment arm
- Common challenges
 - Selection bias
 - Sample selection does not accurately reflect target population
 - Addressed by appropriate ECA study design
 - Confounding
 - Systematic differences in patient characteristics exist between the treatment arm and external control arm
 - Missing data
- This presentation focuses on mitigation methods for confounding and missing data through programming procedures

Mitigation Methods for Confounding

Propensity Score Generation:

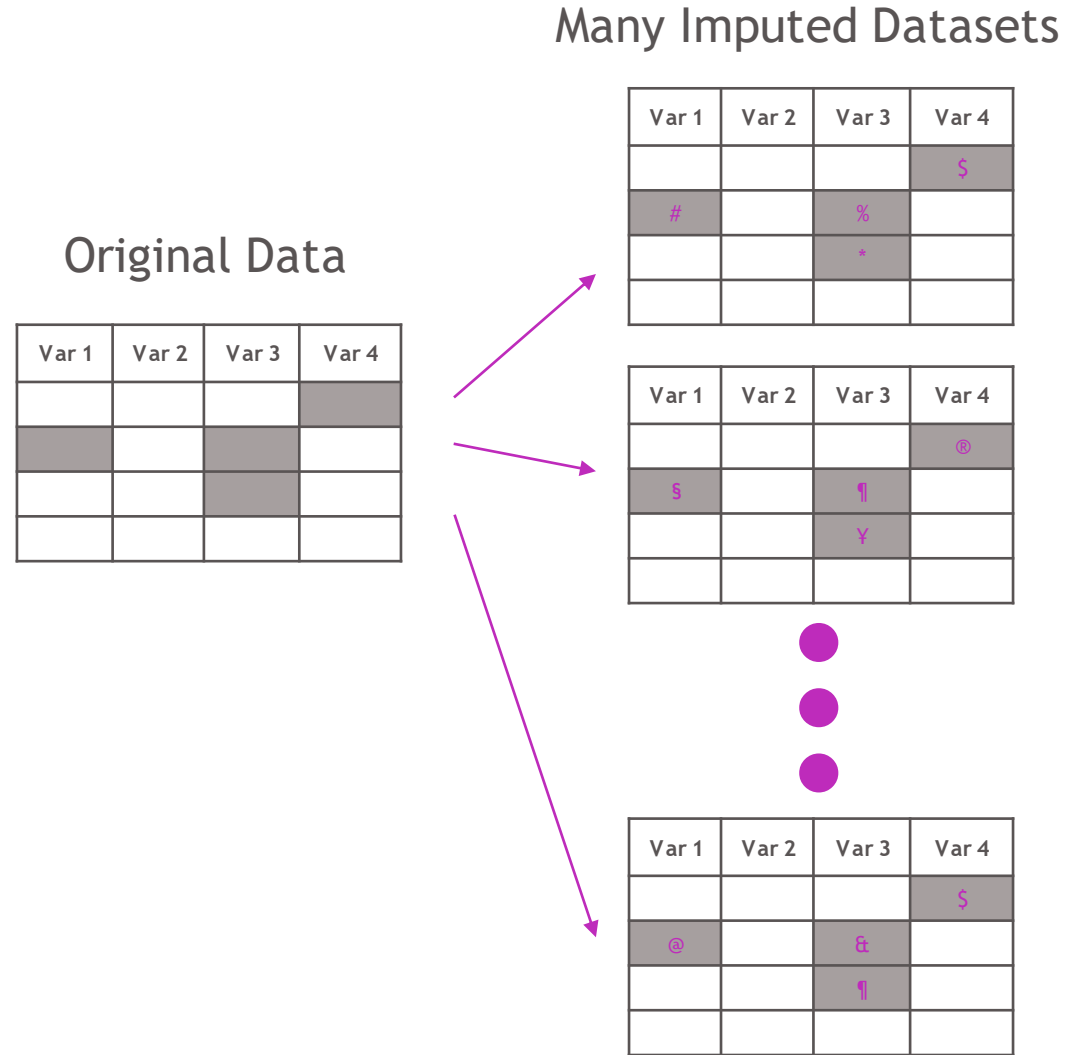
- Potential confounder list used in PS modeling is prespecified in SAP
- PS balancing is outcome independent (firewall/blinding)

Balancing Methods Implemented:

- Stabilized Inverse Probability of Treatment Weighting (sIPTW) to balance patient characteristics between the two arms
 - Based on inverse of propensity score (PS) of receiving treatment and the proportion of patients from single-arm clinical trial or ECA arm
 - Weights used in statistical analysis of outcome data
- Propensity Score Matching (PSM)
 - Patients matched based on PS using 1:n greedy nearest neighbor or optimal matching with a caliper and sampling without replacement

Mitigation Methods for Missing Data

- Use multiple imputation method to fill in missing values
 - Assume missing at random (MAR)
 - Systematic process to generate plausible numbers derived from distributions of, and relationships among, observed variables
 - This is repeated multiple times to reflect the uncertainty around the true value
- Analysis is conducted within each imputed dataset, and parameter estimates obtained from each analyzed dataset are then combined for inference



Challenges in Converting SAS Code to R

- Established SAS pipeline leverages the below main procedures to perform the statistical analyses:
 - **proc phreg**: used to analyze survival data, ability to handle multiply imputed data and weights
 - Weighted binomial using **proc freq**: creates frequency table with binomial proportions, ability to handle weights
 - **proc genmod**: fits generalized linear models to the data, ability to handle multiply imputed data and weights
- These procedures all have an equivalent R function; however, R does not have an established method for handling the added statistical layers needed in RWE studies (for example handling multiple imputations or PS weights)
- Different default values are used by SAS and R, e.g. censoring, options for functions

Challenge #1: proc phreg (IPTW Adjusted Survival Endpoints)

R has a fixed
value of 0 for
censor

→

```
outcome <- outcome %>%  
  mutate(CNSR = case_when(  
    CNSR == 0 ~ 1,  
    CNSR == 1 ~ 0  
  ))
```

group_by didn't
work, so a for loop
was implemented to
account for multiple
imputations

→

```
results_list = data.frame()  
  
for (i in unique(outcome$imputation)) {  
  results <- outcome %>%  
    filter(imputation == i)  
  
  model <- coxph(Surv(AVAL, CNSR) ~ trtpn,  
    weights = STWGT, ties = "efron",  
    data = results  
  )  
  results_list = rbind(results_list, data.frame(coefficients(summary(model))))  
}  
  
ESTHR <- results_list %>%  
  rename("Estimate" = coef, "StdErr" = robust.se) %>%  
  select(-exp.coef., -se.coef., -z) %>%  
  mutate(StmtNo = 1, Label = "JCAR017 VS RWE")
```

```
proc phreg data=outcome covs;  
  by _Imputation_;  
  class trtpn / param=glm;  
  model aval*cnsr(1) = trtpn / ties=Efron;  
  lsestimate trtpn 'JCAR017 VS RWE' -1 1;  
  weight stwgt /normalize;  
  ods output LSMEstimates=LSMEsts;  
  
run;
```

← coxph and Surv from
survival package

Results in R and SAS

	Estimate	StdErr	Pr...z..	StmtNo	Label
trtpn	-0.5325110	0.2407663	0.02698504	1	JCAR017 VS RWE
trtpn1	-0.5259833	0.2408942	0.02900182	1	JCAR017 VS RWE
trtpn2	-0.5412497	0.2394903	0.02382080	1	JCAR017 VS RWE
trtpn3	-0.5218779	0.2399016	0.02960140	1	JCAR017 VS RWE
trtpn4	-0.5224917	0.2397823	0.02932976	1	JCAR017 VS RWE
trtpn5	-0.5413108	0.2398211	0.02399907	1	JCAR017 VS RWE
trtpn6	-0.5561015	0.2389446	0.01994805	1	JCAR017 VS RWE
trtpn7	-0.5201458	0.2409443	0.03086721	1	JCAR017 VS RWE
trtpn8	-0.5347389	0.2395788	0.02561526	1	JCAR017 VS RWE
trtpn9	-0.5411125	0.2388813	0.02350080	1	JCAR017 VS RWE
trtpn10	-0.5357990	0.2402284	0.02572265	1	JCAR017 VS RWE
trtpn11	-0.5579332	0.2391790	0.01966381	1	JCAR017 VS RWE

Label	Estimate	StdErr
JCAR017 VS R...	-0.5325	0.2408
JCAR017 VS R...	-0.5260	0.2409
JCAR017 VS R...	-0.5412	0.2395
JCAR017 VS R...	-0.5219	0.2399
JCAR017 VS R...	-0.5225	0.2398
JCAR017 VS R...	-0.5413	0.2398
JCAR017 VS R...	-0.5561	0.2389
JCAR017 VS R...	-0.5201	0.2409
JCAR017 VS R...	-0.5347	0.2396
JCAR017 VS R...	-0.5411	0.2389
JCAR017 VS R...	-0.5358	0.2402
JCAR017 VS R...	-0.5579	0.2392

Challenge #2: Weighted Binomial (IPTW Adjusted Proportions)

Had to account for weights manually before running prop.test

```
prop <- outcome %>%  
  group_by(imputation, trtpn) %>%  
  summarize(  
    x = sum(CNSR * STWGT),  
    n = sum(((1 - CNSR) * STWGT)) + x  
  )
```

```
binom_results <- prop.test(  
  x = round(prop$x * 1000, digits = 0),  
  n = round(prop$n * 1000, digits = 0)  
)
```

```
binom <- data.frame(  
  imputation = prop$imputation, trtpn = prop$trtpn,  
  prop = round(binom_results$estimate, 6),  
  prop_se = round(sqrt(binom_results$estimate *  
    (1 - binom_results$estimate) / prop$n), 6)  
)
```

```
proc freq data=outcome1;  
  tables cnsr / cl binomial(level=1);  
  by _Imputation_ trtpn;  
  weight stwgt;  
  ODS OUTPUT BINOMIAL=prop;  
run;
```

SAS can accept non-integer counts in a binomial test

Multiplication by 1000 must be done in R to preserve 3 decimal places of precision

Results in R and SAS

imputation Imputation Number	trtpn	prop	prop_se
1	0	0.808649	0.023827
1	1	0.571479	0.060033
2	0	0.809474	0.023789
2	1	0.584992	0.060127
3	0	0.810220	0.023767
3	1	0.581023	0.059131
4	0	0.807683	0.023883
4	1	0.580883	0.060971

IMPUTATION N	TRTPN	Label1	prop	prop_se
1	1	0 ASE	0.808650	0.023827
2	2	0 ASE	0.809471	0.023789
3	3	0 ASE	0.810222	0.023767
4	4	0 ASE	0.807684	0.023883

IMPUTATION N	TRTPN	Label1	prop	prop_se
31	1	1 ASE	0.571488	0.060033
32	2	1 ASE	0.584996	0.060127
33	3	1 ASE	0.581018	0.059132
34	4	1 ASE	0.580884	0.060971

Challenge #3: proc genmod (IPTW Adjusted Response Rate)

```
proc genmod data=outcome descending;  
  by _imputation_;  
  class trtpn(ref="1") aval subjid / param=glm;  
  model aval(event='1') = trtpn /dist= binomial link=log;  
  Estimate 'JCAR017 - RWE' trtpn -1 1 ;  
  weight stwgt;  
  repeated subject=subjid / type=unstr; * robust variance;  
  lsmeans trtpn;  
  ods output estimates= _estd lsmeans=lststr;  
  
run;
```

GLM for fitting the weighted model, then use Coeftest from lmtest package combined with sandwich package to obtain robust covariance matrix estimators

```
results_list <- data.frame()  
for (i in unique(outcome$imputation)) {  
  results <- outcome %>%  
    filter(imputation == i)  
  
  est <- glm(AVAL~TRTPN,family=quasibinomial(link = "log"), weights = STWGT ,data=results)  
  est1 <- tidy(coeftest(est, vcov = sandwich))[2,]  
  results_list <- rbind(results_list, data.frame(est1))  
}  
  
result<- results_list %>%  
  rename(SE = "std.error")
```

Results in R and SAS

1	1	0.7901	0.1641
2	2	0.7989	0.1643
3	3	0.7979	0.1641
4	4	0.7920	0.1650
5	5	0.7990	0.1651
6	6	0.7950	0.1638
7	7	0.8021	0.1633
8	8	0.7833	0.1648
9	9	0.7937	0.1641
10	10	0.8008	0.1631
11	11	0.7946	0.1642
12	12	0.7977	0.1642

	term	estimate	SE	statistic	p.value
1	TRTPN	0.7901015	0.1641434	4.813482	1.483229e-06
2	TRTPN	0.7988869	0.1643219	4.861718	1.163715e-06
3	TRTPN	0.7978873	0.1640706	4.863073	1.155773e-06
4	TRTPN	0.7919978	0.1649646	4.801018	1.578613e-06
5	TRTPN	0.7989618	0.1650837	4.839738	1.300103e-06
6	TRTPN	0.7950148	0.1637949	4.853721	1.211664e-06
7	TRTPN	0.8021452	0.1633281	4.911250	9.049742e-07
8	TRTPN	0.7832619	0.1647803	4.753371	2.000527e-06
9	TRTPN	0.7937247	0.1640810	4.837396	1.315514e-06
10	TRTPN	0.8007592	0.1630902	4.909915	9.111586e-07
11	TRTPN	0.7946242	0.1641506	4.840824	1.293020e-06
12	TRTPN	0.7976779	0.1641938	4.858149	1.184884e-06

Challenge #4: Different Default Values

```
proc means data=_indat QNTLDEF=5;  
  by _imputation_;  
  var stwgt;  
  output out=percout p5=p5 p95=p95;  
run;
```

```
quantile<- adpropef %>%  
  group_by(imputation) %>%  
  summarize(q5 = quantile(STWGT, probs = 0.05, type=2),  
            q95 = quantile(STWGT, probs = 0.95, type=2))
```

Normally as we will not specify this in SAS PROC MEANS, the default quantile definition is QNTLDEF=5 which corresponds to type 2 in all 9 algorithms compiled by Hyndman and Fan in 1986.

Default type used in R is type=7

In order to match SAS results in R, we need to specify type=2 in R quantile function

Hyndman & Fan (also R)	SAS
type= 1	QNTLDEF= 3
type= 2	QNTLDEF= 5
type= 3	QNTLDEF= 2
type= 4	QNTLDEF= 1
type= 6	QNTLDEF= 4

Key Takeaways

- Many SAS programs for tables/graphs were successfully converted to R
- Some complex challenges arose related to the more complex propensity score methodology and multiple imputation
- This type of project requires close collaboration with the Biostats team
- Challenges can be solved; however, they require more time to investigate
- Attempted procedures in R that still need new packages developed to be able to complete (for example a package with a function that can take multiple imputations into account without using a loop)
- This project is a great illustration of how innovation and science thrives with interdisciplinary collaboration

Future Work

- Multiple imputation can be done in R using mice package
 - However, the results generated by SAS and R are different, even if the same random seed is used
 - This impacts propensity score generation, with the differences also cascading down to impact balancing and matching
- Matching can be performed with the **MatchIt** package
 - However, propensity score matching also yields different results from SAS, even if propensity scores in the input data are the same
- Additional work is needed to investigate the disparity between the results generated by SAS and R

Contact Information

Lixiao Chen

- Manager, RWE Statistical Programming
- Email: lixiao.chen@bms.com

Lorraine Fang

- Associate Director, RWE Statistical Programming
- Email: lorraine.fang@bms.com

Thank you