Cohort Building Program

User manual

V1.0

Thuy Thai
Department of Pharmaceutical Outcomes and Policy
University of Florida
October 19, 2021

CoDES

CENTER FOR DRUG EVALUATION & SAFETY

An overview of the Cohort Building Program

This Cohort Building Program was developed to create two analytic datasets with a new user and active comparator cohort design. The program includes two sub-programs (program I and II). The program I is used to identify patients with exposure/control drug, index date, a continuous duration of exposure/control drug use, follow-up time, age and sex at index date. The program II defines outcome, outcome date, and creates as many as desired covariates.

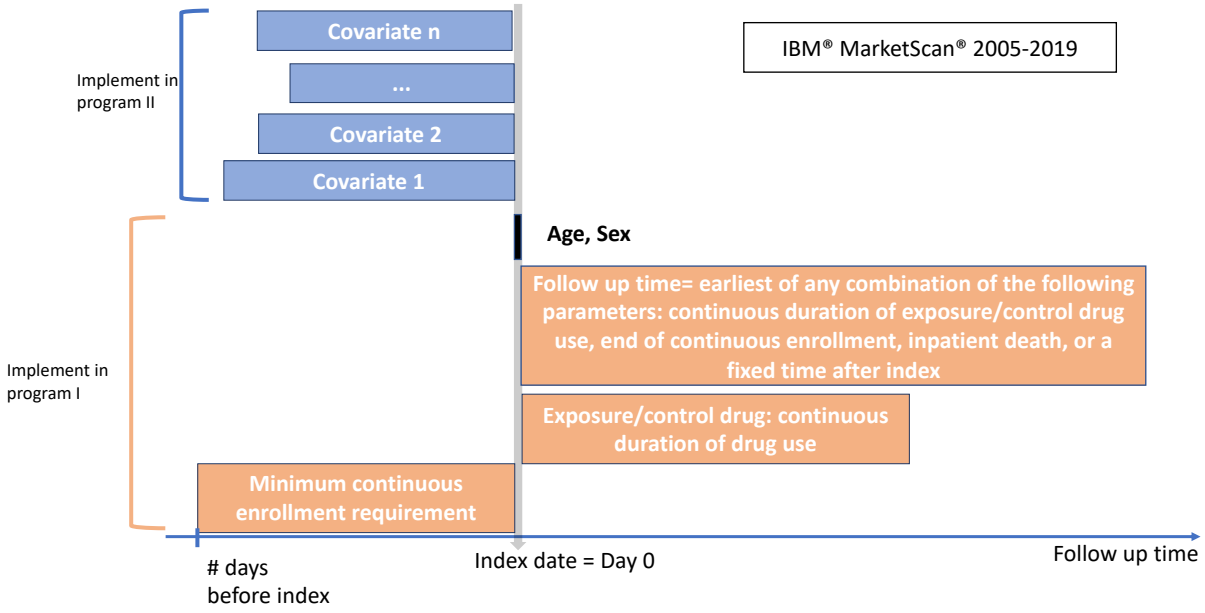
The purpose is to create a reusable program that can be applied to various research questions with a limited program modification. The Cohort Building Program can help to speed up the cohort building process. Second, the Program can be used a tool to do a simple check on the feasibility of a new research question. Last, the program can serve as a basic example to build a cohort for a new SAS programmer.

The analytical environment is SAS. Input data is the IBM® MarketScan® Commercial Claim data 2005-2019 and Redbook 2015-2019. Outputs include two datasets in SAS format.

A user manual guide

a. An overview about the cohort design and the Cohort Building Program

The Cohort Building Program uses IBM MarketScan data 2005-2019 to create a cohort study with a new user, active comparator, and no-cohort reentry design (Figure 1).



Note: Parameters in Orange and Blue boxes are defined by users (i.e. modifiable)

Figure 1. Global design diagram

The user can run the program I alone or both programs I and II (of note: program II can only run after the program I completes). Following is a detail about the fixed and modifiable parameters that are defined in the programs I and II.

Important note: the Cohort-Building Program is not suitable for all research questions. For the study design outside the scope of the program, the user may need to have hard-code modifications.

Program I:

Data source: IBM® MarketScan® Commercial Claims data 2005-2019

Exposure measurement:

The program I defines a patient into the exposure or control group if the patient has at least a fill and/or HCPCs codes related to exposure and control drug during 2005-2019, respectively. Index date is the earliest date of drug fills and/or procedure codes. The program I defines a continuous duration of exposure and control drug use with modifiable days of an allowed gap. And if there is a switching to control drug during the active days of supply of the exposure drug,

the patient will be censored at the time of switching. The same criterion is applied for the control drug.

Continuous enrollment requirement:
The user can define a number of days with continuous enrollment before and after index date.

Follow up time:
The user can specify the end of follow-up time by using one or a combination of any following criteria: end of continuous duration of exposure/control drug use, end of continuous enrollment, a fixed time after the index date, or inpatient death date, whichever comes first.

Program II:
Data source: IBM® MarketScan® Commercial Claims data 2005-2019

Outcome measurement:
A patient has the outcome if there is at least one in- or outpatient related diagnosis or procedure code during the follow-up time. The outcome date is the earliest event date.

Covariate measurement:
The program allows the user to define as many covariates as desired if the user follows the defined macro structures. These covariates can be defined by at least one in- or outpatient related diagnosis or procedure code whose window evaluations can be defined by the users. These covariates can serve as inclusion criteria, exclusion criteria, or confounders.

The program I saves the final dataset named **exposure** with the following variables: patient ID, exposure status, sex, age at index, index date, enrollment start date, enrollment end date, lookback start date, follow-up end date (Table 1).

Program II saves the final dataset named as **cohort** with the following variables: all variables in output in the program I, outcome variable, outcome date, all diagnosis code related covariates, and all procedure code related covariates (Table 1).

b. How to use the program

Once the user understands all the above characteristics, the user simply needs to search for the term “User-check-point” to find where the user needs to input their parameters as desired. There are one “User_check_point” for Program I (User_check_point_1a) and five “User-check-point” for Program II (including User_check_point_2a, User_check_point_2b, User_check_point_2c, User_check_point_2d and User_check_point_2e). At each “User-check-point”, the user needs to input all required parameters until the user reaches “User_check_point_ends” term (a detailed instruction is in Table 1). After all required parameters are input, the user needs to submit the program, and the output datasets are saved in a user-defined folder.
The program I, program II, and example programs are included in this package.

c. Further information and suggestion

The program can have further improvement. If the user has any question or suggestion, please feel free to contact: Thuy Thai, email: thuythai@ufl.edu.

Table 1. Input and output parameters with detailed instruction and examples for Program I and II

Variable name	Variable description	Instructions
Program I		
a. Define SAS library and Redbook names at "User_check_point_1a"		
truven	Libname for IBM Marketscan raw data (including data from 2005-2019)	libname truven "raw_data_location"; The user replaces raw_data_location by a directory to IBM Marketscan 2005-2019 datafiles. *For example:* *libname truven "I:";*
rb	Libname for RedBook (including Redbook 2015-2019)	libname rb "Redbook_location"; The user replaces Redbook_location by a directory to Redbook files. *For example:* *libname rb "I";*
d	Libname for a location to save the final dataset	libname d "saving_data_location"; The user replaces saving_data_location with a directory to save the final dataset. *For example:* *libname d "H:/Test";*
redbook2015 redbook2016 redbook2017 redbook2018 redbook2019	Name of the Redbook® dataset for year 2015-2019, respectively	%let redbook2015= name_Redbook_yr_2015; The user replace name_Redbook_yr_2015 by the name of Redbook® dataset for year 2015. The same logic is applied for the other years. *For example:* *%let redbook2015= redbook_14;*
b. Parameters to define exposure/control group at "User_check_point_1a"		
exporx	Exposure drug is defined by using generic name in the pharmacy encounters.	%let exporx=%str(find(gennme,"your_generic_name",'i')); The user replaces "your_generic_name" with the interested drug name. And if your exposure group includes multiple drugs, you can copy the find(gennme,"your_generic_name",'i') and use "or" to expand your drug list. *For example:* *%let exporx=%str(find(gennme,"chloroquine",'i')) or %str(find(gennme,"hydroxychloroquine",'i'));*

contrx	Control drug is defined by using generic name in the pharmacy encounters.	The same logic to the exporx is applied to contrx. *For example:* *%let contrx=%str(find(gennme,"sulfasalazine",'i'));*
expopx	Exposure drug is defined by using procedure codes in the inpatient and outpatient encounters.	%let expopx=%str('HCPCS_code'); The user replaces 'HCPCS_code' with one or more HCPCS/procedure codes to define exposure drug use. If a drug does not have an HCPCS/procedure code, the user can simply use an invalid code such as 'AAAAA' to get the code run. *For example:* *%let expopx=%str('J0390');*
contpx	Control drug is defined by using procedure codes in the inpatient and outpatient encounters.	%let contpx=%str('HCPCS_code'); The same approach to expopx is applied for contpx.
maxdaysupp	The upper limit for a day supply of a fill is considered valid	%let maxdaysupp=enter_number; The user replaces enter_number with a number, and the unit is day. *For example:* *%let maxdaysupp=180;* *The program removes fills with days of supply more than 180 days.*
gap	The number of day gap between two fills is allowed to create a continuous duration of drug use	%let gap=enter_number; The user replaces enter_number with a number, and the unit is day. *For example:* *%let gap=3;* *The program allows 3 days gap between the last day of the previous fill and the fill date of the next fill.*
c. Parameters to define continuous enrollment requirement at "User_check_point_1a"		
lookbackce	Number of days before index date that a patient has to have a continuous enrollment (both medical and pharmacy)	%let lookbackce=index-enter_number; The user replaces enter_number with a number, and the unit is day. *For example:* *%let lookbackce=index-180;* *The program requires a patient to have at least 180 days of enrollment before the index.*

lookfowardce	Number of days after index date that a patient has to have a continuous enrollment (both medical and pharmacy)	%let lookfowardce=index+enter_number; The user replaces enter_number with a number, and the unit is day. *For example: a requirement of a continuous enrollment until index date* *%let lookfowardce=index+0;*
d. Parameters to define follow up time at "User_check_point_1a"		
fu	Follow-up time from the index date. The user can allow following a patient until the earliest date of any combination of the following 4 parameters: • End of continuous exposure/control drug period (the variable name is rxend) • End of continuous enrollment (the variable name is dtend) • Date of inpatient death (deathdt) • A fixed number of days after the index date (index+number of days)	%let fu=min(fu_paramater1,fu_paramater2,fu_paramater3,fu_paramater4); The user replaces fu_parameter with at least of the 4 parameters (rxend, dtend, deathdt, index+number) For example1: %let fu=min(rxend,dtend,deathdt,index+180); The program will stop following patients at the end of continuous drug duration (rxend), end of continuous enrollment (dtend), date of inpatient death (deathdt), or 180 days after the index date, whichever comes first. For example2: %let fu=min(dtend, index+180); The program will stop following patients at the end of continuous enrollment or 180 days after the index date, whichever comes first. For example3: %let fu=min(dtend); Or %let fu=dtend; The program will follow patients until the end of continuous enrollment.
e. Output parameters for Program I		
exposure	Dataset name	
enrolid	Patient ID	
sex	Patient's sex	1: Male 2: Female
indexage	Patient's age at index date (in year)	

expo	Exposure status	1: exposure group 0: control group
index	Index date	
rxend	End date of continuous exposure/control drug duration	
dtstart	Continuous enrollment start date	
dtend	Continuous enrollment end date	
lookback	Lookback start date	
fu	End date of follow-up time	
Program II		
f. Parameters to define outcome at "User_check_point_2a"		
truven	Libname for IBM Marketscan raw data (including data from 2005-2019)	libname truven "raw_data_location"; The user replaces raw_data_location by a directory to IBM Marketscan 2005-2019 datafiles. *For example:* *libname truven "I:";*
d	Libname for a location to save the final dataset	libname d "saving_data_location"; The user replaces saving_data_location with a directory to save the final dataset. *For example:* *libname d "H:/Test";*
outcomedi9	All ICD9 diagnosis codes to define the outcome	%let outcomedi9=%str('dx_code'); The user replaces dx_code with all ICD9 diagnosis codes to define the outcome. *For example:* *%let outcomedi9=%str('4275' '798' '7981' '7982' '4271' '4274' '42741' '42742');*
outcomedi10	All ICD10 diagnosis codes to define the outcome	%let outcomedi10=%str('dx_code'); The user replaces dx_code with all ICD10 diagnosis codes to define the outcome. *For example:* *%let outcomedi10=%str('I46' 'I469' 'I472' 'I490' 'I4901' 'I4902' 'R99');*

outcomepi9	All ICD9 procedure codes to define the outcome	%let outcomepi9=%str('px_code'); The user replaces px_code with all ICD9 procedure/CPT codes to define the outcome. *For example:* *%let outcomepi9=%str('33361');*
outcomepi10	All ICD10 procedure codes to define the outcome	%let outcomepi10=%str('px_code'); The user replaces px_code with all ICD10 procedure/CPT codes to define the outcome. *For example:* *%let outcomepi10=%str('33361');*
The outcomedi9, outcomedi10, outcomepi9, outcomepi10 are "must be defined" parameters. If the outcome of interest is only defined by either diagnosis or procedure code, the user just simply enters an invalid code (e.g. "AAAAA") for unavailable diagnosis/procedure parameter to run the program.		
g. Parameters to create covariates that are defined by related diagnosis codes at "User_check_point_2a" Because the study year is in both ICD9 and ICD eras, the user must specify related ICD9 and ICD10 codes for each covariate.		
d1i9	All ICD9 diagnosis codes to define covariate d1 (of note: d1 is a covariate defined by diagnosis codes only.)	%let d1i9=%str('dx_code'); The user replaces dx_code with all ICD9 diagnosis codes to define the covariate d1. *For example:* *%let d1i9=%str('3051', '64900', '64901', '64902', '64903', '64904', '98984');*
d1i10	All ICD10 diagnosis codes to define covariate d1	%let d1i10=%str('dx_code'); The user replaces dx_code with all ICD10 diagnosis codes to define the covariate d1. *For example:* *%let d1i10=%str('F17200' 'F17201' 'F17203' 'F17208' 'F17209');*
The user can apply the template to define d2, d3, …, dn variables by specifying d2i9 and d2i10, d3i9 and d3i10, …dni9 and dni10, respectively. In order for the program to run, the user must specify at least one diagnosis covariate (i.e. d1) and all diagnosis covariates have to be in a format: d#		
h. Parameters to create covariates that are defined by related procedure codes at "User_check_point_2a"		

p1i9	All ICD9 procedure codes to define covariate p1 (of note: p1 is a covariate defined by procedure/CPT codes only.)	%let p1i9=%str('px_code'); The user replaces px_code with all ICD9 procedure/CPT codes to define the covariate p1. *For example:* *%let p1i9=%str('99406' '99407');*
p1i10	All ICD10 procedure codes to define covariate p1	%let p1i10=%str('px_code'); The user replaces px_code with all ICD10 procedure/CPT codes to define the covariate p1. *For example:* *%let p1i10=%str('99406' '99407');*
The user can apply the template to define p2, p3, …, pn variables by specifying p2i9 and p2i10, p3i9 and p3i10, …pni9 and pni10, respectively. In order for the program to run, the user must specify at least one procedure covariate (i.e. p1) and all procedure covariates have to be in a format: p#		
i. Parameters that include all diagnosis and procedure codes that are used to create the outcome and covariates To help the program run efficiently, only claims with the related diagnosis or procedure to create the outcome and covariates are pulled from the raw data.		
all_dx	All ICD9 and ICD10 diagnosis codes to define the outcome and all covariates	%let all_dx=%sysfunc(catx(&separator,&outcomedi9, &outcomedi10, &d1i9, &d1i10 /*continue to enter more if more diagnosis variables are created */)); This let statement combines all diagnosis codes included in the let statements *outcomedi9, outcomedi10, d1i9, d1i10, and etc*. The "&separator, &*outcomedi9, &outcomedi10, &d1i9, &d1i10*" is fixed and must be included as it is. If the user defines more covariate, the user must continue specifying until reaching the final covariate. *For example: if the user has the outcome variable and 3 diagnosis covariates, the user specifies the "all_dx" let statement as following:* *%let all_dx=%sysfunc(catx(&separator,&outcomedi9, &outcomedi10, &d1i9, &d1i10, &d2i9, &d2i10, &d3i9, &d3i10));*

all_px	All ICD9 and ICD10 procedure/CPT codes to define the outcome and all procedure covariates	%let all_px=%sysfunc(catx(&separator,&outcomepi9, &outcomepi10, &p1i9, &p1i10 /*continue to enter more if more procedure variables are created */)); This let statement combines all procedure codes included in the let statements *outcomedi9, outcomedi10, p1i9, p1i10, and etc*. The "&separator,&outcomepi9, &outcomepi10, &p1i9, &p1i10" is fixed and must be included as it is. If the user defines more procedure covariate, the user must continue specifying until reaching the final covariate. *For example: if the user has the outcome variable and only 1 procedure covariate, the user specifies the "all_px" let statement as following:* *%let all_px=%sysfunc(catx(&separator,&outcomepi9, &outcomepi10, &p1i9, &p1i10));*
j. Parameters in the diagnosis-code-related outcome macro at "User_check_point_2b"		
%macro vr(var,i9,i10, stdt,eddt);	The macro defines outcome and outcome date by related diagnosis codes. Outcome macro structure: %macro vr(var,i9,i10, stdt,eddt); • vr: macro name • var: output variable name • i9: a parameter to define outcome with ICD9 code • i10: a parameter to define outcome with ICD10 code • stdt: evaluation start date • eddt: evaluation end date	The user only needs to replace evaluation_start_date by the start date of the evaluation window and replace evaluation_end_date by the end date of the evaluation window in the following statement, while all other parameters in the %vr macro keep unchanging. The users can use variables that are created in the "Exposure" dataset in the Program I (e.g., index date (index) or end of follow up (fu)) to define evaluation start/end dates (please check the output of the program I for more parameters). %vr(outcomedx,&outcomedi9, &outcomedi10,evaluation_start_date,evaluation_end_date); *For example:* *%vr(outcomedx,&outcomedi9, &outcomedi10,index+1,fu);* *The program will assess the outcome from one day after the index date to the end of the follow-up. (Index and fu variables are created by the Program I).*
k. Parameters in the procedure-code-related outcome macro at "User_check_point_2b"		

%macro vr(var,i9,i10, stdt,eddt);	The macro defines outcome and outcome date by related procedure codes. The procedure related outcome macro has the same structure as the diagnosis code related outcome macro.	The user only needs to replace evaluation_start_date by the start date of the evaluation window and replace evaluation_end_date by the end date of the evaluation window in the following statement, while all other parameters in the %vr macro keep unchanging. %vr(outcomepx,&outcomepi9, &outcomepi10,evaluation_start_date,evaluation_end_date); *For example:* *%vr(outcomepx,&outcomepi9, &outcomepi10,index+1,fu);* *The program will assess the outcome from one day after the index date to the follow-up time's end. (Index and fu variables are created in Program I).*
l. Parameters in the diagnosis code related covariate macro at "User_check_point_2d"		
%macro vr(var,i9,i10, stdt,eddt);	The diagnosis code related covariate macro has the same structure as the outcome macro.	If the user wants to define d**n** variable, the user needs specify the following statement: %vr(d**n**,&d**n**i9, &d**n**i10,evaluation_start_date,evaluation_start_date); *For example: for d1 variable* *%vr(d1,&d1i9, &d1i10,index-180,index-1);*
m. Parameters in the procedure code related covariate macro at "User_check_point_2e"		
%macro vr(var,i9,i10, stdt,eddt);	The procedure code related covariate macro has the same structure as the outcome macro.	If the user wants to define p**n** variable, the user needs specify the following statement: %vr(p**n**,&p**n**i9, &p**n**i10,evaluation_start_date,evaluation_start_date); *For example: for p1 variable* *%vr(p1,&p1i9, &p1i10,index-180,index-1);*
n. Output parameters for Program II		
cohort	Final dataset name	
All output variables in Program I		

outcome	The outcome variable that is defined by diagnosis and/or procedure codes.	1: Yes 0: No
dtoutcome	Earliest event date	
d1, d2, d3,…, dn	Covariates that are defined by diagnosis codes.	1: Yes 0: No
p1, p2, p3,…, pn	Covariates that are defined by procedure/CPT codes.	1: Yes 0: No