# UTILIZING DATA FROM VARIOUS DATA PARTNERS IN A DISTRIBUTED MANNER

## Documentation of SAS Packages for the Distributed Regression Analysis Software Application

**Prepared by the Sentinel Operations Center**
**June 2018**

The Sentinel System is sponsored by the U.S. Food and Drug Administration (FDA) to proactively monitor the safety of FDA-regulated medical products and complements other existing FDA safety surveillance capabilities. The Sentinel System is one piece of FDA's Sentinel Initiative, a long-term, multi-faceted effort to develop a national electronic system. Sentinel Collaborators include Data and Academic Partners that provide access to healthcare data and ongoing scientific, technical, methodological, and organizational expertise. The Sentinel Coordinating Center is funded by the FDA through the Department of Health and Human Services (HHS) Contract number HHSF223201400030I. This project was funded by the FDA through HHS Mini-Sentinel contract number HHSF223200910006I.

## Table of Contents

# I. INTRODUCTION

The distributed regression analysis (DRA) application has two distinct software components: the computational component which we implemented using SAS and a data transferring software which allows moving data between sites in a distributed data network (DDN). In Sentinel, the DRA application has been integrated with PopMedNet™, a software application that supports automatable file transfer between the analysis center and data partners. The DRA query workflow of the automatable implementation of DRA using PopMedNet™ is described in [1]. In this document, we will focus on the SAS component of the DRA application. The computational algorithm used for linear/logistic models and Cox model will be described in a separate report.

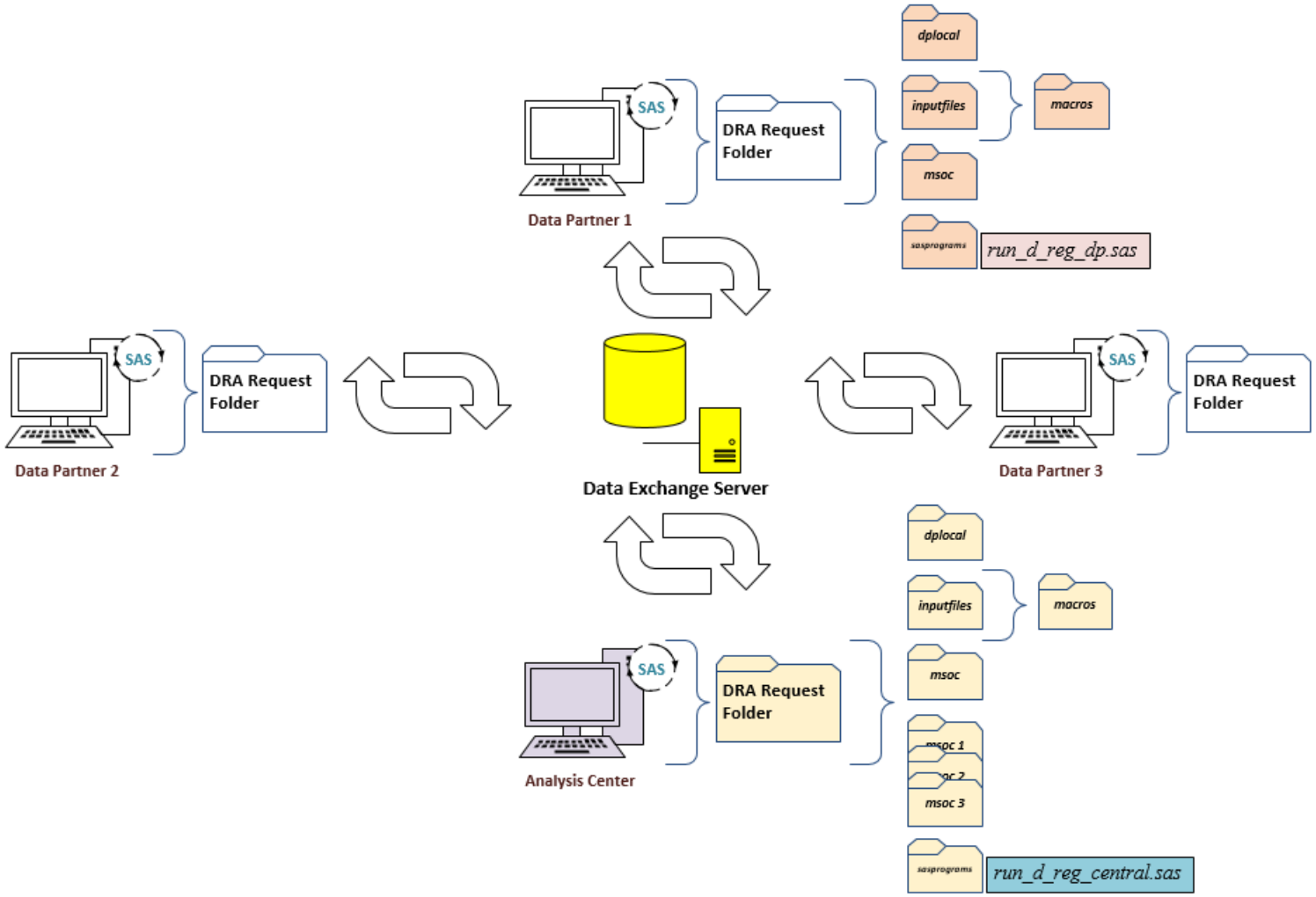
This document is organized as follows. In **Section II**, we explain how to use SAS packages at the analysis center and data partner sites to perform DRA. Specifically, we describe the structure and user-specified parameters of the master programs (wrappers) and how to execute the applications. In **Section III**, we give usage examples of the main macro ***%distributed_regression***. In **Appendix A**, we describe all parameters for the macro ***%distributed_regression***. In **Appendix B**, we describe a tracking table that stores information about execution of SAS programs at all participating site. In Appendices **Error! Reference source not found.** and **Error! Reference source not found.**, we describe output tables for linear/logistic and Cox models, respectively.

# II. HOW TO USE THE SAD-BASED APPLICATION FOR DISTRIBUTED REGRESSION ANALYSIS

## A. OVERVIEW

As an illustrative example of the SAS-based DRA application, in this section we describe a DRA in a DDN of four parties (**Figure 1**). One party was designated as the semi-trusted third-party analysis center (which can also be a data-contributing site) and the remaining three were data partners. Importantly, the SAS-based DRA application follows a master-worker model design, where the analysis center directs the DRA iterative computations, while the data partners compute the required intermediate statistics. Below we describe main steps involved in setting up and executing the SAS-based DRA application.

**Figure 1. SAS-based DRA application with a semi-trusted third party (analysis center)**



## B. CREATION OF AN INDIVIDUAL-LEVEL ANALYTIC DATASETS AT DATA PARTNER SITES

In this step, each data partner assembles an individual-level analytical dataset of the same structure. This can be done by executing a distributed program developed by the analysis center. Here we assume that this step was completed.

### 1. Directory Structures Required to Execute the SAS-based Distributed Regression Application

There are two separate SAS packages in our DRA implementation, one for the analysis center and one for the data partners. Each package employs directory structures used by the Sentinel System (**Figure 1**).[2] At the data partners, this directory structure is composed of four subdirectories: *sasprogram, inputfiles, dplocal,* and *msoc*. At the analysis center, the directory structure has additional *msoc&dp_cd* subdirectories which receive data from a corresponding data partners (*&dp_cd* corresponds to a unique data partner ID; see **Figure 1**). The *sasprogram* subdirectory contains the master wrapper program *run_d_reg_dp_templ.sas* at the data partner site and the master wrapper program *run_d_reg_central_templ.sas* at analysis center. **Table 1** summarizes the required directory structure used to organize the SAS-based DRA application and describes the role of these subdirectories.

**Table 1. SAS-based distributed regression analysis (DRA) application common directory structure**

| Directory | Data partners | Analysis center |
|---|---|---|
| *Dplocal* | File to be kept at a data partner | Intermediate permanent files |
| *sasprogram* | *run_d_reg_dp_templ.sas* | *run_d_reg_central_templ.sas* |
| *inputfiles* | Files received from the analysis center. Subdirectory for macros. | Initial input files and files to be sent to the data partners. Subdirectory for macros. |
| *Msoc* | Files to be shared with analysis center | Directory for DRA final results |
| *msoc&dp_cd* | N/A | Directories with returned summary-level data from data partners. |

### 2. SAS Package for Data Partners

To simplify the DRA process for data partners, the analysis center prepares a package as a zip file with the whole directory structure. All necessary macros are in the *macros* subdirectory, and the main SAS wrapper is located in the subdirectory *sasprograms*. Once the package is received by a data partner, it is unzipped into a pre-defined root directory to create the required directory structure in **Figure 1**. After this is done, the user at the data partner side only needs to update a few site-specific parameters (discussed below). All other site-specific parameters such as the full file path to the directories for the standard folders (*inputfiles, msoc, dplocal,* macros, etc) and the corresponding SAS libraries (*infolder*, *msoc*, *dplocal*) are derived within the wrapper template. The wrapper also has code to compile (*%include*) all necessary macros and calls ***%dp_main***, the main macro to be executed by data partners which calls other macros as necessary. This main macro has only one parameter – a data partner identifier called *dp_cd*. No regression-related parameters are specified on the data partner side. Instead, all parameters come from the analysis center in the form a SAS dataset, *vars_nm_value_pairs*, which stores pairs of parameters names and values. Below we focus on the parameters that have to be updated by users at a given data partner site and give an outline of the main wrappers structure. In the example below the root directory for all distributed regression requests at the data partner with

*dp_cd=1* is called *distrib_regr_all_requests_dp1**, the request id is called *request_1* and the full path for the request is *distrib_regr_all_requests_dp1*\request_1*\

**Main wrapper template for the data partner package, run_d_reg_dp_templ.sas**

```
/* Edit data partner numeric code dp_cd using a lookup table for data partners. Maximum length 3
digits.*/

%LET dp_cd=1;

 * Edit root directory for requests related to distributed regression;
 * This is the directory into which you unzipped the zip file with the current request;

%LET sites_prg_root_dp=\distrib_regr_all_requests_dp1\;

/* Edit directory which has the input regression dataset.
   In production it is likely to point to the full path of the sub-directory "dplocal" for the initial request
which created the analytic dataset.
 */

 %LET data_in_dir=&sites_prg_root_dp.MSReqID_for_step0\dplocal\;

/* Edit parameter min_count_per_grp to define minimum count per cell for summarized data returned
to central site.  This affects only datasets used for residual analysis and some goodness-of-fit measures
(ROC and HL statistic for logistic regression).
 It has no effect on the regression coefficients or any other statistics */

 %LET min_count_per_grp=6;
    *******                    END OF USER INPUT                    ******;

 /*The following are derived from the parameters entered above*/
/*The MSReqID is defined at the central site before distributing request to all data partners.
  Should be the same as name of the subdirectory for the request.
 All data partners participating in the given request should have the same MSReqID.;
*/

%LET MSReqID = request_1;

/* Derive full path for standard Sentinel folders */

%let DPLOCAL = &sites_prg_root_dp.&MSReqID./dplocal/;
%let MSOC = &sites_prg_root_dp.&MSReqID./msoc/;
%let INFOLDER = &sites_prg_root_dp.&MSReqID./inputfiles/;
%let SASPROGRAMS = &sites_prg_root_dp.&MSReqID./sasprograms/;
%let SASMACR=&Infolder.macros/;

/*   Other statements which include: required SAS options, assignments for SAS libraries and some
additional macro variables and the statements which include (compile) all macros in the data partner
package. */

……

* Call to main data partner macro. The macro calls all other macros as needed;

 %dp_main(dp_cd=&dp_cd.);
/*   End main wrapper for the data partner package */
```

## 3. SAS Package for Analysis Center

The main macro in the analysis center package is called %***distributed_regression***. The *run_d_reg_central_templ.sas* wrapper is used to define site-specific macro parameters (e.g., root directory for the package), compile *(%include*) necessary macros, and call the main macro %***distributed_regression***. It is also used to specify all regression-related parameters, including the dependent and independent variables, convergence criteria, the list of participating data partners, and type of regression (e.g., linear). The complete list of user-specified parameters for macro %***distributed_regression*** is described in **Appendix A**. Currently, this macro allows linear, logistic, and Cox DRA.

The wrapper for the SAS package at the analysis center has similar functionality as the wrapper at the data partner sites, but it defines additional folders and SAS libraries *msoc&dp_cd1- msoc&dp_cdN* to store the returned summary-level data from data partners (see **Figure 1**). It also calls the main macro %***distributed_regression***. Below we focus on the parameters that have to be updated by users at the analysis center and give an outline of the main wrappers structure. We describe how to use the main macro %***distributed_regression*** in **Section III**. In the example below the root directory for all DRA requests at the at the analysis center is called *distrib_regr_all_requests_central***, the request ID is called *request_1* and the full path for the request directory is \ *distrib_regr_all_requests_central* **** *request_1*\

**Main wrapper template for the analysis center package, *run_d_reg_central_templ.sas***

```
/* The parameter dp_cd (data partner code) for the central site should be always set to 0.*/
 %LET dp_cd=0;
/* Specify list of data partner codes participating in distributed regression*/
      %LET dp_cd_list=1 2 3;
 * Edit root directory for all requests related for distributed regression at the central site. ;
%LET sites_prg_root_dp=\distrib_regr_all_requests_central\;
/*The MSReqID is defined at the central site.
  Should be the same as name of the subdirectory for the request.
  Should have the same value as the one pre-specified in wrappers for data partners
  participating in the same request.;
*/
%LET MSReqID = request_1;
/* Derive full path for standard Sentinel folders */
%let DPLOCAL = &sites_prg_root_dp.&MSReqID./dplocal/;
%let MSOC = &sites_prg_root_dp.&MSReqID./msoc/;
%let INFOLDER = &sites_prg_root_dp.&MSReqID./inputfiles/;
%let SASPROGRAMS = &sites_prg_root_dp.&MSReqID./sasprograms/;
%let SASMACR=&Infolder.macros/;
/*      Other statements which include: required SAS options, assignments for SAS libraries and some
additional macro variables and the statements which include (compile) all macros in the in the package
for the central site (AC). */
……
* Call  main macro %distributed_regression. The macro calls all other macros as needed;
```

## C. EXECUTION OF SAS PROGRAMS

After necessary parameters are updated in the SAS wrappers, users at data partners and the analysis center can start the execution of their SAS programs at any time within a mutually agreed time window. All programs run continuously but the program at any data partner site goes into a waiting mode almost immediately. In this mode it constantly checks a folder *inputfiles* for the arrival of the files from the analysis center. More specifically, it checks for the existence of the trigger file *files_done.ok* which is always the last file created at any data transfer. At the analysis center, the program creates a set of files in its folder *inputfiles* which are picked up by file transferring software (e.g., PopMedNet) and transferred to the data partners via a data exchange (e.g., PopMedNet server). These files include the parameters dataset *vars_nm_value_pairs* (see **Table 2**) and files required by the DRA algorithm (e.g., a dataset with initial regression estimates).

The process then continues until the program at the analysis center issues a stop instruction to the data partners. This occurs when either the regression algorithm converges (first iteration for linear), reaches the pre-specified maximum number of iterations, or the program catches an error in the process. To stop all SAS processes, the program at the analysis center sets parameters *last_iter_in* and *end_job_dp_in* to 1. These parameters are passed to the data partners' SAS programs using the parameters dataset *vars_nm_value_pairs* **(**Error! Reference source not found.**)**. At the data partner side the condition *last_iter_in=1* and *end_job_dp_in =1* instructs the program to calculate data for final regression statistics (goodness-of-fit measures, summary of residuals, etc.), create the empty file *job_done.ok* in the *msoc* folder, and exit the data partner's SAS program. Finally, after the last batch of data from all data partners are downloaded to the analysis center, the SAS program at the analysis center performs final calculations, creates the empty file *job_done.ok* in the *inputfiles* folder, and exits SAS. This concludes the DRA process.

If the program at a data partner or analysis center side catches an error, it creates the file *job_fail.ok* instead of file *job_done.ok*. These files are used by the PopMedNet software to update status of the request on the PopMedNet portal.

**Table 2. Example of a few records from the parameters dataset *vars_nm_value_pairs*.**

| Name of macro variable | Value of macro variable |
|---|---|
| reg_ds_in | linear_karr_2005 |
| independent_vars | crim indus dis dummy_dp_var2 dummy_dp_var3 |
| dependent_vars | medv_high_flag |
| regr_type_cd | 2 |
| iter_nb | 1 |
| last_iter_in | 0 |
| end_job_dp_in | 0 |

## D. TRACKING THE DRA EXECUTION PROGRESS

To track the execution of DRA process, the SAS program at the analysis center maintains the tracking table which combines information received from each site. The tracking table has data about start and stop time of each iteration step at every site, return code and return message (pass or error message), regression convergence status and some other information. The table *&RunID._track_tbl* is located in the SAS library *infolder* at the analysis center and a copy of it can be viewed on the PopMedNet portal (parameter *&RunID* is an identifier for a given DRA execution). This allows monitoring the progress of DRA execution in real time. The complete structure of the tracking table is given in **Appendix B**.

At the end of the DRA process, the SAS program at the analysis center creates a status report *&RunID._status_and_time_rpt.pdf* in the *msoc* subfolder. The report has information about convergence status, number of iteration and SAS execution and wait times at each site. If there were errors during execution, the report also lists error message from sites in the order in which they occur.

## E. TESTING THE SAS-BASED DRA APPLICATION WITHOUT FILE TRANSFERRING SOFTWARE

The macro ***%distributed_regression*** has the testing mode in which it can run without a file transferring software on a single computer. It allows researchers to test most of the SAS code, including all code for statistical calculations, most of the code for error handling and some elements of the code necessary for data transfer. We did most of the development using this mode.

To run the macro in the test mode, the parameter *test_env_cd* has to be set to 1 (see example below). The directory structure required for this mode is the same as the one used in the production mode for the analysis center, and the template for the SAS wrapper (*run_d_reg_test_mode_tmpl.sas*) to run the program is almost the same as the wrapper for the analysis center except that it has *%include* statements to compile macros from both the analysis center and data partners. It also defines the parameter *&data_in* which has regression datasets for "data partners". The wrapper is a part of the analysis center package. The datasets corresponding to different data partners should be located in the directory *&data_in* and should be named using naming convention: *&reg_ds_in._&dp_cd* (e.g., LINEAR_KARR_2005_1, LINEAR_KARR_2005_2). Below we give an example and outline how the test mode works without a data transferring software.

Example of the call to the macro ***%distributed_regression*** in the test mode:

```
%distributed_regression(RunID=dl16
                        ,dp_cd_list=1 2 3
                        ,reg_ds_in=LINEAR_KARR_2005
                        ,dependent_vars=medv_high_flag
                        ,independent_vars= crim indus dis dummy_dp_var2 dummy_dp_var3
                        ,regr_type_cd=2
                        ,tbl_intial_est=Model_Coeff_0
                        ,test_env_cd=1);
```

The program runs continuously in a single SAS session. As in the production mode, the final statistical results can be found in the *msoc* subfolder.

Below is the brief outline of how the test mode works.

When parameter *test_env_cd=1*, the macro ***%distributed_regression*** calls the macro *%_t_loop_through_dp*. which is executed between the code that creates the files to be sent to data partners (macro *%d_reg_central_step1*), and the code that receives files from the data partners (macro *%d_reg_central_step2*). This macro is only used in the test mode. See code snippet below:

```
%d_reg_central_step1;
/*Execute distributed regression code at each data partner. Test mode only */

%IF  &test_env_cd. NE 0 %THEN %DO;
        %_t_loop_through_dp(prefix=&prefix., dp_cd_list=&dp_cd_list.);
%END;

/* Central site step 2*/
%LET prev_reg_conv_in=&reg_conv_in.;

%d_reg_central_step2 ;
```

The macro *%_t_loop_through_dp* loops through the list of "data partners" codes specified in the parameter *dp_cd_list* and for each *&dp_cd* calls the main macro in the data partner package ***%dp_main***. The macro ***%dp_main*** expects, among other thing, the variable *&msoc_dir*, which in the production mode is defined in the data partner wrapper and points to the subdirectory *msoc*. In the test mode, this parameter is instead set in the macro *%_t_loop_through_dp* and points to the subdirectory *msoc&dp_cd* for the data partner *&dp_cd*. For example, when *&dp_cd=1* the *&msoc_dir* points to the subfolder *msoc1*. As the result, the output files from the macro ***%dp_main*** for *&dp_cd=1* are written to subdirectory *msoc1* which is where the program from the analysis center expects to find them. This eliminates the need for file transferring software in the test mode.

## F. OPERATING SYSTEMS IN WHICH DRA APPLICATION CAN BE USED

The SAS packages for both analysis center and data partners can be used on any operating system on which SAS system can be installed. These include Windows, UNIX, Linux, and some others. The PopMedNet DataMart Client is a Windows® application. For this reason, most of our testing was done on Windows machines. However, we were also able to successfully test SAS programs with a data partner on the Linux server by using the following approach. We put the data partner SAS package into the directory on the Linux server that is also exposed to the Windows network as a mapped drive. This allowed PopMedNet DataMart Client to access the same file system as the SAS program.

# III. USAGE EXAMPLES OF MACRO *%DISTRIBUTED_REGRESSION*

## A. EXAMPLE DATASET FOR LINEAR AND LOGISTIC REGRESSION

The publicly available "Boston housing dataset" is used to illustrate the steps involved in using the SAS-based DRA application [3]. The dataset included 506 observations of Boston medium housing prices and housing or neighborhood characteristics. Karr and colleagues used the dataset to illustrate the theoretical capability of conducting linear DRA in a horizontally partitioned data environment [4]. To stay consistent with these authors, we also randomly partitioned the dataset into three data partners of sizes $n_1$=172, $n_2$=182, and $n_3$=152. Each dataset included the following continuous variables – housing price, crime per capita, industrialization, and distance to employment centers. Housing price served as the dependent variable for the linear DRA. For logistic DRA, we dichotomized housing price into low or high (below or above the median) and used the derived binary variable as the dependent variable. The independent variables in both models included the continuous variables crime per capita, industrialization, and distance to employment centers, and indicator variables for data partner sites. The combined dataset and datasets for data partners can be downloaded from the website: https://www.sentinelinitiative.org/sentinel/methods/utilizing-data-various-data-partners-distributed-manner.

## B. EXAMPLE DATASET FOR COX PROPORTIONAL HAZARDS MODEL

The publicly available dataset of 432 Maryland convicts followed for one year post release is used to illustrate the steps involved in using the SAS-based DRA application for Cox proportional hazards model.[5] We randomly partitioned the dataset into sizes of n1 = 134, n2 = 149, and n3 = 149. Time to re-incarceration (in weeks) was the time-to-event outcome and financial aid (a binary variable), age (a continuous variable), and number of prior convictions (a continuous variable) were the independent covariates. We added to these three datasets a variable *dp_cd* that identifies a "data partner" using values 1, 2, 3. This variable is used as stratification variable in one of the examples below. The combined dataset and datasets for data partners can be downloaded from the website: https://www.sentinelinitiative.org/sentinel/methods/utilizing-data-various-data-partners-distributed-manner.

## C. EXAMPLES OF USING MACRO %DISTRIBUTED_REGRESSION FOR LINEAR AND LOGISTIC REGRESSION

In this section we show some examples of using ***%distributed_regression***, the main macro at the analysis center. The parameters explained below should be sufficient for most practical applications. The complete list of all parameters and their descriptions can be found in **Appendix A**.

***Example 1.*** The code below runs distributed linear regression on the "Boston housing dataset" described above

```
/*
  Parameter RunID specifies an identifier for a given macro call. It is used to form a prefix
%let prefix=&RunID._ for all output datasets names.
  Parameter dp_cd_list specifies a list of data partner sites participating in the current request.
  Parameter reg_ds_in specifies the name of the input dataset for regression at a data partner site,
the name and structure the same at all sites. The dataset must be located in the SAS library data_in
defined in the data partner wrapper.
  Parameters dependent_vars and independent_vars specify dependent and independent variables
for the regression.
  Parameter regr_type_cd defines the type of the regression: 1- linear; 2- logistic;
*/

 %distributed_regression(RunID=dr1
                         ,dp_cd_list=1 2 3
                         ,reg_ds_in=LINEAR_KARR_2005
                         ,dependent_vars=medv
                         ,independent_vars=crim indus dis dummy_dp_var2 dummy_dp_var3
                         ,regr_type_cd=1
                         ) ;
```

***Example 2.*** This example runs distributed logistic regression on the same dataset and the same set of independent variables as in Example 1. It specifies two optional parameters *tbl_inital_est* and *xconv*.

```
/* Parameter tbl_intial_est names the table with initial guesses (starting values) for the regression
parameter estimates. Must have a column for each of the parameter estimates which has the same
name as the corresponding covariate. It should be located in the SAS library named infolder defined in
the analytic center wrapper. In the example below dataset Model_Coeff_0 has all initial values equal
to 0 except for the intercept which is set to average of the outcome variable. This is the same set of
the initial values that is used by default by PROC LOGISTIC on a combined dataset.

Parameter xconv specifies relative convergence criteria.
*/

 %distributed_regression(RunID=dl16
                         ,dp_cd_list=1 2 3
                         ,reg_ds_in=LINEAR_KARR_2005
                         ,dependent_vars=medv_high_flag
                         ,independent_vars= crim indus dis dummy_dp_var2 dummy_dp_var3
                  ,regr_type_cd=2
                         ,tbl_intial_est=Model_Coeff_0
                       ,xconv=1e-4) ;
```

## D. EXAMPLES OF USING MACRO %DISTRIBUTED_REGRESSION FOR COX REGRESSION

***Example 3.*** The code below runs distributed Cox regression using Breslow approximation without any stratification variables. In addition to required parameters, it specifies three optional parameters *tbl_inital_est, tbl_events_time_set* and *xconv*.

```
/*
 Parameter RunID specifies an identifier for a given macro call. It is used to form a prefix
%let prefix=&RunID._ for all output datasets names.
 Parameter dp_cd_list specifies a list of data partner sites participating in the current request.
 Parameter reg_ds_in specifies the name of the input dataset for regression at a data partner site, the
name and structure the same at all sites. The dataset must be located in the SAS library data_in defined
in the data partner master wrapper program.
 Parameters dependent_vars and independent_vars specify dependent and independent variables for
the regression.
 Parameter censoring_var specifies censoring variable.
 Parameter regr_type_cd defines the type of the regression: 1- linear; 2- logistic; 10- Cox;
 Parameter tbl_intial_est names the table with initial guesses (starting values) for the regression
parameter estimates. Must have a column for each of the parameter estimates which has the same
name as the corresponding covariate. It should be located in the SAS library named infolder defined in
the analytic center wrapper. In the example below dataset Cox_Model_Coeff_0 has all initial values
equal to 0.
 Parameter tbl_events_time_set names the table with all values of event times from all data partners.
The dataset must have one column with the same name as dependent variable. It should be located in
the SAS library named infolder. This is an optional parameter but specifying it can reduce the number of
required iterations.
 Parameter xconv specifies relative convergence criteria.
*/
 %distributed_regression(RunID=dc1
                        ,dp_cd_list=1 2 3
                       ,reg_ds_in=RECID_DR
,dependent_vars=week
,independent_vars=fin age  prio
,censoring_var=arrest
,regr_type_cd=10
,tbl_intial_est=Cox_Model_Coeff_0
                        ,tbl_events_time_set=RECID_Events_Time_Set
                        ,xconv=1e-4
                         );
```

***Example 4*.** This example runs distributed Cox regression analysis stratified by a data partner identifier variable *strata_vars=dp_cd*. In addition, it requests to use Efron approximation for ties instead of the default Breslow approximation. All other parameters were described in the Example 3.

```
/*
Parameter strata_vars specifies list of stratification variables.
Parameter ties specifies the method of handling ties in event times in Cox regression.
*/

 %distributed_regression(RunID=dc2
                              ,dp_cd_list=1 2 3
                             ,reg_ds_in=RECID_DR
                              ,dependent_vars=week
                             ,independent_vars=fin age prio
                             ,censoring_var=arrest
,regr_type_cd=10
                            ,strata_vars=dp_cd
                       ,ties=EFRON
                             );
```

## E. CREATION OF OUTPUT TABLES

The macro %***distributed_regression*** creates final output datasets in the subdirectory *msoc* of the request directory at the analysis center. All datasets from a given execution of the macro have the same prefix determined by the parameter &*RunID*. The structure of most of these datasets was modeled on the corresponding datasets generated by PROC REG (linear), PROC LOGISTIC/GENMOD (logistic) and PROC PHREG (Cox model). The complete list of output datasets and their description is given in the **Appendix C** for linear and logistic regression and in the **Appendix D** for Cox regression.

# IV. APPENDICES

## A. APPENDIX A: PARAMETERS FOR THE MAIN MACRO *%DISTRIBUTED_REGRESSION*

The table below describes all parameters for the macro ***%distributed_regression*** used in distributed linear, logistic, and Cox regression.

| Parameter | Description |
|---|---|
| RunID | Identifier for a given macro call. It is used to form a prefix &RunID for the names of all output datasets. This allows calling this macro several times within the same distributed regression request. Preferably less than 4 characters. *Required*.<br>*Example: RunID=dr1* |
| reg_ds_in | The name of the input analytic dataset. The dataset must have the same name at all data partner sites and located in the SAS library called DATA_IN (defined in the data partner's SAS wrapper). *Required.*<br>*Example: reg_ds_in=LINEAR_KARR_2005* |
| dp_cd_list | The list of participating data partners separated by space. *Required.*<br>*Example: dp_cd_list=7 15 19* |
| regr_type_cd | Defines the type of the regression.<br>1=linear; 2=logistic; 10=Cox. *Required.*<br>*Example: regr_type_cd=1* |
| dependent_vars | Name of the dependent variable in the regression. *Required. Example: dependent_vars= medv* |
| independent_vars | List of the independent variables in the regression. *Required. Example: independent_vars=crim indus dis* |
| NOINT | When set to NOINT the regression is fit without intercept. Default is blank which fits the model with the intercept. Not relevant for Cox regression. *Optional.* |
| Freq | Names the variable that indicates frequency of an observation. *Optional.*<br>*Example: freq=freq_variable* |
| Weight | Names the variable that indicates weight of an observation. *Optional.*<br>*Example, weight=weight_variable* |
| tbl_intial_est | Names the table with initial guesses of the regression parameter estimates. Must have a column for each of the parameter estimates. Names of columns for parameter estimates should be the same as names of corresponding independent variables. (Same structure as special SAS dataset of the TYPE=PARM). If not specified, all initial values are set to 0. Note that this is different from the default in PROC LOGISTIC, which sets all initial values to 0 except for the intercept, which is set to the average of the outcome variable. The dataset *tbl_intial_est* should be located in the SAS library infolder defined in the wrapper for the analysis center. Not relevant for linear regression. *Optional.*<br>*Example: tbl_intial_est=Model_Coeff_0* |
| xconv | Relative convergence criteria. The same definition as the one used by SAS. *Optional.* Default is 1E-4. |
| max_iter_nb | Maximum number of allowed iterations. *Optional*. Default is 20. |

| Parameter | Description |
|---|---|
| alpha | Level of statistical significance. *Optional.* Default is 0.05. |
| groups | Number of groups used in the calculation of residuals summary statistics. *Optional*. Default is 10. |
| wait_time_min | Minimum time interval for checking of the trigger file *files_done.ok*. Measured in seconds. Used by the macro *%file_watcher*. *Optional*. Default is 3. |
| wait_time_max | Maximum time interval for checking of the trigger file *files_done.ok*. Measured in seconds. Used by the macro *%file_watcher*. *Optional*. Default is 7,200, which is 2 hours |
| last_runid_in | If one wants to run more than one regression (can be different regression models) within the same request one should specify *last_runid_in=0* for the first few calls of this macro and to 1 for the last call. *Optional*. Default is 1. |
| test_env_cd | Set to 1 to run in the special development/testing environment. The directory structure in this environment is the same as the one at the analysis center. When set to 1 the program can be executed within a single SAS session with the code for different data partners running sequentially. It allows testing of most of the SAS code without the need for data transferring software. *Optional*. Default is 0 which means production environment. |
| max_numb_of_grp | Sets upper limit to the number groups for summarized data returned to the analysis center. Normally the number of groups is determined by parameters *min_count_per_grp* or *min_count_per_grp_glob*. However, for large datasets this can result in large amount of data transferred from data partners to the analysis center. This is often unnecessary and this parameter puts a cap on the number of rows returned to the analysis center. *Optional*. Default is 10,000. |
| min_count_per_grp_glob | Sets minimum count per cell for summarized data returned to the analysis center. It is only used if a data partner site does not specify parameter *min_count_per_grp* in their master program. This affects datasets used for residual analysis and goodness-of-fit measures (receiver operating characteristic [ROC] curve and Hosmer-Lemeshow statistic for logistic regression). *Optional*. Default is 6. |

## B. APPENDIX B: TRACKING TABLE

The table below describes columns in the tracking table created by the macro %***distributed_regression***. The table is located in the *infolder* SAS library. It is named *&RunID._track_tbl* where *&RunID* is an identifier for a given macro call of %***distributed_regression***. For example, for *&RunID=dr1*.

| Variable name | Variable description |
|---|---|
| dp_cd | The column *dp_cd* is an integer that identifies a data partner. For the analysis center it is always set to *p_cd=0*. |
| MSReqId | The column stores the request ID which is set when the request is created by the analysis center. It also determines the name of the subfolder for the request. For example, *MSReqID=request_1*. |
| RunID | The column identifies a specific call to the macro %***distributed_regression*** For example, *RunID=dr1* |
| ITER_NB | The column identifies iteration number. |
| STEP_NB | The column identifies step within an iteration. At the analysis center (dp_cd=0) there are two steps for each iteration. In the first step SAS program prepares datasets to be sent to data partners while at the second step the analysis center receives data from data partners. At a data partner site there is only one *STEP_NB=1* for each iteration. |
| START_DTM | Start time of the iteration step at a site. The time is local for a given site. |
| END_DTM | End time of the iteration step at a site. The time is local for a given site. |
| CURR_STEP_IN | Indicate iteration step that currently processing. |
| STEP_RETURN_CD | Return code from iteration step at a site. Value *STEP_RETURN_CD =0* indicate successful competition of the step. |
| STEP_RETURN_MSG | Return message. When there is an error the column will store the error message. Otherwise it has value "PASS". |
| REG_CONV_IN | Indicate if the convergence was achieved at a given iteration. |
| REG_CONV_MSG | The value is "SUCCESS" when *REG_CONV_IN=1*. Otherwise it is blank. |
| LAST_ITER_IN | Indicate if the iteration is the last in regression process and the DRA process switched to calculation of final statistics and residual analysis. |
| LAST_RUNID_IN | When one runs more than one regression (can be different regression models) within the same request the value *LAST_ITER_IN=0* for the first few calls of this macro and *LAST_ITER_IN=1* for the last call. |
| UTC_OFFSET_DISPLAY | Local time offset relative to Coordinated Universal Time in formatted form. For example for EST it is -04:00. |
| UTC_OFFSET_SEC | Local time offset relative to Coordinated Universal Time in seconds. |
| REGR_TYPE_CD | Type of regression. 1-Linear; 2-Logistic; 10-Cox regression |

## C. APPENDIX C: FINAL OUTPUT DATASETS FOR LINEAR AND LOGISTIC DRA

Below is a table with the list and description of final output datasets created by the macro %***distributed_regression*** for distributed linear and logistic regressions. All datasets are located in the subdirectory *msoc* at the analysis center. The datasets from a given run have the same prefix equal *&RunID*. For example, for *&RunID=dr1* the *&prefix=dr1_*.

| Dataset Name | Dataset Description |
|---|---|
| &PREFIX.ANOVA | Has the same structure as the ODS table *ANOVA* generated by SAS PROC REG. Includes standard statistics for analysis of variance: various sum of squares, F-value and corresponding p-value. Used for analysis of linear regression only. |
| &PREFIX.COLLINDIAG | Has the same structure as the ODS table *CollinDiag* generated by SAS PROC REG. Provides collinearity diagnostic between independent variables. Since it involves only independent variables, it useful for collinearity diagnostic of both linear and non linear models. |
| &PREFIX.CONVRG_STATUS | Has similar structure as the ODS table *ConvergenceStatus* generated by SAS PROC GENMOD. In addition to convergence status, it has information about number of iteration and convergence criteria. Not applicable to linear model. |
| &PREFIX.COV_EST | Has the same structure as the ODS table *CovB* generated by SAS PROC REG and PROC LOGISTIC. Includes information about model-based covariance of estimates. Applicable to all models.. |
| &PREFIX.HC_COV | The same structure as the table *&PREFIX.COV_EST* above but has the information about covariance of estimates based on the robust sandwich estimator (heteroscedastic covariance). Applicable to all models. |
| &PREFIX.GLOB_NULL_CHISQ | Has the same structure as the ODS table *GlobalTests* generated by PROC LOGISTIC. Includes Chi-Square statistic, degrees of freedom and p-value for test of global null hypothesis. Not used for linear model which uses F-test (table *&PREFIX.ANOVA*). |
| &PREFIX.HL_CHISQ | Has structure similar to the ODS table *LackFitChiSq* generated by SAS PROC LOGISTIC. Includes information about Hosmer-Lemeshow chi-square test results. |
| &PREFIX.HL_PARTITION | Has the same structure as the ODS table *LackFitPartition* generated by PROC LOGISTIC. Includes information about partition for the Hosmer-Lemeshow test. |
| &PREFIX.INVXPX | Has inverse of the negative Hessian matrix. For linear regression it is the same as inverse of the matrix **X'X**. Applicable to all models. |
| &PREFIX.ITER_PARMS_HIST | Has the same structure as the ODS table *IterHistory* generated by PROC LOGISTIC. Includes information about iteration history. Not applicable to linear model. |
| &PREFIX.MODELFIT | Has structure similar to the ODS table *ModelFit* generated by SAS PROC GENMOD. It has information about various goodness-of-fit measures including AIC, AICC, BIC, R-square. For non-linear regression the R-square represents generalized R-square (coefficient of determination). Applicable to all models. |

| Dataset Name | Dataset Description |
|---|---|
| &PREFIX.MODEL_COEFF | Has the same structure as the output dataset specified by option *OUTEST* in SAS PROC REG/PROC LOGISTIC. Includes information about regression coefficient (parameter estimates) in the longitudinal form: a single row with a column for each of the parameter estimates. |
| &PREFIX.P_EST_HC | Has the same structure as the ODS table *ParameterEstimates* generated by PROC REG. Includes information about regression coefficient (parameter estimates) in the vertical form with separate row for each parameter. In addition, it has columns for model and robust standard errors, p-values, upper and lower confidence limits. Applicable to all models. |
| &PREFIX.P_EST | Has the same structure as *&PREFIX.P_EST_HC* but without columns based on robust sandwich estimator. Applicable to all models. |
| &PREFIX.RESID_SUM | Has overall sum and/or mean values for a number of quantities including observed and predicted outcome, residuals, square of residuals, log likelihood and various goodness-of-fit measures. Applicable to all models. |
| &PREFIX.RESID_SUM_BY_PCT | Has summary statistics based on final output dataset at each data partner. The data are grouped by percentiles of predicted values. The number of observation per group for a data partner can vary slightly due to ties. The number of groups is determined by the macro parameter *groups* specified in the main macro ***%distributed_regression***. The default value is *groups=10*. The summary statistics include mean value for observed and predicted outcome, residuals, square of residuals and model variance. It also includes number of observations per group and a number of distinct values of predicted outcome. The dataset can be used to generate various plots and visually evaluate the goodness-of-fit by the regression model. |
| &PREFIX.RESID_SUM_BY_PCT2 | Has the same structure as *&PREFIX.RESID_SUM_BY_PCT* but with the number of groups determined by the parameter *min_count_per_grp* specified by the data partner. If the parameter *min_count_per_grp* is not specified by a data partner then the parameter *min_count_per_grp_glob*, specified in the macro ***%distributed_regression***, is used. The dataset provides the most detailed level of summarization allowed by each data partner. It is used for approximate calculation of the ROC curve, AUC, and Hosmer-Lemeshow statistic. It can also be used for residual analysis. Applicable to all models. |
| &PREFIX.ROC | It has the same structure as the output dataset specified by option *OUTROC* in PROC LOGISTIC plus a column with the value of AUC characteristic. Includes data necessary for plotting ROC curve. Applicable to logistic regression only. |

## D. APPENDIX D: FINAL OUTPUT DATASETS FOR COX DRA

Below is a table with the list and description of final output datasets created by the macro %***distributed_regression*** for distributed Cox regression. All datasets are located in the subdirectory *msoc* at the analysis center. The datasets from a given run have the same prefix equal *&RunID*. For example, for *&RunID=dc1* the *&prefix=dc1_* .

| Dataset Name | Dataset Description |
|---|---|
| &PREFIX.BASELN_HAZARD | The dataset has an estimate of cumulative baseline hazard function at the event times of each stratum. The non-stratified case is treated as the case with single strata. |
| &PREFIX.BASELN_SURVIVAL | The dataset has an estimate of survival function at the event times of each stratum. The survival function is evaluated at average values of covariates per stratum. The non-stratified case is treated as the case with single strata. |
| &PREFIX.CENS_SUM | Has similar structure as the ODS table *CensoredSummary* generated by SAS PROC PHREG. It has summary of event and censored observations. |
| &PREFIX.CONVRG_STATUS | Has similar structure as the ODS table *ConvergenceStatus* generated by SAS PROC PHREG. In addition to convergence status, it has information about number of iterations. |
| &PREFIX.COV_EST | Has the same structure as the ODS table *CovB* generated by SAS PROC PHREG. Includes information about model-based covariance of estimates. |
| &PREFIX.GLOB_NULL_CHISQ | Has the same structure as the ODS table *GlobalTests* generated by PROC PHREG. Includes Likelihood Ratio, Chi-Square statistic, degrees of freedom and p-value for test of global null hypothesis. |
| &PREFIX.ITER_PARMS_HIST | Has the same structure as the ODS table *IterHistory* generated by SAS PROC PHREG. It includes information about iteration history. |
| &PREFIX.MODELFIT | Has structure similar to the ODS table *ModelFit* generated by SAS PROC PHREG. It has information about various goodness-of-fit measures including -2Log L (L is likelihood), AIC, BIC criteria. |
| &PREFIX.MODELINFO | Has the same structure as the ODS table *ModelInfo* generated by SAS PROC PHREG. It includes information about names of the input dataset, dependent variable, censoring variable, and type of approximation for handling event ties (Breslow or Efron). |
| &PREFIX.MODEL_COEFF | Has the similar structure as the output dataset specified by option *OUTEST* in PROC PHREG. Includes information about regression coefficient (parameter estimates) in the longitudinal form: a single row with a column for each of the parameter estimates. |
| &PREFIX.P_EST | Has the same structure as the ODS table *ParameterEstimates* generated by PROC PHREG. Includes information about regression coefficient (parameter estimates) in the vertical form with separate row for each parameter. In addition, it has columns for model standard errors, p-values, upper and lower confidence limits. |
| &PREFIX.RESID_SUM | The dataset has overall summaries for a number of quantities including number of events, log likelihood and various goodness-of-fit measures. |

| Dataset Name | Dataset Description |
|---|---|
| &PREFIX.RESID_SUM_BY_PCT | Has summary statistics based on final output dataset at each data partner. The data are grouped by percentiles of linear predictor $\theta_j = \boldsymbol{\beta}^T \boldsymbol{Z}_j$ values. The number of observation per bin for a data partner can vary slightly due to ties. The number of bin is determined by the macro parameter *groups* specified in the main macro %***distributed_regression***. The default value is *groups=10*. The summary statistics include mean values of linear predictor, martingale and deviance residuals. It also includes data partner identifier variable *dp_cd* and a number of observations per bin. The dataset can be used to generate plots of martingale and deviance residuals and visually evaluate the goodness-of-fit by the regression model. |

# V. REFERENCES


1. Her Q, Malenfant J, Malek S, et al. A query workflow design to perform automatable distributed regression analysis in large distributed data networks. EGEMS (Washington, DC) In Review;**4**(2):1213 doi: 10.13063/2327-9214.1213[published Online First: Epub Date]|.
2. Sentinel System. Routine Querying System. Secondary Routine Querying System 2018. https://www.sentinelinitiative.org/sentinel/surveillance-tools/routine-querying-tools/routine-querying-system.
3. Harrison D, Rubinfeld DL. Hedonic housing prices and the demand for clean air. Journal of environmental economics and management 1978;**5**(1):81-102
4. Karr AF, Lin X, Sanil AP, Reiter JP. Analysis of Integrated Data without Data Integration. Chance 2004;**17**(3):26-29 doi: 10.1080/09332480.2004.10554910[published Online First: Epub Date]|.
5. Rossi PH, Henry JP. Seriousness: A measure for all purposes. Handbook of criminal justice evaluation 1980:489-505