The stratify macro

Klaus Rostgaard

10 August 2013

1 What is this?

This is a SAS style documentation of the SAS macro %stratify. The macro user interface resembles a SAS procedure and it is documented accordingly. This document (stratifymanual.pdf) is available from Sourceforge [1] and StatLib [2]. There it is situated together with pyrsstep.sas which contains the macro itself and some example code.

Contents

2 Overview: %stratify

2.1 What does %stratify do?

%stratify is useful when modelling hazard rates that depend on time-varying covariates. %stratify take input data sets in specific formats representing potential time intervals at risk, outcome events and exposure events and generates an output data set containing aggregated number of events and time at risk in strata characterised by combinations of the values of the time-varying covariates. This output is suitable for e.g. Poisson regression. Alternatively an unaggregated data set may be created, with entry and exit times and a censoring variable suitable as input for e.g. Cox-regression. All the most common types of

time-varying covariates can be generated and grouped by %stratify. %stratify can simultaneously generate similar output for different outcomes. The output can be extended by supplying a rate data set, in which case %stratify calculates and adds the expected number of events for aggregated output or adds the relevant rates for unaggregated output. We assume follow-up of known persons, but it could be any enumerated set of objects.

2.2 What types of input data sets are required by %stratify?

The basic input data set to %stratify contains information about who are followed-up when. Each record in this data set must contain the variables entry and exit designating start and stop of follow-up. Typically this data set will contain further information characterising the follow-up interval, such as the birthdate and sex of the person being followed-up. The exit time used in this data set supplies a maximal exit time, %stratify may modify the real exit time according to the occurrence of outcome events and the type of analyses intended. To suffice for analysis on its own this data set must also contain a variable designating the time of occurrence of outcome events. %stratify also offers the possibility of having the characteristics of the potential outcome events (such as time of occurrence and type of event) in an external data set, which is then linked to the base data set by %stratify, using a person-identifying variable present in both data sets. By the same token it is possible to extend the exposure history contained in the base data set by having %stratify link it to one or more exposure data sets regarding so-called zero-rate time factors. A zero-rate time factor is defined by being constant between status-changes. Typical examples are: marital status, place of residence, number of pregnancies. Each record in these exposure data sets must contain the person-identifying variable, the time of occurrence of the status change and a character variable signalling which status is changing. They may also contain a numeric and/or character variable containing attributes of the exposure event. When using %stratify to generate data sets regarding several outcomes at a time, these outcomes have to be specified in an external data set. This external data would often be a data set containing rates, which is needed anyway if you intend to enhance your output data set with background disease rates or expected numbers of events.

2.3 What types of output does %stratify produce?

%stratify produces a data set in one of two flavours for statistical analyses, and optionally an additional data set characterising each outcome event according to the cross-classifying variables in the above data set possibly enhanced with information in the input data sets, e.g. the identity of the person having the outcome event. The standard flavour of output data set from %stratify is an event-time table containing aggregated number of events and time at risk in strata characterised by combinations of the values of the time-varying covariates (including constants) specified in %stratify, e.g. sex and age in 5-year age groups. The number of events and time at risk are put in variables called events and pyrs, respectively. This output is suitable for Poisson regression and other methodology based on piece-wise constant hazard rates. Alternatively an unaggregated data set may be created, with records containing entry and exit times

and a censoring variable for each individual time segment in strata characterised as above. This data set yields input suitable for e.g. Cox-regression. If a rate input data set is supplied an extra variable is added to the output: expected number of events for the event-time table and the rates for the unaggregated output.

3 Syntax: %stratify

```
%stratify( <options> ;
 AXIS variable origin= cuts= <speed=> ;
 CLASS variable(s) ;
 DROP variable(s) ;
 EVENTID variable(s) ;
 EXPDATS SAS-data-set(s) ;
 SRTF variable speed= cuts= <time=> <value=> ;
 ZRTF variable value= condition= <n=> <groups=> <missing=> <length=> ;
 );
```

There can be any number of each statement in any sequence, but for CLASS, DROP, EVENTID and EXPDATS only the last version is in effect. The last statement (before the ending paranthesis) need not be terminated with ";". As in SAS procedures you can make a statement into a comment by prefixing it with "*", and add comments in the usual way and the base input data set defaults to &SYSLAST and the basic output data set defaults to the value of the base input data set.

Task	Statement
Generate a stratified version of a unity-rate time factor	"AXIS statement" on page 12
Stratify data by variables in the base input data	"CLASS statement" on page 12
Drop and aggregate over variables only needed for linkage with a rate data set	"DROP statement" on page 13
Specify additional variables characterising each outcome event	"EVENTID statement" on page 13
Specify a list of auxillary input data sets containing exposure information in a particular format	"EXPDATS statement" on page 14
Generate a stratified version of a simple-rate time factor	"SRTF statement" on page 14
Generate a zero-rate time factor from information held in the EXPDATS data sets	"ZRTF statement" on page 16

3.1 %stratify options

```
%stratify( <options> ;
```

Task	Option
Specify the base input data set	DATA=
Specify the base output data set	OUT=
Specify the external outcome events input data set	OUTCOMES=
Specify the outcome events output data set	EVENTDAT=
Specify the external input rate data set	RATES=
Specify the types of outcome to be analysed with more than one outcome	EVENTVALUES=
Specify the "type of outcome" variable for analyses with more than one outcome	EVENTTYPE=
Specify the value of the "type of outcome" variable denoting follow-up time	NOEVENTVALUE=
Specify the outcome event time variable	EVENTTIME=
Specify the exposure event time variable	TIME=
Specify the person ID variable for record linkage between input data sets	SUBJECT=
Specify the numeric exposure event attribute variable in the EXPDATS data sets	VALUE=
Specify the character exposure event attribute variable in the EXPDATS data sets	VALUEC=
Specify the character variable containing the exposure event	EXPEVENT=
Specify the birthdate variable	BIRTHDATE=
Specify the scale relation between time units for time points in input data and cut points/PYRS	SCALE=
Specify the overhang: mostly the follow-up terminates at EVENTTIME+GRANULARITY	GRANULARITY=
Specify when and how follow-up terminates	MODE=
Specify the algorithm for data aggregation or choose non-aggregated output data	METHOD=
Specify the value of a parameter in one of the data aggregation algorithms	CHUNKSIZE=
Specify whether further processing should occur after stratification and aggregation	COMPLETE=
Specify the origin of the underlying time scale for Cox regression	COXORIGIN=

BIRTHDATE
Specifies the numeric variable containing the birthdate of the person in the base input data set. It must be on the same time scale as entry and exit.
Interaction: Only needed if you use a ZRTF statement with VALUE=a, i.e. when calculating the age at the time of some exposure event.
Default: birthdate.

CHUNKSIZE
A strictly positive integer-valued numeric parameter needed if you use the chunk

algorithm for aggregation, see section 4.3 on page 20 for details.
Interaction: Only needed when METHOD=chunk.
Default: 5000.

COMPLETE
Specify whether further data processing should occur after generating the output data set from the stratification and aggregation step. Valid values for COMPLETE is yes, y, no, n. When MODE=single and METHOD=fst,sum,arr,chunk and no RATES are specified COMPLETE=yes and COMPLETE=no is the same thing. Incomplete processing may be useful when generating very large data sets for several outcomes, because it allows you to store a comparatively small data set data set to be processed for each outcome at the time of analysis as opposed to storing one very large output data set ready for analysis using BY-processing by the various outcomes. See examples 2, 3, 9 and 10 on pages 22, 24, 36 and 39 for comparisons. For MODE=competing or intensity you may prefer to have a data set with differently named (number of) outcomes, rather than having essentially a copy of the data set for each outcome for by-processing. If so, this is also made directly and efficiently from the output data set obtained using COMPLETE=no, both for aggregated and unaggregated output.
Default: yes.

COXORIGIN
Specify the origin and thereby implicitly the primary time scale for a Cox regression on unaggregated data. Thus entry from DATA are transformed into entry:=(entry-COXORIGIN)/SCALE and similarly for the end points of all the time segments generated by stratifying the input data. These new entry and exit variables can then be used for counting-process style input to a Cox regression, see example 2 on page 22. COXORIGIN must be a numeric variable present in DATA or generated by ZRTF statements or a numeric constant. If it is a variable it will automatically be added to the output.
Interaction: COXORIGIN is ignored unless COMPLETE=yes and METHOD=noagg.

DATA
The mandatory base input data set. It must contain the numeric variables entry and exit, containing the time of start and stop of follow-up, where the latter is the maximal (potential) stopping time, which may be cut shorter by the occurrence of an outcome event. It may have to contain further variables depending on the setting of other options and statements. It does not cause problems if multiple or overlapping time segments for the same person appears in this data set. This may be the case when the same person is part of different cohorts that you want to analyse. %stratify will still link each contribution correctly with other (non-duplicated!) external data (OUTCOMES and EXPDATS data sets).

EVENTDAT
An optional extra output data set, containing one record for each outcome event occurring during follow-up. It contains the cross-classifying variables from the ordinary output data set, together with optional extra variables specified in the EVENTID statement. %stratify ignores variables in the EVENTID statement that are not available from the input data sets or calculated in a ZRTF, BRTF or AXIS statement. See example 4 on page 26.

Interaction: If EVENTDAT is unspecified, any EVENTID statement is ignored.

EVENTTIME
EVENTTIME specifies the numeric variable containing the time of outcome events. This variable is mandatory and must reside in the base input data set or the external outcome events input data set.
Interaction: The variable must reside in the external input data set specified by OUTCOMES=... when MODE=multiple or intensity. If you use the OUTCOMES option the variable will exclusively be fetched from the external input data set, otherwise it will exclusively be fetched from the base input data set.
Default: eventtime.

EVENTTYPE
EVENTTYPE specifies the variable containing the type of outcome event. It must be used when you want to generate output data for several different outcomes simultaneously. The output data will then also become stratified by the EVENTTYPE variable.
Interaction: The EVENTTYPE option must be used when MODE=intensity, multiple or competing. When used you must also use the NOEVENTVALUE option. When MODE=intensity or multiple the EVENTTYPE variable must reside in the data set specified with the OUTCOMES option.
Default: eventtype.

EVENTVALUES
EVENTVALUES specifies a data set that contains all the values of EVENTTYPE that you are going to generate data for. I.e. if you intend to generate data for three types of cancer, then these three and only these three types of cancer should be represented in the variable EVENTTYPE in the data set EVENTVALUES. By default EVENTVALUES=RATES.
Interaction: The EVENTTYPE option must be used when MODE=intensity, multiple or competing. The EVENTTYPE option is ignored when MODE=single. If MODE=intensity, multiple or competing and both EVENTVALUES and RATES are specified they are co-dominant: a value of EVENTTYPE has to be present in both data sets to force it to appear in the output.
Default: RATES. If not specified then OUTCOMES. If not specified then DATA.

EXPEVENT
EXPEVENT specifies the name of the character variable used in all of the EXPDATS data sets to signal what exposure event has happened through the content of the EXPEVENT variable. See section 3.8 on page 16 for details.
Interaction: The EXPEVENT option must be used whenever you use ZRTF statements.
Default: expevent.

GRANULARITY
You have to be at risk to have an event happen to you. To ensure that there is risk time with the same covariate values as characterises the event, you must let the follow-up extend a little beyond the (infinitesimal) time point of occur-

rence of the outcome event. This overhang is specified in the GRANULARITY option. It must be a strictly positive numerical constant. Any overhang past exit for particular records are ignored. For further details see section 4.1 about timing on page 18.
Interaction: The GRANULARITY option only has effect when MODE=single, competing or multiple and at the same time METHOD not noagg.
Default: 1.

METHOD
The possible values for METHOD is fst, arr, sum, chunk and noagg. The first four values refers to different algorithms for aggregating the variables pyrs and events over the cross-classifying variables. The default method is generally the fastest, but if it requires too much memory (RAM) choose another, see section 4.3 on page 19 for details. Choosing METHOD=noagg will cause the output to be unaggregated with records containing entry and exit times for each individual time segment in strata defined by the cross-classifying variables. If you use COMPLETE=no then time intervals at risk and events are put in separate records, the latter category distinguished by having a missing value of exit. Further processing of this data set yields input suitable for e.g. Cox-regression, see example 10 on page 36. If you use COMPLETE=yes this output is automatically transformed into time segments characterised by entry and exit times and the censoring variable censored. If an outcome event occurs, the time segment with exit=EVENTTIME for the relevant person will have the censored variable set to 0, while later time segments, including the one with the event itself are discarded. If an event didn't occur censored equals 1. See example 2 on page 22.
Default: fst.

MODE
MODE determines the mode of follow-up. There are four valid values with various abbreviations:
s=sing=single
m=mult=multiple
c=comp=competing
i=intens=intensity.
MODE=single is the default. Here we await a single outcome, and follow-up is terminated thereafter by %stratify. When MODE=competing we await the first occurrence of one of a set of outcome events, and follow-up is terminated thereafter, but here we also need to record the type of outcome event as specified with the EVENTTYPE option. This output is suitable for competing risks analysis. MODE=multiple allows the recording of the first occurrence of each of several outcomes, typed by the EVENTTYPE variable. Follow-up after the occurrence of an outcome event of a certain type is then marked with the relevant value of the EVENTTYPE variable, so that these contributions may later be subtracted from the relevant analysis data set, as is done automatically when using COMPLETE=yes. Finally MODE=intensity allows any number of outcome events, typed by the EVENTTYPE variable, to occur within the original follow-up intervals. It has no effect on the follow-up intervals. For examples of all modes, see examples 2 and 3 on page 22 and page 24. If you want to prepare data for analysis with e.g. an Andersen-Gill model [6] or other model

for recurrent events [7], you can use MODE=i, see example 8 on page 34.
Interaction:

- When MODE=single, competing or multiple the GRANULARITY option is in effect if method is not NOAGG.
- When MODE=intensity, competing or multiple the EVENTTYPE option and the NOEVENTVALUE option is in effect.
- When MODE=competing and the OUTCOMES option is used, %stratify uses the first outcome event for each person, after having sorted a copy of the outcome events input data set according to SUBJECT, EVENTTIME, EVENTTYPE.
- When MODE=multiple %stratify uses the first outcome event in each event category for each person, after having sorted a copy of the outcome events input data set according to SUBJECT, EVENTTYPE, EVENTTIME.
- When MODE=single and you use the OUTCOMES option %stratify uses the first outcome event for each person, after having sorted a copy of the outcome events input data set according to SUBJECT, EVENTTIME.
- When MODE=intensity, a copy of the records in the OUTCOMES data set is pruned of multiple occurences of the same eventtype in the same person at the same time.

Default: single.

NOEVENTVALUE
The value assigned to the EVENTTYPE variable to designate risk-time contributions. Needed when MODE=competing, intensity, or multiple. It is not neccessary for you to set this value but you may prefer to control this value yourself when using COMPLETE=no, as it will then appear in the output. If the EVENTTYPE variable is a character variable you should enclose the value in double quotes, and avoid using special characters that may interfere with %stratify. There is a hierarchy of default values that will be checked against the set of values for EVENTTYPE that you actually use. If all the proposed default values are in use NOEVENTVALUE will be set to ceil(max(of EVENTVALUES)+1) for numeric EVENTVALUES or byte(rank(substr(max(EVENTVALUES),1,1))+1) for character EVENTVALUES.
Interaction: If the EVENTTYPE variable is a character variable it's length will be determined from the length of the variable in input data. Accordingly, if your choice of NOEVENTVALUE is too long it will be truncated.
Default: If EVENTTYPE is numeric it takes the last value of -9999,-999,-99,-9,0 that is not used as an eventvalue. If EVENTTYPE is a character variable it takes the last value of p,py,pyr,pyrs that is not used as an eventvalue.

OUT
The standard output data set. Contains the cross-classifying variables specified with the CLASS, AXIS, SRTF and ZRTF statements and the EVENTTYPE variable if MODE=competing, intensity or multiple. Only non-empty cells are

output - if there is no risk time or events for some possible combination of levels of the cross-classifying variables no record is generated for that combination. Variables generated with ZRTF statements are excluded if they are used in SRTF statements or value=t. Such variables can be included in the output by inclusion in the CLASS statement. The standard flavour of the output data set is an event-time table containing aggregated number of events and time at risk in strata characterised by combinations of the values of the time-varying covariates above. The number of events and time at risk are put in variables called events and pyrs, respectively. Choosing METHOD=noagg will cause the output to be unaggregated with records containing entry time, exit times and censored for each individual time segment in strata defined by the cross-classifying variables. The variable censored will be 0 if an event occurred, and 1 otherwise. If you use COMPLETE=no then censored does not appear in the output and time intervals at risk and events are put in separate records, the latter category distinguished by having a missing value of exit. The variables pyrs and events are not included in the output when METHOD=noagg. If you use METHOD=noagg then the variable SUBJECT will automatically be added to OUT. If you specify RATES then the expected number of events is added to OUT in the variable expected, while if you use METHOD=noagg the relevant rate from RATE is added to OUT in the variable rate.

OUTCOMES
The optional external outcome events input data set. Can be used in any MODE, but is neccessary when MODE=multiple. It must contain the SUBJECT variable, the EVENTTIME variable and if MODE=multiple, competing or intensity it must also contain the EVENTTYPE variable. If you use the OUTCOMES option these variables will exclusively be fetched from the external input data set. If you have specified variables in the EVENTID statement present in both the external input data set and the base input data set it will be exclusively fetched from the external input data set.

RATES
An external rate data set. It must contain the numeric variable rate and at least one variable (such as sex or age) charactering the rate. If COMPLETE=yes the output from %stratify will be restricted to records where all the other variables in RATES than rate have the exact same combination of values and data types as a record in RATES. This restriction applies to variables that would appear in the normal output, some of them may later be dropped using the DROP statement. If you specify RATES and COMPLETE=yes then the expected number of events is added to OUT in the variable expected, while if you use METHOD=noagg the relevant rate from RATE is added to OUT in the variable rate. Apart from the effect on EVENTVALUES (see page 7), RATES have no effect at all on the output when COMPLETE=no. In order to calculate correct expected numbers of events etc. the rate must be number of events per time unit, where the time unit is SCALE times the unit of time in the input data (entry, exit, EVENTTIME, TIME etc.). Thus if entry and exit are SAS dates and SCALE is the default 365.25, the rate specified in RATES should be number of events per person year.
Interaction: Background rates or expected number of events are not calculated based on a RATES data set, unless COMPLETE=yes. RATES together with

EVENTVALUES determine the values of EVENTTYPE occurring in OUT.

SCALE
A mandatory strictly positive numerical constant. The scale relation between time units for time points in the input data sets (entry, exit, origins) and time units for the cut points specified in AXIS and SRTF statements. The same scale relation is used between input data set time units and the aggregated risk time in the pyrs variable. When using value=a in ZRTF statements, this scaling is also used. If the input data time points are given as SAS dates and the cut points are specified in years from the origin, SCALE should be 365.25.
Default: 365.25.

SUBJECT
The name of the person-identifying variable used by %stratify to link records in the base input data set (DATA) to records in any external input data sets specified in the EXPDATS statement and the OUTCOMES option. The SUBJECT variable has to be present in all the data sets that should be linked together according to your specification. It is also needed to in order to remove excess follow-up past outcome events and to link outcome events to the relevant follow-up intervals when using un-aggregated data.
Interaction: SUBJECT must be specified when using ZRFT or EXPDATS statements, when using the OUTCOMES option and when using the METHOD=noagg option.

TIME
The name of the time variable used in all of the EXPDATS data sets to signal when an exposure event happened. This time variable must be on the same time scale as that used for entry and exit.
Interaction: TIME must be specified when using ZRFT or EXPDATS statements.
Default: time.

VALUE
The name of the numerical attribute variable used in EXPDATS data sets to provide numerical attributes to exposure events. The VALUE variable only needs to be present in EXPDATS data sets containing exposure events with numerical attributes you want to analyse.
Interaction: The VALUE option must be used when using ZRTF statements with value=v or s.
Default: value.

VALUEC
The name of the character attribute variable used in EXPDATS data sets to provide character attributes to exposure events. The VALUEC variable only needs to be present in EXPDATS data sets containing exposure events with character attributes you want to analyse.
Interaction: The VALUEC option must be used when using ZRTF statements with value=cv.
Default: valuec.

3.2 AXIS statement

Generates a stratified version of a unity-rate time factor, such as age or calendar time.

```
AXIS variable origin= cuts= <speed=> ;
```

Required arguments

variable
The generated stratified version of a unity-rate time factor. Must be a valid SAS variable name and must be placed right after AXIS.

origin
A numeric constant or variable. When specifying a date constant, use double quotes as in

```
origin="01JAN1960"D ,
```

single quotes causes an error. If the value of origin is missing, the largest value in the cuts specification is assigned to the AXIS variable. If origin is a variable it must be present in the base input data set, or be generated in a ZRTF statement.
Abbreviation: origin=orig=o.

cuts
Specifies the lower endpoint of the intervals the variable is grouped into. This is the same value the variable will be assigned in the output. If the value of the AXIS variable somewhen during follow-up is below the lowest cut point a fatal error will occur. Cuts can be specified in one of two ways, either in the form cuts=X to Y by Z or cuts=X1 X2 X3 ... where X, Y, Z, and any element in the list is a non-missing numerical constant. Further both specifications must form a strictly increasing sequence.
Abbreviation: cuts=c.

Other arguments

speed
A numeric constant or variable used to specify whether the AXIS variable is incremented with rate 1 (as in the definition of a unity-rate time factor) or another rate (positive or negative, as used in connection with simple-rate time factors). When you use SRTF statements, %stratify internally calculates the relevant origins for segments of follow-up and transforms SRTF statements into AXIS statements. It is unlikely that you are ever going to use speed explicitly in an AXIS statement.
Abbreviation: speed=s.
Default: 1.

3.3 CLASS statement

Specifies the subset of variables whose values define the subgroup combinations in the output data other than those generated in AXIS, SRTF and ZRTF state-

ments or due to the value of MODE.

```
CLASS variable(s) ;
```

Required arguments

variable(s)
Must be a list of valid SAS variable names. The variables must reside in the base input data set or be generated in ZRTF statements. However, specifying variables generated in ZRTF statements is unneccessary, unless the point is to save ZRTF variables that are used in SRTF statements or are generated with value=t. The latter types of ZRTF variables are assumed by %stratify to be intermediates of no intrinsic value, and as such not made part of the final output data set. The CLASS variables can be of any type, but in typical applications they will have only a few discrete values that define levels of the variable. You can have any number of CLASS statements, but only the last one is used.
Interaction: If you use MODE=multiple, intensity or competing the output will automatically contain the variable EVENTTYPE, you should not specify it in a CLASS statement. If the point of retaining some of the CLASS variables is only to use it for calculating expected number of events, they may ultimately be dropped using the DROP statement.

3.4 DROP statement

Specifies variables used for calculating expected number of events that are to be dropped and aggregated over for the final output.

```
DROP variable(s) ;
```

Required arguments

variable(s)
The point of the statement is to allow you to drop some very detailed variables that are made just for calculating expected numbers of events to end up with a simpler and smaller data set for analysis. Specified variables that cannot be found are ignored. You can have any number of DROP statements, but only the last one is used. See **example 5** on page 30.
Interaction: The DROP statement only has effect if you use COMPLETE=yes, specify RATES and use an aggregating METHOD, i.e. not METHOD=noagg.

3.5 EVENTID statement

Specifies extra variables to characterise each outcome event in the EVENTDAT output data set.

```
EVENTID variable(s) ;
```

Required arguments

variable(s)
Must be a list of valid SAS variable names. The variables can be from the

base input data set or the external outcome events input data (OUTCOMES=) or be generated in ZRTF statements. If you have specified a variable in the EVENTID statement present in both the external input data set and the base input data set it will be exclusively fetched from the external input data set. Specified variables that cannot be found are ignored. You can have any number of EVENTID statements, but only the last one is used. The variables specified in the EVENTID statement are added to the cross-classifying variables from the ordinary output. A typical use of the statement is to see who contributes which events to which strata, requiring EVENTID variables that together identify cases and characterise them clinically. See example 4 on page 26.
Interaction: The EVENTID statement has no effect if you do not use the EVENTDAT option.

3.6 EXPDATS statement

Specifies external exposure data sets to be used to create time-varying covariates in ZRTF statements.

```
EXPDATS SAS-data-set(s) ;
```

Required arguments

SAS-data-set(s)
In each of the SAS data sets specified three variables are neccessary, named in the SUBJECT, TIME and EXPEVENT options. Depending on the type of ZRTF statements you use, you may also need the variables named in the VALUE and VALUEC options. The VALUE variable only needs to be present in EXPDATS data sets containing exposure events with numerical attributes you want to analyse. Similarly for the VALUEC variable and character attributes. %stratify automatically collects and sorts the relevant variables and linkable records, discarding records with missing values of the TIME variable, and events happening after termination of follow-up.
Interaction: EXPDATS must be non-empty if you use ZRTF statements.
If you use ZRTF statements with value=v or s, the numeric variable specified with the VALUE option must be present in at least one EXPDATS data set.
If you use ZRTF statements with value=cv, the character variable specified with the VALUEC option must be present in at least one EXPDATS data set.

3.7 SRTF statement

Generates a stratified version of a simple-rate time factor. By definition the derivative of a simple-rate time factor is a piece-wise constant function of time. Examples include cumulative length of employment which grows at a pace of 1 during employment, and 0 during unemployment and cumulative residential background radiation where the increment per time depends on where you live.

```
SRTF variable speed= cuts= <time=> <value=> ;
```

Required arguments

variable
The generated stratified version of a simple-rate time factor. Must be a valid SAS variable name and must be placed right after SRTF.

speed
A numeric variable used to specify the rate of change (positive or negative) in the SRTF as a function of time. When one of the other arguments value or time is unspecified, the value of the SRTF variable will be determined entirely from speed under the assumption that the SRTF variable is zero until and including the time point when the speed variable is first encountered. When you use SRTF statements, %stratify internally calculates the relevant origins and transforms SRTF statements into AXIS statements. See example 4 on page 26.
Abbreviation: speed=s.
Interaction: speed must be created in a ZRTF statement.

cuts
Specifies the lower endpoint of the intervals the SRTF variable is grouped into. This is the same value the variable will be assigned in the output. If the value of the SRTF variable somewhen during follow-up is below the lowest cut point a fatal error will occur. See example 7 on page 32. Cuts can be specified in one of two ways, either in the form cuts=X to Y by Z or cuts=X1 X2 X3 ... where X, Y, Z, and any element in the list is a non-missing numerical constant. Further both specifications must form a strictly increasing sequence.
abbreviation: cuts=c.

Other arguments

time
Used in conjunction with value and speed, time is a numeric variable used to specify the time at which the current values of value and speed were obtained. If used it must be applied through the whole follow-up interval. To yield correct results your must generate it at least every time time the speed variable or the value variable changes.
abbreviation: time=t.
Interaction: Time must be created in a ZRTF statement.
Time only has effect on the calculation of the SRTF variable when you also specify variables for value and speed.

value
Used in conjunction with time and speed, value is a numeric variable used to specify the raw or ungrouped value of the SRTF variable at times specified in the time variable. This construct allows non-trivial initial and intermediate values of the SRTF variable, but if used must be applied through the whole follow-up interval. To yield correct results your must have an updated version of the underlying exposure event attribute at least every time the speed variable changes. It must be on the same scale as the input time points. I.e. you may need to multiply your value variable by SCALE in order to obtain the expected results.
abbreviation: value=val=v.
Interaction: value must be created in a ZRTF statement.

Value only has effect on the calculation of the SRTF variable when you also specify variables for time and speed.

3.8 ZRTF statement

Generates a variable representing a zero-rate time factor based on information in a specific format in the EXPDATS input data set(s).

```
ZRTF variable value= condition= <n=> <groups=> <missing=> <length=> ;
```

A zero-rate time factor is defined by being constant between status changes. Typical examples are: marital status, place of residence, number of pregnancies. The EXPDATS data set(s) contain information about what exposure event happened (in the EXPEVENT variable) to whom (in the SUBJECT variable) at what time (in the TIME variable). There may be a numeric and/or character value attribute variable in each record in an EXPDATS data set, named in the VALUE and VALUEC options, respectively. The ZRTF statement then organises this information into a time-varying covariate, for example counting the number of occurrences of a specific type of event so far, or keeping the date of the latest birth so far as input (origin) to an AXIS statement measuring time since last birth. See examples 4 and 5 on page 26 and page 28.

Required arguments

variable
Must be a valid SAS variable name and must be placed right after ZRTF.

condition
The non-empty value of the EXPEVENT character variable that will trigger the (re-)evaluation of the ZRTF variable. It must be placed in double quotes ("") and must not contain signs (commas, blanks, semi-colons,..) conflicting with the interface of %stratify. A valid SAS variable name will always be a valid trigger. Condition is case-sensitive.
Abbreviation: condition=cond=c.

value
The value assigned to the ZRTF variable before grouping. Possible values are:

- v (VALUE at occurrence of condition= and n=)
- s (sum of VALUE at occurrence of condition= so far, initially 0)
- i (indicator that the condition= and n= has happened)
- n (number of occurrences of condition= so far)
- cv (VALUEC at occurrence of condition= and n=)
- t (TIME at occurrence of condition= and n=)
- a (age at occurrence of condition= and n=).

If you use value=i, n or s the ZRTF variable is initialised to 0, otherwise it is initialised to missing. It can never become missing when you use value=i. When you use value=n it can only become missing if your grouping does not cover the range of positive integers. It may become missing when using value=v, s, or cv when a missing value of the attribute variable is encountered. If this happens using value=s all later exposures of that type for that SUBJECT will be missing too. Using value=a or t it can only remain missing if the relevant exposure event doesn't happen prior to or during follow-up, or if the value of the BIRTHDATE variable is unknown.
Abbreviation: value=val=v.
Interaction: If you use value=v or s you must use the VALUE option.
If you use value=cv you must use the VALUEC option.
If you use value=a you must use the BIRTHDATE option.
If you use a ZRTF statement you must use the SUBJECT, EXPEVENT and TIME options. The latter two have default values expevent and time, respectively. The VALUE, VALUEC and BIRTHDATE options have default values value, valuec and birthdate, respectively.

Other arguments

n
Which occurrence of condition= that will trigger the (re-)evaluation of the ZRTF variable. Possible values are f, first, l, last or any integer > 0, where 1 means first, 2 means second etc. f and l are abbreviations of first and last, respectively. Choosing n=last implies that the value of the ZRTF variable is (re-)evaluated every time condition=... occurs, choosing any of the other values for n implies that the value of the ZRTF variable is evaluated at most once.
Default: last.

groups
Specifies how the raw ZRTF variable is grouped into categories. If the ZRTF variable is numeric the specification is as for cuts in the AXIS and SRTF statements, alternatively you can specify a format. If the ZRTF variable is a character variable the specification is a format. If unspecified no grouping occurs. If you use a format always include the period, and include the dollar-sign in character value formats. If you use a grouping specification similar to the one in an AXIS statement it works a little different here: all values are rounded downwards to the nearest cut point, and all values below the smallest cut point and missing values are assigned missing values.
Abbreviation: groups=g.

missing
Offers the opportunity to assign a special value to the ZRTF variable when missing. If the ZRTF variable is a character variable the value must be placed in double quotes ("") and must not contain signs (commas, blanks, parantheses,..) conflicting with the interface of %stratify. A valid SAS variable name will always be a valid specification in this case.
Abbreviation: missing=miss=m.

length
Length of the ZRTF variable in bytes, specified as in a data step, i.e. with a preceding $ for character variables.
Abbreviation: lenght=len=l.
Default: For numeric variables 8 bytes, for character variables the maximum length of the VALUEC variable in the EXPDATS data set(s).

4 Concepts

4.1 Timing conventions

The timing convention is that the characteristics of an interval a to b is correct for the interval [a,b), i.e. inclusive of a and exclusive of b. In post-processing for non-aggregated output, suited for e.g. Cox-regression, this is partly reversed so that an outcome event occurring at time b will be attached to the interval (a,b] which will then be presented as not ending in a censoring. In the following we will motivate and elaborate on these conventions. We will also suggest data edits that may be neccessary in order to perform your desired analysis.

Follow-up time needs to be split into disjoint intervals, which therefore have to be half-open: inclusive at one end and exclusive at the other end. Our convention of counting events in and assigning exposure to intervals of the form [a,b) corresponds to what happens in official statistics, is the natural way to count, and enables efficient aggregation of risk time and events, because we do not need to remember a past that an outcome event might need to be the termination of. That it is the natural and official way to count is exemplified by rights pertaining to age: you get the right to vote, have a driver's licence, get a pension etc. immediately on the day of your birthday, rather than at midnight bordering on the next day, which would be the earliest time you could be sure that you had attained a certain age. On the other hand the convention in Cox analyses and other risk set sampling is to analyse intervals on the form (a,b], so we also provide that possibility for non-aggregated output. For output in the form of counts and person-time at risk we need a little overhang (specified in the GRANULARITY option) to ensure follow-up time with covariates corresponding to events. However this overhang can never extend beyond the value of exit.

By default we consider time intervals, i.e. from entry to exit, to be inclusive of the former and exclusive of the latter. Thus when we say that something happens on a particular day, we assume it to happen at midnight at the very beginning of that day. The same logic applies for intervals specified for cut points etc. For example when we specify cuts=0 to 20 by 5 the output will be stratified into intervals 0-4.9999..., 5-9.9999..., 10-14.9999..., 15-19.9999..., 20+. Thus if you intend to follow a cohort of people through the calendar years 2000 to 2007 inclusive, you should have entry=max(entry,"01JAN2000"D) and exit=min(exit,"01JAN2008"D) if your time points are SAS dates. Similarly the value of zero-rate time factors generated in ZRTF statements are effective from the start of the day (or whatever time unit you use) when they change. If it is

critical in your particular application only to have persons exposed to something when you are sure they were exposed (i.e. the day after they got the exposure), you must generate/edit your input data accordingly.

%stratify owes much of it's capabilities to the almost complete internal separation of handling of outcome events and time at risk. Thus internally there are no censoring variables. To ensure that there is risk time associated with every outcome event you must supply it through your setting of the GRANULARITY option. The default value (GRANULARITY=1) corresponds to the situation where your input data time points are SAS dates, and we then consider the followed-up persons to be at risk throughout the day when the outcome event occurred. You can make GRANULARITY arbitrarily small, as long as it is still strictly positive. By the convention above we assume that the person had the outcome event at 00.00, i.e. at the very beginning of the day the outcome occurred. If you generate output for Cox regression, i.e. use the options METHOD=noagg and COMPLETE=yes then this event time is inserted as the uncensored termination of the previous follow-up interval for that person. I.e. if a person had an event at time b, the interval (a,b] for that person is characterised by having the variable censored set to 0. This post-processing of non-aggregated output is the only deviation from the above [entry,exit) conventions, in order to be in agreement with the conventions to risk set (Cox regression) analysis.

The time intervals specified by cuts= etc. are precise, and since the actual length of each calendar year varies, there may be slight discrepancies between intended and actual cut points. This may provoke a fatal error if entry is smaller than the lowest cutpoint for some AXIS. For a way to avoid this, see example 7 on page 32. For more about timing pit-falls, see Rostgaard [3].

4.2 Automatic censoring

%stratify automatically and correctly handles everything to do with censoring. Follow-up intervals only contribute to output data sets if . $<$ entry $<$ exit. Persons who have had an outcome event prior to the specified follow-up interval are removed from risk and do not contribute anything for that type of outcome event when MODE=single, competing or multiple. Outcome events are only counted if entry $\leq$ EVENTTIME $<$ exit. See example 1 on page 21 to see how this works. When MODE=intensity %stratify allows any number of occurrences of an event and do not terminate follow-up when it happens. Using this mode you can produce rates directly comparable to public rates based on population figures of person-time at risk, ignoring prevalent cases.

4.3 Algorithms and computational ressources

ZRTF variables are created before SRTF and AXIS variables so you can use ZRTF statements to create intermediate variables to be used in SRTF and AXIS statements. For example, a ZRTF variable may be the origin in an AXIS statement.

%stratify may require some time and memory during an initial phase when input

data are sorted and linked (using standard SAS SQL code), and in a later phase which combines stratification by unity-rate time factors with aggregation of events and time at risk. In all cases this stratification/aggregation phase is preceeded by sorting the data. When METHOD=fst or METHOD=arr this stratification/aggregation phase consists of one data step, when METHOD=sum the data are stratified in a data step and then aggregated (using proc summary) and finally METHOD=chunk is an extension of METHOD=sum where the data to be stratified is input into the stratification data step in chunks of approximately the number of observations specified with the CHUNKSIZE option, and the resulting aggregated output appended to previous output. METHOD=chunk is preferable to METHOD=sum when the data to process become exceedingly large, usually due to a combination of many persons under follow-up and a very detailed modelling of the exposures. METHOD=sum is only preferable to METHOD=fst when you cannot use the latter due to lack of RAM. METHOD=arr can under ideal conditions (few potential strata af all types) be the fastest of all methods, but can be exceedingly slow compared to the alternatives under very non-ideal conditions. METHOD=fst is the fastest of the methods over a very wide range of conditions. It is the default and should be used whenever possible. When you use METHOD=fst or METHOD=arr temporary arrays are used, the size of which is $16\prod_{a=1}^{A} p_a + 8A\prod_{a=1}^{A} p_a$ bytes and $16\prod_{a=1}^{A} p_a$ bytes, respectively, where there are A AXIS statements, and p_a denote the number of cut points of AXIS a. Either of these may become prohibitively large when you have many AXIS statements with many cut points. SRTF statements are internally transformed into AXIS statements, and therefore count here too. See Rostgaard [3] for further details about the algorithms.

The algorithm for stratifying individual follow-time is an instance of Macaluso's method B [4], realising the full generality of this method, which was originally described by Clayton in Breslow & Day [5].

4.4 Internal post-processing

When COMPLETE=yes (the default), some post-processing may occur after the stratification and aggregation phase depending on the values of MODE and RATES. The output from the stratification/aggregation phase contains all relevant follow-up time once, all the outcome-specific events and for MODE=multiple outcome-specific follow-up after events. If MODE=multiple, competing or intensity this minimalistic output is turned into a data set that can be by-processed by each outcome, containing all events and all follow-up time for each outcome. If METHOD=noagg the output from the stratification phase will be transformed into something that can be analysed directly by PROC PHREG, i.e. entry and exit are put on the relevant scale and the variable censored is generated. If RATES are specified the resulting data set is further equipped with expected number of events for aggregated output (METHOD=fst,sum,arr,chunk) or rate (METHOD=noagg). Using the DROP statement it is possible to drop and aggregate over variables that were just used to create the matching with the RATES data set.

5 Examples

5.1 Example 1: How to use %stratify - follow-up and automatic censoring

The idea behind %stratify is that you should only specify the minimal necessary information about who you wish to follow-up when and what outcomes occur, and then let %stratify do the censorings corresponding to the data and the planned mode of follow-up. This is demonstrated below for the default simplest mode of follow-up, where the outcome of interest occurs at the time held in the variable eventtime, if it occurs at all. The person with idnr=2 is not at risk prior to the outcome and hence does not appear in the output data when we generate output for Cox regression (METHOD=noagg and COMPLETE=yes). Had we instead generated the default events and time at risk output for Poisson regression (Method=fst and COMPLETE=yes), that person would have contributed an event, and a little follow-up time. See section 4.1 and section 4.2 for further details.

Program

```
data a;
 input idnr birthdate migdate eventtime; datalines;
 1 1995    . 1996
 2 1996    . 2000
 3 1998    . 2000.01
 4 1995    . 2003.5
 5 2002    . 2008
 6 1991 2005 2006.5
 7 1995    .  .
 ;
run;

data a; set a; entry=max(birthdate,2000); exit=min(migdate,2008); run;

%stratify(data=a out=b subject=idnr scale=1 METHOD=NOAGG COXORIGIN=0);
title 'Follow-up for an outcome at eventtime';
proc print noobs; run;
```

Output

```
Follow-up for an outcome at eventtime

idnr    entry      exit     censored

  3      2000    2000.01        0
  4      2000    2003.50        0
  5      2002    2008.00        1
  6      2000    2005.00        1
  7      2000    2008.00        1
```

5.2 Example 2: Use of the MODE option with unaggregated output

Here we illustrate by example how different choices of MODE affect follow-up and what the output looks like when using unaggregated output (METHOD=NOAGG), suitable for counting-process style input for PROC PHREG or other tools for Cox regression. We look at a single person who gets two different diseases at different times. It is illustrated how the macro when confronted with more than one outcome event time in MODE=single assumes that the relevant event time is the first, otherwise you should edit your input data so as to only present the relevant outcome events to the macro. You could assume that the two diseases are two different cancers, and what is obtained by the convention in MODE=single then corresponds to an assessment of the incidence of cancer overall, if any cancer were to appear in the OUTCOMES data set. Note that if a variable is specified for COXORIGIN it will automatically be fetched from the input data set or as generated in a ZRTF statement and will appear in the output. The SUBJECT variable will appear in the output.

Program

```
data studybase;
 input idnr sex birthdate entry exit;
 datalines;
 1 1 1935 2000 2008
 ;
run;

data disease;
 input idnr disease eventtime;
 datalines;
 1 1 2003.5
 1 2 2005.5
 ;
run;

%let setup= data=studybase outcomes=disease out=output
            eventtype=disease subject=idnr scale=1
            METHOD=NOAGG COXORIGIN=BIRTHDATE;

%stratify(&setup MODE=SINGLE);
title 'Follow-upollow-up for a single outcome';
proc print noobs; run;

%stratify(&setup MODE=MULTIPLE);
title 'Simultaneous follow-up for many single outcomes';
proc print noobs; run;
```

```
%stratify(&setup MODE=COMPETING);
title 'Follow-up for many outcomes to the occurrence of the first (Competing Risks)';
proc print noobs; run;

%stratify(&setup MODE=INTENSITY);
title 'Follow-up for many outcomes without censoring at outcome event';
proc print noobs; run;
```

Output

```
Follow-up for a single outcome

idnr birthdate    entry     exit    censored

 1      1935       65.0     68.5        0

Simultaneous follow-up for many single outcomes

disease    idnr    birthdate    entry     exit   censored

   1         1        1935       65.0     68.5        0
   2         1        1935       65.0     70.5        0

Follow-up for many outcomes to the occurrence of the first (Competing Risks)

disease    idnr    birthdate    entry     exit   censored

   1         1       1935       65.0      68.5        0
   2         1       1935       65.0      68.5        1

Follow-up for many outcomes without censoring at outcome event

disease    idnr    birthdate    entry     exit   censored

   1         1       1935       65.0      68.5        0
   1         1       1935       68.5      73.0        1
   2         1       1935       65.0      70.5        0
   2         1       1935       70.5      73.0        1
```

5.3 Example 3: Use of the MODE option with aggregated output

Here we illustrate by example how different choices of MODE affect follow-up. We look at a single person who gets two different diseases at different times. It is illustrated how the macro when confronted with more than one outcome event time in MODE=single assumes that the relevant event time is the first, otherwise you should edit your input data so as to only present the relevant outcome events to the macro. You could assume that the two diseases are two different cancers, and what is obtained by the convention in MODE=single then corresponds to an assessment of the incidence of cancer overall, if any cancer were to appear in the OUTCOMES data set.

Program

```
options nocenter nodate;

data studybase;
 input idnr sex birthdate entry exit;
 datalines;
 1 1 1935 2000 2008
 ;
run;

data disease;
 input idnr disease eventtime;
 datalines;
 1 1 2003.5
 1 2 2005.5
 ;
run;

data ev; do disease=1 to 2; output; end; run;

%let setup= data=studybase outcomes=disease out=output eventvalues=ev
            eventtype=disease subject=idnr scale=1 granularity=0.001 ;

%stratify(&setup MODE=SINGLE);
title 'Follow-up for a single outcome';
proc print noobs; run;

%stratify(&setup MODE=MULTIPLE);
title 'Simultaneous follow-up for many single outcomes';
proc print noobs; run;

%stratify(&setup MODE=COMPETING);
title 'Follow-up for many outcomes to the occurrence of the first (Competing Risks)';
proc print noobs; run;
```

```
%stratify(&setup MODE=INTENSITY);
title 'Follow-up for many outcomes ignoring prevalent cases';
proc print noobs; run;
```

Output

```
Follow-up for a single outcome

events     pyrs

   1      3.501

Simultaneous follow-up for many single outcomes

disease    events     pyrs

   1          1      3.501
   2          1      5.501

Follow-up for many outcomes to the occurrence of the first (Competing Risks)

disease    events     pyrs

   1          1      3.501
   2          0      3.501

Follow-up for many outcomes ignoring prevalent cases

disease    events    pyrs

   1          1        8
   2          1        8
```

5.4 Example 4: Use of the AXIS and SRTF statements

In this example we show how to use the AXIS statement and the SRTF statement to stratify output according to unity-rate time factors (here age, calendar period and time since last birth) and simple-rate time factors (here cumulative years employed) which may increase at a rate of 0 or 1 depending in this case upon employement status (here working). Here the value of the simple-rate time factor is determined exclusively from the speed variable, assuming it to be zero before the speed variable is first available for a person. For an example of a more general and complicated handling of bit-rate time factors, see Rostgaard [3]. The SRTF statement for simple-rate time factors generalizes and replaces the BRTF statement in that paper. We illustrate how to use ZRTF statements to generate an origin for an axis variable (here time_since_last_birth), as would be the typical and recommended way of doing things when origins may change over time.

Program

```
options nocenter nodate;

data studybase;
 input idnr sex birthdate entry exit diseasedate diseasetype;
 datalines;
 1 1 1975 2000 2008 2005.5 3
 ;
run;

data work;
 length expevent $20;
 input expevent time idnr value;
 datalines;
 working 1991 1 1
 working 2001 1 0
 working 2003 1 1
 ;
run;

data births;
 length expevent $20;
 input expevent time idnr;
 datalines;
 giving_birth 2000.5 1
 giving_birth 2004.0 1
 ;
run;
```

```
%stratify(data=studybase out=output eventdat=close_look
 eventtime=diseasedate subject=idnr scale=1 granularity=0.001;
 eventid idnr diseasetype;
 expdats work births;
 class sex;
 zrtf working          c="working" v=v;
 zrtf last_birth       c="giving_birth" v=t;
 srtf worked_yrs       s=working c=0 to 20 by 1;
 axis time_since_birth o=last_birth c=0 1 2 3 4 5 7 10 15 999;
 axis period           o=0 c=2000 to 2020 by 0.5;
 axis age              o=birthdate c=0 to 100 by 5);

proc sort data=output; by period time_since_birth; run;

title 'Standard output - follow-up for a single outcome';
proc print noobs data=output; run;

title 'Using the EVENTDAT option - content of the EVENTDAT data set';
proc print noobs data=close_look; run;
```

Output

Standard output - follow-up for a single outcome

sex	time_since_birth	period	age	worked_yrs	events	pyrs
1	999	2000.0	25	9	0	0.500
1	0	2000.5	25	9	0	0.500
1	0	2001.0	25	10	0	0.500
1	1	2001.5	25	10	0	0.500
1	1	2002.0	25	10	0	0.500
1	2	2002.5	25	10	0	0.500
1	2	2003.0	25	10	0	0.500
1	3	2003.5	25	10	0	0.500
1	0	2004.0	25	11	0	0.500
1	0	2004.5	25	11	0	0.500
1	1	2005.0	30	12	0	0.500
1	1	2005.5	30	12	1	0.001

Using the EVENTDAT option - content of the EVENTDAT data set

sex	diseasetype	idnr	time_since_birth	period	age	worked_yrs
1	3	1	1	2005.5	30	12

5.5 Example 5: Use of ZRTF statements

Below we exemplify some more features of the ZRTF statement. Note especially that using formats to group things work somewhat heavy-handed here. In the line where we generate SEX_OF_LAST_CHILD if we had not specified l=$7 the result would have been truncated to the first two letters, i.e. the maximum length of the VALUEC variables in the EXPDATS data sets. Also numeric formats have to yield numeric values to be any good, as seen in the two contrasting versions of BWFIRSTBIRTH3G.

Program

```
options nocenter nodate ls=80;

data studybase;
 input idnr sex birthdate entry exit diseasedate;
 datalines;
 1 1 1975 2000 2008 2008.5
 ;
run;

data births;
 length expevent $20 valuec $2;
 input idnr expevent time value valuec;
 datalines;
 1 giving_birth 2000.5 3700 g
 1 res_of_child 2000.5 . VA
 1 giving_birth 2004.0 3830 b
 1 res_of_child 2004.0 . CA
 ;
run;

proc format ;
 value $sex  'g'='girl' 'b'='boy' 'u'='unknown' m='both';
 value bw4kg  0-2999='light' 3000-3999='normal' 4000-high='heavy';
 value bw4kgr 0-2999='0' 3000-3999='3' 4000-high='4';
run;

%stratify(data=studybase out=output
 eventtime=diseasedate subject=idnr scale=1 granularity=0.001;
 expdats births;
 zrtf age_at_lastbirth       c="giving_birth" v=a g=12 18 25 30 35;
 zrtf birthweightfirstbirth  c="giving_birth" v=v n=1 g=0 to 5000 by 500 m=9999;
 zrtf bwfirstbirth3gWRONG    c="giving_birth" v=v n=1 g=bw4kg.;
 zrtf bwfirstbirth3gRIGHT    c="giving_birth" v=v n=1 g=bw4kgr. m=.m;
 zrtf lastbirthweight        c="giving_birth" v=v;
 zrtf residence_at_birth     c="res_of_child" v=cv;
 zrtf sex_of_last_child      c="giving_birth" v=cv g=$sex. l=$7;
 zrtf no_of_births           c="giving_birth" v=n);
```

```
proc sort data=output; by no_of_births; run;

title 'Standard output - zero-rate time factors at play';
proc print noobs data=output; run;
```

Output

```
Standard output - zero-rate time factors at play

        b
        i
        r     b    b
        t     w    w           r
        h     f    f           e      s
 a      w     i    i           s      e
 g      e     r    r      l    i      x
 e      i     s    s      a    d      _
 _      g     t    t      s    e      o
 a      h     b    b      t    n      f      n
 t      t     i    i      b    c      _      o
 _      f     r    r      i    e      l      _
 l      i     t    t      r    _      a      o
 a      r     h    h      t    a      s      f
 s      s     3    3      h    t      t      _
 t      t     g    g      w    _      _      b    e
 b      b     W    R      e    b      c      i    v
 i      i     R    I      i    i      h      r    e     p
 r      r     O    G      g    r      i      t    n     y
 t      t     N    H      h    t      l      h    t     r
 h      h     G    T      t    h      d      s    s     s

 .    9999    .    M      .                  0    0    0.5
25    3500    .    3   3700    VA    girl    1    0    3.5
25    3500    .    3   3830    CA    boy     2    0    4.0
```

5.6 Example 6: Calculating Standardized Incidence Ratios (SIRs)

Here we illustrate how to calculate data for analysis of standardised incidence ratios. We show it in the simple situation with one outcome, and in a more complicated situation with many outcomes. We see how the rate data set RATES determine what outcomes we calculate statistics for. The effect of the DROP statement is also shown.

Program

```
data studybase;
 input idnr sex birthdate entry exit;
 datalines;
 1 1 1935 2000 2008
 ;
run;

data disease;
 input idnr disease eventtime;
 datalines;
 1 1 2003.5
 1 2 2005.5
 ;
run;

data rates;
  do disease=1 to 3;
    do sex=1 to 2;
      do period=2000 to 2010;
         do age=60 to 75;
            rate=0.1+0.1*mod(period,2); output;
         end;
      end;
    end;
  end;
run;

data srates;
    do sex=1 to 2;
      do period=2000 to 2010;
         do age=60 to 75;
            rate=0.1+0.1*mod(period,2); output;
         end;
      end;
    end;
run;
```

```
%let setup= data=studybase outcomes=disease out=output
            eventtype=disease
            subject=idnr scale=1 granularity=0.001 ;

%stratify(&setup RATES=SRATES MODE=SINGLE ;
 class sex  ;
 axis period o=0 c=2000 to 2008 by 1;
 axis age o=birthdate c=0 to 100 by 1;
 drop age period );

title 'Data for SIR analysis with a single outcome';
proc print noobs; run;

%stratify(&setup RATES=RATES MODE=MULTIPLE ;
 class sex  ;
 axis period o=0 c=2000 to 2008 by 1;
 axis age o=birthdate c=0 to 100 by 1;
 drop age period );

title 'Data for SIR analysis with multiple outcomes';
proc print noobs; run;
```

Output

```
Data for SIR analysis with a single outcome

sex    events     pyrs    expected

 1        1      3.501     0.5002

Data for SIR analysis with multiple outcomes

disease    sex    events     pyrs    expected

   1        1        1      3.501     0.5002
   2        1        1      5.501     0.8002
   3        1        0      8.000     1.2000
```

5.7 Example 7: Avoiding fatal errors due to AXIS statements

There may be discrepancies between intended and actual cut points due to the varying real length of the calendar year. If the value of the variable entry is less than the lowest cut point on some time axis, a fatal error occurs. Here we illustrate the problem, and one way of avoiding it.

Program

```
* Illustrating timing problems ;

title "Will not work" ;

data studybase;
 entry="01JAN1968"D; exit="16APR1981"D; eventtime=.;
run;

%stratify(axis period  o="01JAN1900"D c=68 to 108 by 5);

* See LOG ;

title "Will work" ;

data studybase;
 entry="01JAN1968"D; exit="16APR1981"D; eventtime=.;
 bc="01JAN1968"D-1968*365.25;
run;

%stratify(axis period o=bc c=1968 to 2008 by 5);

proc sort; by period; run;

proc print; run;
```

Log (Partial output only)

```
ERROR:  An exception has been encountered.
Please contact technical support and provide them with the following
       traceback information:

The SAS task name is [DATASTEP]
ERROR:  Read Access Violation DATASTEP
Exception occurred at (0B88466D)
Task Traceback
Address   Frame     (DBGHELP API Version 4.0 rev 5)
```

```
NOTE: The SAS System stopped processing this step because of errors.
NOTE: There were 1 observations read from the data set
      WORK.THANK_YOU_FOR_USING_FSTPYRS.
WARNING: The data set WORK.STUDYBASE may be incomplete.  When this
         step was stopped there were 0 observations and 3 variables.
WARNING: Data set WORK.STUDYBASE was not replaced because this step
         was stopped.
NOTE: DATA statement used (Total process time):
      real time           0.06 seconds
      cpu time            0.03 seconds
```

Out

```
Will work

Obs    period    events      pyrs

 1      1968        0      5.00000
 2      1973        0      5.00000
 3      1978        0      3.28953
```

5.8 Example 8: The Andersen-Gill model

The Andersen-Gill model [6] allows an outcome event to recur ever so often. Once the outcome event have occurred follow-up for the next occurrence begins. To obtain this we need to make use of MODE=noagg so that follow-up is not terminated by an outcome event, and at the same time we must ensure that the follow-up time is cut into segments such that there is always a time segment terminated by the time of the occurrence of the outcome event. This is accomodated here by the ZRTF condstruct. Similar techniques in data preparation are needed for other approaches to survival analysis involving multiple outcome events in a subject [7].

Program

```
options nodate nocenter;

data studybase;
 input idnr sex birthdate entry exit;
 datalines;
 1 1 1935 2000 2008
 ;
run;

data disease;
 input idnr disease eventtime;
 datalines;
 1 1 2003.5
 1 1 2005.5
 1 1 2006
 ;
run;

proc sql;
 create table AGhelp as select unique idnr, eventtime as time,
   'OutcomeEventChopper' as expevent from disease;
quit;

%stratify( data=studybase outcomes=disease out=output
           eventtype=disease subject=idnr scale=1
           METHOD=NOAGG COXORIGIN=BIRTHDATE MODE=INTENSITY;
 expdats AGhelp;
 zrtf AGHELP c="OutcomeEventChopper" v=t);

title 'Follow-up for a single recurring  (Andersen-Gill model) outcome';
proc print noobs; run;
```

Output

```
Follow-up for a single recurring  (Andersen-Gill model) outcome

disease    idnr    birthdate    entry    exit    censored

   1         1        1935       65.0    68.5        0
   1         1        1935       68.5    70.5        0
   1         1        1935       70.5    71.0        0
   1         1        1935       71.0    73.0        1
```

5.9 Example 9: Unaggregated output using COMPLETE=no

In this example we show how you can post-process un-aggregated output yourself from %stratify, rather than leaving it to %stratify. This requires use of METHOD=noagg and COMPLETE=no. We exemplify this for MODE=multiple only and sketch how to proceed with the analysis using PROC PHREG. This example is a simplification of what %stratify does internally when COMPLETE=yes. The most likely situation where it may be beneficial to use COMPLETE=no will be when you work with extremely large data sets with many rare outcomes, in which case you can post-process and analyse the data for each outcome separately, starting from a data set that will typically not be much larger than that required for the analysis of just one of the outcomes. You should be able to see how to modify the presented code for that purpose, and for the other modes of follow-up. For MODE=competing or intensity you may prefer to have a data set with differently named (number of) outcomes, rather than having essentially a copy of the data set for each outcome for by-processing. This is also made directly and efficiently from the output dataset using COMPLETE=no, both for aggregated and unaggregated output. However in that case you must ensure that the data is split according to all studied outcomes, see example 9. In this example we look at number of pregnancies as a risk factor for two diseases. We follow-up two women with different pregnancy histories and prepare for Cox regression with age as the underlying time scale.

Program

```
data studybase;
 input idnr birthdate entry exit; datalines;
 1 1975 1990 2008
 2 1980 1995 2008
 ;
run;

data disease;
 input idnr disease eventtime; datalines;
 1 1 2003.5
 1 2 2005.5
 ;
run;

data exposure; length expevent $10;
 input idnr date expevent; datalines;
 1 1995 pregnant
 1 1998 pregnant
 1 2002 pregnant
 2 1999 pregnant
 2 2004 pregnant
 ;
run;
```

```
%let setup= data=studybase outcomes=disease out=output
            eventtype=disease noeventvalue=0
            time=date subject=idnr scale=1 COMPLETE=NO;

%stratify(&setup METHOD=NOAGG MODE=MULTIPLE;
 class idnr birthdate;
 expdats exposure;
 zrtf pregnancies v=n c="pregnant");

data a (keep=idnr disease age_in)
     b (drop=entry exit birthdate);
 set output;
 age_in  =entry-birthdate;
 age_out =exit -birthdate;
 if age_out=. then output a;
 if disease=0 then do;
   do disease=1 to 2; output b; end;
 end;
run;

proc sort data=b; by disease idnr; run;

proc sql;
 create table a as select unique disease, idnr, min(age_in) as _fullstop
   from a group disease, idnr ;
quit;

data multi(drop=_fullstop);
 merge b a; by disease idnr;
 if _fullstop>. then age_out=min(age_out,_fullstop);
 if age_in<age_out;
 censored=1-(_fullstop=age_out);
run;

proc sort data=multi; by disease idnr age_in; run;

title 'Cox type multiple outcome follow-up';
proc print noobs data=multi; run;

/*
proc phreg data=multi;
 model (age_in,age_out)*censored(1)=pregnancies;
 by disease;
run;
*/
```

Output

Cox type multiple outcome follow-up

idnr	pregnancies	disease	age_in	age_out	censored
1	0	1	15	20.0	1
1	1	1	20	23.0	1
1	2	1	23	27.0	1
1	3	1	27	28.5	0
2	0	1	15	19.0	1
2	1	1	19	24.0	1
2	2	1	24	28.0	1
1	0	2	15	20.0	1
1	1	2	20	23.0	1
1	2	2	23	27.0	1
1	3	2	27	30.5	0
2	0	2	15	19.0	1
2	1	2	19	24.0	1
2	2	2	24	28.0	1

5.10 Example 10: Aggregated output using COMPLETE=no

In this example we show how you can post-process aggregated output yourself from %stratify rather than leaving it to %stratify. This requires COMPLETE=no. In this example we follow-up a single person who gets two diseases, say, two different types of cancer at different times. Using a little additional code in connection with the various settings of MODE we produce the type of data needed to do the following types of analyses:

- follow-up for cancer overall (MODE=single),
- independent follow-up for each of several cancer subtypes (MODE=multiple),
- follow-up for the first occurrence of cancer, noting the subtype (MODE=competing),
- independent follow-up for each of several cancer subtypes, ignoring prevalent cases (MODE=intensity).

The latter type of data would often be the starting point for calculating standardized incidence ratios (SIRs). If in doubt, see [3] for how to proceed with the analysis and further details in each case. The examples shown here are essentially what %stratify does internally if you use the default value of the COMPLETE option. The most likely situation where it may be beneficial to use COMPLETE=no will be when you work with extremely large data sets with many rare outcomes, in which case you can post-process and analyse the data for each outcome seperately, starting from a data set that will not be much larger than that required for the analysis of just one of the outcomes. You should be able to see how to modify the presented code for that purpose. To see why this works, consult Rostgaard [3] or experiment with the code. For MODE=competing or intensity you may prefer to have a data set with differently named (number of) outcomes, rather than having essentially a copy of the data set for each outcome for by-processing. If so, this is also made directly and efficiently from the output data set obtained using COMPLETE=no.

Program

```
options nocenter nodate;

data studybase;
 input idnr sex birthdate entry exit;
 datalines;
 1 1 1935 2000 2008
 ;
run;

data disease;
 input idnr disease eventtime;
 datalines;
 1 1 2003.5
 1 2 2005.5
 ;
run;

data ev; do disease=1 to 2; output; end; run;

%let setup= data=studybase outcomes=disease out=output eventvalues=ev
            eventtype=disease noeventvalue=0 COMPLETE=NO
            subject=idnr scale=1 granularity=0.001 ;

%stratify(&setup MODE=SINGLE);
title 'Follow-up for a single outcome';
proc print noobs; run;

%stratify(&setup MODE=MULTIPLE);

data output; set output;
 if disease ne 0 then do; pyrs=-pyrs; output; end;
 if disease=0 then do; do disease=1,2; output; end; end;
run;
proc summary nway; class disease; var events pyrs;
 output out=output(drop=_freq_ _type_ where=(pyrs>0)) sum= ;
run;

title 'Simultaneous follow-up for many single outcomes';
proc print noobs; run;

%stratify(&setup MODE=COMPETING);

data output; set output;
 if disease ne 0 then do; output; end;
 if disease=0 then do; do disease=1,2; output; end; end;
run;
```

```
proc summary nway; class disease; var events pyrs;
 output out=output(drop=_freq_ _type_ where=(pyrs>0)) sum= ;
run;

title 'Follow-up for many outcomes to the occurrence of the first (Competing Risks)';
proc print noobs; run;

%stratify(&setup MODE=INTENSITY);

data output; set output;
 if disease ne 0 then do; output; end;
 if disease=0 then do; do disease=1,2; output; end; end;
run;

proc summary nway; class disease; var events pyrs;
 output out=output(drop=_freq_ _type_ where=(pyrs>0)) sum= ;
run;

title 'Follow-up for many outcomes ignoring prevalent cases';
proc print noobs; run;
```

Output

```
Follow-up for a single outcome

events     pyrs

   1      3.501

Simultaneous follow-up for many single outcomes

disease    events     pyrs

   1          1      3.501
   2          1      5.501

Follow-up for many outcomes to the occurrence of the first (Competing Risks)

disease    events     pyrs

   1          1      3.501
   2          0      3.501

Follow-up for many outcomes ignoring prevalent cases

disease    events    pyrs

   1          1        8
   2          1        8
```

References

[1] Rostgaard K. Stratify manual. http://sourceforge.net/projects/pyrsstep.

[2] Rostgaard K. Pyrsstep. The manual and the macro is zipped into pyrsstep.zip available at StatLib to be found at http://lib.stat.cmu.edu.

[3] Rostgaard K. Methods for stratification of person-time and events - a prerequisite for Poisson regression and SIR estimation. Epidemiologic Perspectives and Innovations 2008:5;7.

[4] Macaluso M. Exact stratification of person-years. Epidemiology 1992;3:441-8.

[5] Breslow NE, Day NE. Statistical methods in Cancer Research, Vol. II: The Design and Analysis of Cohort Studies. Lyon: IARC scientific publications 82, International Agency for Research on Cancer; 1987.

[6] Andersen PK, Gill RD. Cox's regression model for counting processes: A large sample study. Annals of Statistics 1982;10:1100-20.

[7] Therneau TM, Grambsch PM. Modeling Survival Data - Extending the Cox Model. New York: Springer-Verlag; 2000.