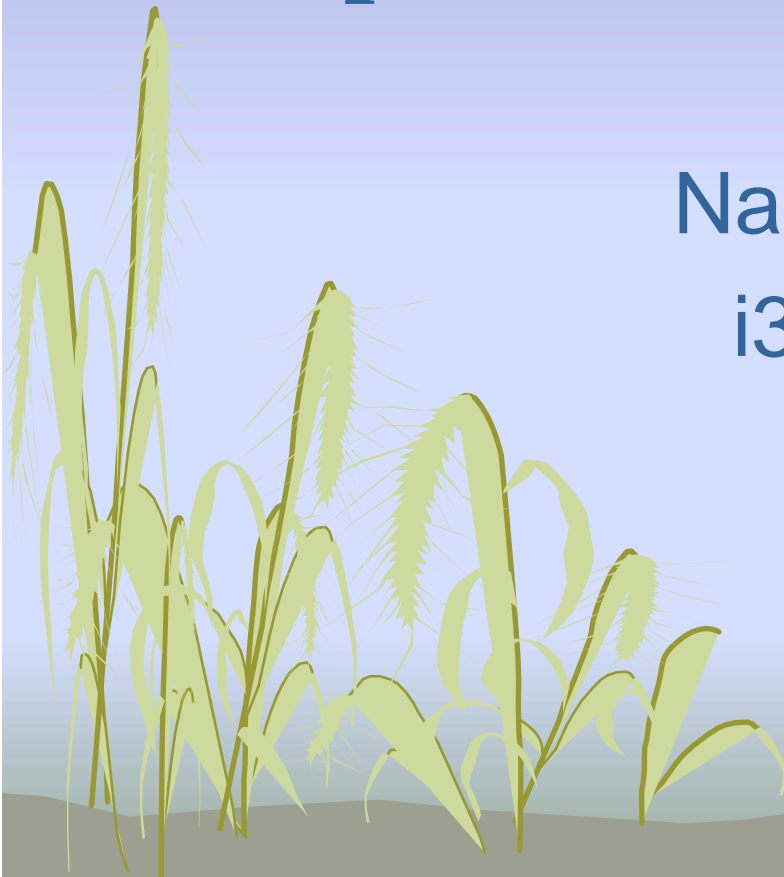


# One-Step Change from Baseline Calculations and Other DOW- Loop Tricks

Nancy Brucken  
i3 Statprobe



# *Input Dataset*

USUBJID	VISITC	VISITN	HR
1	Screening	1	91
1	Day 1	2	.
1	Week 1	3	68
1	Week 2	4	73
1	Week 4	5	96
2	Screening	1	.
2	Day 1	2	73
2	Week 1	3	73
2	Week 2	4	52
2	Week 4	5	59

# *The Task*

Compute change from baseline for each post-baseline record, where baseline = the last record on or before Day 1.

# *Output Dataset*

USUBJID	VISITC	VISITN	HR	BL	CHGBL
1	Week 1	3	68	91	-23
1	Week 2	4	73	91	-18
1	Week 4	5	96	91	5
2	Week 1	3	73	73	0
2	Week 2	4	52	73	-21
2	Week 4	5	59	73	-14

# *Common Approach*

- Separate baseline and post-baseline values
- Sort baseline dataset, and take the last value for each subject
- Merge baseline and post-baseline datasets, and compute change from baseline

# *Alternate Solution*

- DOW-Loop
  - Also known as Whitlock DO-Loop or Dorfman-Whitlock DO-Loop

# *Proposal*

- One pass through the dataset
- DOW-Loop takes advantage of PDV handling of variables
  - SET statement resets variables from input dataset
  - DATA statement resets assigned variables

# *DOW-Loop Code*

```
data postbase;  
  do until (last.usubjid);  
  *** Only keep non-missing pre-dose values;  
  set save.vitals (where=(not(visitn <= 2 and hr is missing)));  
  by usubjid visitn;  
  if visitn <= 2 then bl = hr;  
  else do;  
    chgbl = hr - bl;  
    output;  
  end;  
end;  
run;
```

# *PDV in Action*

- Start of DATA step:

LAST. USUBJID	USUBJID	VISITC	VISITN	HR	FIRST. USUBJID	FIRST. VISITN	LAST. VISITN	BL	CHGBL
1	.	.	.	.	1	1	1	.	.



**by usubjid visitn;**

# *First Record in Dataset*

```
data postbase;  
do until (last.usubjid);  
  *** Only keep non-missing pre-dose values;  
  set save.vitals (where=(not(visitn <= 2 and hr is missing)));  
  by usubjid visitn;  
  if visitn <= 2 then bl = hr;  
  else do;  
    chgbl = hr - bl;  
    output;  
  end;  
end;  
run;
```

LAST. USUBJID	USUBJID	VISITC	VISITN	HR	FIRST. USUBJID	FIRST. VISITN	LAST. VISITN	BL	CHGBL
0	1	Screening	1	91	1	1	1	.	.

# First Record- Continued

```
data postbase;  
do until (last.usubjid);  
  *** Only keep non-missing pre-dose values;  
  set save.vitals (where=(not(visitn <= 2 and hr is missing)));  
  by usubjid visitn;  
  if visitn <= 2 then bl = hr;  
  else do;  
    chgbl = hr - bl;  
    output;  
  end;  
end;  
run;
```

Baseline value  
is assigned

LAST. USUBJID	USUBJID	VISITC	VISITN	HR	FIRST. USUBJID	FIRST. VISITN	LAST. VISITN	BL	CHGBL
0	1	Screening	1	91	1	1	1	91	.

# Next Iteration

```
data postbase;
```

```
do until (last.usubjid);
```

```
*** Only keep non-missing pre-dose values;
```

```
set save.vitals (where=(not(visitn <= 2 and hr is missing)));
```

```
by usubjid visitn;
```

```
if visitn <= 2 then bl = hr;
```

```
else do;
```

```
    chgbl = hr - bl;
```

```
    output;
```

```
end;
```

```
end;
```

```
run;
```

Control returns to top of DO-Loop, not to top of DATA step- assigned variables are not reset

LAST. USUBJID	USUBJID	VISITC	VISITN	HR	FIRST. USUBJID	FIRST. VISITN	LAST. VISITN	BL	CHGBL
0	1	Week 1	3	68	0	1	1	91	.

# Next Record

```
data postbase;  
do until (last.usubjid);  
*** Only keep non-missing pre-dose values;  
set save.vitals (where=(not(visitn <= 2 and hr is missing)));  
by usubjid visitn;  
if visitn <= 2 then bl = hr;  
else do;  
  chgbl = hr - bl;  
  output;  
end;  
end;  
run;
```

Next record is read in, and  
change from baseline  
calculated

LAST. USUBJID	USUBJID	VISITC	VISITN	HR	FIRST. USUBJID	FIRST. VISITN	LAST. VISITN	BL	CHGBL
0	1	Week 1	3	68	0	1	1	91	-23

# *Last Record for Subject*

## **data postbase;**

```
do until (last.usubjid);
```

```
  *** Only keep non-missing pre-dose values;
```

```
  set save.vitals (where=(not(visitn <= 2 and hr is missing)));
```

```
  by usubjid visitn;
```

```
  if visitn <= 2 then bl = hr;
```

```
  else do;
```

```
    chgbl = hr - bl;
```

```
    output;
```

```
  end;
```

```
end;
```

```
run;
```

Top of DATA step- variables coming from dataset or created by SAS have not been overwritten yet: Assigned variables are reinitialized.

LAST. USUBJID	USUBJID	VISITC	VISITN	HR	FIRST. USUBJID	FIRST. VISITN	LAST. VISITN	BL	CHGBL
<b>1</b>	<b>1</b>	<b>Week 4</b>	<b>5</b>	<b>59</b>	<b>0</b>	<b>1</b>	<b>1</b>	<b>.</b>	<b>.</b>

# *Summary*

- Explicit DO-Loop prevents SAS from reinitializing assigned variables until end of BY-group processing
- Enables one-pass change from baseline calculations

# Multi-Column Transpose

- Original dataset (sorted by ANALYTEC and LISTNUM):

ANALYTEC	LISTTYPE	LISTNUM	RXGRP	PATCT	DENOM
CALCIUM	Subjects With One or More Abnormal Screen/Baseline Values	1	1	6	260
CALCIUM	Subjects With One or More Abnormal Screen/Baseline Values	1	2	4	227
CALCIUM	Subjects With Normal Screen/Baseline Values and One Abnormal Value After Initiation of Treatment	2	1	5	260
CALCIUM	Subjects With Normal Screen/Baseline Values and One Abnormal Value After Initiation of Treatment	2	2	6	227

# Goal

- Desired dataset:

ANALYTEC	LISTTYPE	LISTNUM	PATCT1	PATCT2	DENOM1	DENOM2
CALCIUM	Subjects With One or More Abnormal Screen/Baseline Values	1	6	4	260	227
CALCIUM	Subjects With Normal Screen/Baseline Values and One Abnormal Value After Initiation of Treatment	2	5	6	260	227

# *PROC TRANSPOSE Limitation*

- Multiple variables in VAR statement → one record per variable in output dataset

# *Solution*

- Run a separate PROC TRANSPOSE for each variable, and then merge resulting datasets.

# *Drawbacks*

- Requires a separate pass through the dataset for each PROC TRANSPOSE
- Requires another pass through each output dataset for the merge

# *Solution- DOW-Loop*

- Moves DATA step SET statement inside of a DO-Loop
  - Gives complete control over retention of variable values and timing of the population of Program Data Vector (PDV)
  - Requires just one pass through the dataset

# *Resulting Code*

```
data out_ct (drop=patct denom rxgrp);  
  array patcts (*) patct1-patct2;  
  array denoms (*) denom1-denom2;  
  
do until (last.listnum or eof);  
  set out_ct1 end=eof;  
  by analytec listnum;  
  
  patcts(rxgrp) = patct;  
  denoms(rxgrp) = denom;  
end;  
  
output;  
run;
```

# *How It Works (Part 1)*

- Initialization
  - DATA statement initializes PDV
  - ARRAY statements are only executed the first time through the DATA step

# *Resulting Code*

```
data out_ct (drop=patct denom rxgrp);  
  array patcts (*) patct1-patct2;  
  array denoms (*) denom1-denom2;
```

```
do until (last.listnum or eof);  
  set out_ct1 end=eof;  
  by analytec listnum;
```

```
    patcts(rxgrp) = patct;  
    denoms(rxgrp) = denom;  
end;
```

```
output;  
run;
```

## *How It Works (Part 2)*

- Individual record processing
  - DO-loop executes over all records with the same ANALYTEC/LISTNUM value
  - SET statement populates PDV
  - Appropriate array elements are assigned values based on the value of RXGRP
  - Note that array elements are retained until all BY LISTNUM processing is done

# *Resulting Code*

```
data out_ct (drop=patct denom rxgrp);  
  array patcts (*) patct1-patct2;  
  array denoms (*) denom1-denom2;  
  
do until (last.listnum or eof);  
  set out_ct1 end=eof;  
  by analytec listnum;  
  
  patcts(rxgrp) = patct;  
  denoms(rxgrp) = denom;  
end;  
  
output;  
run;
```

# *How It Works (Part 3)*

- Output
  - OUTPUT statement is executed after the DO-loop completes
  - Output dataset is 1 record per ANALYTEC/LISTNUM

# *Resulting Code*

```
data out_ct (drop=patct denom rxgrp);  
  array patcts (*) patct1-patct2;  
  array denoms (*) denom1-denom2;
```

```
do until (last.listnum or eof);  
  set out_ct1 end=eof;  
  by analytec listnum;
```

```
  patcts(rxgrp) = patct;  
  denoms(rxgrp) = denom;  
end;
```

```
output;
```

```
run;
```

# *Conclusion*

- The DOW-Loop is an extremely powerful technique
  - Takes advantage of implicit retain of assigned variables until completion of by-group processing
  - Reduces the number of passes through a dataset required by the program
  - Fewer passes → faster programs!

# *Acknowledgments*

- Paul Dorfman, Ian Whitlock and Don Henderson for their many SAS-L postings demonstrating this technique
- Venky Chakravarthy, “RETAIN or NOT? Is LAG Far Behind?”, PharmaSUG 2005 Conference Proceedings
- Paul Dorfman, “The DOW-Loop Unrolled”, PharmaSUG 2008 Conference Proceedings

*Conference papers housed on <http://www.lexjansen.com>*

# *Contact Information*

Nancy Brucken

i3 Statprobe

Nancy.Brucken@i3statprobe.com

*Questions?*

