



Science For A Better Life

Coding Solutions for Resampling Methods with SAS

Volker Ham and Richardus Vonk, PhUSE Conference 2013, Brussels

Resampling

Resampling is a way of constructing a population distribution by sampling from samples.

Reason to do that is often

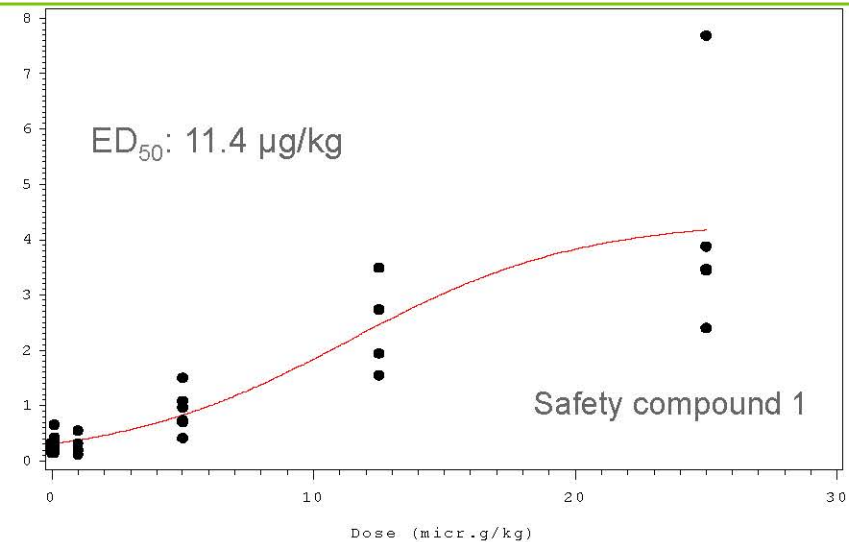
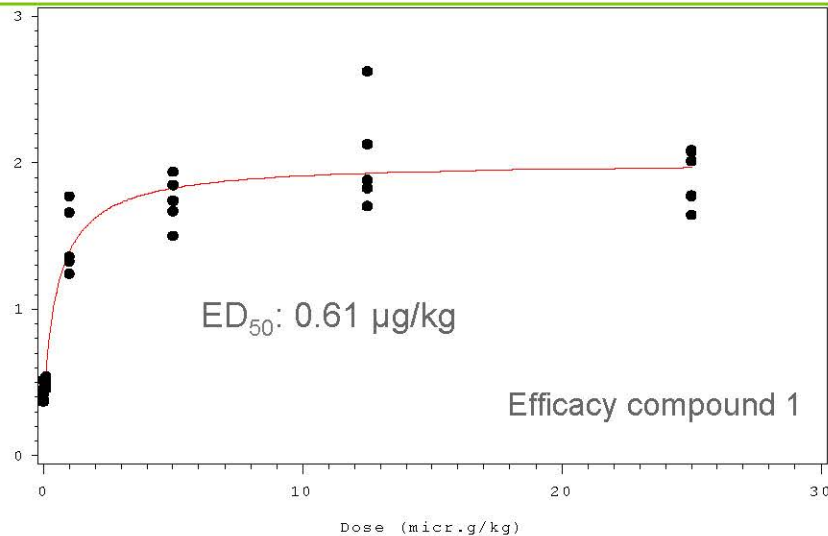
- that assumptions for theoretical distribution are not met or
- the structure of the estimator is too complex.

There are several ways to resample, i. e. building resampling schemes as

- Drawing randomly from the sample: Bootstrapping
- Drawing designed subsets from our sample: Jackknifing, Cross-validation, ...
- Pull the data but exchange the labels on the data points: permutation tests, randomization tests

In this presentation we will concentrate on building and evaluating bootstrap resampling schemes.

Example - Dissociation Effects



Assessment of dissociation $D_i = \frac{ED_{50,S}}{ED_{50,E}}$

- $D_i > 1$ indicates favorable dissociation

Results

- Dissociation factor of 18.7, 95% bootstrap CI [9.9, 34.0]
- $D_{\text{test}} / D_{\text{control}}$ as a comparison criterion

Assessment of relevance through bootstrapping

Otto, Ch, Fuchs, I, Altmann, H, Klewer, M, Walter, A, Prella, K, Vonk, R, Fritzemeier, K (2008). Comparative Analysis of the Uterine and Mammary Gland Effects of Drospirone and Medroxyprogesterone Acetate. Endocrinology, 149: 3952-3959

Bootstrapping

Quite common in statistical research to assess properties of a statistic, if underlying (theoretical) distribution is not known

Random sampling, with replacement, from a dataset

Basically:

- Start with set a of values of size N
- Draw a random sample from these values, with replacement, also of size N
- Perform your analysis on these samples
- Repeat these steps lots of times (10,000 or so...)
- Use the results of these 10,000 analyses to draw conclusions

How can this be done in SAS?

Bootstrapping in SAS

Available in some procedures, eg

- PROC GLM
- PROC MIXED
- PROC MULTTEST

See also: Usage Note 22220: Procedures with bootstrapping, crossvalidation, or jackknifing capabilities

But for more custom problems, we need to program this ourselves. There are multiple ways to do this:

- In data steps
- Using PROC IML
- Using PROC SURVEYSELECT
- “SAS” macros

Example Data: Survival Time in Days for Guinea Pigs



Create the data set

```
data GuineaPigs;  
  AnimalNo = _n_;  
  input SurvivalTime @@;  
  datalines;  
  43 45 53 56 56 57 58 66 67 73 74 79 80 80 81 81 81 82 83 83 84 88 89 91  
  91 92 92 97 99 99 100 100 101 102 102 102 103 104 107 108 109 113 114 118  
  121 123 126 128 137 138 139 144 145 147 156 162 174 178 179 184 191 198  
  211 214 243 249 329 380 403 511 522 598  
  ;  
run;
```

T. Bjerkedal, "Acquisition of resistance in guinea pigs infected with different doses of virulent tubercle bacilli," *American Journal of Hygiene*, 72 (1960), pp. 130–148.

Create histogram and QQ plot

```
ods noproctitle;  
ods graphics on;  
  
proc univariate data = GuineaPigs noprint;  
  histogram SurvivalTime /kernel vscale = count;  
  inset n mean median /position = ne;  
  qqplot SurvivalTime /normal(mu = est sigm a= est) square;  
run;  
  
ods graphics off;
```

Use of proc surveyselect

```
proc surveyselect method = urs  
  data = GuineaPigs  
  out = BootDS  
  sampsize = 72  
  reps = 10000;  
run;
```

Input data
set

Data set containing
bootstrap resamples

Number of replicates
(bootstrap resamples)

Samplesize for
each bootstrap

Note:

- Use "SAMPRATE = 1" instead of SAMPSIZE
- Use "OUTHITS" to obtain one observation for each bootstrap sample in the output dataset (Cave: increased size!)
- URS = Unrestricted Random Sampling selects units with equal probability and with replacement



Use of proc surveyselect

Reduced data set

```
* set number of resamples to be used;
%let NoOfResamples = 1000;

* get the sample size of the original data;
%let SampleSize =;

data _null_;
    call symputx ('SampleSize', NObs);

    if 0;
        set GuineaPigs nobs = NObs;
run;

* reduced output data set;
proc surveyselect method = urs
    data = GuineaPigs
    out = BootDS
    sampsize = &SampleSize.
    reps = &NoOfResamples.;
run;
```

Full data set

```
* set number of resamples to be used;
%let NoOfResamples = 1000;

* full output data set;
proc surveyselect method = urs samprate = 1 outhits
    data = GuineaPigs
    out = BootDS
    reps = &NoOfResamples.;
run;
```

Bootstrap Resamples

Reduced data set

Replicate	Animal No	Survival Time	Number Hits
:	:	:	:
:	:	:	:
3	1	43	1
3	3	53	1
3	4	56	3
3	5	56	1
3	6	57	1
3	7	58	3
3	12	79	3
3	13	80	1
3	14	80	1
3	15	81	3
:	:	:	:
:	:	:	:

Full data set

Replicate	Animal No	Survival Time	Number Hits
:	:	:	:
:	:	:	:
3	1	43	1
3	3	53	1
3	4	56	3
3	4	56	3
3	4	56	3
3	5	56	1
3	6	57	1
3	7	58	3
3	7	58	3
3	7	58	3
3	12	79	3
3	12	79	3
3	12	79	3
3	13	80	1
3	14	80	1
3	15	81	3
3	15	81	3
3	15	81	3
:	:	:	:
:	:	:	:



Bootstrap through Data Step

```
data BootDS (drop = i);  
  call streaminit (4321);  
  
  do Replicate = 1 TO &NoOfResamples.;  
  
    * SampleSize is determined before the data step runs due to the nobs = option;  
    do i = 1 to SampleSize;  
  
      * Choice is randomly selected from values 1 to SampleSize;  
      Choice = ceil (SampleSize*rand ('Uniform'));  
      set GuineaPigs nobs = SampleSize point = Choice;  
      output;  
    end;  
  end;  
  
  * Required when using the point = option;  
  stop;  
run;
```

Adapted from Wicklin, Rick. (2013). Simulating Data with SAS®, Cary, NC

Bootstrap in PROC IML

```
proc iml;
start SampleReplace(A, n, k);
    r = j(n, k);
    call randgen (r, "Uniform");
    r = ceil(nrow(A)*ncol(A)*r);
    return(shape(A[r], n));
finish;
store module=(SampleReplace);
quit;

proc iml;
call randseed (4321);
load module = (SampleReplace);
use GuineaPigs;
read all var {SurvivalTime} into x;
close GuineaPigs;

* get all bootstrap resamples with a single call;
s = SampleReplace(x, nrow(x), &NoOfResamples.);
create Bootiml from s;
append from s;
quit;
```

Adapted from Wicklin, Rick. (2013). Simulating Data with SAS®, Cary, NC



Getting the bootstrap estimates: Caveat!

Keep the output small when looping through the resamples!

```
%macro ODSoff();  
    * reduce ods output prior to BY-group processing;  
    ods graphics off;  
    ods exclude all;  
    ods noresults;  
%mend;  
  
%macro ODSOn();  
    * get full ods output after to BY-group processing;  
    ods graphics on;  
    ods exclude none;  
    ods results;  
%mend;
```

Try to avoid appending data sets when looping through the resamples!

Do not call macros when looping through the resamples!

Adapted from Wicklin, Rick. (2013). Simulating Data with SAS®, Cary, NC

Getting the Bootstrap estimates

Create bootstrap sampling distribution (BSD) and get estimates

```
%ODSoff;

proc univariate data = BootDS noprint;
  by Replicate;
  freq NumberHits;
  var SurvivalTime;
  output out = BSD mean = Mean;
run;

proc univariate data = BSD noprint;
  var Mean;
  output out = PctlMethod
    mean = MeanBSD
    median = MedianBSD
    pctlpts = 2.5 97.5 pctlpre = P;
run;

%ODSon;
```

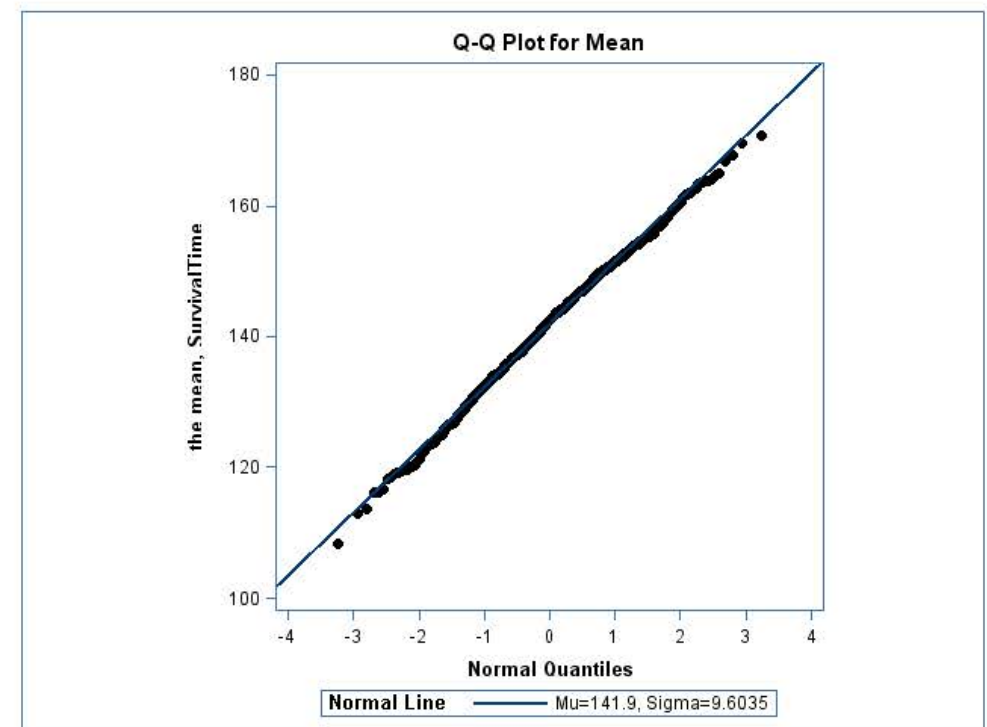
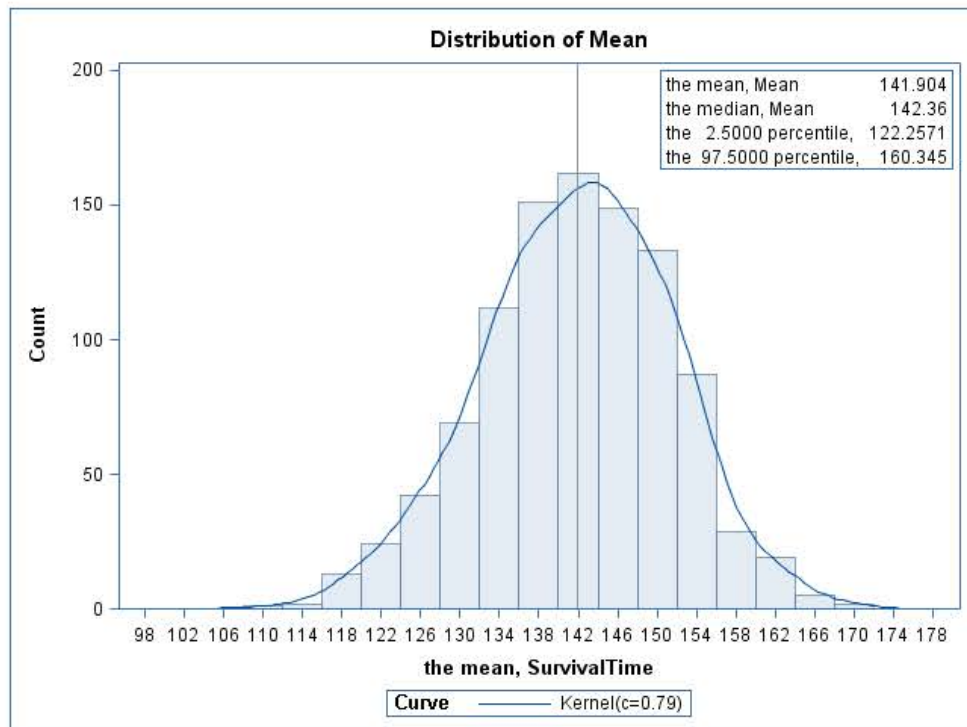
Display BSD and estimates

```
proc transpose data = PctlMethod out = out;
run;

data Estimates;
  set out (rename = (coll = _value_));
  if _name_ = 'MeanBSD' then
    call symputx ('MeanBSD', _value_);
run;

proc univariate data = BSD noprint;
  histogram Mean
    /kernel
    vscale = count
    href = &MeanBSD.;
  inset data = Estimates /position = ne;
  qqplot Mean
    /normal(mu=est sigma=est) square;
run;
```

Guinea pigs: Results



Macros by SAS

%jack and %boot macros

“Trick”: create a %analyze macro that contains the code to analyze the original sample and each resampled data set

```
%macro analyze(data =, out =);  
    proc univariate data = &data;  
        var SurvivalTime;  
        output out = =&out. mean = &mean.;  
        %bystmt;  
    run;  
%mend;  
  
%ODSoff;  
%boot(data = GuineaPigs, samples = 1000, random = 4321);  
%ODSon;  
%bootci(pctl);
```

<http://support.sas.com/kb/24/982.html>



Results from SAS macros

from %boot

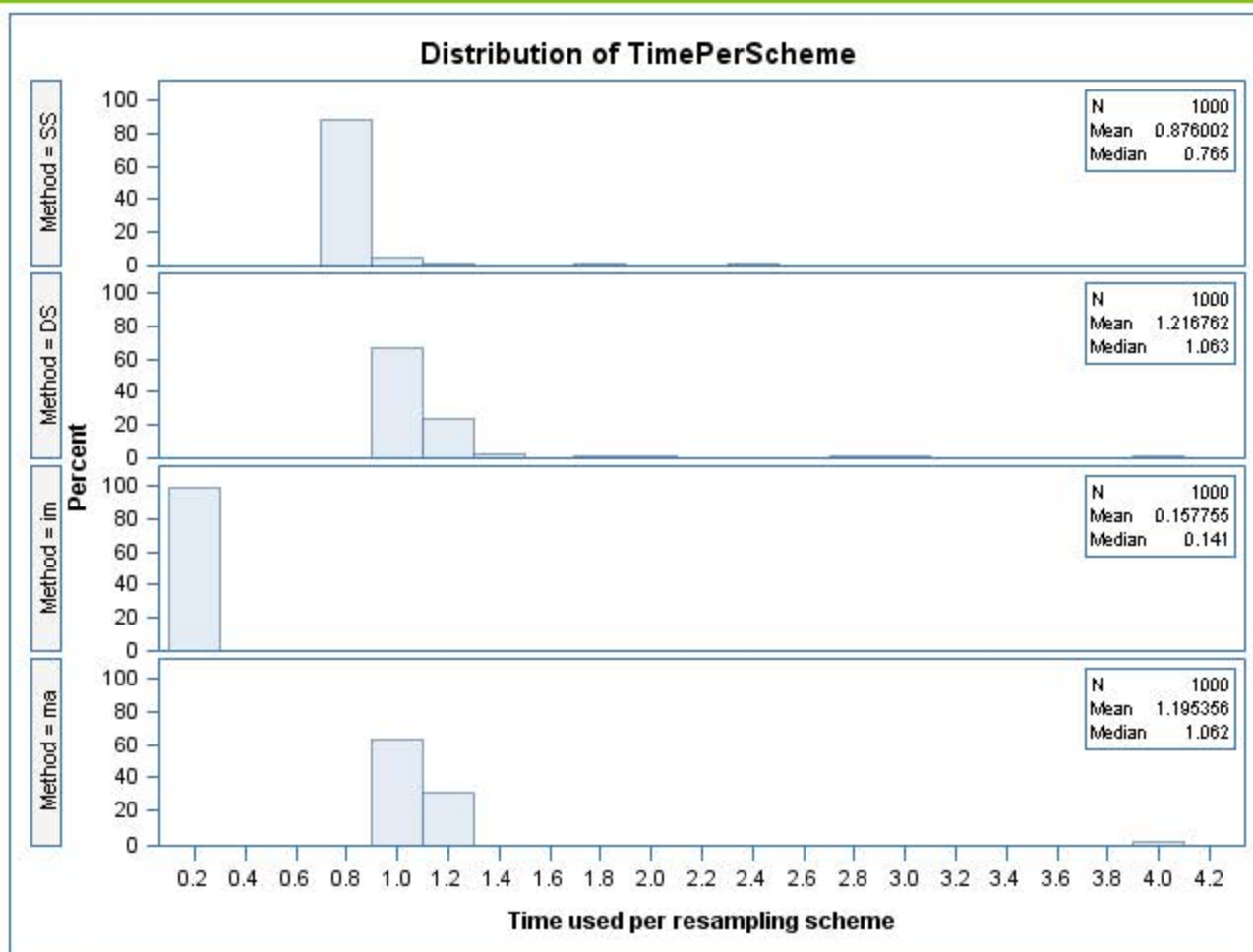
Name	Observed Statistic	Bootstrap Mean	Approximate Bias	Approximate Standard Error	Approximate Lower Confidence Limit	Bias-Corrected Statistic	Approximate Upper Confidence Limit
mean	141.847	141.769	-0.078194	12.9624	116.520	141.925	167.331

Name	Confidence Level (%)	Method for Confidence Interval	Minimum Resampled Estimate	Maximum Resampled Estimate	Number of Resamples	LABEL OF FORMER VARIABLE
mean	95	Bootstrap Normal	105.444	189.486	1000	the mean, survtime

from %bootCI

Name	Observed Statistic	Approximate Lower Confidence Limit	Approximate Upper Confidence Limit	Confidence Level (%)	Method for Confidence Interval	Number of Resamples	LABEL OF FORMER VARIABLE
mean	141.847	117.729	169.25	95	Bootstrap pctl	1000	the mean, survtime

Comparison of Runtimes



10,000 bootstraps, repeated 1000 times



Take Home

Several options available to implement resampling methods in SAS

- In procedures, or custom build

Bootstrap methods are rather straightforward

- PROC SURVEYSELECT can be used to build “bootstrap” dataset(s)
- May be problematic for very large datasets – then data steps can be used
- Macros provided by SAS do work.
- proc iml seems to be fast

proc univariate does a great job in visualizing results.

Care for the Caveats



Science For A Better Life

Thank you!

How does the Bootstrap Perform for Normally Distributed Data?

Sampling Distribution of the Mean

- The sampling distribution of the mean is well understood.
- The central limit theorem says that the sampling distribution of the mean is approximately normally distributed with parameters (μ, σ) .
- The sample mean is an estimate for μ .
- An estimate for σ is the quantity $\sqrt{s^2/n}$, where s^2 is the sample variance and n is the size of the sample.
- Furthermore, the two-sided $100(1-\alpha)\%$ confidence interval has upper and lower limits given by $\bar{x} \pm t_{1-\alpha/2, n-1} \sqrt{s^2/n}$, where $t_{1-\alpha/2, n-1}$ is the $1 - \alpha/2$ percentile of the Student's t distribution with $n - 1$ degrees of freedom.

How does the Bootstrap Perform for Normally Distributed Data?

