# *Sample size by simulation for clinical trials with survival outcomes: the **simsam** package in action*

**Richard Hooper**

Senior Lecturer in Medical Statistics

Barts and The London
School of Medicine and Dentistry

Queen Mary
University of London

## The **`simsam`** package

**`simsam`** uses simulation to determine the sample size required to achieve given statistical power to detect a given effect, for *any* hypothesis test under *any* statistical model that can be programmed in Stata.

**Hooper R. Versatile sample size calculation using simulation.**
***Stata Journal*** **2013;13(1):21-38**

## Why worry about sample size?

“The number of subjects in a clinical trial should always be large enough to provide a reliable answer to the questions addressed. This number is usually determined by the primary objective of the trial.”

**International Conference on Harmonisation of technical requirements for registration of pharmaceuticals for human use**

## Why worry about sample size?

"The number of subjects in a clinical trial should always be large enough to provide a reliable answer to the questions addressed. This number is usually determined by the primary objective of the trial."

**International Conference on Harmonisation of technical requirements for registration of pharmaceuticals for human use**

"For scientific and ethical reasons, the sample size for a trial needs to be planned carefully, with a balance between medical and statistical considerations."

**CONSORT statement on the reporting of clinical trials, endorsed by leading general medical journals**

## Why worry about sample size?

“The number of subjects in a clinical trial should always be large enough to provide a reliable answer to the questions addressed. This number is usually determined by the primary objective of the trial.”

**International Conference on Harmonisation of technical requirements for registration of pharmaceuticals for human use**

“For scientific and ethical reasons, the sample size for a trial needs to be planned carefully, with a balance between medical and statistical considerations.”

**CONSORT statement on the reporting of clinical trials, endorsed by leading general medical journals**

“This [sample size calculation] is frequently one of the least credible components of a trial [funding] application.”

**UK National Institute for Health Research**

## Basic syntax of **simsam**

```
. simsam subcommand_name n_name , ///
>       detect(parameter_name(parameter_value))   ///
>       null(parameter_name(null_value))   ///
>       assuming(nuisance_parameter1(par1_value) … )   ///
>       p(.8) inc(10) prec(0.01)
```

where *subcommand_name* is the name of a user-written program which codes the statistical model and the hypothesis test

## Basic syntax of **`simsam`**

```
. simsam subcommand_name n_name ,  ///
>       detect(parameter_name(parameter_value))   ///
>       null(parameter_name(null_value))   ///
>       assuming(nuisance_parameter1(par1_value)  … )   ///
>       p(.8) inc(10) prec(0.01)
```

where *subcommand_name* is the name of a user-written program which codes the statistical model and the hypothesis test

NB **`simsam`** doesn't do anything by itself – **it needs software**

# A modular view of a `simsam` subcommand

```
program define subcommand_name, rclass

        syntax , n_name(integer)  ///
              parameter_name(real)  ///
              nuisance_parameter1(real)  ///
              :

        drop _all

        [generate data-set]

        [analyse data-set]

        return scalar p = expression_for_pvalue

end
```

## Something more complex: a two-stage adaptive design

```
program define subcommand_name, rclass
    syntax , …
    drop _all
    [generate data from stage 1]
    [analyse data from stage 1 and calculate p-value]
    [choose to stop there, or else adapt the protocol based
    on stage 1 results, then generate data from stage 2]
    [analyse data from stage 2 and calculate p-value]
    [return a combined p-value from the two stages]
end
```

## Trials with survival (time-to-event) outcomes

For an individually-randomised trial where the outcome is time until death (possibly censored), the total number of deaths that must be observed to detect hazard ratio $\Delta$ with given power is approximately (Schoenfeld, 1983)

$$4(z_{\beta} + z_{1-\alpha/2})/\log^2 \Delta$$

Jahn-Eimermacher et al (2011) extend this to cluster-randomised trials analysed with frailty models, for which the above formula underestimates sample size. Their extended formula still underestimates sample size when the cluster size is variable.

```
program define s_survival, rclass
        syntax , recrdur(integer) recrrate(integer)  ///
            hr(real) failratec(real)  ///
            folldur(real) droprate(real)
        drop _all
        set obs `=`recrdur'*`recrrate''
        gen group=mod(_n,2)
        gen abs_trecr=sum(-log(runiform())/`recrrate'
        gen tfail=-log(runiform())/`failratec'*`hr'^group
        gen tdrop=-log(runiform())/`droprate'
        gen tstop=`recrdur'+`folldur'-abs_trecr
        drop if tstop<0
        gen t=min(tfail, tdrop, tstop)
        gen fail=(t<min(tdrop, tstop)
        stset t, failure(fail)
        stcox group
        return scalar p=2*normal(-abs(_b[group]/_se[group]))
end
```

## Capturing errors in the analysis

- If any untrapped errors occur when you run the subcommand, **`simsam`** will fall over.

## Capturing errors in the analysis

- If any untrapped errors occur when you run the subcommand, `simsam` will fall over.
- **You have to decide how you would handle errors if they occurred in the analysis of the real data.**

## Capturing errors in the analysis

- If any untrapped errors occur when you run the subcommand, **`simsam`** will fall over.
- **You have to decide how you would handle errors if they occurred in the analysis of the real data.**
- Simplest approach: treat the result as non-significant.
  - To do this you just need to exit the subcommand without returning a p-value
  - A general approach is to encase the analysis "module" in **`capture noisily`** brackets

## Capturing errors in the analysis

- If any untrapped errors occur when you run the subcommand, **`simsam`** will fall over.
- **You have to decide how you would handle errors if they occurred in the analysis of the real data.**
- Simplest approach: treat the result as non-significant.
  - To do this you just need to exit the subcommand without returning a p-value

```
capture noisily {
    stcox group
    return scalar p=2*normal(-abs(_b[group]/_se[group]))
}
```

## Capturing errors in the survival analysis

- Assuming the data are legitimate, the only error you are really likely to encounter with **`stcox`** is failure to converge.
  - This turns out to be especially a problem with frailty analyses of cluster-randomised trials.

## Capturing errors in the survival analysis

- Assuming the data are legitimate, the only error you are really likely to encounter with **`stcox`** is failure to converge.
  - This turns out to be especially a problem with frailty analyses of cluster-randomised trials.
- **You have to decide how you would handle errors if they occurred in the analysis of the real data**.

## Capturing errors in the survival analysis

- Assuming the data are legitimate, the only error you are really likely to encounter with `stcox` is failure to converge.
  - This turns out to be especially a problem with frailty analyses of cluster-randomised trials.
- **You have to decide how you would handle errors if they occurred in the analysis of the real data**.
- *i.e.* you need to specify an Analysis Plan

## Capturing errors in the survival analysis

*e.g.* if Cox regression fails to converge, try parametric regression with a Weibull model for survival times

```
capture noisily {
    stcox group
    return scalar p=2*normal(-abs(_b[group]/_se[group]))
}
if _rc~=0 {
    streg group, dist(weibull)
    return scalar p=2*normal(-abs(_b[group]/_se[group]))
}
```

# Convergence problems that don't lead to errors: controlling the number of iterations used for estimation

- Generally **`stcox`** converges after a few iterations
- Very occasionally it will continue on to the maximum number of iterations (16,000 by default) without producing a non-convergence error
- Hence **`simsam`** will appear to be hung up but will not halt with an error message

## Convergence problems that don't lead to errors: controlling the number of iterations used for estimation

The solution is to re-set the maximum number of iterations:

```
. set maxiter 20
. simsam s_survival recrrate, ///
>     detect(hr(1.5)) null(hr(1.0))  ///
>     assuming(failratec(0.5) ///
>         recrdur(2) folldur(1))  ///
>     p(.8) inc(1) prec(0.001)
```

```
-----------------------------------------------------
iteration recrrate              power (99% CI)
-----------------------------------------------------
        1      100 .......... 0.6500 (0.5172, 0.7681)
        2      143 .......... 0.8120 (0.7782, 0.8428)
        3      139 .......... 0.7971 (0.7866, 0.8074)
        4      141 .......... 0.8004 (0.7972, 0.8037)
        5      141 .......... 0.8009 (0.7999, 0.8019)
        6      140 .......... 0.7988 (0.7978, 0.7998)
-----------------------------------------------------
     null      141 .......... 0.0499 (0.0489, 0.0509)
-----------------------------------------------------

      recrrate = 141
        achieves 80.09% power (99% CI 79.99, 80.19)
          at the 5% significance level
    to detect
            hr = 1.5
     assuming
     failratec = 0.5
       recrdur = 2
       folldur = 1

     under null:  4.99% power (99% CI  4.89,  5.09)
```

## Concluding remarks

Simulation for sample size calculation

- is accurate and versatile
- but must anticipate every contingency
- needs statistician input
- forces you to think about the analysis in detail (no bad thing)
- helps others to develop related applications

- More info at http://webspace.qmul.ac.uk/rlhooper/simsam
- simsam update planned for Jan 2014

## Thank you