R code and packages to facilitate HTA: Industry context

Iain Bennett - Global Access Evidence
Tadeusz Lewandowski - Pharma Development Data Sciences Acceleration

Role of Open Source R packages in industry

Introduction & requirements

Example 1: rTables

Example 2: flexsurvPlus

Questions

Introduction

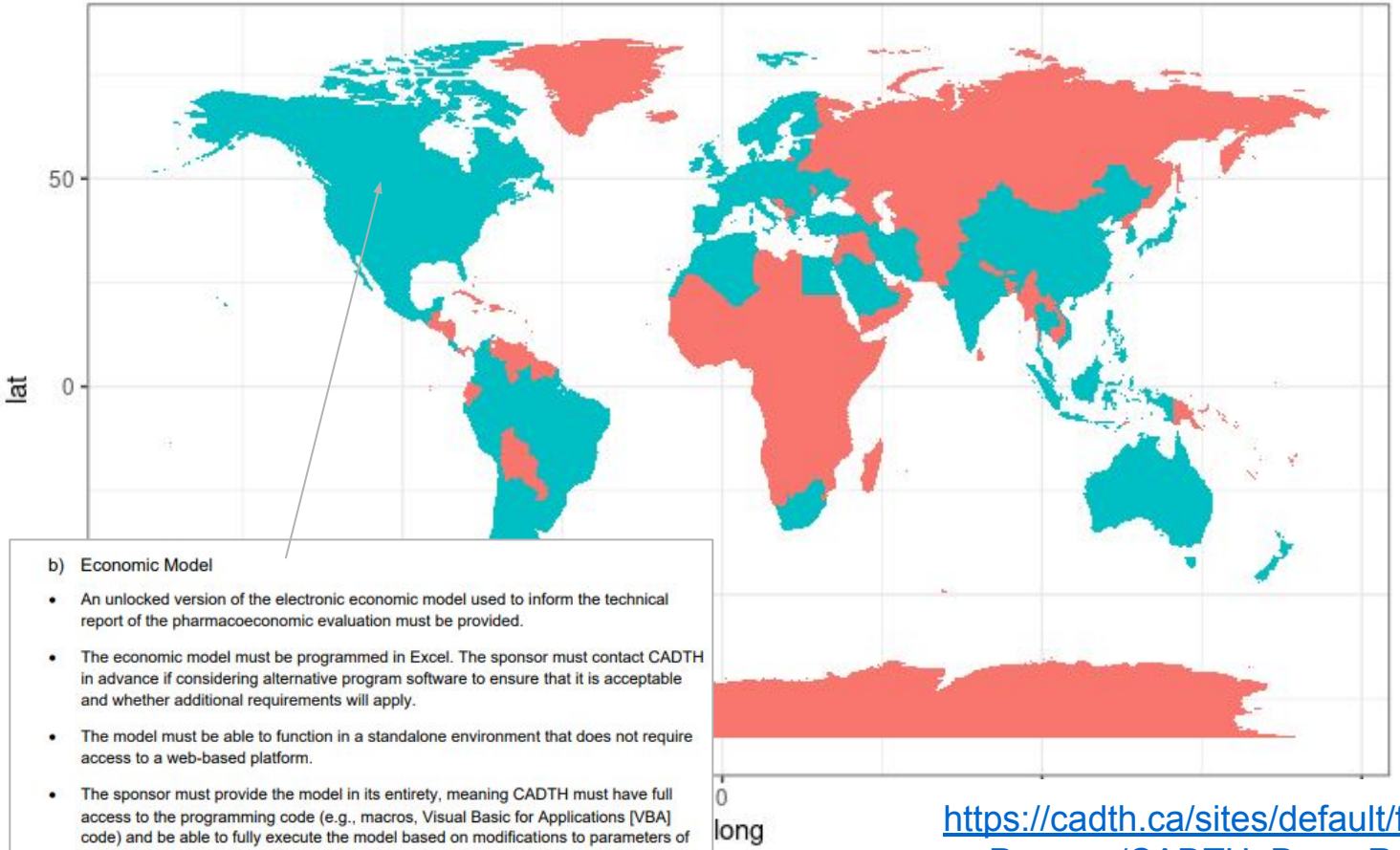
Two motivating requirements

Global focus for HTA analysis

- Global access support HTA Evidence needs across multiple countries
- Requirements differ greatly but Excel and Word based file formats typically accepted

GCP requirements on Computer Software validity

- Validation of Computerized Systems A process of establishing and documenting that the specified requirements of a computerized system can be consistently fulfilled from design until decommissioning:
  - ICH E6 (R2) Good clinical practice
  - E6(R2) Good Clinical Practice: Integrated Addendum to ICH E6(R1)



b) Economic Model

- An unlocked version of the electronic economic model used to inform the technical report of the pharmacoeconomic evaluation must be provided.
- The economic model must be programmed in Excel. The sponsor must contact CADTH in advance if considering alternative program software to ensure that it is acceptable and whether additional requirements will apply.
- The model must be able to function in a standalone environment that does not require access to a web-based platform.
- The sponsor must provide the model in its entirety, meaning CADTH must have full access to the programming code (e.g., macros, Visual Basic for Applications [VBA] code) and be able to fully execute the model based on modifications to parameters of interest. CADTH must be able to vary individual parameters, view the calculations, and run the model to generate results.

Recent download of Excel CE model

FALSE
TRUE

https://cadth.ca/sites/default/files/Drug_Review_Process/CADTH_Drug_Reimbursement_Review_Procedures.pdf

If Excel is so needed how does Roche use R today with a HTA focus?

- Internal models to inform decision making e.g. early CE models to explore trial design choices are typically built in R Shiny
- Account management tools when presented by a Roche employee on a Roche laptop e.g. BIMs (R Shiny nice for this, however, ipad compatibility limits uptake)
- Statistical analysis for inclusion in Excel based models/word dossiers
  - For Bayesian analysis (typically to run and report JAGS models)
  - For analysis of clinical trial data

Case study 1

rTables

Step 1 - installing the packages

Repository link for multiple packages:

https://github.com/Roche/rtables

The `rtables` R package was designed to create and display complex tables with R. Currently, `rtables` can be outputted in `ascii` and `html`.*

`rtables` is now available on CRAN and you can install the latest released version with:

```
install.packages("rtables")
```

* https://github.com/Roche/rtables

Step 2 - building the summary output

```
library(rtables)

lyt <- basic_table() %>%
  split_cols_by("ARM")

build_table(lyt, ex_adsl)
```

```
   A: Drug X   B: Placebo   C: Combination
-----------------------------------------
```

Step 2 - building the summary output

```
library(rtables)
library(dplyr)

lyt <- basic_table() %>%
  split_cols_by("ARM") %>%
  analyze(c("AGE", "BMRKR2"), function(x, ...) {
    if (is.numeric(x)) {
      in_rows(
        "Mean (sd)" = c(mean(x), sd(x)),
        "Median" = median(x),
        "Min - Max" = range(x),
        .formats = c("xx.xx (xx.xx)", "xx.xx", "xx.xx - xx.xx")
      )
    } else if (is.factor(x) || is.character(x)) {
      in_rows(.list = list_wrap_x(table)(x))
    } else {
      stop("type not supported")
    }
  })

build_table(lyt, ex_adsl)
```

	A: Drug X	B: Placebo	C: Combination
AGE			
Mean (sd)	33.77 (6.55)	35.43 (7.9)	35.43 (7.72)
Median	33	35	35
Min - Max	21 - 50	21 - 62	20 - 69
BMRKR2			
LOW	50	45	40
MEDIUM	37	56	42
HIGH	47	33	50

Step 2 - building the summary output

```r
library(rtables)
library(dplyr)

lyt <- basic_table() %>%
  split_cols_by("ARM") %>%
  add_colcounts() %>%
  analyze(c("AGE", "BMRKR2"), function(x, ...) {
    if (is.numeric(x)) {
      in_rows(
        "Mean (sd)" = c(mean(x), sd(x)),
        "Median" = median(x),
        "Min - Max" = range(x),
        .formats = c("xx.xx (xx.xx)", "xx.xx", "xx.xx - xx.xx")
      )
    } else if (is.factor(x) || is.character(x)) {
      in_rows(.list = list_wrap_x(table)(x))
    } else {
      stop("type not supported")
    }
  })

build_table(lyt, ex_adsl)
```

	A: Drug X (N=134)	B: Placebo (N=134)	C: Combination (N=132)
AGE			
Mean (sd)	33.77 (6.55)	35.43 (7.9)	35.43 (7.72)
Median	33	35	35
Min - Max	21 - 50	21 - 62	20 - 69
BMRKR2			
LOW	50	45	40
MEDIUM	37	56	42
HIGH	47	33	50

Step 2 - building the summary output

```
library(rtables)
library(dplyr)

lyt <- basic_table() %>%
  split_cols_by("ARM") %>%
  add_colcounts() %>%
  analyze(c("AGE", "BMRKR2", "RACE"), function(x, ...) {
    if (is.numeric(x)) {
      in_rows(
        "Mean (sd)" = c(mean(x), sd(x)),
        "Median" = median(x),
        "Min - Max" = range(x),
        .formats = c("xx.xx (xx.xx)", "xx.xx", "xx.xx - x
      )
    } else if (is.factor(x) || is.character(x)) {
      in_rows(.list = list_wrap_x(table)(x))
    } else {
      stop("type not supported")
    }
  })

build_table(lyt, ex_adsl)
```

	A: Drug X (N=134)	B: Placebo (N=134)	C: Combination (N=132)
AGE			
Mean (sd)	33.77 (6.55)	35.43 (7.9)	35.43 (7.72)
Median	33	35	35
Min - Max	21 - 50	21 - 62	20 - 69
BMRKR2			
LOW	50	45	40
MEDIUM	37	56	42
HIGH	47	33	50
RACE			
ASIAN	68	67	73
BLACK OR AFRICAN AMERICAN	31	28	32
WHITE	27	26	21
AMERICAN INDIAN OR ALASKA NATIVE	8	11	6
MULTIPLE	0	1	0
NATIVE HAWAIIAN OR OTHER PACIFIC ISLANDER	0	1	0
OTHER	0	0	0
UNKNOWN	0	0	0

Step 2 - building the summary output

```r
library(rtables)
library(dplyr)

myfun <- function(x, ...) {
    if (is.numeric(x)) {
      in_rows(
        "Mean (sd)" = c(mean(x), sd(x)),
        "Median" = median(x),
        "Min - Max" = range(x),
        .formats = c("xx.xx (xx.xx)", "xx.xx", "xx.xx - xx.xx")
      )
    } else if (is.factor(x) || is.character(x)) {
      in_rows(.list = list_wrap_x(table)(x))
    } else {
      stop("type not supported")
    }
  }

lyt <- basic_table() %>%
  split_cols_by("ARM") %>%
  add_colcounts() %>%
  analyze(c("AGE", "BMRKR2", "RACE"), myfun)

build_table(lyt, ex_adsl)
```

	A: Drug X (N=134)	B: Placebo (N=134)	C: Combination (N=132)
AGE			
Mean (sd)	33.77 (6.55)	35.43 (7.9)	35.43 (7.72)
Median	33	35	35
Min - Max	21 - 50	21 - 62	20 - 69
BMRKR2			
LOW	50	45	40
MEDIUM	37	56	42
HIGH	47	33	50
RACE			
ASIAN	68	67	73
BLACK OR AFRICAN AMERICAN	31	28	32
WHITE	27	26	21
AMERICAN INDIAN OR ALASKA NATIVE	8	11	6
MULTIPLE	0	1	0
NATIVE HAWAIIAN OR OTHER PACIFIC ISLANDER	0	1	0
OTHER	0	0	0
UNKNOWN	0	0	0

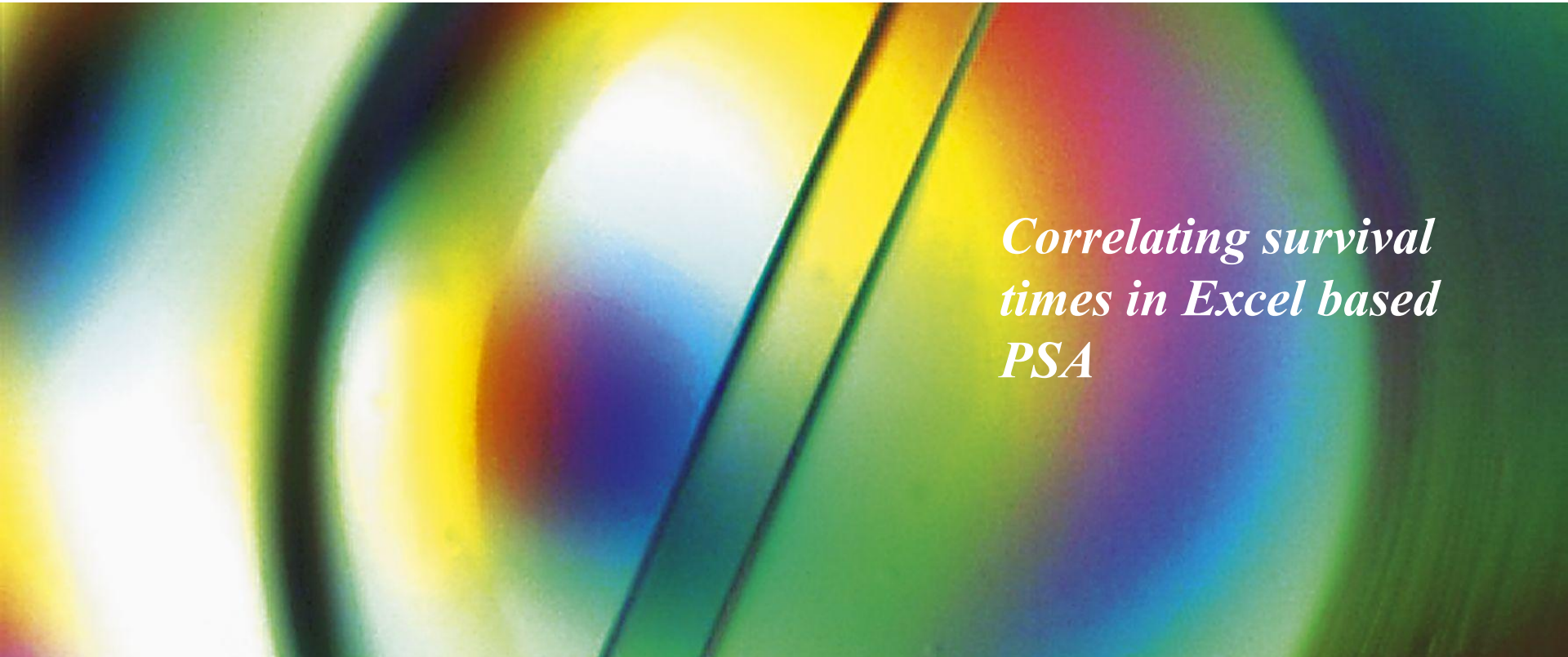
Step 2 - building the summary output

```r
library(rtables)
library(dplyr)

myfun <- function(x, ...) {
    if (is.numeric(x)) {
      in_rows(
        "Mean (sd)" = c(mean(x), sd(x)),
        "Median" = median(x),
        "Min - Max" = range(x),
        .formats = c("xx.xx (xx.xx)", "xx.xx", "xx.xx - xx.xx")
      )
    } else if (is.factor(x) || is.character(x)) {
      in_rows(.list = list_wrap_x(table)(x))
    } else {
      stop("type not supported")
    }
  }

lyt <- basic_table() %>%
  split_cols_by("ARM") %>%
  add_colcounts() %>%
  analyze(c("AGE", "BMRKR2", "RACE"), myfun)

build_table(lyt, ex_adsl %>% filter(RACE == “ASIAN”) )
```

	A: Drug X (N=68)	B: Placebo (N=67)	C: Combination (N=73)
AGE			
Mean (sd)	32.53 (6.02)	36.66 (8.93)	36.92 (8.2)
Median	32	36	35
Min - Max	23 - 48	23 - 62	24 - 69
BMRKR2			
LOW	22	21	18
MEDIUM	17	28	21
HIGH	29	18	34
RACE			
ASIAN	68	67	73
BLACK OR AFRICAN AMERICAN	0	0	0
WHITE	0	0	0
AMERICAN INDIAN OR ALASKA NATIVE	0	0	0
MULTIPLE	0	0	0
NATIVE HAWAIIAN OR OTHER PACIFIC ISLANDER	0	0	0
OTHER	0	0	0
UNKNOWN	0	0	0

Case study 2

flexsurvPlus



Open sourced packages/vignettes

Repository link: https://github.com/Roche/Global-HTA-Evidence-Open

Documentation link: https://roche.github.io/Global-HTA-Evidence-Open/

- **flexsurvPlus** - helper functions to work with flexsurv to fit some standard models used in economic models easily
- **gemtcPlus** - adds helper functions and new models to gemtc to handle NMA reporting and definition of complex models such as fractional polynomials for time to event NMA
- **rpsftmPlus** - helper functions to work with rpsftm package to automate some diagnostic plots and enable easy sensitivity analysis
- **MAIC** - helper functions to simplify execution and reporting of MAICs in a consistent way
- **descem -** This package facilitates performing discrete event simulations without resource constraints for cost-effectiveness analysis - Javier & Valerie

All released under Apache 2.0 open source license with no warranty

Focus on transparency rather than efficiency of coding

Acknowledgements: many Roche colleagues, Mango, Bresmed

Example - flexsurvPlus

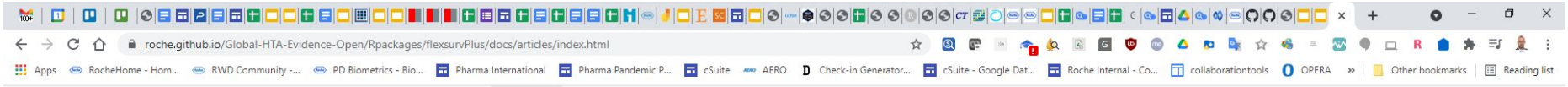
What can it do?

1) Single call to fit 8 parametric forms across different assumptions on treatment effect
    a) For 2 arm trials 3 treatment effect assumptions possible:;
        i) Common shape (flexsurv default) treatment effect only on intercept
        ii) Independent shape with treatment effect on all parameters (e.g. intercept and shape)
        iii) Separate models (calls flexsurv twice for each arm separately
    b) For 1 arm trials only a model without treatment effect possible
2) Simple setup to bootstrap multiple correlated time to event parameters jointly
3) Conversion of flexsurv parameterisations to a SAS Lifereg parameterisation (useful for backward compatibility in existing models)

Example - flexsurvPlus

Why developed or “hang on can’t survHE/flexsurv do this already”?

1) Efficiency of execution multiple models with limited code (replacing SAS macro used previously)
2) Limited dependencies which simplifies use on validated systems
3) Consistency and transparency in model definitions



flexsurvPlus 1.0.1 | Vignettes | Functions | Other Packages

Articles

Introduction

These vignettes illustrate how **flexsurvPlus** can be used to prepare inputs in R for inclusion into Excel based economic models.

Fitting parametric survival models in R
PSA and correlated endpoints (bootstrap approach)

Methods appendix

These vignettes provide more details on the methods used and formulas for parametric forms and related surival and hazard functions.

Parametric survival analysis using the flexsurvPlus package: understanding the theory

STEM appendix

These vignettes describe how to convert to parametrizations that were used in a prior SAS macro. For most users this vignette can be safely ignored.

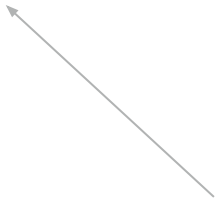
STEM compatibility

https://roche.github.io/Global-HTA-Evidence-Open/Rpackages/flexsurvPlus/docs/articles/index.html

Developed by Iain Bennett, Jo Gregory, Sarah Smith, Richard Birnie.

Site built with pkgdown 1.5.1.

```
install(devtools)

devtools::install_github(
  "roche/Global-HTA-Evidence-Open",
  subdir = "/Rpackages/flexsurvPlus"
   )
```



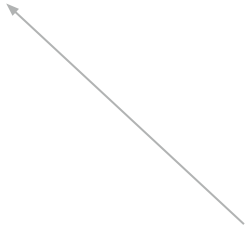
For other packages can change this line

Generating some simulated data

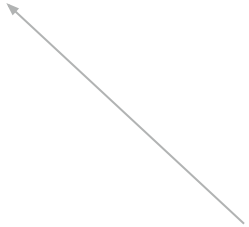
```
adtte <- sim_adtte(
     seed = 2020,
     rho = 0.6
      )
```

https://roche.github.io/Global-HTA-Evidence-Open/Rpackages/flexsurvPlus/docs/reference/sim_adtte.html

```
psm_PFS_all <- runPSM(data=PFS_data,
                      time_var="PFS_days",
                      event_var="PFS_event",
                      model.type= c("Common shape",
                                    "Independent shape",
                                    "Separate"),
                      distr = c('exp',
                                'weibull',
                                'gompertz',
                                'lnorm',
                                'llogis',
                                'gengamma',
                                'gamma',
                                'genf'),
                      strata_var = "ARMCD",
                      int_name="A",
                       ref_name = "B")
```



Treatment effect assumptions



Parametric forms

Example - flexsurvPlus

Why bother with independent shapes models?

Enables testing of common shape assumption for observed data using GLRT as the common shape form is nested in the independent shape form

Separate models retained in case of convergence issues with the additional parameters

Uncertainty?

```
set.seed(2358)
# For speed and illustration only 4 models are selected for
bootstrapping - all could be included.
# To minimize vignette computation time only 100 bootstrap samples are
taken. In general more samples should be used.
n.sim <- 100

PSM_bootstraps_PFS <- boot(
  statistic = bootPSM, # bootstrap function
  R=n.sim, # number of bootstrap samples
  data=PFS_data,
  time_var="PFS_days",
  event_var="PFS_event",
  model.type = c("Common shape", "Separate"),
  distr = c('weibull', 'gamma'),
  strata_var = "ARMCD",
  int_name = "B",
  ref_name = "A"
 )
```

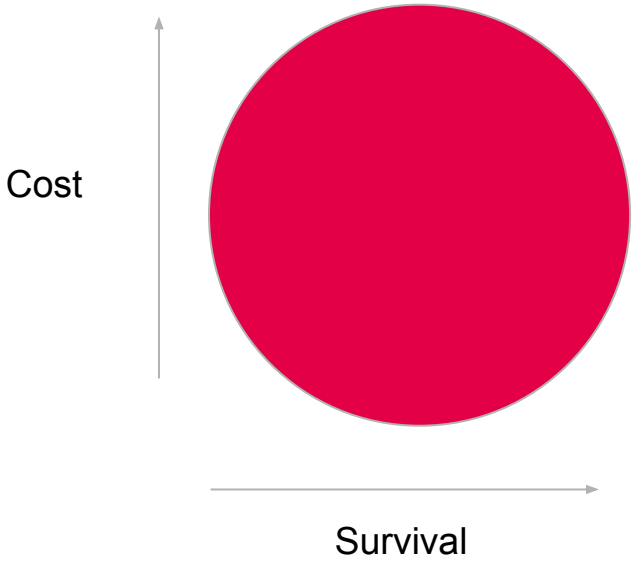
Same syntax to the runPSM function

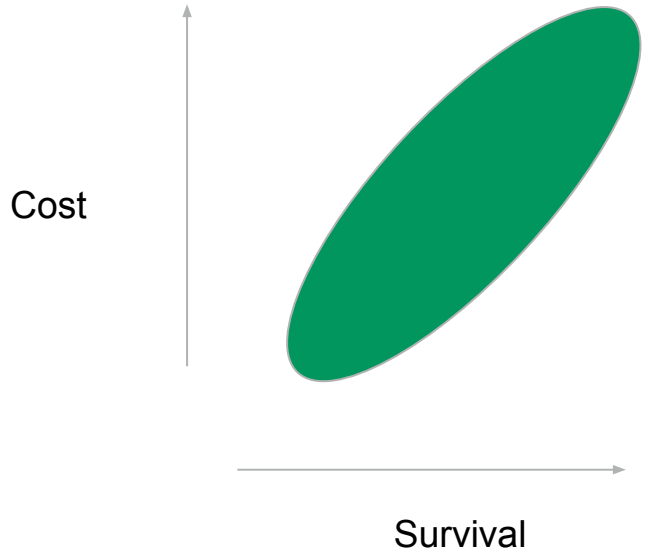
```
PFS_bootsamples <- as_tibble(PSM_bootstraps_PFS$t)

# then add column names so can identify model and parameter more easily
colnames(PFS_bootsamples) <- names(PSM_bootstraps_PFS$t0)

# show the first 3 samples
PFS_bootsamples[1:3,]
#> # A tibble: 3 x 18
#>   comshp.weibull.sca... comshp.weibull.sca... comshp.weibull.sha... comshp.weibull.sh...
#>                   <dbl>                 <dbl>                 <dbl>                 <dbl>
#> 1                  454.                  230.                  1.41                  1.41
#> 2                  453.                  252.                  1.29                  1.29
#> 3                  425.                  224.                  1.42                  1.42
#> # ... with 14 more variables: comshp.weibull.scale.TE <dbl>,
#> #   comshp.gamma.rate.int <dbl>, comshp.gamma.rate.ref <dbl>,
#> #   comshp.gamma.shape.int <dbl>, comshp.gamma.shape.ref <dbl>,
#> #   comshp.gamma.rate.TE <dbl>, sep.weibull.scale.int <dbl>,
#> #   sep.weibull.scale.ref <dbl>, sep.weibull.shape.int <dbl>,
#> #   sep.weibull.shape.ref <dbl>, sep.gamma.rate.int <dbl>,
#> #   sep.gamma.rate.ref <dbl>, sep.gamma.shape.int <dbl>,
#> #   sep.gamma.shape.ref <dbl>
```

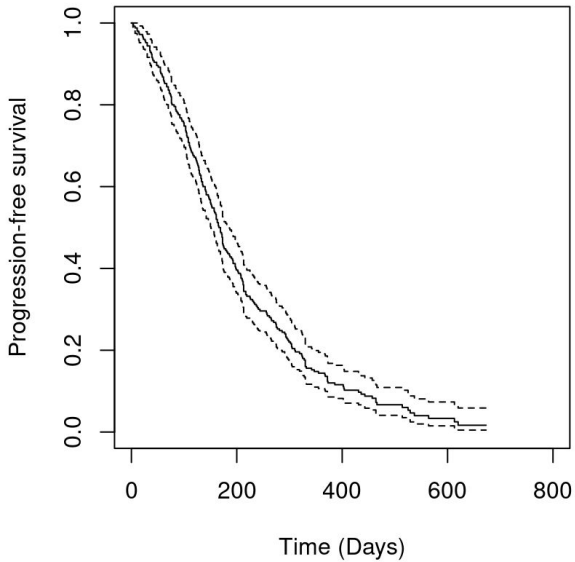
Output for Excel :)

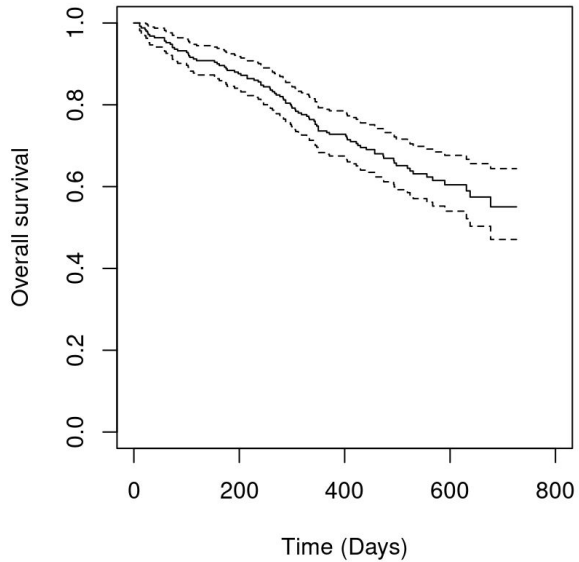
PSA - is to then sample these not Cholesky decomposition

Why care about correlation in PSA?



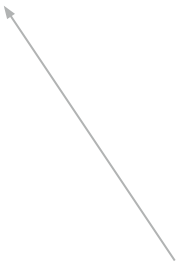



Progression-free survival
0.0
0.2
0.4
0.6
0.8
1.0
0
200
400
600
800
Time (Days)


Overall survival
0.0
0.2
0.4
0.6
0.8
1.0
0
200
400
600
800
Time (Days)

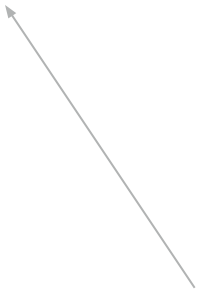
```r
# for illustration and speed only 100 samples used
n.sim <- 100

# define a seed that will affect the random numbers used by boot for sampling
set.seed(2020)

PSM_bootstraps_PFScor <- boot(
  statistic = bootPSM, # bootstrap function
  R=n.sim, # number of bootstrap samples
  data=analysis_data, # the dataset
  time_var="PFS_days", # the time variable
  event_var="PFS_event", # the event variable coded as 1 for event
  model.type= "One arm",
  distr = "weibull", # for speed we only fit a single model but multiple could be specified,
  int_name = "A"  # needed for one arm even as essential meta data
)

# reuse the same seed that will affect the random numbers used by boot for sampling
set.seed(2020)

PSM_bootstraps_OScor <- boot(
  statistic = bootPSM, # bootstrap function
  R=n.sim, # number of bootstrap samples
  data=analysis_data, # the dataset
  time_var="OS_days",  # the time variable
  event_var="OS_event", # the event variable coded as 1 for event
  model.type="One arm", # again for speed only a single model is specified
  distr = "gamma",
  int_name = "A"  # needed for one arm even as essential meta data
  )
```

```r
# this returns the indexes selected in each sample
index_PFScor <- boot.array(PSM_bootstraps_PFScor, indices = TRUE)
index_OScor  <- boot.array(PSM_bootstraps_OScor, indices = TRUE)

# as desired all match between the two sampled sets
all(index_OScor == index_PFScor)
 #> [1] TRUE
```

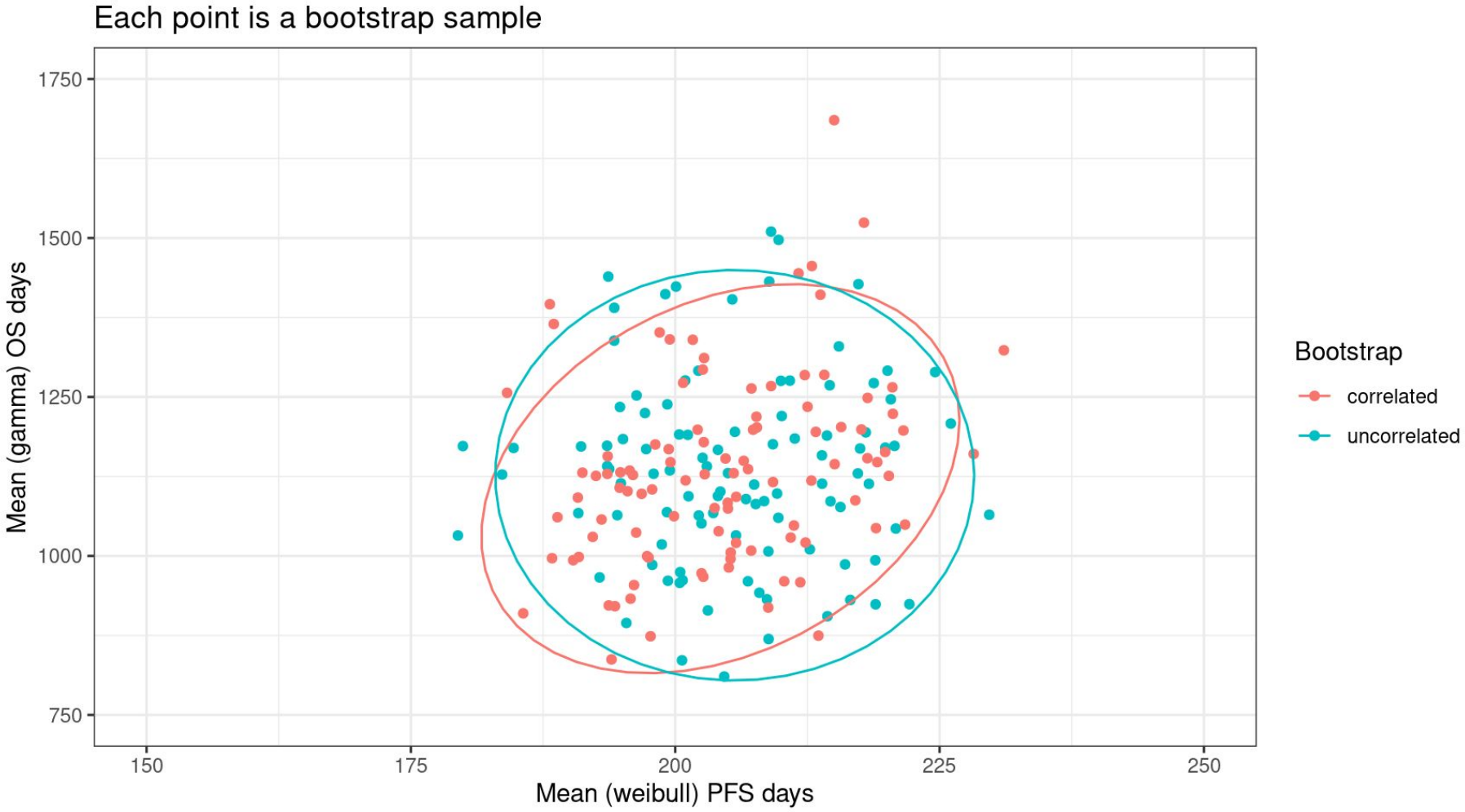


Same underlying samples used

```r
mean_PFS <- with(bootsamples_PFS, flexsurv::mean_weibull(scale =
onearm.weibull.scale.int, shape = onearm.weibull.shape.int))

mean_OS <- with(bootsamples_OS, flexsurv::mean_gamma(shape = onearm.gamma.shape.int, rate
= onearm.gamma.rate.int))

mean_PFScor <- with(bootsamples_PFScor, flexsurv::mean_weibull(scale =
onearm.weibull.scale.int, shape = onearm.weibull.shape.int))

mean_OScor <- with(bootsamples_OScor, flexsurv::mean_gamma(shape =
onearm.gamma.shape.int, rate = onearm.gamma.rate.int))
```



Parameterisations as per flexsurv


Each point is a bootstrap sample
Mean (gamma) OS days
1750
1500
1250
1000
750
150
175
200
225
250
Mean (weibull) PFS days
Bootstrap
correlated
uncorrelated

Conclusions

- R is definitely able to perform HTA relevant analysis
- In short term R is replacing SAS quicker than Excel in our workflow
- Moving computation from Excel into R could be a bridging step (as is currently done for NMA analysis by limiting Excel computation to sampling from parameter distributions)
- Open for collaboration to build tools with industry and non-industry partners

First name . Last name @roche.com

Doing now what patients need next