A software tool for the design and analysis of small population group trials

Diane Uschner      David Schindler      Ralf-Dieter Hilgers

September 24th, 2015
Motivation

- **Situation**: Randomized controlled clinical trial, two treatment arms.
- **Aim**: Estimate the treatment effect.
- Randomization is used in order to
  - balance (unknown) prognostic factors
  - to control selection bias (in combination with blinding)
- **But**: No sound scientific basis exists for choosing a randomization procedure
- Incorporate the constraints that peril the estimation of the treatment effect in the design stage.
- We present a software tool that provides a basis for choosing a randomization procedure.
This research is part of the IDeAl project which has received funding from the European Union’s Seventh Framework Programme for research, technological development and demonstration under Grant Agreement no 602552. Please visit

http://www.ideal.rwth-aachen.de/

to get the latest news.
There exist more than 7,600 rare diseases worldwide and the European Society of Paediatric Oncology stated that 75% of rare diseases affect children and 30% of rare disease patients die before the age of five.

Usual statistical methods for proving efficacy and safety of therapies fail to provide cost-efficient and reliable results in small populations.

There is a pressing need to integrate a broad range of innovative methodologies improving clinical trials in the setting of small sample population groups (SPG).

The objective of this research is to produce methods of general applicability irrespective of indication by Integrated DEsign and ANalysis of clinical trials in SPG (DEAN) through a multidisciplinary closely collaborating consortium of researchers from European universities, research institutes and industry.

The consortium will work in 11 WPs, focussed on assessment of randomization procedures, extrapolating dose-response information, investigation of adaptive designs, optimal designs in mixed models, pharmacogenetic designs, simulation of clinical trials, genetic factors influencing the response, decision analysis and biomarker surrogate endpoints as well as WPs on project management and dissemination of results. Relevant stakeholder concerns (patient needs, regulatory issues, reimbursement, clinical feasibility) will be monitored by a Clinical Scientific Advisory Board. Because of its integrative structure, this research program extends previous approaches, which focus on a certain methodology only. In its totality, the WPs constitute a logically coherent set of methodologies that is of sufficient breadth to tackle the multidisciplinary challenges. By combining, enhancing and developing different statistical methodologies and assessment methods, this research program will impact the scientific discussion in promoting efficient statistical methodology for clinical trials in SPG, also in view of existing regulatory guidance in the EU.
DOCUMENTS FOR DOWNLOAD

In this section, you can download the files that have been produced as official output in the project.

- Articles in peer-reviewed journals
- Presentations
- Input to Regulatory Documents
- Short Courses
- Conference Posters
- Comments
- Conferences
- Statistical Software Programs

- R package for testing similarity of dose response curves [published 2015-09-16]
- R package on Randomization for clinical trials [published 2015-09-09]
- R package "MIXFIM" for the evaluation and optimization of the Fisher Information Matrix in NonLinear Mixed Effect Models using Markov Chains Monte Carlo for both discrete and continuous data [published 2015-09-01]
- R package for the estimation of within subject correlations based on linear mixed effects models [published 2015-04-29]
- R package on the Prediction of Therapeutic Success [published 2015-04-27]
- R package for dimensionality reduction via variables clustering [published 2014-12-06]
- R-Code to calculate worst case type I error inflation in multiarmed clinical trials [published 2014-04-23]
- R package on Surrogate Markers [published 2014-03-18]
How to get randomizeR on my computer? - it’s easy!

install.packages("randomizeR")
library("randomizeR")

Schindler and Uschner (2015)
Structure of the package

issues

randomization procedures

randomization sequences

assessment/comparison

gensEq
getAllSeq
Let \( N \) be the total sample size.

**Definition**

A *randomization procedure* \( \mathcal{M} \) is a probability distribution on \( \Gamma = \{0, 1\}^N \). \( \mathcal{M} \) produces the sequences

\[
\Gamma_\mathcal{M} = \{ T \in \Gamma \mid \mathbb{P}_\mathcal{M}(T) \neq 0 \}
\]

A *randomization sequence* is a vector \( t \in \Gamma \) with

\[
t_i = \begin{cases} 
0 & \text{if subject } i \text{ is assigned to the control group,} \\
1 & \text{if subject } i \text{ is assigned to the experimental group.}
\end{cases}
\]

Rosenberger and Lachin (2016)
Several randomization procedures exist.

Formal definition:

(a) Random allocation rule:
\[ \mathbb{P}_{RAR}(t) = \binom{N}{N/2}^{-1}, t \in \Gamma_{RAR} \]

(b) Permutated block randomization:
\[ \mathbb{P}_{PBR}(t) = \binom{k}{k/2}^{-N/k}, t \in \Gamma_{PBR} \]

(c) Maximal procedure:
\[ \mathbb{P}_{MP}(t) = |\Gamma_{MP(N,b)}|^{-1}, t \in \Gamma_{MP(N,b)} \]

(d) Big Stick design:
\[ \mathbb{P}_{BSD}(t) = 0.5^{N-da(t)}, t \in \Gamma_{BSD}, da(t) = \text{number of deterministic allocations of } t \]
Several randomization procedures exist.

Sampling algorithms model the probability \( \Rightarrow \) sampled relative frequency is equal to theoretical probability.

Representation in randomeR:
(a) Random allocation rule: \( \text{rarPar}(N) \)
(b) Permutated block randomization: \( \text{pbrPar}(bc) \)
(c) Maximal procedure: \( \text{mpPar}(N,b) \)
(d) Big Stick design: \( \text{bsdPar}(N,b) \)
Randomization procedures implemented in randomizeR

- Complete Randomization
- Random Allocation Rule
- Permutated Block Randomization
- Permutated Block Randomization with random block lengths
- Truncated Binomial Design (in blocks)
- Truncated Binomial Design with random block lengths
- Efron’s Biased Coin Design
- Big Stick Design
- Maximal Procedure
- Hadamard Randomization
- Wei’s Urn Design

Rosenberger and Lachin (2016)
Representation of randomization procedures

\[ \text{bsdPar}(8,2, \text{groups}= \text{c}("E","C")) \]

Object of class "bsdPar"

- design = BSD(2)
- mti = 2
- N = 8
- groups = E C
Generate randomization sequences

```r
bsd <- bsdPar(8,2, groups= c("E","C"))
genSeq(bsd)
```

Object of class "rBsdSeq"

```r
design = BSD(2)
seed = 1425574311
N = 8
groups = E C
mti = 2

The sequence M:

1 E E C E C C E C
```
Generate randomization sequences

Object of class "rBsdSeq"

design = BSD(2)
seed = 1425574311
N = 8
groups = E C
mti = 2

The first 3 of 4 sequences of M:

1 E E C E C C E C
2 C E E C C C E E
3 C E E C E C C E
...

genSeq(bsd, 4)
Access the randomization sequences

getRandList(genSeq(bsd, 4))

```
[1,] "E" "E" "C" "E" "C" "C" "E" "C"
[2,] "C" "E" "E" "C" "C" "C" "E" "E"
[3,] "C" "C" "E" "E" "C" "E" "C" "E"
[4,] "C" "C" "C" "E" "C" "C" "E" "C"
```
Performance of **genSeq**

Performance of generating $10^x$ RAR sequences, $x \in \{3, 4, 5, 6\}$.

<table>
<thead>
<tr>
<th>System Time</th>
<th>genSeq(rarPar(100), $10^3$)</th>
<th>genSeq(rarPar(100), $10^5$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>0.08</td>
<td>6.88</td>
</tr>
<tr>
<td>System</td>
<td>0.00</td>
<td>0.06</td>
</tr>
<tr>
<td>Elapsed</td>
<td>0.08</td>
<td>7.06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System Time</th>
<th>genSeq(rarPar(100), $10^4$)</th>
<th>genSeq(rarPar(100), $10^6$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>0.70</td>
<td>67.85</td>
</tr>
<tr>
<td>System</td>
<td>0.03</td>
<td>0.44</td>
</tr>
<tr>
<td>Elapsed</td>
<td>0.73</td>
<td>68.79</td>
</tr>
</tbody>
</table>
randomizeR supports several functions for randomization procedures:

- `createParam()`: Creates a `<.>Par` object according to user input.
- `createSeq()`: Generates a random sequence according to user input.
- `getAllSeq(myPar)`: Compute $\Gamma_M$ for $N \leq 24$.
- `getProb(seqs)`: Compute the theoretical probabilities for an object `seqs` of type `randSeq`.
- `saveRand(seqs)`: Save the randomization protocol including the randomization sequence(s) to `.csv`.
Choose a **suitable** randomization procedure!

- Provide sound scientific basis for the selection of a tailored randomization procedure.
- Assumptions concerning the distribution of the responses may not be fulfilled.
- Biases may compromise the estimation of the treatment effect.
  - Selection bias
  - Chronological bias
  - Imbalances

**Process**

1. **Identify** constraints that impact the validity of the trial.
2. Define issues that **measure** the constraint.
3. **Assess** randomization procedures according to the issues.
4. **Select** appropriate randomization procedure on the basis of the assessment.
Choose a **suitable** randomization procedure!

- Provide sound scientific basis for the selection of a tailored randomization procedure.
- Assumptions concerning the distribution of the responses may not be fulfilled.
- Biases may compromise the estimation of the treatment effect.
  - Selection bias
  - Chronological bias
  - Imbalances

**Process**

1. **Identify** constraints that impact the validity of the trial.
2. Define issues that **measure** the constraint.
3. **Assess** randomization procedures according to the issues.
4. **Select** appropriate randomization procedure on the basis of the assessment.
randomizeR implements several issues.

Simple issues:

- **corGuess**: Represent the proportion of correct guesses (i.e. predictability).
- **imbal**: Represent the imbalance in allocation numbers (i.e. balance).

Sophisticated issues:

- **selBias**: Represent exact rejection probability (size/ power) in case the responses are influenced by third order selection bias.
- **chronBias**: Represent exact rejection probability (size/ power) in case the responses are influenced by chronological bias.
  - Tamm and Hilgers (2014), Rosenkranz (2011)
- **setPower**: Represent the power for a given detectable effect and size.
  - Lachin (1988)
Selection bias under convergence strategy

Third order selection bias

- Trial is randomized.
- Allocation list is concealed.
- But: the investigator can guess the next treatment assignment due to
  - unmasking of past assignments (due to side effects).
  - restrictions of the randomization procedure.
- Investigator can deny enrollment due to soft inclusion criteria.

Berger (2005)
Selection bias under convergence strategy

Convergence strategy

In the situation of third order selection bias, the investigator can count $N_E(i)$ and $N_C(i)$ respectively. ⇒ Under the assumption that $N_E(N) = N_C(N)$ for all $t \in \Gamma$ it is opportune for the investigator to guess

$$g(t_{i+1}) = \begin{cases} 
0 & N_E(i) > N_C(i) \\
\text{Ber}(0.5) & N_E(i) = N_C(i) \\
1 & N_E(i) < N_C(i),
\end{cases}$$

Blackwell and Hodges Jr. (1957)
Correct guesses - a measure for selection bias

Expected number of correct guesses

The expected number of correct guesses of a randomization sequence $t$ is given by

$$CG(t) := \mathbb{E}(\#\{i = 1, \ldots, N : g(t_{i+1}) = t_{i+1}\}).$$

The expected proportion of correct guesses of $t$ is

$$propCG(t) := \frac{CG(t)}{N}.$$
Use all the information! Take the **sequence based approach**!

- Model selection bias, e.g. as the proportion of correct guesses.
- For each randomization sequence $t$, compute the proportion of correct guesses.
- Compute descriptive measures, s.a. mean and quantiles, weighting the proportion of correct guesses of each sequence with its probability of occurrence.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Probability</th>
<th>propCG(CS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CCEE</td>
<td>0.1666667</td>
</tr>
<tr>
<td>2</td>
<td>CECE</td>
<td>0.1666667</td>
</tr>
<tr>
<td>3</td>
<td>ECCE</td>
<td>0.1666667</td>
</tr>
<tr>
<td>4</td>
<td>CEEC</td>
<td>0.1666667</td>
</tr>
<tr>
<td>5</td>
<td>ECEC</td>
<td>0.1666667</td>
</tr>
<tr>
<td>6</td>
<td>EECC</td>
<td>0.1666667</td>
</tr>
</tbody>
</table>

mean = 0.71
With randomizeR this is easy.

First, represent the issue of correct guesses:

```r
cg <- corGuess(type = "CS")
```

Then assess the sequences:

```r
rarS <- genSeq(rarPar(6), 10)
assess(rarS, cg)
```

Object of class "assessment"

design = RAR
N = 6
K = 2
groups = A B

The first 3 rows of 10 rows of D:

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Relative_Frequency</th>
<th>propCG(CS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAABAB</td>
<td>0.1</td>
<td>0.75000000</td>
</tr>
<tr>
<td>BABBAA</td>
<td>0.1</td>
<td>0.66666667</td>
</tr>
<tr>
<td>AAABBB</td>
<td>0.1</td>
<td>0.58333333</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Putting things together: compare

- Compare a number of randomization procedures according to one issue.
First, set the parameters:

```r
bsd <- genSeq(bsdPar(100,5),1000)
mp <- genSeq(mpPar(100,5),1000)
rar <- genSeq(rarPar(100),1000)
pbr <- genSeq(pbrPar(rep(10,10)),1000)

cg <- corGuess(type = "CS")
```

Next, compare the randomization procedures according to selection bias:

```r
compare(cg, bsd, mp, rar, pbr)
```

Comparison for propCG(CS)

<table>
<thead>
<tr>
<th></th>
<th>BSD.5</th>
<th>MP.5</th>
<th>RAR</th>
<th>PBR.10</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>0.546</td>
<td>0.590</td>
<td>0.558</td>
<td>0.652</td>
</tr>
<tr>
<td>sd</td>
<td>0.026</td>
<td>0.025</td>
<td>0.028</td>
<td>0.019</td>
</tr>
<tr>
<td>max</td>
<td>0.645</td>
<td>0.665</td>
<td>0.675</td>
<td>0.715</td>
</tr>
<tr>
<td>min</td>
<td>0.485</td>
<td>0.520</td>
<td>0.505</td>
<td>0.595</td>
</tr>
<tr>
<td>x05</td>
<td>0.505</td>
<td>0.550</td>
<td>0.515</td>
<td>0.620</td>
</tr>
<tr>
<td>x25</td>
<td>0.525</td>
<td>0.575</td>
<td>0.535</td>
<td>0.640</td>
</tr>
<tr>
<td>x50</td>
<td>0.545</td>
<td>0.590</td>
<td>0.555</td>
<td>0.650</td>
</tr>
<tr>
<td>x75</td>
<td>0.565</td>
<td>0.605</td>
<td>0.575</td>
<td>0.665</td>
</tr>
<tr>
<td>x95</td>
<td>0.595</td>
<td>0.635</td>
<td>0.610</td>
<td>0.685</td>
</tr>
</tbody>
</table>
Finally, visualize the comparison with

\[
C \leftarrow \text{compare}(cg, bsd, mp, rar, pbr)
\]

\[
\text{plot}(C)
\]
And it’s fast, too!

Input:

```r
bsd = genSeq(bsdPar(100,5),1000)
mp = genSeq(mpPar(100,5),1000)
rar = genSeq(rarPar(100),1000)
pbr = genSeq(pbrPar(rep(10,10)),1000)
cg <- corGuess(type = "CS")

comp <- compare(cg, bsd, mp, rar, pbr)
```

Performance [sec]:

<table>
<thead>
<tr>
<th>test</th>
<th>elapsed</th>
<th>replications</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>CG</td>
<td>100</td>
<td>0.0002</td>
</tr>
<tr>
<td>3</td>
<td>RAR</td>
<td>100</td>
<td>0.0615</td>
</tr>
<tr>
<td>4</td>
<td>PBR</td>
<td>100</td>
<td>0.1565</td>
</tr>
<tr>
<td>2</td>
<td>MP</td>
<td>100</td>
<td>0.6785</td>
</tr>
<tr>
<td>1</td>
<td>BSD</td>
<td>100</td>
<td>0.6856</td>
</tr>
<tr>
<td>6</td>
<td>COMP</td>
<td>100</td>
<td>1.5908</td>
</tr>
</tbody>
</table>
randomizeR

- establishes an open source, easily extendable, strongly structured framework for randomization.
- incorporates various randomization procedures and issues.
- provides a sound scientific basis for the assessment of randomization procedures according to pre-specified criteria.
- enables the user to chose a tailored randomization procedure.
- will lead to more reliable treatment effect estimates.
randomizer

- establishes an open source, easily extendable, strongly structured framework for randomization.
- incorporates various randomization procedures and issues.
- provides a sound scientific basis for the assessment of randomization procedures according to pre-specified criteria.
- enables the user to choose a tailored randomization procedure.
- will lead to more reliable treatment effect estimates.

Next
Propose a unified criterion that comprises several issues

⇒ David Schindler: “Selecting an appropriate randomization procedure for a small population group trial on the basis of a linked optimization criterion”.

FP7 HEALTH 2013 - 602552


Hilgers, R.-D. et al (2016). A general framework to assess randomization procedures under selection and chronological bias with respect to type I error probability. *(in preparation)*


References II


