Patients assignments to specTra and SWT groups through minimization

September 29, 2020

Author: Pauline Gut

Definitions [1–4]

- Minimization is a method of ensuring excellent balance between groups for several prognostic factors, even in small samples.
- Minimization is a widely acceptable approach.
- Minimization is a form of restricted randomization.
- Minimization is an **adaptive randomization** type: the treatment allocated to the next participant enrolled in the trial depends (wholly or partly) on the characteristics of those participants already enrolled.
- Minimization is a dynamic method since the randomization list is not produced before the trial starts, but during participant recruitment.
- Minimization balances treatment assignments across multiple prognostic factors simultaneously. The aim is that each allocation should minimize the imbalance across multiple factors.
- Minimization has been first proposed by *Taves* [1] and independently generalized by *Pocock and Simon* [2].

Notations

The following is described in [2].

- N different treatments to assign participants to.
- M prognostic factors that we want to balance.
- Each factor has n_i levels $(n_1, \cdots, n_M)$.
- At any point during the trial, x_{ijk} is the number of patients who have been assigned to treatment k with level j of factor i for $j = 1, 2, \cdots, n_i$; $i = 1, 2, \cdots, M$ and $k = 1, \cdots, N$.

Consider a new patient entering the trial. Let r_1, ..., r_M be the levels of factors $1, \cdots, M$ for this patient.

For each treatment k one considers the new x_{ijl}, denoted by x^k_{ijl}, that would arise if that treatment were assigned to the patient, where the superscript k refers to the particular treatment under consideration:

$$x^k_{ir_ik} = x_{ijk} + 1$$
$$x^k_{ijl} = x_{ijl} \text{ for } l \neq k, j \neq r_i$$

i.e. we add 1 to all the counts x_{ijk} that match the treatment k and the factor levels the participant has, and all the other counts are unchanged.

1. **Imbalance within each factor**

   We start by calculating the imbalance within each factor.

   Let $D(\{z\}_{l=1}^N)$ be some function which measures the ***amount of variation*** in any set of non-negative integers $\{z\}_{l=1}^N$.

   Then, if treatment k were assigned to the new patient, we can get the resulting amount of variation that would be produced by assigning the patient to treatment k, and calculate

   $$d_{ik} = D(\{x^k_{ir_il}\}_{l=1}^N)$$

   the resultant *lack of balance* or *imbalance* in treatment numbers across all treatments, for all patients with level r_i of factor i.

   Note that only patients who match the new patient's factor level r_i matter for the calculation of imbalance for factor i, participants with other factor levels don't affect the imbalance calculation.

   We can choose different functions for D, see below.

2. **Total imbalance**

   Once we have the imbalance within each factor, we combine them to produce a total imbalance score.

   Let G be some function from $\mathbf{R}^M \to \mathbf{R}$ that combines the d_{ik} for all the M factors:

   $$G_k = G(d_{1k}, \cdots, d_{Mk}).$$

   A choice for G can just the sum.

   Then, G_k represents the ***total amount of imbalance*** in treatment numbers which would exist at all the factor levels of the new patient if treatment k were assigned to that patient. In other words, G_k is the total imbalance we would have if the new patient is assigned to treatment k. We calculate G_k for all the N treatments.

3. **Assigning probabilities**

   We can then rank the treatments according to their values G_k (ascending ranks with increasing G_k values corresponding to larger amounts of imbalance), treatment (1) having minimal G_k, treatment (2) having the next smallest G_k, etc., so that (s) ¡ (t) iff $G_{(s)} < G_{(t)}$. In the case of ties, a random ordering can be determined.

   The assigned treatment T can be determined from the following set of probabilities:

   $$\text{prob}(T = (k)) = p_k \quad \text{where} \quad p_1 \geq p_2 \geq \cdots \geq p_N \quad \text{and} \quad \sum p_k = 1.$$

   The values of p_k can be fixed constants or functions of G_k. This ordering of probabilities means that treatments with small values of G_k have a higher probability of being chosen.

   The entire procedure is repeated when the next patient enters the trial.

4. **The choice of D**

   Four possible formulae for D are considered in [4]. Here we present the one we are using:

(i) *The Range distance measure* of $\{z\}_l$ is defined to be the absolute value of the difference between each $\{z\}_{l=1}^{N}$.

If we are interested in comparing pairs of treatments in analysis it may be preferable since D would then be measuring the most imbalance in any pair. Also, with only two treatments, the range ($= |z_1 - z_2|$) is equivalent to the standard deviation.

5. **The choice of G**

One reasonable way of combining $\{d_{ik}\}_{i=1}^{M}$ is to sum then. That is,

$$G_k = G(d_{ik}, \cdots, d_{Mk}) = \sum_{i=1}^{M} d_{ik}.$$

In the case where some prognostic factors are considered more important than others. One can then make G_k a weighted sum of $\{d_{ik}\}$ so that

$$G_k = G(d_{ik}, \cdots, d_{Mk}) = \sum_{i=1}^{M} w_i \; d_{ik}$$

where $\{w_i\}$ are constants.

6. **The choice** $\{p_k\}$

The formula for $\{p_k\}$ determines the extent to which one wishes to bias treatment assignment in favor of those treatments with small G_k. Different types of formula are suggested in [4].

Summary:
Minimization is a dynamic algorithm involving three process steps [2]:

1. First, an **imbalance score** is computed for each available treatment based on all previous allocations, as well as the hypothetical allocation of the current patient to each treatment.
2. In the second step, the **preferred treatment** is selected by choosing the treatment allocation associated with the **smallest imbalance score** after the hypothetical allocation — the remaining treatments are considered non-preferred. If several treatments are tied with respect to imbalance score, the tie can be broken by selecting the preferred treatment at random.
3. Finally, in the third step, **allocation probabilities** are computed for the treatments and the patient is randomized accordingly.

References

[1] D. R. Taves. Minimization: A new method of assigning patients to treatment and control groups. *CLINICAL PHARMACOLOGY and THERAPEUTICS*, 15(5), 1974. ISSN 01406736. doi: 10.1016/S0140-6736(64)90114-X.

[2] Stuart J. Pocock and Richard Simon. Sequential Treatment Assignment with Balancing for Prognostic Factors in the Controlled Clinical Trial. *Biometrics*, 31(1):103, 1975. ISSN 0006341X. doi: 10.2307/2529712.

[3] Douglas G. Altman and J. Martin Bland. Treatment allocation by minimisation. *BMJ*, 330(7495): 843, 2005. ISSN 14685833. doi: 10.1136/bmj.330.7495.843.

[4] Neil W. Scott, Gladys C. McPherson, Craig R. Ramsay, and Marion K. Campbell. The method of minimization for allocation to clinical trials: A review. *Controlled Clinical Trials*, 23(6):662–674, 2002. ISSN 01972456. doi: 10.1016/S0197-2456(02)00242-8.

Minimization algorithm

```
clear; close all; clc;

% Choose the excel file containing the previous allocated patients to the trial
% Each row is a patient
% The first column corresponds to the patient ID
% Each other column corresponds to a variate to be balanced over the treatment groups
disp('Choose the excel file containing the previous allocated patients to the trial.')
disp('Each row must be a patient. The first column corresponds to the patient ID.')
disp('Each other column corresponds to a variate to be balanced over the treatment groups.')
[Filename,Pathname]=uigetfile('*.xlsx','Allocated patients to the trial');
cd(Pathname)
Allocated_Patients = readtable(Filename);


% Enter the number of patients you want to randomize with simple
% randomization
n = size(Allocated_Patients,1);
clc;
New_Patient.ID = input('New patient ID: ', 's');
New_Patient.SixMWT = input('6MWT of the new patient (in meters): ');
New_Patient.Age = input('Age of the new patient: ');

if n == 0
p = size(Allocated_Patients,1) + 1;
PatientID{1,1} = New_Patient.ID;
Patient6MWT(1,1) = New_Patient.SixMWT;
PatientAge(1,1) = New_Patient.Age;
Allocated_Patients(p,1) = table(PatientID(1,1));
Allocated_Patients(p,2) = table(Patient6MWT(1,1));
Allocated_Patients(p,3) = table(PatientAge(1,1));

% Treatment allocation to the patient through simple randomization
Treat = randi(2,1,1);
SWT{1,1} = 'SWT';
DOT{1,1} = 'DOT';
  if Treat == 1
    Allocated_Patients(p,4) = DOT;
    disp('Patient allocated to DOT')
  else
    Allocated_Patients(p,4) = SWT;
    disp('Patient allocated to SWT')
  end
```

```
Allocated_Patients.Properties.VariableNames =  {'PatientID', 'x6MWT', 'Age', 'Treatment'};
writetable(Allocated_Patients, Filename, 'WriteVariableNames', true)
disp('Finished!')

else
% Treatment allocation to the patient through minimization
% Calculate the mean 6MWT and the mean age of the patients previsouly
% allocated to the trial
Mean_SixMWT = mean(cell2mat(table2cell(Allocated_Patients(:,2))));
Mean_Age = mean(cell2mat(table2cell(Allocated_Patients(:,3))));

% Computation of the amount of variation among treatments
Smaller_Mean_SixMWT = 0;
Equal_Mean_SixMWT = 0;
Larger_Mean_SixMWT = 0;
Smaller_Mean_Age = 0;
Equal_Mean_Age = 0;
Larger_Mean_Age = 0;

for i = 1:size(Allocated_Patients,1)
if strcmp(table2cell(Allocated_Patients(i,4)), 'DOT')
   Smaller_Mean_SixMWT = Smaller_Mean_SixMWT + sum(cell2mat(table2cell(Allocated_Patients(i,2)))
< Mean_SixMWT);
   Equal_Mean_SixMWT = Equal_Mean_SixMWT + sum(cell2mat(table2cell(Allocated_Patients(i,2))) ==
Mean_SixMWT);
   Larger_Mean_SixMWT = Larger_Mean_SixMWT + sum(cell2mat(table2cell(Allocated_Patients(i,2))) >
Mean_SixMWT);

   Smaller_Mean_Age = Smaller_Mean_Age + sum(cell2mat(table2cell(Allocated_Patients(i,3))) <
Mean_Age);
   Equal_Mean_Age = Equal_Mean_Age + sum(cell2mat(table2cell(Allocated_Patients(i,3))) ==
Mean_Age);
   Larger_Mean_Age = Larger_Mean_Age + sum(cell2mat(table2cell(Allocated_Patients(i,3))) >
Mean_Age);
   T(1,:) = table(Smaller_Mean_SixMWT, Equal_Mean_SixMWT, Larger_Mean_SixMWT,
Smaller_Mean_Age, Equal_Mean_Age, Larger_Mean_Age);

end
end

clear Smaller_Mean_SixMWT Equal_Mean_SixMWT Larger_Mean_SixMWT Smaller_Mean_Age
Equal_Mean_Age Larger_Mean_Age
```

```
Smaller_Mean_SixMWT = 0;
Equal_Mean_SixMWT = 0;
Larger_Mean_SixMWT = 0;
Smaller_Mean_Age = 0;
Equal_Mean_Age = 0;
Larger_Mean_Age = 0;

for i = 1:size(Allocated_Patients,1)
if strcmp(table2cell(Allocated_Patients(i,4)), 'SWT')
   Smaller_Mean_SixMWT = Smaller_Mean_SixMWT + sum(cell2mat(table2cell(Allocated_Patients(i,2)))
< Mean_SixMWT);
   Equal_Mean_SixMWT = Equal_Mean_SixMWT + sum(cell2mat(table2cell(Allocated_Patients(i,2))) ==
Mean_SixMWT);
   Larger_Mean_SixMWT = Larger_Mean_SixMWT + sum(cell2mat(table2cell(Allocated_Patients(i,2))) >
Mean_SixMWT);

   Smaller_Mean_Age = Smaller_Mean_Age + sum(cell2mat(table2cell(Allocated_Patients(i,3))) <
Mean_Age);
   Equal_Mean_Age = Equal_Mean_Age + sum(cell2mat(table2cell(Allocated_Patients(i,3))) ==
Mean_Age);
   Larger_Mean_Age = Larger_Mean_Age + sum(cell2mat(table2cell(Allocated_Patients(i,3))) >
Mean_Age);
   T(2,:) = table(Smaller_Mean_SixMWT, Equal_Mean_SixMWT, Larger_Mean_SixMWT,
Smaller_Mean_Age, Equal_Mean_Age, Larger_Mean_Age);

end
end

clear Smaller_Mean_SixMWT Equal_Mean_SixMWT Larger_Mean_SixMWT Smaller_Mean_Age
Equal_Mean_Age Larger_Mean_Age



if New_Patient.SixMWT < Mean_SixMWT
   T(3,1) = cell2table(num2cell(1));
   T(4,1) = cell2table(num2cell(sum(T{[1 3],1}, 1)));
   T(5,1) = T(2,1);
   T(6,1) = cell2table(num2cell(abs(diff(T{[4 5],1}, 1))));
   T(7,1) = T(1,1);
   T(8,1) = cell2table(num2cell(sum(T{[2 3],1}, 1)));
   T(9,1) = cell2table(num2cell(abs(diff(T{[7 8], 1}, 1))));

elseif New_Patient.SixMWT == Mean_SixMWT
   T(3,2) = cell2table(num2cell(1));
```

```
    T(4,2) = cell2table(num2cell(sum(T{[1 3],2}, 1)));
    T(5,2) = T(2,2);
    T(6,2) = cell2table(num2cell(abs(diff(T{[4 5],2}, 1))));
    T(7,2) = T(1,2);
    T(8,2) = cell2table(num2cell(sum(T{[2 3],2}, 1)));
    T(9,2) = cell2table(num2cell(abs(diff(T{[7 8], 2}, 1))));

elseif New_Patient.SixMWT > Mean_SixMWT
    T(3,3) = cell2table(num2cell(1));
    T(4,3) = cell2table(num2cell(sum(T{[1 3],3}, 1)));
    T(5,3) = T(2,3);
    T(6,3) = cell2table(num2cell(abs(diff(T{[4 5],3}, 1))));
    T(7,3) = T(1,3);
    T(8,3) = cell2table(num2cell(sum(T{[2 3],3}, 1)));
    T(9,3) = cell2table(num2cell(abs(diff(T{[7 8], 3}, 1))));
end

if New_Patient.Age < Mean_Age
    T(3,4) = cell2table(num2cell(1));
    T(4,4) = cell2table(num2cell(sum(T{[1 3],4}, 1)));
    T(5,4) = T(2,4);
    T(6,4) = cell2table(num2cell(abs(diff(T{[4 5],4}, 1))));
    T(7,4) = T(1,4);
    T(8,4) = cell2table(num2cell(sum(T{[2 3],4}, 1)));
    T(9,4) = cell2table(num2cell(abs(diff(T{[7 8], 4}, 1))));

elseif New_Patient.Age == Mean_Age
    T(3,5) = cell2table(num2cell(1));
    T(4,5) = cell2table(num2cell(sum(T{[1 3],5}, 1)));
    T(5,5) = T(2,5);
    T(6,5) = cell2table(num2cell(abs(diff(T{[4 5],5}, 1))));
    T(7,5) = T(1,5);
    T(8,5) = cell2table(num2cell(sum(T{[2 3],5}, 1)));
    T(9,5) = cell2table(num2cell(abs(diff(T{[7 8], 5}, 1))));

elseif New_Patient.Age > Mean_Age
    T(3,6) = cell2table(num2cell(1));
    T(4,6) = cell2table(num2cell(sum(T{[1 3],6}, 1)));
    T(5,6) = T(2,6);
    T(6,5) = cell2table(num2cell(abs(diff(T{[4 5],6}, 1))));
    T(7,6) = T(1,6);
    T(8,6) = cell2table(num2cell(sum(T{[2 3],6}, 1)));
    T(9,6) = cell2table(num2cell(abs(diff(T{[7 8], 6}, 1))));
end
```

```
T(6,7) = cell2table(num2cell(sum(T{6,:}, 2)));
T(9,7) = cell2table(num2cell(sum(T{9,:}, 2)));


T.Properties.RowNames =  {'DOT group', 'SWT group', 'New patient', 'DOT + new patient', 'SWT',
'Absolute difference 1', 'DOT', 'SWT + new patient', 'Absolute difference 2'};

% Treatment allocation to the patient
p = size(Allocated_Patients,1) + 1;
PatientID{1,1} = New_Patient.ID;
Patient6MWT(1,1) = New_Patient.SixMWT;
PatientAge(1,1) = New_Patient.Age;
Allocated_Patients(p,1) = table(PatientID(1,1));
Allocated_Patients(p,2) = table(Patient6MWT(1,1));
Allocated_Patients(p,3) = table(PatientAge(1,1));

SWT{1,1} = 'SWT';
DOT{1,1} = 'DOT';

if cell2mat(table2cell(T(6,7))) > cell2mat(table2cell(T(9,7)))
   Allocated_Patients(p,4) = SWT;
   disp('Patient allocated to SWT')

elseif cell2mat(table2cell(T(6,7))) == cell2mat(table2cell(T(9,7)))
   disp('Simple randomization')
   Treat = randi(2,1,1);
   if Treat == 1
      Allocated_Patients(p,4) = DOT;
      disp('Patient allocated to DOT')
   else
      Allocated_Patients(p,4) = SWT;
      disp('Patient allocated to SWT')
   end

elseif cell2mat(table2cell(T(6,7))) < cell2mat(table2cell(T(9,7)))
   Allocated_Patients(p,4) = DOT;
   disp('Patient allocated to DOT')
end


S{1,1} = 'Number of patients in DOT: ';
S{2,1} = 'Number of patients in SWT: ';
S{3,1} = 'Mean 6MWT in DOT: ';
```

```
S{4,1} = 'Mean 6MWT in SWT: ';
S{5,1} = 'Mean age in DOT: ';
S{6,1} = 'Mean age in SWT: ';

S{1,2} = sum(strcmp(table2cell(Allocated_Patients(:,4)), 'DOT'),1);
S{2,2} = sum(strcmp(table2cell(Allocated_Patients(:,4)), 'SWT'),1);

for i = 1:size(Allocated_Patients,1)
  if strcmp(table2cell(Allocated_Patients(i,4)), 'DOT')
    Mean_DOT1(i,1) = cell2mat(table2cell(Allocated_Patients(i,2)));
    Mean_DOT2(i,1) = cell2mat(table2cell(Allocated_Patients(i,3)));
  elseif strcmp(table2cell(Allocated_Patients(i,4)), 'SWT')
    Mean_SWT1(i,1) = cell2mat(table2cell(Allocated_Patients(i,2)));
    Mean_SWT2(i,1) = cell2mat(table2cell(Allocated_Patients(i,3)));
  end
end

for i = 1:size(Allocated_Patients,1)
  if ~strcmp(table2cell(Allocated_Patients(i,4)), 'DOT')
    Mean_DOT1(i,1) = 0;
    Mean_DOT2(i,1) = 0;
  elseif ~strcmp(table2cell(Allocated_Patients(i,4)), 'SWT')
    Mean_SWT1(i,1) = 0;
    Mean_SWT2(i,1) = 0;
  end
end

Mean_SixMWT_DOT = mean(Mean_DOT1);
Mean_Age_DOT = mean(Mean_DOT2);
Mean_SixMWT_SWT = mean(Mean_SWT1);
Mean_Age_SWT = mean(Mean_SWT2);

S{3,2} = round(Mean_SixMWT_DOT);
S{4,2} = round(Mean_SixMWT_SWT);
S{5,2} = round(Mean_Age_DOT);
S{6,2} = round(Mean_Age_SWT);

Population_Stat = table(S);
T.Properties.VariableNames([7]) = {'Sum'};

writetable(Allocated_Patients, Filename, 'WriteVariableNames', true)
writetable(Population_Stat, 'TrialPopStat.xlsx', 'WriteVariableNames', false)

disp('Finished!')
```

```
%clearvars -except Allocated_Patients Population_Stat T

end


system(['set PATH=' Pathname ' && ' Filename]);
```