Published in final edited form as:
Comput Methods Programs Biomed. 2013 October ; 112(1): 211–218. doi:10.1016/j.cmpb.2013.05.020.

pocrm: An R-package for Phase I trials of combinations of agents

Nolan A. Wages[*] and **Nikole Varhegyi**
Division of Translational Research and Applied Statistics, Department of Public Health Sciences University of Virginia, Charlottesville, VA 22908, USA

Abstract

This paper presents the R package **pocrm** for implementing and simulating the partial order continual reassessment method (PO-CRM; [1,2]) in Phase I trials of combinations of agents. The aim of this article is to illustrate, through examples of the **pocrm** package, how the PO-CRM works and how its operating characteristics can inform clinical trial investigators. This should promote the use of the PO-CRM in designing and conducting dose-finding Phase I trials of combinations of agents.

Keywords

Continual reassessment method; Dose finding; Phase I trials; Partial ordering; Drug combination

1. Introduction

Dose-finding trials of combinations of agents are becoming increasingly popular in cancer research. The majority of methods for the design of Phase I trials in oncology rely on the monotonicity assumption of the dose-toxicity curve. In this case, the curve is said to follow a "complete order" because the ordering of probabilities of dose-limiting toxicity (DLT) for any pair of doses is known and administration of greater doses of the agent can be expected to produce DLT's in increasing proportions of patients. In studies of combinations of agents, the dose-toxicity relationship may not be monotone for all combinations, in which case the toxicity probabilities follow a partial order. The continual reassessment method for partial orders (PO-CRM) [1,2] is a design for Phase I trials of combinations that leans upon specifying possible complete orders associated with the partial order.

In this article, we present a new R [3] package **pocrm**, which provides functions for implementation and simulation of the PO-CRM. The package allows freedom in many of its design specifications, and offers outputs designed for quick assessment of the design. The primary features of **pocrm** are as follows.

1. The flexibility to specify any initial escalation scheme. This includes flexibility in both the order in which the combinations are tried in the first stage, as well as the size of the patient cohorts.

[*]*Corresponding author at*: Public Health Sciences, P.O. Box 800717, Charlottesville, VA 22908, USA. Tel.: +1 434 924 8222. nwages@virginia.edu. .

6. Availability The package pocrm and the relevant documentation can be freely downloaded from CRAN webpage http://cran.r-project.org/web/packages=pocrm.

Conflict of interest Dr. Wages and Ms. Varhegyi report no biomedical financial interests or potential conflicts of interest.

2. The ability to exhaust a fixed sample size or to stop the trial once a prespecified number of patients have been allocated to particular combination.
3. The flexibility to change other design specifications such as the possible orderings of the toxicity probabilities, the initial guesses of the toxicity probabilities under each ordering, and the target toxicity rate.

The primary objective of this paper is to demonstrate the operating characteristics of **pocrm** through exploration of its various options via Phase I drug combination trial examples. In Section 2, we outline the models and computational methods of the POCRM. The functions of the package are described in Section 3, while examples are provided in Section 4. We conclude with some discussion and a statement regarding the availability of the package.

1.1. Specifying a partial ordering in R

A detailed discussion of partial ordering among doses in a Phase I trial of combinations can be found in Wages, Conaway and O'Quigley [1,2,4], as well as in Wages and Conaway [5]. As an example, consider a recently FDA approved, Phase I combination trial designed at the University of Virginia Cancer Center.

Example 1—This is a Phase I trial designed to determine the MTD of a combination of a toll-like receptor (TLR) agonists with or without a form of incomplete Freund's adjuvant (IFA) for the treatment of melanoma. TLR, denoted A, had 4 dose levels and IFA had three subgroups: 0 – IFA is not administered with any vaccine, V1 – IFA is administered with just the first vaccine, and V6 – IFA is administered in all vaccines. Therefore, there exist a total of 12 combinations under consideration, creating a 4 × 3 matrix order. The treatments are labeled as in Table 1.

A reasonable assumption to be made in these type of studies is that toxicity increase monotonically with dose of an agent, if the other agent is held fixed (i.e. across rows and up columns of the matrix in Table 1). If Agent A is fixed at level 2, then we can completely order combinations 4, 5, and 6 with respect to their toxicity, which for simplicity we express as 4–5–6. It may not be possible to arrive at a complete ordering among the combinations along the diagonals of the drug combination matrix. The conditions 1–2 and 1–4 hold without it being possible to order 2 and 4 with respect to one another. In drug combination matrices, Wages and Conaway [5] recommend formulating a subset of possible orderings by ordering the combinations by moving across rows, up columns, and up or down the diagonals. In doing so, the six orderings considered would be: (1) Across rows: 1-2-3-4-5-6-7-8-9-10-11-12, (2) Up columns: 1-4-7-10-2-5-8-11-3-6-9-12, (3) Up diagonals: 1-2-4-3-5-7-6-8-10-9-11-12, (4) Down diagonals: 1-4-2-7-5-3-10-8-6-11-9-12, (5) Alternating down-up diagonals: 1-2-4-7-5-3-6-8-10-11-9-12, (6) Alternating up-down diagonals: 1-4-2-3-5-7-10-8-6-9-11-12. As is discussed in Section 3, specifying the possible orderings of the toxicity probabilities will be one of the necessary arguments for using **pocrm**, and is done using the following code.

```
#Set 1 of possible orderings R> orders< -matrix(nrow=6,ncol=12)
R> orders[1,]< -c(1,2,3,4,5,6,7,8,9,10,11,12)
#across rows
R> orders[2,]< -c(1,4,7,10,2,5,8,11,3,6,9,12)
#up columns
R> orders[3,]< -c(1,2,4,3,5,7,6,8,10,9,11,12)
#up diagonals
R> orders[4,]< -c(1,4,2,7,5,3,10,8,6,11,9,12)
#down diagonals
```

```
R> orders[5,]< -c(1,2,4,7,5,3,6,8,10,11,9,12)
#down-up diagonals
R> orders[6,]< -c(1,4,2,3,5,7,10,8,6,9,11,12)
#up-down diagonals
```

Table 1 is simply one of several ways in which the combinations could be labeled. Another natural manner in which to do so is in increasing order moving from the lower left corner to the upper right corner, according to the diagonals. This makes sense in that combination 1 is the lowest dose of both agents (1, 1), combinations 2 and 3 is dose 1 of one agent and dose 2 of the other (1, 2) and (2, 1), etc. Labeling in such a way, Table 1 now takes on the appearance of Table 2 and the subset of possible orderings in R becomes

```
#Set 2 of possible orderings
R> orders< -matrix(nrow=6,ncol=12)
R> orders[1,]< -c(1,2,4,3,5,7,6,8,10,9,11,12)
#across rows
R> orders[2,]< -c(1,3,6,9,2,5,8,11,4,7,10,12)
#up columns
R> orders[3,]< -c(1,2,3,4,5,6,7,8,9,10,11,12)
#up diagonals
R> orders[4,]< -c(1,3,2,6,5,4,9,8,7,11,10,12)
#down diagonals
R> orders[5,]< -c(1,3,2,4,5,6,9,8,7,10,11,12)
#up-down diagonals
R> orders[6,]< -c(1,2,3,6,5,4,7,8,9,11,10,12)
#down-up diagonals
```

It is imperative in using **pocrm** correctly that the user maintains the monotonicity assumption across rows and up columns of the matrix of combinations. This can be done by adhering to the across row, up columns, up/down diagonal specification technique outlined by Wages and Conaway [5]. Further, it is important for the user to be consistent in specifying the possible orderings to be considered and the true toxicity probabilities, according to the manner in which he or she chooses to label the combinations in the matrix. For example consider the scenario of true toxicity probabilities displayed in Table 3. The specification of the vector of true toxicity probabilities, which we denote r, for use in the package would be dependent upon how the combinations are labeled. The treatment labeled 1 would need to go first, the treatment labeled 2 second, and so on. So for the labels in Table 1, the true toxicity probabilities would be

```
#Set 1 of true toxicity probabilities
R> r<
-c(0.03,0.06,0.12,0.08,0.14,0.20,0.16,0.22,
0.28,0.24,0.30,0.36).
```

And for the labels in Table 2, the vector would be:

```
#Set 2 of true toxicity probabilities
R> r<
-c(0.03,0.06,0.08,0.12,0.14,0.16,0.20,0.22,
0.24,0.28,0.30,0.36).
```

In other words, r in Set 1 is meant to be used in simulation with the first set of possible orders above and r in Set 2 is meant for the second set of orderings. One can always obtain the true ordering of the toxicity probabilities according to the chosen labels via the code order(r).These two distinct way of approaching a drug combination Phase I trial in R will generate identical operating characteristics.

2. Computational methods

The details of the two-stage POCRM can be found in Wages, Conaway and O'Quigley [2] and Wages and Conaway [5], so we only briefly recall them here. We begin by assuming that there are k drug combinations under investigation and that the DLT probabilities of the combinations follow a partial order for which there are M complete orders under consideration, specified as described in the previous section. For a particular ordering, m, we model the true probability of a DLT at combination i via

$$\pi_i = Pr\,(\mathrm{DLT\ \ at\ \ combination}\ i) \approx \alpha_{im}^{a}, \quad i=1,\ldots,k; m=1,\ldots,M$$

where $\{\alpha_{1m},\ldots,\alpha_{km}\}$ is the skeleton of the model under ordering m and $a = (0, \infty)$ is an unknown parameter. The **pocrm** package only accommodates this power model into its DLT probability estimation. There are certainly other CRM-type models (i.e. hyperbolic tangent, logistic) that could be implemented in the POCRM, but the power model is one of the most commonly used in CRM literature and has been demonstrated to work well in practice [6,7]. Prior information regarding the plausibility of each ordering is specified by a set of prior probabilities $p(m) = \{p(1), \ldots, p(M)\}$, where $p(m) \geq 0$ and $\sum_m p(m) = 1$.

After inclusion of the first j patients into the trial, if the data are to be analyzed under ordering m, we obtain an estimate, $\widehat{a}_{jm}$, based on the maximum likelihood estimate (MLE) of a Suppose, after j inclusions, the likelihood, $L_{jm}(a)$, is maximized at $\widehat{a}_{jm}$. Having taken some value of m, the dose to given to the $(j + 1)$th patient is determined. We weight each of the M candidate orderings as we make progress. The updated probability of each ordering is then given by

$$\omega_j\,(m) = \frac{\left\{L_{jm}\left(\widehat{a}_{jm}\right)\right\} p\,(m)}{\sum_{m=1}^{M} L_{jm}\left(\widehat{a}_{jm}\right)\Big\} p\,(m)}.$$

Wages et al. [1,2] suggest an escalation method that selects the ordering with the largest probability $\omega_j(m)$. Therefore, we choose a single ordering, m^*, such that $m^* = arg\max_m \omega_j\,(m)$. Given $m8$, we can generate estimates of the toxicity probabilities at each combination by calculation $\widehat{\pi}_i = \alpha_{im^*}^{\widehat{a}_{jm^*}}$. We then allocate the next entered patient cohort to the combination with an estimated DLT probability closest to the target; i.e. minimizes $|\widehat{\pi}_i - \theta|$ for some target toxicity rate, θ.

3. Package description

3.1. getwm function

The function getwm returns a matrix of skeleton values corresponding to the possible orderings of the toxicity probabilities using the following syntax: getwm(orders,skeleton). The required arguments are:

- orders – A matrix specifying the possible orderings of the combinations with regards to their toxicity probabilities.
- skeleton – A vector of values for the initial guesses of toxicity probabilities. The location of these values will be adjusted to correspond to the orderings specified by orders.

The output is composed of:

- alpha – Matrix of skeleton values corresponding to the possible orderings of the toxicity probabilities specified by orders.

3.2. pocrm.imp function

The function pocrm.imp is used to compute a combination recommendation for the next patient according to the partial order continual reassessment method (PO-CRM) using the syntax pocrm.imp(alpha,prior.o,theta,y,combos). The required arguments are:

- alpha – A matrix of initial guesses of the toxicity probabilities generated by getwm.
- prior.o – A vector of prior probabilities on the orderings.
- theta – The target DLT rate.
- y – A vector of patient outcomes; 1 indicates toxicity, 0 otherwise.
- combos – A vector of dose levels assigned to patients. The length of combos must be equal to y.

The output is composed of:

- order.est – Updated estimate of the ordering of toxicity probabilities.
- a.est – The estimate of the model parameter.
- ptox.est – Updated estimates of the toxicity probabilities.
- dose.rec – The combination recommended for the next patient cohort.

3.3. pocrm.sim function

The function pocrm.sim is used to simulate Phase I trials of combined drugs according to the partial order continual reassessment method (PO-CRM) using the following syntax pocrm.sim(r,alpha,prior.o,x0,stop,n,theta,nsim, tox.range). The required arguments are:

- r – A vector of true toxicity probabilities.
- alpha – A matrix of initial guesses of the toxicity probabilities generated by getwm.
- prior.o – A vector of prior probabilities on the orderings.
- x0 – A vector specifying an initial escalation scheme. It should be a sequence of combinations to which the trial would proceed prior to the observance of the first DLT and the patient cohort size. The length of x0 cannot exceed n.
- stop – The total number of patients treated on any combination required to stop the trial.
- n – The maximum sample size. If stop>n, then the trial will exhaust a pre-determined, fixed sample size of n patients.
- theta – The target DLT rate.
- nsim – The number of simulations.

- tox.range – A single numeric value used to define a range of "acceptable" DLT rates. The simulation results will report the percentage of simulated trials that recommended a combination within ±tox.range of the target rate.

The output is composed of:

- trial – Data frame containing output of a single simulated trial. Output includes the combination on which each patient was treated, toxicity information for each patient, the estimated value of the model parameter and the estimated ordering of toxicity probabilities. Only output if nsim=1.
- true.prob – True toxicity probabilities.
- MTD.selection – The distribution of MTD estimates. If nsim=1, this is a single numeric value of the recommended MTD combination of a single simulated trial.
- patient.allocation – The distribution of patient allocation at each combination.
- percent.DLT – Average percentage of DLT's across all simulated trials.
- mean.n – Average number or patients treated in simulated trials.
- acceptable – Percentage selection of combinations within ± tox.range of the target rate.

4. Examples

4.1. getwm function

In order to illustrate the features of the getwm function, we will use the example from Section 1.1. The twelve combinations under investigation are labled as in Table 1. The subset of possible orderings of the toxicity probabilities will be the first argument, orders, input into the function getwm. For this example, the subset of orders are given in Set 1 in Section 1.1. The next argument to input is the skeleton values associated with the toxicity probability of each combination. We merely need to input a single vector representing a reasonable skeleton[8], which is characterized by spacing rather than the actual values themselves [9]. The getwm function will adjust the location of the values to automatically correspond to the possible orderings of the toxicity probabilities specified by orders.

Specifying skeletons with adequate spacing on the interval (0, 1) can be more challenging when there are many combinations. We can rely on the algorithm of Lee and Cheung [10] to generate adequate spacing between skeleton values at neighboring combinations, without having to rely on a clinician's estimate at every combination. The algorithm of Lee and Cheung [10] requires four pieces of information in order to generate the skeleton; the prior MTD (ν), the target toxicity rate, θ, the number of combinations, d and the value of the indifference interval half-width, δ. It is necessary to ensure that there is enough spacing both below and above the prior MTD (i.e. target value). Therefore, we recommend placing the prior MTD, ν, at a combination roughly toward the middle dose levels, so that there is an approximately equal number of combinations below and above the prior MTD. Prior MTD's placed toward the boundary of the dose space will not produce skeletons with adequate spacing when there are many combinations being investigated. Using the required information, we can generate skeleton values using the getprior function in R package dfcrm. (i.e. getprior(δ, θ, ν, d)). We simply need to specify skeleton values at each combination that are adequately spaced, and adjust them to correspond to each of the possible orderings, in order for PO-CRM to have good performance in terms of identifying an MTD.

```
R> skeleton<
-c(0.01,0.09,0.17,0.25,0.33,0.41,0.49,0.57,
0.65,0.73,0.81,0.89)
R> alpha< -getwm(orders,skeleton)
```

The output generated from getwm is a matrix of skeleton values corresponding to the possible orders of toxicity probabilities specified by orders.

```
R> alpha
[,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
[,10] [,11] [,12]
[1,] 0.01 0.09 0.17 0.25 0.33 0.41 0.49 0.57
0.65 0.73 0.81 0.89
[2,] 0.01 0.33 0.65 0.09 0.41 0.73 0.17 0.49
0.81 0.25 0.57 0.89
[3,] 0.01 0.09 0.25 0.17 0.33 0.49 0.41 0.57
0.73 0.65 0.81 0.89
[4,] 0.01 0.17 0.41 0.09 0.33 0.65 0.25 0.57
0.81 0.49 0.73 0.89
[5,] 0.01 0.17 0.25 0.09 0.33 0.65 0.41 0.57
0.73 0.49 0.81 0.89
[6,] 0.01 0.09 0.41 0.17 0.33 0.49 0.25 0.57
0.81 0.65 0.73 0.89
```

The user can confirm that each row of skeleton values is ordered according to the corresponding row in orders by using the code order(alpha[1,]) (for the first row). The matrix alpha will be an argument needed to use the following two pocrm functions. As an exercise, one can enter the eight possible orderings provided in Table 2 (using the subscripts of the d_i) of Wages, Conaway and O'Quigley [2] and the skeleton values in the first row of Table 4 of that article and getwm will generate the remainder of Table 4.

4.2. pocrm.imp function

We will illustrate the features of pocrm.imp through the same example as above, using twelve combinations and six possible orderings of DLT probabilities. This function takes cumulative patient data and returns a recommended combination for the next patient entered, according to the model-based DLT probability estimates. It provides (1) the estimated ordering of DLT probabilities, (2) the estimated value of the parameter a, (3) the dose combination recommended to the next patient and (4) the toxicity outcome of each entered patient. The first argument needed is the matrix of skeleton values generated from getwm after specifying the possible orderings a single skeleton vector.

```
R> alpha< -getwm(orders,skeleton)
```

Next, we need to specify a vector of prior probabilities representing our belief regarding the plausibility of each ordering. The length of this vector must be equal to the number of orderings and must sum to 1. For this example, we place equal probability on each ordering and use the following code.

```
R> prior.o< -rep(1/6,6)
```

For the final three arguments, we need a target toxicity rate, the toxicity indicators for each patient enrolled and combinations tried. After observing each patient's response, the user can update the toxicity data and combinations tried in order to generate the recommendation for the next patient. Suppose we have data for the first 11 patients entered into the example of Table 1. This information can be captured by the following code.

```
#Target toxicity rate
R> theta< -0.25
#Toxicity outcomes on the first 11 patients.
R> y< -c(0,0,0,1,0,0,1,0,0,0,1)
#Combinations tried on the first 11 patients.
R> combos< -c(1,2,4,3,2,5,8,7,5,5,3)
```

The output from the use of the pocrm.imp function is given by the following.

```
R> fit< -
pocrm.imp(alpha,prior.o,theta,y,combos)
R> fit
$order.est
[1] 2
$a.est
[1] 1.546
$ptox.est
[1] 0.00 0.18 0.51 0.02 0.25 0.61 0.06 0.33
0.72 0.12 0.42 0.84
$dose.rec
[1] 5
```

Combination 5 is recommended for the 12th patient because it has an estimated toxicity rate (0.25) closest to the target rate of 25%. One may notice that we have moved from combination 3 to 5 after a DLT was observed. Keep in mind that this is not an escalation because we do not know whether combination 3 or 5 is more toxic due to the partial ordering among combinations. Therefore, it is unknown whether we are in fact escalating or de-escalating. Again as an exercise, the user can use the alpha matrix from Table 4 of [2], as well as the toxicity information and combinations tried in Table 5, to generate the estimated ordering, estimated parameter value, estimated toxicity probabilities, and recommended combination for any patient in Table 5 of [2]. Note that it's possible for the estimated ordering to change from some of the values given in Table 5 of [2] given the fact that if two or more orderings have equal probability, the estimate is randomly chosen from the tied orderings.

4.3. pocrm.sim function

For the pocrm.sim function, the initial escalation scheme, x0, is a vector specifying the order in which combinations should be tried before the first DLT and the size of the cohort at each combination in Stage 1. The clinical motivation underlying the use of an initial escalation stage is the desire of a clinician to be conservative in escalation and begin at the lowest combination. Mathematically, the likelihood fails to have a solution in the absence of at least one toxic and one nontoxic response [11]. Therefore, we need an initial escalation scheme in order to obtain this required heterogeneity in the observed responses. This could be done in groups of 1, 2, or 3 patients. Wages, Conaway and O'Quigley [1,2] suggest formulating the initial escalation scheme by "zoning" the matrix of combinations according to the diagonals of Table 1. For instance, if working in cohorts of 2, enter the first two patients on combination 1 (Zone 1). If no DLT's, escalate to Zone 2 and enter the next cohort on a combination chosen from 2 or 4. If no DLT's in that cohort, enter the next cohort on the combination in Zone 2 that has not yet been tried. Escalation to a higher zone occurs only when all combinations in the lower zone have been tried, and no DLT has been observed. In R, this initial escalation scheme would require the following input.

```
R> x0< -c(rep(1,2),rep(2,2),rep(4,2),rep(3,2),
rep(5,2),rep(7,2),rep(6,2),
R> rep(8,2))rep(10,2),rep(9,2),rep(11,2),
rep(12,2)).
```

This scheme will only be relied upon until the first DLT is observed, when the design switches to Stage 2 and the user is free to specify any reasonable scheme he or she sees fit. For instance, some may not feel the necessity to exhaust every combination in each zone before escalating to a higher zone. In this case, it may be appropriate to enroll cohorts of patients at only one combination within a zone and, if no DLT, escalate to the next highest zone, especially if it is felt that combinations within the same zone have similar toxicity profiles. Wages, Conaway and O'Quigley [2] discussed the notion of randomizing patients to combinations within a zone, since we do not know the toxicity ordering between combinations in the same zone. For the purposes of simulation in this package, this randomization would need to be done separately by the user prior to specifying x0. A stipulation on x0 is that its length must not exceed n.

In practice, investigators are likely to stop the trial if there are already many patients on the suggested treatment for the next patient. The stopping rules for the pocrm.sim function are as follows. The design will stop at the end of Stage 1 if escalation proceeds to the highest dose combination and stop patients are treated with no DLT's. In this case, the study is stopped and the highest dose combination is declared the MTD combination. In the Stage 2, if the recommendation is to assign the next patient to a combination that already has stop patients treated on the combination, the study is stopped and the recommended combination is declared the MTD combination. The maximum sample size is denoted n. If stop>n, then the trial will exhaust a pre-determined, fixed sample size of n patients.

We continue utilizing the example of Table 1 to illustrate the features of the **pocrm** package. The pocrm.sim function outputs different results, depending on whether the user specifies the number of simulations to be 1 or some integer greater than 1. The initial escalation scheme is conducted in cohorts of 1 patient. We set the maximum sample size to n=36 and stop=50, so as to exhaust a predetermined, fixed sample size of 36 patients. The range of toxicity probabilities that defines "acceptable" combinations is set to 5%. That is, an "acceptable" combination is defined as any combination with a true DLT probability within ±0.05 of the target toxicity rate, theta. This argument is not used when only simulating a single trial and is ignored if nsim=1. It will only be used when generating results over multiple simulation runs. The results of pocrm.sim for a single simulated trial is generated via the code below, using alpha, theta and prior.o from above. For the sake of brevity, we omit the output from patients 13–33.

```
#Vector of true toxicity rates
R> r< -c(0.03,0.06,0.12,0.08,0.14,0.20,
0.16,0.22,0.28,0.24,0.30,0.36)
#Initial escalation in Stage 1 proceeds
according to the zones (diagonals).
```

```
#Single patient cohorts are used.
R> x0< -c(rep(1,1),rep(2,1),rep(4,1),rep(3,1),
rep(5,1),rep(7,1),rep(6,1),rep(8,1),
R> rep(10,1),rep(9,1),rep(11,1),rep(12,1))
#Total number of patients used to define
stopping rule
R> stop< -50
#Maximum sample size.
R> n< -36
#Number of simulations
R> nsim< -1
R> fit< -
pocrm.sim(r,alpha,prior.o,x0,stop,n,theta,nsim)
R> fit
$trial

     patient  level  tox  a         order
 1   1        1      0    0.000000  99
 2   2        2      0    0.000000  99
 3   3        4      0    0.000000  99
 4   4        3      0    0.000000  99
 5   5        5      0    0.000000  99
 6   6        7      0    0.000000  99
 7   7        6      1    2.082869  5
 8   8        10     0    2.955495  2
 9   9        3      0    2.523537  5
 10  10       8      1    1.828420  5
 11  11       10     0    2.026418  5
 12  12       10     0    2.207831  5
 .
 .
 .
 34  34       10     0    2.068440  4
 35  35       10     0    2.115491  4
 36  36       10     0    2.161201  5
 37  37       10     0    0.000000  0

$MTD.selection
[1] 10
```

It is important to note that, in Stage 1, the estimated ordering is set to some arbitrary number (99) since we are not estimating the ordering until Stage 2. This also applies to the estimates for the parameter *a* which also aren't generated until Stage 2. We can see that once the first DLT is observed (patient 7), the model estimates are generated. Further notice that we enrolled 36 patients on the trial and the MTD selected is the combination that would have been recommended for the 37th patient, had he or she been enrolled. Combination 10 has true DLT probability 0.24, which is close to our target rate of 25%.

We will now illustrate performance of the POCRM over many multiple simulation runs. Using the same arguments as above, with the exception of changing nsim=1 to nsim=250, we can generate performance of 250 simulated trials.

```
$true.prob
[1] 0.03 0.06 0.12 0.08 0.14 0.20 0.16 0.22
0.28 0.24 0.30 0.36
$MTD.selection
[1] 0.00 0.01 0.06 0.01 0.08 0.22 0.08 0.15
0.14 0.09 0.12 0.04
$patient.allocation
[1] 0.03 0.05 0.10 0.05 0.09 0.15 0.09 0.11
0.10 0.10 0.09 0.03
$percent.DLT
[1] 0.1946667
$mean.n
[1] 36
$acceptable
[1] 0.728
```

The results indicate that POCRM recommended, as the MTD, a combination with a true toxicity probability between 0.20 and 0.30 (inclusive) in 72.8% of simulated trials. The average sample size is our fixed value of 36 patients and 19.5% of patients experienced a DLT. As an exercise, a user could rerun the last example but instead use the labels in Table 2, Set 2 of true probabilities and the orderings associated with these labels that are given in Section 1.1. Differences in the results may exist due to simulation error, but will disappear for higher values of nsim.

As a final example, we are going to rerun 250 simulated trials after changing some of the design specifications. We are going to use a different skeleton, give more prior weight to ordering 3, using a stopping rule of stop=10 and define the acceptable toxicity range as ±3%. Therefore, using the same true toxicity probabilities, the simulation results will report the percentage of trials in which a combination with a true DLT rate between 0.22 and 0.28 was selected as the MTD.

```
#Specify the skeleton values.
R> skeleton<
-c(0.02,0.05,0.09,0.12,0.16,0.24,0.30,0.36,
0.42,0.50,0.59,0.65)
#Initial guesses of toxicity probabilities for
each ordering.
R> alpha< -getwm(orders,skeleton)
#We consider ordering 3 to be the most likely.
R> prior.o< -c(0.15,0.15,0.25,0.15,0.15,0.15)
#Total number of patients used to define
stopping rule
R> stop< -10
#Maximum sample size.
R> n< -36
#The target toxicity rate
R> theta< -0.25
```

```
#Number of simulations
R> nsim< -250
#Definition of acceptable DLT rates
R> tox.range< -0.03
R> fit< -pocrm.sim(r,alpha,prior.o,x0,stop,n,
theta,nsim,tox.range)
R> fit
$true.prob
[1] 0.03 0.06 0.12 0.08 0.14 0.20 0.16 0.22
0.28 0.24 0.30 0.36
$MTD.selection
[1] 0.00 0.00 0.08 0.03 0.10 0.17 0.06 0.13
0.14 0.12 0.08 0.09
$patient.allocation
[1] 0.06 0.06 0.10 0.07 0.10 0.12
0.09 0.11 0.09 0.09 0.07 0.05
$percent.DLT
[1] 0.1876829
$mean.n
[1] 28.708
$acceptable
[1] 0.39
```

The results indicate that in approximately 39% of simulated trials, an “acceptable” combination was selected as the MTD, with an average sample size of 28.71 patients. On average, 18.77% of patients experienced DLT. We have provided a snapshot of examples demonstrating the use of the **pocrm** package. The package may not exactly reproduce the results of Wages, Conaway and O’Quigley [1] due to the fact that, in its original form, the POCRM was described as a Bayesian method. In Wages, Conaway and O’Quigley [1], POCRM did not contain a Stage 1, which will slightly alter operating characteristics of the design. This package was written for the more practical setting of beginning at a low combination and slowly escalating until a DLT is observed, before beginning the model-based portion of the design.

5. Conclusions

The **pocrm** package provides a means for implementing and simulating Phase I trials of combinations of agents according to the continual reassessment method for partial orders (POCRM). It has flexible operating characteristics, including the ability to specify any initial escalation scheme, to exhaust a fixed sample size or to stop the trial once a prespecified number of patients have been allocated to particular combination, to change other design specifications such as the possible orderings of the toxicity probabilities, the initial guesses of the toxicity probabilities under each ordering, and the target toxicity rate. We have not equipped the package with the ability to specify varying functional forms of the parameterized working model, and have limited its use to the commonly used power model. Many CRM-type one-parameter model possess similar operating characteristics and we feel the inclusion of just one model simplifies the implementation of the package for the user. The objective of **pocrm’s** output is to clearly and concisely report results so that biostatistician–clinician teams may easily engage in a discussion regarding the POCRM’s operating characteristics.

Acknowledgments

This research was supported by NCI grant 1R01CA142859. The authors would like to acknowledge the comments made by a reviewer, which have helped us sharpen the original submission.

REFERENCES

[1]. Wages N, Conaway M, O'Quigley J. Continual Reassessment Method for Partial Ordering. Biometrics. 2011; 67:1555–1563. [PubMed: 21361888]

[2]. Wages N, Conaway M, O'Quigley J. Dose-finding design for multi-drug combinations. Clinical Trials. 2011; 8:380–389. [PubMed: 21652689]

[3]. R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing; Vienna, Austria: 2012. ISBN: 3-900051-07-0,http://www.R-project.org/

[4]. Wages N, Conaway M, O'Quigley J. Using the time-to-event continual reassessment method in the presence of partial orders. Statistics in Medicine. 2013; 32:131–141. [PubMed: 22806898]

[5]. Wages N, Conaway M. Specifications of a continual reassessment method design for phase I trials of combined drugs. Pharmaceutical Statistics. 2013 http://dx.doi.org/10.1002/pst.1575.

[6]. Chevret S. The continual reassessment method in cancer phase I clinical trials: a simulation study. Statistics in Medicine. 1993; 12:1093–1108. [PubMed: 8210815]

[7]. Paoletti X, Kramar A. A comparison of model choices for the continual reassessment method in phase I cancer trials. Statistics in Medicine. 2009; 28:3012–3028. [PubMed: 19672839]

[8]. O'Quigley J, Zohar S. Retrospective robustness of the continual reassessment method. Journal of Biopharmaceutical Statistics. 2010; 20:1013–1025. [PubMed: 20721788]

[9]. Iasonos A, O'Quigley J. Interplay of priors and skeletons in two-stage continual reassessment method. Statistics in Medicine. 2012; 31:4321–4336. [PubMed: 22893483]

[10]. Lee SM, Cheung YK. Model calibration in the continual reassessment method. Clinical Trials. 2009; 6:227–238. [PubMed: 19528132]

[11]. O'Quigley J, Shen L. Continual reassessment method: a likelihood approach. Biometrics. 1996; 52:673–684. [PubMed: 8672707]

Table 1

Combination labels across rows for TLR and IFA forming a 4 × 3 matrix order.

Doses of *A*	**IFA**		
	0	**V1**	**V6**
4	10	11	12
3	7	8	9
2	4	5	6
1	1	2	3

Table 2

Combination labels up diagonals for TLR and IFA forming a 4 × 3 matrix order.

Doses of A	**IFA**		
	0	**V1**	**V6**
4	9	11	12
3	6	8	10
2	3	5	7
1	1	2	4

Table 3

True toxicity probabilities for combinations forming a 4 × 3 matrix order.

Doses of *A*	**IFA**		
	0	**V1**	**V6**
4	0.24	0.30	0.36
3	0.16	0.22	0.28
2	0.08	0.14	0.20
1	0.03	0.06	0.12