Continual Reassessment Method

Adrian Mander

MRC Biostatistics Unit Hub for Trials Methodology Research, Cambridge

Sep 2011

Outline

- Introduction to oncology phase I trials
- Describe the original continual reassessment model (CRM)
- How to run the Mata code
- Programming difficulties

Oncology Phase I trials

Want to find maximum tolerated dose(MTD) of a novel drug

- The outcome is binary
  - A patient has a dose-limiting toxicity or not (e.g. Neutropenia)
- A set of doses are (usually) pre-specified
- Assume a monotonically increasing relationship between dose and outcome
- Want to find a safe dose that has a certain target probability of toxicity (TTL)
  - If the MTD is d^* then $\mathbb{P}(\text{Tox}|d^*) = TTL$
- the target probability is often between 0.2 and 0.33

Continual Reassessment Model

This model-based Bayesian method was introduced by J. O'Quigley *Biometrics* 1990.

- A working model is specified for the dose-outcome relationship
- Prior information is required
- Then the study begins by dosing the first person at the "best" dose
- The analysis is updated given the data obtained
- For the next patient pick the "best" dose and continue

Sample size generally fixed at the outset (20-30 participants)

Single-parameter working models

$$\mathbb{P}(\mathbf{d}, a) = \text{prob of a Tox at dose } \mathbf{d}$$

The following models were suggested in the original paper

- 1-parameter logistic : $\mathbb{P}(\mathbf{d}, a) = \frac{\exp(-3+a\mathbf{d})}{1+\exp(-3+a\mathbf{d})}$
- power : $\mathbb{P}(\mathbf{d}, a) = p_{\mathbf{d}}^{\exp a}$
- hyperbolic tangent : $\mathbb{P}(\mathbf{d}, a) = \left(\frac{\exp(\mathbf{d})}{\exp(\mathbf{d})+\exp(-\mathbf{d})}\right)^{a}$

Implementing the CRM

a is the parameter that is going to be updated during the trial

- exponential prior $\pi(a) = \exp(-a)$ with mean 1

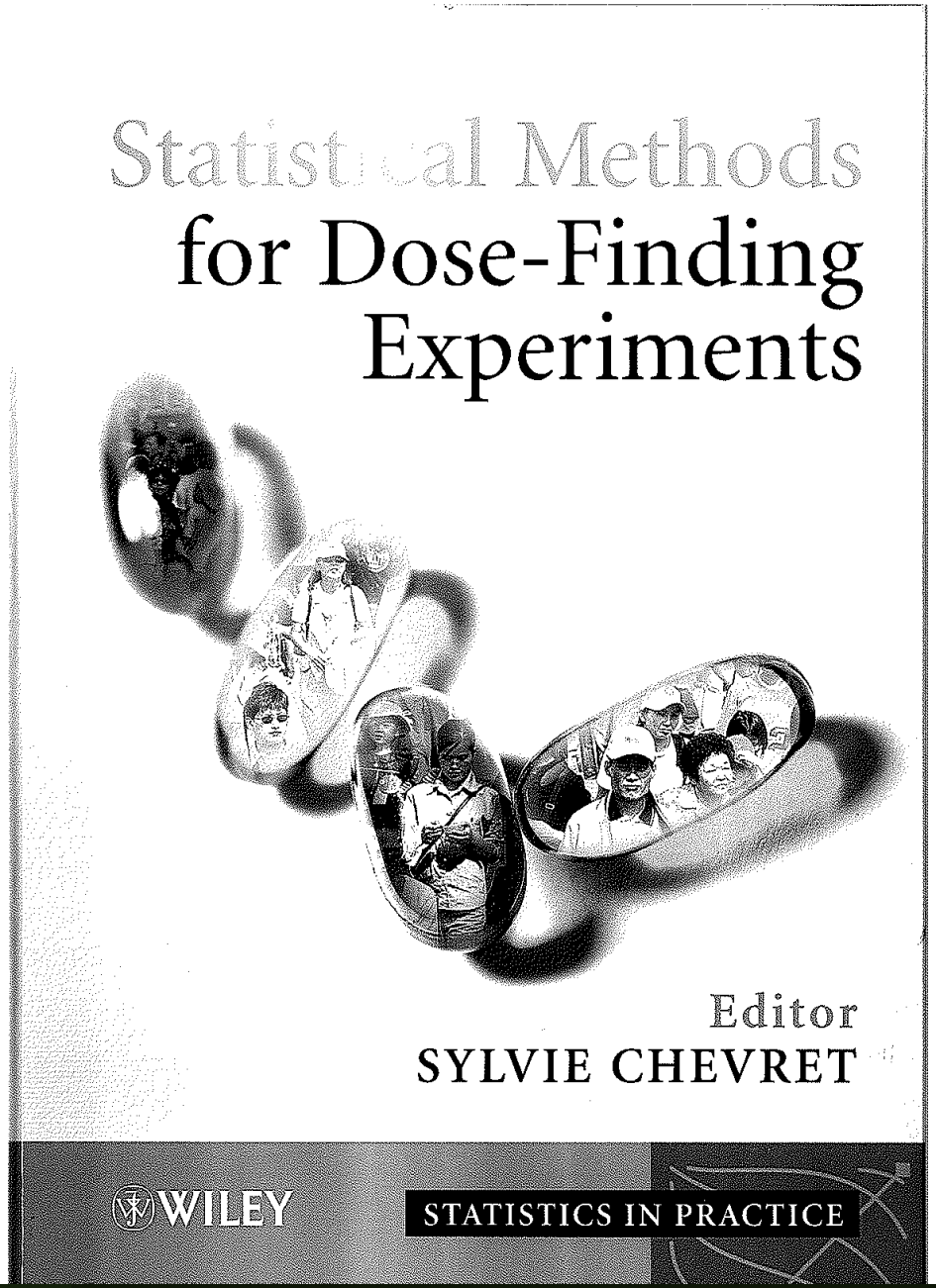
Given the data for doses x_i and outcomes y_i the likelihood is

$$f(x|a) = \prod_i \mathbb{P}(x_i, a)^{y_i}(1 - \mathbb{P}(x_i, a))^{1-y_i}$$

The posterior is

$$\pi(a|x) = \frac{f(x|a)\pi(a)}{\int_0^\infty f(x|a)\pi(a)da}$$

Calculate the posterior mean (another integration) and then substitute this into the model to pick the best dose


Statistical Methods
for Dose-Finding
Experiments
Editor
SYLVIE CHEVRET
WILEY
STATISTICS IN PRACTICE

15.2 Computation methods using statistical software

Most common commercial statistical software give all mathematical functions in order to write and implement dose-finding analysis. Integration functions, for Bayesian methods or the maximum likelihood function, are often available and easy to use. Nevertheless, statisticians need to have some minimal knowledge and practice in programming language. In this chapter, a simple S program is presented where the continual reassessment method (CRM) is implemented (for S-PLUS 6.2 and R 2.1)[1]. This method was chosen for the increased number of publications that deal with it and applications in dose-finding clinical trials [2–5, 7–10]. In this example the used dose-response model is a power function, α_i^a (with $i = 1, \ldots, k$), where k is the number of dose levels and a is random with a unit exponential prior distribution [4]. The original CRM is a Bayesian method (Chapter 6), so it requires an integration function. Firstly, the function to be integrated needs to be defined:

```
crmh <- function(x){(exp(-x)*vcrm(x))}
crmht <-function(x){x*(exp(-x)*vcrm(x))}
```

with `vcrm` the likelihood function:

```
vcrm <- function(a){
v <- 1
for(i in (1:length(x1p)))
{v <- v*((x1p[i]^a)^y1p[i])*(((1-x1p[i]^a)^(1-y1p[i])))}
return(v)}
```

Then, the CRM function is defined by

```
crm <- function(n,prior,target,tox,dose){

### function parameters initializing
ptox <- matrix(NA,nrow=n,ncol=length(prior))
dcrm <- x1p <- y1p <- rep(0,n)

### dose allocation procedure
for (i in 1:n)
{
### x1p and y1p are global variable needed for the integration
x1p <<- c(prior[dose[1:i]])
y1p <<- tox[1:i]

### ptox is the matrix of the estimated probabilities of
   responses
ptox[i,]<- prior^(integrate(crmht,0,Inf)[[1]]/integrate
   (crmh,0,Inf)[[1]])
```



```
### dcrm is the vector of the sequential recommended dose level
dcrm[i] <-which(abs(ptox[i,]-target)==min(abs(ptox[i,]-
   target)))
}
### results printing
cat("N","\t","Dose","\t","Tox","\t", paste("pt",seq(1,length
   (prior))), "\t","Recommended dose","\n")
for (i in 1:n){
cat(i,"\t",dose[i],"\t",tox[i],"\t",round(ptox[i,],3),"\t",
   dcrm[i],"\n")}
}
```

The input of the `crm` function is the following:

1. `"n"` is the number of patients to be included in the trial for which observations are available.
2. `"prior"` is the vector of prior response probabilities associated with each dose level.
3. `"target"` is the toxicity target.
4. `"tox"` is the vector of patient's observations (1 = toxicity and 0 = no toxicity).
5. `"dose"` is the vector of dose levels attributed to each patient included in the trial.

To run the program[4] under either (a) R software, where the user has to load the functions `vcrm`, `crmh`, `crmht` and `crm` by writing the following commande `source(file="my save path/crm.r")`, or (b) S-SPLUS software, where the user has to go to the menu option `"script"` and select `"run"`. For example, for a dose-finding where 10 patients were included, the execution commands for the two software are respectively:

```
crm(n=10,prior=c(0.04,0.07,0.2,0.35,0.55,0.7),target=0.2,
tox=c(0,1,0,0,0,1,0,0,0,0),dose=c(1,3,2,2,2,3,2,2,2,2))
```

Figure 15.1 represents the output of the program where (a) `"N"` is the patient rank, (b) `"Dose"` is the given dose level to those patients, (c) `"Tox"` is the patient observation, (d) `"Pt1, ..., Pt6"` are the sequential estimation of the probabilities of response at each dose level and (e) `"Recommended dose"` is the dose level recommended by the CRM. In this example, after the inclusion of 10 patients the

[4] This program can be downloaded at: http://dbim.chu-stlouis.fr/soft.html.crm.r for R software and crm.ssc for S-PLUS software.

Example from Statistical Methods for Dose-Finding Experiments

The prior probabilities for each toxicity ("skeleton") for 6 dose levels need to be specified

d_1	d_2	d_3	d_4	d_5	d_6
.04	.07	.2	.35	.55	.7

This is fixed throughout the trial and dictates the shape of the curve

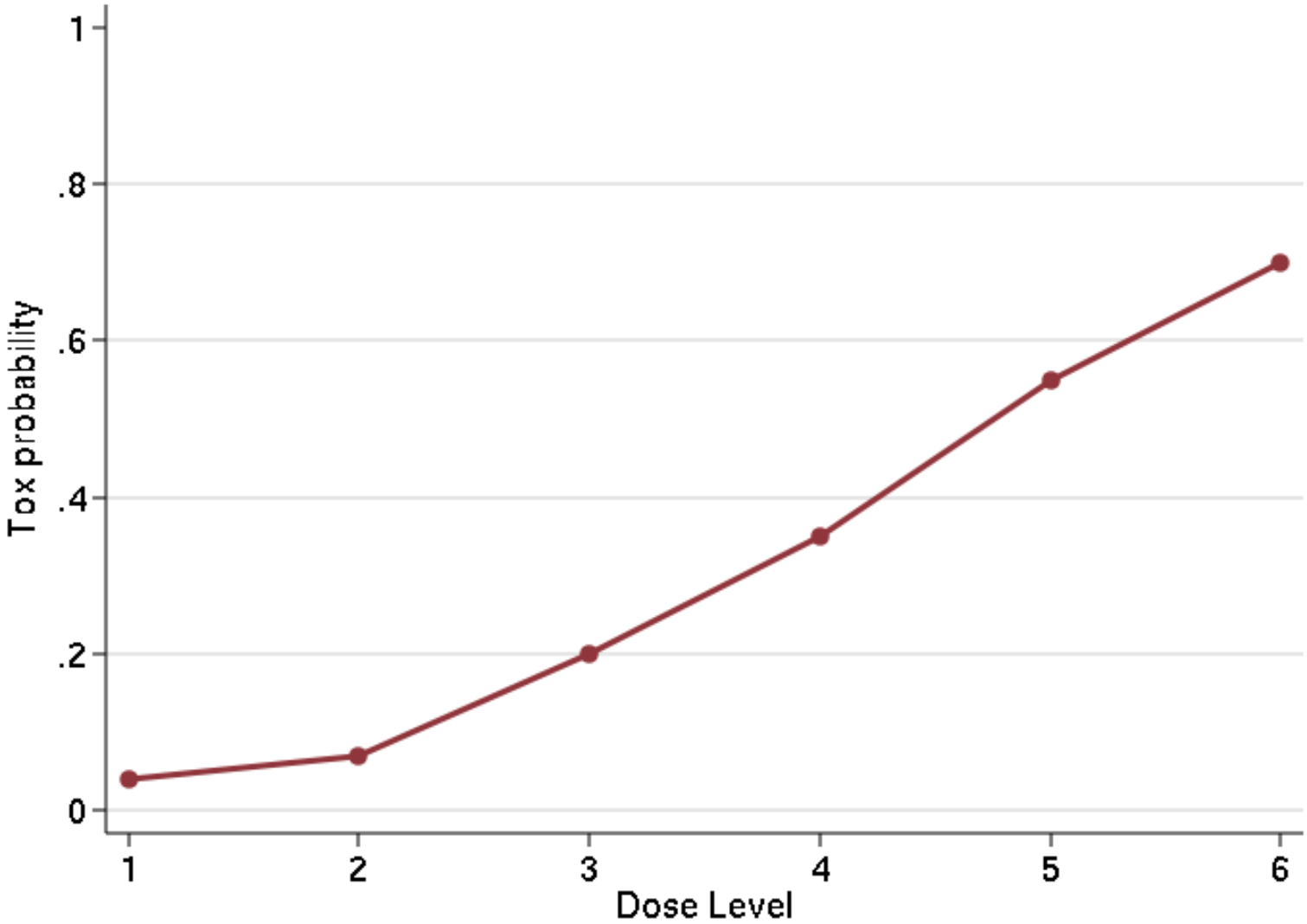
- TTL is specified as 0.2
- Default working model is hyperbolic tangent
- Default prior is exponential

Stata Code

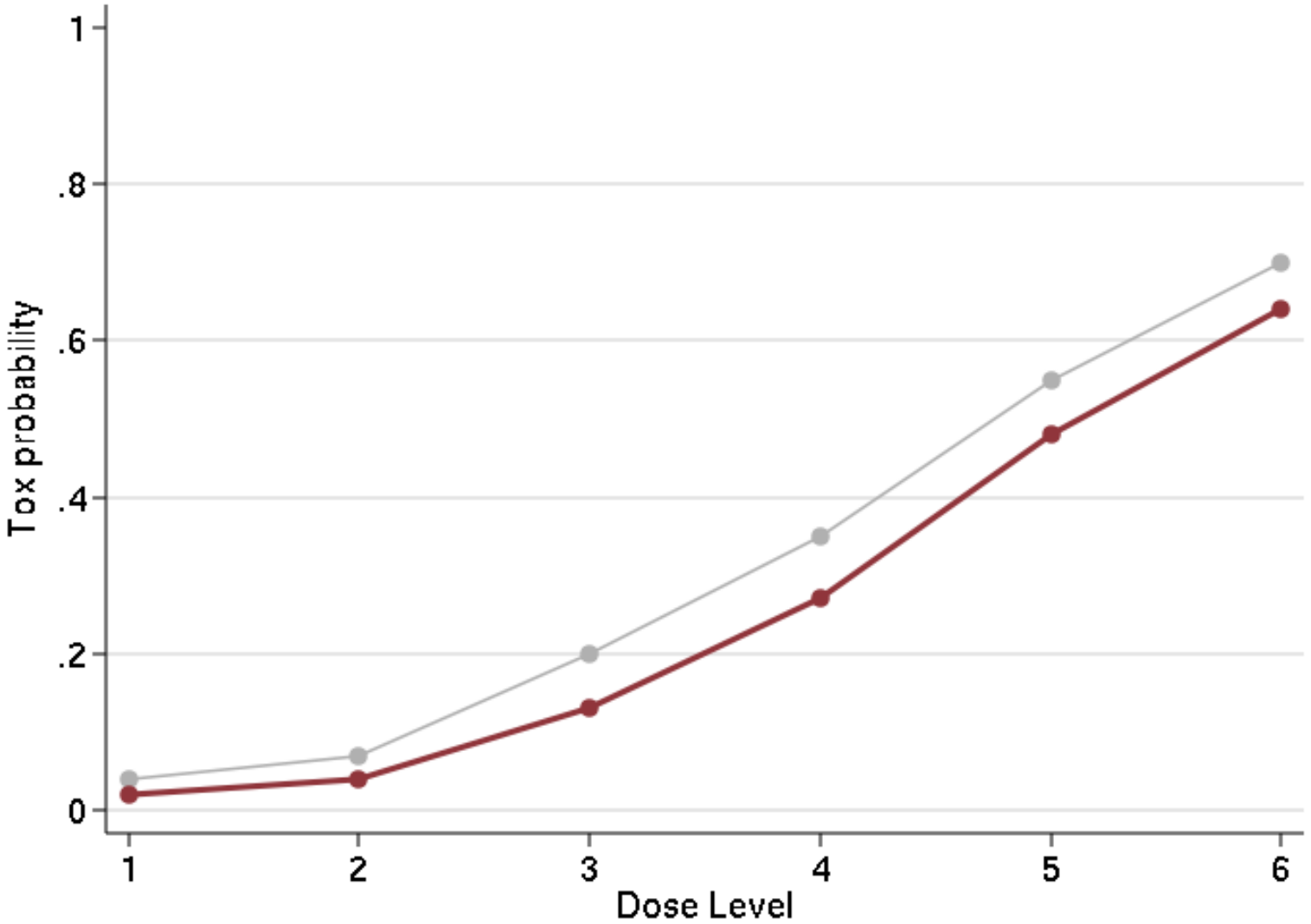
```
crm y dose, target(0.2) prior(0.04 0.07 0.2 0.35 0.55
0.7)
```

```
. list y dose
     +-------------+
     | y    dose   |
     |-------------|
  1. | 0       1   |
  2. | 1       3   |
  3. | 0       2   |
  4. | 0       2   |
  5. | 0       2   |
     |-------------|
  6. | 1       3   |
  7. | 0       2   |
  8. | 0       2   |
  9. | 0       2   |
 10. | 0       2   |
     +-------------+
```

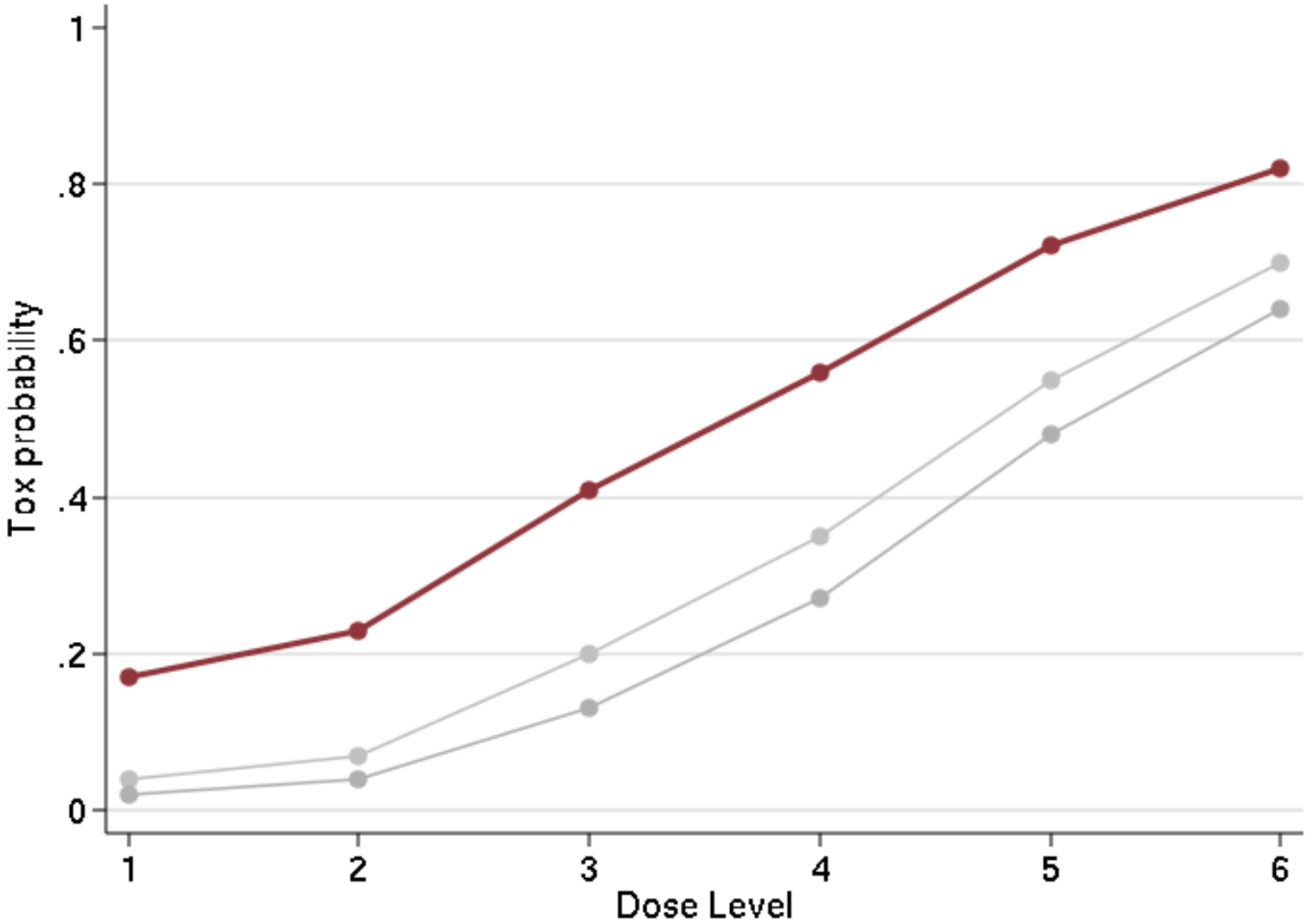
Prior model, but dose the first patient at dose level 1



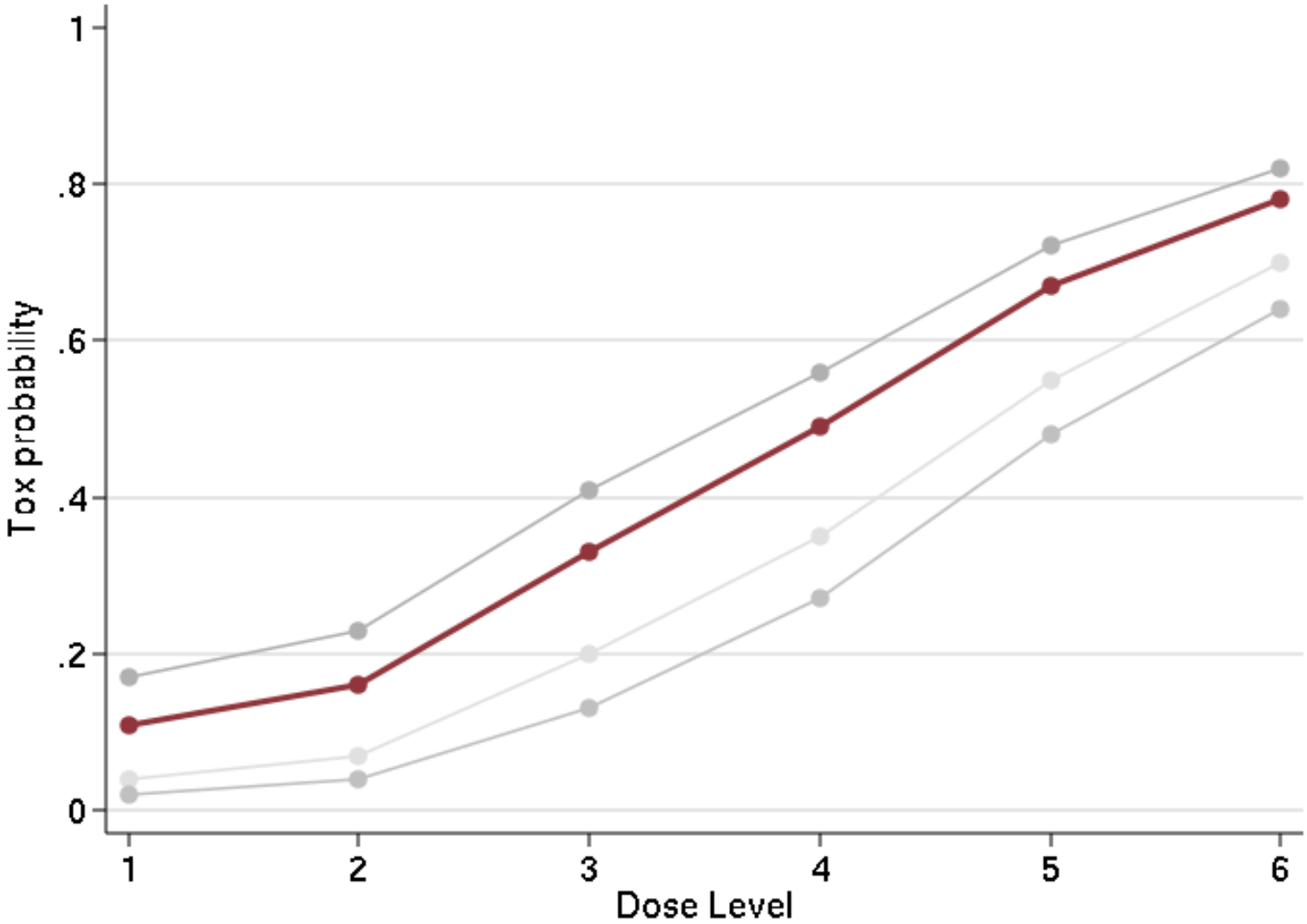
First person had no tox at dose 1, next patient is given dose 3



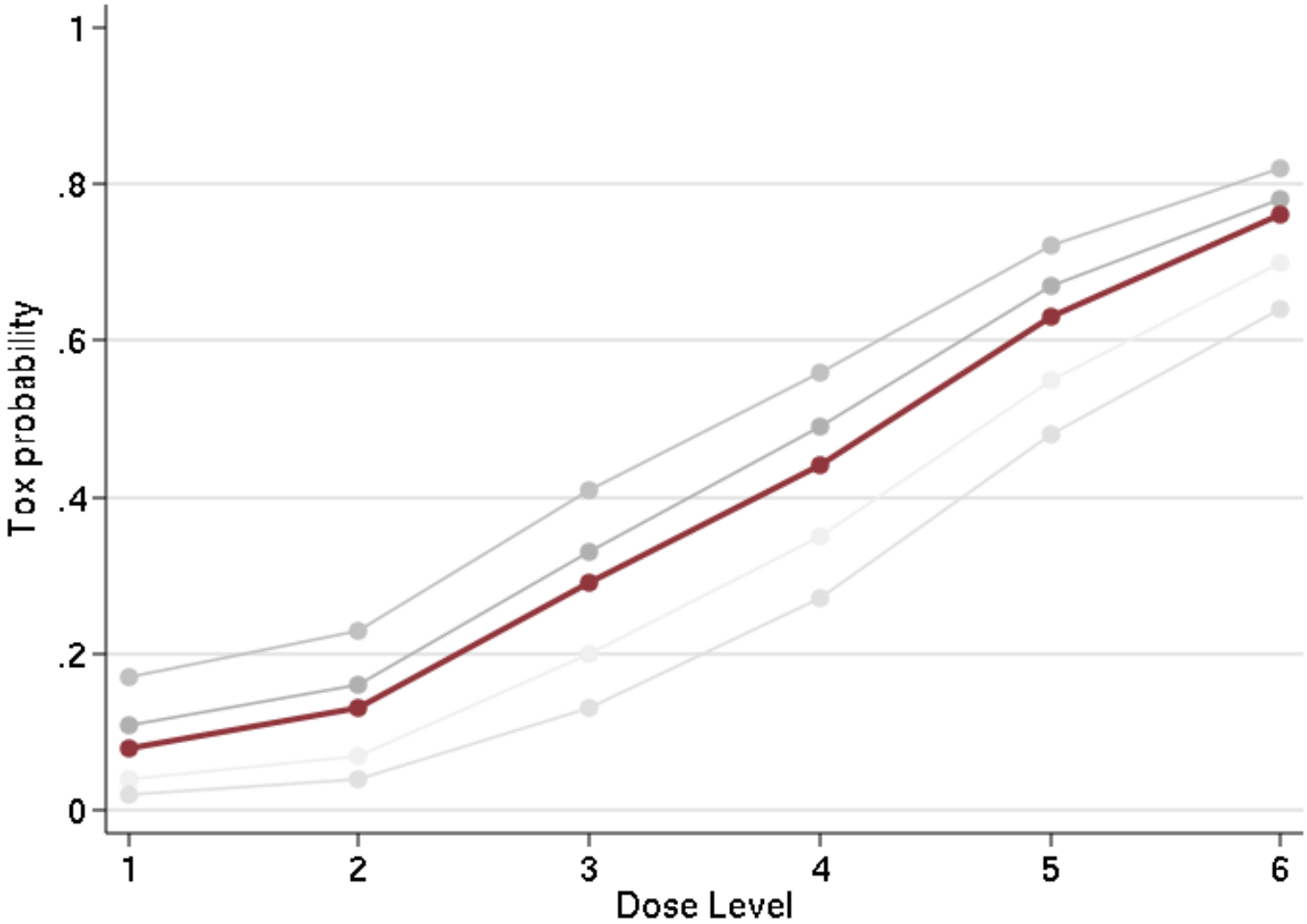
Patient 2 experienced tox at dose 3, next patient is given dose 2



Patient 2 no tox at dose 2, next patient is given dose 2



Patient 2 no tox at dose 2, next patient is given dose 2



Numerical Integration

In order to integrate I had to program up the GaussLaguerre quadrature (see Wiki).

```
matrix Ln_coef(real n)
{
  first=1
  for (i=0;i<=n;i++) {
    if (first++==1)  coef = (-1)^i*comb(n,i)/factorial(i)
    else coef = coef, (-1)^i*comb(n,i)/factorial(i)
  }
  return(coef)
}

real integrate0inf(real fn, real obs, scalar model, dose, y, d, quadpts)
{
  roots= sort(polyroots(Ln_coef(quadpts))',1)'
  for(i=1;i<=quadpts;i++) {
    if (i==1) wt = roots[i]/((quadpts+1)^2*(Ln(quadpts+1, roots[i])^2))
    else    wt=wt, roots[i]/((quadpts+1)^2*(Ln(quadpts+1, roots[i])^2))
  }
  integral=0
  for(i=1;i<=quadpts;i++) {
    integral = integral+ wt[i]*exp(roots[i])*f(fn, obs, roots[i], model, dose, y, d)
  }
  return(integral)
}
```

Final Thoughts

- Numerical integration could be done at a higher precision (but not by me!). Perhaps have a mechanism for doing some calculations at quad-precision?
  - Perhaps R has a better integrate function than the one I wrote?
- To generalise the code it would be good to implement the following syntax, is this possible?

```
crm y dose, target(0.2) prior(0.04 0.07 0.2 0.35 0.55 0.7)
  model( x^exp(a) ) prior( exp(-2*a) ) }
```