Designing Phase I Single Agent Dose-Escalation Studies

Lecture 5: Introduction to crmPack

Pavel Mozgunov & Thomas Jaki

MRC Biostatistics Unit

September 26, 2023

Why we wrote `crmPack`

- More flexible than commercial software (FACTS, East, Addplan DF)
- Wish to adopt and implement new designs quickly
- Other R-packages on CRAN (bcrm, dfcrm, CRM) are not (easily) extensible
- `crmPack` development started in 2014
- Package is available on CRAN since 2016 (latest version: 0.2.7)

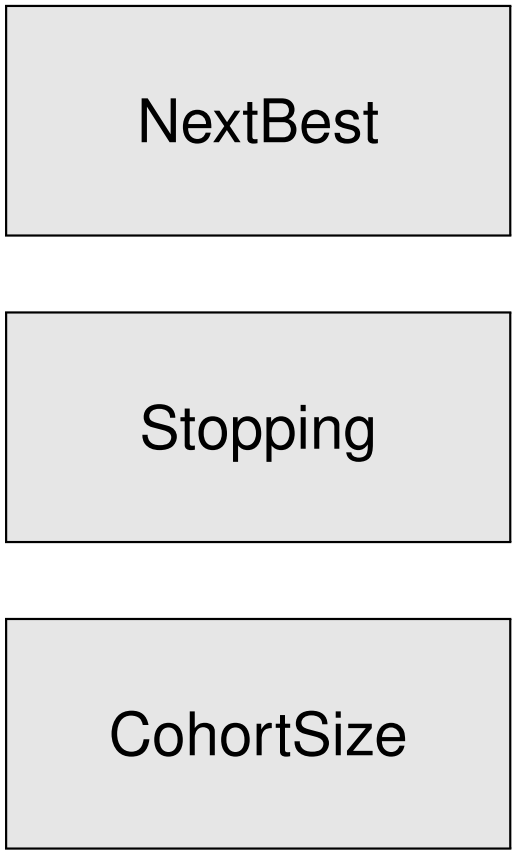
Differences to other packages

- Allows inclusion of placebo control
- Implements bivariate methods (safety and efficacy)
- Is based on S4 classes and methods
- It can make nice plots . . .

Object oriented programming

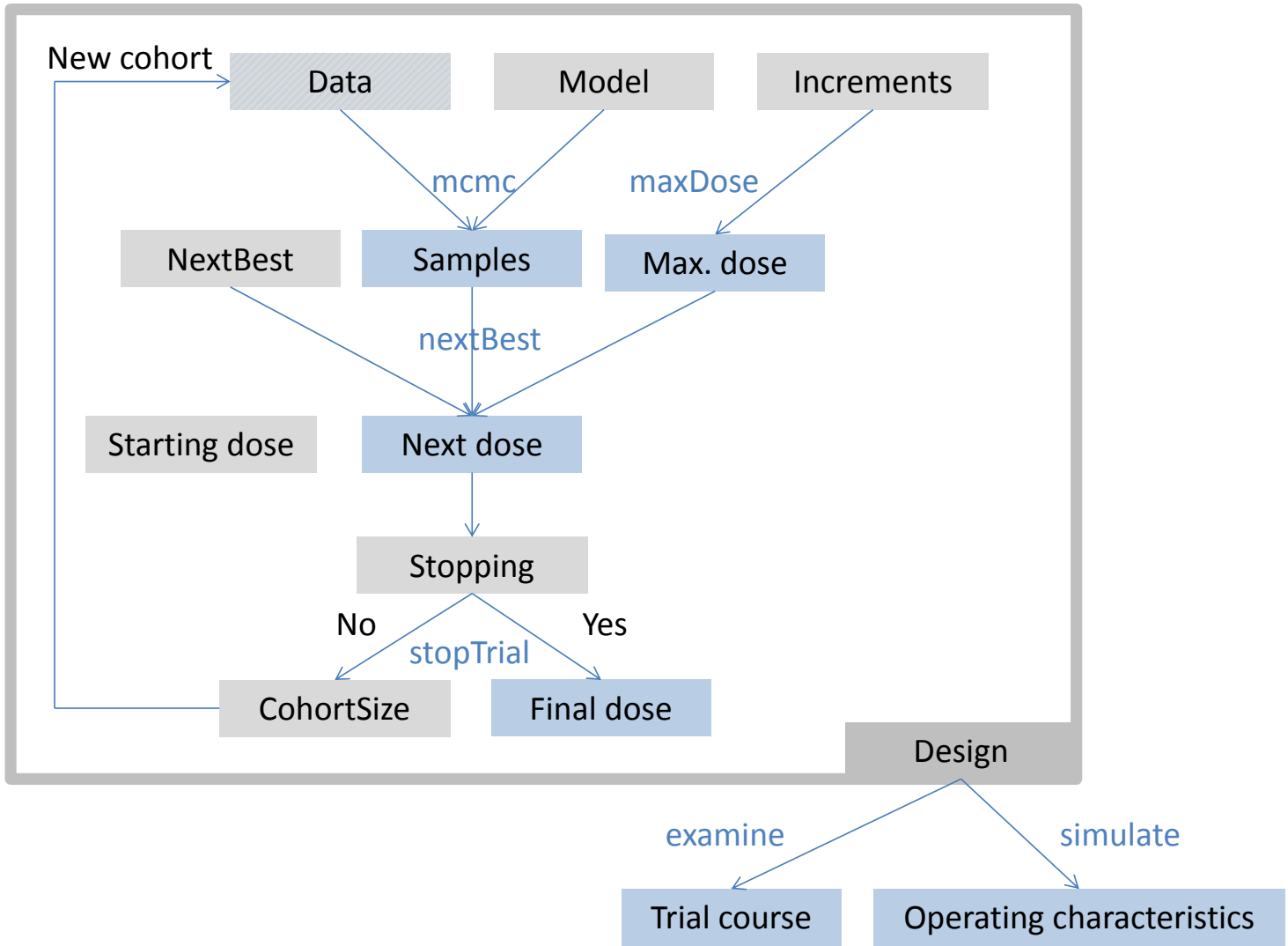
- Identify real objects and operations on them that are interesting
- Examples:
  - Object: cdf or pdf
  - Method: mean, median, skewness,...
- Objects should know how to deal with all relevant operations
  - cdf should know how to find a mean, but not how to fit a linear model

Class structure in `crmPack`



- How to identify the next dose given the model and data
- conditions under which the dose escalation is stopped
- Determines number of patients in the next cohort of the trial

`crmPack` classes and methods



Data object

Contains dose grid and current data

- Load package

Data

```
library(crmPack)
Loading required package: ggplot2
Type crmPackHelp() to open help browser
Type crmPackExample() to open example
```

- Create dose grid

```
> doseGrid <- seq(from=40, to=200, by=10)
> doseGrid
 [1] 40  50  60  70  80  90 100 110 120 130 140 150 160 170 180 190 200
```

Data object

Contains dose grid and current data

- Create data object
  - 2 patients on 40mg without DLTs
  - 2 patients on 50mg with one DLT

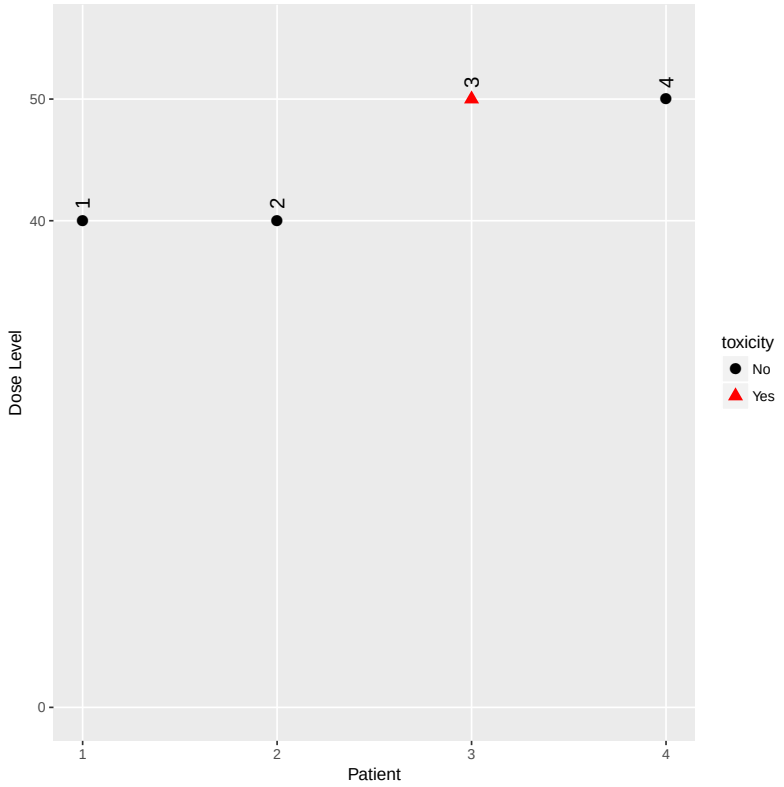
Data

```
> data <- Data(x=c(40,40,50,50), y=c(0,0,1,0),
+              doseGrid=doseGrid)
Warning messages:
1: In Data(x = c(40, 40, 50, 50), y = c(0, 0, 1, 0), doseGrid = doseGrid) :
  Used default patient IDs!
2: In Data(x = c(40, 40, 50, 50), y = c(0, 0, 1, 0), doseGrid = doseGrid) :
  Used best guess cohort indices!
> data@ID
[1] 1 2 3 4
> data@cohort
[1] 1 1 2 2
```

Data object

Contains dose grid and current data

```
> plot(data)
```



Data

Model object

Specification

The following model for the probability of toxicity at dose x, $\phi(x)$, is used:

Model

$$logit(\phi(x)) = \alpha_0 + \alpha_1 * log(x)$$

and prior distribution is

$$(\alpha_0, log(\alpha_1))^T \sim N\left(\begin{pmatrix} -4.47 \\ 0.0033 \end{pmatrix}, \begin{pmatrix} \sigma_0^2 & \rho\sigma_0\sigma_1 \\ \rho\sigma_0\sigma_1 & \sigma_1^2 \end{pmatrix}\right)$$

with $\sigma_0 = 1.0278,\ \sigma_1 = 1.65,\ \rho = 0.5.$

NOTE: No reference dose used!

Model object

Specification

Can use `LogisticLogNormal` as a readily available model

Model

```
> sigma0 <- 1.0278
> sigma1 <- 1.65
> rho <- 0.5
> cov <- matrix(c(sigma0^2,rho*sigma0*sigma1,
+                 rho*sigma0*sigma1,sigma1^2),
+                 nrow=2, ncol=2)
> model <- LogisticLogNormal(mean = c(-4.47, 0.0033),
+                            cov = cov, refDose = 1)
```

Obtaining samples

- The usual MCMC options

```
> mcmcOptions <- McmcOptions()
```

  - 10,000 burn-in iterations
  - step size 2
  - target sample size 10,000

- For prior use empty dataset

```
> emptyData <- Data(doseGrid=doseGrid)
```

- Run the sampler (normal distribution)

```
> set.seed(12)
> priorSamples <- mcmc(emptyData, model,
+                      mcmcOptions)
```

Obtaining samples

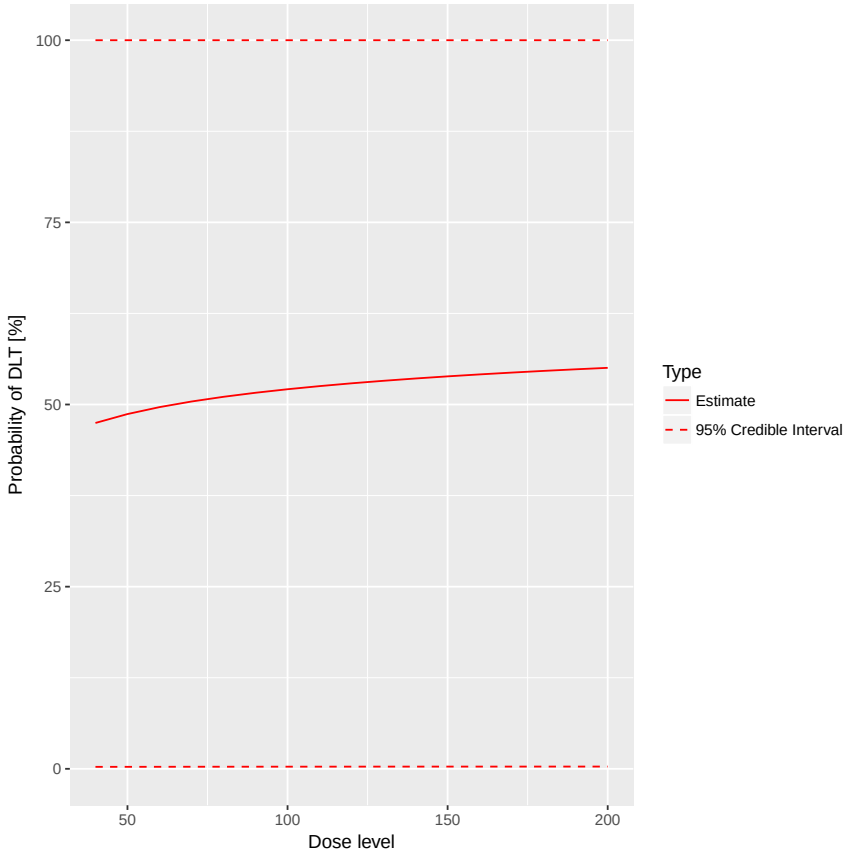
- Similar for posterior samples

```
> set.seed(92)
> postSamples <- mcmc(data, model,
+                     mcmcOptions)
```

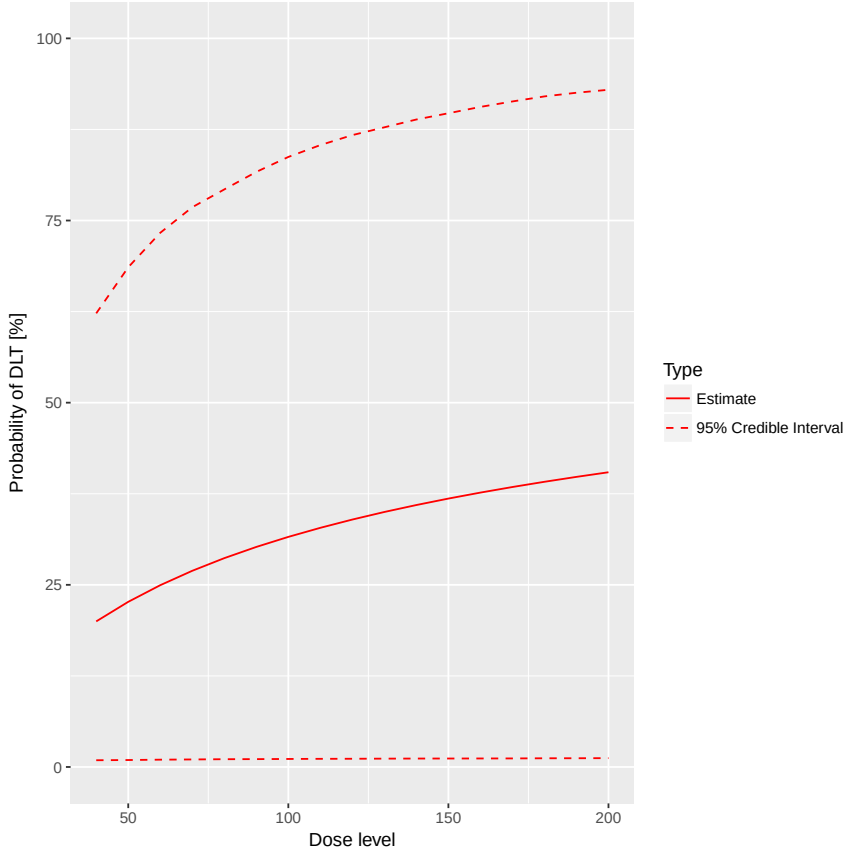
- Internally JAGS (Just another Gibbs Sampler) is used for sampling

Plotting the models

```
> plot(priorSamples,
+ model, emptyData)
```




```
> plot(postSamples,
+ model, data)
```



Increments object

Specification

Increments

- Define that dose can increase by 100% at most

```
> increments <- IncrementsRelative(interval=0,
+                                  increments=1)
```

  - `interval` specifies the range of doses that applies to
- The increments rule is used by `maxDose` to obtain the maximum allowable dose given the current data:

```
> maxDose(increments, data)
[1] 100
```

NextBest object

Specification

NextBest

- In order to use the CRM with target toxicity interval from 16% to 33%, and a maximum overdosing probability of 25%, we specify[1]:

```
> ncrm <- NextBestNCRM(target = c(0.16, 0.33),
+          overdose = c(0.33, 1),
+          maxOverdoseProb = 0.25)
```

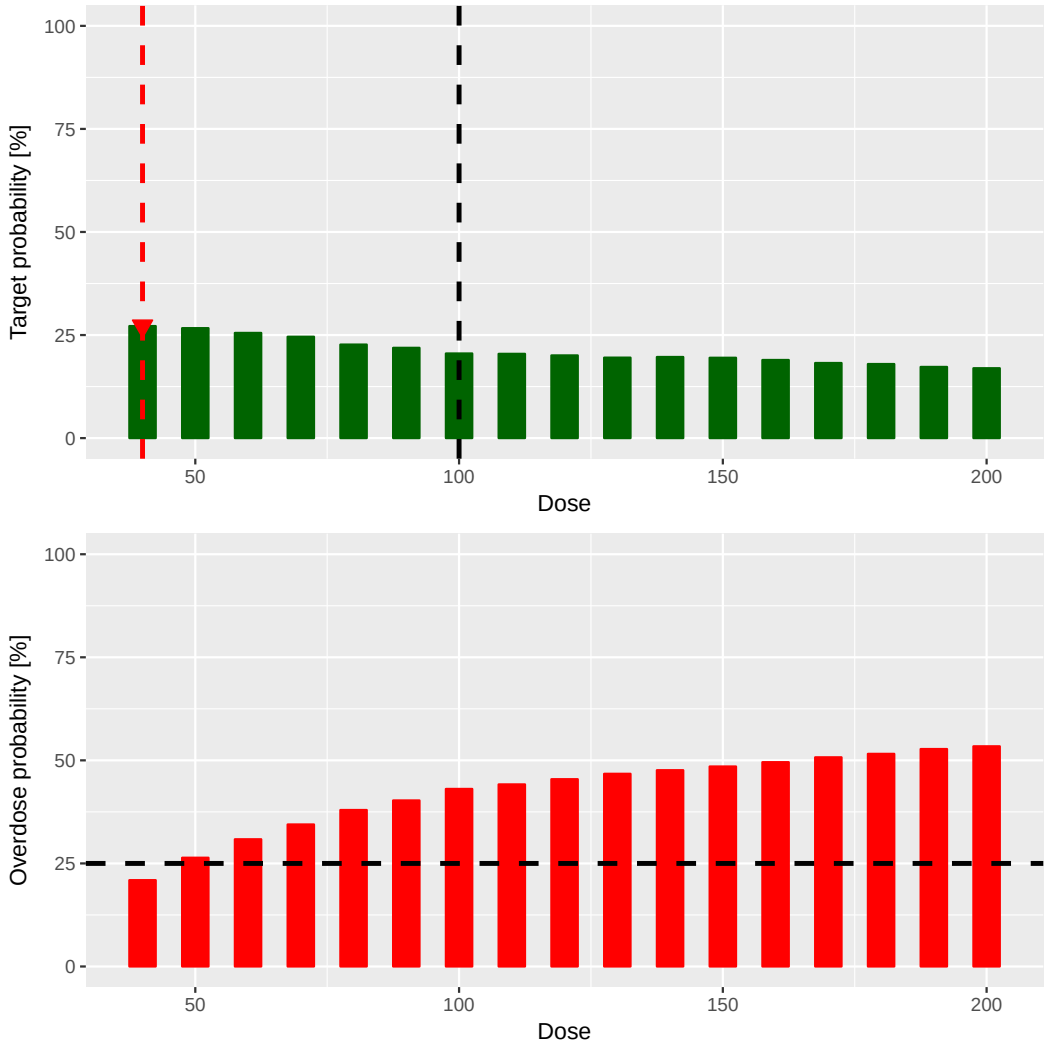
- The NextBest rule is used by the nextBest function in order to calculate the dose recommendation:

```
> doseRec <- nextBest(ncrm,
+             doselimit = maxDose(increments, data),
+             postSamples, model, data)
> doseRec$value
[1] 40
```

[1] NCRM called in the package, because Neuenschwander and colleagues first published this method.

Summary plot

```
> doseRec$plot
```



Cohort size and stopping rules

CohortSize

Always 3 patients in a cohort:

```
cohort <- CohortSizeConst(size = 3)
```

Cohort size and stopping rules

Always 3 patients in a cohort:

CohortSize

```
cohort <- CohortSizeConst(size = 3)
```

We would like to stop the trial if:

Stopping

- there are at least 2 cohorts at the MTD and
- at least 6 cohorts in total

OR the maximum sample size of 30 has been reached.

```
> stop1 <- StoppingCohortsNearDose(nCohorts = 2, percentage = 0)
# percentage specifies doses that are considered similar
> stop2 <- StoppingMinCohorts(nCohorts = 6)
> stop3 <- StoppingMinPatients(nPatients = 30)
> stopRule <- (stop1 & stop2) | stop3
```

Stopping

We can check if stopping criterion is fulfilled using `stopTrial`

```
 stopTrial(stopRule, doseRec$value, postSamples, model, data)
[1] FALSE
attr(,"message")
attr(,"message")[[1]]
attr(,"message")[[1]][[1]]
[1] "1 cohorts lie within 0% of the next best dose 40.
     This is below the required 2 cohorts"

attr(,"message")[[1]][[2]]
[1] "Number of cohorts is 2 and thus below the prespecified
     minimum number 6"

attr(,"message")[[2]]
[1] "Number of patients is 4 and thus below the prespecified minimum
     number 30"
```

Simulations

- We bundle all parts of the design in the Design object:

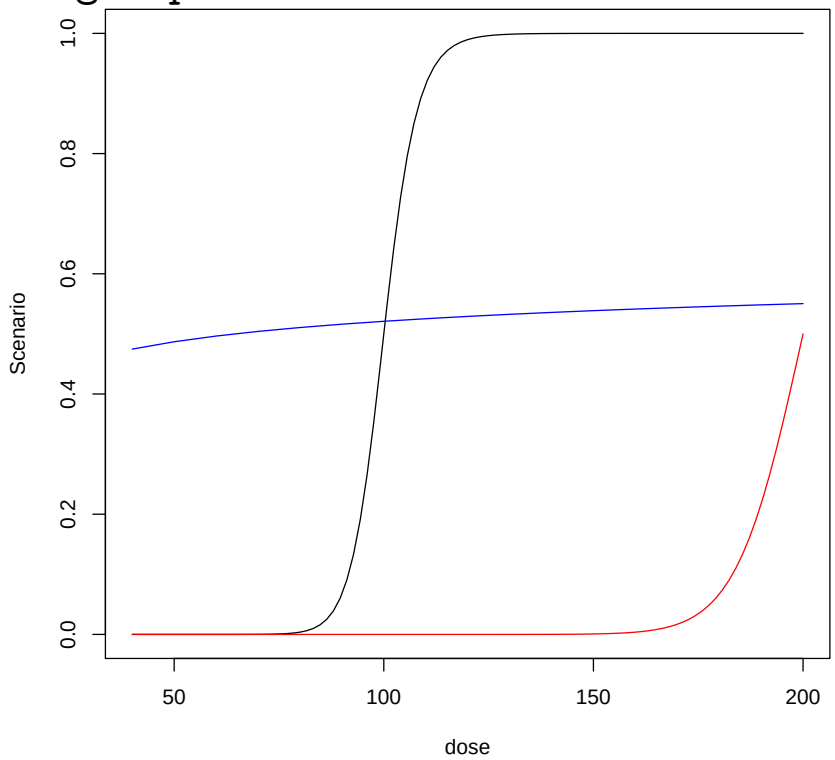
```
> design <- Design(model=model, stopping=stopRule,
+                  increments=increments,
+                  nextBest=ncrm,
+                  cohortSize=cohort,
+                  data=emptyData,
+                  startingDose=40)
```

- Scenario is given in terms of ED50 and slope:

```
> scenario <- function(dose, ED50, alpha1){
+    alpha0 <- qlogis(0.5)-alpha1*log(ED50)
+    model@prob(dose,alpha0=alpha0, alpha1 = alpha1)
+ }
```

Scenarios

```
> curve(scenario(dose, ED50=100, alpha1=25), from=40, to=200,
+    xname="dose",ylab="Scenario") # Scenario 1
> curve(scenario(dose, ED50=200, alpha1=25), from=40, to=200,
+    xname="dose", add=TRUE, col="red") # Scenario 2
> # Prior model
> lines(fit(priorSamples, model, emptyData), col="blue")
# Use fit function to get prior model fit
```



Running simulations

- Generate 10 hypothetical trials ($\sim$ 10 secs)

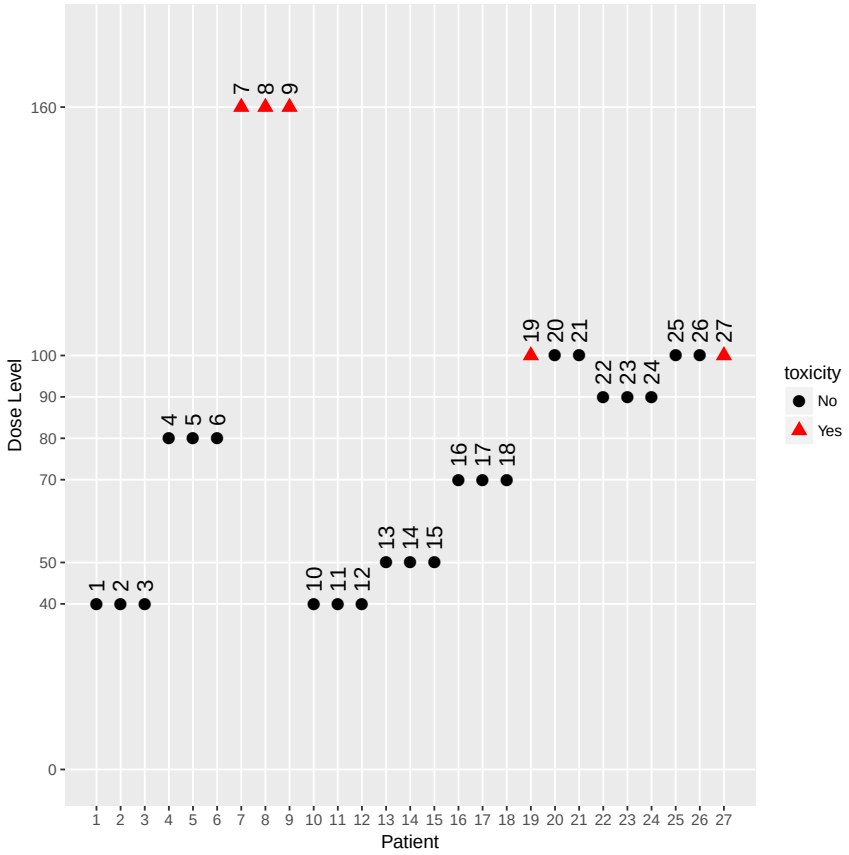
```
> sims1 <- simulate(design, nsim=10,
+     seed = 456, truth=scenario, args=list(ED50=100, alpha1=25),
+     mcmcOptions = mcmcOptions,
+     parallel = TRUE)
```

- Analogously for the second scenario:

```
> sims2 <- simulate(design, nsim=10,
+     seed = 457, truth=scenario, args=list(ED50=200, alpha1=25),
+     mcmcOptions = mcmcOptions,
+     parallel = TRUE)
```

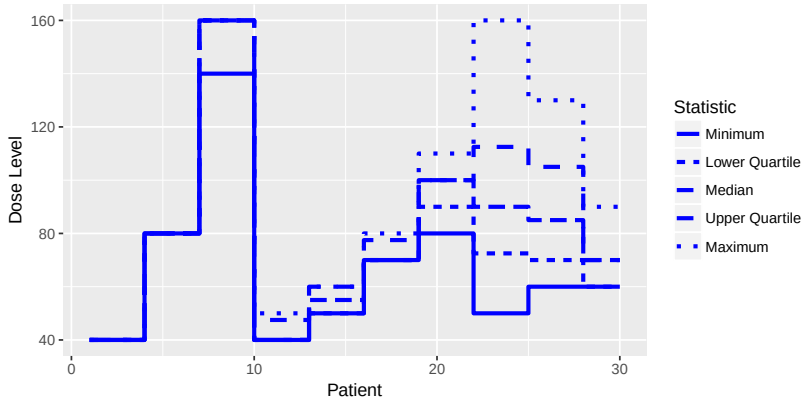
Plotting simulated trials

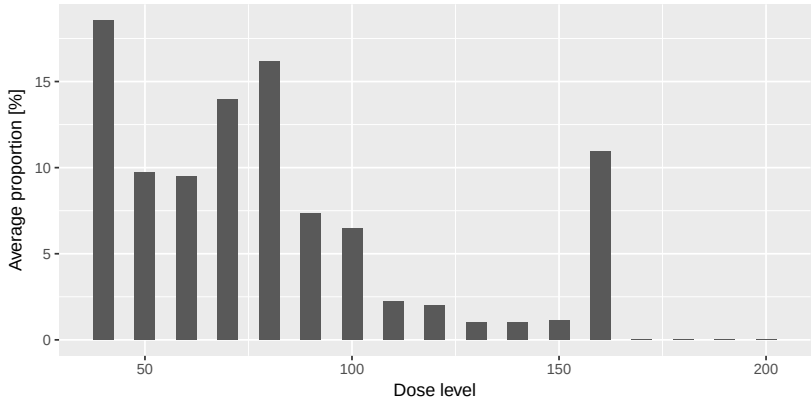
```
plot(sims1@data[[3]])

# 3rd run Scenario
```



1

```
plot(sims1)

# Summary
```



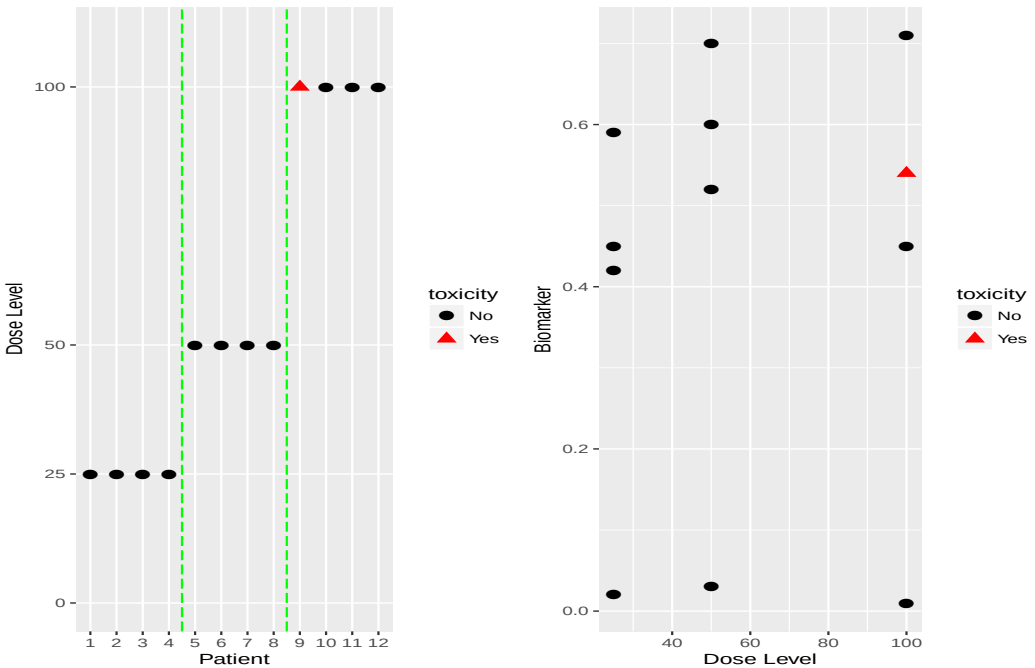


Summarizing simulations

```
> summary(sims1, truth=scenario,
+    target=ncrm@target, ED50=100, alpha1=25)
Summary of 10 simulations

Target toxicity interval was 16, 33 %
Target dose interval corresponding to this was 93.6, 97.2
Intervals are corresponding to 10 and 90 % quantiles

Number of patients overall : mean 28 (24, 30)
Number of patients treated above target tox interval : mean 7 (6, 9)
Proportions of DLTs in the trials : mean 21 % (18 %, 24 %)
Mean toxicity risks for the patients on active : mean 22 % (18 %, 26 %)
Doses selected as MTD : mean 79 (68, 100)
True toxicity at doses selected : mean 11 % (0 %, 50 %)
Proportion of trials selecting target MTD: 0 %
Dose most often selected as MTD: 70
Observed toxicity rate at dose most often selected: 0 %
Fitted toxicity rate at dose most often selected : mean 21 % (18 %, 23 %)
```

Dual Endpoints

```
> PL <- 0.001
> data2 <- DataDual(x = c(PL, 25, 25, 25, PL, 50, 50, 50,
+                         PL, 100, 100, 100),
+    y = c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0),
+    w = c(0.02, 0.42, 0.59, 0.45, 0.03, 0.7, 0.6, 0.52,
+          0.01, 0.71, 0.54, 0.45),
+    cohort = c(1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3),
+    doseGrid = c(PL, seq(25, 300, 25)), ID = 1:12, placebo = TRUE)
> plot(data2, blind=TRUE)
```



Two models and a gain function

Safety and efficacy model

```
> emptydata2 <- DataDual(doseGrid = data2@doseGrid, placebo = TRUE)
> DLTmodel <- LogisticIndepBeta(binDLE = c(1.05, 1.8),
+              DLEweights = c(3, 3), DLEdose = c(25, 300),
+              data = emptydata2)
> Effmodel <- Effloglog(Eff = c(1.223, 2.513), Effdose = c(25, 300),
+              nu = c(a = 1, b = 0.025), data = emptydata2, c = 2)
```

Gain function depends on safety and effiacy parameters which will be estimated using posterior modal estimates

```
> newDLTmodel <- update(object = DLTmodel, data = data2)
> newEffmodel <- update(object = Effmodel, data = data2)
> GainNextBest <- NextBestMaxGain(DLEDuringTrialtarget = 0.35,
+                 DLEEndOfTrialtarget = 0.3)
```

Next dose

- A more advanced increment rule

```
> myIncrements <- IncrementsRelative(intervals = c(0, 125, 200),
+                                    increments = c(1, 0.75, 0.5))
```

  A maximum increase of 100% for doses below 125 mg, 75% for doses in the range from 125 mg to 200 mg, and 50% for doses equal or above 200 mg

- Identify maximum next dose

```
> nextMaxDose <- maxDose(myIncrements, data2)
> nextMaxDose
[1] 200
```

Next dose

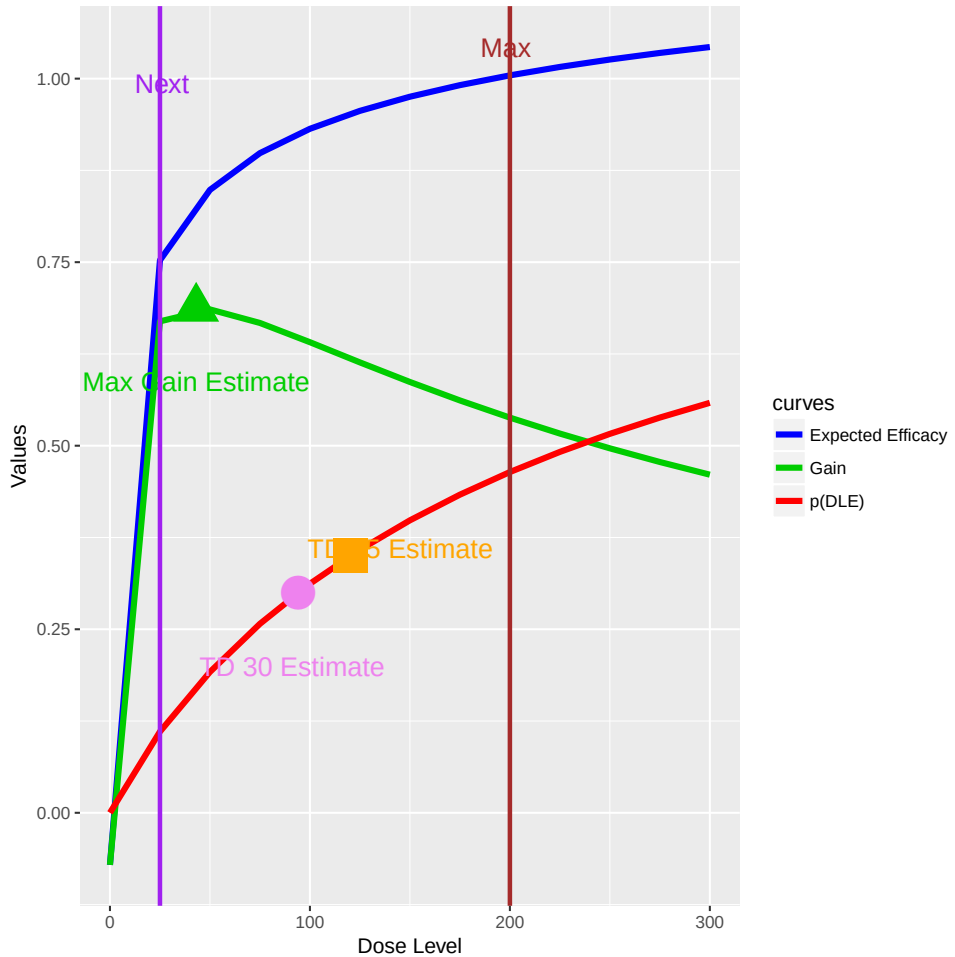
Finding the next dose

```
> doseRecGain <- nextBest(GainNextBest, doselimit = nextMaxDose,
+                 model = newDLTmodel, Effmodel = newEffmodel,
+                 data = data2)
> nextDoseVal <- doseRecGain$nextdose
> nextDoseVal
[1] 25
```

- After new data are observed `newDLTmodel` and `newEffmodel` need to be updated
- No MCMC used here

Plotting the decision

```
> doseRecGain$plot
```



Stopping

- Stop when maximum number of patients are observed
- The optimal dose can be estimated precisely enough

```
> stop4 <- StoppingGstarCIRatio(targetRatio = 5,
+           targetEndOfTrial = GainNextBest@DLEEndOfTrialtarget)
> myStoppingDual <- stop3 | stop4
```

Simulations

- Define design object

```
> design2 <- DualResponsesDesign(nextBest = GainNextBest,
+             model = DLTmodel, Effmodel = Effmodel,
+             data = emptydata2,
+             stopping = myStoppingDual,
+             increments = myIncrements,
+             cohortSize = cohort, startingDose = 25)
```

- Define scenarios

```
> trueDLT<- function(dose){DLTmodel@prob(dose,
+                           phi1 = -53, phi2 = 10)}
> trueEff<- function(dose){Effmodel@ExpEff(dose,
+                           theta1 = -4.8, theta2 = 3.7)}
> trueGain <- function(dose){trueEff(dose) * (1 - trueDLT(dose))}
```

Simulations

- Simulate

```
> sims3 <- simulate(object = design2, args = NULL,
+                   trueDLE = trueDLT, trueEff = trueEff,
+                   trueNu = 1 / 0.025, nsim = 100,
+                   seed = 19, parallel = TRUE)
```

- Summarize and plot

```
> mean(sims3@FinalOptimalDose)
[1] 149.0867
> summary(sims3,trueDLE=trueDLT,trueEff=trueEff)
> plot(sims3@data[[90]])
> plot(sims3)
```

Simulation plots