Detailed Analysis

Lisa Hu, Niek Scholten

Data loading

```r
## Read the data
MPL_data <- read.csv("MPL.csv", na.strings = "NA")
## Rename the receptor
names(MPL_data)[4:5] <- c("R.m0", "R.0")

## Create the medians subsets
medians <- aggregate(MPL_data[,c("MPL_conc","R.m0","R.0")],
                     list(MPL_data$dose, MPL_data$time), median, na.rm=T)
names(medians)[1:2] <- c("dose","time")
```

Assignment 1

Implement the model

Following experiments with methylprednisolone in rats, a set of values for the model is defined:

Table 1: Parameter values for MPL

Parameter	Value	Unit	Explanation
k_{s_Rm}	2.90	fmol/g	Zero-order rate constant: GR mRNA synthesis
IC_{50_Rm}	26.2	fmol/mg	Concentration DR(N) that inhibits mRNA synthesis (50%)
k_{on}	0.00329	L/nmol/h	Second-order rate constant: forming MPL-receptor complex
k_T	0.63	1/h	First-order rate constant: translocation of receptor (cytosol -> nucleus)
k~re	0.57	1/h	First-order rate constant: recovery of receptor (nucleus -> cytosol)
R_f	0.49	-	Fraction free receptor that gets recycled
k_{d_R}	0.0572	1/h	First-order rate constant: breakdown of receptor
k_{d_Rm}	0.612	-	First-order rate constant: breakdown of GR mRNA
k_{s_R}	3.22	-	First-order rate constant: production of receptor
D	~	nmol/L	Concentration MPL (Calculated with molecular weight = 374.471

Table 2: Initial values for MPL

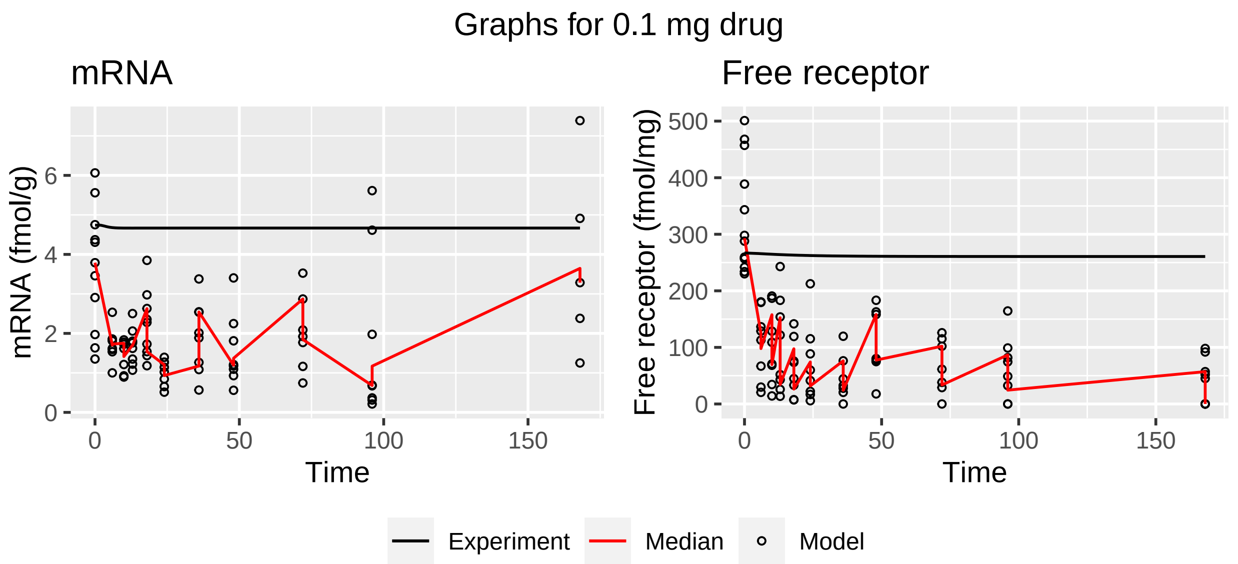
Parameter	Value	Unit	Explanation
R_{m0} (mRNA)	4.74	fmol/g	Concentration $mRNA_R$
R_0 (Free_receptor)	267	fmol/mg	Concentration free receptors
DR	0	fmol/mg	MPL in the cytosol
DR(N)	0	fmol/mg	MPL in the nucleus

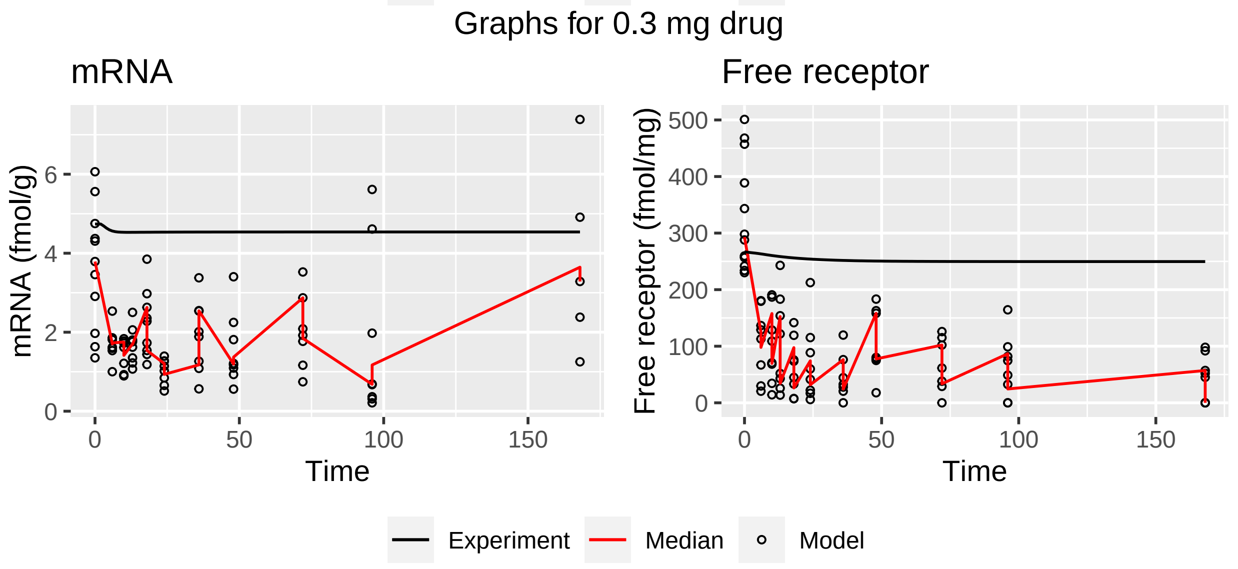
To create the plots for the model, functions can be used:

```r
## Model
grd_model <- function(t, y, parms){
  with(as.list(c(parms, y)),{
    delta.R.m0 <- k.s_Rm * (1 - ( DRN / (IC.50_Rm + DRN) ) ) - k.d_Rm * R.m0
    delta.R.0 <- k.s_R * R.m0 + R.f * k.re * DRN - k.on * D * R.0 - k.d_R * R.0
    delta.DR <- k.on * D * R.0 - k.T * DR
    delta.DRN <- k.T * DR - k.re * DRN
    return( list( c(delta.R.m0, delta.R.0, delta.DR, delta.DRN ) ) )
  })
}

## Data frame with the different units and titles of the data
data.values <- data.frame(
        name = c("R.m0", "R.0", "DR", "DRN"),
        ylabel = c("mRNA", "Free receptor", "Receptor complex", "Receptor complex"),
        unit = c("(fmol/g)", "(fmol/mg)", "(fmol/mg)", "(fmol/mg)"),
        title = c("mRNA", "Free receptor", "Receptor complex (cytosol)",
                   "Receptor complex (nucleus)") )
rownames(data.values) <- data.values$name

## Function that calculates D
calc.D <- function(dose){
  return( (dose * 1000)/374.471 )
}

## Function to create a plot depending on wanted data
create.plot1 <- function(plot.value, simulation, median.data, model.data){
  # y.val inserts the plot.value for the corresponding row of data.values
  y.val <- data.values[plot.value,]
  # The plot
  plt <- ggplot(data = simulation, mapping = aes(x = time, y = !!sym(y.val$name) ) ) +
            # Points and lines
            geom_point(size = 1, shape = 1, aes(color = "Experiment") ) +
            geom_line(data = median.data, aes(color = "Median")) +
            geom_line(data = model.data, aes(color = "Model")) +
            # Labels
            labs(x = "Time", y = paste(y.val$ylabel, y.val$unit), title = y.val$title) +
            theme(legend.position = "bottom") +
            # Legend data
            scale_colour_manual(values = c("black", "red", "black"),
                                limits = c("Experiment", "Median", "Model") ) +
            # Legend correction
            guides(color = guide_legend(title = "",
                                        override.aes = list(linetype = c(1, 1, NA),
                                                            shape = c(NA, NA, 1)) ) )
  return(plt)
}

## Function arranges plots and legend
arrange.plots <- function(plots, title){
  my.grid <- ggarrange(plotlist = plots, ncol = 2, nrow = length(plots)/2,
                       common.legend = TRUE, legend = "bottom")
  print( annotate_figure(my.grid, top = text_grob(title) ) )
}

## Function joins other functions with data
main <- function(dose){
  # Change D depending on dose
  params$D <- calc.D(dose)
  # The different datas
  simulation <- subset(MPL_data, MPL_data$dose == 0 | MPL_data$dose == dose)
  median.data <- subset(medians, medians$dose == 0 | medians$dose == dose)
  # Run the model
  model.data <- as.data.frame( ode(times = times, y = state, parms = params,
                                   func = grd_model, method = "euler"))
  # Create the plots
  plots <- lapply(c("R.m0", "R.0"), create.plot1, simulation, median.data, model.data)
  # Arrange the plots
  arrange.plots(plots, sprintf("Graphs for %s mg drug", dose))
}
```

The experiment followed rats for 7 days under constant infusion of the drug: 0.1 or 0.3 $\mathrm{mg_{drug}/kg_{rat}/h}$. With this information, models can be created:

```r
## Set the data for the model
params <- c(k.s_Rm = 2.90, IC.50_Rm = 26.2, k.on = 0.00329, k.T = 0.63, k.re = 0.57,
            R.f = 0.49, k.d_R = 0.0572, k.d_Rm = 0.612, k.s_R = 3.22, D = 0)

state <- c(R.m0 = 4.74, R.0 = 267, DR = 0, DRN = 0)

times <- seq(0, 168, by = 1)

## Check the doses
doses <- unique(MPL_data$dose)
doses <- doses[!doses == 0]
sapply(doses, main)
```





Questions

- [1] Why is it best practice to plot the median for the experimental data?

  The median does not change with huge outliers, so the data is more reliable. If there is a significant difference, it will show.

- [2] How do the results of the simulations depend on the dose and concentration of the drug?

  The dose can influence the steady state of the different concentrations. The shape of the median lines is generally unaffected.

- [3] Are the results of the model in line with experimental data?

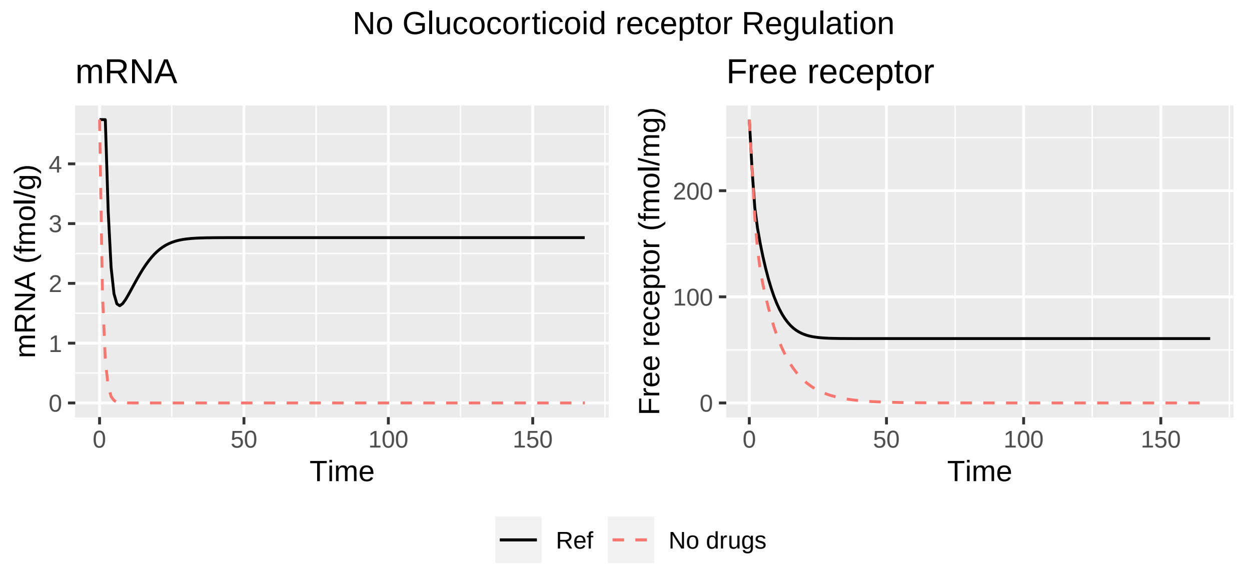
  Yes, to some extent. The values that the experiment and the model end on is similar, but the values between the start and the end fluctuate a bit. The graphs of the free receptor concentration show that the experimental data conform with the model. In the mRNA graphs, there are some deviations from the model around Time = 25.

Assignment 2

No Auto-Regulation Glucocorticoid Receptor

```r
## Create function for no drugs
receptor.model <- function(t, y, parms){
  with(as.list(c(parms, y)),{
    delta.R.m0 <- -k.d_Rm * R.m0
    delta.R <- k.s_R * R.m0 + R.f * k.re * DRN - k.on * D * R.0 - k.d_R * R.0
    delta.DR <- k.on * D * R.0 - k.T * DR
    delta.DRN <- k.T * DR - k.re * DRN
    return( list( c(delta.R.m0, delta.R, delta.DR, delta.DRN ) ) )
  })
}

changing.plots <- function(plot.value, ref.data, changed.data){
  # Create colours for the different lines (except for reference)
  colours <- hue_pal()(length(changed.data))
  data.names <- names(changed.data)
  # y.val inserts the plot.value for the corresponding row of data.values
  y.val <- data.values[plot.value,]
  # The plot
  plt <- ggplot(data = ref.data, mapping = aes(x = time, y = !!sym(y.val$name) ) ) +
           # Lines (reference data stays black)
           geom_line(aes(color = "Ref")) +
           unlist( mapply(function(single.data, data.name)
                      geom_line(data = single.data, linetype = "dashed",
                                aes(color = data.name ) ),
                  changed.data, data.names ) ) +
           # Labels
           labs(x = "Time", y = paste(y.val$ylabel, y.val$unit), title = y.val$title) +
           theme(legend.position = "bottom") +
           # Legend data
           scale_colour_manual(values = c("black", colours),
                               limits = c("Ref", names(changed.data) ) ) +
           # Legend correction
           guides(color = guide_legend(
                   title = "",
                   override.aes = list(linetype = c(1, rep(2, length(changed.data)))))
           )
  return(plt)
}

## The model with a dose of 20
params$D <- calc.D(20)
model_20 <- ode(times = times, y = state,
                 parms = params, func = grd_model, method = "euler")
model_20 <- as.data.frame(model_20)

## The model without drug influence
model_noDrugs <- as.data.frame(ode(times = times, y = state,
                 parms = params, func = receptor.model, method = "euler"))

data.list <- list("No drugs" = model_noDrugs)
```

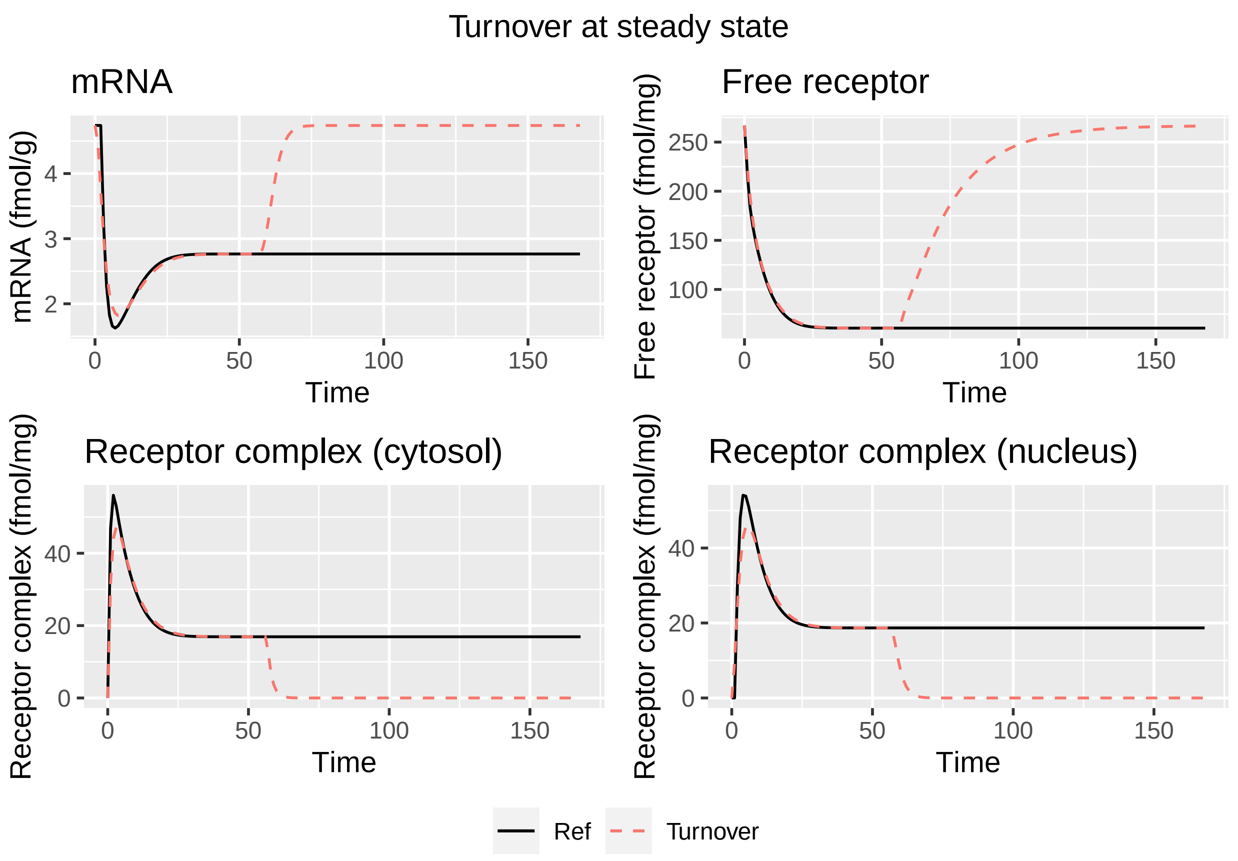
```
plots <- lapply(c("R.m0", "R.0"), changing.plots, model_20, data.list)
arrange.plots(plots, "No Glucocorticoid receptor Regulation")
```



When the DR(N) has no influence on the mRNA$_R$ synthesis, a big part of the first equation disappears: $k_{s_Rm} \cdot (1 - \frac{DR(N)}{IC_{50_Rm}+DR(N)})$ The figures show, when DR(N) has no influence on the mRNA$_R$ synthesis, the mRNA$_R$ concentration drops to zero fairly quickly and stays zero due to no mRNA$_R$ synthesis. The blue line details the normal scenario in which the drug is working correctly with a dose of 20.

Stopping drug treatment at Steady State

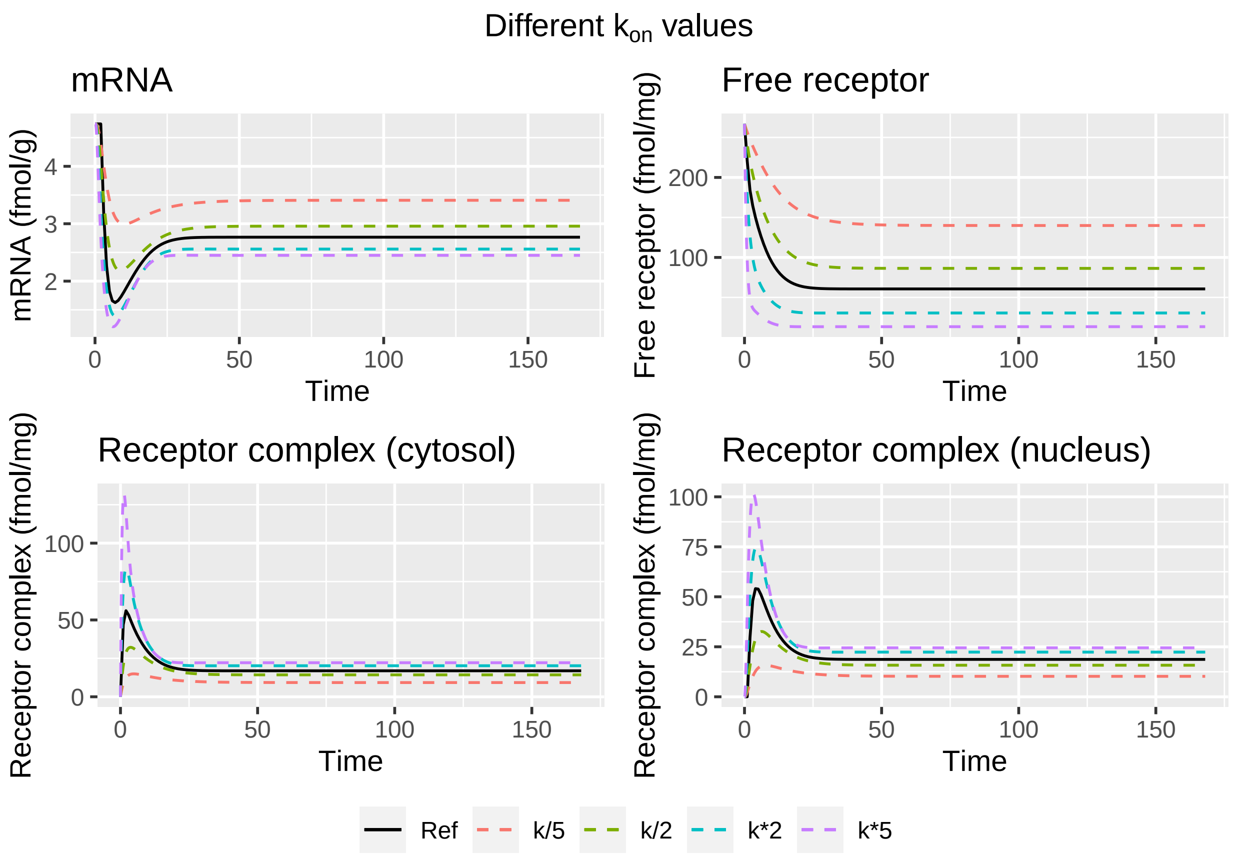
```r
## Define new model
steady_model <- function(t, y, parms){
  with(as.list(c(parms, y)),{
    delta.R.m0 <- k.s_Rm * (1 - ( DRN / (IC.50_Rm + DRN) ) ) - k.d_Rm * R.m0
    delta.R.0 <- k.s_R * R.m0 + R.f * k.re * DRN - k.on * D * R.0 - k.d_R * R.0
    delta.DR <- k.on * D * R.0 - k.T * DR
    delta.DRN <- k.T * DR - k.re * DRN
    delta.D <- 0
    return( list( c(delta.R.m0, delta.R.0, delta.DR, delta.DRN, delta.D) ) )
  })
}

params_steady <- c(k.s_Rm = 2.90, IC.50_Rm = 26.2, k.on = 0.00329, k.T = 0.63,
                   k.re = 0.57, R.f = 0.49, k.d_R = 0.0572, k.d_Rm = 0.612, k.s_R = 3.22)

state_steady <- c(R.m0 = 4.74, R.0 = 267, DR = 0, DRN = 0, D = calc.D(20))

## Create the event for the model
trigger <- function(t, y, params){
  y["D"] <- 0
  return(y)
}

## Check when the steady state is found
root <- function (t, y, params){
  x <- unlist(grd_model(t, y, params))
  numb1 <- sum(abs(x)) - 1e-4
  numb2 <- numb1 + y["D"]

  return(c(numb1, numb2))
}

## Run the model with the turnover
steadys <- ode( times = times, y = state_steady, parms = params_steady,
                func = steady_model, rootfun = root,
                events = list(func = trigger, root = TRUE, terminalroot = 2) )

last.root <- attributes(steadys)$troot
steadys <- as.data.frame(steadys)

data.list <- list("Turnover" = steadys)
plots <- lapply(c("R.m0", "R.0", "DR", "DRN"), changing.plots, model_20,data.list)
arrange.plots(plots, expression("Turnover at steady state"))
```



Turnover at steady state
mRNA
mRNA (fmol/g)
Free receptor
Free receptor (fmol/mg)
Receptor complex (cytosol)
Receptor complex (fmol/mg)
Receptor complex (nucleus)
Receptor complex (fmol/mg)
Time
Ref
Turnover

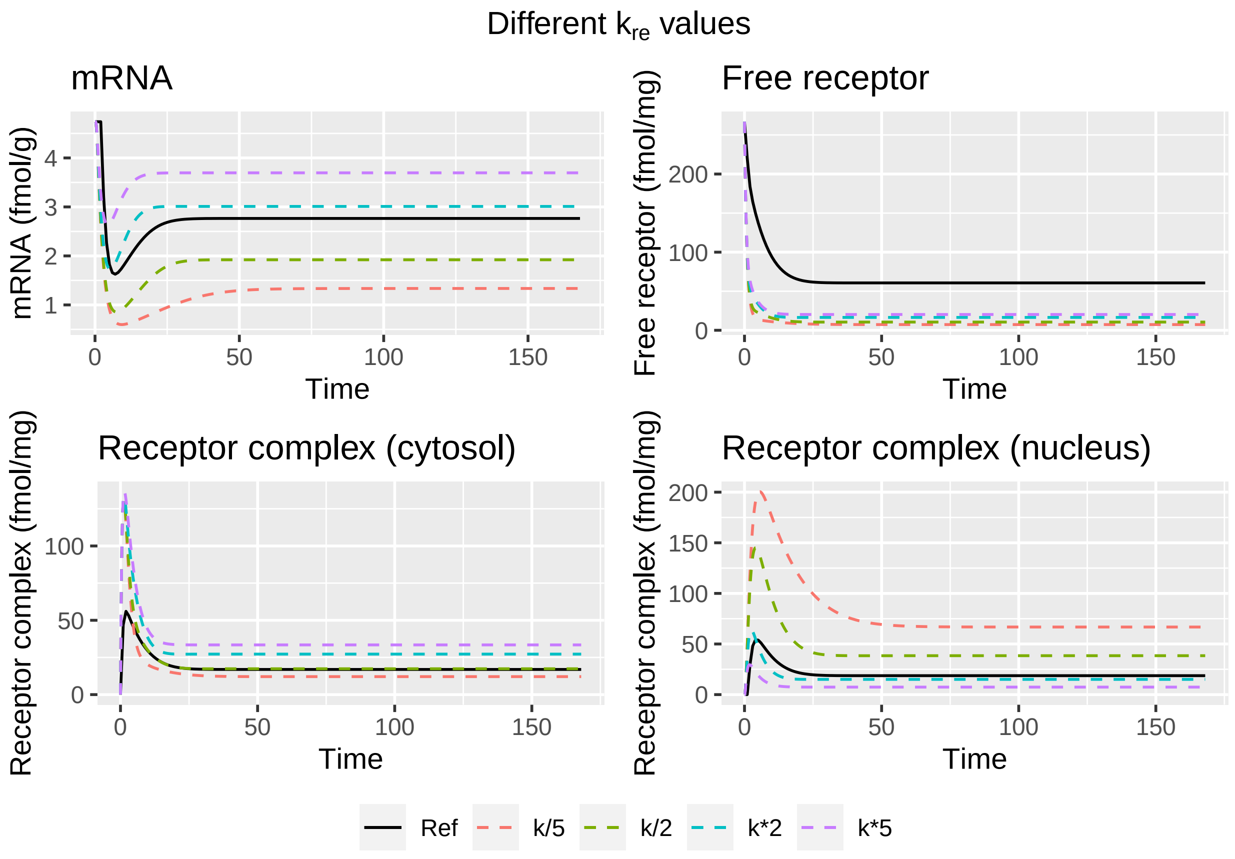
Changes in k_{on} and k_{re}

```r
times <- seq(0, 168, by = 0.1)

state <- c(R.m0 = 4.74, R.0 = 267, DR = 0, DRN = 0)

params <- c(k.s_Rm = 2.90, IC.50_Rm = 26.2, k.on = 0.00329, k.T = 0.63, k.re = 0.57,
            R.f = 0.49, k.d_R = 0.0572, k.d_Rm = 0.612, k.s_R = 3.22, D = calc.D(20))

k.on_models <- list("k/5" = 0.0039/5,
                    "k/2" = 0.0039/2,
                    "k*2" = 0.0039*2,
                    "k*5" = 0.0039*5)

for (i in seq_along(k.on_models)){
  params$k.on <- k.on_models[[i]]
  k.on_models[[i]] <- as.data.frame(ode(times = times, y = state, parms = params,
                                        func = grd_model, method = "euler"))
}

plots <- lapply(c("R.m0", "R.0", "DR", "DRN"), changing.plots, model_20, k.on_models)
arrange.plots(plots, expression("Different k"["on"]*" values"))
```



```r
k.re_models <- list("k/5" = 0.57/5,
                    "k/2" = 0.57/2,
                    "k*2" = 0.57*2,
                    "k*5" = 0.57*5)
```

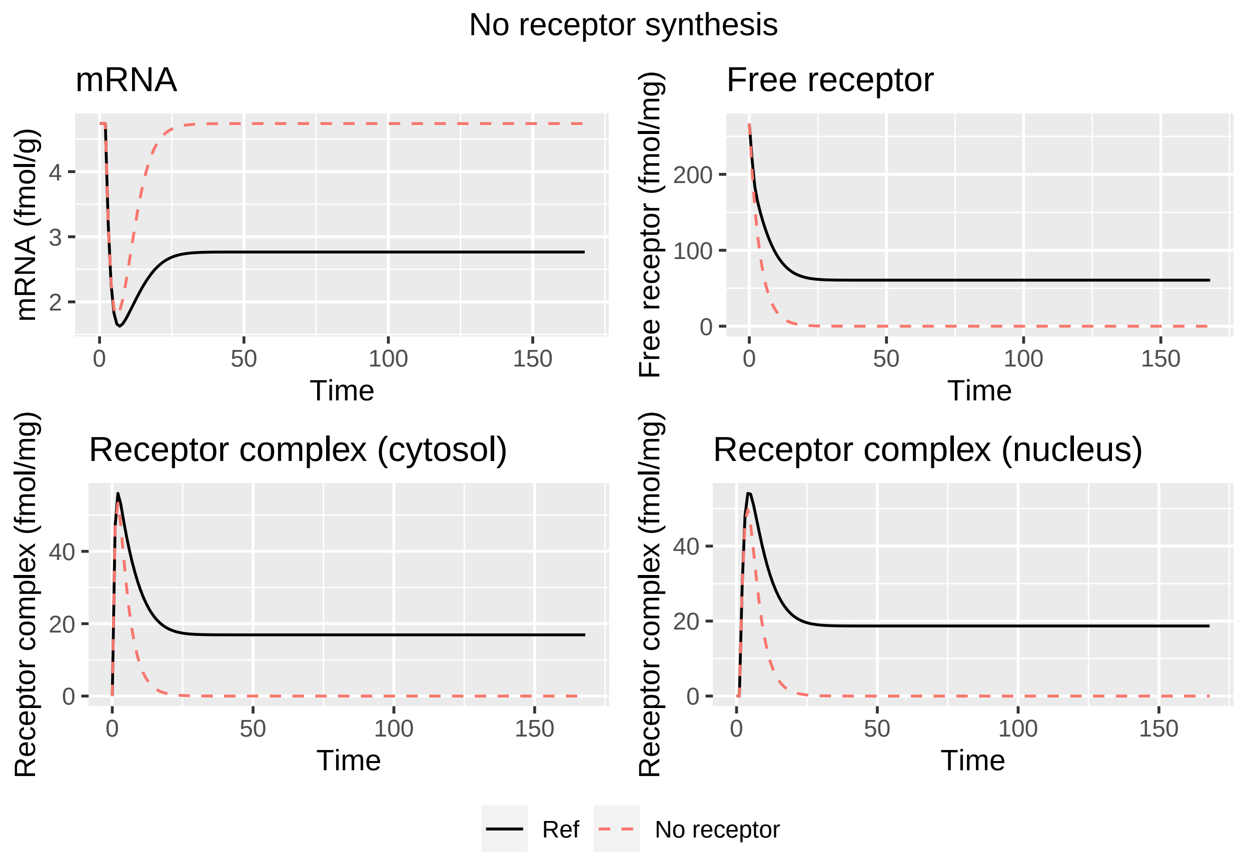
```r
for (i in seq_along(k.re_models)){
  params$k.re <- k.re_models[[i]]
  k.re_models[[i]] <- as.data.frame(ode(times = times, y = state, parms = params,
                                        func = grd_model, method = "euler"))
}

plots <- lapply(c("R.m0", "R.0", "DR", "DRN"), changing.plots, model_20, k.re_models)
arrange.plots(plots, expression("Different k"["re"]*" values"))
```



Block Synthesis of Receptor

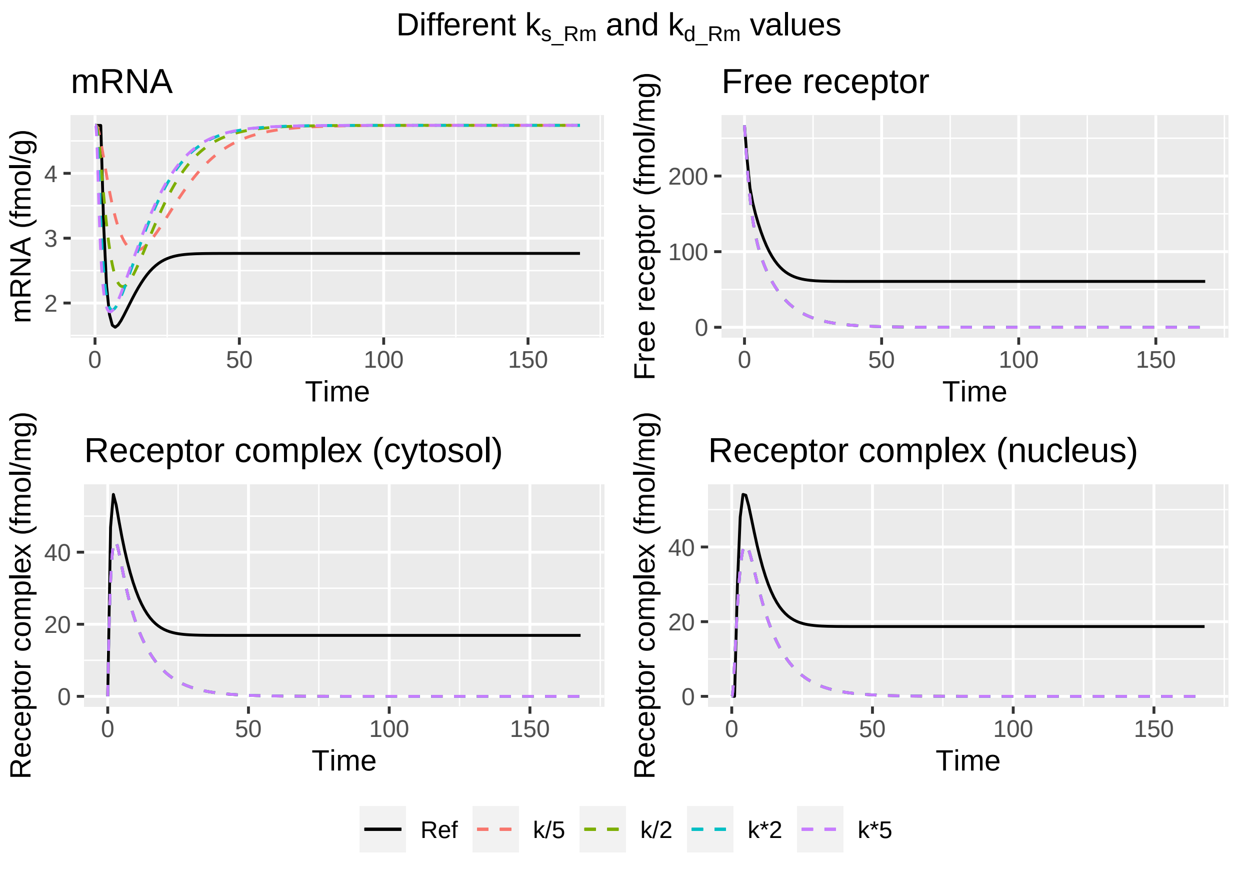
To simulate this situation, k_{s_R} needs to be put to zero and R_f and k_{re} should output no influence on R. This changes the model a bit:

```
block_model <- function(t, y, parms){
  with(as.list(c(parms, y)),{
    delta.R.m0 <- k.s_Rm * (1 - ( DRN / (IC.50_Rm + DRN) ) ) - k.d_Rm * R.m0
    delta.R.0 <- k.s_R * R.m0 - k.on * D * R.0 - k.d_R * R.0
    delta.DR <- k.on * D * R.0 - k.T * DR
    delta.DRN <- k.T * DR - k.re * DRN
    return( list( c(delta.R.m0, delta.R.0, delta.DR, delta.DRN ) ) )
  })
}

params <- c(k.s_Rm = 2.90, IC.50_Rm = 26.2, k.on = 0.00329, k.T = 0.63, k.re = 0.57,
            R.f = 0.49, k.d_R = 0.0572, k.d_Rm = 0.612, k.s_R = 0, D = calc.D(20))
state <- c(R.m0 = 4.74, R.0 = 267, DR = 0, DRN = 0)
times <- seq(0, 168, by = 1)

no.Receptor <- as.data.frame(ode(times = times, y = state, parms = params,
                                 func = block_model, method = "euler"))

data.list <- list("No receptor" = no.Receptor)
plots <- lapply(c("R.m0", "R.0", "DR", "DRN"), changing.plots, model_20, data.list)
arrange.plots(plots, expression("No receptor synthesis"))
```



Increased and decreased baseline mRNA production

```
params <- c(k.s_Rm = 2.90, IC.50_Rm = 26.2, k.on = 0.00329, k.T = 0.63, k.re = 0.57,
            R.f = 0.49, k.d_R = 0.0572, k.d_Rm = 0.612, k.s_R = 0, D = calc.D(20))
state <- c(R.m0 = 4.74, R.0 = 267, DR = 0, DRN = 0)
times <- seq(0, 168, by = 0.1)

k.models <- list("k/5" = 2.9/5,
                 "k/2" = 2.9/2,
                 "k*2" = 2.9*2,
                 "k*5" = 2.9*5)

for (i in seq_along(k.models)){
  params$k.s_Rm <- k.models[[i]]
  params$k.d_Rm <- k.models[[i]]/4.74
  k.models[[i]] <- as.data.frame(ode(times = times, y = state, parms = params,
                                     func = grd_model, method = "euler"))
}

plots <- lapply(c("R.m0", "R.0", "DR", "DRN"), changing.plots, model_20, k.models)
arrange.plots(plots, expression("Different k"["s_Rm"]*" and k"["d_Rm"]*" values"))
```


Different k_{s_Rm} and k_{d_Rm} values
mRNA
mRNA (fmol/g)
Free receptor
Free receptor (fmol/mg)
Receptor complex (cytosol)
Receptor complex (fmol/mg)
Receptor complex (nucleus)
Time
Ref
k/5
k/2
k*2
k*5