

```

##### Mutual repression #####
##### hysteresis switch #####
simTime <- 100
maxIn <- 40
myModel <- new("odeModel",
main = function (time, init, parms, ...) {
  x      <- init
  p      <- parms
  U1     <- approxTime1(inputs, time, rule = 2)["A.in"]
  U2     <- 20 # arbitrarily set to ~ maxIn/2      ### U2=constant driver of G2
  dx1 <- p["kt1"] * (U1/(x[2]^2+p["KD1"]^2)) - p["kd1"] * x[1]
  dx2 <- p["kt2"] * (U2/(x[1]^2+p["KD2"]^2)) - p["kd2"] * x[2]
  list(c(dx1, dx2))
},
parms = c(kt1=1, kd1=1, KD1=1, kt2=1, kd2=1, KD2=1),
times = c(from=0, to=simTime, by=0.5),
solver = "rk4"                                     ### "lsoda"
)

nSims      <- 101
tOff       <- 30
results1   <- matrix(1:nSims, nrow=nSims, ncol=3)
for (i in 1:nSims) {
  initVal <- 1                                     ### ((i-1)/(nSims-1))
  inputVal <- maxIn*((i-1)/(nSims-1))             ### 1
  init(myModel) <- c("Gene1"=initVal, "Gene2"=0)
  inputs(myModel) <- as.matrix(data.frame(time = c(0, tOff, (tOff+0.1), simTime),
                                           A.in = c(maxIn, maxIn,
inputVal, inputVal)))
  myModel <- sim(myModel)
  lastG1Value <- out(myModel)[dim(out(myModel))[1],2]
  lastG2Value <- out(myModel)[dim(out(myModel))[1],3]
  results1[i, ] <- c(input=inputVal, G1=lastG1Value, G2=lastG2Value)
}

windows(height=4,width=6)
plot.new()
plot(results1[, 1], results1[, 2], xlab="input amplitude", ylab="[G1]", col="gray70")
lines(results1[, 1], results1[, 2], col="red")

nSims      <- 101
tOff       <- 30
results2   <- matrix(1:nSims, nrow=nSims, ncol=3)
for (i in 1:nSims) {
  inputVal <- maxIn*((i-1)/(nSims-1))             ### 1
  init(myModel) <- c("Gene1"=0, "Gene2"=initVal)
  inputs(myModel) <- as.matrix(data.frame(time = c(0, tOff, (tOff+0.1), simTime),
                                           A.in = c(inputVal, inputVal,
inputVal, inputVal)))
  myModel <- sim(myModel)
  lastG1Value <- out(myModel)[dim(out(myModel))[1],2]

```

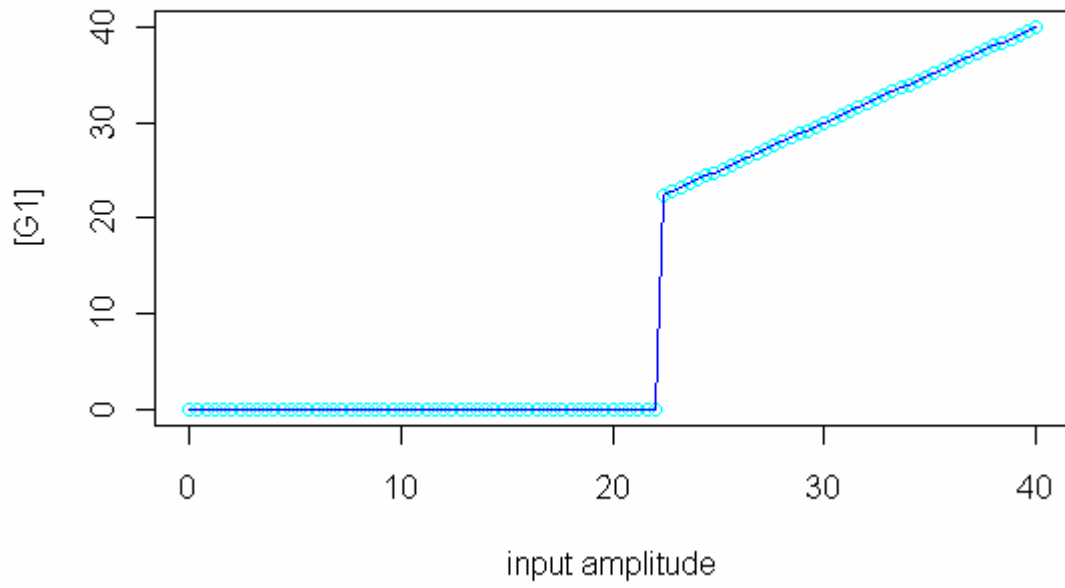
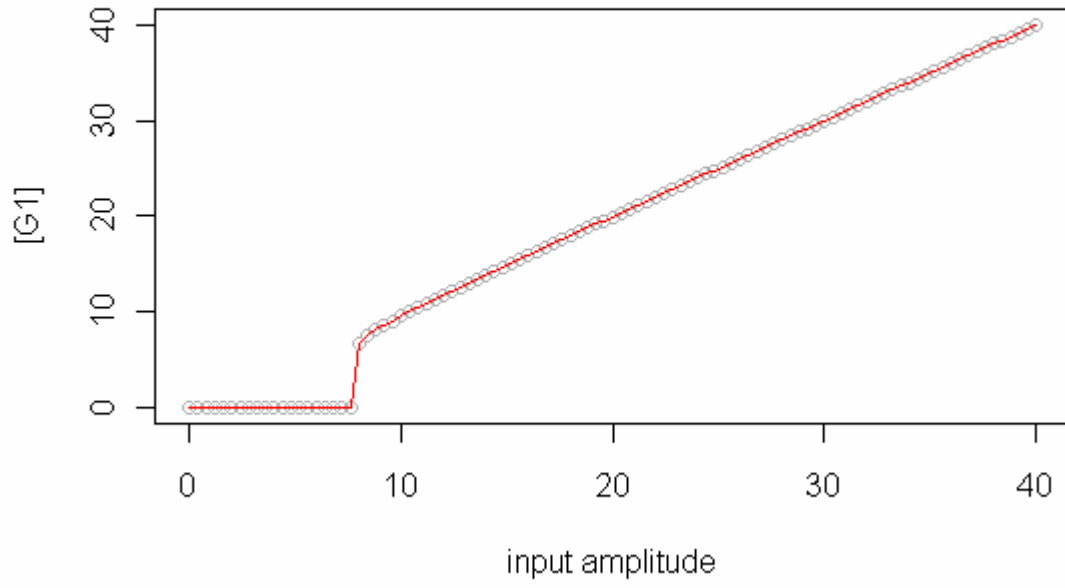
```
lastG2Value <- out(myModel)[dim(out(myModel))[1],3]
results2[j, ] <- c(input=inputVal, G1=lastG1Value, G2=lastG2Value)
}
```

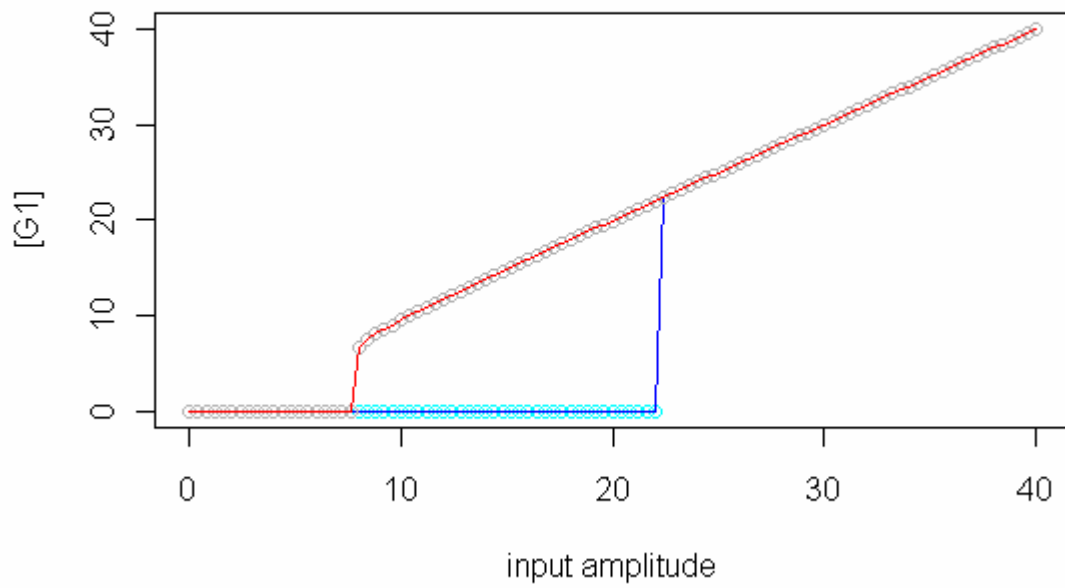
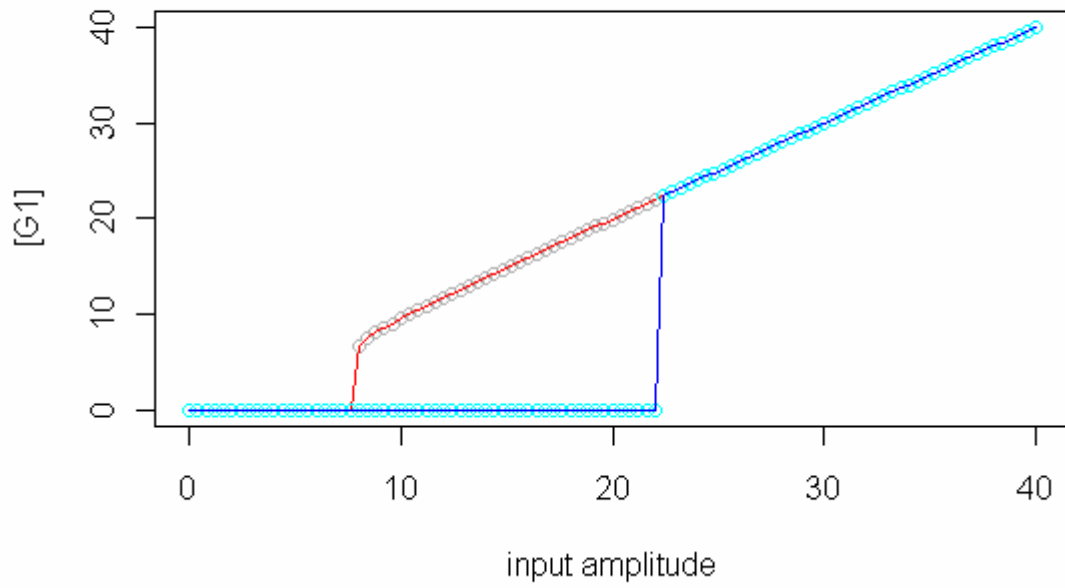
```
windows(height=4,width=6)
plot.new()
plot(results2[,1], results2[, 2], xlab="input amplitude", ylab="[G1]", col="cyan")
lines(results2[,1], results2[, 2], col="blue")
```

```
windows(height=4,width=6)
plot.new()
plot(results2[,1], results2[, 2], xlab="input amplitude", ylab="[G1]", col="cyan")
lines(results2[,1], results2[, 2], type="l", col="blue")
lines(results1[,1], results1[, 2], type="p", col="gray70")
lines(results1[,1], results1[, 2], type="l", col="red")
```

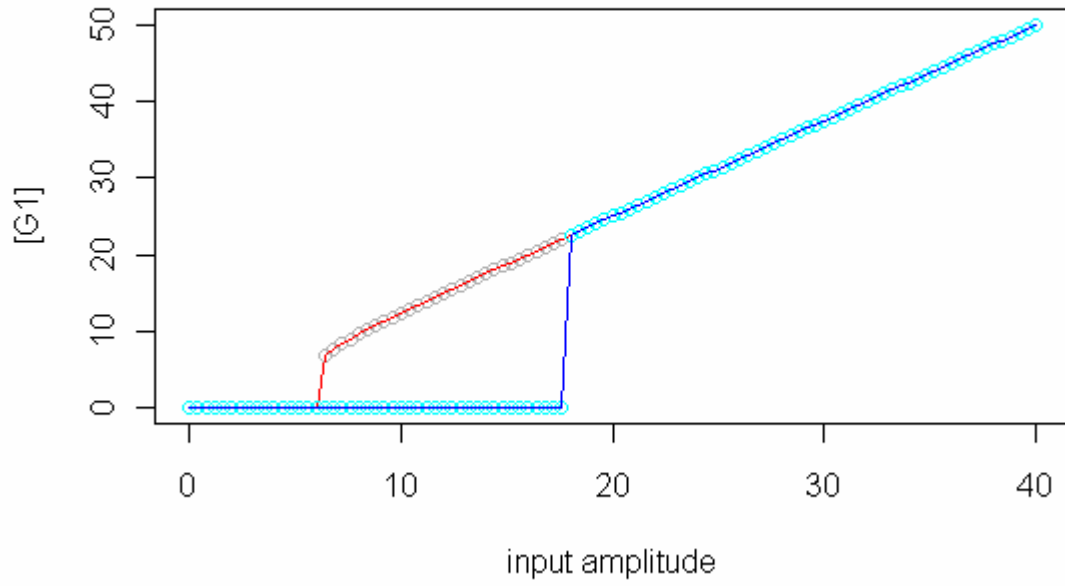
```
windows(height=4,width=6)
plot.new()
plot(results1[,1], results1[, 2], xlab="input amplitude", ylab="[G1]", col="gray70")
lines(results1[,1], results1[, 2], type="l", col="red")
lines(results2[,1], results2[, 2], type="p", col="cyan")
lines(results2[,1], results2[, 2], type="l", col="blue")
```

WT simulation results:

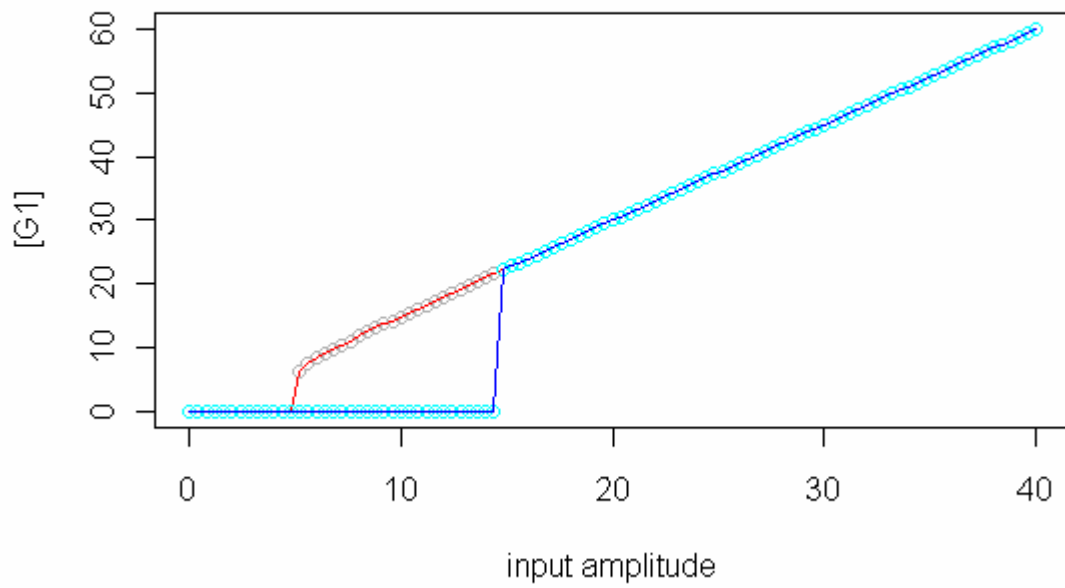




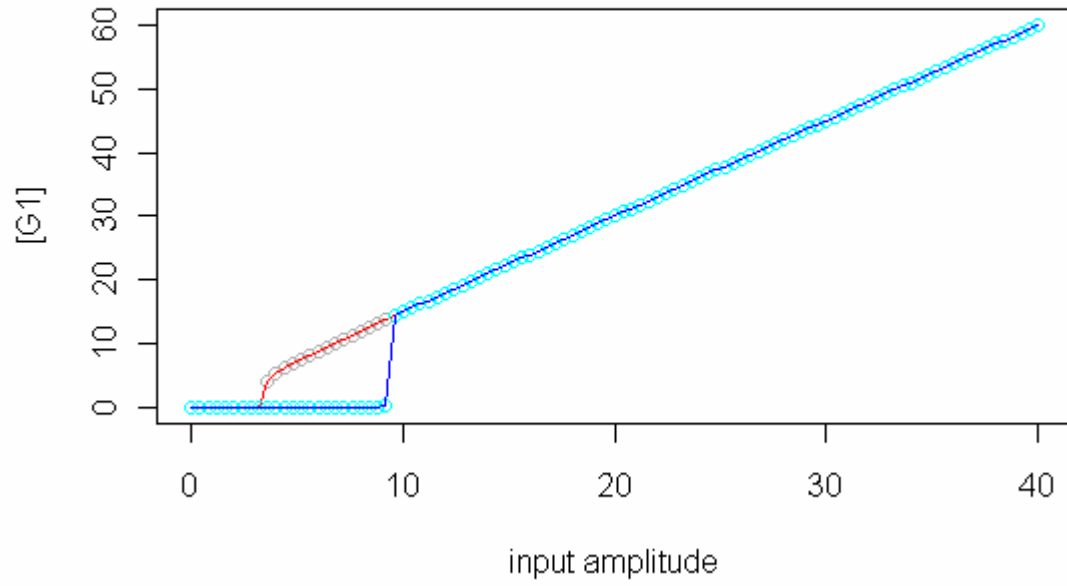
kt1 → 1.25



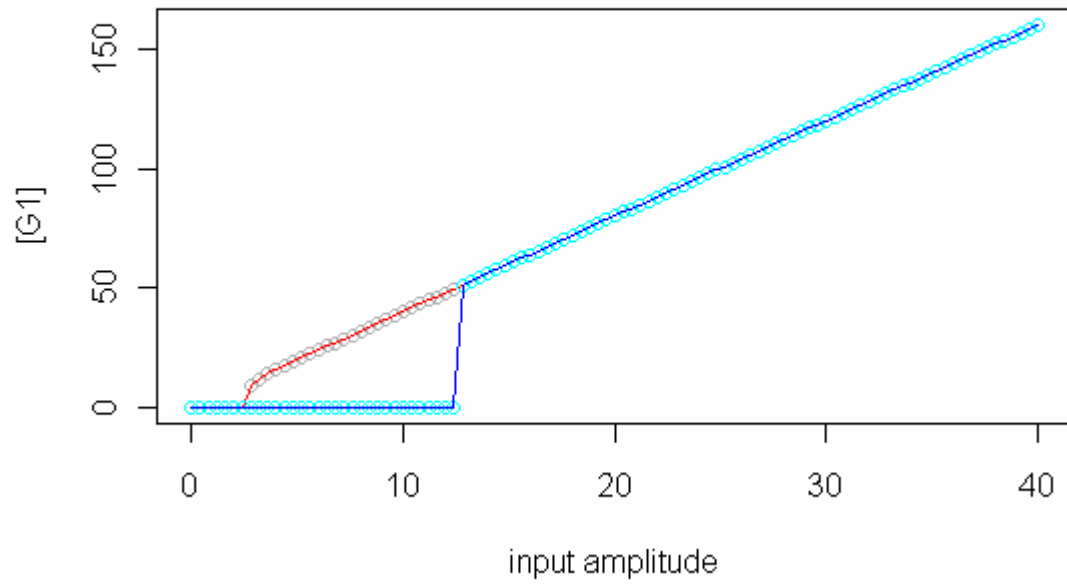
kt1 → 1.5



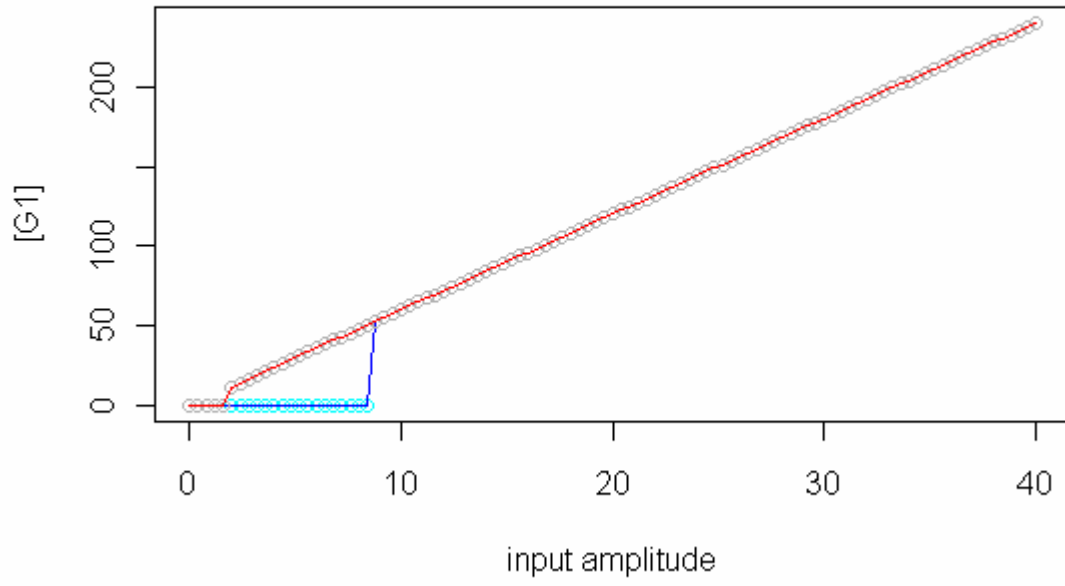
kt1 \rightarrow 1.5 & Gene2 input \rightarrow down 50% (10 instead of 20)

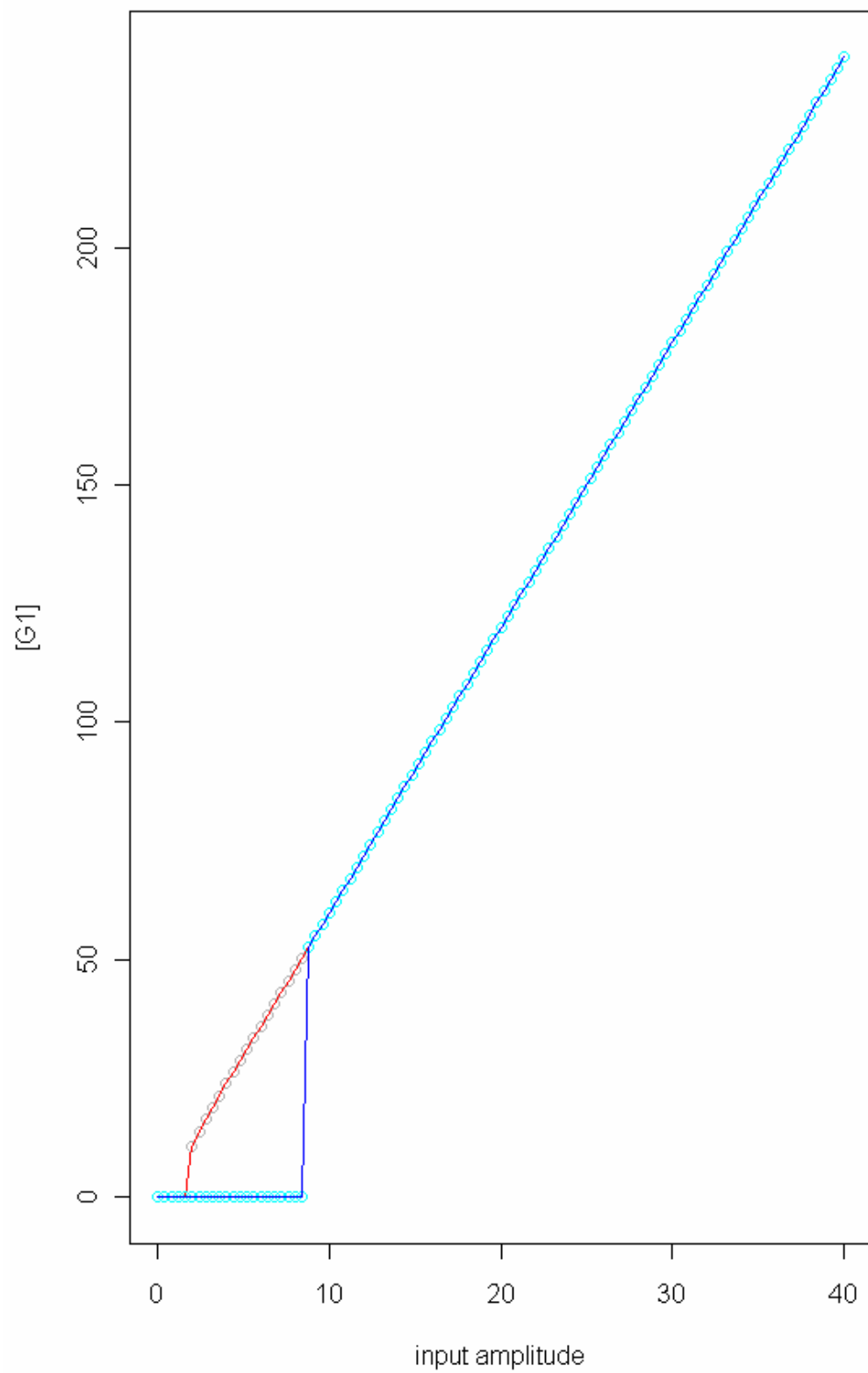


KD1 \rightarrow 0.5



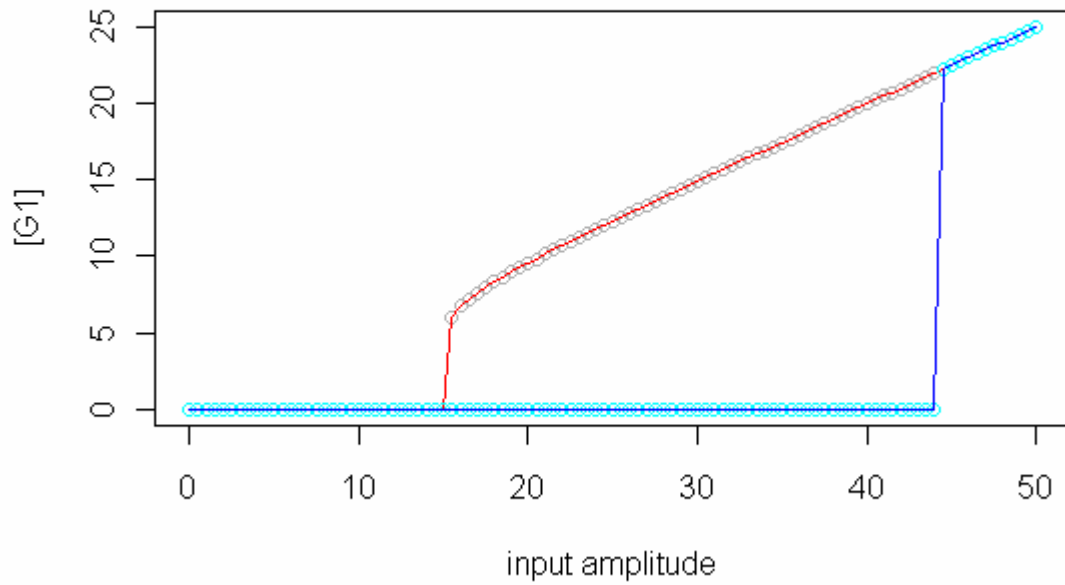
kt1 \rightarrow 1.5 & KD1 \rightarrow 0.5





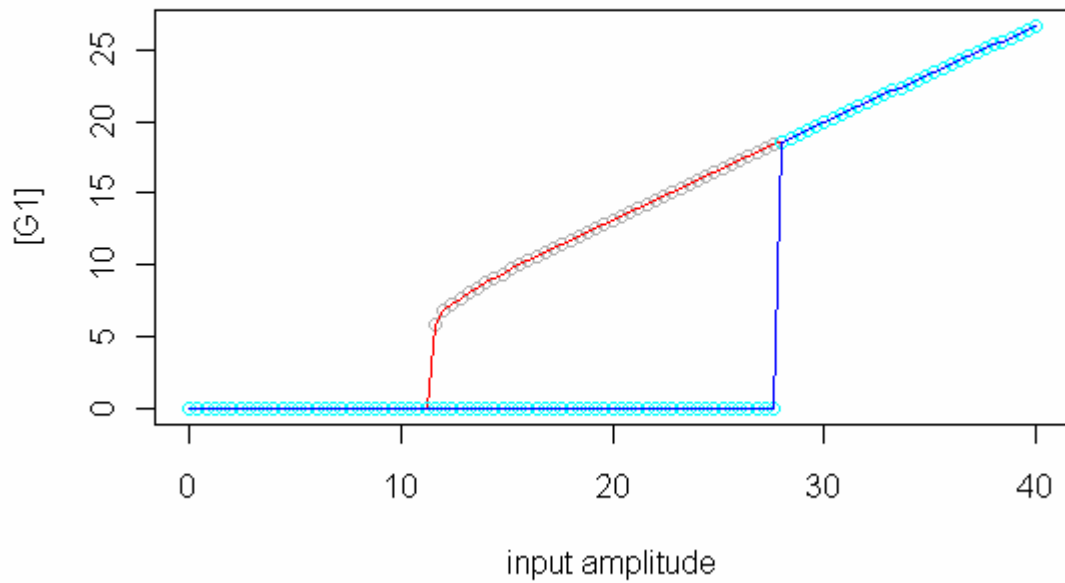
kt1 \rightarrow 0.5

Note switch on threshold is now > normal max [G1]



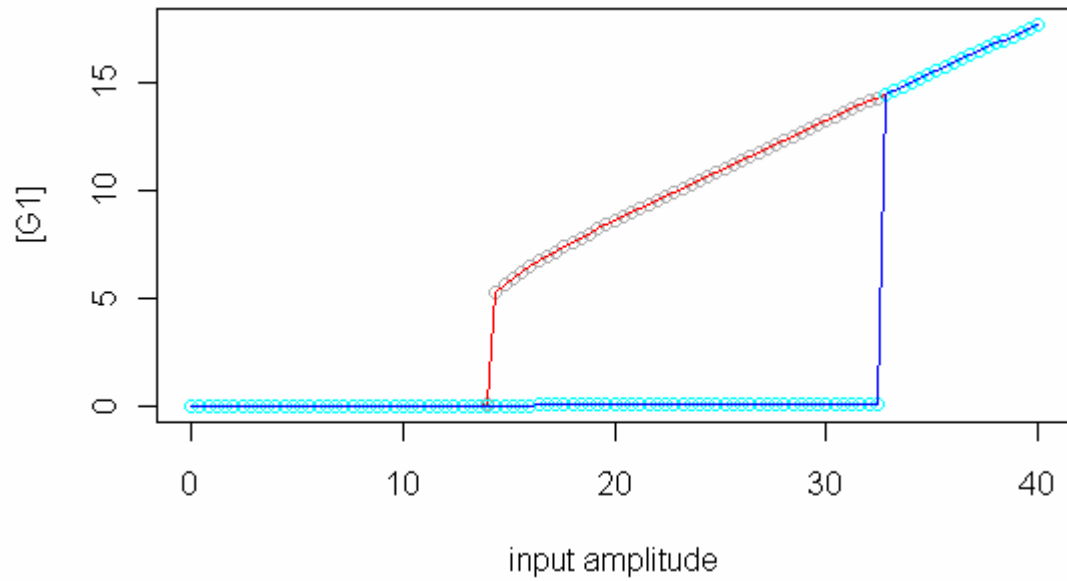
kd1 \rightarrow 1.5

No big deal in itself



KD1 \rightarrow 1.5

Note G1 max is not high enough to activate a downstream WT switch



KD1 \rightarrow 1.5 & kd1 \rightarrow 1.5

Now the switch can not be turned ON by a WT switch

