Supplementary Methods - Automatic generation of paper figures

Joseph D. Barry, Erika Donà, Darren Gilmour, Wolfgang Huber

Contents

1 Introduction

In this vignette we automatically generate figure panels produced for the paper by Barry *et al*.

```
library(TimerQuant)
library(ggplot2)
library(grid)
library(deSolve)
library(reshape2)
library(dplyr)
data("profileGradients")
data("maturationData")
data("FRETdata")
data("FRETphotoconversionEfficiency")
myTheme <- theme_bw()+theme(title=element_text(size=32), text=element_text(size=24),
    axis.title.y=element_text(vjust=1.5), panel.grid=element_blank())
```

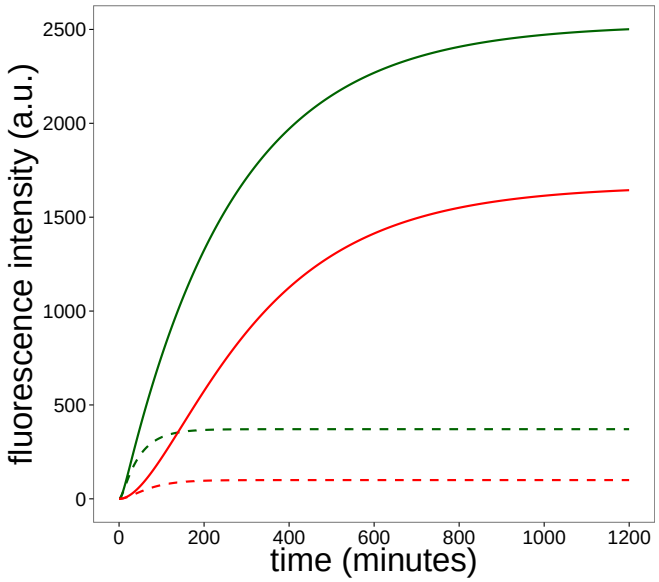

Figure 1: Figure 1B: Time-dependent model solutions for FP1 and FP2 fluorescence intensities.

2 Figure 1

2.1 Figure 1B

```
t <- seq(0.001, 1200, by=0.1)
p0 <- 10
TA <- 30
TB <- 180
kA <- log(2)/TA
kB <- log(2)/TB
T1 <- 5
T2 <- 100
m1 <- log(2)/T1
m2 <- log(2)/T2
df <- data.frame(t=t, x1B=x1(p0, m1, kB, t), x2B=x1(p0, m2, kB, t),
   x1A=x1(p0, m1, kA, t), x2A=x1(p0, m2, kA, t))
df <- melt(df, id="t")
df$FP <- df[, "halflife"] <- NA
df[df$variable %in% c("x1B", "x1A"), "FP"] <- "FP1"
df[df$variable %in% c("x2B", "x2A"), "FP"] <- "FP2"
df[df$variable %in% c("x1A", "x2A"), "halflife"] <- paste(TA, "minutes")
df[df$variable %in% c("x1B", "x2B"), "halflife"] <- paste(TB, "minutes")
ggplot(df, aes(x=t, y=value, color=FP, linetype=halflife))+geom_line(size=1)+
    scale_color_manual(values=c("darkgreen", "red"))+
    scale_linetype_manual(values=c("solid", "dashed"))+myTheme+
    scale_x_continuous(breaks=seq(0, 1200, by=200))+
    theme(legend.position="none")+
    labs(list(linetype="half-life", color="fluorophore"))+
    xlab("time (minutes)")+
    ylab("fluorescence intensity (a.u.)")
```

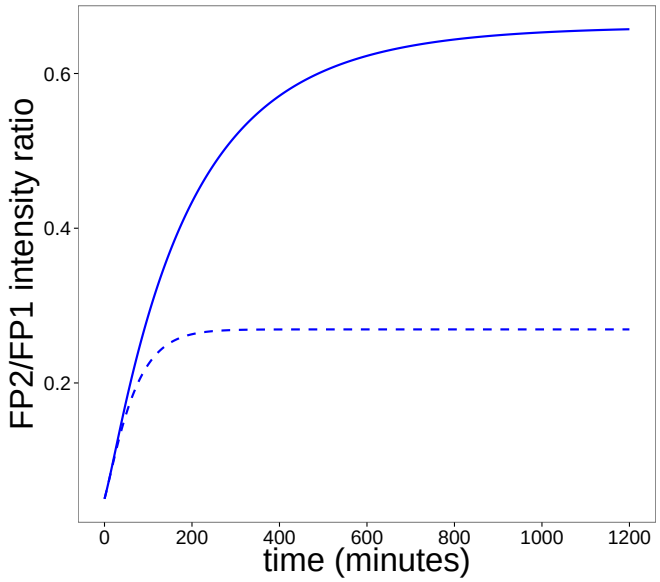

Figure 2: Figure 1C: Time-dependent model solution for the FP2/FP1 fluorescence intensity ratio.

2.2 Figure 1C

```
df <- data.frame(t=t[-1], RatioB=x1(p0, m2, kB, t[-1])/x1(p0, m1, kB, t[-1]),
    RatioA=x1(p0, m2, kA, t[-1])/x1(p0, m1, kA, t[-1]))
df <- melt(df, id="t")
ggplot(df, aes(x=t, y=value, color=variable, linetype=variable))+geom_line(size=1)+
    scale_color_manual(values=c("blue", "blue"))+
    scale_linetype_manual(values=c("solid", "dashed"))+myTheme+
    scale_x_continuous(breaks=seq(0, 1200, by=200))+
    theme(legend.position="none")+xlab("time (minutes)")+
    ylab("FP2/FP1 intensity ratio")
```

2.3 Figure 1D

```
T3 <- 15
halfLifeSeq <- seq(1, 1000, by=0.1)
df <- data.frame(halfLife=halfLifeSeq, rssT2=ratioSteadyState(T1, T2, halfLifeSeq),
    rssT3=ratioSteadyState(T1, T3, halfLifeSeq))
df <- melt(df, id="halfLife")
df$FP2 <- NA
df[df$variable == "rssT2", "FP2"] <- paste(T2, "minute maturation time")
df[df$variable == "rssT3", "FP2"] <- paste(T3, "minute maturation time")
df$FP2 <- factor(df$FP2, levels=c("15 minute maturation time",
        "100 minute maturation time"))
ggplot(df, aes(x=halfLife, y=value, color=FP2))+
    geom_line(size=1)+scale_color_manual(values=c("darkgrey", "blue"))+
    scale_x_log10(breaks=getBreaks10(halfLifeSeq))+
    myTheme+
    theme(legend.position="none")+
    xlab(expression("protein half-life"~T[1/2]~"(minutes)"))+
    ylab("steady-state ratio R")+
```

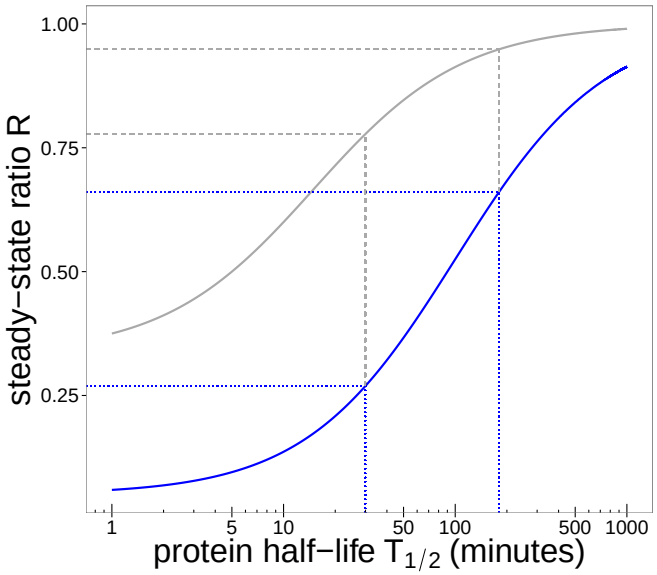

Figure 3: Figure 1D: Comparing steady-state ratios of two timers with different FP2 maturation times.

```
    geom_segment(aes(x=0, xend=TA, y=ratioSteadyState(T1, T2, TA),
        yend=ratioSteadyState(T1, T2, TA)), linetype="dotted", colour="blue")+
    geom_segment(aes(x=0, xend=TB, y=ratioSteadyState(T1, T2, TB),
        yend=ratioSteadyState(T1, T2, TB)), linetype="dotted", colour="blue")+
    geom_segment(aes(x=0, xend=TA, y=ratioSteadyState(T1, T3, TA),
        yend=ratioSteadyState(T1, T3, TA)), linetype="dashed", colour="darkgrey")+
    geom_segment(aes(x=0, xend=TB, y=ratioSteadyState(T1, T3, TB),
        yend=ratioSteadyState(T1, T3, TB)), linetype="dashed", colour="darkgrey")+
    geom_segment(aes(x=TA, xend=TA, y=-Inf, yend=ratioSteadyState(T1, T2, TA)),
        linetype="dotted", colour="blue")+
    geom_segment(aes(x=TB, xend=TB, y=-Inf, yend=ratioSteadyState(T1, T2, TB)),
        linetype="dotted", colour="blue")+
    geom_segment(aes(x=TA, xend=TA, y=ratioSteadyState(T1, T2, TA),
        yend=ratioSteadyState(T1, T3, TA)), linetype="dashed", colour="darkgrey")+
    geom_segment(aes(x=TB, xend=TB, y=ratioSteadyState(T1, T2, TB),
        yend=ratioSteadyState(T1, T3, TB)), linetype="dashed", colour="darkgrey")+
    annotation_logticks(sides="b")
```

2.4 Figure 1E

```
tRangeFP1 <- c(1, 50)
tRangeFP2 <- c(10, 500)
colRamp <- colorRampPalette(colors=c('darkgreen','yellow', 'orange', 'red'))(100)
h <- genRatioHeatmap(tRangeFP=tRangeFP2, Tfixed=T1, TA=min(halfLifeSeq),
    TB=max(halfLifeSeq), channel=2, n=150, ramp=colRamp)
h <- h+geom_hline(yintercept=TA, linetype="dashed", size=1)+
    geom_hline(yintercept=TB, linetype="dashed", size=1)+myTheme+
    annotation_logticks()
print(h)
```

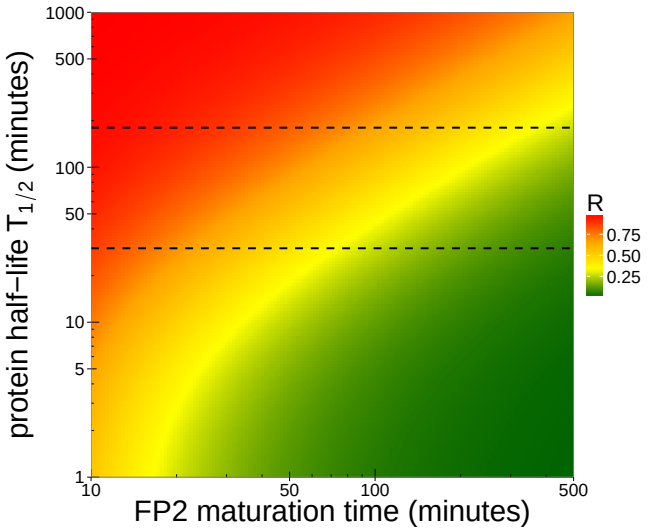

Figure 4: Figure 1E: Steady-state ratios as a funciton of FP2 maturation time and protein half-life.

2.5 Figure 1F

```
pSeq <- c(0.5, 2, 8)
nRealizations <- 500
sigmaAdd <- 1
T2 <- getSpacedSeq(c(10, 500), n=100)
res <- lapply(pSeq, function(p) lapply(T2, function(T)
    simulatedSignalN(T1, T2=T, TA, TB, sigmaAdd, nRealizations, p=p, E=0)))
dfFP2 <- lapply(seq_along(res), function(i) {
    df2 <- lapply(seq_along(res[[i]]), function(j)
        return(data.frame(p=pSeq[i], T=T2[j], D=res[[i]][[j]])))
    df2 <- do.call("rbind", df2)
    return(df2)
})
dfFP2 <- do.call("rbind", dfFP2)
dfFP2$FP <- "FP2"
dfFP2s <- dfFP2 %>% group_by(p, T, FP) %>%
    summarise(D.mean=mean(D, na.rm=TRUE), D.sd=sd(D, na.rm=TRUE))
dfFP2s$D0 <- simulatedSignal(T1, dfFP2s$T, TA, TB, sigmaAdd=0, p=dfFP2s$p)
T1 <- getSpacedSeq(c(1, 30), n=100)
T2 <- 100
res <- lapply(pSeq, function(p) lapply(T1, function(T)
    simulatedSignalN(T1=T, T2, TA, TB, sigmaAdd, nRealizations, p=p, E=0)))
dfFP1 <- lapply(seq_along(res), function(i) {
    df2 <- lapply(seq_along(res[[i]]), function(j)
        return(data.frame(p=pSeq[i], T=T1[j], D=res[[i]][[j]])))
    df2 <- do.call("rbind", df2)
    return(df2)
})
dfFP1 <- do.call("rbind", dfFP1)
dfFP1$FP <- "FP1"
dfFP1s <- dfFP1 %>% group_by(p, T, FP) %>%
    summarise(D.mean=mean(D, na.rm=TRUE), D.sd=sd(D, na.rm=TRUE))
dfFP1s$D0 <- simulatedSignal(T1=dfFP1s$T, T2, TA, TB, sigmaAdd=0, p=dfFP1s$p)
```

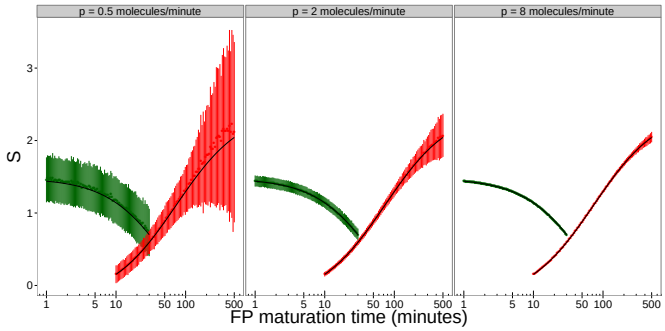

Figure 5: Figure 1F: Results of simulations with an additive error model to investigate the effect of noise on timer signal.

```
df <- bind_rows(dfFP1s, dfFP2s)
df$p <- paste("p =", df$p, "molecules/minute")
df$FPline <- df$FP
df$FPline <- paste0(df$FP, "line")
ggplot(df, aes(T, D.mean, colour=FP))+geom_point()+myTheme+
    xlab("FP maturation time (minutes)")+ylab(expression(S))+
    scale_color_manual(values=c("FP1"="darkgreen", "FP2"="red", "FP1line"="black", "FP2line"="black"))+
    scale_x_log10(breaks=getBreaks10(range(df$T)))+
    geom_errorbar(aes(ymin=D.mean-D.sd, ymax=D.mean+D.sd))+
    geom_line(aes(T, D0, colour=FPline), data=df)+
    facet_wrap(~p)+theme(legend.position="none")+
    annotation_logticks(sides="b")
```

```
T1 <- 5; T2 <- 100
```

2.6 Figure 1G

```
dfs <- filter(df, FP == "FP2")
dfs$CV <- dfs$D.sd/dfs$D.mean

dfFit <- dfs %>% group_by(p, FP) %>% do(fitCV(.))
ggplot(dfs, aes(T, D.sd/D.mean, color=p))+geom_point()+myTheme+
    scale_x_log10(breaks=getBreaks10(range(dfs$T)))+
    scale_y_log10(breaks=getBreaks10(range(dfs$CV)))+
    xlab("FP2 maturation time (minutes)")+ylab(expression(sigma[S]~"/"~mu[S]))+
    geom_line(aes(T, CV), data=dfFit)+
    geom_vline(aes(xintercept=FP2optimumTime, color=p), data=dfFit,
        linetype="dashed")+
    theme(legend.position="none")+
    annotation_logticks()
```

2.7 Figure S2

```
Eseq <- seq(0, 1, length=100)
p <- 2
res <- lapply(Eseq, function(E)
    simulatedSignalN(T1, T2=T2, TA, TB, sigmaAdd, nRealizations, p=p, E=E))
```

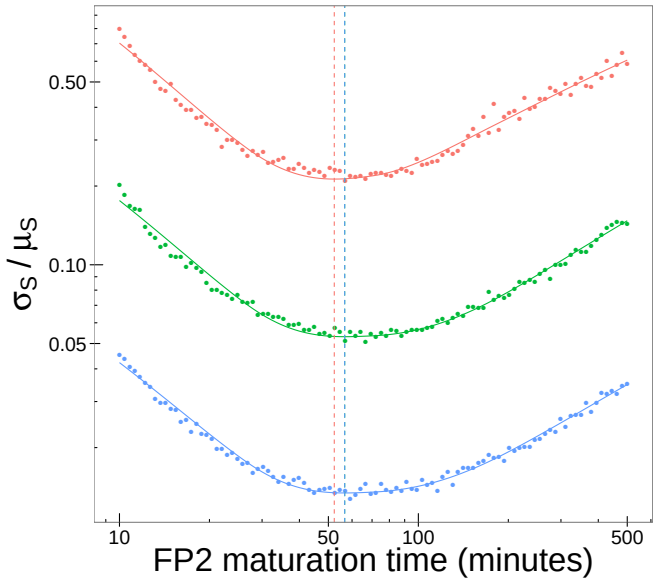

Figure 6: Figure 1G: Signal-to-noise as a function of FP2 maturation time for different protein production rates.

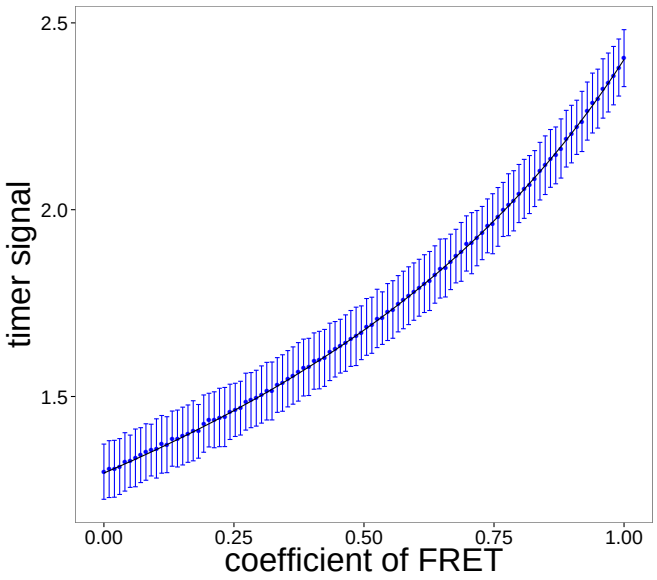

Figure 7: Figure S2: The effect of FRET on timer signal.

```
dfE <- lapply(seq_along(res), function(i) data.frame(E=Eseq[i], D=res[[i]]))
dfE <- do.call("rbind", dfE)
dfEs <- dfE %>% group_by(E) %>%
    summarise(D.mean=mean(D, na.rm=TRUE), D.sd=sd(D, na.rm=TRUE))
dfEs$D0 <- simulatedSignal(T1=T1, T2, TA, TB, sigmaAdd=0, p=p, E=dfEs$E)
ggplot(dfEs, aes(E, D.mean))+geom_point(col="blue")+myTheme+
    xlab("coefficient of FRET")+ylab("timer signal")+
    geom_errorbar(aes(ymin=D.mean-D.sd, ymax=D.mean+D.sd), col="blue")+
    geom_line(aes(E, D0), data=dfEs)
```

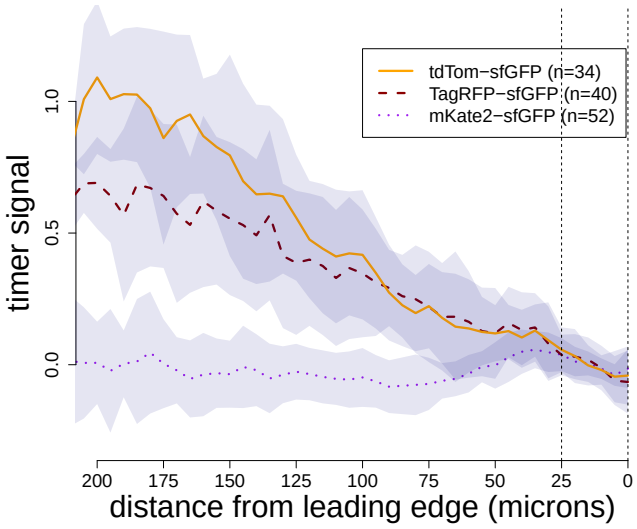

Figure 8: Figure 2B: Comparing the timer signal profiles of three different timers in the zebrafish posterior lateral line primordium.

3 Figure 2

3.1 Figure 2B

```
myCol <- c(tdTom="orange", TagRFP="darkred", mKate2="purple")
A <- profileGradients
Ahat <- apply(A, c(1, 2), function(x) x/mean(x[1:6]))
Ahat <- aperm(Ahat, c(2, 3, 1))
Ahat <- log2(Ahat)
plotPrimordiumProfile(Ahat["mKate2", , ], ylim=c(-0.3, 1.3), xlim=c(-200, 0),
    ylab="timer signal", col=myCol["mKate2"], lwd=3, lty=3,
    cex.axis=1.5, cex.lab=2.5, alpha=0.1)
plotPrimordiumProfile(Ahat["TagRFP", , ], add=TRUE, lty=2, col=myCol["TagRFP"],
    lwd=3, alpha=0.1)
plotPrimordiumProfile(Ahat["tdTom", , ], add=TRUE, lty=1, col=myCol["tdTom"],
    lwd=3, alpha=0.1)
abline(v=0, lty=2)
abline(v=-25, lty=2)
nSamples <- c(tdTom=34, TagRFP=40, mKate2=52)
myLegend <- paste0(names(nSamples), "-sfGFP (n=", nSamples, ")")
legend(x=-100, y=1.2, legend=myLegend, lty=1:3, lwd=3, cex=1.5, col=myCol)
```

3.2 Figure S1

```
A <- maturationData
A[, "meanGFP", 4, 3, "mKate2"] <- NA # jump in signal in gfp
A[, "meanGFP", 2, 2, "TagRFP"] <- NA # jump in signal in gfp
A[, "meanGFP", 1, 4, "tdTom"] <- NA # jump in gfp
A[, "meanRFP", 1:2, 1:4, "TagRFP"] <- NA
```

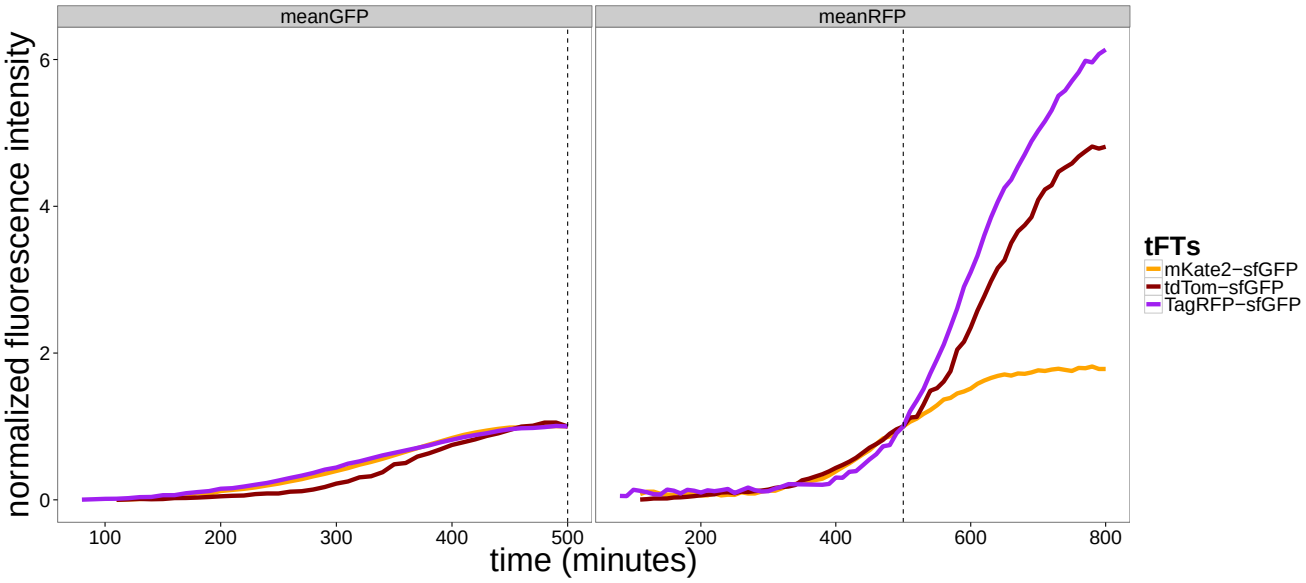

Figure 9: Figure S1: Comparing the maturation rates of three different RFPs.

```
A <- A[as.integer(dimnames(A)$t) <= 800, c("meanGFP", "meanRFP"), , , ]
A[as.integer(dimnames(A)$t) > 500, "meanGFP", , , ] <- NA
indT <- which(dimnames(A)$t == "500")

myNorm <- function(x, indT) {
    x <- x-min(x, na.rm=TRUE)
    x[x < 0] <- NA
    x <- x/x[indT]
    return(x)
}

Anorm <- apply(A, c(2, 3, 4, 5), myNorm, indT)
dimnames(Anorm) <- dimnames(A)
```

```
AnormSummarizeViews <- apply(Anorm, c(1, 2, 3, 5), mean, na.rm=TRUE)
AnormSummarizeSamples <- apply(AnormSummarizeViews, c(1, 2, 4), mean, na.rm=TRUE)
Amelt <- melt(AnormSummarizeSamples, id="t")
Amelt$tFTs <- as.factor(paste0(Amelt$tFTs, "-sfGFP"))
o <- match(levels(Amelt$tFT), paste0(names(myCol), "-sfGFP"))
Amelt$tFTs <- factor(Amelt$tFTs, levels=levels(Amelt$tFTs)[o])
ggplot(na.omit(Amelt), aes(x=t, y=value, color=tFTs))+geom_line(size=2)+
    facet_wrap(~data, scales="free_x")+
    scale_color_manual(values=as.vector(myCol))+
    geom_vline(xintercept=500, linetype="dashed")+
    xlab("time (minutes)")+ylab("normalized fluorescence intensity")+myTheme
```

3.3 Figure 2C

```
F <- FRETdata
myColReduced <- myCol[names(myCol) %in% F$tFT]
F$tFT <- paste0(F$tFT, "-sfGFP")
```

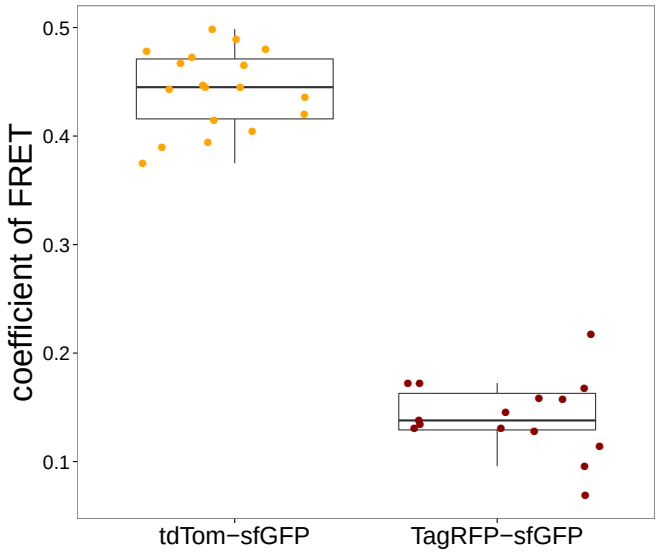

Figure 10: Figure 2C: FRET coefficient measurements for tdTom-sfGFP and TagRFP-sfGFP.

```
F$tFT <- as.factor(F$tFT)
o <- match(levels(F$tFT), paste0(names(myCol), "-sfGFP"))
F$tFT <- factor(F$tFT, levels=levels(F$tFT)[o])
ggplot(F, aes(tFT, FRET, fill="white"))+geom_boxplot(outlier.size=0, fill="white")+
    geom_point(aes(x=tFT, y=FRET, colour=tFT, size=4), position="jitter")+
    scale_color_manual(values=as.vector(myColReduced))+xlab("")+myTheme+
    ylab("coefficient of FRET")+theme(legend.position="none")+
    theme(axis.text.x=element_text(size=24))
```

```
Fs <- tapply(F$FRET, F$tFT, median)
print(Fs)
```

```
##  tdTom-sfGFP TagRFP-sfGFP
##     0.445049     0.137931
```

3.4 Figure 2D

```
FPestimate <- c(mKate2=15, tdTom=80, TagRFP=100)
tFTs <- names(FPestimate)
Rhat <- sapply(tFTs, function(ch) median(Ahat[ch, , "200"], na.rm=TRUE))
Tfront <- 60; Tback <- 5*Tfront
Tseq <- seq(Tfront, Tback, length=100)
T1 <- 6
df <- data.frame(
    halfLife=Tseq,
    tdTomNoFret=signal(T1=T1, T2=FPestimate["tdTom"], TA=Tfront, TB=Tseq, E=0),
    tdTomFret=signal(T1=T1, T2=FPestimate["tdTom"], TA=Tfront, TB=Tseq, E=Fs["tdTom-sfGFP"]),
    TagRFPNoFret=signal(T1=T1, T2=FPestimate["TagRFP"], TA=Tfront, TB=Tseq, E=0),
    TagRFPFret=signal(T1=T1, T2=FPestimate["TagRFP"], TA=Tfront, TB=Tseq, E=Fs["TagRFP-sfGFP"]))
df <- melt(df, id="halfLife")
df$FRET <- !grepl(x=as.character(df$variable), pattern="NoFret")
```

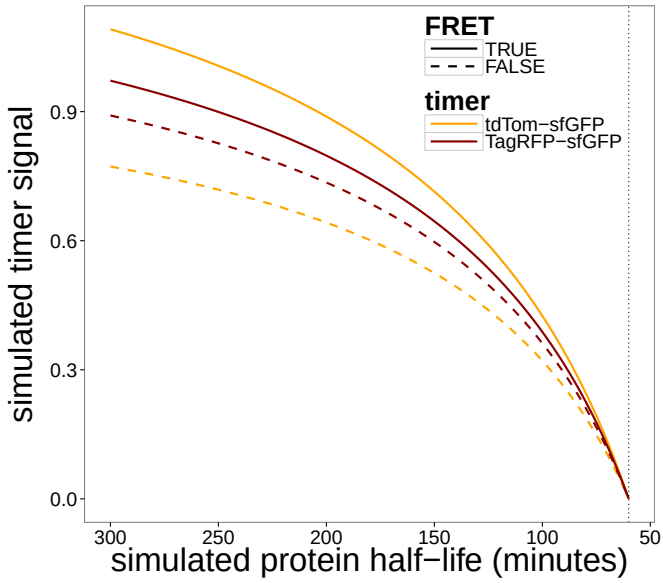

Figure 11: Figure 2D: Model shows that the addition of FRET can cause a reordering of profile slopes.

```
df$FRET <- factor(df$FRET, levels=c("TRUE", "FALSE"))
df$tFT[grepl(x=as.character(df$variable), pattern="mKate2")] <- "mKate2-sfGFP"
df$tFT[grepl(x=as.character(df$variable), pattern="tdTom")] <- "tdTom-sfGFP"
df$tFT[grepl(x=as.character(df$variable), pattern="TagRFP")] <- "TagRFP-sfGFP"
df$tFT <- factor(df$tFT, levels=paste0(names(myCol), "-sfGFP"))
ggplot(df, aes(x=halfLife, y=value, color=tFT, linetype=FRET))+
    geom_line(size=1)+
    scale_color_manual(values=as.vector(myColReduced), guide=guide_legend(title="timer"))+
    scale_linetype_manual(values=c("solid", "dashed"))+
    xlab("simulated protein half-life (minutes)")+
    ylab(expression("simulated timer signal"))+
    geom_vline(xintercept=Tfront, linetype="dotted")+
    scale_x_reverse()+myTheme+
    theme(legend.position=c(0.77, 0.85), legend.key.width=unit(2, "cm"), legend.box.just="left")
```

3.5 Figure S3A

```
t <- seq(0.001, 1000, by=0.1)
x1ss0 <- x1ss(p0, m2, kB, f=1)
tss0 <- tss(m2, kB)
df <- data.frame(t=t, x1=x1(p0, m2, kB, t))
ggplot(df, aes(t, x1))+geom_line(col="red", size=1)+xlab("time (minutes)")+
    ylab("FP2 fluorescence intensity (a.u.)")+
    geom_hline(yintercept=x1ss0)+
    geom_abline(intercept=TimerQuant:::inflectionOffset(p0, m2, kB, 0),
        slope=TimerQuant:::inflectionSlope(p0, m2, kB, 0))+
    geom_vline(xintercept=tss0, linetype="dotted", size=2)+
    geom_point(data=NULL, aes(x=tss0, y=x1ss0), color="blue", size=4)+myTheme
```

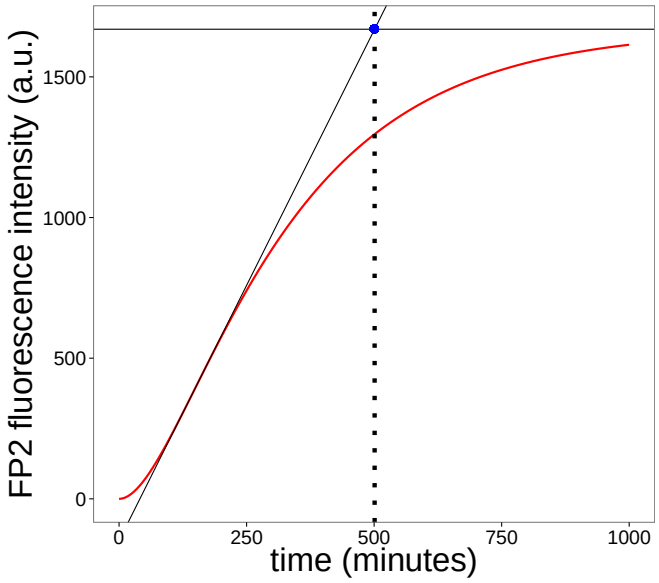

Figure 12: Figure S4A: Quantification of the time to reach steady-state.

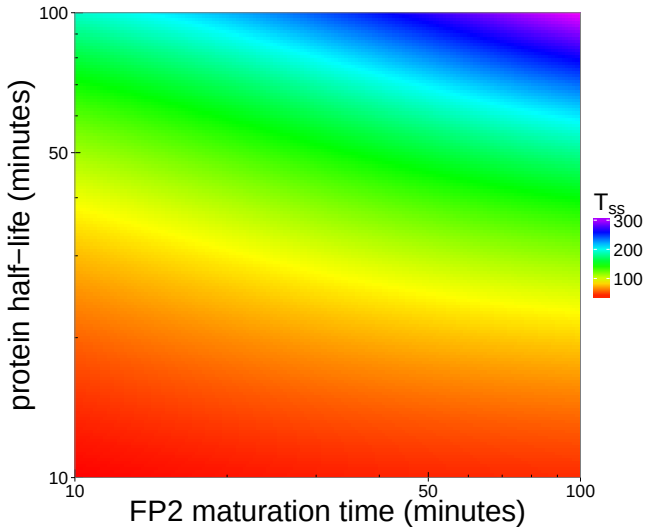

Figure 13: Figure S4B: Time to reach steady-state.

3.6 Figure S3B

```
tRange <- c(10, 100)
colRamp <- rainbow(120)[1:100]
h <- genTimeSteadyStateHeatmap(tRange, tRange, n=150, ramp=colRamp)+myTheme+
    annotation_logticks()
print(h)
```

4 Figure 3

```
t <- seq(0, 600, length=1000)
t0 <- min(t); tf <- max(t)
p0 <- 10; k0 <- log(2)/60
k1 <- log(2)/30
m1 <- log(2)/T1; m2 <- log(2)/T2
x01 <- c(X1=x0ss(p0, m1, k0), X2=x1ss(p0, m1, k0))
x02 <- c(X1=x0ss(p0, m2, k0), X2=x1ss(p0, m2, k0))
```

4.1 Figure 3A

```
myBreaks <- seq(0, 600, by=120)
dfp <- data.frame(t=t,
    linearIncrease=rFun("linearIncrease", r0=p0, tf=tf)(t),
    linearDecrease=rFun("linearDecrease", r0=p0, tf=tf)(t))
types <- colnames(dfp)[-1]
dfp <- melt(dfp, id="t")
gp <- ggplot(dfp, aes(t, value))+geom_line(size=1)+facet_wrap(~variable, ncol=1)+
    xlab("time (minutes)")+ylab("production rate (molecules per minute)")+
    scale_x_continuous(breaks=myBreaks)+myTheme
dfk <- data.frame(t=t,
    linearIncrease=rFun("linearIncrease", r0=k0, tf=tf)(t),
    linearDecrease=rFun("linearDecrease", r0=k0, tf=tf)(t))
dfk <- melt(dfk, id="t")
gk <- ggplot(dfk, aes(t, value))+geom_line(size=1)+facet_wrap(~variable, ncol=1)+
    xlab("time (minutes)")+ylab(expression(paste("degradation rate (minutes"^{-1}, ")")))+
    scale_x_continuous(breaks=myBreaks)+myTheme
df <- lapply(types, function(n) solveModel(x01, x02, t, m1=m1, m2=m2,
    typeProd=n, typeDeg=n, p0=p0, k0=k0))
for (i in seq_along(df)) df[[i]]$type <- types[i]
df <- do.call("rbind", df)
df <- na.omit(df)

reorderFactors <- function(x, channel="uid", ordering) {
    x[, channel] <- as.factor(x[, channel])
    o <- match(ordering, levels(x[, channel]))
    x[, channel] <- factor(x[, channel], levels(x[, channel])[o])
    return(x)
}

df <- reorderFactors(df, "type", types)
df2 <- melt(df, id=c("time", "typeProd", "typeDeg", "type"))
df2 <- filter(df2, variable %in% c("FP1", "FP2"))
g2 <- ggplot(df2, aes(time, value, color=variable))+geom_line(size=1)+
    facet_wrap(~type, ncol=1, scales="fixed")+xlab("time (minutes)")+
    scale_x_continuous(breaks=myBreaks)+
    scale_color_manual(values=c("darkgreen", "red"))+
    ylab("FP1, FP2 fluorescence intensity (a.u.)")+myTheme+
    theme(legend.position="none")
g3 <- ggplot(df, aes_string("time", "Ratio"))+geom_line(color="blue", size=1)+
    facet_wrap(~type, ncol=1, scales="fixed")+xlab("time (minutes)")+
    ylab("FP2/FP1 intensity ratio")+
    scale_x_continuous(breaks=myBreaks)+
```

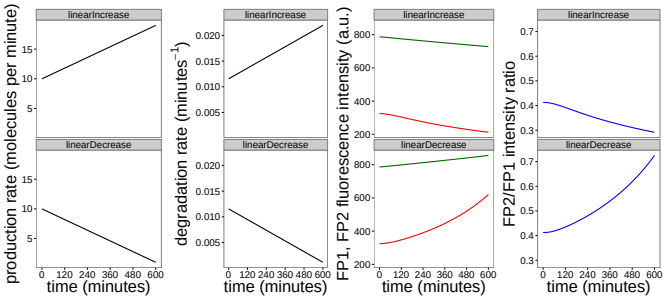

Figure 14: Figure 3A: Timer readouts for simulations where production and degradation are changing over time, but adding or removing proteins from the system at comparable rates.

```
    myTheme
grid.newpage()
pushViewport(viewport(layout=grid.layout(1, 4)))
print(gp, vp=viewport(layout.pos.row=1, layout.pos.col=1))
print(gk, vp=viewport(layout.pos.row=1, layout.pos.col=2))
print(g2, vp=viewport(layout.pos.row=1, layout.pos.col=3))
print(g3, vp=viewport(layout.pos.row=1, layout.pos.col=4))
```

4.2 Figure 3B

```
dfp <- data.frame(t=t,
    burst=rFun("burst", r0=p0, tf=tf)(t),
    stepUp=rFun("stepUp", r0=p0, tf=tf)(t))
types <- colnames(dfp)[-1]
dfp <- melt(dfp, id="t")
gp <- ggplot(dfp, aes(t, value))+geom_line(size=1)+facet_wrap(~variable, ncol=1)+
    xlab("time (minutes)")+ylab("production rate (molecules per minute)")+
    scale_x_continuous(breaks=myBreaks)+myTheme
df <- lapply(types, function(n) solveModel(x01, x02, t, m1=m1, m2=m2,
    typeProd=n, typeDeg="constant", p0=p0, k0=k0))
for (i in seq_along(df)) df[[i]]$type <- types[i]
df <- do.call("rbind", df)
df <- na.omit(df)
df2 <- melt(df, id=c("time", "typeProd", "typeDeg", "type"))
df2 <- filter(df2, variable %in% c("FP1", "FP2"))
g2 <- ggplot(df2, aes(time, value, color=variable))+geom_line(size=1)+
    facet_wrap(~type, ncol=1, scales="fixed")+xlab("time (minutes)")+
    scale_x_continuous(breaks=myBreaks)+
    scale_color_manual(values=c("darkgreen", "red"))+
    ylab("FP1, FP2 fluorescence intensity (a.u.)")+myTheme+
    theme(legend.position="none")
g3 <- ggplot(df, aes_string("time", "Ratio"))+geom_line(color="blue", size=1)+
    facet_wrap(~type, ncol=1, scales="fixed")+xlab("time (minutes)")+
    ylab("FP2/FP1 intensity ratio")+
    scale_x_continuous(breaks=myBreaks)+
    myTheme
grid.newpage()
pushViewport(viewport(layout=grid.layout(1, 3)))
```

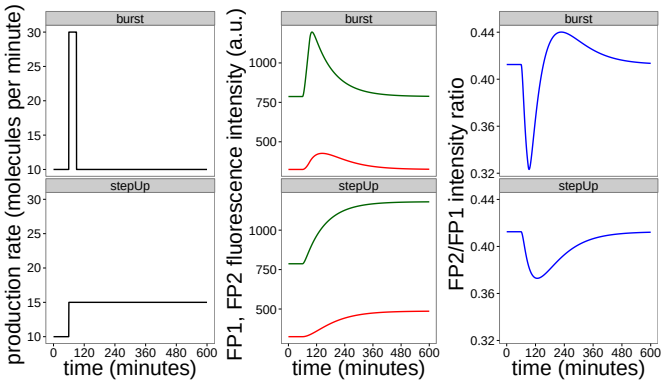

Figure 15: Figure 3B: The effect of rapidly changing production dynamics on timer readouts.

```
print(gp, vp=viewport(layout.pos.row=1, layout.pos.col=1))
print(g2, vp=viewport(layout.pos.row=1, layout.pos.col=2))
print(g3, vp=viewport(layout.pos.row=1, layout.pos.col=3))
```

4.3 Figure 3C

```
dfk <- data.frame(t=t,
    stepDown=rFun("stepDown", r0=k0, tf=tf)(t),
    stepUp=rFun("stepUp", r0=k0, tf=tf)(t))
types <- colnames(dfk)[-1]
dfk <- melt(dfk, id="t")
gk <- ggplot(dfk, aes(t, value))+geom_line(size=1)+xlab("time (minutes)")+
    ylab(expression(paste("degradation rate (minutes"^{-1}, ")")))+
    facet_wrap(~variable, ncol=1)+scale_x_continuous(breaks=myBreaks)+myTheme
df <- lapply(types, function(n) solveModel(x01, x02, t, m1=m1, m2=m2,
    typeProd="constant", typeDeg=n, p0=p0, k0=k0))
for (i in seq_along(df)) df[[i]]$type <- types[i]
df <- do.call("rbind", df)
df <- na.omit(df)
df2 <- melt(df, id=c("time", "typeProd", "typeDeg", "type"))
df2 <- filter(df2, variable %in% c("FP1", "FP2"))
g2 <- ggplot(df2, aes(time, value, color=variable))+geom_line(size=1)+
    xlab("time (minutes)")+
    facet_wrap(~typeDeg, ncol=1, scales="fixed")+
    scale_x_continuous(breaks=myBreaks)+
    scale_color_manual(values=c("darkgreen", "red"))+
    ylab("FP1, FP2 fluorescence intensity (a.u.)")+myTheme+
    theme(legend.position="none")
g3 <- ggplot(df, aes_string("time", "Ratio"))+geom_line(color="blue", size=1)+
    facet_wrap(~typeDeg, ncol=1, scale="fixed")+xlab("time (minutes)")+
    ylab("FP2/FP1 intensity ratio")+scale_x_continuous(breaks=myBreaks)+myTheme
grid.newpage()
pushViewport(viewport(layout=grid.layout(1, 3)))
print(gk, vp=viewport(layout.pos.row=1, layout.pos.col=1))
print(g2, vp=viewport(layout.pos.row=1, layout.pos.col=2))
print(g3, vp=viewport(layout.pos.row=1, layout.pos.col=3))
```

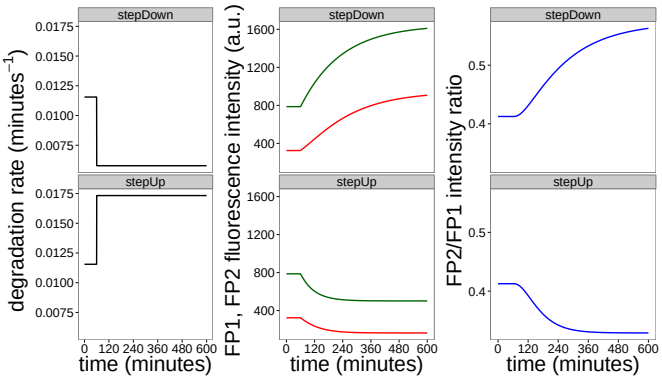

Figure 16: Figure 3C: The effect of rapidly changing degradation dynamics on timer readouts.