

# Animated Longitudinal Trajectories

The idea here is to create all plots (each *frame* in the animation) in advance, which is then stitched together using the `animation` package in `LATEX`.

First, we define longitudinal trajectories for a stable and a declining patient, separately:

```
stable <- function(time) 60 - 0.01 * time + rnorm(n = length(time), sd = sqrt(5))
declining <- function(time) 60 - 3 * time + rnorm(n = length(time), sd = sqrt(5))
```

We then simulate two subjects:

```
set.seed(375683275)

patient_stable <- data.frame(
  id = 1,
  type = "Stable",
  ad cens = 12,
  obtime = c(0, rep(3, 4))
) %>%
  mutate(t = cumsum(obtime)) %>%
  mutate(y = stable(t)) %>%
  filter(t <= ad cens)

patient_declining <- data.frame(
  id = 2,
  type = "Declining",
  ad cens = runif(1, 10, 12),
  obtime = c(0, rgamma(n = 100, shape = 2, scale = 0.25))
) %>%
  mutate(t = cumsum(obtime)) %>%
  mutate(y = declining(t)) %>%
  filter(t <= ad cens)
```

We create a single dataset:

```
hcd <- bind_rows(patient_stable, patient_declining)
```

We also create a dataset to annotate outcomes:

```
outcome <- hcd %>%
  group_by(id) %>%
  filter(t == max(t))
```

We now start creating plots for each time point, which are:

```
time_points <- sort(unique(hcd$t))
time_points
```

```
## [1] 0.000000 1.280807 1.509051 2.184977 2.704514 3.000000 3.054377
## [8] 3.443566 4.131913 4.164542 4.271853 4.712551 5.431285 5.793062
## [15] 6.000000 6.411639 6.612505 7.428142 7.902333 8.141015 9.000000
## [22] 9.024912 9.260073 9.682912 9.904780 10.525818 11.151080 11.185485
```

```
## [29] 12.000000
```

Here we loop over possible time points:

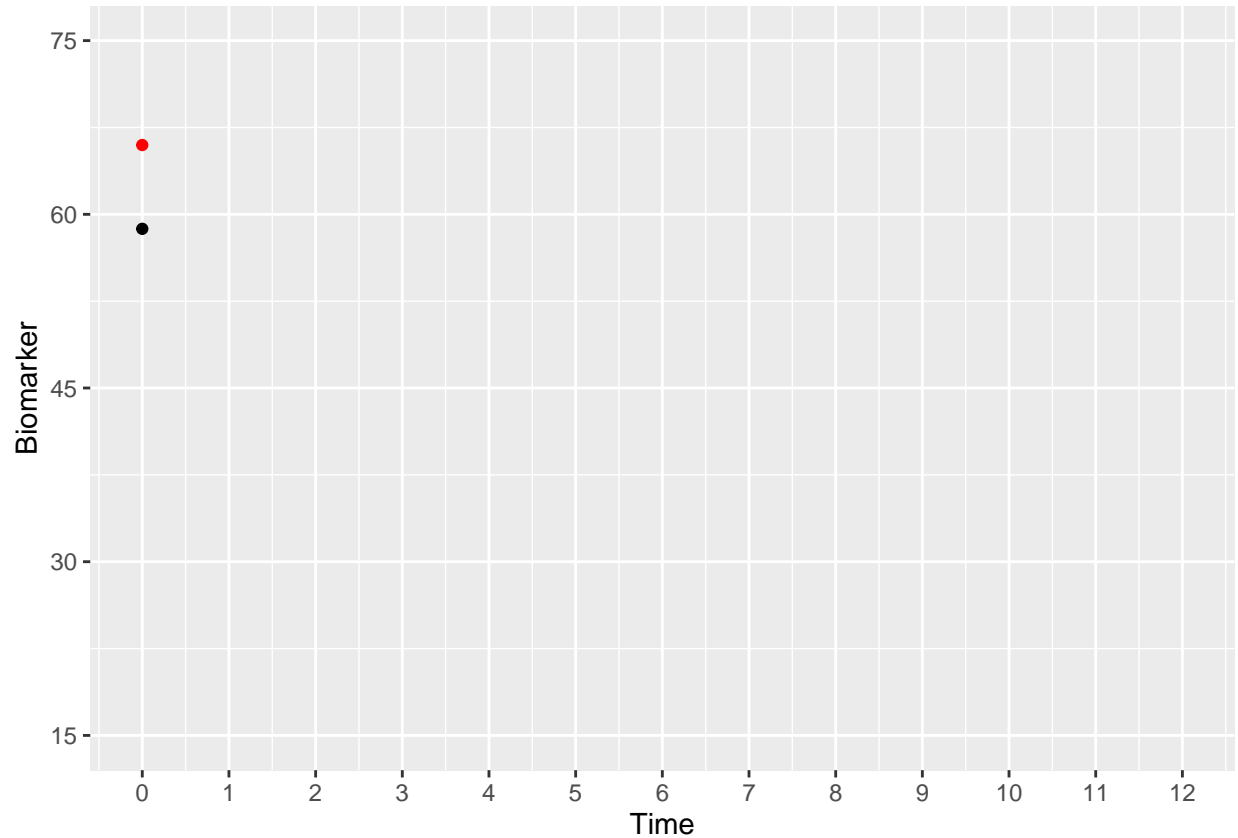
```
for (i in seq_along(time_points)) {  
  nm1 <- nrow(filter(hcd, t <= time_points[i] & id == 1))  
  nm2 <- nrow(filter(hcd, t <= time_points[i] & id == 2))  
  plot <- ggplot(  
    data = filter(hcd, t <= time_points[i]),  
    aes(x = t, y = y, group = id, colour = type)  
  ) +  
    # geom_vline() highlights the current time  
    # green for illustration purposes only  
    geom_vline(xintercept = time_points[i], color = "green") +  
    # geom_line() and geom_point() for each observation  
    geom_line() +  
    geom_point() +  
    # annotation for the cumulative number of measurements  
    annotate("text", x = 0, y = 15, label = paste("#:", nm2), color = "red") +  
    annotate("text", x = 0, y = 75, label = paste("#:", nm1), color = "black") +  
    # using color to match number of observations  
    # (might need some fiddling around)  
    scale_color_manual(values = c("red", "black")) +  
    # scales and plotting area needs to be fixed, consistent over frames  
    scale_y_continuous(breaks = seq(15, 75, by = 15)) +  
    scale_x_continuous(breaks = 0:12) +  
    coord_cartesian(xlim = c(0, 12), ylim = c(15, 75)) +  
    # no need for a legend here  
    theme(legend.position = "none") +  
    labs(x = "Time", y = "Biomarker")  
  # we add mortality/censoring outcomes when/if they happen  
  # this is what the 'outcome' data.frame is used for  
  if (time_points[i] >= outcome$t[1]) {  
    plot <- plot +  
      annotate("point", x = outcome$t[1], y = outcome$y[1], shape = 5, color = "green")  
  }  
  if (time_points[i] >= outcome$t[2]) {  
    plot <- plot +  
      annotate("point", x = outcome$t[2], y = outcome$y[2], shape = 10, color = "green")  
  }  
  # then, saving each 'frame'  
  ggplot2::ggsave(  
    plot = plot,  
    filename = paste0("Plots/frame-", i, ".pdf"),  
    width = 5, height = 5, dpi = 300  
  )  
}
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust  
## the group aesthetic?
```

The plot at “time zero” to set up the stage is:

```
ggplot(filter(hcd, t == 0), aes(x = t, y = y, group = id, colour = type)) +  
  geom_point() +  
  # note:
```

```
# same scale, plotting area, labels, etc.
scale_color_manual(values = c("red", "black")) +
scale_y_continuous(breaks = seq(15, 75, by = 15)) +
scale_x_continuous(breaks = 0:12) +
coord_cartesian(xlim = c(0, 12), ylim = c(15, 75)) +
labs(x = "Time", y = "Biomarker") +
theme(legend.position = "none")
```



Then, the following L<sup>A</sup>T<sub>E</sub>X code combines the frame that we created before:

```
\begin{center}
\animategraphics[autoplay, width = \textwidth]{4}{Plots/frame-}{1}{29}
\end{center}
```

29 here is the total number of time points:

```
length(time_points)
```

```
## [1] 29
```

The result is:

Note that this requires `\usepackage{animate}` in the header of the document.