Fine-Gray regression and the problem of total failure probability exceeding one

Hein Putter

10 januari 2021

Contents

Introduction

The purpose of this document is to study the extent of the problem of the total failure probability exceeding one, when Fine-Gray models are fitted for all causes. We restrict attention here to the case of two causes of failure and a single covariate x which has has a standard normal distribution.

The problem of the total failure probability exceeding one

We start from two proportional subdistribution models, given by

$$F_1(t\,|\,x) = 1-(1-F_{10}(t))^{\gamma_1(x)}, \qquad F_2(t\,|\,x) = 1-(1-F_{20}(t))^{\gamma_2(x)},$$

where $F_{10}(t)$ and $F_{20}(t)$ are the baseline cumulative incidences of cause 1 and 2, respectively, corresponding to $x=0$, and $\gamma_1(x) = e^{\beta_1 x}$ and $\gamma_2(x) = e^{\beta_2 x}$ are the subdistribution hazard ratios for x of cause 1 and cause 2, respectively.

Fix a time point t, and define $p_1 = F_{10}(t)$ and $p_2 = F_{20}(t)$. Then the total failure probability is given by

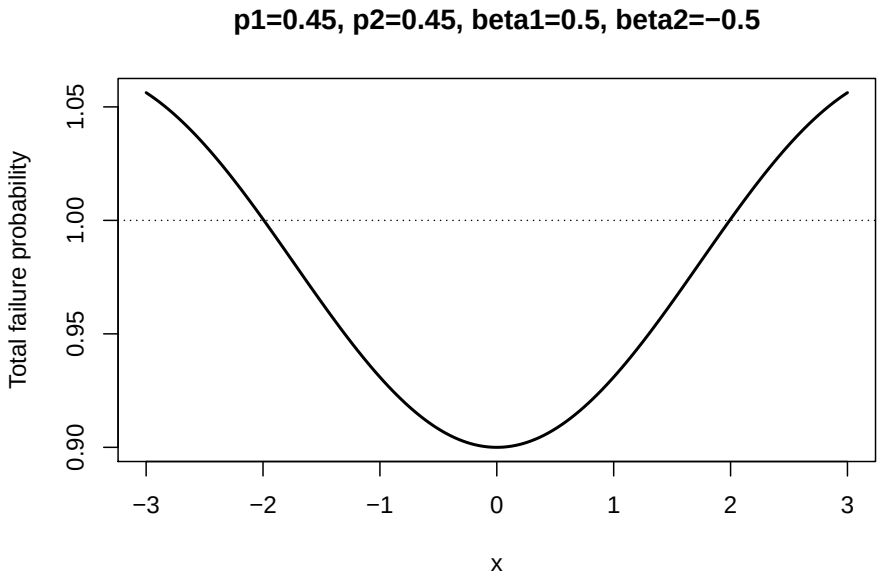
$$\text{TFP}(x) = F_1(t\,|\,x) + F_2(t\,|\,x) = 1-(1-p_1)^{\gamma_1(x)} + 1-(1-p_2)^{\gamma_2(x)}.$$

What I will do in the remainder of this document is see, for given baseline cumulative incidence values at time t, p_1 and p_2, and for given β_1 and β_2, what values of x will result in non-admissible $\text{TFP}(x) > 1$. After determining what values of x result in non-admissible $\text{TFP}(x) > 1$, we can quantify the probability that this happens (knowing X comes from a standard normal distribution). Intuition says that $\text{TFP}(x) > 1$ will happen more frequently for higher p_1 and p_2, and for larger (in absolute value) β_1 and β_2.

I will start by defining the function `TFP`, and making a plot of $\text{TFP}(x)$ against x, for $p_1 = p_2 = 0.45$, and $\beta_1 = -\beta_2 = 0.5$, reasonably close to the simulation earlier, for $t = 20$.

```
TFP <- function(x, p1, p2, beta1, beta2)
  1 - (1-p1)^(exp(beta1*x)) + 1 - (1-p2)^(exp(beta2*x))
xseq <- seq(-3, 3, by=0.01)
p1 <- 0.45; p2 <- 0.45
beta1 <- 0.5; beta2 <- -0.5
plot(xseq, TFP(xseq, p1=p1, p2=p2, beta1=beta1, beta2=beta2), type="l", lwd=2,
     xlab="x", ylab="Total failure probability")
```

```
abline(h=1, lty=3)
title(main="p1=0.45, p2=0.45, beta1=0.5, beta2=-0.5")
```



We can find the points where TFP(x) crosses 1 by applying the function `uniroot`.

```
TFPmin1 <- function(x, p1, p2, beta1, beta2)
  1 - (1-p1)^(exp(beta1*x)) + 1 - (1-p2)^(exp(beta2*x)) - 1
# Positive x
TFPplus <- uniroot(TFPmin1, c(0, 10), p1=p1, p2=p2, beta1=beta1, beta2=beta2)$root
TFPplus
```

```
## [1] 1.992583
```

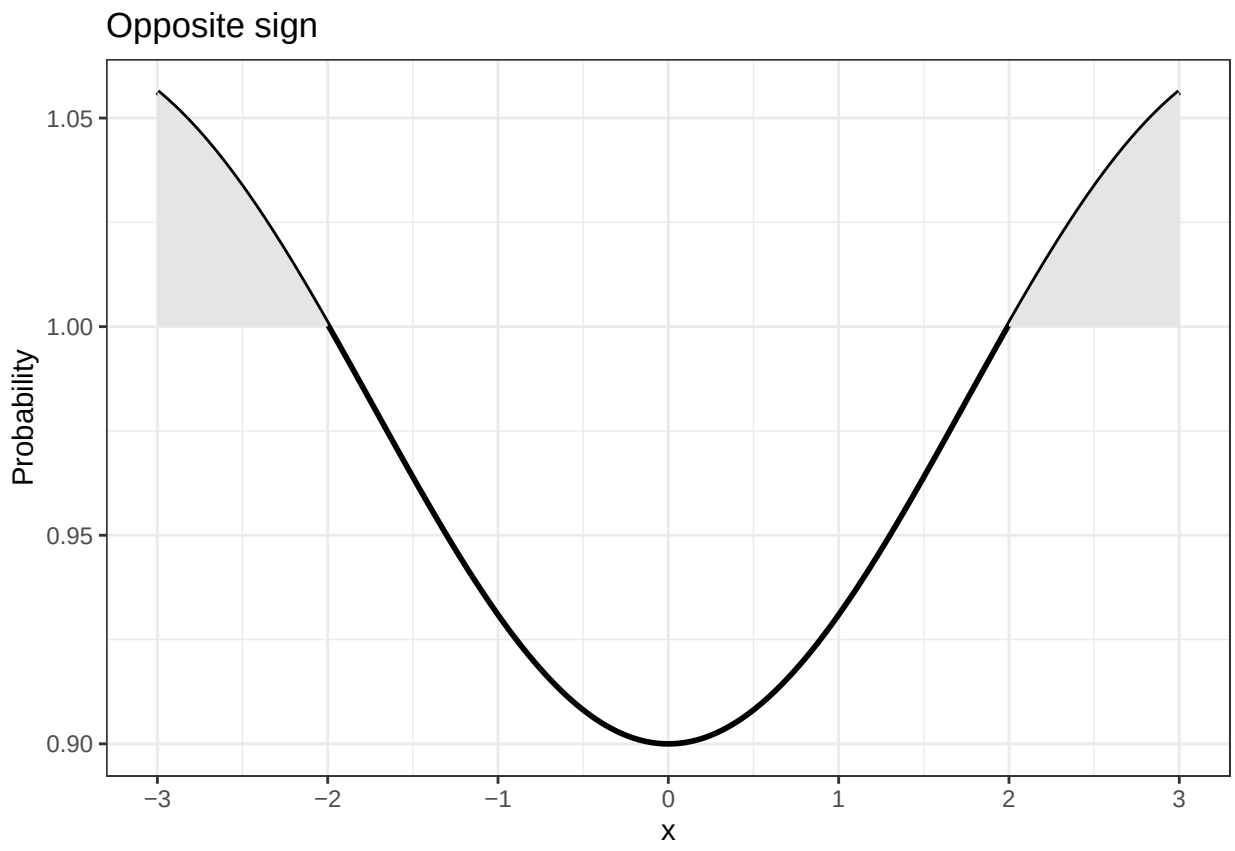
```
# Negative x
TFPmin <- uniroot(TFPmin1, c(-10, 0), p1=p1, p2=p2, beta1=beta1, beta2=beta2)$root
TFPmin
```

```
## [1] -1.992583
```

Let's go for a ggplot version of this.

```
library(ggplot2)
library(tidyverse)
tmp <- tibble(x=xseq, tfp=TFP(xseq, p1=p1, p2=p2, beta1=beta1, beta2=beta2))

ggp1 <- tmp %>%
  ggplot(aes(x = x, y = tfp)) +
  geom_line(size = 1) +
  labs(
    title = "Opposite sign",
    x = "x",
    y = "Probability"
  ) +
  scale_x_continuous(breaks = -3:3) +
```

```
  # We need two ribbons to avoid line joining them
  geom_ribbon(
    data = tmp %>% filter(tfp >= 1 & x < 0),
    aes(x = x, ymin = 1, ymax = tfp),
    fill = "grey90"
  ) +
  geom_ribbon(
    data = tmp %>% filter(tfp >= 1 & x >= 0),
    aes(x = x, ymin = 1, ymax = tfp),
    fill = "grey90"
  ) +
  theme_bw()
ggp1
```



The corresponding normal probabilities are given by

```
pnorm(TFPplus, lower.tail=FALSE)
```

```
## [1] 0.02315354
```

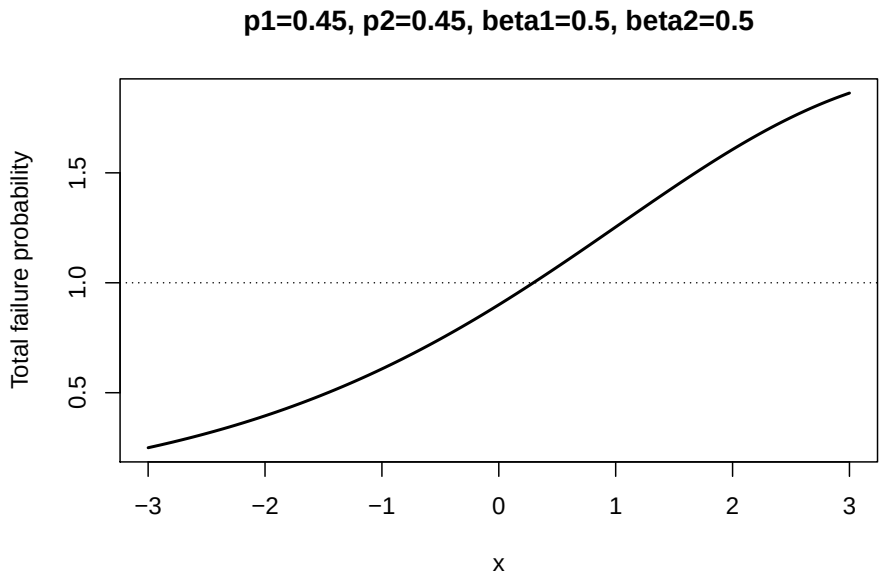
```
pnorm(TFPmin, lower.tail=TRUE)
```

```
## [1] 0.02315354
```

The sum of these probabilities, 0.0463071, is the probability of a total failure probability exceeding one, for the above values of p_1, p_2, β_1 and β_2.

Let's have a look at $\beta_1 = \beta_2 = 0.5$.

```
xseq <- seq(-3, 3, by=0.01)
p1 <- 0.45; p2 <- 0.45
beta1 <- 0.5; beta2 <- 0.5
plot(xseq, TFP(xseq, p1=p1, p2=p2, beta1=beta1, beta2=beta2), type="l", lwd=2,
     xlab="x", ylab="Total failure probability")
abline(h=1, lty=3)
title(main="p1=0.45, p2=0.45, beta1=0.5, beta2=0.5")
```




```
# Positive x
TFPplus <- uniroot(TFPmin1, c(0, 10), p1=p1, p2=p2, beta1=beta1, beta2=beta2)$root
TFPplus
```

```
## [1] 0.2958484
```

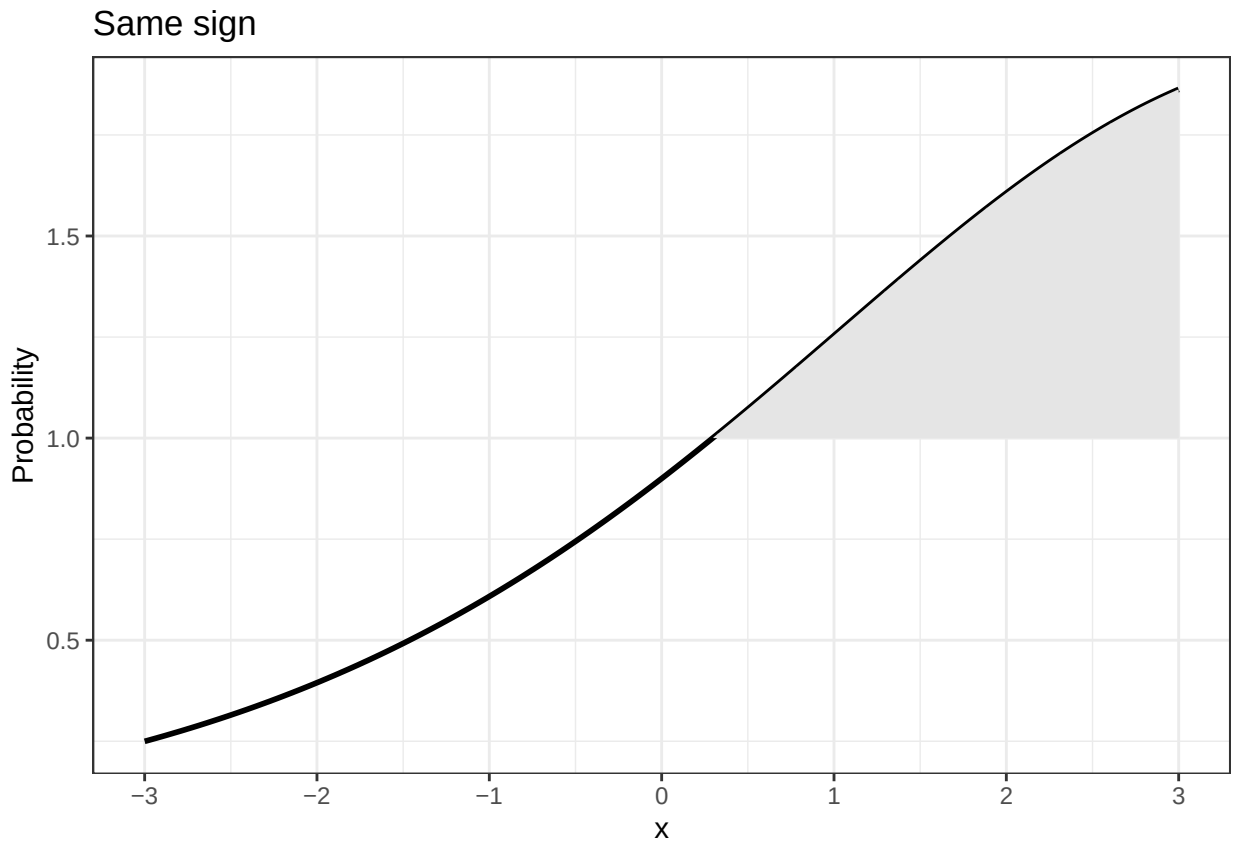
```
pnorm(TFPplus, lower.tail=FALSE)
```

```
## [1] 0.3836729
```

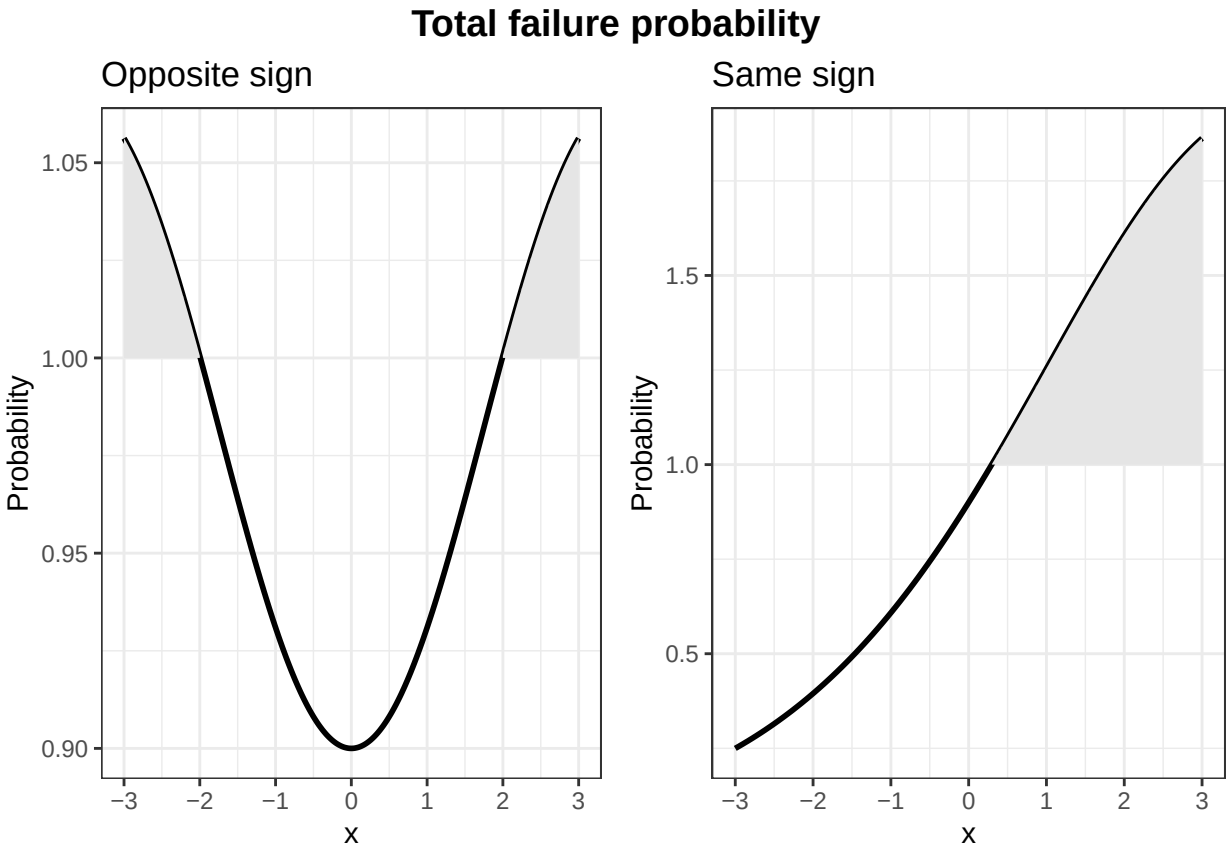
The ggplot version of this plot is now combined with the previous (opposite sign) ggplot.

```
tmp <- tibble(x=xseq, tfp=TFP(xseq, p1=p1, p2=p2, beta1=beta1, beta2=beta2))

ggp2 <- tmp %>%
  ggplot(aes(x = x, y = tfp)) +
  geom_line(size = 1) +
  labs(
    title = "Same sign",
    x = "x",
    y = "Probability"
  ) +
  scale_x_continuous(breaks = -3:3) +

  # We need two ribbons to avoid line joining them
  geom_ribbon(
    data = tmp %>% filter(tfp >= 1 & x >= 0),
```

```
    aes(x = x, ymin = 1, ymax = tfp),
    fill = "grey90"
  ) +
  theme_bw()
ggp2
```




```
library(ggpubr)
figure <- ggarrange(ggp1, ggp2, common.legend=TRUE)
annfigure <- annotate_figure(figure,
                             top = text_grob("Total failure probability", face = "bold", size = 14))
annfigure
```




```
# ggsave("TFPplots.pdf")
```

We can make all this into a function that finds the probability of a total failure probability exceeding one, for given values of p_1, p_2, β_1 and β_2.

```
pTFPexc1 <- function(p1, p2, beta1, beta2, boundary=10) {
  TFPmin1 <- function(x, p1, p2, beta1, beta2)
    1 - (1-p1)^(exp(beta1*x)) + 1 - (1-p2)^(exp(beta2*x)) - 1
  pTFPplus <- pTFPmin <- 0
  # Find root for positive x, will work unless beta1 and beta2 are both negative
  if (!(beta1<0 & beta2<0)) {
    TFPplus <- uniroot(TFPmin1, c(0, boundary), p1=p1, p2=p2, beta1=beta1, beta2=beta2)$root
    pTFPplus <- pnorm(TFPplus, lower.tail=FALSE)
  }
  # Find root for negative x, will work unless beta1 and beta2 are both positive
  if (!(beta1>0 & beta2>0)) {
    TFPmin <- uniroot(TFPmin1, c(-boundary, 0), p1=p1, p2=p2, beta1=beta1, beta2=beta2)$root
    pTFPmin <- pnorm(TFPmin, lower.tail=TRUE)
  }
  return(pTFPplus + pTFPmin)
}
pTFPexc1(p1=0.45, p2=0.45, beta1=0.5, beta2=-0.5, boundary=10)
```

```
## [1] 0.04630708
```

We can now check our intuition, which says that $\text{TFP}(x) > 1$ will happen more frequently for higher p_1 and p_2, and for larger (in absolute value) β_1 and β_2.

```
pTFPexc1(p1=0.46, p2=0.46, beta1=0.5, beta2=-0.5, boundary=10)
```

```
## [1] 0.07637164
```

```
pTFPexc1(p1=0.45, p2=0.45, beta1=0.6, beta2=-0.6, boundary=10)
```

```
## [1] 0.09681915
```

So, in a situation where Fine-Gray models have been fitted for cause 1 and cause 2, if the estimated cumulative incidences at some time point t are 0.45 for both causes, and the estimated β_1 and β_2 are equal to 0.6 and -0.6, respectively, then for 10% of the patients the model-based total failure probability (TFP) will exceed one.

If β_1 and β_2 have the same sign the probability of TFP exceeding one is a lot larger for the same p_1 and p_2.

```
pTFPexc1(p1=0.45, p2=0.45, beta1=0.5, beta2=0.5, boundary=10)
```

```
## [1] 0.3836729
```

What is perhaps less obvious is that, while keeping the total baseline cumulative incidence $p_1 + p_2$ constant, and the total effect size $|\beta_1| + |\beta_2|$ constant, the case $p_1 = p_2$ and $\beta_1 = -\beta_2$ is the most favorable in keeping the TFP below 1.

```
# Base case
pTFPexc1(p1=0.45, p2=0.45, beta1=0.5, beta2=-0.5, boundary=10)
```

```
## [1] 0.04630708
```

```
# Making p1 unequal to p2
pTFPexc1(p1=0.46, p2=0.44, beta1=0.5, beta2=-0.5, boundary=10)
```

```
## [1] 0.04656941
```

```
# Making beta1 unequal to -beta2
pTFPexc1(p1=0.45, p2=0.45, beta1=0.51, beta2=-0.49, boundary=10)
```

```
## [1] 0.04779931
```

```
# Making p1 unequal to p2 and beta1 unequal to -beta2
pTFPexc1(p1=0.46, p2=0.44, beta1=0.51, beta2=-0.49, boundary=10)
```

```
## [1] 0.04940599
```

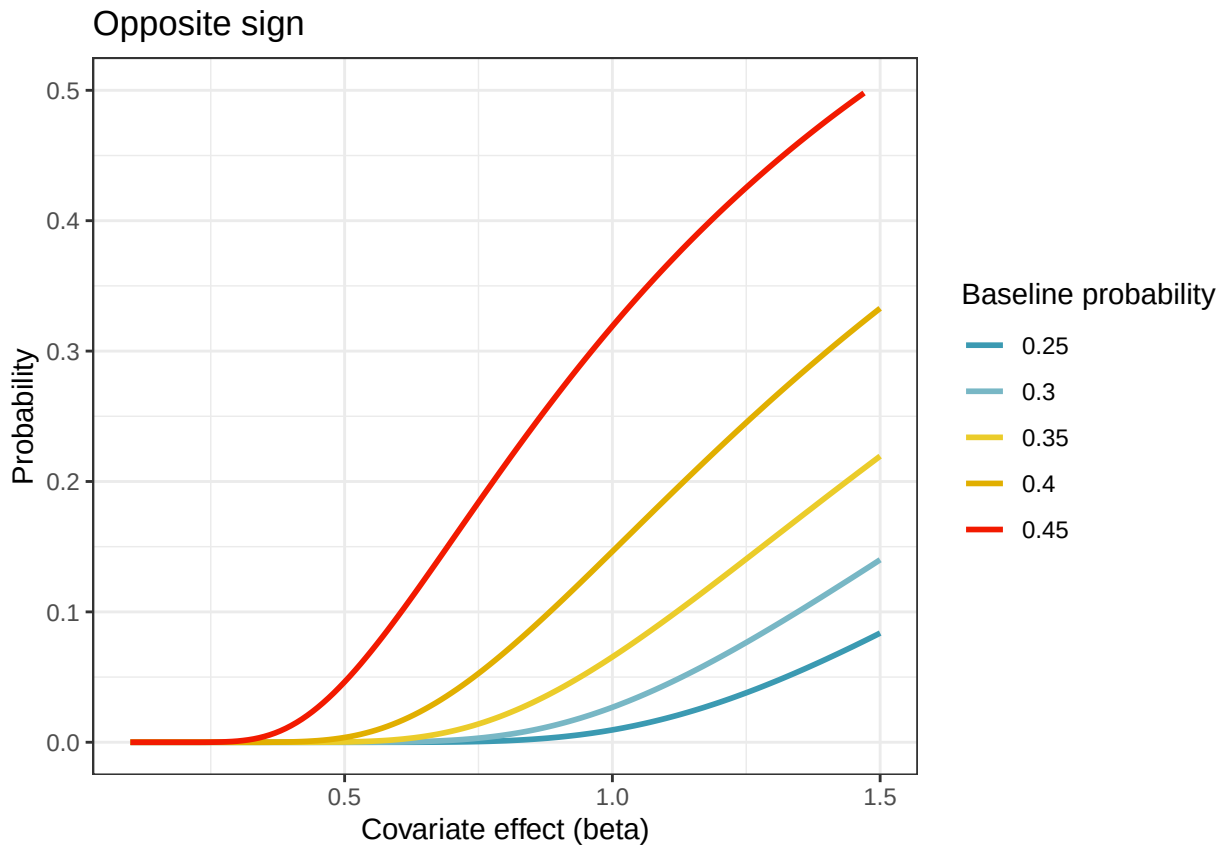
In showing that TFP exceeding one is a problem we will therefore restrict to $p_1 = p_2$ and $\beta_1 = -\beta_2$; changing to $p_1 \neq p_2$ or $\beta_1 \neq -\beta_2$ will increase the probability of TFP exceeding one, subject to the total baseline cumulative incidence $p_1 + p_2$, and the total effect size $|\beta_1| + |\beta_2|$ staying the same. That simplifies life, because we only have to keep track of two variables, $p = p_1 = p_2$ and $\beta = |\beta_1| = -|\beta_2|$.

I will now record the probabilities of TFP exceeding one for a set of baseline probabilities $p = 0.25, 0.3, \ldots, 0.45$, and a range of β values, ranging from 0.1 to 2, and make a nice plot of it.

```
pseq <- seq(0.25, 0.45, by=0.05)
betaseq <- seq(0.1, 1.5, by=0.01)
np <- length(pseq)
nbeta <- length(betaseq)
# Store results in res
res <- matrix(NA, nbeta, np)
for (i in 1:np) {
  p <- pseq[i]
  for (j in 1:nbeta) {
    beta <- betaseq[j]
```

```
    boundary <- ifelse(beta>0.3, 10, 100) # trial and error
    res[j, i] <- pTFPexc1(p1=p, p2=p, beta1=beta, beta2=-beta, boundary=boundary)
  }
}
# Prepare for plotting
library(wesanderson)
dfres <- data.frame(beta=rep(betaseq, np), p=rep(pseq, each=nbeta), prob=as.vector(res))
dfres$p <- as.factor(dfres$p)
# Plot
ggp <- ggplot(data = dfres,
              aes(x=beta, y=prob, col=p)) +
  geom_line(size=1) +
  labs(title = "Opposite sign",
       x = "Covariate effect (beta)",
       y = "Probability") +
  ylim(0, 0.5) +
  scale_x_continuous(breaks=c(0, 0.5, 1, 1.5)) +
  scale_color_manual(name="Baseline probability", values=wes_palette(n=5, name="Zissou1")) +
  theme_bw()
print(ggp)
```

```
## Warning: Removed 3 row(s) containing missing values (geom_path).
```

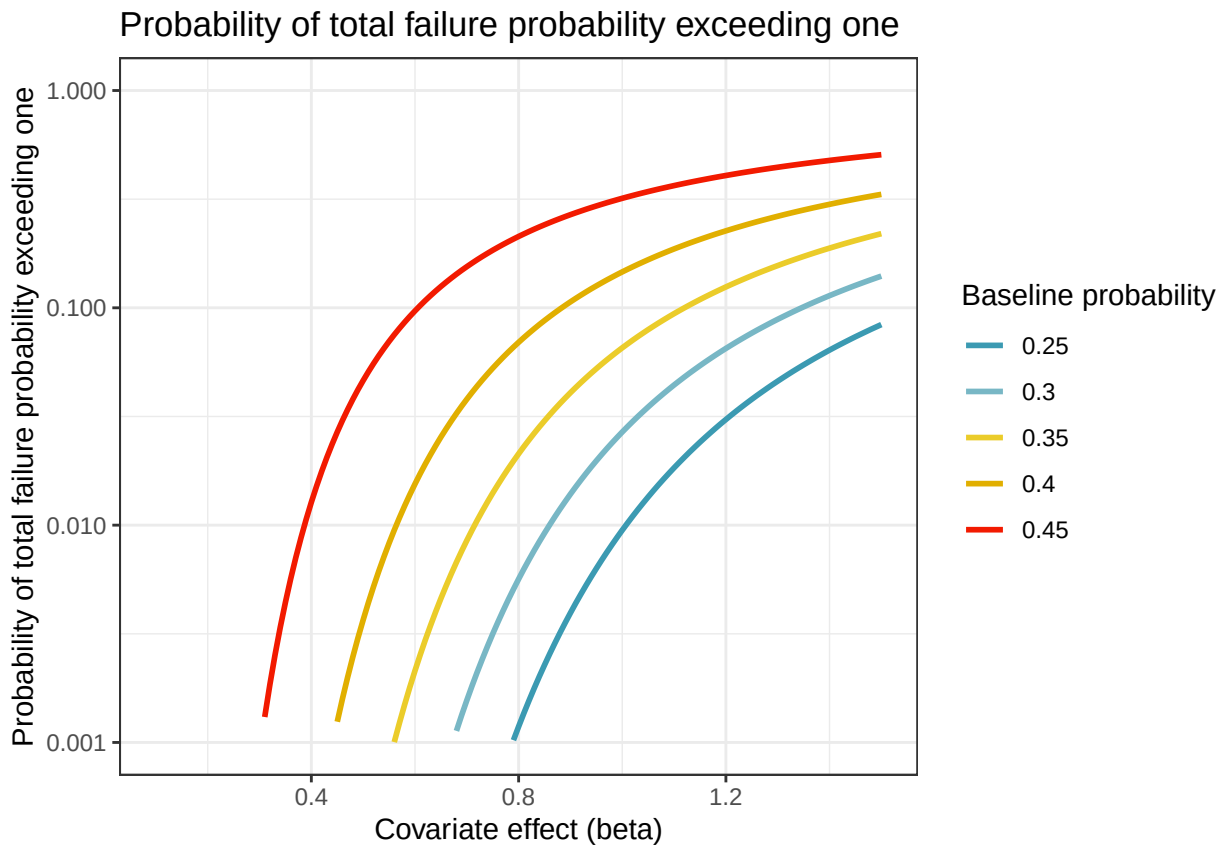



```
ggpoppsign <- ggp
```

Here is the same plot, but then with logarithmic scale on the y-axis.

```
ggp <- ggplot(data = dfres,
              aes(x=beta, y=prob, col=p)) +
  geom_line(size=1) +
  labs(title = "Probability of total failure probability exceeding one",
       x = "Covariate effect (beta)",
       y = "Probability of total failure probability exceeding one") +
  scale_color_manual(name="Baseline probability", values=wes_palette(n=5, name="Zissou1")) +
  scale_y_log10(limits=c(0.001, 1)) +
  theme_bw()
print(ggp)
```

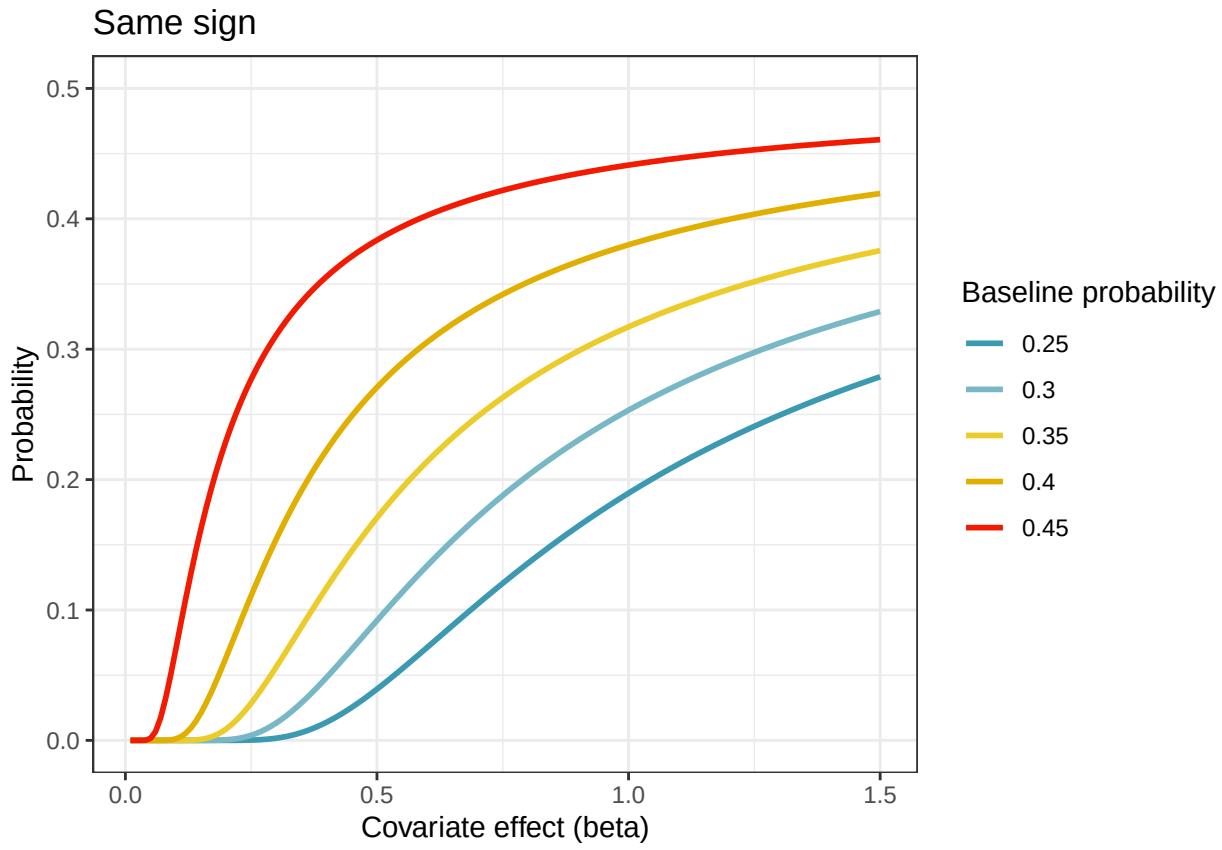
```
## Warning: Removed 229 row(s) containing missing values (geom_path).
```



It may be interesting to do the same thing, now for β_1 and β_2 having the same sign.

```
betaseq <- seq(0.01, 1.5, by=0.01)
nbeta <- length(betaseq)
res <- matrix(NA, nbeta, np)
for (i in 1:np) {
  p <- pseq[i]
  for (j in 1:nbeta) {
    beta <- betaseq[j]
    boundary <- ifelse(beta>0.3, 10, 100) # trial and error
    res[j, i] <- pTFPexc1(p1=p, p2=p, beta1=beta, beta2=beta, boundary=boundary)
  }
}
dfres <- data.frame(beta=rep(betaseq, np), p=rep(pseq, each=nbeta), prob=as.vector(res))
```

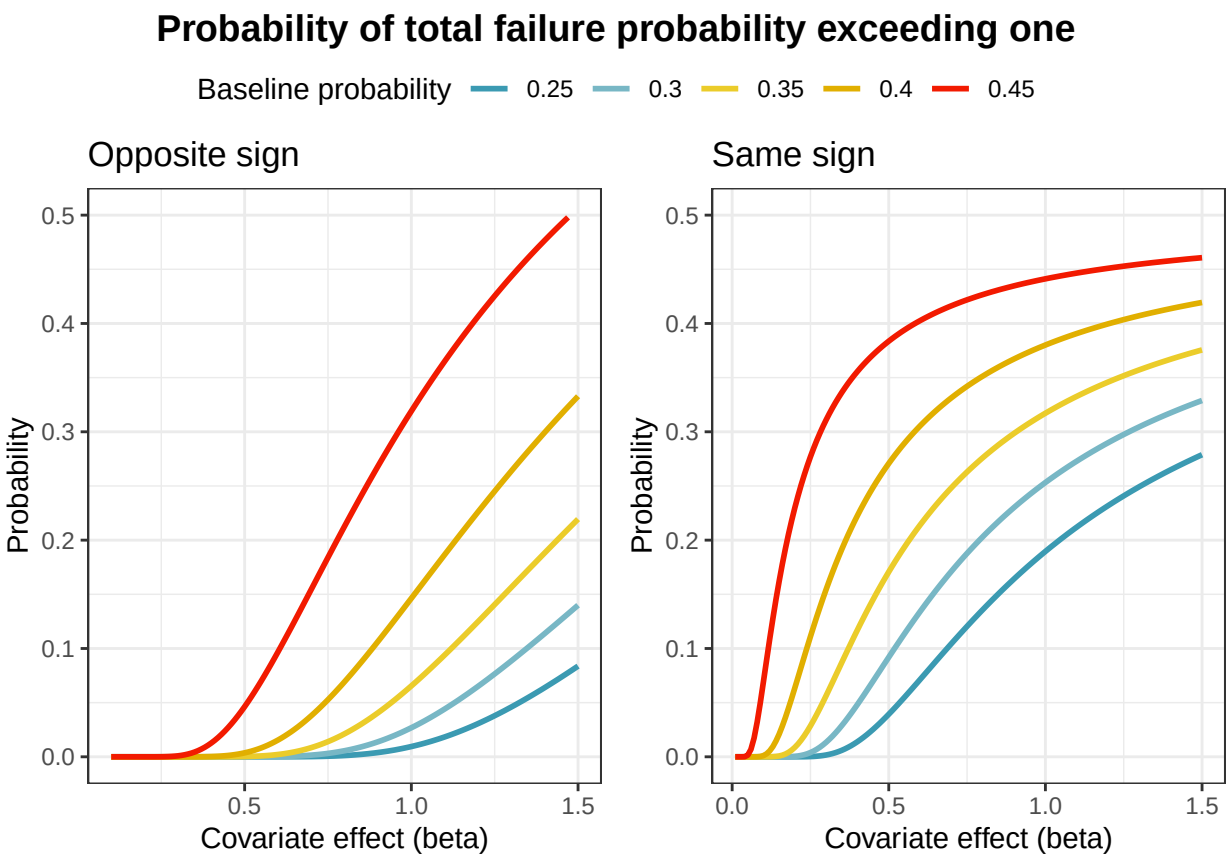
```r
dfres$p <- as.factor(dfres$p)
# Plot
ggp <- ggplot(data = dfres,
              aes(x=beta, y=prob, col=p)) +
  geom_line(size=1) +
  labs(title = "Same sign",
       x = "Covariate effect (beta)",
       y = "Probability") +
  ylim(0, 0.5) +
  scale_x_continuous(breaks=c(0, 0.5, 1, 1.5)) +
  scale_color_manual(name="Baseline probability", values=wes_palette(n=5, name="Zissou1")) +
  theme_bw()
print(ggp)
```




```r
ggpsamesign <- ggp

figure <- ggarrange(ggpoppsign, ggpsamesign, common.legend=TRUE)
```

```
## Warning: Removed 3 row(s) containing missing values (geom_path).

## Warning: Removed 3 row(s) containing missing values (geom_path).
```

```r
annfigure <- annotate_figure(figure,
                             top = text_grob("Probability of total failure probability exceeding one",
annfigure
```




```
# ggsave("TFPexc1plots.pdf")
```