The impact of the number and the size of clusters on prediction performance of the stratified and the conditional shared gamma frailty Cox proportional hazards models

R code for the illustrative case studies

Daniele Giardiello[*] Edoardo Ratti[†] Peter C. Austin[‡]

Contents

[*]University of Milan-Bicocca, daniele.giardiello1@gmail.com
[†]University of Milan-Bicocca
[‡]Institute for Clinical Evaluative Sciences, ICES

Overview

We illustrate the analysis of the case studies accompanying the simulation study of the manuscript *"The impact of the size and the number of clusters on prediction performance of the stratified and the conditional shared gamma frailty Cox proportional hazards models."*

Some useful notations

Two type of predictions may be possible using frailty:

- Conditional: predictions given the frailty/random effects
  For the shared gamma frailty:

  $$S_{ij}(t) = exp[-z_i e^{x_{ij}\beta} H_o(t)] = exp[-H_0(t)e^{x_{ij}\beta+\mu_j}] = exp[-z_j H(t)] = exp[-H(t)e^{\mu_j}]$$

  where

  $$Z_j \sim \Gamma(\theta, \theta)$$

  and

  $$\mu_j = log(Z_j)$$

  where $i = 1, ..., n_j$ represents the i^{th} subject and $j = 1, ..., N$ represents the j^{th} cluster.
  The measure of within-cluster correlation is Kendall's τ. For the shared gamma frailty model, $\tau = \sigma^2/(\sigma^2+2)$ where σ^2 is the variance of Z_j. The higher is the variance, the higher within-cluster correlation (i.e., homogeneity). Note that $Z_j \sim \Gamma(\theta, \theta)$ then the variance is $\sigma^2 = 1/\theta$ then $\tau = 1/(\theta+2)$.
  For details see Hougaard *Analysis of Multivariate Survival Data (chapter 4)* link here, and Collett *Modelling Survival Data in Medical Research*, $4^{th} edition, (chapter 10)$ link here.

- Marginal: integrating over frailty/random effects
  For the shared gamma frailty the marginal predictions may be used when the new individual does not belong to any cluster used to develop the model or to estimate predictions integrating over all frailty Conditional predictions can be used when the new individual belongs to a cluster used to develop the model.

$$S_i^*(t) = [1 + \theta^{-1} e^{\beta x_{ij}} H_0(t)]^{-\theta}$$

Load packages

The following libraries are needed to achieve the following goals, if you have not them installed, please use install.packages(' ') (e.g. install.packages('survival')) or use the user-friendly approach if you are using RStudio.

```
# Use pacman to check whether packages are installed, if not load
if (!require("pacman")) install.packages("pacman")
library(pacman)

pacman::p_load(
  rio,
  stringr,
  knitr,
  kableExtra,
  Hmisc,
  lattice,
  riskRegression,
  polspline,
  gtsummary,
  tidyverse,
  grid,
  gridExtra,
  webshot,
  plotly,
  ggplot2,
  ggpubr,
  patchwork,
  rms,
  webshot2,
  parfm,
  frailtyHL
)
```

Import useful functions

We import useful functions

```
# Useful functions
source(here::here("Functions/predict.coxph.gammafrail.R"))


#' @description
#'
#' Function to estimate conditional and marginal
#' survival probabilities at specified time horizons
#' for shared Gamma frailty Cox proportional hazards models
#'
#' @author Daniele Giardiello
#' @param model cox model using survival::coxph including frailty gamma term
#' The event of interest must have the indicator variable equal to 1.
#' The model survival::coxph() must specify y = T survival::coxph(..., y = T)
#' The user must define a reference value
#' for all predictors, especially the continuous predictors
#' Example: age50 = age - 50 or age_c = age - mean(age)
#' Then "the baseline" corresponds to a subject with
#' the specified reference predictors values.
#' Ties/methods must be "breslow".
#' @param newdata newdata (for validation).
#' This should have the same variables of the model.
#' Predictors with levels should be coded
#' with exactly the same way and levels of the predictors
#' with factors used to develop the model
#' @param cluster cluster variable. Only one variable possible
#' @param times prediction horizon time(s)
#' @examples
#' # example code
#' data(lung, package = "survival")
```

```r
#' lung$age_c <- lung$age - mean(lung$age)
#' cox_frailty_lung <- survival::coxph(Surv(time, status) ~ age_c + frailty(inst,
#' distribution = "gamma"),
#' data = lung,
#' x = TRUE,
#' y = TRUE,
#' model = TRUE,
#' ties = "breslow")
#' predict.coxph.gammafrail(model = cox_frailty_lung,
#' newdata = lung_sel,
#' cluster = "inst",
#' times = c(100, 200, 300))
#' @references Collett D - Modelling Survival data in Medical Research,
#' 4th edition chapter 10 (formula 10.9 and 10.12)


# **Disclaimers**
# 1. The current functions provided the conditional estimated
# predictions using the shared gamma frailty Cox model using
# survival::coxph.
# Left-truncation, non-linear terms and stratification terms
# have not been implemented and tested yet.

# 2. Marginal prediction estimation are still
# experimental. We do not advice to use it yet.
# Your suggestions to improve the function are kindly and warmly welcome.

predict.coxph.gammafrail <- function(model,
                                     newdata,
                                     cluster,
                                     times) {

  # sanity check message warning
  on.exit(base::message("
  Sanity check warning:
  please center all variables before fitting the model or assign reference values
                for continuous variable.
                For example age50 = age - 50.
                newdata should contain the centered/referenced values"))

  # Stopping messages
  if (!inherits(model, "coxph"))
    stop("`model` must be a survival::coxph fit.
         The model must specify y = T and Breslow method.
         Please center all variables before fitting the model or assign reference values
         for continuous variable. For example age50 = age - 50")

  if (missing(cluster))
    stop("Please supply the name of the cluster variable (character).")

  if (!is.character(cluster) || length(cluster) != 1L)
    stop("`cluster` must be a single character string giving the cluster column name.")
```

```r
if (!cluster %in% names(newdata))
  stop("`cluster` not found in `newdata`.")


# Start function ----
# Step0:
# Formula with all terms including cluster as it is a fixed covariate
frm_allterms <- reformulate(base::all.vars(model$formula[[3]]))

# Select the complete cases of fixed + frailty terms
# of the development data
# (same data used to develop the model)
# The argument survival::coxph(..., x = T) provides
# the development data
# NOTE: the development data is necessary to
# estimate the "baseline" (cumulative) hazard H0 later

devdata <- stats::model.frame(frm_allterms,
                              data = base::eval(model$call$data,
                                                envir = environment(formula(model))),
                              # NOTE: data.frame(model$x) did not
                              # provide the same variable names
                              # (e.g. frailty(id_cluster) instead of id_cluster)
                              # Alternative: add devdata as a new argument of the
                              # function.
                              # The alternative option is probably the most
                              # intuitive
                              na.action = na.omit) # not sure if necessary but safer


# Select only the complete cases of fixed + frailty terms
# of the newdata

newdata <- stats::model.frame(frm_allterms,
                              data = newdata,
                              na.action = na.omit)

# Step 1: survfit for
# a baseline cumulative hazard / survival

# Survfit of the coxph with frailty
# sf_model <- survival::survfit(model)

# NOTE: the survival::survfit with frailty does not accept newdata
# thus, it is important to center all variables or give them a reference.
# In this setting survfit$cumhaz will directly represent our baseline

# NOTE: [20251029]: survfit() with frailty with no newdata
# seems not to include the frailty terms in the "baseline"
# cumulative hazard estimation for a reference subject

# Step 1: linear predictors of fixed effects
# Estimate linear predictors only
# for fixed-effect effects without frailties
# This will be also useful later when
# marginal predictions are estimated


# Since with clusters size < vs >= 5
# frailty terms are treated as coefficients or not,
# I prefer calculate lp for the only fixed terms only manually

# lp calculation for marginal
# save all model coefficients
# for cluster < 5, frailty terms are included in coefficients (not practical)
coefs <- model$coefficients

# Remove frailty named "gamma"
# NOTE/WARNING:
# if a variables called gamma it must be a problem
# to be adjusted
coefs_fixed <- coefs[!base::grepl("gamma", names(coefs))]

# create formula of fixed terms only
terms_vec <- attr(terms(model$formula),
                  "term.labels")

# Remove specials using grep/grepl
# NOTE: working in progress
specials <- c("frailty", "strata", "cluster", "offset", "tt")
fixed_terms <- terms_vec[!grepl(paste(specials, collapse = "|"), terms_vec)]


formula_fixed <- stats::reformulate(fixed_terms,
                                    response = NULL)

# create design matrix of fixed term only
# of the new data
X_fixed <- model.matrix(formula_fixed,
                        newdata)

X_fixed <- X_fixed[, colnames(X_fixed) != "(Intercept)"]

# estimate linear predictor X*beta fixed term only
lp_fixed <- X_fixed %*% cbind(coefs_fixed)

# this should be equivalent to this (when cluster > 5)
# lp_nofrail <- cbind(predict(model,
#                             newdata = newdata,
#                             type = "lp"))

# create design matrix of fixed term only
# of the development data
# (data used to develop the model)
# useful to estimate the "baseline" cumulative hazard
```

```
X_fixed_dev <- model.matrix(formula_fixed,
                            devdata)

X_fixed_dev <- X_fixed_dev[, colnames(X_fixed_dev) != "(Intercept)"]

# estimate linear predictor X*beta fixed term only
lp_fixed_dev <- X_fixed_dev %*% cbind(coefs_fixed)


# Step 2: linear predictors for frailty terms

# NOTE/WARNING:
# for cluster < 5 predict.coxph(model, newdata) includes also frailty
# and frailty terms must not be included for marginal predictions.
# This leads to wrong estimations of marginal predictions
# and the corresponding performance metrics in riskRegression::Score()

# For frailty with not model$frail (where clusters < 5)
# add the model$frail and keep only fixed-effect coefficients
if(is.null(model$frail)) {
  coefs_model <- model$coefficients
  model$frail <- unname(coefs_model[base::grep("gamma", names(coefs_model))])
}

# Force cluster as a factor in the newdata
# number of clusters in the newdata

# otherwise a factor
# NOTE: for newdata with one cluster, factor does not work
# newdata[[cluster]] <- base::as.character(newdata[[cluster]])

# newdata
 newdata[[cluster]] <- base::factor(newdata[[cluster]],
                                    levels = unique(newdata[[cluster]]))

# development data
 devdata[[cluster]] <- base::factor(devdata[[cluster]],
                                    levels = unique(devdata[[cluster]]))

# newdata[[cluster]] <- base::factor(newdata[[cluster]],
#                                     levels = 1:length(model$frail))

# Create the design matrix for frailty terms
all_terms <- base::all.vars(model$formula[[3]])
frailty_terms <- c(-1, all_terms[all_terms == cluster])

# design matrix for the frailty terms
# of the new data
X_frail <- stats::model.matrix(stats::reformulate(frailty_terms),
                               response = NULL,
                               data = newdata)

# design matrix for the frailty terms
```

```r
# of the development data (data used to develop the model)
# useful to estimate the "baseline" cumulative hazard H0
X_frail_dev <- stats::model.matrix(stats::reformulate(frailty_terms),
                                   response = NULL,
                                   data = devdata)

# Frailty terms linear predictors
# new data
lp_frail <- X_frail %*% cbind(model$frail)

# Frailty terms linear predictors
# development data
lp_frail_dev <- X_frail_dev %*% cbind(model$frail)

# Total linear predictor
# new data
lp_total <- lp_fixed + lp_frail

# Total linear predictor
# development data
lp_total_dev <- lp_fixed_dev + lp_frail_dev

# Step 3:
# estimate the "baseline" cumulative hazard
# for a reference subject
# NOTE: we use the development data
# NOTE: recalibration might be possible using
# the linear predictors (lp) of the development data
# but the risk setting of the newdata (i.e., validation)
# It might be considered as future developments

risk_dev <- exp(lp_total_dev) # exp(lp + frailty)

# Unique event times of the
# model fitted using the development data

model_times <- model$y[, "time"] # event time for development data
model_status <- model$y[, "status"] # event indicator for development data

# NOTE: the object model must specify y = T
# Example: survival::coxph(..., y = T)
# NOTE: status == 1 must be the event of reference

# Extract the unique event time
# from the development data
# used to develop the model
unique_event_times <- sort(unique(model_times[model_status == 1]))
n_times <- length(unique_event_times)

# Initialize vectors
h0 <- numeric(n_times)    # incremental baseline hazard
d_i <- numeric(n_times)   # number of events at each time
```

```r
for (i in seq_along(unique_event_times)) {
  t <- unique_event_times[i]
  d_i[i] <- sum(model_status[model_times == t])

  # Risk set: all individuals still at risk at time t
  risk_set <- which(model_times >= t)

  # Breslow baseline hazard
  h0[i] <- d_i[i] / sum(risk_dev[risk_set])
}

# Cumulative "baseline" hazard
# Create the sf model
H0 <- cumsum(h0)

# Cumulative hazards
# at fixed time horizons
# H_0(t)
cumhaz0_thor <- stats::approx(
  x = unique_event_times,
  y = H0,
  yleft = 0, # y values below min(x). Cumulative hazard zero if below min(time)
  yright = NA, # y values above max(x). Cumulative hazard NA if above max(time)
  xout = times,
  method = "constant",
  f = 0,
  rule = 1)$y

# NOTE:
# stats:approx should work fine since
# all values from survival::survfit are unique
# for the event time of development data

# H(t) using the newdata (i.e., validation data)
cumhaz_thor <- exp(lp_total) %*% cumhaz0_thor

# conditional S(t)
S_cond_t <- exp(-cumhaz_thor)

# marginal S(t)
# NOTE: marginal predictions needs
# to be further investigated
# to check if they are corrected
#
# NOTE/QUESTION
# does the marginal use the baseline hazard including the frailty terms?

theta_hat <- model$history[[1]]$theta
S_marg_t <- (1 + theta_hat * (exp(lp_fixed) %*% cumhaz0_thor))^(-1/theta_hat)

# Output
# NOTE: to be better defined
res <- list("conditional" = S_cond_t,
              "marginal" = S_marg_t)

  return(res)
  # Report only conditional predictions
  # return(S_cond_t)

}


# Other ad-hoc functions
# Score function with TryCatch
Score_tryCatch <- function(predictions,
                           df,
                           t_hors,
                           frm) {
  tryCatch(

  {
    sc <-
      riskRegression::Score(
        list("prediction_model" = predictions),
        data = df,
        formula = frm,
        times = t_hors,
        metrics = "auc",
        summary = "ipa")

    # Merge AUC + IPA into a single tidy table
    score_metrics <-
      base::merge(
        sc$AUC$score[, c("model", "times", "AUC")],
        sc$Brier$score[, c("model", "times", "IPA")],
        by = c("model", "times"),
        all = TRUE)

    score_metrics_models <-
      score_metrics %>%
      dplyr::filter(model %nin% c("Null model"))

    score_metrics_models},

  error = function(msg) {
        message(paste("Prediction not possible"))

    base::expand.grid(
      times = t_hors,
      model = c("prediction_model")) %>%
      base::transform(AUC = NA_real_,
                      IPA = NA_real_)
  })

}
```

```
# smoothed calibration function

calhaz_map <- function(time,
                       status,
                       covariates,
                       t_hors,
                       predictions) {

  tryCatch(
    {

  # Hazard regression
  calhaz_mod <- polspline::hare(data = time,
                                delta = status,
                                cov = covariates)

  # estimated cumulative probability
  pred_calhaz <- polspline::phare(t_hors,
                                  cov = covariates,
                                  fit = calhaz_mod)

  # ICI/E50/E90
  ICI <- mean(abs(pred_calhaz - predictions),
              na.rm = T)

  E50 <- median(abs(pred_calhaz - predictions),
                na.rm = T)

  E90 <- quantile(abs(pred_calhaz - predictions),
                  probs = .90,
                  na.rm = T)

  res <- c(ICI, E50, E90)
  res},

  error = function(msg) {
    message(paste("Calibration not possible"))
    rep(NA, 3)})

}
```

Import data

Insem data can be imported using the `parfm` R package. We also saved insem data in our repository in the .rds format.

```
# Import from the repository
cimbtr_P5300 <- readRDS(here::here("Data/cimbtr_P5300.rds"))
bladder <- readRDS(here::here("Data/bladder.rds"))
insem <- readRDS(here::here("Data/insem.rds"))

# Import from xxx package
# data(insem,
#      package = "parfm")
#
# data(bladder,
#      package = "survival")
```

First case study: mortality after transplant in myelodysplastic syndrome patients

1. Descriptive statistics We provide some descriptive statistics.

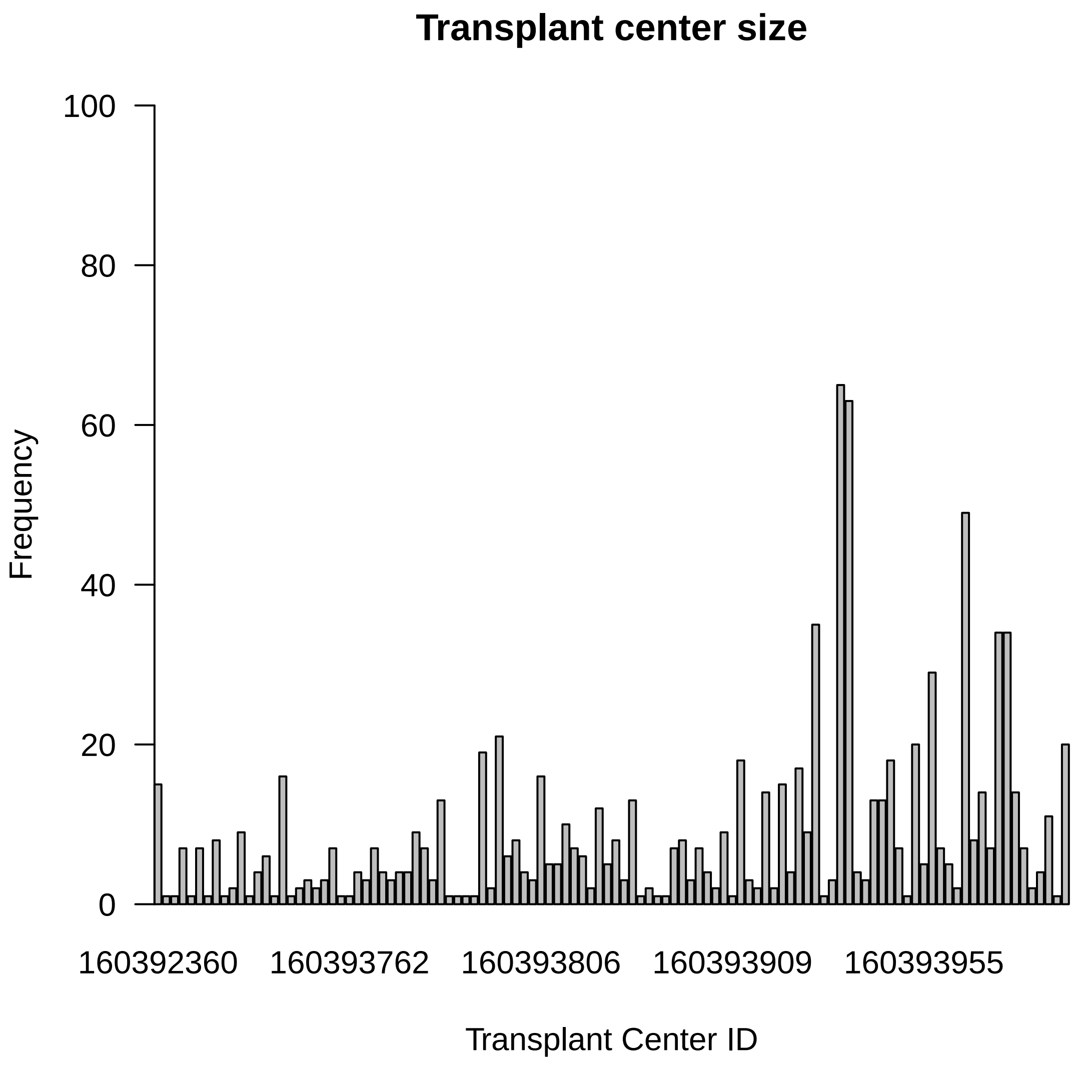
```r
# Import data
# cimbtr_P5300 <- rio::import(here::here("Data/P-5300 (CK18-01)/pub.sas7bdat"))

df_cimbtr_pred <-
  cimbtr_P5300 %>%
  dplyr::select(
    pseudocrid, # patient ID
    pseudocenterid, # center ID
    intxsurv, # time from HCT to death (months)
    dead, # death (any cause)
    age, # age at transplant
    sex, # sex
    racenewgp, # race (to be adjusted)
    ethgp, # race + ethnicity (to be adjusted)
    kps, # Karnofsky
    ncgp, # CIBMTR MDS score at HCT
    # See: https://pmc.ncbi.nlm.nih.gov/articles/PMC4047501/
    ipsspr, # IPSS score (continuous)
    ipssrpr, # IPSS score categorical

    dnrgp, # donor type and age
    dnrage) %>%

  dplyr::mutate(

    # Sex
    sex = factor(sex,
                 levels = c(1, 2),
                 labels = c("male", "female")),

    # Race
    racenewgp = factor(racenewgp,
                       levels = c(1, 2, 3, 4, 5, 6),
                       labels = c("Caucasian",
                                  "African American",
                                  "Asian",
                                  "Pacific Islander",
                                  "Native American",
                                  "Other")),
    # Karnofsky
    kps = factor(kps,
                 levels = c(0, 1),
                 labels = c("10-80", "90-100")),

    # Donor type
    dnrgp = factor(dnrgp,
                   levels = c(1, 2, 3),
                   labels = c("Unrelated",
                              "Related",
                              "Cord blood")),
    # MDS scoring system
    ncgp = factor(ncgp,
                  levels = c(1, 2, 3, 4),
                  labels = c("Low",
                             "Intermediate",
                             "High",
                             "Very high")),

    # IPSS scoring categorical
    ipssrpr = factor(ipssrpr,
                     levels = c(1, 2, 3, 4, 5),
                     labels = c("Very low",
                                "Low",
                                "Intermediate",
                                "High",
                                "Very high")))


# Selection of only essential predictors
df_cimbtr_sel <-
  df_cimbtr_pred %>%
  dplyr::select(
    pseudocrid, # patient ID
    pseudocenterid, # center ID
    intxsurv, # time from HCT to death (months)
    dead, # death (any cause)
    ncgp, # CIBMTR MDS score at HCT
    # See: https://pmc.ncbi.nlm.nih.gov/articles/PMC4047501/
    ipsspr, # IPSS score (continuous)
    ipssrpr # IPSS score categorical
    ) %>%
  dplyr::filter(!is.na(ncgp))

# For table 1

cimbtr_tab_01 <-
  df_cimbtr_pred %>%
  dplyr::select(
    age,
    sex,
    kps,
    dnrgp,
    ncgp,
    ipsspr,
    ipssrpr)

label(cimbtr_tab_01$age) <- "Age, years"
label(cimbtr_tab_01$sex) <- "Sex"
label(cimbtr_tab_01$kps) <- "Karnofsky score"
label(cimbtr_tab_01$dnrgp) <- "Donor type"
label(cimbtr_tab_01$ncgp) <- "CIBMTR-MDS score"
label(cimbtr_tab_01$ipsspr) <- "IPPS score"
label(cimbtr_tab_01$ipssrpr) <- "Revised IPSS"
```

```
gtsummary_cimbtr_table1 <-
  gtsummary::tbl_summary(
    data = cimbtr_tab_01 ,
    label = list(
      age ~ "Age, years",
      sex ~  "Sex",
      kps ~ "Karnofsky score",
      dnrgp ~ "Donor type",
      ncgp ~ "CIBMTR-MDS score",
      ipsspr ~ "IPPS score",
      ipssrpr ~ "Revised IPSS"),
    type = all_continuous() ~ "continuous2",
    statistic = all_continuous() ~ c(
      "{mean} ({sd})",
      "{median} ({min}, {max})"))


gtsummary_cimbtr_table1
```

Characteristic	**N = 1,503**
Age, years	
Mean (SD)	54 (16)
Median (Range)	59 (0, 77)
Sex	
male	905 (60%)
female	598 (40%)
Karnofsky score	
10-80	416 (34%)
90-100	810 (66%)
Unknown	277
Donor type	
Unrelated	1,156 (77%)
Related	181 (12%)
Cord blood	166 (11%)
CIBMTR-MDS score	
Low	120 (13%)
Intermediate	516 (54%)
High	273 (29%)
Very high	41 (4.3%)
Unknown	553
IPPS score	
1	149 (13%)
2	586 (50%)
3	366 (31%)
4	61 (5.2%)
Unknown	341
Revised IPSS	
Very low	120 (10%)
Low	292 (25%)
Intermediate	345 (30%)
High	223 (19%)

Characteristic	**N = 1,503**
Very high	169 (15%)
Unknown	354

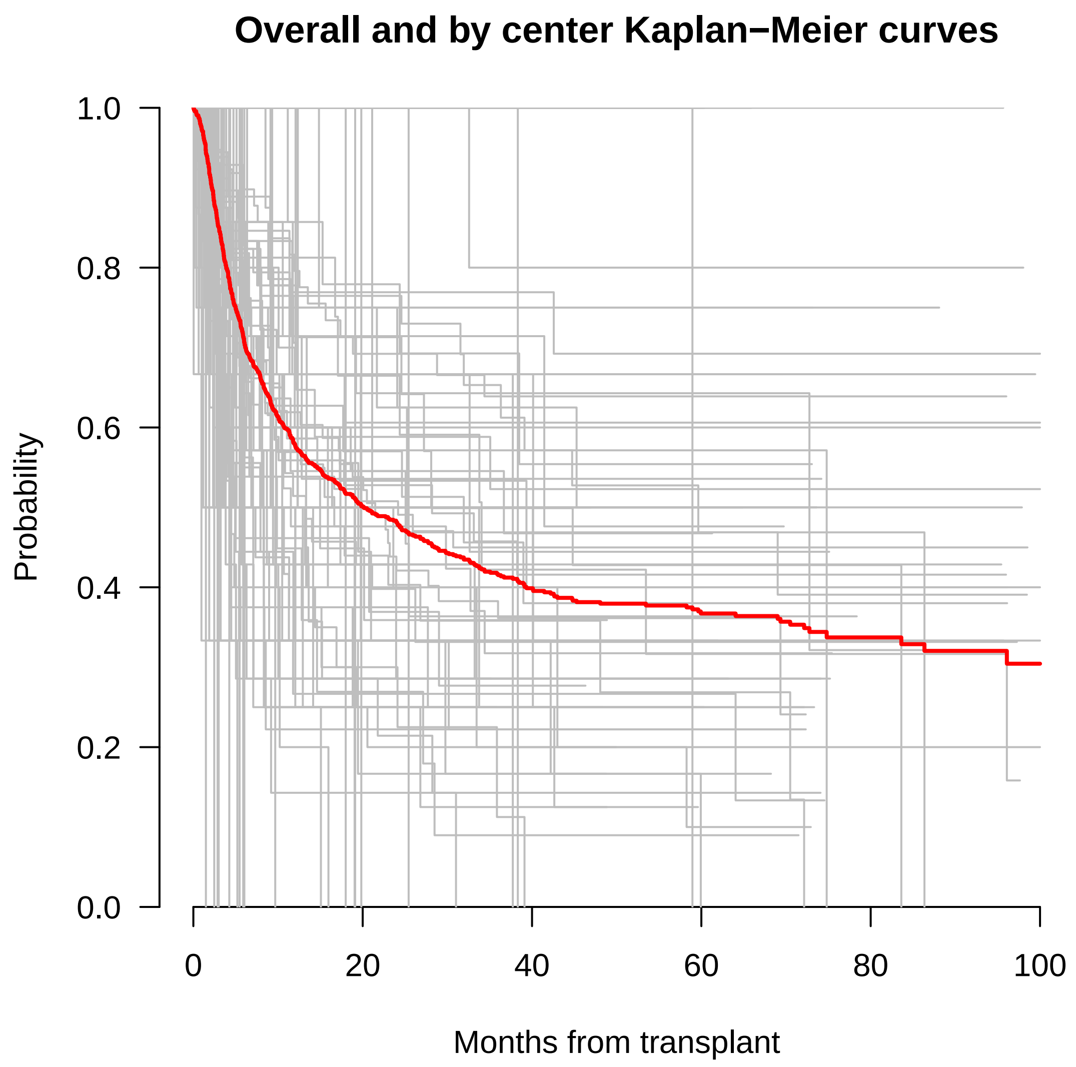
```r
# Barplot
par(xaxs = "i",
    yaxs = "i",
    las = 1)
barplot(table(df_cimbtr_sel$pseudocenterid),
        ylab = "Frequency",
        xlab = "Transplant Center ID",
        ylim = c(0, 100),
        main = "Transplant center size")
```



```r
# Table cluster
table_cluster <-
  df_cimbtr_sel %>%
  dplyr::group_by(pseudocenterid) %>%
  dplyr::summarise(n = n()) %>%
  dplyr::arrange(n)

# Summary cluster size
summary_cluster <-
  table_cluster %>%
  dplyr::summarise(min = min(n),
                   q25 = quantile(n, probs = .25),
                   mean = mean(n),
                   sd = sd(n),
                   median = median(n),
                   q75 = quantile(n, probs = .75),
                   max = max(n)) %>%
  round(., 1)

kable(summary_cluster) %>%
  kableExtra::kable_styling("striped") %>%
    kableExtra::add_header_above(
     c("Cluster size statistics" = 7))
```

Cluster size statistics						
min	q25	mean	sd	median	q75	max
1	2	8.6	11.2	5	9.8	65

```r
# Survfit: overall and by cluster
# survfit
sf_overall <- survival::survfit(Surv(intxsurv,
                                     dead) ~ 1,
                                data = df_cimbtr_sel)

q_overall <- quantile(sf_overall,
                      probs = c(.25, .50, .75))$quantile

sf_cluster <- survival::survfit(Surv(intxsurv,
                                     dead) ~ pseudocenterid,
                                data = df_cimbtr_sel)

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(sf_cluster,
     col = "gray",
     ylim = c(0, 1),
     xlim = c(0, 100),
     bty = "n",
     xlab = "Months from transplant",
     ylab = "Probability",
     main = "Overall and by center Kaplan-Meier curves")
lines(sf_overall,
      lwd = 2,
      col = "red",
      conf.int = FALSE)
```



```
# Remove centers below 10 patients
table_cimbtr_centers <-
  df_cimbtr_sel %>%
  dplyr::count(pseudocenterid) %>%
  dplyr::filter(n >= 10)

df_cimbtr_sel <-
  df_cimbtr_sel %>%
  dplyr::filter(pseudocenterid %in%
                table_cimbtr_centers$pseudocenterid)
```

2. Analysis We used the internationally recognized CIMBTR-MDS prognostic score as the solely predictor. The score combined different patients and donors' clinical, haematological and genetics characteristics without considering data clustered in different transplantation centres.
Details are provided by Nazha A, et al. (2020) here and by Shaffer BC et al. (2016) here. We considered patients with non-missing CIMBTR-MDS prognostic score.

We provide the accuracy of predictions from the stratified and the shared gamma frailty Cox proportional hazards models using the transplantation centres as cluster.

The accuracy of predictions are evaluated in terms of:

- Discrimination using time-varying AUC
- Calibration using O/E ratio, integrated calibration index (ICI), E50 and E90
- Overall performance using time-varying Index of Prediction Accuracy (IPA)

We use the Monte Carlo cross-validation procedure to assess prediction performances of the stratified and frailty Cox model.

- The sample was split into two parts to define derivation (70%) and validation samples (30%)
- The 70/30% random splits are repeated 100 times
- The repeated random splits are stratified by transplant centre (i.e., the cluster) to obtain the same centre in both the derivation and validation samples

We excluded centers with less than 10 patients since it was not possible to obtain sufficient patients in both derivation and validation sample.

The mean of the performances metrics across the 100 random validation samples with the corresponding standard deviation are reported for each scenario and at each prediction time horizon to assess the prediction performances of the stratified and the frailty Cox models.

The fixed time horizons are at 12, 24 and 36 months since transplant since most of the mortality events occurred in the first three years.

A total of 631 patients from 28 transplantation centers were included.

```r
# Rewrite B
B <- 100
# time_hors <- c(6, 12, 24)
time_hors <- c(12, 24, 36)

# Models + predictions + performances
# tictoc::tic()
df_tbl_cimbtr <-
  df_tbl_cimbtr %>%
  dplyr::filter(check == TRUE) %>%
  dplyr::select(data,
                development,
                validation) %>%

  dplyr::mutate(
    cox_strat = purrr::map(.x = development,
                           ~ survival::coxph(Surv(intxsurv, dead) ~
                                               ncgp +
                                               strata(pseudocenterid),
                                             data = .x,
                                             x = T,
                                             y = T,
                                             ties = "breslow")),

    cox_frail = purrr::map(.x = development,
                           ~ survival::coxph(Surv(intxsurv, dead) ~
                                               ncgp +
                                               frailty(pseudocenterid),
                                             data = .x,
                                             x = T,
                                             y = T,
                                             ties = "breslow")),

    # predictions at t25/t50/t75
    # stratified cox
    pred_strat = purrr::map2(.x = cox_strat,
                             .y = validation,
                             ~ 1 - pec::predictSurvProb(.x,
                                                        newdata = .y,
                                                        times = time_hors)),
    # Extremes to make calibration suitable
    pred_strat = purrr::map(pred_strat,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # separate prediction by time horizons
    # due to possible missing values
    pred_strat_t25 = purrr::map(.x = pred_strat,
                                ~ .x[, 1]),

    pred_strat_t50 = purrr::map(.x = pred_strat,
                                ~ .x[, 2]),

    pred_strat_t75 = purrr::map(.x = pred_strat,
                                ~ .x[, 3]),

    pred_frail = purrr::map2(.x = cox_frail,
                             .y = validation,
                             ~ 1 - predict.coxph.gammafrail(model = .x,
                                                            newdata = .y,
                                                            cluster = "pseudocenterid",
                                                            times = time_hors)$conditional),

    pred_frail = purrr::map2(.x = cox_frail,
                             .y = validation,
                             ~ 1 - predict.coxph.gammafrail(model = .x,
                                                            newdata = .y,
```

```r
                                                       cluster = "pseudocenterid",
                                                       times = time_hors)$conditional),
    pred_frail_t25 = purrr::map(.x = pred_frail,
                                ~ .x[, 1]),

    pred_frail_t50 = purrr::map(.x = pred_frail,
                                ~ .x[, 2]),

    pred_frail_t75 = purrr::map(.x = pred_frail,
                                ~ .x[, 3]),

    # Score at t25 - frail
    score_frail_t25 = purrr::pmap(
      list(predictions = pred_frail_t25,
           df = validation,
           frm = list(as.formula(Surv(intxsurv, dead) ~ 1)),
           t_hors = list(time_hors[1])),
      Score_tryCatch),

    # Score at t50 - frail
    score_frail_t50 = purrr::pmap(
      list(predictions = pred_frail_t50,
           df = validation,
           frm = list(as.formula(Surv(intxsurv, dead) ~ 1)),
           t_hors = list(time_hors[2])),
      Score_tryCatch),

    # Score at t75 - frail
    score_frail_t75 = purrr::pmap(
      list(predictions = pred_frail_t75,
           df = validation,
           frm = list(as.formula(Surv(intxsurv, dead) ~ 1)),
           t_hors = list(time_hors[3])),
      Score_tryCatch),

    # Extremes to make calibration suitable
    pred_frail = purrr::map(pred_frail,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # Score at t25 - strata
    score_strat_t25 = purrr::pmap(
      list(predictions = pred_strat_t25,
           df = validation,
           frm = list(as.formula(Surv(intxsurv, dead) ~ 1)),
           t_hors = list(time_hors[1])),
      Score_tryCatch),


    # Score at t50 - strata
    score_strat_t50 = purrr::pmap(
      list(predictions = pred_strat_t50,
           df = validation,
           frm = list(as.formula(Surv(intxsurv, dead) ~ 1)),
           t_hors = list(time_hors[2])),
      Score_tryCatch),

    # Score at t75 - strata
    score_strat_t75 = purrr::pmap(
      list(predictions = pred_strat_t75,
           df = validation,
           frm = list(as.formula(Surv(intxsurv, dead) ~ 1)),
           t_hors = list(time_hors[3])),
      Score_tryCatch),


    # cloglog
    # cloglog - strat
    cloglog_pred_strat = purrr::map(pred_strat,
                                    function(x) {
                                      apply(x, 2, function(y) {
                                        log(-log(1 - y))
                                      })
                                    }),

    # cloglog - strat t25/t50/t75
    cloglog_pred_strat_t25 = purrr::map(.x = cloglog_pred_strat,
                                        ~ .x[, 1]),

    cloglog_pred_strat_t50 = purrr::map(.x = cloglog_pred_strat,
                                        ~ .x[, 2]),

    cloglog_pred_strat_t75 = purrr::map(.x = cloglog_pred_strat,
                                        ~ .x[, 3]),

    # cloglog
    # cloglog - frail
    cloglog_pred_frail = purrr::map(pred_frail,
                                    function(x) {
                                      apply(x, 2, function(y) {
                                        log(-log(1 - y))
                                      })
                                    }),
    # cloglog - frail t25/t50/t75
    cloglog_pred_frail_t25 = purrr::map(.x = cloglog_pred_frail,
                                        ~ .x[, 1]),

    cloglog_pred_frail_t50 = purrr::map(.x = cloglog_pred_frail,
                                        ~ .x[, 2]),

    cloglog_pred_frail_t75 = purrr::map(.x = cloglog_pred_frail,
                                        ~ .x[, 3]),
```

```r
  # average expected risk
  pred_avg_strat = purrr::map(pred_strat,
                              function(x){
                                apply(x, 2, mean)
                              }),

  pred_avg_frail = purrr::map(pred_frail,
                              function(x) {
                                apply(x, 2, mean)
                              }),

  # estimated actual risk (estimated obs proportion, observed)
  sf_obs = purrr::map(validation,
                      function(x) {
                        sf_sum <-
                          summary(survival::survfit(Surv(intxsurv, dead) ~ 1,
                                                    data = x),
                                  extend = TRUE,
                                  times = time_hors)

                        sf_sum$surv[sf_sum$n.risk == 0] <- NA
                        sf_obs_event <- 1 - sf_sum$surv}),

  # calibration

  # OE ratio - strat
  OE_strat = purrr::map2(.x = sf_obs,
                         .y = pred_avg_strat,
                         ~ .x / .y),

  # OE ratio - frail
  OE_frail = purrr::map2(.x = sf_obs,
                         .y = pred_avg_frail,
                         ~ .x / .y),

  # ICI/E50/E90 - frail t25
  cal_meas_frail_t25  = purrr::pmap(
    list(time = purrr::map(validation, ~ .x$intxsurv),
         status = purrr::map(validation, ~ .x$dead),
         covariates = cloglog_pred_frail_t25,
         t_hors = list(time_hors[1]),
         predictions = purrr::map(pred_frail, ~ .x[, 1])),
    calhaz_map),

  # ICI/E50/E90 - frail t50
  cal_meas_frail_t50  = purrr::pmap(
    list(time = purrr::map(validation, ~ .x$intxsurv),
         status = purrr::map(validation, ~ .x$dead),
         covariates = cloglog_pred_frail_t50,
         t_hors = list(time_hors[2]),
         predictions = purrr::map(pred_frail, ~ .x[, 2])),
    calhaz_map),
    # ICI/E50/E90 - frail t75
    cal_meas_frail_t75  = purrr::pmap(
      list(time = purrr::map(validation, ~ .x$intxsurv),
           status = purrr::map(validation, ~ .x$dead),
           covariates = cloglog_pred_frail_t75,
           t_hors = list(time_hors[3]),
           predictions = purrr::map(pred_frail, ~ .x[, 3])),
      calhaz_map),

    # stratify
    # ICI/E50/E90 - strat t25
    cal_meas_strat_t25  = purrr::pmap(
      list(time = purrr::map(validation, ~ .x$intxsurv),
           status = purrr::map(validation, ~ .x$dead),
           covariates = cloglog_pred_strat_t25,
           t_hors = list(time_hors[1]),
           predictions = purrr::map(pred_strat, ~ .x[, 1])),
      calhaz_map),

    # ICI/E50/E90 - strat t50
    cal_meas_strat_t50  = purrr::pmap(
      list(time = purrr::map(validation, ~ .x$intxsurv),
           status = purrr::map(validation, ~ .x$dead),
           covariates = cloglog_pred_strat_t50,
           t_hors = list(time_hors[2]),
           predictions = purrr::map(pred_strat, ~ .x[, 2])),
      calhaz_map),

    # ICI/E50/E90 - strat t75
    cal_meas_strat_t75  = purrr::pmap(
      list(time = purrr::map(validation, ~ .x$intxsurv),
           status = purrr::map(validation, ~ .x$dead),
           covariates = cloglog_pred_strat_t75,
           t_hors = list(time_hors[3]),
           predictions = purrr::map(pred_strat, ~ .x[, 3])),
      calhaz_map))

# tictoc::toc()
```

CIMBTR dataset						
			AUC		IPA	
model	times	pct_comp	AUC_mean	AUC_sd	IPA_mean	IPA_sd
frail	12	100	0.619	0.037	0.045	0.032
strat	12	99	0.623	0.031	0.009	0.049
frail	24	100	0.608	0.039	0.031	0.039
strat	24	95	0.641	0.031	0.024	0.052
frail	36	100	0.631	0.040	0.053	0.042
strat	36	37	0.664	0.035	0.054	0.051

CIMBTR dataset				
			O-E ratio	
model	times	pct_comp	OE_ratio_mean	OE_ratio_sd
frail	12	100	0.994	0.094
strat	12	99	1.044	0.103
frail	24	100	0.995	0.088
strat	24	95	1.041	0.087
frail	36	100	1.001	0.080
strat	36	37	1.043	0.072

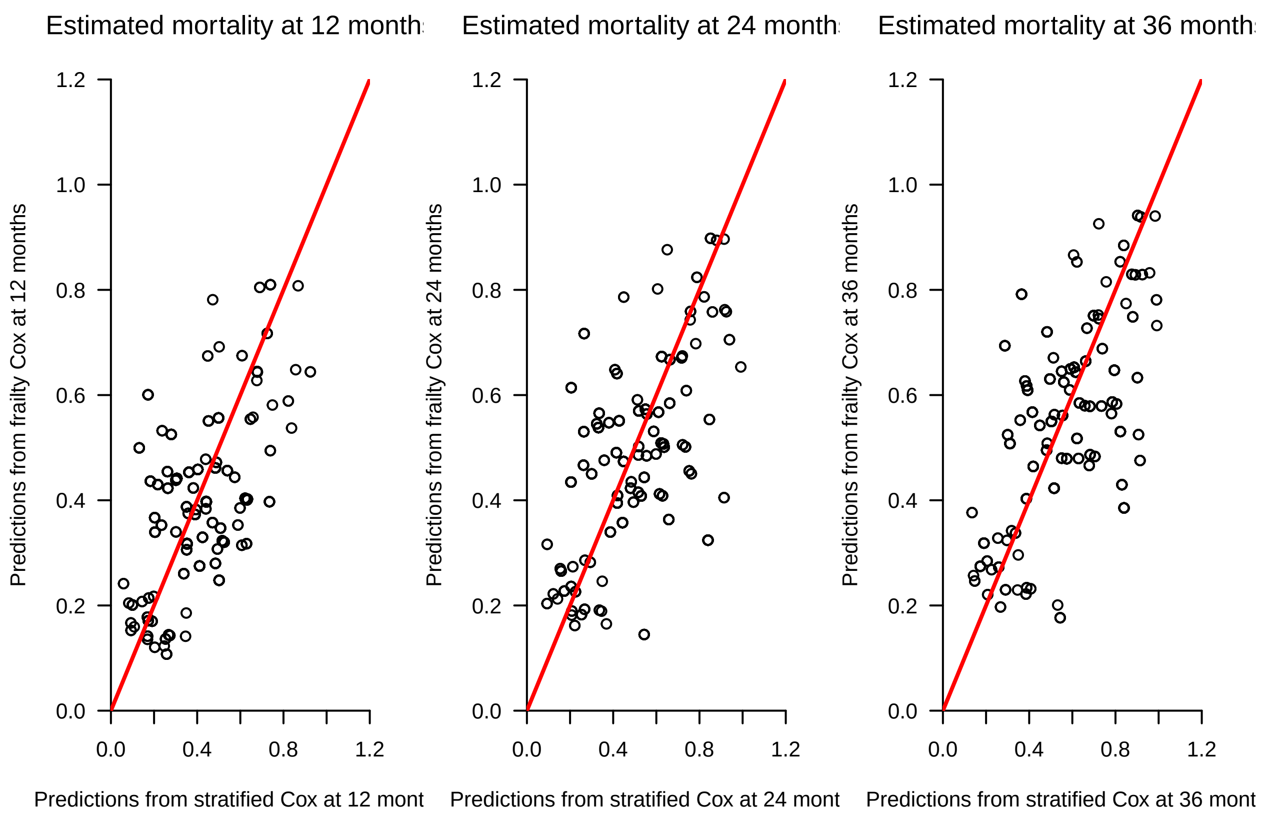
CIMBTR dataset								
			ICI		E50		E90	
model	times	pct_comp	ICI_mean	ICI_sd	E50_mean	E50_sd	E90_mean	E90_sd
frail	12	100	0.059	0.058	0.050	0.059	0.112	0.059
strat	12	99	0.092	0.050	0.077	0.046	0.183	0.046
frail	24	100	0.065	0.056	0.058	0.057	0.124	0.057
strat	24	95	0.087	0.038	0.077	0.033	0.167	0.033
frail	36	100	0.065	0.035	0.057	0.034	0.127	0.034
strat	36	37	0.087	0.076	0.079	0.081	0.158	0.081

3. Additional investigations We additionally estimate predictions from the stratified and the shared gamma frailty Cox model using all data without splitting among clusters in which predictions were possible in both models.

We provide the corresponding plots of predictions at the different specified time horizons from the stratified versus the shared gamma frailty model

```
# Stratified Cox
cox_strat_cimbtr <-
  survival::coxph(Surv(intxsurv, dead) ~
                    ncgp +
                    strata(pseudocenterid),
      data = df_cimbtr_sel,
      x = T,
      y = T,
```

```r
      ties = "breslow")

# Frailty Cox
cox_frail_cimbtr <-
  survival::coxph(Surv(intxsurv, dead) ~
                    ncgp +
                    frailty(pseudocenterid),
      data = df_cimbtr_sel,
      x = T,
      y = T,
      ties = "breslow")



## Predictions horizons
# at 12, 24, and 36 months
time_hors <- c(12, 24, 36)


# Predictions
# Stratified small
pred_strat_cimbtr <- 1 - pec::predictSurvProb(cox_strat_cimbtr,
                                              newdata = df_cimbtr_sel,
                                              times = time_hors)

# Frailty small
pred_frail_cimbtr <- 1 - predict.coxph.gammafrail(model = cox_frail_cimbtr,
                                                  newdata = df_cimbtr_sel,
                                                  cluster = "pseudocenterid",
                                                  times = time_hors)$conditional


# Plot predictions
# from stratify and frailty model

# Small scenario at T25, T50 and T75
oldpar <-
  par(
    mfrow = c(1, 3),
    xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_cimbtr[, 1],
     pred_frail_cimbtr[, 1],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at 12 months"),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at 12 months"),
     main = expression("Estimated mortality at 12 months"),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_cimbtr[, 2],
     pred_frail_cimbtr[, 2],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at 24 months"),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at 24 months"),
     main = expression("Estimated mortality at 24 months"),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_cimbtr[, 3],
     pred_frail_cimbtr[, 3],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at 36 months"),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at 36 months"),
     main = expression("Estimated mortality at 36 months"),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")
```



NOTE: comparisons only among clusters where predictions were possible for both models

Second case study: bladder data

1. Descriptive statistics We provide some descriptive statistics of the case study

```
df_bladder <-
  bladder %>%
  dplyr::rename_all(tolower) %>%
  dplyr::mutate(

    # combine status
    cstatus = dplyr::case_when(
      status %in% c(1, 2) ~ 1,
      status == 0 ~ 0),

    age = factor(age,
                 levels = c(0, 1),
                 labels = c("<= 65 yr", "> 65 yr")),

    sex = factor(sex,
                 levels = c(0, 1),
                 labels = c("male", "female")),

    priorrec = factor(priorrec,
                      levels = c(0, 1, 2),
                      labels = c("primary",
                                 "<= 1 yr",
                                  "> 1 yr")),

    chemo = factor(chemo,
                   levels = c(0, 1),
                   labels = c("no", "yes")),

    notum = factor(notum,
                   levels = c(0, 1, 2),
                   labels = c("single tumor",
                              "2-7 tumors",
                              ">= 8 tumors")),

    tum3cm = factor(tum3cm,
                    levels = c(0, 1),
                    labels = c("< 3 cm", ">= 3 cm")),

    tlocc = factor(tlocc,
                   levels = c(0, 1),
                   labels = c("Ta", "T1")),

    cis = factor(cis,
                 levels = c(0, 1),
                 labels = c("no", "yes")),

    glocal = factor(glocal,
                    levels = c(0, 1, 2),
                    labels = c("G1", "G2", "G3")))

# Table 1
bladder_tab_01 <-
  df_bladder %>%
  dplyr::mutate(

        status = factor(status,
                        levels = c(0, 1, 2),
                        labels = c("Recurrence",
                                   "Death before recurrence",
                                   "Censored"))) %>%
  dplyr::select(age,
                sex,
                chemo,
                priorrec,
                notum,
                tum3cm,
                tlocc,
                cis,
                glocal,
                cstatus,
                status)
```

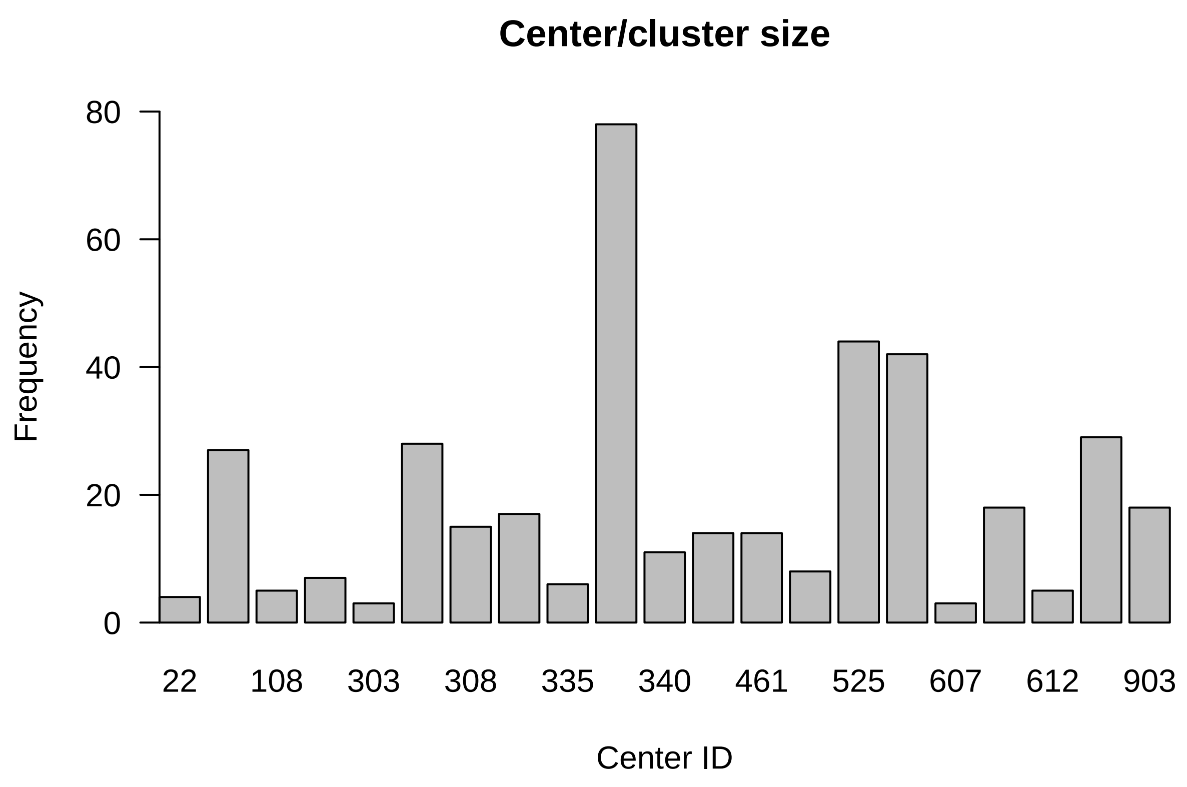
```r
label(bladder_tab_01$age) <- "Age, years"
label(bladder_tab_01$sex) <- "Sex"
label(bladder_tab_01$chemo) <- "Chemotherapy"
label(bladder_tab_01$priorrec) <- "Prior recurrent rate"
label(bladder_tab_01$tum3cm) <- "Tumor size"
label(bladder_tab_01$notum) <- "Number of tumors"
label(bladder_tab_01$tlocc) <- "T category"
label(bladder_tab_01$cis) <- "Carcinoma in situ"
label(bladder_tab_01$glocal) <- "Tumor grade"

gtsummary_bladder_table1 <-
  gtsummary::tbl_summary(
    data = bladder_tab_01 ,
    label = list(
      age ~ "Age, years",
      sex ~  "Sex",
      chemo ~ "Chemotherapy",
      priorrec ~ "Prior recurrent rate",
      tum3cm ~ "Tumor size",
      notum ~ "Number of tumors",
      tlocc ~ "T category",
      cis ~ "Carcinoma in situ",
      glocal ~ "Tumor grade"),
    type = all_continuous() ~ "continuous2",
    statistic = all_continuous() ~ c(
      "{mean} ({sd})",
      "{median} ({min}, {max})"))


gtsummary_bladder_table1
```

Characteristic	**N = 396**
Age, years	
<= 65 yr	183 (46%)
> 65 yr	213 (54%)
Sex	
male	330 (83%)
female	66 (17%)
Chemotherapy	330 (83%)
Prior recurrent rate	
primary	272 (69%)
<= 1 yr	42 (11%)
> 1 yr	82 (21%)
Number of tumors	
single tumor	171 (43%)
2-7 tumors	190 (48%)
>= 8 tumors	35 (8.8%)
Tumor size	
< 3 cm	290 (73%)
>= 3 cm	106 (27%)
T category	
Ta	218 (55%)

Characteristic	**N = 396**
T1	178 (45%)
Carcinoma in situ	24 (6.1%)
Tumor grade	
G1	191 (48%)
G2	167 (42%)
G3	38 (9.6%)
cstatus	281 (71%)
status	
Recurrence	115 (29%)
Death before recurrence	200 (51%)
Censored	81 (20%)

```
  # gtsummary_bladder_table1 %>%
  # gtsummary::as_gt(.)

# knitr::knit_print(gtsummary_bladder_table1)
```

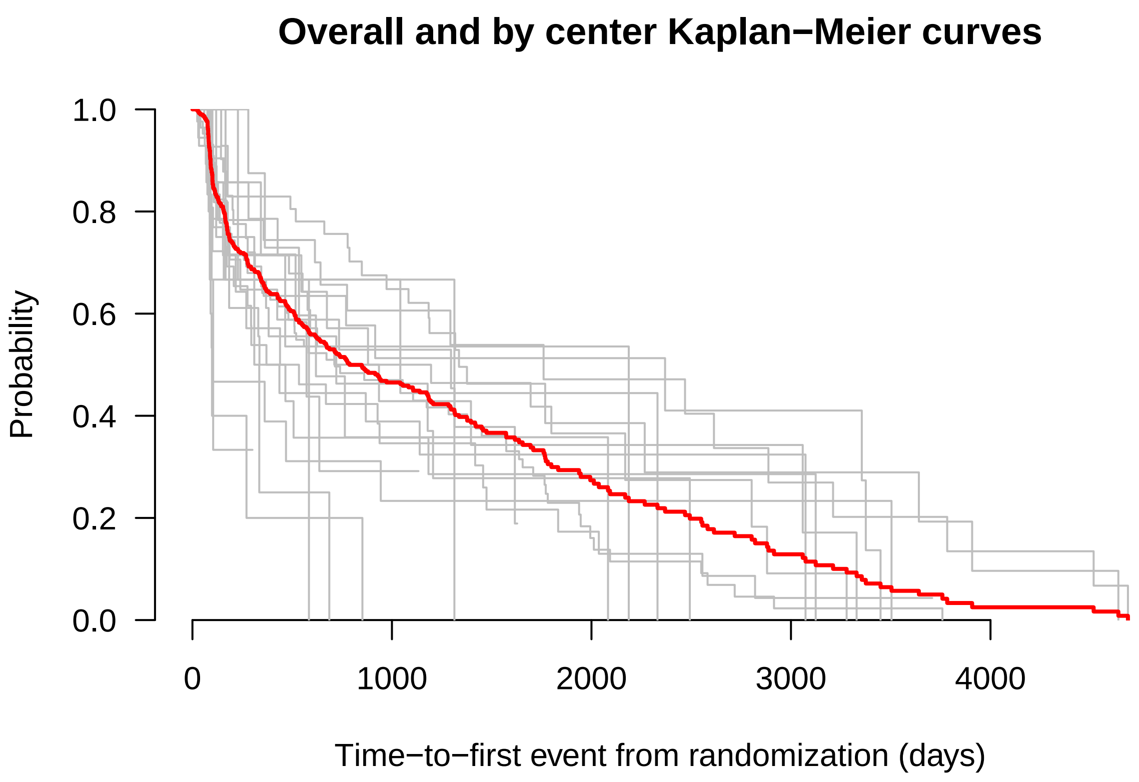
```r
# Cluster size distribution
table_cluster <-
  df_bladder %>%
  dplyr::group_by(center) %>%
  dplyr::summarise(n = n()) %>%
  dplyr::arrange(n)

par(xaxs = "i",
    yaxs = "i",
    las = 1)
barplot(table(df_bladder$center),
        ylab = "Frequency",
        xlab = "Center ID",
        ylim = c(0, 80),
        main = "Center/cluster size")
```



```r
# kable(table_cluster) %>%
#   kableExtra::kable_styling("striped",
#                             position = "center") %>%
#     kableExtra::add_header_above(
#      c("Cluster size distribution" = 2))

# Summary cluster size
summary_cluster <-
  table_cluster %>%
  dplyr::summarise(min = min(n),
                   q25 = quantile(n, probs = .25),
                   mean = mean(n),
                   sd = sd(n),
                   median = median(n),
                   q75 = quantile(n, probs = .75),
                   max = max(n)) %>%
  round(., 1)

kable(summary_cluster) %>%
  kableExtra::kable_styling("striped") %>%
    kableExtra::add_header_above(
     c("Cluster size statistics" = 7))
```

Cluster size statistics						
min	q25	mean	sd	median	q75	max
3	6	18.9	18.1	14	27	78

```r
# Event distribution per cluster
# kable(
#   addmargins(
#     table(df_bladder$center,
#           df_bladder$cstatus,
#           useNA = "ifany",
#           dnn = c("Cluster",
#                   "Status")))) %>%
#     kableExtra::kable_styling("striped") %>%
#   kableExtra::add_header_above(
#      c("Cluster" = 1,
#        "Status" =  2,
#        "Sum" = 1))

# survfit
sf_overall <- survival::survfit(Surv(surtime,
                                     cstatus) ~ 1,
                                data = df_bladder)

sf_cluster <- survival::survfit(Surv(surtime,
                                     cstatus) ~ center,
                                data = df_bladder)


par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(sf_cluster,
     col = "gray",
     ylim = c(0, 1),
     bty = "n",
     xlab = "Time-to-first event from randomization (days)",
     ylab = "Probability",
     main = "Overall and by center Kaplan-Meier curves")
lines(sf_overall,
      lwd = 2,
      col = "red",
      conf.int = FALSE)
```



```
# Time quantiles
q_overall <- quantile(sf_overall,
                      probs = c(.25, .50, .75))$quantile
q_cluster <-
  quantile(sf_cluster,
           probs = c(.25, .50, .75))$quantile

q_surv <- rbind.data.frame("Overall" = q_overall,
                           q_cluster)

# kable(q_surv[1, ]) %>%
#   kableExtra::kable_styling("striped",
#                             position = "center") %>%
#   kableExtra::add_header_above(
#     c(" " = 1,
#       "Time quantile" = 3))
```

2. Analysis We also apply the same analysis framework described previously using data on bladder cancer patients participating in a European Organization for Research and Treatment of Cancer (EORTC) trial ref.

We used the free available data bladder from the R package `survival` consisting of 396 patients from 21 cancer centres. As described in the previous paragraph, we excluded centres with less than 10 patients to assess the prediction performances of the stratified and the shared frailty gamma Cox models to estimate disease-free survival at 365.25, 730.5 and 1095.75 days (i.e., 12, 24 and 36 months) using the Monte Carlo cross-validation procedure. The following predictors were included: age, sex, chemotherapy, number of tumours, tumour size, T stage and grade. A total of 355 patients from 14 centres were included. Two hundred and fifty-three events were observed (175 recurrences and 78 deaths before recurrence). The number of patients per centre varied between 11 and 78 with mean 27.3 and median 18.

```r
# time horizons
# time_hors <- c(185, 787, 2093)
time_hors <- c(365.25, 730.5, 1095.75)

# reference values for continuous
# predictors

# Centres with >= 10 patients
table_bladder_sel <-
  df_bladder %>%
  dplyr::count(center) %>%
  dplyr::filter(n >= 10)

df_bladder_sel <-
  df_bladder %>%
  dplyr::filter(center %in% table_bladder_sel$center)

# Repeat B times
B <- 101
df_bladder_sel_ext <- do.call(rbind,
                              lapply(seq(1:B),
                                     function(k) {
                                       cbind(id = k, df_bladder_sel)}))
# Nested tibble
set.seed(20250924)
df_tbl_bladder <-
  df_bladder_sel_ext %>%
  dplyr::group_nest(id) %>%
  dplyr::mutate(
    # Splitting
    split = purrr::map(.x = data,
                       ~ rsample::initial_split(.x,
                                                strata = center,
                                                prop = 0.7,
                                                pool = 0.001)),
    # Development
    development = purrr::map(.x = split,
                             ~ rsample::training(.x)),

    # Validation
    validation = purrr::map(.x = split,
                            ~ rsample::testing(.x)),
    # Check
    unique_cluster_dev = purrr::map(.x = development,
                                    ~ sort(unique(.x$center))),

    unique_cluster_val = purrr::map(.x = validation,
                                    ~ sort(unique(.x$center))),

    check = purrr::map2(.x = unique_cluster_dev,
                        .y = unique_cluster_val,
                        ~ all.equal(.x, .y)))


# table(do.call(rbind, df_tbl_bladder$check))
rm(df_bladder_sel_ext)
# gc()
# check
# table(do.call(rbind, df_tbl_bladder$check))
# all TRUE

# Rewrite B
B <- 100

# Models + predictions + performances
# tictoc::tic()
df_tbl_bladder <-
  df_tbl_bladder %>%
  dplyr::filter(check == TRUE) %>%
  dplyr::select(data,
                development,
                validation) %>%

  dplyr::mutate(
    cox_strat = purrr::map(.x = development,
                           ~ survival::coxph(Surv(surtime, cstatus) ~
                                                 age +
                                                 sex +
                                                 chemo +
                                                 tum3cm +
                                                 notum +
                                                 tlocc +
                                                 glocal +
                                                 strata(center),
                                               data = .x,
                                               x = T,
                                               y = T,
                                               ties = "breslow")),

    cox_frail = purrr::map(.x = development,
                           ~ survival::coxph(Surv(surtime, cstatus) ~
                                                 age +
                                                 sex +
                                                 chemo +
                                                 tum3cm +
                                                 notum +
                                                 tlocc +
                                                 glocal +
                                                 frailty(center),
                                               data = .x,
                                               x = T,
                                               y = T,
                                               ties = "breslow")),

    # predictions at t25/t50/t75
    # stratified cox
    pred_strat = purrr::map2(.x = cox_strat,
                             .y = validation,
                             ~ 1 - pec::predictSurvProb(.x,
                                                        newdata = .y,
                                                        times = time_hors)),
    # Extremes to make calibration suitable
    pred_strat = purrr::map(pred_strat,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # separate prediction by time horizons
    # due to possible missing values
    pred_strat_t25 = purrr::map(.x = pred_strat,
                                ~ .x[, 1]),

    pred_strat_t50 = purrr::map(.x = pred_strat,
                                ~ .x[, 2]),

    pred_strat_t75 = purrr::map(.x = pred_strat,
                                ~ .x[, 3]),

    pred_frail = purrr::map2(.x = cox_frail,
                             .y = validation,
                             ~ 1 - predict.coxph.gammafrail.dev(model = .x,
                                                                newdata = .y,
                                                                cluster = "center",
                                                                times = time_hors)$conditional),
    # Extremes to make calibration suitable
    pred_frail = purrr::map(pred_frail,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # Score at t25/t50/t75 - frail
    score_frail = purrr::map2(.x = pred_frail,
                              .y = validation,
                              ~ riskRegression::Score(
                                list("frail" = .x),
                                data = .y,
                                formula = Surv(surtime, cstatus) ~ 1,
                                metrics = "auc",
                                summary = "ipa",
                                times = time_hors)),

  # save results score - frail
  score_frail_res = purrr::map(score_frail,
                                 function(x) {

                                   res_AUC <-
                                     x$AUC$score %>%
                                     dplyr::select(model, times, AUC)

                                   res_IPA <-
                                     x$Brier$score %>%
                                     dplyr::select(model, times, IPA) %>%
                                     dplyr::filter(model %nin% c("Null model"))

                                   res <- cbind(res_AUC,
                                                "IPA" = res_IPA$IPA)

                                   return(res)
                                 }),

  # Score at t25 - strata
 score_strat_t25 = purrr::pmap(
    list(predictions = pred_strat_t25,
         df = validation,
         frm = list(as.formula(Surv(surtime, cstatus) ~ 1)),
         t_hors = list(time_hors[1])),
    Score_tryCatch),


  # Score at t50 - strata
  score_strat_t50 = purrr::pmap(
    list(predictions = pred_strat_t50,
         df = validation,
         frm = list(as.formula(Surv(surtime, cstatus) ~ 1)),
         t_hors = list(time_hors[2])),
    Score_tryCatch),

  # Score at t75 - strata
  score_strat_t75 = purrr::pmap(
    list(predictions = pred_strat_t75,
         df = validation,
         frm = list(as.formula(Surv(surtime, cstatus) ~ 1)),
         t_hors = list(time_hors[3])),
    Score_tryCatch),


  # cloglog
  # cloglog - strat
  cloglog_pred_strat = purrr::map(pred_strat,
                                  function(x) {
                                    apply(x, 2, function(y) {
                                      log(-log(1 - y))
                                    })
                                  }),

 # cloglog - strat t25/t50/t75
 cloglog_pred_strat_t25 = purrr::map(.x = cloglog_pred_strat,
                                     ~ .x[, 1]),

 cloglog_pred_strat_t50 = purrr::map(.x = cloglog_pred_strat,
                                     ~ .x[, 2]),

 cloglog_pred_strat_t75 = purrr::map(.x = cloglog_pred_strat,
                                     ~ .x[, 3]),

  # cloglog
  # cloglog - frail
  cloglog_pred_frail = purrr::map(pred_frail,
                                  function(x) {
                                    apply(x, 2, function(y) {
                                      log(-log(1 - y))
                                    })
                                  }),
 # cloglog - frail t25/t50/t75
  cloglog_pred_frail_t25 = purrr::map(.x = cloglog_pred_frail,
                                      ~ .x[, 1]),

  cloglog_pred_frail_t50 = purrr::map(.x = cloglog_pred_frail,
                                      ~ .x[, 2]),

  cloglog_pred_frail_t75 = purrr::map(.x = cloglog_pred_frail,
                                      ~ .x[, 3]),


  # average expected risk
  pred_avg_strat = purrr::map(pred_strat,
                              function(x){
                                apply(x, 2, mean)
                              }),

  pred_avg_frail = purrr::map(pred_frail,
                              function(x) {
                               apply(x, 2, mean)
                             }),

  # estimated actual risk (estimated obs proportion, observed)
  sf_obs = purrr::map(validation,
                      function(x) {
                        sf_sum <-
                          summary(survival::survfit(Surv(surtime, cstatus) ~ 1,
                                                    data = x),
                                  extend = TRUE,
                                  times = time_hors)

                        sf_sum$surv[sf_sum$n.risk == 0] <- NA
                        sf_obs_event <- 1 - sf_sum$surv}),

 # calibration

 # OE ratio - strat
 OE_strat = purrr::map2(.x = sf_obs,
                        .y = pred_avg_strat,
                        ~ .x / .y),

  # OE ratio - frail
 OE_frail = purrr::map2(.x = sf_obs,
                        .y = pred_avg_frail,
                        ~ .x / .y),

 # ICI/E50/E90 - frail t25
 cal_meas_frail_t25  = purrr::pmap(
   list(time = purrr::map(validation, ~ .x$surtime),
        status = purrr::map(validation, ~ .x$cstatus),
        covariates = cloglog_pred_frail_t25,
        t_hors = list(time_hors[1]),
        predictions = purrr::map(pred_frail, ~ .x[, 1])),
        calhaz_map),

 # ICI/E50/E90 - frail t50
 cal_meas_frail_t50  = purrr::pmap(
   list(time = purrr::map(validation, ~ .x$surtime),
        status = purrr::map(validation, ~ .x$cstatus),
        covariates = cloglog_pred_frail_t50,
        t_hors = list(time_hors[2]),
        predictions = purrr::map(pred_frail, ~ .x[, 2])),
        calhaz_map),

 # ICI/E50/E90 - frail t75
 cal_meas_frail_t75  = purrr::pmap(
   list(time = purrr::map(validation, ~ .x$surtime),
        status = purrr::map(validation, ~ .x$cstatus),
        covariates = cloglog_pred_frail_t75,
        t_hors = list(time_hors[3]),
        predictions = purrr::map(pred_frail, ~ .x[, 3])),
        calhaz_map),

 # stratify
 # ICI/E50/E90 - strat t25
 cal_meas_strat_t25  = purrr::pmap(
   list(time = purrr::map(validation, ~ .x$surtime),
        status = purrr::map(validation, ~ .x$cstatus),
        covariates = cloglog_pred_strat_t25,
        t_hors = list(time_hors[1]),
        predictions = purrr::map(pred_strat, ~ .x[, 1])),
        calhaz_map),
  # ICI/E50/E90 - strat t50
  cal_meas_strat_t50  = purrr::pmap(
    list(time = purrr::map(validation, ~ .x$surtime),
         status = purrr::map(validation, ~ .x$cstatus),
         covariates = cloglog_pred_strat_t50,
         t_hors = list(time_hors[2]),
         predictions = purrr::map(pred_strat, ~ .x[, 2])),
         calhaz_map),

  # ICI/E50/E90 - strat t75
  cal_meas_strat_t75  = purrr::pmap(
    list(time = purrr::map(validation, ~ .x$surtime),
         status = purrr::map(validation, ~ .x$cstatus),
         covariates = cloglog_pred_strat_t75,
         t_hors = list(time_hors[3]),
         predictions = purrr::map(pred_strat, ~ .x[, 3])),
         calhaz_map))

# tictoc::toc()
```

Bladder cancer data						
			AUC		IPA	
model	times	pct_comp	AUC_mean	AUC_sd	IPA_mean	IPA_sd
frail	365.25	100	0.634	0.049	0.044	0.043
strat	365.25	99	0.623	0.050	0.019	0.062
frail	730.50	100	0.616	0.046	0.024	0.050
strat	730.50	99	0.605	0.046	-0.015	0.070
frail	1095.75	100	0.594	0.049	0.001	0.054
strat	1095.75	91	0.565	0.049	-0.058	0.075

Bladder cancer data				
			O-E ratio	
model	times	pct_comp	OE_ratio_mean	OE_ratio_sd
frail	365.25	100	0.985	0.107
strat	365.25	99	1.045	0.148
frail	730.50	100	1.011	0.122
strat	730.50	99	1.050	0.129
frail	1095.75	100	0.996	0.128
strat	1095.75	91	1.041	0.109

Bladder cancer data								
			ICI		E50		E90	
model	times	pct_comp	ICI_mean	ICI_sd	E50_mean	E50_sd	E90_mean	E90_sd
frail	365.25	100	0.101	0.066	0.090	0.068	0.191	0.104
strat	365.25	99	0.126	0.078	0.113	0.078	0.239	0.128
frail	730.50	100	0.103	0.065	0.093	0.067	0.189	0.101
strat	730.50	99	0.131	0.069	0.120	0.071	0.246	0.114
frail	1095.75	100	0.104	0.068	0.093	0.070	0.192	0.104
strat	1095.75	91	0.141	0.070	0.131	0.069	0.265	0.116

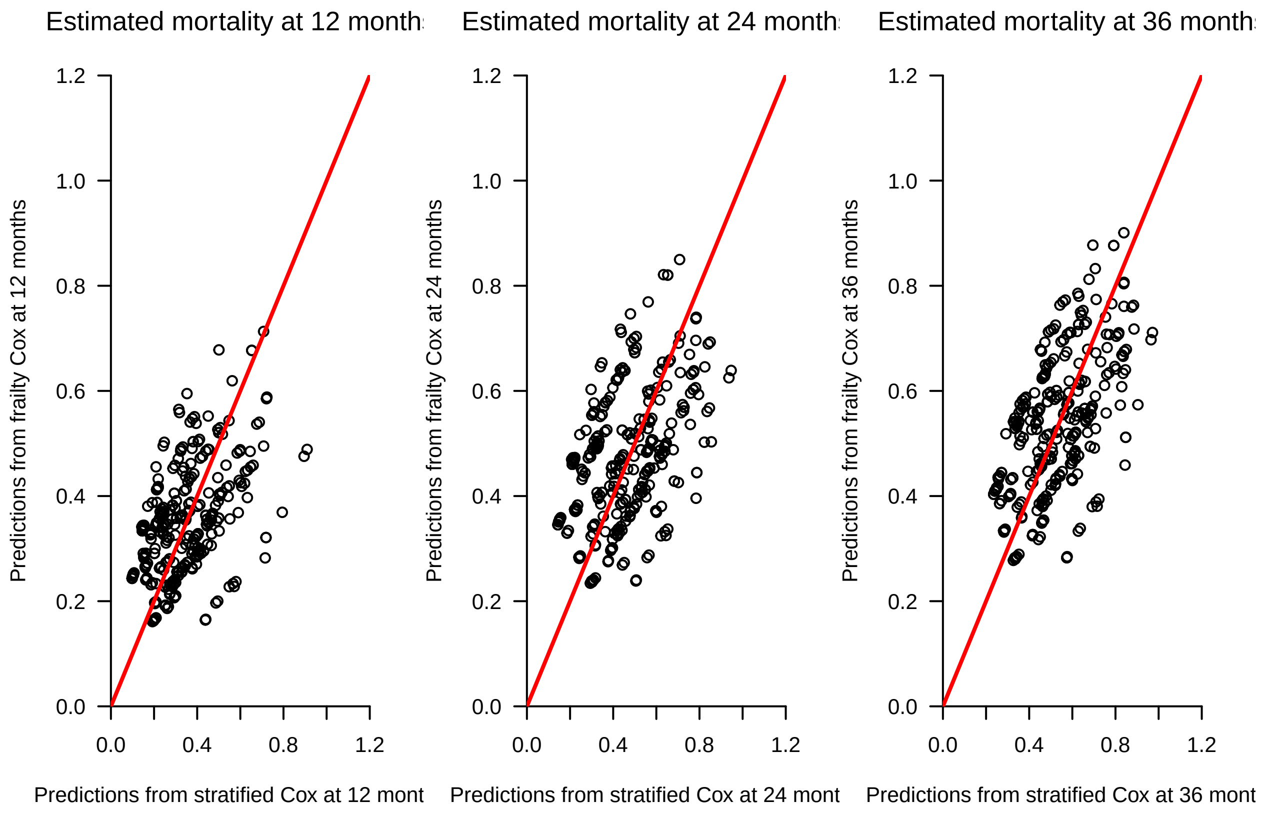
3. Additional investigations We additionally estimate predictions from the stratified and the shared gamma frailty Cox model using all data without splitting among clusters in which predictions were possible in both models.

We provide the corresponding plots of predictions at the different specified time horizons from the stratified versus the shared gamma frailty model

```r
bladder <- readRDS(here::here("Data/bladder.rds"))

df_bladder <-
  bladder %>%
  dplyr::rename_all(tolower) %>%
  dplyr::mutate(

    # combine status
    cstatus = dplyr::case_when(
      status %in% c(1, 2) ~ 1,
      status == 0 ~ 0),

    age = factor(age,
                 levels = c(0, 1),
                 labels = c("<= 65 yr", "> 65 yr")),

    sex = factor(sex,
                 levels = c(0, 1),
                 labels = c("male", "female")),

    priorrec = factor(priorrec,
                      levels = c(0, 1, 2),
                      labels = c("primary",
                                 "<= 1 yr",
                                  "> 1 yr")),

    chemo = factor(chemo,
                   levels = c(0, 1),
                   labels = c("no", "yes")),

    notum = factor(notum,
                   levels = c(0, 1, 2),
                   labels = c("single tumor",
                              "2-7 tumors",
                              ">= 8 tumors")),

    tum3cm = factor(tum3cm,
                    levels = c(0, 1),
                    labels = c("< 3 cm", ">= 3 cm")),

    tlocc = factor(tlocc,
                   levels = c(0, 1),
                   labels = c("Ta", "T1")),

    cis = factor(cis,
                 levels = c(0, 1),
                 labels = c("no", "yes")),
    glocal = factor(glocal,
                    levels = c(0, 1, 2),
                    labels = c("G1", "G2", "G3")))

# Centres with >= 10 patients
table_bladder_sel <-
  df_bladder %>%
  dplyr::count(center) %>%
  dplyr::filter(n >= 10)

df_bladder_sel <-
  df_bladder %>%
  dplyr::filter(center %in% table_bladder_sel$center)

# Stratified Cox
cox_strat_bladder <-
  survival::coxph(Surv(surtime, cstatus) ~
                    age +
                    sex +
                    chemo +
                    tum3cm +
                    notum +
                    tlocc +
                    glocal +
                    strata(center),
                  data = df_bladder_sel,
                  x = T,
                  y = T,
                  ties = "breslow")

# Frailty Cox
cox_frail_bladder <-
  survival::coxph(Surv(surtime, cstatus) ~
                    age +
                    sex +
                    chemo +
                    tum3cm +
                    notum +
                    tlocc +
                    glocal +
                    frailty(center),
                  data = df_bladder_sel,
                  x = T,
                  y = T,
                  ties = "breslow")

## Predictions horizons
# at 12, 24, and 36 months
time_hors <- c(365.25, 730.5, 1095.75)


# Predictions
# Stratified small
```

```r
pred_strat_bladder <- 1 - pec::predictSurvProb(cox_strat_bladder,
                                               newdata = df_bladder_sel,
                                               times = time_hors)

# Frailty small
pred_frail_bladder <- 1 - predict.coxph.gammafrail(model = cox_frail_bladder,
                                                   newdata = df_bladder_sel,
                                                   cluster = "center",
                                                   times = time_hors)$conditional


# Plot predictions
# from stratify and frailty model

# Small scenario at T25, T50 and T75
oldpar <-
  par(
    mfrow = c(1, 3),
    xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_bladder[, 1],
     pred_frail_bladder[, 1],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at 12 months"),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at 12 months"),
     main = expression("Estimated mortality at 12 months"),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_bladder[, 2],
     pred_frail_bladder[, 2],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at 24 months"),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at 24 months"),
     main = expression("Estimated mortality at 24 months"),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
```

```r
plot(pred_strat_bladder[, 3],
     pred_frail_bladder[, 3],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at 36 months"),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at 36 months"),
     main = expression("Estimated mortality at 36 months"),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")
```



NOTE: comparisons only among clusters where predictions were possible for both models

Extra case study: insem data

We provide an empirical bonus case study analysis comparing the performance of the stratified and the shared gamma frailty Cox proportional hazards models for predicting time to first insemination in dairy heifer cows coming from different herds.

The case study is the insem data from the R package `parfm` described by the book of Duchateau and Janssen The Frailty Model (2008) (example 1.8)

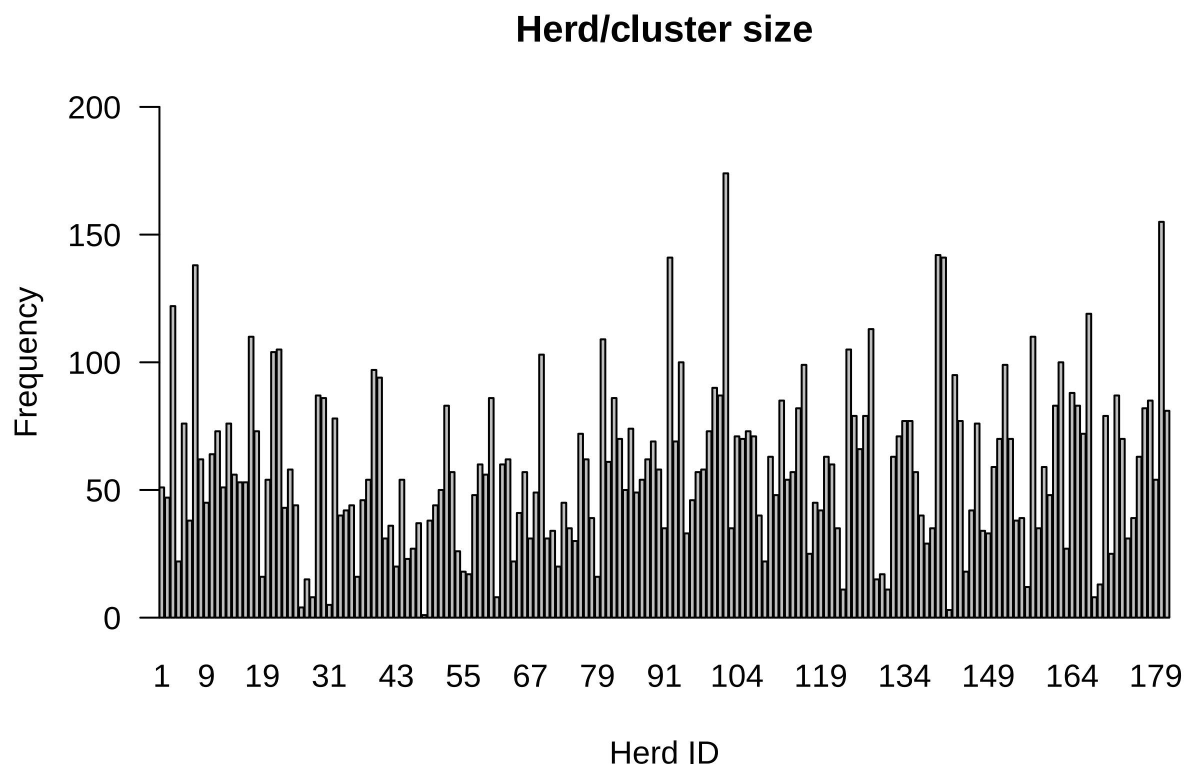
1. Descriptive statistics We provide some descriptive statistics

```
insem_tab_01 <-
  insem %>%
  dplyr::select(Ureum,
                Protein,
                Parity,
                Heifer,
                Status)

label(insem_tab_01$Ureum) <- "Milk urem concentration (%) at the start of the
                              lactation period"
label(insem_tab_01$Protein) <- "Protein concentration (%) at the start of the
                                lactation period"
label(insem_tab_01$Parity) <- "The number of calvings"
label(insem_tab_01$Heifer) <- "Primiparous cow"
label(insem_tab_01$Status) <- "Inseminated cow"

units(insem_tab_01$Ureum) <- "%"
units(insem_tab_01$Protein) <- "%"

gtsummary_table1 <-
  gtsummary::tbl_summary(
    data = insem_tab_01 ,
    label = list(Parity ~ "Number of calvings",
                 Heifer ~ "Primiparous cow",
                 Ureum ~ "Milk urem concentration (%) ",
                 Protein ~ "Protein concentration (%)",
                 Status ~ "Inseminated cows"),
    type = all_continuous() ~ "continuous2",
    statistic = all_continuous() ~ c(
      "{mean} ({sd})",
      "{median} ({min}, {max})"
    ),
  )


gtsummary_table1
```

Characteristic	**N = 10,513**
Milk urem concentration (%)	
Mean (SD)	2.58 (0.74)
Median (Range)	2.55 (0.54, 8.24)

Characteristic	**N = 10,513**
Protein concentration (%)	
Mean (SD)	3.25 (0.34)
Median (Range)	3.21 (2.25, 6.48)
Number of calvings	
Mean (SD)	2 (2)
Median (Range)	2 (1, 14)
Primiparous cow	4,200 (40%)
Inseminated cows	9,939 (95%)

```r
# knitr::knit_print(gtsummary_table1)

# Cluster size distribution
table_cluster <-
  insem %>%
  dplyr::group_by(Herd) %>%
  dplyr::summarise(n = n()) %>%
  dplyr::arrange(n)

par(xaxs = "i",
    yaxs = "i",
    las = 1)
barplot(table(insem$Herd),
        ylab = "Frequency",
        xlab = "Herd ID",
        ylim = c(0, 200),
        main = "Herd/cluster size")
```

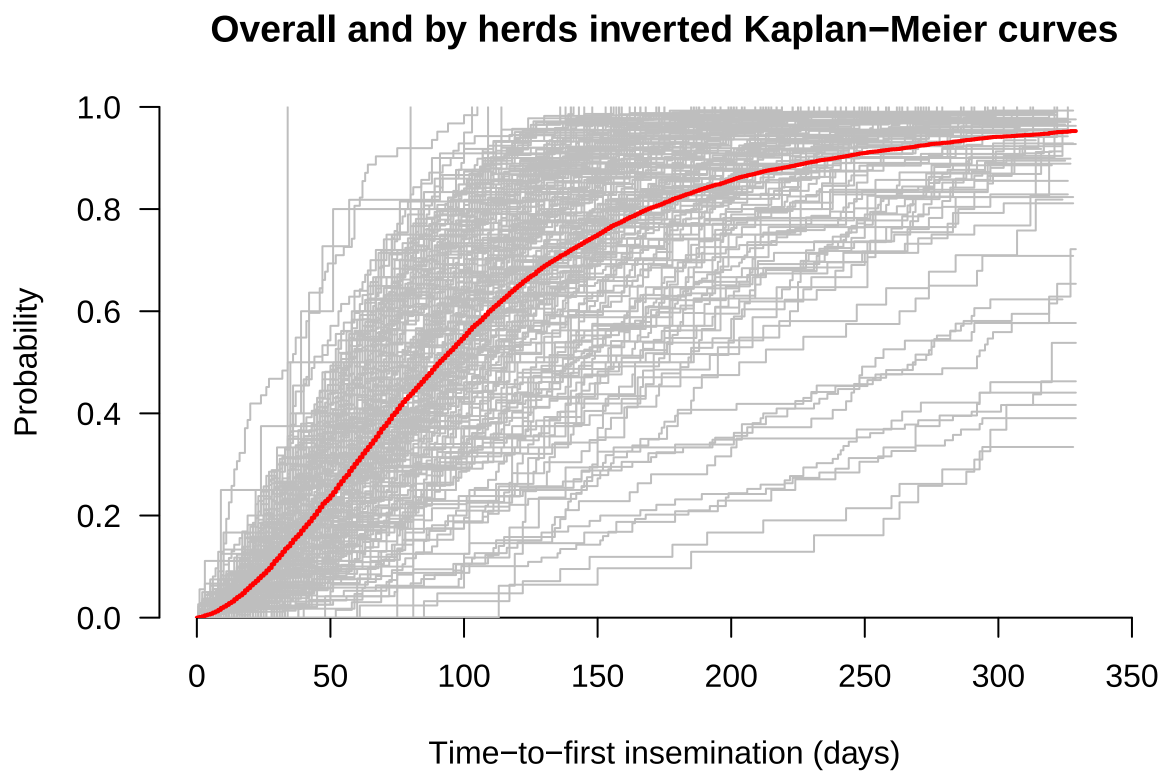


```
# kable(table_cluster) %>%
#   kableExtra::kable_styling("striped",
#                             position = "center") %>%
#     kableExtra::add_header_above(
#      c("Cluster size distribution" = 2))

# Summary cluster size
summary_cluster <-
  table_cluster %>%
  dplyr::summarise(min = min(n),
                   q25 = quantile(n, probs = .25),
                   mean = mean(n),
                   sd = sd(n),
                   median = median(n),
                   q75 = quantile(n, probs = .75),
                   max = max(n)) %>%
  round(., 1)

kable(summary_cluster) %>%
  kableExtra::kable_styling("striped") %>%
    kableExtra::add_header_above(
     c("Cluster size statistics" = 7))
```

Cluster size statistics

min	q25	mean	sd	median	q75	max
1	35	58.1	31.8	56	77	174

```r
# Event distribution per cluster
# kable(
#   addmargins(
#     table(insem$Herd,
#           insem$Status,
#           useNA = "ifany",
#           dnn = c("Cluster",
#                   "Status")))) %>%
#     kableExtra::kable_styling("striped") %>%
#   kableExtra::add_header_above(
#      c("Cluster" = 1,
#        "Status" =  2,
#        "Sum" = 1))

# survfit
sf_overall <- survival::survfit(Surv(Time,
                                     Status) ~ 1,
                                data = insem)

sf_cluster <- survival::survfit(Surv(Time,
                                     Status) ~ Herd,
                                data = insem)


par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(sf_cluster,
     col = "gray",
     fun = "event",
     xlim = c(0, 350),
     ylim = c(0, 1),
     bty = "n",
     xlab = "Time-to-first insemination (days)",
     ylab = "Probability",
     main = "Overall and by herds inverted Kaplan-Meier curves")
lines(sf_overall,
      fun = "event",
      lwd = 2,
      col = "red",
      conf.int = FALSE)
```



```r
# Time quantiles
q_overall <- quantile(sf_overall,
                      probs = c(.25, .50, .75))$quantile
q_cluster <-
  quantile(sf_cluster,
           probs = c(.25, .50, .75))$quantile

q_surv <- rbind.data.frame("Overall" = q_overall,
                           q_cluster)

kable(q_surv[1, ]) %>%
  kableExtra::kable_styling("striped",
                            position = "center") %>%
  kableExtra::add_header_above(
    c(" " = 1,
      "Time quantile" = 3))
```

	Time quantile		
	25	50	75
Overall	52	91	150

2. Analysis We provide the accuracy of predictions from the stratified and the shared gamma frailty Cox proportional hazards models in two scenarios

- Small cluster scenario: it includes herds with less than 50 cows
- Large cluster scenario: it includes herds with at least 50 cows

The accuracy of predictions are evaluated in terms of:

- Discrimination using time-varying AUC
- Calibration using O/E ratio, integrated calibration index (ICI), E50 and E90
- Overall performance using time-varying Index of Prediction Accuracy (IPA)

We use the Monte Carlo cross-validation procedure to assess prediction performances of the stratified and frailty Cox model.

- The sample was split into two parts to define derivation (70%) and validation samples (30%)
- The 70/30% random splits are repeated 100 times
- The repeated random splits are stratified by herds(i.e., the cluster) to obtain the same herds in both the derivation and validation samples

We excluded herds with less than 10 cows since it was not possible to obtain sufficient cows in both derivation and validation sample.

The mean of the performances metrics across the 100 random validation samples with the corresponding standard deviation are reported for each scenario and at each prediction time horizon to assess the prediction performances of the stratified and the frailty Cox models.

The fixed time horizons are defined based on the 25^{th}, 50^{th} and 75^{th} quantile fo event time overall distribution using the Kaplan-Meier distribution which corresponds to 51, 91, 150 days, respectively.

```r
# time horizons
time_hors <- c(52, 91, 150)

# reference values for continuous
# predictors
insem_ref <-
  insem %>%
  dplyr::mutate(
    Ureum_ref = Ureum - 2.5,
    Protein_ref = Protein - 2.5,
    Parity_ref = Parity - 2.5)


# Herds >= 50 cows
table_insem_small <-
  insem_ref %>%
  dplyr::count(Herd) %>%
  dplyr::filter(n < 50 & n >= 10)

df_insem_small <-
  insem_ref %>%
  dplyr::filter(Herd %in% table_insem_small$Herd)

# Repeat B times
B <- 110
df_tbl_insem_ext <- do.call(rbind,
                        lapply(seq(1:B),
                               function(k) {
                                 cbind(id = k, df_insem_small)}))
# Nested tibble
set.seed(20250924)
df_tbl_insem <-
  df_tbl_insem_ext %>%
  dplyr::group_nest(id) %>%
  dplyr::mutate(
    # Splitting
    split = purrr::map(.x = data,
                       ~ rsample::initial_split(.x,
                                                strata = Herd,
                                                prop = 0.7,
                                                pool = 0.001)),
    # Development
    development = purrr::map(.x = split,
                             ~ rsample::training(.x)),

    # Validation
    validation = purrr::map(.x = split,
                            ~ rsample::testing(.x)),

    # Check
    unique_cluster_dev = purrr::map(.x = development,
                                    ~ unique(.x$Herd)),
    unique_cluster_val = purrr::map(.x = validation,
                                    ~ unique(.x$Herd)),

    check = purrr::map2(.x = unique_cluster_dev,
                        .y = unique_cluster_val,
                        ~ all.equal(.x, .y)))


# table(do.call(rbind, df_tbl_insem$check))
rm(df_tbl_insem_ext)
# gc()
# check
# table(do.call(rbind, df_tbl_insem$check))
# all TRUE

# Rewrite B
B <- 100

# Models + predictions + performances
# tictoc::tic()
df_tbl_insem <-
  df_tbl_insem %>%
  dplyr::filter(check == TRUE) %>%
  dplyr::select(data,
                development,
                validation) %>%

  dplyr::mutate(
    cox_strat = purrr::map(.x = development,
                           ~ survival::coxph(Surv(Time, Status) ~
                                               Heifer +
                                               Parity_ref +
                                               Ureum_ref +
                                               Protein_ref +
                                               strata(Herd),
                                             data = .x,
                                             x = T,
                                             y = T,
                                             ties = "breslow")),

    cox_frail = purrr::map(.x = development,
                           ~ survival::coxph(Surv(Time, Status) ~
                                               Heifer +
                                               Parity_ref +
                                               Ureum_ref +
                                               Protein_ref +
                                               frailty(Herd),
                                             data = .x,
                                             x = T,
                                             y = T,
                                             ties = "breslow")),

    # predictions at t25/t50/t75
    # stratified cox
    pred_strat = purrr::map2(.x = cox_strat,
                             .y = validation,
                             ~ 1 - pec::predictSurvProb(.x,
                                                        newdata = .y,
                                                        times = time_hors)),
    # Extremes to make calibration suitable
    pred_strat = purrr::map(pred_strat,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # separate prediction by time horizons
    # due to possible missing values
    pred_strat_t25 = purrr::map(.x = pred_strat,
                                ~ .x[, 1]),

    pred_strat_t50 = purrr::map(.x = pred_strat,
                                ~ .x[, 2]),

    pred_strat_t75 = purrr::map(.x = pred_strat,
                                ~ .x[, 3]),

    pred_frail = purrr::map2(.x = cox_frail,
                             .y = validation,
                             ~ 1 - predict.coxph.gammafrail(model = .x,
                                                            newdata = .y,
                                                            cluster = "Herd",
                                                            times = time_hors)$conditional),
    # Extremes to make calibration suitable
    pred_frail = purrr::map(pred_frail,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # Score at t25/t50/t75 - frail
    score_frail = purrr::map2(.x = pred_frail,
                              .y = validation,
                              ~ riskRegression::Score(
                                list("frail" = .x),
                                data = .y,
                                formula = Surv(Time, Status) ~ 1,
                                metrics = "auc",
                                summary = "ipa",
                                times = time_hors)),

    # save results score - frail
    score_frail_res = purrr::map(score_frail,
                                 function(x) {

                                   res_AUC <-
```

```
                                    x$AUC$score %>%
                                    dplyr::select(model, times, AUC)

                                  res_IPA <-
                                    x$Brier$score %>%
                                    dplyr::select(model, times, IPA) %>%
                                    dplyr::filter(model %nin% c("Null model"))

                                  res <- cbind(res_AUC,
                                               "IPA" = res_IPA$IPA)

                                  return(res)
                                }),

 # Score at t25 - strata
score_strat_t25 = purrr::pmap(
   list(predictions = pred_strat_t25,
        df = validation,
        frm = list(as.formula(Surv(Time, Status) ~ 1)),
        t_hors = list(time_hors[1])),
   Score_tryCatch),


 # Score at t50 - strata
 score_strat_t50 = purrr::pmap(
   list(predictions = pred_strat_t50,
        df = validation,
        frm = list(as.formula(Surv(Time, Status) ~ 1)),
        t_hors = list(time_hors[2])),
   Score_tryCatch),

 # Score at t75 - strata
 score_strat_t75 = purrr::pmap(
   list(predictions = pred_strat_t75,
        df = validation,
        frm = list(as.formula(Surv(Time, Status) ~ 1)),
        t_hors = list(time_hors[3])),
   Score_tryCatch),


 # cloglog
 # cloglog - strat
 cloglog_pred_strat = purrr::map(pred_strat,
                                 function(x) {
                                   apply(x, 2, function(y) {
                                     log(-log(1 - y))
                                   })
                                 }),

# cloglog - strat t25/t50/t75
cloglog_pred_strat_t25 = purrr::map(.x = cloglog_pred_strat,
                                    ~ .x[, 1]),
cloglog_pred_strat_t50 = purrr::map(.x = cloglog_pred_strat,
                                    ~ .x[, 2]),

cloglog_pred_strat_t75 = purrr::map(.x = cloglog_pred_strat,
                                    ~ .x[, 3]),

 # cloglog
 # cloglog - frail
 cloglog_pred_frail = purrr::map(pred_frail,
                                function(x) {
                                  apply(x, 2, function(y) {
                                    log(-log(1 - y))
                                  })
                                }),
# cloglog - frail t25/t50/t75
 cloglog_pred_frail_t25 = purrr::map(.x = cloglog_pred_frail,
                                    ~ .x[, 1]),

 cloglog_pred_frail_t50 = purrr::map(.x = cloglog_pred_frail,
                                    ~ .x[, 2]),

 cloglog_pred_frail_t75 = purrr::map(.x = cloglog_pred_frail,
                                    ~ .x[, 3]),


 # average expected risk
 pred_avg_strat = purrr::map(pred_strat,
                            function(x){
                              apply(x, 2, mean)
                            }),

 pred_avg_frail = purrr::map(pred_frail,
                            function(x) {
                             apply(x, 2, mean)
                           }),

 # estimated actual risk (estimated obs proportion, observed)
 sf_obs = purrr::map(validation,
                    function(x) {
                      sf_sum <-
                        summary(survival::survfit(Surv(Time, Status) ~ 1,
                                                  data = x),
                               extend = TRUE,
                               times = time_hors)

                      sf_sum$surv[sf_sum$n.risk == 0] <- NA
                      sf_obs_event <- 1 - sf_sum$surv}),

# calibration

# OE ratio - strat
OE_strat = purrr::map2(.x = sf_obs,
                       .y = pred_avg_strat,
                      ~ .x / .y),

 # OE ratio - frail
OE_frail = purrr::map2(.x = sf_obs,
                       .y = pred_avg_frail,
                      ~ .x / .y),

# ICI/E50/E90 - frail t25
cal_meas_frail_t25  = purrr::pmap(
  list(time = purrr::map(validation, ~ .x$Time),
       status = purrr::map(validation, ~ .x$Status),
       covariates = cloglog_pred_frail_t25,
       t_hors = list(time_hors[1]),
       predictions = purrr::map(pred_frail, ~ .x[, 1])),
       calhaz_map),

# ICI/E50/E90 - frail t50
cal_meas_frail_t50  = purrr::pmap(
  list(time = purrr::map(validation, ~ .x$Time),
       status = purrr::map(validation, ~ .x$Status),
       covariates = cloglog_pred_frail_t50,
       t_hors = list(time_hors[2]),
       predictions = purrr::map(pred_frail, ~ .x[, 2])),
       calhaz_map),

# ICI/E50/E90 - frail t75
cal_meas_frail_t75  = purrr::pmap(
  list(time = purrr::map(validation, ~ .x$Time),
       status = purrr::map(validation, ~ .x$Status),
       covariates = cloglog_pred_frail_t75,
       t_hors = list(time_hors[3]),
       predictions = purrr::map(pred_frail, ~ .x[, 3])),
       calhaz_map),

# stratify
# ICI/E50/E90 - strat t25
cal_meas_strat_t25  = purrr::pmap(
  list(time = purrr::map(validation, ~ .x$Time),
       status = purrr::map(validation, ~ .x$Status),
       covariates = cloglog_pred_strat_t25,
       t_hors = list(time_hors[1]),
       predictions = purrr::map(pred_strat, ~ .x[, 1])),
       calhaz_map),

# ICI/E50/E90 - strat t50
cal_meas_strat_t50  = purrr::pmap(
  list(time = purrr::map(validation, ~ .x$Time),
       status = purrr::map(validation, ~ .x$Status),
       covariates = cloglog_pred_strat_t50,
       t_hors = list(time_hors[2]),
       predictions = purrr::map(pred_strat, ~ .x[, 2])),
       calhaz_map),
# ICI/E50/E90 - strat t75
cal_meas_strat_t75  = purrr::pmap(
  list(time = purrr::map(validation, ~ .x$Time),
       status = purrr::map(validation, ~ .x$Status),
       covariates = cloglog_pred_strat_t75,
       t_hors = list(time_hors[3]),
       predictions = purrr::map(pred_strat, ~ .x[, 3])),
       calhaz_map))

# tictoc::toc()
```

2.1 Small cluster size

Small cluster scenario

			AUC		IPA	
model	times	pct_comp	AUC_mean	AUC_sd	IPA_mean	IPA_sd
frail	52	100	0.688	0.015	0.075	0.016
strat	52	95	0.658	0.015	0.039	0.021
frail	91	100	0.730	0.016	0.161	0.024
strat	91	78	0.723	0.016	0.146	0.026
frail	150	100	0.813	0.016	0.262	0.033

Small cluster scenario

			O-E ratio	
model	times	pct_comp	OE_ratio_mean	OE_ratio_sd
frail	52	100	0.994	0.055
strat	52	95	1.024	0.087
frail	91	100	0.999	0.059
strat	91	78	1.023	0.045
frail	150	100	0.987	0.056

Small cluster scenario

			ICI		E50		E90	
model	times	pct_comp	ICI_mean	ICI_sd	E50_mean	E50_sd	E90_mean	E90_sd
frail	52	100	0.025	0.013	0.023	0.013	0.046	0.013
strat	52	95	0.037	0.011	0.032	0.013	0.071	0.013
frail	91	100	0.022	0.011	0.021	0.011	0.038	0.011
strat	91	78	0.037	0.014	0.033	0.014	0.064	0.014
frail	150	100	0.022	0.011	0.020	0.011	0.039	0.011

```r
# time horizons
time_hors <- c(52, 91, 150)

# reference values for continuous
# predictors
insem_ref <-
  insem %>%
  dplyr::mutate(
    Ureum_ref = Ureum - 2.5,
    Protein_ref = Protein - 2.5,
    Parity_ref = Parity - 2.5)


# Herds >= 50 cows
table_insem_large <-
  insem_ref %>%
  dplyr::count(Herd) %>%
  dplyr::filter(n >= 50)

df_insem_large <-
  insem_ref %>%
  dplyr::filter(Herd %in% table_insem_large$Herd)

# Repeat B times
B <- 100
df_tbl_insem_ext <- do.call(rbind,
                        lapply(seq(1:B),
                               function(k) {
                                 cbind(id = k, df_insem_large)}))
# Nested tibble
set.seed(20250924)
df_tbl_insem <-
  df_tbl_insem_ext %>%
  dplyr::group_nest(id) %>%
  dplyr::mutate(
    # Splitting
    split = purrr::map(.x = data,
                       ~ rsample::initial_split(.x,
                                                strata = Herd,
                                                prop = 0.7)),
    # Development
    development = purrr::map(.x = split,
                             ~ rsample::training(.x)),

    # Validation
    validation = purrr::map(.x = split,
                            ~ rsample::testing(.x)),

    # Check
    unique_cluster_dev = purrr::map(.x = development,
                                    ~ unique(.x$Herd)),
    unique_cluster_val = purrr::map(.x = validation,
                                    ~ unique(.x$Herd)),

    check = purrr::map2(.x = unique_cluster_dev,
                        .y = unique_cluster_val,
                        ~ all.equal(.x, .y)))



rm(df_tbl_insem_ext)
# gc()
# check
# table(do.call(rbind, df_tbl_insem$check))
# all TRUE

# Models + predictions + performances
# tictoc::tic()
df_tbl_insem <-
  df_tbl_insem %>%
  dplyr::select(data,
              development,
              validation) %>%

  dplyr::mutate(
    cox_strat = purrr::map(.x = development,
                          ~ survival::coxph(Surv(Time, Status) ~
                                             Heifer +
                                             Parity_ref +
                                             Ureum_ref +
                                             Protein_ref +
                                             strata(Herd),
                                           data = .x,
                                           x = T,
                                           y = T,
                                           ties = "breslow")),

    cox_frail = purrr::map(.x = development,
                          ~ survival::coxph(Surv(Time, Status) ~
                                             Heifer +
                                             Parity_ref +
                                             Ureum_ref +
                                             Protein_ref +
                                             frailty(Herd),
                                           data = .x,
                                           x = T,
                                           y = T,
                                           ties = "breslow")),

    # predictions at t25/t50/t75
    # stratified cox
    pred_strat = purrr::map2(.x = cox_strat,
                            .y = validation,
                            ~ 1 - pec::predictSurvProb(.x,
                                                  newdata = .y,
                                                  times = time_hors)),
    # Extremes to make calibration suitable
    pred_strat = purrr::map(pred_strat,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # separate prediction by time horizons
    # due to possible missing values
    pred_strat_t25 = purrr::map(.x = pred_strat,
                                ~ .x[, 1]),

    pred_strat_t50 = purrr::map(.x = pred_strat,
                                ~ .x[, 2]),

    pred_strat_t75 = purrr::map(.x = pred_strat,
                                ~ .x[, 3]),

    pred_frail = purrr::map2(.x = cox_frail,
                             .y = validation,
                             ~ 1 - predict.coxph.gammafrail(model = .x,
                                                     newdata = .y,
                                                     cluster = "Herd",
                                                     times = time_hors)$conditional),
    # Extremes to make calibration suitable
    pred_frail = purrr::map(pred_frail,
                            function(x) {
                              x <- ifelse(x == 0, 0.000001, x)
                              x <- ifelse(x == 1, 0.999999, x)
                              x}),

    # Score at t25/t50/t75 - frail
    score_frail = purrr::map2(.x = pred_frail,
                              .y = validation,
                              ~ riskRegression::Score(
                                list("frail" = .x),
                                data = .y,
                                formula = Surv(Time, Status) ~ 1,
                                metrics = "auc",
                                summary = "ipa",
                                times = time_hors)),

    # save results score - frail
    score_frail_res = purrr::map(score_frail,
                                 function(x) {

                                   res_AUC <-
                                     x$AUC$score %>%
                                     dplyr::select(model, times, AUC)

                                   res_IPA <-
```

```
                                  x$Brier$score %>%
                                  dplyr::select(model, times, IPA) %>%
                                  dplyr::filter(model %nin% c("Null model"))

                                res <- cbind(res_AUC,
                                             "IPA" = res_IPA$IPA)

                                return(res)
                              }),

 # Score at t25 - strata
score_strat_t25 = purrr::pmap(
   list(predictions = pred_strat_t25,
        df = validation,
        frm = list(as.formula(Surv(Time, Status) ~ 1)),
        t_hors = list(time_hors[1])),
   Score_tryCatch),


 # Score at t50 - strata
 score_strat_t50 = purrr::pmap(
   list(predictions = pred_strat_t50,
        df = validation,
        frm = list(as.formula(Surv(Time, Status) ~ 1)),
        t_hors = list(time_hors[2])),
   Score_tryCatch),

 # Score at t75 - strata
 score_strat_t75 = purrr::pmap(
   list(predictions = pred_strat_t75,
        df = validation,
        frm = list(as.formula(Surv(Time, Status) ~ 1)),
        t_hors = list(time_hors[3])),
   Score_tryCatch),


 # cloglog
 # cloglog - strat
 cloglog_pred_strat = purrr::map(pred_strat,
                                 function(x) {
                                   apply(x, 2, function(y) {
                                     log(-log(1 - y))
                                   })
                                 }),

# cloglog - strat t25/t50/t75
cloglog_pred_strat_t25 = purrr::map(.x = cloglog_pred_strat,
                                    ~ .x[, 1]),

cloglog_pred_strat_t50 = purrr::map(.x = cloglog_pred_strat,
                                    ~ .x[, 2]),

cloglog_pred_strat_t75 = purrr::map(.x = cloglog_pred_strat,
                                    ~ .x[, 3]),

    # cloglog
    # cloglog - frail
    cloglog_pred_frail = purrr::map(pred_frail,
                                   function(x) {
                                     apply(x, 2, function(y) {
                                       log(-log(1 - y))
                                     })
                                   }),
   # cloglog - frail t25/t50/t75
    cloglog_pred_frail_t25 = purrr::map(.x = cloglog_pred_frail,
                                        ~ .x[, 1]),

    cloglog_pred_frail_t50 = purrr::map(.x = cloglog_pred_frail,
                                        ~ .x[, 2]),

    cloglog_pred_frail_t75 = purrr::map(.x = cloglog_pred_frail,
                                        ~ .x[, 3]),


    # average expected risk
    pred_avg_strat = purrr::map(pred_strat,
                               function(x){
                                 apply(x, 2, mean)
                               }),

    pred_avg_frail = purrr::map(pred_frail,
                               function(x) {
                                apply(x, 2, mean)
                              }),

    # estimated actual risk (estimated obs proportion, observed)
    sf_obs = purrr::map(validation,
                       function(x) {
                         sf_sum <-
                           summary(survival::survfit(Surv(Time, Status) ~ 1,
                                                         data = x),
                                   extend = TRUE,
                                   times = time_hors)

                         sf_sum$surv[sf_sum$n.risk == 0] <- NA
                         sf_obs_event <- 1 - sf_sum$surv}),

   # calibration

   # OE ratio - strat
   OE_strat = purrr::map2(.x = sf_obs,
                          .y = pred_avg_strat,
                          ~ .x / .y),

    # OE ratio - frail
   OE_frail = purrr::map2(.x = sf_obs,
                           .y = pred_avg_frail,
                           ~ .x / .y),

   # ICI/E50/E90 - frail t25
   cal_meas_frail_t25  = purrr::pmap(
     list(time = purrr::map(validation, ~ .x$Time),
          status = purrr::map(validation, ~ .x$Status),
          covariates = cloglog_pred_frail_t25,
          t_hors = list(time_hors[1]),
          predictions = purrr::map(pred_frail, ~ .x[, 1])),
          calhaz_map),

   # ICI/E50/E90 - frail t50
   cal_meas_frail_t50  = purrr::pmap(
     list(time = purrr::map(validation, ~ .x$Time),
          status = purrr::map(validation, ~ .x$Status),
          covariates = cloglog_pred_frail_t50,
          t_hors = list(time_hors[2]),
          predictions = purrr::map(pred_frail, ~ .x[, 2])),
          calhaz_map),

   # ICI/E50/E90 - frail t75
   cal_meas_frail_t75  = purrr::pmap(
     list(time = purrr::map(validation, ~ .x$Time),
          status = purrr::map(validation, ~ .x$Status),
          covariates = cloglog_pred_frail_t75,
          t_hors = list(time_hors[3]),
          predictions = purrr::map(pred_frail, ~ .x[, 3])),
          calhaz_map),

   # stratify
   # ICI/E50/E90 - strat t25
   cal_meas_strat_t25  = purrr::pmap(
     list(time = purrr::map(validation, ~ .x$Time),
          status = purrr::map(validation, ~ .x$Status),
          covariates = cloglog_pred_strat_t25,
          t_hors = list(time_hors[1]),
          predictions = purrr::map(pred_strat, ~ .x[, 1])),
          calhaz_map),

   # ICI/E50/E90 - strat t50
   cal_meas_strat_t50  = purrr::pmap(
     list(time = purrr::map(validation, ~ .x$Time),
          status = purrr::map(validation, ~ .x$Status),
          covariates = cloglog_pred_strat_t50,
          t_hors = list(time_hors[2]),
          predictions = purrr::map(pred_strat, ~ .x[, 2])),
          calhaz_map),

   # ICI/E50/E90 - strat t75
   cal_meas_strat_t75  = purrr::pmap(
     list(time = purrr::map(validation, ~ .x$Time),
          status = purrr::map(validation, ~ .x$Status),
          covariates = cloglog_pred_strat_t75,
          t_hors = list(time_hors[3]),
          predictions = purrr::map(pred_strat, ~ .x[, 3])),
          calhaz_map))

# tictoc::toc()
```

2.2 Large cluster size

Large cluster scenario						
			AUC		IPA	
model	times	pct_comp	AUC_mean	AUC_sd	IPA_mean	IPA_sd
frail	52	100	0.670	0.010	0.068	0.009
strat	52	100	0.656	0.009	0.055	0.009
frail	91	100	0.706	0.008	0.130	0.011
strat	91	97	0.701	0.008	0.124	0.011
frail	150	100	0.776	0.010	0.201	0.017

Large cluster scenario				
			O-E ratio	
model	times	pct_comp	OE_ratio_mean	OE_ratio_sd
frail	52	100	1.005	0.028
strat	52	100	1.015	0.045
frail	91	100	1.001	0.029
strat	91	97	1.011	0.024
frail	150	100	0.998	0.030

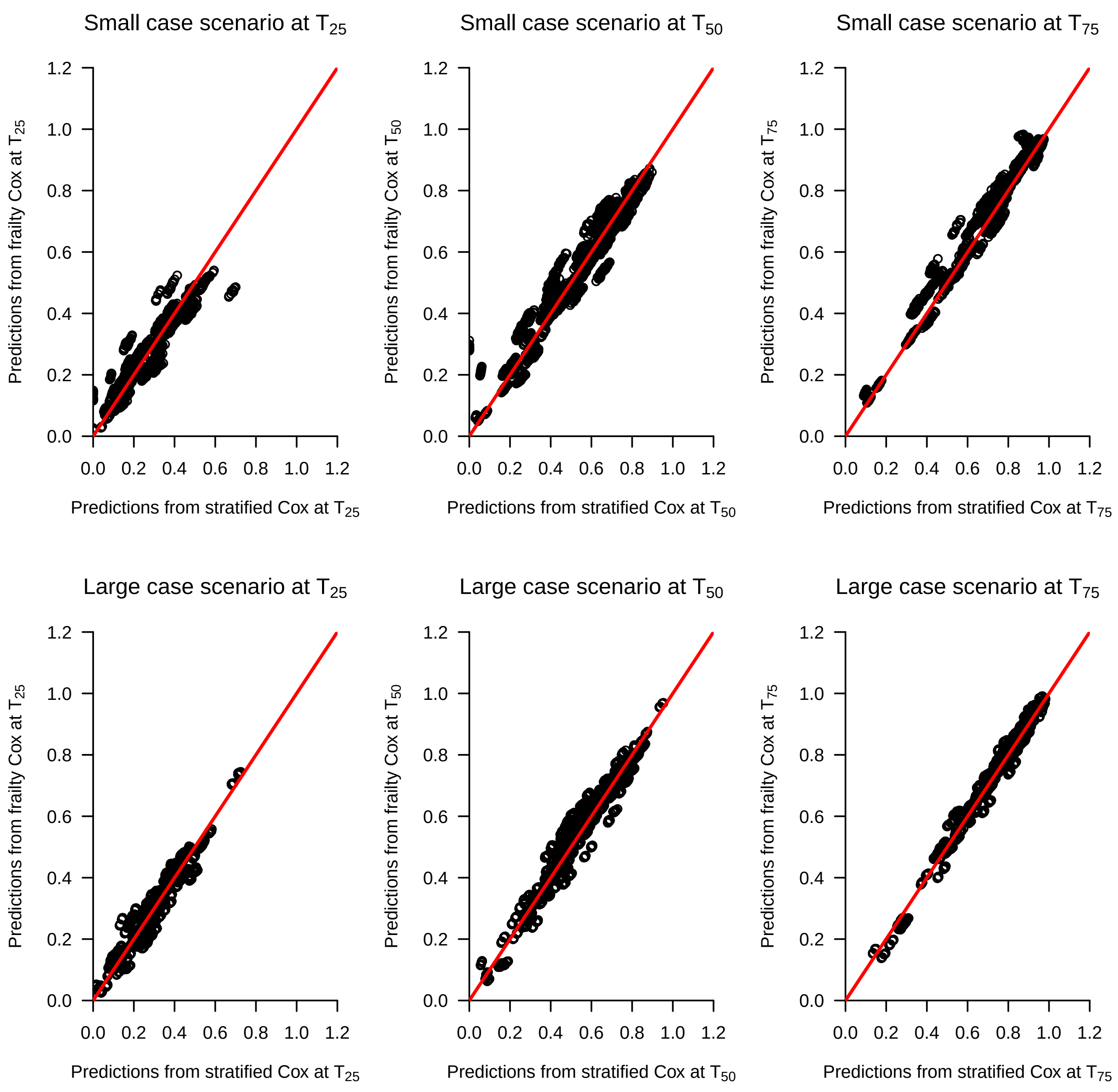
Large cluster scenario								
			ICI		E50		E90	
model	times	pct_comp	ICI_mean	ICI_sd	E50_mean	E50_sd	E90_mean	E90_sd
frail	52	100	0.013	0.014	0.012	0.008	0.023	0.021
strat	52	100	0.026	0.047	0.023	0.039	0.048	0.087
frail	91	100	0.013	0.018	0.011	0.006	0.024	0.041
strat	91	97	0.020	0.009	0.019	0.010	0.036	0.018
frail	150	100	0.015	0.023	0.012	0.007	0.028	0.069

3. Additional investigations We additionally estimate predictions from the stratified and the shared gamma frailty Cox model using all data without splitting among clusters in which predictions were possible in both models.

We provide the corresponding plots of predictions at different time horizons from the stratified versus the shared gamma frailty model in the small and large cluster scenario.

```
# Herds < 50 (herds >= 10)
table_insem_small <-
  insem %>%
```

```r
  dplyr::count(Herd) %>%
  dplyr::filter(n < 50 & n >= 10)

# Herds >= 50 cows
table_insem_large <-
  insem %>%
  dplyr::count(Herd) %>%
  dplyr::filter(n >= 50)

df_insem <-
  insem %>%
  # dplyr::filter(Herd %in% table_insem_50$Herd) %>%
  # reference values for all continuous variable
  # to estimate predictions from frailty correctly
  dplyr::mutate(
    Ureum_ref = Ureum - 2.5,
    Protein_ref = Protein - 2.5,
    Parity_ref = Parity - 2.5)

# Small scenario overall data
df_insem_small <-
  df_insem %>%
  dplyr::filter(Herd %in% table_insem_small$Herd)

# Large scenario overall data
df_insem_large <-
  df_insem %>%
  dplyr::filter(Herd %in% table_insem_large$Herd)

# Stratified Cox - small
cox_strat_small <-
  survival::coxph(Surv(Time, Status) ~
                  Heifer +
                  Parity_ref +
                  Ureum_ref +
                  Protein_ref +
                  strata(Herd),
      data = df_insem_small,
      x = T,
      y = T,
      ties = "breslow")

# Frailty Cox - small
cox_frail_small <-
  survival::coxph(Surv(Time, Status) ~
                  Heifer +
                  Parity_ref +
                  Ureum_ref +
                  Protein_ref +
                  frailty(Herd),
      data = df_insem_small,
      x = T,
      y = T,
      ties = "breslow")

# Stratified Cox - large
cox_strat_large <-
  survival::coxph(Surv(Time, Status) ~
                    Heifer +
                    Parity_ref +
                    Ureum_ref +
                    Protein_ref +
                    strata(Herd),
      data = df_insem_large,
      x = T,
      y = T,
      ties = "breslow")

# Frailty Cox - small
cox_frail_large <-
  survival::coxph(Surv(Time, Status) ~
                    Heifer +
                    Parity_ref +
                    Ureum_ref +
                    Protein_ref +
                    frailty(Herd),
      data = df_insem_large,
      x = T,
      y = T,
      ties = "breslow")

## Predictions
# at 25th, 50th, 75th overall percentiles
time_hors <- c(52, 91, 150)
# time_hors <- q_surv[1, ]

# Predictions
# Stratified small
pred_strat_small <- 1 - pec::predictSurvProb(cox_strat_small,
                                             newdata = df_insem_small,
                                             times = time_hors)

# Frailty small
pred_frail_small <- 1 - predict.coxph.gammafrail(model = cox_frail_small,
                                                 newdata = df_insem_small,
                                                 cluster = "Herd",
                                                 times = time_hors)$conditional

# Predictions
# Stratified large
pred_strat_large <- 1 - pec::predictSurvProb(cox_strat_large,
                                             newdata = df_insem_large,
                                             times = time_hors)

# Frailty large
pred_frail_large <- 1 - predict.coxph.gammafrail(model = cox_frail_large,
                                                 newdata = df_insem_large,
                                                 cluster = "Herd",
                                                 times = time_hors)$conditional

# Plot predictions
# from stratify and frailty model

# Small scenario at T25, T50 and T75
oldpar <-
  par(
    mfrow = c(2, 3),
    xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_small[, 1],
     pred_frail_small[, 1],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at T"[25]),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at T"[25]),
     main = expression("Small case scenario at T"[25]),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_small[, 2],
     pred_frail_small[, 2],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at T"[50]),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at T"[50]),
     main = expression("Small case scenario at T"[50]),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_small[, 3],
     pred_frail_small[, 3],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at T"[75]),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at T"[75]),
     main = expression("Small case scenario at T"[75]),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

# Large scenario at T25, T50 and T75
par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_large[, 1],
     pred_frail_large[, 1],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at T"[25]),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at T"[25]),
     main = expression("Large case scenario at T"[25]),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_large[, 2],
     pred_frail_large[, 2],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at T"[50]),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at T"[50]),
     main = expression("Large case scenario at T"[50]),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
       col = "red")

par(xaxs = "i",
    yaxs = "i",
    las = 1)
plot(pred_strat_large[, 3],
     pred_frail_large[, 3],
     xlim = c(0, 1.2),
     xlab = expression("Predictions from stratified Cox at T"[75]),
     ylim = c(0, 1.2),
     ylab = expression("Predictions from frailty Cox at T"[75]),
     main = expression("Large case scenario at T"[75]),
     bty = "n")
abline(a = 0,
       b = 1,
       lwd = 2,
        col = "red")
```



NOTE: comparisons only among clusters where predictions were possible for both models

Data availability statement

The first case study dataset was collected by the Center for International Blood and Marrow Transplant Research (CIBMTR) which is supported primarily by the Public Health Service U24CA076518 from the National Cancer Institute; the National Heart, Lung, and Blood Institute; the National Institute of Allergy and Infectious Diseases; 75R60222C00011 from the Health Resources and Services Administration; N00014-23-1-2057 and N00014-24-1-2507 from the Office of Naval Research; NMDP; and the Medical College of Wisconsin.

Reproducibility ticket

```r
sessionInfo()
```

```
## R version 4.3.3 (2024-02-29 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19045)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.utf8
## [2] LC_CTYPE=English_United Kingdom.utf8
## [3] LC_MONETARY=English_United Kingdom.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.utf8
##
## time zone: Europe/Rome
## tzcode source: internal
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] frailtyHL_2.3              cmprsk_2.2-12
##  [3] Matrix_1.6-5               parfm_2.7.8
##  [5] optimx_2025-4.9            survival_3.8-3
##  [7] webshot2_0.1.2             rms_6.8-0
##  [9] patchwork_1.2.0            ggpubr_0.6.0
## [11] plotly_4.10.4              webshot_0.5.5
## [13] gridExtra_2.3              lubridate_1.9.4
## [15] forcats_1.0.0              dplyr_1.1.4
## [17] purrr_1.0.2                readr_2.1.5
## [19] tidyr_1.3.1                tibble_3.2.1
## [21] ggplot2_3.5.2              tidyverse_2.0.0
## [23] gtsummary_1.7.2            polspline_1.1.25
## [25] riskRegression_2025.05.20 lattice_0.22-5
## [27] Hmisc_5.2-3                kableExtra_1.4.0
## [29] knitr_1.50                 stringr_1.5.1
## [31] rio_1.2.3                  pacman_0.5.1
##
```

```
## loaded via a namespace (and not attached):
##   [1] RColorBrewer_1.1-3   rstudioapi_0.17.1    jsonlite_1.8.9
##   [4] shape_1.4.6.1        magrittr_2.0.3       TH.data_1.1-3
##   [7] farver_2.1.2         nloptr_2.2.1         rmarkdown_2.29
##  [10] vctrs_0.6.5          base64enc_0.1-3      rstatix_0.7.2
##  [13] tinytex_0.57         htmltools_0.5.8.1    broom_1.0.9
##  [16] Formula_1.2-5        parallelly_1.45.1    pracma_2.4.4
##  [19] htmlwidgets_1.6.4    sandwich_3.1-1       zoo_1.8-14
##  [22] gt_1.0.0             lifecycle_1.0.4      iterators_1.0.14
##  [25] pkgconfig_2.0.3      R6_2.6.1             fastmap_1.2.0
##  [28] future_1.67.0        digest_0.6.35        numDeriv_2016.8-1.1
##  [31] colorspace_2.1-1     ps_1.9.1             rprojroot_2.1.0
##  [34] timechange_0.3.0     httr_1.4.7           abind_1.4-8
##  [37] compiler_4.3.3       here_1.0.1           withr_3.0.2
##  [40] htmlTable_2.4.3      backports_1.5.0      carData_3.0-5
##  [43] ggsignif_0.6.4       MASS_7.3-60.0.1      lava_1.8.1
##  [46] quantreg_6.1         tools_4.3.3          chromote_0.5.1
##  [49] foreign_0.8-90       future.apply_1.20.0  nnet_7.3-20
##  [52] glue_1.7.0           mets_1.3.5           nlme_3.1-164
##  [55] promises_1.3.2       checkmate_2.3.2      cluster_2.1.8.1
##  [58] generics_0.1.4       gtable_0.3.6         tzdb_0.5.0
##  [61] sn_2.1.1             websocket_1.4.2      data.table_1.16.2
##  [64] hms_1.1.3            xml2_1.3.6           car_3.1-3
##  [67] foreach_1.5.2        pillar_1.11.0        later_1.4.2
##  [70] splines_4.3.3        SparseM_1.84-2       tidyselect_1.2.1
##  [73] svglite_2.1.3        stats4_4.3.3         xfun_0.52
##  [76] expm_1.0-0           stringi_1.8.4        lazyeval_0.2.2
##  [79] yaml_2.3.10          pec_2025.06.24       evaluate_1.0.4
##  [82] codetools_0.2-20     msm_1.8.2            cli_3.6.2
##  [85] rpart_4.1.24         systemfonts_1.2.3    processx_3.8.6
##  [88] Rcpp_1.1.0           globals_0.18.0       parallel_4.3.3
##  [91] MatrixModels_0.5-4   listenv_0.9.1        glmnet_4.1-10
##  [94] viridisLite_0.4.2    broom.helpers_1.15.0 mvtnorm_1.3-3
##  [97] timereg_2.0.6        scales_1.4.0         prodlim_2025.04.28
## [100] rlang_1.1.6          multcomp_1.4-28      mnormt_2.1.1
```