biniLasso vignette

AUTHORS
Abdollah Safari
Hamed Helisaz

1 Introduction

The biniLasso and its sparse variant, miniLasso, are novel methods for prognostic analysis of high-dimensional survival data that enable detection of multiple cut-points per continuous feature to categorize them for obtaining more interpretable results from prediction models. This approach leverages the Cox proportional hazards model with two key innovations: (1) a cumulative binarization scheme with L1-penalized coefficients operating on context-dependent cut-point candidates, and (2) for miniLasso, additional uniLasso regularization, a recently developed two-stage regularized regression, to enforce sparsity while preserving univariate coefficient patterns. For details, see the original paper by Safari et al. available at https://arxiv.org/abs/2503.16687

2 Initial setup

```
library(tidyverse)
library(magrittr)
library(survival)
library(glmnet)
library(pec)
library(uniLasso)
library(kableExtra)
library(biniLasso)

gbm_data_fnl <- read_rds("data/gbm_fnl.rds")
brca_data_fnl <- read_rds("data/brca_fnl.rds")
kirc_data_fnl <- read_rds("data/kirc_fnl.rds")

colnames(gbm_data_fnl) <-
  stringr::str_replace_all(colnames(gbm_data_fnl), "-", "_")
colnames(brca_data_fnl) <-
  stringr::str_replace_all(colnames(brca_data_fnl), "-", "_")
colnames(kirc_data_fnl) <-
```

```
  stringr::str_replace_all(colnames(kirc_data_fnl), "-", "_")


bussy_est <- read.csv("data/binacox_tcga_cuts.csv",
                      header = TRUE, row.names = NULL)
Bussy_cuts <-
  bussy_est[-1, ] %>%
  select(data, cut_points_estimate_x) %>%
  rowwise %>%
  mutate(opt_cuts_bina = list(cuts_to_list(cut_points_estimate_x))) %>%
  select(data, opt_cuts_bina) %>%
  ungroup

names(Bussy_cuts$opt_cuts_bina[Bussy_cuts$data == "GBM"][[1]]) <-
  colnames(gbm_data_fnl)[! colnames(gbm_data_fnl) %in% c("barcode", "vital_status", "tte")]
names(Bussy_cuts$opt_cuts_bina[Bussy_cuts$data == "KIRC"][[1]]) <-
  colnames(kirc_data_fnl)[! colnames(kirc_data_fnl) %in% c("barcode", "vital_status", "tte")]
names(Bussy_cuts$opt_cuts_bina[Bussy_cuts$data == "BRCA"][[1]]) <-
  colnames(brca_data_fnl)[! colnames(brca_data_fnl) %in% c("barcode", "vital_status", "tte")]

n_bins <- 50
set.seed(12345)
```

3 Data preproccessing

Before fitting the model to extract optimal cut-points, one needs to specify cut-points candidates. We will do this by using the num_to_cat function. This can be done either by simply specifying number of bins for numeric covariates (set n_bins argument), or more explicitly, passing the candidate cut-points per covariate by setting the cuts_list argument (see function help for more details and a simple example). Here, we use the former and set the number of bins to 50 for all genes.

```
gbm_converted_obj <-
  cumBinarizer(data = gbm_data_fnl,
               cols = colnames(gbm_data_fnl)[! colnames(gbm_data_fnl) %in%
                                               c("barcode", "vital_status", "tte")],
               method = "quantile",
               n_bins = 50)
brca_converted_obj <-
  cumBinarizer(data = brca_data_fnl,
```

```
            cols = colnames(brca_data_fnl)[! colnames(brca_data_fnl) %in%
                                           c("barcode", "vital_status", "tte")],
            method = "quantile",
            n_bins = 50)
kirc_converted_obj <-
  cumBinarizer(data = kirc_data_fnl,
            cols = colnames(kirc_data_fnl)[! colnames(kirc_data_fnl) %in%
                                           c("barcode", "vital_status", "tte")],
            method = "quantile",
            n_bins = 50)
```

Candidate cut-points for the first gene (COPS7B) in the GBM dataset:

```
gbm_converted_obj$x_cuts[[1]][[1]]
```

```
 [1] -0.120676799 -0.111887338 -0.105920688 -0.101116373 -0.097607231
 [6] -0.095670007 -0.094208786 -0.089869405 -0.089061306 -0.086847336
[11] -0.081633436 -0.079574444 -0.077205496 -0.075799625 -0.074559802
[16] -0.072788626 -0.071493453 -0.069069156 -0.068072869 -0.064364469
[21] -0.060202206 -0.052076935 -0.045180418 -0.037586501 -0.025797110
[26] -0.010066852 -0.003912015  0.027426731  0.033005936  0.042448519
[31]  0.067078936  0.090491670  0.107849196  0.131184441  0.149117599
[36]  0.164737158  0.179360430  0.207699248  0.218005279  0.325183572
[41]  0.370658518  0.467453291  0.568211071  0.646928778  0.708731754
[46]  0.777231989  0.983053720  2.064810586  3.174076031  3.678285584
```

Candidate cut-points for the second gene (CYP3A7_CYP3A51P) in the GBM dataset:

```
gbm_converted_obj$x_cuts[[1]][[2]]
```

```
 [1] 0.1128850 0.1410298 0.1474293 0.1519350 0.1584651 0.1621220 0.1652564
 [8] 0.1695010 0.1820388 0.2002578 0.2105754 0.2154077 0.2255946 0.2418546
[15] 0.2715013 0.2914834 0.3366718 0.3506462 0.4023646 0.4070663 0.4095478
[22] 0.4286157 0.4517323 0.5712332 0.6358160 0.6624588 0.7786294 0.9674801
[29] 1.0773818 1.1146035 1.3831868 1.8145657 2.0600324 2.6422568 3.1741340
[36] 4.4748008 7.1954394
```

As it can be seen, the first gene has 50 candidate cut-points as it was indicated in the input arguments of the cumBinarizer function. However, for the second gene, due to limited number of unique values in the gene expression data in the given

gene, the function returns only 37 candidate cut-points, and therefore, the counts of different levels of the resulting categorical variables after categorization will be unbalanced.

4 Obtain optimal cut-points

After generating candidate cut-points, the next step is to identify which cut-points are optimal based on the outcome variable. This is done using the `opt_cuts_finder()` function, which fits Lasso-regularized Cox models to select the most informative thresholds. The argument `method = "both"` runs both biniLasso and miniLasso at once, so their results can be compared directly. The function also uses cross-validation to choose the best regularization parameter. The output provides a list of estimated cut-points for each gene across all datasets (GBM, BRCA, and KIRC). The results are then displayed in tables showing the detected cut-points from both methods, making it easy to compare how biniLasso and miniLasso differ in their selected thresholds.

```
gbm_cuts_comp <-
  opt_cuts_finder(x = gbm_converted_obj$x,
                  y = survival::Surv(gbm_data_fnl$tte, gbm_data_fnl$vital_status),
                  method = "both",
                  family = "cox",
                  lasso_rule = "min",
                  lasso_nfolds = 10,
                  cols = colnames(gbm_data_fnl)[! colnames(gbm_data_fnl) %in%
                                                  c("barcode", "vital_status", "tte")],
                  x_cuts = gbm_converted_obj$x_cuts)
brca_cuts_comp <-
  opt_cuts_finder(x = brca_converted_obj$x,
                  y = survival::Surv(brca_data_fnl$tte, brca_data_fnl$vital_status),
                  method = "both",
                  family = "cox",
                  lasso_rule = "min",
                  lasso_nfolds = 10,
                  cols = colnames(brca_data_fnl)[! colnames(brca_data_fnl) %in%
                                                  c("barcode", "vital_status", "tte")],
                  x_cuts = brca_converted_obj$x_cuts)
kirc_cuts_comp <-
  opt_cuts_finder(x = kirc_converted_obj$x,
                  y = survival::Surv(kirc_data_fnl$tte, kirc_data_fnl$vital_status),
                  method = "both",
```

```
                  family = "cox",
                  lasso_rule = "min",
                  lasso_nfolds = 10,
                  cols = colnames(kirc_data_fnl)[! colnames(kirc_data_fnl) %in%
                                                   c("barcode", "vital_status", "tte")],
                  x_cuts = kirc_converted_obj$x_cuts)

Bussy_cuts$opt_cuts <-
  gbm_cuts_comp$opt_cuts[gbm_cuts_comp$method == "biniLasso"]
Bussy_cuts$opt_cuts[2] <-
  brca_cuts_comp$opt_cuts[brca_cuts_comp$method == "biniLasso"]
Bussy_cuts$opt_cuts[3] <-
  kirc_cuts_comp$opt_cuts[kirc_cuts_comp$method == "biniLasso"]

gbm_cuts_comp %>%
  select(method, opt_cuts) %>%
  rowwise %>%
  mutate(cuts_est_selected = list(unlist(opt_cuts)[! is.na(unlist(opt_cuts))])) %>%
  ungroup %>%
  unnest_longer(cuts_est_selected) %>%
  rowwise %>%
  mutate(gene = unlist(strsplit(cuts_est_selected_id, "_"))[1]) %>%
  group_by(method, gene) %>%
  reframe(cuts = paste0(sort(round(cuts_est_selected, 3)), collapse = " , ")) %>%
  pivot_wider(id_cols = gene,
              names_from = method,
              values_from = cuts) %>%
  rowwise %>%
  mutate(n_bini = as.numeric(! is.na(biniLasso)) * str_count(biniLasso, ","),
         n_Sbini = as.numeric(! is.na(miniLasso)) * str_count(miniLasso, ",")) %>%
  ungroup %>%
  arrange(desc(n_bini), desc(n_Sbini)) %>%
  select(gene, biniLasso, miniLasso) %>%
  kable(Caption = "Optimal cut-points found by different methods for the GBM daatset") %>%
  kable_styling()
```

gene	biniLasso	miniLasso
PTPRN2	-0.974 , -0.901 , -0.601 , -0.141 , 0.941	-0.141 , 0.941

gene	biniLasso	miniLasso
LBH	-0.461 , -0.163 , 1.089	-0.163 , -0.099 , 0.879
AC008875.3	-0.715 , -0.536 , -0.294	-0.294 , 0.109
SCARA3	-0.626 , -0.558 , -0.446	0.025
AL592064.1	1.865 , 2.605	0.691 , 1.3
CASC20	-0.012 , 0.022	-0.012 , 0.005
FUT4	0.781 , 1.049	0.781 , 1.049
HPCAL1	0.01 , 0.556	0.01 , 0.556
AC073332.1	-0.588 , 0.314	0.148
AC083906.5	1.292 , 3.19	1.292
AC090114.1	0.041 , 1.397	1.397
PLK2	0.37 , 0.628	0.628
SSX7	3.126 , 3.274	3.274
BMP2	-0.497 , -0.37	NA
TSPAN13	0.494	0.184 , 0.494
CLEC5A	0.49	0.49
CPPED1	-0.853	-0.107
CPQ	0.323	0.323
DUSP6	0.347	0.347
ID1	3.227	3.227
INTS6P1	-0.67	1.224
LINC00906	1.164	1.164
P2RY6	0.454	0.454
PARP4P3	0.467	0.568
PODNL1	0.219	0.219
RNF175	0.591	0.591
SLC20A1	1.419	1.419
SLC43A3	1.035	1.035

gene	biniLasso	miniLasso
TNFSF14	2.082	2.082
ZMIZ1	1.384	1.384
CYB561	-1.062	NA
KCNMB3P1	-0.259	NA
KCNN4	0.5	NA
L2HGDH	-0.275	NA
LINC01674	2.851	NA
NCSTNP1	3.434	NA
TBX2	2.11	NA
TP73	-0.954	NA
VDR	1.237	NA
GARS1P1	NA	2.904 , 3.446
ABI1	NA	1.693
MTHFD2	NA	0.03
TPTEP1	NA	0.171

```
brca_cuts_comp %>%
  select(method, opt_cuts) %>%
  rowwise %>%
  mutate(cuts_est_selected = list(unlist(opt_cuts)[! is.na(unlist(opt_cuts))])) %>%
  ungroup %>%
  unnest_longer(cuts_est_selected) %>%
  rowwise %>%
  mutate(gene = unlist(strsplit(cuts_est_selected_id, "_"))[1]) %>%
  group_by(method, gene) %>%
  reframe(cuts = paste0(sort(round(cuts_est_selected, 3)), collapse = " , ")) %>%
  pivot_wider(id_cols = gene,
              names_from = method,
              values_from = cuts) %>%
  rowwise %>%
  mutate(n_bini = as.numeric(! is.na(biniLasso)) * str_count(biniLasso, ","),
```

```
        n_Sbini = as.numeric(! is.na(miniLasso)) * str_count(miniLasso, ",")) %>%
  ungroup %>%
  arrange(desc(n_bini), desc(n_Sbini)) %>%
  select(gene, biniLasso, miniLasso) %>%
  kable(Caption = "Optimal cut-points found by different methods for the BRCA daatset") %>%
  kable_styling()
```

gene	biniLasso	miniLasso
MAPT	-0.482 , -0.381 , -0.268	-0.268 , 1.429
LIMCH1	0.085 , 1.229	0.085 , 0.422 , 1.229
AL355864.1	-0.373 , 0.35	0.35
HNRNPC	-0.677 , 0.021	0.021
PSME1	-0.697 , -0.3	NA
ABCB5	2.334	1.115 , 3.286
AL033397.1	-0.045	-0.045
ANO6	-0.126	-0.126
CCNI2	1.139	1.139
CYRIA	2.298	2.654
FGF7	1.577	1.577
LINC02159	1.055	1.055
NFKB2	1.984	1.984
PICALM	0.706	0.706
PSME2	0.127	0.127
PSME2P1	-0.235	-0.11
SPPL2C	0.371	0.371
STXBP1	1.221	1.221
TAPBP	0.002	0.002
TMEM163	0.455	0.455
TMEM164	0.596	0.596

gene	biniLasso	miniLasso
AL096701.1	-0.396	NA
CLTA	-0.586	NA
FBXO6	-1.236	NA
MAGEB4	0.466	NA
NFKBIE	-0.415	NA
POLR2G	-0.565	NA
PPIB	-0.89	NA
SLIT3	-0.748	NA
ABCA1	NA	2.103
EXOC1	NA	0.694
GEMIN6	NA	1.601
NT5E	NA	0.716
RPLP1	NA	2.18
STX7	NA	0.321

```
kirc_cuts_comp %>%
  select(method, opt_cuts) %>%
  bind_rows(Bussy_cuts %>%
              mutate(method = "Binacox") %>%
              filter(data == "KIRC") %>%
              select(method, opt_cuts = opt_cuts_bina)) %>%
  rowwise %>%
  mutate(cuts_est_selected = list(unlist(opt_cuts)[! is.na(unlist(opt_cuts))])) %>%
  ungroup %>%
  unnest_longer(cuts_est_selected) %>%
  rowwise %>%
  mutate(gene = unlist(strsplit(cuts_est_selected_id, "_"))[1]) %>%
  group_by(method, gene) %>%
  reframe(cuts = paste0(sort(round(cuts_est_selected, 3)), collapse = " , ")) %>%
  pivot_wider(id_cols = gene,
              names_from = method,
              values_from = cuts) %>%
```

```
  rowwise %>%
  mutate(n_bini = as.numeric(! is.na(biniLasso)) * str_count(biniLasso, ","),
         n_Sbini = as.numeric(! is.na(miniLasso)) * str_count(miniLasso, ","),
         n_bina = as.numeric(! is.na(`Binacox`)) * str_count(`Binacox`, ",")) %>%
  ungroup %>%
  arrange(desc(n_bini), desc(n_Sbini), desc(n_bina)) %>%
  select(gene, biniLasso, miniLasso, Binacox) %>%
  kable(Caption = "Optimal cut-points found by different methods for the KIRC daatset") %>%
  kable_styling()
```

gene	biniLasso	miniLasso	Binacox
DLGAP1	-0.162 , 0.229 , 0.55	0.451	-0.165
IL4	-0.149 , 0.332 , 1.392	1.392	NA
ANAPC7	0.423 , 0.594	0.423	NA
MBOAT7	-0.112 , 0.871	0.508	NA
SGCB	0.199	0.199	-0.33 , 0.2
CARS1	0.25	0.25	0.258
CUBN	0.655	0.655	-0.201
EIF4EBP2	0.407	0.338	0.408
SLC2A9	-0.066	-0.066	-0.137
CKAP4	0.927	1.294	NA
HJURP	0.186	0.186	NA
IMPDH1	1.961	1.961	NA
TXLNA	-0.045	-0.045	NA
DONSON	0.104	NA	0.071
SORBS2	-0.656	NA	-0.654
AR	0.775	NA	NA
CYP3A7	-0.555	NA	NA
DVL3	0.339	NA	NA
HES7	1.38	NA	NA

gene	biniLasso	miniLasso	Binacox
LIN54	-0.284	NA	NA
MGAT2P1	4.22	NA	NA
PTPRB	-0.76	NA	NA
SNORD100	-0.11	NA	NA
USB1	0.126	NA	NA
ADH5	NA	NA	-0.271
GIPC2	NA	NA	-0.11
MGAM	NA	NA	-0.266
MSH3	NA	NA	-0.386
MXD3	NA	NA	0.001
NCKAP5L	NA	NA	-0.032
SELENOP	NA	NA	-0.42
SLC16A12	NA	NA	-0.735
SLC27A2	NA	NA	-0.179

4.1 Compare models' performance

Once the optimal cut-points are obtained, this section evaluates how well the resulting models perform. The function `biniFit()` is used to fit Cox proportional hazards models based on the detected cut-points. The fitted models from biniLasso, miniLasso, and Binacox are compared using three performance measures: the Akaike Information Criterion (AIC), the Integrated Brier Score (IBS), and the Concordance Index (C-index). The AIC reflects the trade-off between model fit and complexity, the IBS measures prediction accuracy over time, and the C-index evaluates the model's ability to rank survival times correctly. The results show that biniLasso and miniLasso consistently outperform Binacox, providing lower AIC and IBS values and higher C-index scores. These findings confirm the improved predictive and computational performance of the proposed methods.

```
gbm_cuts_comp %>%
  select(method, opt_cuts) %>%
  bind_rows(Bussy_cuts %>%
                mutate(method = "Binacox") %>%
                filter(data == "GBM") %>%
```

```
              select(method, opt_cuts, opt_cuts_bina)) %>%
group_by(method) %>%
unnest_longer(opt_cuts) %>%
mutate(`n-cuts` = if_else(method == "Binacox",
                          sum(! is.na(unlist(opt_cuts_bina))),
                          sum(! is.na(unlist(opt_cuts))))) %>%
group_by(method, `n-cuts`) %>%
do(gbm_biniFit =
     list(biniFit(data = gbm_data_fnl,
             optCuts = .,
             y = Surv(gbm_data_fnl$tte, gbm_data_fnl$vital_status),
             family = "cox",
             col_cuts = "opt_cuts",
             col_x = "opt_cuts_id")$fit),
   gbm_biniFit_bina =
     list(biniFit(data = gbm_data_fnl,
             optCuts = .,
             y = Surv(gbm_data_fnl$tte, gbm_data_fnl$vital_status),
             family = "cox",
             col_cuts = "opt_cuts_bina",
             col_x = "opt_cuts_id")$fit),
   gbm_bini_dataFit =
     list(biniFit(data = gbm_data_fnl,
             optCuts = .,
             y = Surv(gbm_data_fnl$tte, gbm_data_fnl$vital_status),
             family = "cox",
             col_cuts = "opt_cuts",
             col_x = "opt_cuts_id")$dataFit)) %>%
rowwise %>%
mutate(AIC = if_else(method == "Binacox",
                     round(AIC(gbm_biniFit_bina[[1]]), 0),
                     round(AIC(gbm_biniFit[[1]]), 0)),
       IBS = ibs(pec(list(Cox = gbm_biniFit[[1]]),
                     Hist(time, event) ~ 1,
                     data = gbm_bini_dataFit[[1]],
                     verbose = F))[if_else(method == "Binacox", 1, 2), 1],
       `C-index` = if_else(method == "Binacox",
                           paste0(round(concordance(gbm_biniFit_bina[[1]])$concordance, 3),
                          " (", round(sqrt(concordance(gbm_biniFit_bina[[1]])$var), 3),
                          ")"),
```

```
                          paste0(round(concordance(gbm_biniFit[[1]])$concordance, 3),
                          " (", round(sqrt(concordance(gbm_biniFit[[1]])$var), 3),
                          ")"))) %>%
  ungroup %>%
  mutate(Dataset = "GBM") %>%
  relocate(Dataset, .before = "method") %>%
  select(! c(gbm_biniFit, gbm_biniFit_bina, gbm_bini_dataFit)) %>%
  kable(digits = 3,
        Caption = "Compare AIC of fitted Cox models by using detected optimal cut-points - GBM dataset") %>%
  kable_styling()
```

Dataset	method	n-cuts	AIC	IBS	C-index
GBM	Binacox	0	3027	0.064	0.5 (0)
GBM	biniLasso	59	2739	0.030	0.8 (0.013)
GBM	miniLasso	43	2803	0.037	0.779 (0.015)

```
brca_cuts_comp %>%
  select(method, opt_cuts) %>%
  bind_rows(Bussy_cuts %>%
              mutate(method = "Binacox") %>%
              filter(data == "BRCA") %>%
              select(method, opt_cuts, opt_cuts_bina)) %>%
  group_by(method) %>%
  unnest_longer(opt_cuts) %>%
  mutate(`n-cuts` = if_else(method == "Binacox",
                          sum(! is.na(unlist(opt_cuts_bina))),
                          sum(! is.na(unlist(opt_cuts))))) %>%
  group_by(method, `n-cuts`) %>%
  do(brca_biniFit =
       list(biniFit(data = brca_data_fnl,
              optCuts = .,
              y = Surv(brca_data_fnl$tte, brca_data_fnl$vital_status),
              family = "cox",
              col_cuts = "opt_cuts",
              col_x = "opt_cuts_id")$fit),
     brca_biniFit_bina =
       list(biniFit(data = brca_data_fnl,
```

```
              optCuts = .,
              y = Surv(brca_data_fnl$tte, brca_data_fnl$vital_status),
              family = "cox",
              col_cuts = "opt_cuts_bina",
              col_x = "opt_cuts_id")$fit),
    brca_bini_dataFit =
      list(biniFit(data = brca_data_fnl,
              optCuts = .,
              y = Surv(brca_data_fnl$tte, brca_data_fnl$vital_status),
              family = "cox",
              col_cuts = "opt_cuts",
              col_x = "opt_cuts_id")$dataFit)) %>%
  rowwise %>%
  mutate(AIC = if_else(method == "Binacox",
                       round(AIC(brca_biniFit_bina[[1]]), 0),
                       round(AIC(brca_biniFit[[1]]), 0)),
         IBS = ibs(pec(list(Cox = brca_biniFit[[1]]),
                       Hist(time, event) ~ 1,
                       data = brca_bini_dataFit[[1]],
                       verbose = F))[if_else(method == "Binacox", 1, 2), 1],
         `C-index` = if_else(method == "Binacox",
                             paste0(round(concordance(brca_biniFit_bina[[1]])$concordance, 3),
                            " (", round(sqrt(concordance(brca_biniFit_bina[[1]])$var), 3),
                            ")"),
                            paste0(round(concordance(brca_biniFit[[1]])$concordance, 3),
                            " (", round(sqrt(concordance(brca_biniFit[[1]])$var), 3),
                            ")"))) %>%
  ungroup %>%
  mutate(Dataset = "BRCA") %>%
  relocate(Dataset, .before = "method") %>%
  select(! c(brca_biniFit, brca_biniFit_bina, brca_bini_dataFit)) %>%
  kable(digits = 3,
        Caption = "Compare AIC of fitted Cox models by using detected optimal cut-points - BRCA dataset") %>%
  kable_styling()
```

Dataset	method	n-cuts	AIC	IBS	C-index
BRCA	Binacox	0	2109	0.163	0.5 (0)
BRCA	biniLasso	35	1974	0.089	0.772 (0.018)

Dataset	method	n-cuts	AIC	IBS	C-index
BRCA	miniLasso	30	1966	0.096	0.761 (0.017)

```
kirc_cuts_comp %>%
  select(method, opt_cuts) %>%
  bind_rows(Bussy_cuts %>%
               mutate(method = "Binacox") %>%
               filter(data == "KIRC") %>%
               select(method, opt_cuts = opt_cuts_bina)) %>%
  group_by(method) %>%
  unnest_longer(opt_cuts) %>%
  mutate(`n-cuts` = sum(! is.na(unlist(opt_cuts)))) %>%
  group_by(method, `n-cuts`) %>%
  do(kirc_biniFit =
       list(biniFit(data = kirc_data_fnl,
              optCuts = .,
              y = Surv(kirc_data_fnl$tte, kirc_data_fnl$vital_status),
              family = "cox",
              col_cuts = "opt_cuts",
              col_x = "opt_cuts_id")$fit),
     kirc_bini_dataFit =
       list(biniFit(data = kirc_data_fnl,
              optCuts = .,
              y = Surv(kirc_data_fnl$tte, kirc_data_fnl$vital_status),
              family = "cox",
              col_cuts = "opt_cuts",
              col_x = "opt_cuts_id")$dataFit)) %>%
  rowwise %>%
  mutate(AIC = round(AIC(kirc_biniFit[[1]]), 0),
         IBS = ibs(pec(list(Cox = kirc_biniFit[[1]]),
                       Hist(time, event) ~ 1,
                       data = kirc_bini_dataFit[[1]],
                       verbose = F))[2, 1],
         `C-index` = paste0(round(concordance(kirc_biniFit[[1]])$concordance, 3),
                            " (", round(sqrt(concordance(kirc_biniFit[[1]])$var), 3),
                            ")")) %>%
  ungroup %>%
  mutate(Dataset = "KIRC") %>%
  relocate(Dataset, .before = "method") %>%
```

```
  select(! c(kirc_biniFit, kirc_bini_dataFit)) %>%
  kable(digits = 3,
        Caption = "Compare AIC of fitted Cox models by using detected optimal cut-points - KIRC dataset") %>%
  kable_styling()
```

Dataset	method	n-cuts	AIC	IBS	C-index
KIRC	Binacox	18	2109	0.154	0.736 (0.018)
KIRC	biniLasso	30	2073	0.123	0.777 (0.016)
KIRC	miniLasso	13	2069	0.152	0.764 (0.017)

5 Fixed number of optimal cut-points

In some applications, it is desirable to limit the number of cut-points per variable for easier interpretation. This section explains how to use the `opt_fixed_nCuts()` function to restrict the model to a fixed number of thresholds—here, at most two per feature. The function identifies the best-performing cut-points within this constraint while maintaining good predictive ability. The results are summarized in tables showing the selected thresholds for each gene in all datasets. As it can be seen, limiting the number of cut-points makes the final model simpler and more interpretable, while still keeping its predictive performance close to the unrestricted version. This approach is particularly useful in clinical studies where fewer, well-defined thresholds are preferred.

```
gbm_fixedCuts_comp <-
 opt_fixed_nCuts(x = gbm_converted_obj$x,
                 y = survival::Surv(gbm_data_fnl$tte, gbm_data_fnl$vital_status),
                 max_nCuts = 2,
                 method = "both",
                 family = "cox",
                 lasso_rule = "min",
                 lasso_nfolds = 10,
                 cols = colnames(gbm_data_fnl)[! colnames(gbm_data_fnl) %in%
                                               c("barcode", "vital_status", "tte")],
                 x_cuts = gbm_converted_obj$x_cuts)
brca_fixedCuts_comp <-
 opt_fixed_nCuts(x = brca_converted_obj$x,
                 y = survival::Surv(brca_data_fnl$tte, brca_data_fnl$vital_status),
                 max_nCuts = 2,
```

```
                method = "both",
                family = "cox",
                lasso_rule = "min",
                lasso_nfolds = 10,
                cols = colnames(brca_data_fnl)[! colnames(brca_data_fnl) %in%
                                    c("barcode", "vital_status", "tte")],
                x_cuts = brca_converted_obj$x_cuts)
kirc_fixedCuts_comp <-
 opt_fixed_nCuts(x = kirc_converted_obj$x,
                y = survival::Surv(kirc_data_fnl$tte, kirc_data_fnl$vital_status),
                max_nCuts = 2,
                method = "both",
                family = "cox",
                lasso_rule = "min",
                lasso_nfolds = 10,
                cols = colnames(kirc_data_fnl)[! colnames(kirc_data_fnl) %in%
                                    c("barcode", "vital_status", "tte")],
                x_cuts = kirc_converted_obj$x_cuts)


gbm_fixedCuts_comp %>%
  select(method, opt_cuts) %>%
  rowwise %>%
  mutate(cuts_est_selected = list(unlist(opt_cuts)[! is.na(unlist(opt_cuts))])) %>%
  ungroup %>%
  unnest_longer(cuts_est_selected) %>%
  rowwise %>%
  mutate(ind_tmp = gregexpr("_ENSG", cuts_est_selected_id)[[1]][1] - 1,
         gene = substr(cuts_est_selected_id, 1, ind_tmp)) %>%
  group_by(method, gene) %>%
  reframe(cuts = paste0(sort(round(cuts_est_selected, 3)), collapse = " , ")) %>%
  pivot_wider(id_cols = gene,
              names_from = method,
              values_from = cuts) %>%
  rowwise %>%
  mutate(n_bini = as.numeric(! is.na(biniLasso)) * str_count(biniLasso, ","),
         n_Sbini = as.numeric(! is.na(miniLasso)) * str_count(miniLasso, ",")) %>%
  ungroup %>%
  arrange(desc(n_bini), desc(n_Sbini)) %>%
  select(gene, biniLasso, miniLasso) %>%
```

```
  kable(Caption = "Optimal cut-points found by different methods for the GBM daatset") %>%
  kable_styling()
```

gene	biniLasso	miniLasso
AC008875.3	-0.536 , -0.294	-0.294 , 0.167
GARS1P1	0.932 , 1.823	2.775 , 3.446
LBH	-0.163 , 0.879	-0.163 , 0.879
PTPRN2	-0.601 , -0.141	-0.141 , 0.941
SSX7	3.126 , 3.566	3.126 , 3.566
AC073332.1	-0.588 , 0.148	0.148
AC083906.5	1.232 , 3.19	1.232
AL592064.1	1.47	0.691 , 1.3
HPCAL1	0.556	0.01 , 0.556
SCARA3	-0.426	-0.426 , 0.025
ABI1	1.693	1.693
AC090114.1	1.397	1.397
CLEC5A	0.49	0.49
CPQ	0.208	0.323
DUSP6	0.347	0.347
FUT4	1.049	1.049
HTR7	-0.258	-0.258
ID1	3.227	3.227
INTS6P1	-0.67	1.224
KLKP1	-0.302	1.711
LINC00906	1.164	1.164
MTHFD2	0.03	0.03
P2RY6	0.454	0.454
PARP4P3	0.568	0.568

gene	biniLasso	miniLasso
PLK2	0.37	0.628
RNF175	-0.687	0.591
SLC20A1	1.419	1.419
TNFSF14	2.082	2.082
TSPAN13	0.184	0.184
ZMIZ1_AS1	1.384	1.384
AC003688.2	3.96	NA
ANKH	-0.667	NA
CFTR	1.638	NA
CPPED1	-0.853	NA
CYB561	-1.062	NA
KCNN4	0.5	NA
L2HGDH	-0.275	NA
LINC01674	2.851	NA
NCSTNP1	4.812	NA
OSMR	-0.953	NA
SNRPB	2.247	NA
TBX2_AS1	2.11	NA
TP73_AS1	-0.954	NA
SLC43A3	NA	1.035

It should be noted that some genes show only one threshold or `NA` instead of two. This happens for two main reasons. In cases where two detected thresholds are very close to each other, the algorithm treats them as overlapping and merges them into a single effective cut-point, since they represent nearly the same boundary in the variable's range. This is often observed when the predictor has a limited range or when the outcome does not vary much between nearby values. On the other hand, `NA` values occur when no meaningful threshold is detected for that variable. This typically happens when the variable shows no strong association with the outcome after penalization, or when its effect is roughly linear and cannot be improved by categorization. In both cases, the algorithm avoids introducing unnecessary or redundant thresholds, resulting in a simpler and more interpretable model.

```
brca_fixedCuts_comp %>%
  select(method, opt_cuts) %>%
  rowwise %>%
  mutate(cuts_est_selected = list(unlist(opt_cuts)[! is.na(unlist(opt_cuts))])) %>%
  ungroup %>%
  unnest_longer(cuts_est_selected) %>%
  rowwise %>%
  mutate(ind_tmp = gregexpr("_ENSG", cuts_est_selected_id)[[1]][1] - 1,
         gene = substr(cuts_est_selected_id, 1, ind_tmp)) %>%
  group_by(method, gene) %>%
  reframe(cuts = paste0(sort(round(cuts_est_selected, 3)), collapse = " , ")) %>%
  pivot_wider(id_cols = gene,
              names_from = method,
              values_from = cuts) %>%
  rowwise %>%
  mutate(n_bini = as.numeric(! is.na(biniLasso)) * str_count(biniLasso, ","),
         n_Sbini = as.numeric(! is.na(miniLasso)) * str_count(miniLasso, ",")) %>%
  ungroup %>%
  arrange(desc(n_bini), desc(n_Sbini)) %>%
  select(gene, biniLasso, miniLasso) %>%
  kable(Caption = "Optimal cut-points found by different methods for the BRCA daatset") %>%
  kable_styling()
```

gene	biniLasso	miniLasso
MAPT_AS1	-0.381 , -0.268	-0.268 , 1.429
AL096701.1	-0.396 , 0.161	0.161
AL355864.1	-0.301 , 0.35	0.35
EXOC1	-0.22 , 0.694	0.694
LINC00488	0.058 , 0.141	0.058
RFTN1P1	2.07 , 2.933	2.07
AC136601.1	0.908 , 2.152	NA
ANO6	-0.126 , 1.273	NA
POLR2G	-1.148 , -0.565	NA
AL033397.1	0.006	-0.045 , 0.006

gene	biniLasso	miniLasso
PNMA6B	0.453	0.453 , 0.587
PSME2P1	-0.235	-0.235 , -0.11
ABCB5	3.286	2.334
AC091078.1	2.959	2.959
AL049548.1	1.487	1.351
C2orf91	0.142	0.142
CCNI2	1.139	1.139
CLTA	-0.562	-0.562
FANCF	-0.04	-0.04
GEMIN6	1.601	1.601
LIMCH1	0.422	0.422
LINC02159	1.055	1.055
MAGEB4	0.466	0.466
MAPKAPK3	-0.545	-0.545
PARP12	1.046	1.046
PICALM	0.706	0.706
SPPL2C	0.371	0.371
TAPBP	-0.048	-0.048
TMEM163	0.455	0.455
TMEM164	0.667	0.596
ZNF25	1.384	1.384
AC034159.2	-0.049	NA
AC133065.2	-0.238	NA
AKR1B10	-0.183	NA
DAXX	0.344	NA
FBXO6	-1.236	NA
NT5E	-0.258	NA

gene	biniLasso	miniLasso
PPIB	-0.89	NA
PSME2	0.127	NA
STX7	0.321	NA
TPMT	-0.85	NA
CYRIA	NA	2.298
HNRNPC	NA	0.021
NFKB2	NA	1.984
PRR32	NA	0.171
PSME2P2	NA	0.328
PTPN18	NA	1.013
RIOX1	NA	1.114
RPLP1	NA	2.18
STXBP1	NA	1.221
TMEM223	NA	1.421

```
kirc_fixedCuts_comp %>%
  select(method, opt_cuts) %>%
  rowwise %>%
  mutate(cuts_est_selected = list(unlist(opt_cuts)[! is.na(unlist(opt_cuts))])) %>%
  ungroup %>%
  unnest_longer(cuts_est_selected) %>%
  rowwise %>%
  mutate(ind_tmp = gregexpr("_ENSG", cuts_est_selected_id)[[1]][1] - 1,
         gene = substr(cuts_est_selected_id, 1, ind_tmp)) %>%
  group_by(method, gene) %>%
  reframe(cuts = paste0(sort(round(cuts_est_selected, 3)), collapse = " , ")) %>%
  pivot_wider(id_cols = gene,
              names_from = method,
              values_from = cuts) %>%
  rowwise %>%
  mutate(n_bini = as.numeric(! is.na(biniLasso)) * str_count(biniLasso, ","),
         n_Sbini = as.numeric(! is.na(miniLasso)) * str_count(miniLasso, ",")) %>%
```

```
  ungroup %>%
  arrange(desc(n_bini), desc(n_Sbini)) %>%
  select(gene, biniLasso, miniLasso) %>%
  kable(Caption = "Optimal cut-points found by different methods for the KIRC daatset") %>%
  kable_styling()
```

gene	biniLasso	miniLasso
DLGAP1_AS2	-0.162 , 0.229	0.55
MBOAT7	-0.112 , 0.508	0.508
CYP3A7	-0.555 , 0.043	NA
RORA	-0.446 , 0.183	NA
SORBS2	-0.656 , -0.247	NA
ANAPC7	0.423	0.423
AR	-0.45	0.693
CARS1	0.25	0.25
CUBN	0.655	0.655
EIF4EBP2	0.338	0.407
HJURP	0.186	0.186
IL4	1.392	1.392
IMPDH1	1.961	1.577
SGCB	0.199	0.199
SLC2A9	-0.066	-0.066
TXLNA	-0.045	0.184
USB1	0.493	0.331
AC080129.2	1.218	NA
AL118508.2	0.179	NA
CRYZ	0.736	NA
CYP3A7_CYP3A51P	-0.041	NA
DONSON	0.104	NA

gene	biniLasso	miniLasso
DVL3	0.339	NA
LIN54	-0.284	NA
MGAT2P1	4.22	NA
PTPRB	-0.76	NA
PURA	0.208	NA
SNORD100	-0.11	NA
SPTBN1	-0.066	NA
TFEC	-0.246	NA
TRIM27	0.091	NA
CKAP4	NA	1.294
HES7	NA	2.896

5.1 Compare models' performance:

The final part summarizes the main outcomes of the vignette and presents concise comparison tables for the GBM, BRCA, and KIRC datasets. The tables show the detected cut-points across biniLasso, miniLasso, and Binacox, confirming that the proposed cumulative binarization and uniLasso regularization result in models that are both interpretable and computationally efficient. The findings in this section are consistent with the simulation and real-data results presented in the main article, demonstrating the practical advantages of biniLasso and miniLasso for high-dimensional survival analysis.

```
gbm_fixedCuts_comp %>%
  select(method, opt_cuts) %>%
  group_by(method) %>%
  unnest_longer(opt_cuts) %>%
  mutate(`n-cuts` = sum(! is.na(unlist(opt_cuts)))) %>%
  group_by(method, `n-cuts`) %>%
  do(gbm_biniFit =
       list(biniFit(data = gbm_data_fnl,
                    optCuts = .,
                    y = Surv(gbm_data_fnl$tte, gbm_data_fnl$vital_status),
                    family = "cox",
                    col_cuts = "opt_cuts",
                    col_x = "opt_cuts_id")$fit),
```

```
    gbm_bini_dataFit =
      list(biniFit(data = gbm_data_fnl,
             optCuts = .,
             y = Surv(gbm_data_fnl$tte, gbm_data_fnl$vital_status),
             family = "cox",
             col_cuts = "opt_cuts",
             col_x = "opt_cuts_id")$dataFit)) %>%
  rowwise %>%
  mutate(AIC = round(AIC(gbm_biniFit[[1]]), 0),
         IBS = ibs(pec(list(Cox = gbm_biniFit[[1]]),
                       Hist(time, event) ~ 1,
                       data = gbm_bini_dataFit[[1]],
                       verbose = F))[2, 1],
         `C-index` = paste0(round(concordance(gbm_biniFit[[1]])$concordance, 3),
                            " (", round(sqrt(concordance(gbm_biniFit[[1]])$var), 3),
                            ")")) %>%
  ungroup %>%
  mutate(Dataset = "GBM") %>%
  relocate(Dataset, .before = "method") %>%
  select(! c(gbm_biniFit, gbm_bini_dataFit)) %>%
  kable(digits = 3,
        Caption = "Compare AIC of fitted Cox models by using detected optimal cut-points - GBM dataset") %>%
  kable_styling()
```

Dataset	method	n-cuts	AIC	IBS	C-index
GBM	biniLasso	50	2801	0.034	0.777 (0.015)
GBM	miniLasso	39	2806	0.037	0.777 (0.015)

```
brca_fixedCuts_comp %>%
  select(method, opt_cuts) %>%
  group_by(method) %>%
  unnest_longer(opt_cuts) %>%
  mutate(`n-cuts` = sum(! is.na(unlist(opt_cuts)))) %>%
  group_by(method, `n-cuts`) %>%
  do(brca_biniFit =
       list(biniFit(data = brca_data_fnl,
              optCuts = .,
              y = Surv(brca_data_fnl$tte, brca_data_fnl$vital_status),
```

```
              family = "cox",
              col_cuts = "opt_cuts",
              col_x = "opt_cuts_id")$fit),
    brca_bini_dataFit =
      list(biniFit(data = brca_data_fnl,
              optCuts = .,
              y = Surv(brca_data_fnl$tte, brca_data_fnl$vital_status),
              family = "cox",
              col_cuts = "opt_cuts",
              col_x = "opt_cuts_id")$dataFit)) %>%
  rowwise %>%
  mutate(AIC = round(AIC(brca_biniFit[[1]]), 0),
         IBS = ibs(pec(list(Cox = brca_biniFit[[1]]),
                       Hist(time, event) ~ 1,
                       data = brca_bini_dataFit[[1]],
                       verbose = F))[2, 1],
         `C-index` = paste0(round(concordance(brca_biniFit[[1]])$concordance, 3),
                            " (", round(sqrt(concordance(brca_biniFit[[1]])$var), 3),
                            ")")) %>%
  ungroup %>%
  mutate(Dataset = "BRCA") %>%
  relocate(Dataset, .before = "method") %>%
  select(! c(brca_biniFit, brca_bini_dataFit)) %>%
  kable(digits = 3,
        Caption = "Compare AIC of fitted Cox models by using detected optimal cut-points - BRCA dataset") %>%
  kable_styling()
```

Dataset	method	n-cuts	AIC	IBS	C-index
BRCA	biniLasso	50	1956	0.078	0.786 (0.016)
BRCA	miniLasso	42	1958	0.084	0.783 (0.016)

```
kirc_fixedCuts_comp %>%
  select(method, opt_cuts) %>%
  group_by(method) %>%
  unnest_longer(opt_cuts) %>%
  mutate(`n-cuts` = sum(! is.na(unlist(opt_cuts)))) %>%
  group_by(method, `n-cuts`) %>%
  do(kirc_biniFit =
```

```
      list(biniFit(data = kirc_data_fnl,
             optCuts = .,
             y = Surv(kirc_data_fnl$tte, kirc_data_fnl$vital_status),
             family = "cox",
             col_cuts = "opt_cuts",
             col_x = "opt_cuts_id")$fit),
   kirc_bini_dataFit =
     list(biniFit(data = kirc_data_fnl,
             optCuts = .,
             y = Surv(kirc_data_fnl$tte, kirc_data_fnl$vital_status),
             family = "cox",
             col_cuts = "opt_cuts",
             col_x = "opt_cuts_id")$dataFit)) %>%
rowwise %>%
mutate(AIC = round(AIC(kirc_biniFit[[1]]), 0),
       IBS = ibs(pec(list(Cox = kirc_biniFit[[1]]),
                     Hist(time, event) ~ 1,
                     data = kirc_bini_dataFit[[1]],
                     verbose = F))[2, 1],
       `C-index` = paste0(round(concordance(kirc_biniFit[[1]])$concordance, 3),
                          " (", round(sqrt(concordance(kirc_biniFit[[1]])$var), 3),
                          ")")) %>%
ungroup %>%
mutate(Dataset = "KIRC") %>%
relocate(Dataset, .before = "method") %>%
select(! c(kirc_biniFit, kirc_bini_dataFit)) %>%
kable(digits = 3,
      Caption = "Compare AIC of fitted Cox models by using detected optimal cut-points - KIRC dataset") %>%
kable_styling()
```

Dataset	method	n-cuts	AIC	IBS	C-index
KIRC	biniLasso	36	2081	0.119	0.785 (0.016)
KIRC	miniLasso	16	2070	0.155	0.758 (0.017)