```
library(flexsurv)
```

```
## Loading required package:  survival
## Loading required package:  splines
```

```
library(boot)
```

```
##
## Attaching package:  'boot'
##
## The following object is masked from 'package:survival':
##
##    aml
```

```
library(randomForestSRC)
```

```
## Loading required package:  parallel
##
##  randomForestSRC 1.5.5
##
##  Type rfsrc.news() to see new features, changes, and bug fixes.
##
```

```
library(timeROC)
```

```
## Loading required package:  pec
## Loading required package:  mvtnorm
## Loading required package:  timereg
```

```
library(risksetROC)
```

```
## Loading required package:  MASS
```

```
library(RColorBrewer)
pal = brewer.pal(6, "Dark2")
names(pal) = c("gg", "cph", "rsf", "km0", "mskcc.pre", "mskcc.post")
```

1 Preparation

Construct a *preoperative* function based on the Brennan nomogram. The preoperative nature will mean that most prognostic components will need to be marginalized out.

So the preoperative MSKCC score would be:

$$S = 1.4 + 6.1 + 0.8 + 18.2 + 18.9 + 15 + 9 + 15 * Back.pain + 3 * Weight.Loss + -2/15 * Age + 12 + 3\,[Sex = M] + 51\,[Hea \tag{1}$$

```
fit.mskcc = list(
        inputs = list(
        History.Diagnosis.AgeAt = list(
                margins = data.frame(value = 65, fraction = 1),
                scorefunc = function(x) { x = x; -2/15*pmin(pmax(x, 0), 90) + 12 }),
        Patient.Sex = list(
                margins = data.frame(value = c("M", "F"), fraction = c(0.501, 1-0.501)),
                scorefunc = function(x) { 3*I(x == "M") }),
        Portal.Vein = list(
```

Variable	Preoperative?	Available?	Marginals
Age	Yes	Yes	Linear. 90 =>0, 30 =>8. Therefore $f(x) = -2/15(x - 90) = -2/15x +$
Sex	Yes	Yes	Male risk delta 3
Portal Vein	NO		14.4% YES, risk delta 10, marginal 1.4
Splenectomy	NO		9.9% YES, risk delta 62, marginal 6.1
Margin of resection	NO		20.7% POS, risk delta 4, marginal 0.8
Head.vs.Other	Yes	Yes	Head risk delta 51
Differentiation	NO		14.2% Well, risk delta 0, marginal 0
			56.4% Mod, risk delta 14, marginal 7.9
			29.5% Poor, risk delta 35, marginal 10.3. Overall marginal 18.2
Posterior.margin	NO		86.0% POS, risk delta 22, marginal 18.9
Numb.pos.nodes	NO		Mean 2.1, approx marginal 15
Numb.neg.nodes	NO		Mean 16.9, approx marginal 9
Back.pain	Yes	NO	13.7% YES, risk delta 15, marginal 2.0
T.stage	Yes	Yes	
Weight Loss	Yes	NO	53.7% YES, risk delta 3, marginal 1.6
Max.path.axis	Yes	Yes	

```
            margins = data.frame(value = c(TRUE, FALSE), fraction = c(0.144, 1-0.144)),
            scorefunc = function(x) { 10*I(x == TRUE) }),
        Splenectomy = list(
            margins = data.frame(value = c(TRUE, FALSE), fraction = c(0.099, 1-0.099)),
            scorefunc = function(x) { 62*I(x == TRUE) }),
        Treat.MarginPositive = list(
            margins = data.frame(value = c(TRUE, FALSE), fraction = c(0.207, 1-0.207)),
            scorefunc = function(x) { 4*I(x == TRUE) }),
        Path.LocationBody = list(
            margins = data.frame(value = c(FALSE, TRUE), fraction = c(0.894, 1-0.894)),
            scorefunc = function(x) { 51*I(x == TRUE) }),
        Path.Differentiation = list(
            margins = data.frame(value = c("1", "2", "3", "4"), fraction = c(0.142, 0.564, 1-0.142-0
            scorefunc = function(x) { 14*I(x == "2") + 35*I(x == "3") + 35*I(x == "4") }),
        Posterior.Margin = list(
            margins = data.frame(value = c(TRUE, FALSE), fraction = c(0.86, 1-0.86)),
            scorefunc = function(x) { 22*I(x == TRUE) }),
        Path.LN.Involved = list(
            margins = data.frame(value = 2.1, fraction = 1),
            scorefunc = function(x) {
                    x = pmin(40, pmax(x, 0))
                    fitfun = splinefun(c(0, 1, 2, 3, 4, 10, 15, 20, 25, 30, 35, 40), c(0, 14.56, 24.
                    fitfun(x)
            }),
        Path.LN.Negative = list(
            margins = data.frame(value = 16.9, fraction = 1),
            scorefunc = function(x) { (pmin(pmax(x, 0), 90)-90)*-11/90 }),
        Back.pain = list(
            margins = data.frame(value = c(TRUE, FALSE), fraction = c(0.137, 1-0.137)),
            scorefunc = function(x) { 15*I(x == TRUE) }),
        Stage.pT.Simplified = list(
            margins = data.frame(value = c("T1", "T2", "T34"), fraction = c(0.037, 0.119, 1-0.037-0.
            scorefunc = function(x) { 36*I(x == "T1") + 11*I(x == "T34") }),
            # The following matches the original Brennan nomogram, but was not used as there are too
            # tumours in either the NSWPCN *or* the MSKCC cohorts -- how the T4 coefficient was even
```

```
                # I'll never know.  The T34 coefficient of 11 was arrived at as (0.828*10+(1-0.037-0.11
                # being a frequency-weighted average of the T3 and T4 coefficients.
                # margins = data.frame(value = c("T1", "T2", "T3", "T4"), fraction = c(0.037, 0.119, 0.8
                # scorefunc = function(x) { 36*I(x == "T1") + 10*I(x == "T3") + 63*I(x == "T4") }),
        Weight.loss = list(
                margins = data.frame(value = c(TRUE, FALSE), fraction = c(0.537, 1-0.537)),
                scorefunc = function(x) { 3*I(x == TRUE) }),
        Path.Size = list(
                margins = data.frame(),
                scorefunc = function(x) {
                        x = pmin(16, pmax(x, 0))
                        fitfun = splinefun(c(0, 1, 2, 3, 4, 6, 8, 10, 12, 14, 16), c(0, 29.74, 59.48, 86
                        fitfun(x)
                }) ),
        outputs = list(
                DSS12mo = function(s) {
                        x = pmax(50, pmin(350, s))
                        fitfun = splinefun(c(79.0323, 115.02, 165.524, 197.278, 221.774, 242.339, 261.08
                        y = fitfun(x)
                        pmax(0, pmin(1, y))
                },
                DSS24mo = function(s) {
                        x = pmax(50, pmin(350, s))
                        fitfun = splinefun(c(71.1694, 97.7823, 129.536, 153.73, 174.294, 193.347, 211.79
                        y = fitfun(x)
                        pmax(0, pmin(1, y))
                },
                DSS36mo = function(s) {
                        x = pmax(50, pmin(350, s))
                        fitfun = splinefun(c(69.3548, 101.109, 125.302, 145.867, 164.919, 183.367, 202.7
                        y = fitfun(x)
                        pmax(0, pmin(1, y))
                })
        )

applyNomogram = function(nomogram, data)
{
        scores = rowSums(sapply(names(nomogram$inputs), function(input) {
                if (input %in% colnames(data)) {
                        return(nomogram$inputs[[input]]$scorefunc(data[,input]))
                }
                warning(sprintf("Marginalizing missing variable: %s", input))
                margin_score = sum(nomogram$inputs[[input]]$scorefunc(nomogram$inputs[[input]]$margins$v
                return(rep(margin_score, nrow(data)))
        }))

        outputs = sapply(nomogram$outputs, function(f) f(scores))
        cbind(Score = scores, outputs)
}
```

2 Model and data loading

Trained models:

```
temp = readRDS("05_final_model.rds")
fit.gg = temp$gg
fit.cph = temp$cph
fit.km0 = temp$km0
fit.rsf = temp$rsf
data.nswpcn = temp$data.train
```

```
data.glasgow = readRDS("06_Glasgow.rds")
data.glasgow$Path.LN.Negative = data.glasgow$Path.LN.Inspected - data.glasgow$Path.LN.Involved
data.glasgow$History.Diagnosis.AgeAt = data.glasgow$History.Diagnosis.AgeAt.Cent + 68
data.glasgow$Path.Size = data.glasgow$Path.Size.Cent + 30
data.glasgow$SexM = data.glasgow$Patient.Sex == "M"
data.glasgow$AgeCent = data.glasgow$History.Diagnosis.AgeAt.Cent
data.glasgow$SizeCent = data.glasgow$Path.Size.Cent
data.glasgow$A2 = data.glasgow$Molec.S100A2.DCThresh
data.glasgow$A4 = data.glasgow$Molec.S100A4.DCThresh
data.glasgow$LocBody = data.glasgow$Path.Location != "HOP"
data.glasgow$Time = data.glasgow$History.Death.EventTimeDays
data.glasgow$DSD = data.glasgow$History.DSDeath.Event
```

3 Score calculation

```
temp = applyNomogram(fit.mskcc, data.glasgow)

## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Portal.Vein
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Splenectomy
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Posterior.Margin
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Back.pain
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Weight.loss

mskcc_post.linpred.glasgow = temp[,1]
mskcc_post.12mo.glasgow = temp[,2]
mskcc_post.24mo.glasgow = temp[,3]
mskcc_post.36mo.glasgow = temp[,4]
temp = applyNomogram(fit.mskcc, data.glasgow[,c("History.Diagnosis.AgeAt", "Patient.Sex", "Path.Location

## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Portal.Vein
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Splenectomy
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Treat.MarginPositive
```

```
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Path.Differentiation
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Posterior.Margin
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Path.LN.Involved
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Path.LN.Negative
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Back.pain
## Warning in FUN(c("History.Diagnosis.AgeAt", "Patient.Sex", "Portal.Vein", :  Marginalizing
missing variable:  Weight.loss
```

```
mskcc_pre.linpred.glasgow = temp[,1]
mskcc_pre.12mo.glasgow = temp[,2]
mskcc_pre.24mo.glasgow = temp[,3]
mskcc_pre.36mo.glasgow = temp[,4]
```

Get approximate linear predictors from the GG model, by just calculating the location term effect.

```
val.prob.times = seq(0, max(data.glasgow$Time), 1)

gg.path.glasgow = summary(fit.gg, newdata = data.glasgow, ci = FALSE)
temp.coefs = coef(fit.gg)
gg.linpred.glasgow = sapply(1:length(temp.coefs), function(coef_i) {
        if (names(temp.coefs)[coef_i] %in% colnames(data.glasgow)) {
                temp.coefs[coef_i] * data.glasgow[,names(temp.coefs)[coef_i]]
        } else if (gsub("TRUE$", "", names(temp.coefs)[coef_i]) %in% colnames(data.glasgow)) {
                temp.coefs[coef_i] * data.glasgow[,gsub("TRUE$", "", names(temp.coefs)[coef_i])]
        } else {
                rep(0, nrow(data.glasgow))
        } })
gg.linpred.glasgow = -rowSums(gg.linpred.glasgow)       # Negate to bring into concordance with the dire
temp = summary(fit.gg, newdata = data.glasgow, ci = FALSE)
gg.prob.glasgow = sapply(temp, function(x) approx(x[,1], x[,2], xout = val.prob.times, yleft = 1, yright
colnames(gg.prob.glasgow) = rownames(data.glasgow)

gg.linpred.nswpcn = sapply(1:length(temp.coefs), function(coef_i) {
        if (names(temp.coefs)[coef_i] %in% colnames(data.nswpcn)) {
                temp.coefs[coef_i] * data.nswpcn[,names(temp.coefs)[coef_i]]
        } else if (gsub("TRUE$", "", names(temp.coefs)[coef_i]) %in% colnames(data.nswpcn)) {
                temp.coefs[coef_i] * data.nswpcn[,gsub("TRUE$", "", names(temp.coefs)[coef_i])]
        } else {
                rep(0, nrow(data.nswpcn))
        } })
gg.linpred.nswpcn = -rowSums(gg.linpred.nswpcn)         # Negate to bring into concordance with the dire
temp = summary(fit.gg, newdata = data.nswpcn, ci = FALSE)
gg.prob.nswpcn = sapply(temp, function(x) approx(x[,1], x[,2], xout = val.prob.times, yleft = 1, yright
colnames(gg.prob.nswpcn) = rownames(data.nswpcn)
```

```
temp.coefs = coef(fit.cph)
cph.linpred.glasgow = sapply(1:length(temp.coefs), function(coef_i) {
        if (names(temp.coefs)[coef_i] %in% colnames(data.glasgow)) {
```

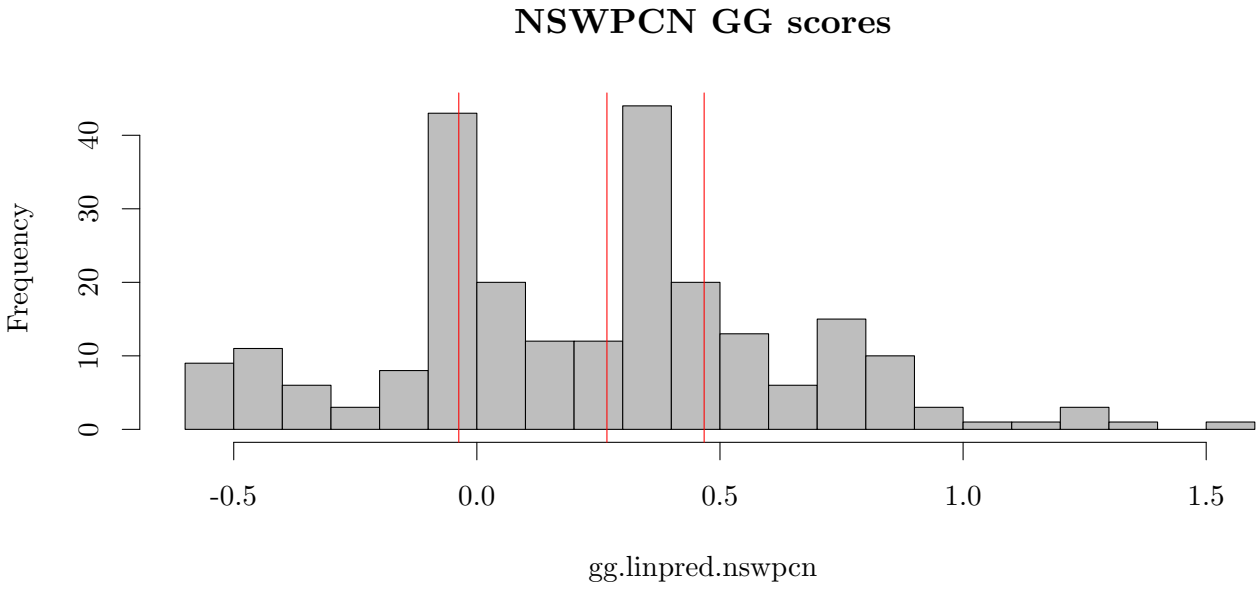
```
                temp.coefs[coef_i] * data.glasgow[,names(temp.coefs)[coef_i]]
        } else if (gsub("TRUE$", "", names(temp.coefs)[coef_i]) %in% colnames(data.glasgow)) {
                temp.coefs[coef_i] * data.glasgow[,gsub("TRUE$", "", names(temp.coefs)[coef_i])]
        } else {
                rep(0, nrow(data.glasgow))
        } })
cph.linpred.glasgow = rowSums(cph.linpred.glasgow)
temp = survfit(fit.cph, newdata = data.glasgow)
cph.prob.glasgow = simplify2array(tapply(1:length(temp$surv), rep(names(temp$strata), temp$strata), func

cph.linpred.nswpcn = sapply(1:length(temp.coefs), function(coef_i) {
        if (names(temp.coefs)[coef_i] %in% colnames(data.nswpcn)) {
                temp.coefs[coef_i] * data.nswpcn[,names(temp.coefs)[coef_i]]
        } else if (gsub("TRUE$", "", names(temp.coefs)[coef_i]) %in% colnames(data.nswpcn)) {
                temp.coefs[coef_i] * data.nswpcn[,gsub("TRUE$", "", names(temp.coefs)[coef_i])]
        } else {
                rep(0, nrow(data.nswpcn))
        } })
cph.linpred.nswpcn = rowSums(cph.linpred.nswpcn)
temp = survfit(fit.cph, newdata = data.nswpcn)
cph.prob.nswpcn = simplify2array(tapply(1:length(temp$surv), rep(names(temp$strata), temp$strata), funct

# Doesn't work for some obscure reason, I suspect to do with strata and environments:
# cph.linpred.glasgow = predict(fit.cph, newdata = data.glasgow)
# cph.linpred.nswpcn = predict(fit.cph, newdata = data.nswpcn)
```

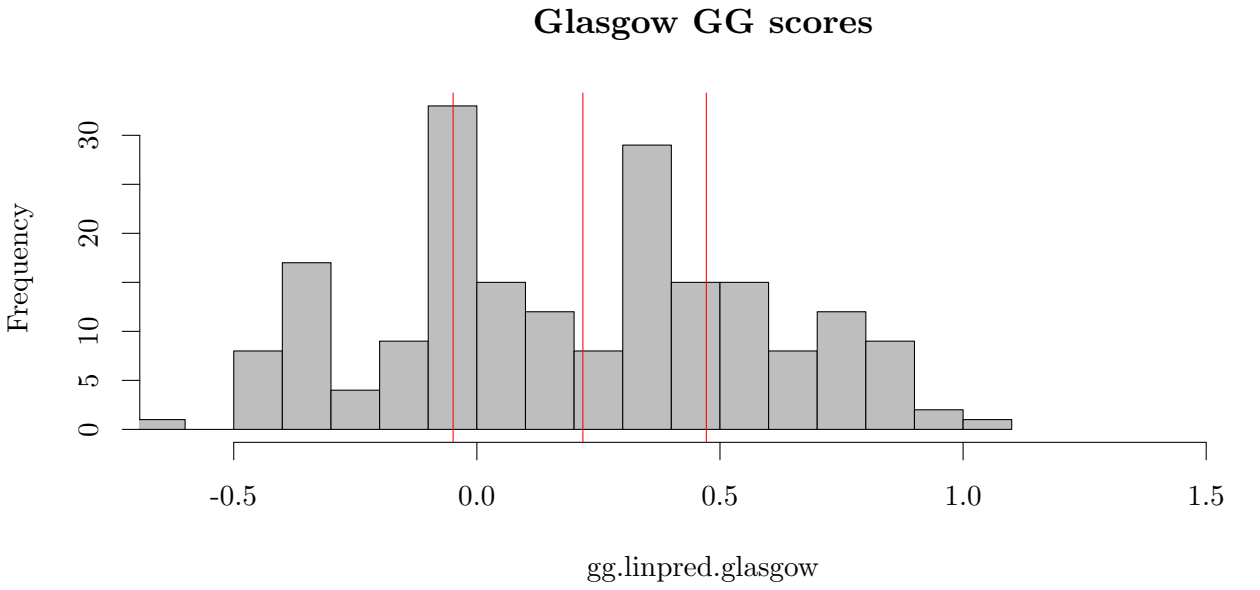
```
temp = predict(fit.rsf, newdata = data.glasgow)
rsf.linpred.glasgow = apply(temp$survival, 1, function(s1) {
    sfunc = approxfun(temp$time.interest, s1, yleft = 1, yright = 0, rule = 2)
    med = uniroot(function(x) sfunc(x) - 0.5, lower = min(temp$time.interest), upper = max(temp$time.int
    med
})
rsf.linpred.glasgow = -rsf.linpred.glasgow
rsf.prob.glasgow = apply(temp$survival, 1, function(s1) approx(temp$time.interest, s1, xout = val.prob.t
colnames(rsf.prob.glasgow) = rownames(data.glasgow)

temp = predict(fit.rsf, newdata = data.nswpcn)
rsf.linpred.nswpcn = apply(temp$survival, 1, function(s1) {
    sfunc = approxfun(temp$time.interest, s1, yleft = 1, yright = 0, rule = 2)
    med = uniroot(function(x) sfunc(x) - 0.5, lower = min(temp$time.interest), upper = max(temp$time.int
    med
})
rsf.linpred.nswpcn = -rsf.linpred.nswpcn
rsf.prob.nswpcn = apply(temp$survival, 1, function(s1) approx(temp$time.interest, s1, xout = val.prob.ti
colnames(rsf.prob.nswpcn) = rownames(data.nswpcn)
```

4 Validation

4.1 Altman diagnostic 1: score histograms

```r
par(mfrow = c(2, 1))
hist(gg.linpred.nswpcn, main = "NSWPCN GG scores", xlim = range(c(gg.linpred.nswpcn, gg.linpred.glasgow)
abline(v = quantile(gg.linpred.nswpcn, probs = c(0.25, 0.5, 0.75)), col = "red")
hist(gg.linpred.glasgow, main = "Glasgow GG scores", xlim = range(c(gg.linpred.nswpcn, gg.linpred.glasgo
abline(v = quantile(gg.linpred.glasgow, probs = c(0.25, 0.5, 0.75)), col = "red")
```






```r
par(mfrow = c(1, 1))

# par(mfrow = c(2, 1))
# hist(cph.linpred.nswpcn, main = "NSWPCN CPH scores", xlim = range(c(cph.linpred.nswpcn, cph.linpred.gl
# abline(v = quantile(gg.linpred.nswpcn, probs = c(0.25, 0.5, 0.75)), col = "red")
# hist(cph.linpred.glasgow, main = "Glasgow CPH scores", xlim = range(c(cph.linpred.nswpcn, cph.linpred
# abline(v = quantile(gg.linpred.glasgow, probs = c(0.25, 0.5, 0.75)), col = "red")
# par(mfrow = c(1, 1))
```

```
# par(mfrow = c(2, 1))
# hist(rsf.linpred.nswpcn, main = "NSWPCN RSF scores", xlim = range(c(rsf.linpred.nswpcn, rsf.linpred.gl
# abline(v = quantile(rsf.linpred.nswpcn, probs = c(0.25, 0.5, 0.75)), col = "red")
# hist(rsf.linpred.glasgow, main = "Glasgow RSF scores", xlim = range(c(rsf.linpred.nswpcn, rsf.linpred
# abline(v = quantile(rsf.linpred.glasgow, probs = c(0.25, 0.5, 0.75)), col = "red")
# par(mfrow = c(1, 1))
```

4.2 Altman method 1 (D,F)

```
summary(coxph(Surv(Time, DSD) ~ mskcc_post.linpred.glasgow, data.glasgow))

## Call:
## coxph(formula = Surv(Time, DSD) ~ mskcc_post.linpred.glasgow,
##     data = data.glasgow)
##
##   n= 198, number of events= 170
##
##                               coef exp(coef) se(coef)    z Pr(>|z|)
## mskcc_post.linpred.glasgow 0.01484   1.01495  0.00405 3.67  0.00025
##
##                            exp(coef) exp(-coef) lower .95 upper .95
## mskcc_post.linpred.glasgow      1.01      0.985      1.01      1.02
##
## Concordance= 0.576  (se = 0.025 )
## Rsquare= 0.067   (max possible= 0.999 )
## Likelihood ratio test= 13.6  on 1 df,   p=0.000221
## Wald test            = 13.4  on 1 df,   p=0.000245
## Score (logrank) test = 13.6  on 1 df,   p=0.000229

summary(coxph(Surv(Time, DSD) ~ mskcc_pre.linpred.glasgow, data.glasgow))

## Call:
## coxph(formula = Surv(Time, DSD) ~ mskcc_pre.linpred.glasgow,
##     data = data.glasgow)
##
##   n= 198, number of events= 170
##
##                                coef exp(coef)  se(coef)     z Pr(>|z|)
## mskcc_pre.linpred.glasgow -0.000423  0.999577  0.007318 -0.06     0.95
##
##                           exp(coef) exp(-coef) lower .95 upper .95
## mskcc_pre.linpred.glasgow         1          1     0.985      1.01
##
## Concordance= 0.421  (se = 0.025 )
## Rsquare= 0   (max possible= 0.999 )
## Likelihood ratio test= 0  on 1 df,   p=0.954
## Wald test            = 0  on 1 df,   p=0.954
## Score (logrank) test = 0  on 1 df,   p=0.954

summary(coxph(Surv(Time, DSD) ~ gg.linpred.glasgow, data.glasgow))

## Call:
```

```
## coxph(formula = Surv(Time, DSD) ~ gg.linpred.glasgow, data = data.glasgow)
##
##   n= 198, number of events= 170
##
##                      coef exp(coef) se(coef)    z Pr(>|z|)
## gg.linpred.glasgow 0.746     2.109    0.221 3.38  0.00073
##
##                    exp(coef) exp(-coef) lower .95 upper .95
## gg.linpred.glasgow      2.11      0.474      1.37      3.25
##
## Concordance= 0.607  (se = 0.025 )
## Rsquare= 0.056   (max possible= 0.999 )
## Likelihood ratio test= 11.5  on 1 df,   p=0.000707
## Wald test            = 11.4  on 1 df,   p=0.000732
## Score (logrank) test = 11.5  on 1 df,   p=0.000693

# summary(coxph(Surv(Time, DSD) ~ cph.linpred.glasgow, data.glasgow))
# summary(coxph(Surv(Time, DSD) ~ rsf.linpred.glasgow, data.glasgow))

anova(coxph(Surv(Time, DSD) ~ offset(gg.linpred.glasgow) + gg.linpred.glasgow, data.glasgow))

## Analysis of Deviance Table
##  Cox model: response is Surv(Time, DSD)
## Terms added sequentially (first to last)
##
##                    loglik Chisq Df Pr(>|Chi|)
## NULL                 -724
## gg.linpred.glasgow   -723  1.32  1       0.25

# anova(coxph(Surv(Time, DSD) ~ offset(cph.linpred.glasgow) + cph.linpred.glasgow, data.glasgow))
# anova(coxph(Surv(Time, DSD) ~ offset(rsf.linpred.glasgow) + rsf.linpred.glasgow, data.glasgow))
```

Booyah.

4.3 Altman method 2 (F)

```
summary(coxph(Surv(Time, DSD) ~ offset(mskcc_pre.linpred.glasgow) + AgeCent + SexM + SizeCent + A2 + A4,

## Warning in fitter(X, Y, strats, offset, init, control, weights = weights, :  Ran out of
iterations and did not converge
## Error in fitter(X, Y, strats, offset, init, control, weights = weights, :  NA/NaN/Inf in
foreign function call (arg 6)

summary(coxph(Surv(Time, DSD) ~ offset(mskcc_post.linpred.glasgow) + AgeCent + SexM + SizeCent + A2 + A4

## Call:
## coxph(formula = Surv(Time, DSD) ~ offset(mskcc_post.linpred.glasgow) +
##     AgeCent + SexM + SizeCent + A2 + A4, data = data.glasgow)
##
##   n= 198, number of events= 170
##
##                coef exp(coef)  se(coef)      z Pr(>|z|)
## AgeCent    0.22831   1.25648   0.01006  22.69  < 2e-16
## SexMTRUE  -5.22725   0.00537   0.30189 -17.32  < 2e-16
```

```
## SizeCent   0.14973   1.16152   0.01910   7.84  4.6e-15
## A2TRUE    -2.29883   0.10038   0.37880  -6.07  1.3e-09
## A4TRUE     4.93307 138.80556   0.29941  16.48  < 2e-16
##
##          exp(coef) exp(-coef) lower .95 upper .95
## AgeCent   1.26e+00     0.7959   1.23194    1.2815
## SexMTRUE  5.37e-03   186.2805   0.00297    0.0097
## SizeCent  1.16e+00     0.8609   1.11884    1.2058
## A2TRUE    1.00e-01     9.9625   0.04777    0.2109
## A4TRUE    1.39e+02     0.0072  77.18720  249.6137
##
## Concordance= 0.587  (se = 0.025 )
## Rsquare= 1   (max possible= 1 )
## Likelihood ratio test= 1719  on 5 df,   p=0
## Wald test            = 2210  on 5 df,   p=0
## Score (logrank) test = 12193  on 5 df,   p=0

summary(coxph(Surv(Time, DSD) ~ offset(gg.linpred.glasgow) + AgeCent + SexM + SizeCent + A2 + A4, data.g

## Call:
## coxph(formula = Surv(Time, DSD) ~ offset(gg.linpred.glasgow) +
##     AgeCent + SexM + SizeCent + A2 + A4, data = data.glasgow)
##
##   n= 198, number of events= 170
##
##              coef exp(coef) se(coef)     z Pr(>|z|)
## AgeCent  -0.03255   0.96797  0.00860 -3.78  0.00015
## SexMTRUE  0.66683   1.94805  0.16160  4.13  3.7e-05
## SizeCent  0.02516   1.02547  0.00737  3.41  0.00065
## A2TRUE    0.34535   1.41249  0.17387  1.99  0.04701
## A4TRUE   -0.07127   0.93121  0.17723 -0.40  0.68757
##
##          exp(coef) exp(-coef) lower .95 upper .95
## AgeCent      0.968      1.033     0.952     0.984
## SexMTRUE     1.948      0.513     1.419     2.674
## SizeCent     1.025      0.975     1.011     1.040
## A2TRUE       1.412      0.708     1.005     1.986
## A4TRUE       0.931      1.074     0.658     1.318
##
## Concordance= 0.681  (se = 0.025 )
## Rsquare= 0.205   (max possible= 0.999 )
## Likelihood ratio test= 45.5  on 5 df,   p=1.12e-08
## Wald test            = 46.3  on 5 df,   p=7.74e-09
## Score (logrank) test = 48.4  on 5 df,   p=2.95e-09

# summary(coxph(Surv(Time, DSD) ~ offset(cph.linpred.glasgow) + AgeCent + SexM + SizeCent + A2 + A4, dat
# summary(coxph(Surv(Time, DSD) ~ offset(rsf.linpred.glasgow) + AgeCent + SexM + SizeCent + A2 + A4, dat
```

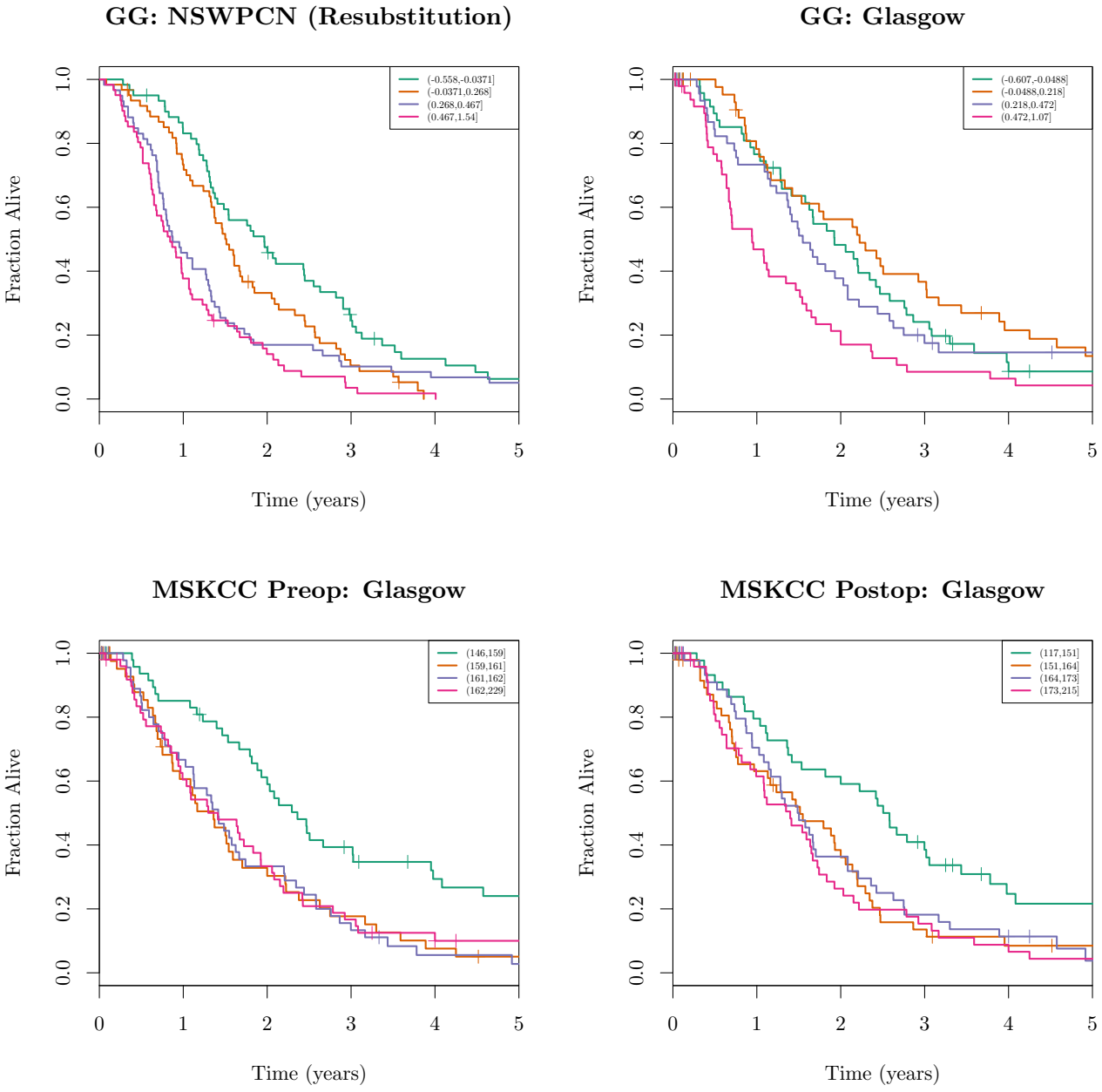
Still strong evidence of misspecification or poor fit. However, the above calibration slope was not significantly different from 1. Hmm. This doesn't necessarily sink the method, but will need checking as we go along.

4.4 Altman method 3 (D)

Look at the CIs above.

4.5 Altman method 4 (D,C)

```
group_quantiles = c(0, 0.25, 0.5, 0.75, 1)
mskcc_pre.groups.glasgow = cut(mskcc_pre.linpred.glasgow, quantile(mskcc_pre.linpred.glasgow, group_quar
mskcc_post.groups.glasgow = cut(mskcc_post.linpred.glasgow, quantile(mskcc_post.linpred.glasgow, group_c
gg.groups.glasgow = cut(gg.linpred.glasgow, quantile(gg.linpred.glasgow, group_quantiles))
gg.groups.nswpcn = cut(gg.linpred.nswpcn, quantile(gg.linpred.nswpcn, group_quantiles))
cph.groups.glasgow = cut(cph.linpred.glasgow, quantile(cph.linpred.glasgow, group_quantiles))
cph.groups.nswpcn = cut(cph.linpred.nswpcn, quantile(cph.linpred.nswpcn, group_quantiles))
rsf.groups.glasgow = cut(rsf.linpred.glasgow, quantile(rsf.linpred.glasgow, group_quantiles))
rsf.groups.nswpcn = cut(rsf.linpred.nswpcn, quantile(rsf.linpred.nswpcn, group_quantiles))

par(mfrow = c(2, 2))
temp = survfit(Surv(data.nswpcn$Time/365.25, data.nswpcn$DSD) ~ gg.groups.nswpcn)
plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", mai
legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strata)

temp = survfit(Surv(data.glasgow$Time/365.25, data.glasgow$DSD) ~ gg.groups.glasgow)
plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", mai
legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strata)

# temp = survfit(Surv(data.nswpcn$Time/365.25, data.nswpcn$DSD) ~ cph.groups.nswpcn)
# plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", m
# legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strat

# temp = survfit(Surv(data.glasgow$Time/365.25, data.glasgow$DSD) ~ cph.groups.glasgow)
# plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", m
# legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strat

# temp = survfit(Surv(data.nswpcn$Time/365.25, data.nswpcn$DSD) ~ rsf.groups.nswpcn)
# plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", m
# legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strat

# temp = survfit(Surv(data.glasgow$Time/365.25, data.glasgow$DSD) ~ rsf.groups.glasgow)
# plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", m
# legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strat

temp = survfit(Surv(data.glasgow$Time/365.25, data.glasgow$DSD) ~ mskcc_pre.groups.glasgow)
plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", mai
legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strata)

temp = survfit(Surv(data.glasgow$Time/365.25, data.glasgow$DSD) ~ mskcc_post.groups.glasgow)
plot(temp, col = pal[1:(length(group_quantiles)-1)], xlab = "Time (years)", ylab = "Fraction Alive", mai
legend("topright", col = pal[1:(length(group_quantiles)-1)], legend = gsub(".*=", "", names(temp$strata)
```




```
par(mfrow = c(1, 1))
```

Weird. MSKCC somehow is still finding a subgroup, and it's somehow even clearer in preop! This is based on an approximation to GG only, but should be pretty close. It certainly does OK on resubstituted data, but not so well on the Glasgow patients.

Decision curve analysis.

```
source("stdca.R")
temp.data = data.frame(Time = data.glasgow$Time, DSD = data.glasgow$DSD*1,
    gg.1 = 1-gg.prob.glasgow[val.prob.times == 365,], gg.2 = 1-gg.prob.glasgow[val.prob.times == 365*2,]
    cph.1 = 1-cph.prob.glasgow[val.prob.times == 365,], cph.2 = 1-cph.prob.glasgow[val.prob.times == 365
    rsf.1 = 1-rsf.prob.glasgow[val.prob.times == 365,], rsf.2 = 1-rsf.prob.glasgow[val.prob.times == 365
    mskcc.pre.1 = 1-mskcc_pre.12mo.glasgow, mskcc.pre.2 = 1-mskcc_pre.24mo.glasgow, mskcc.pre.3 = 1-msk
    mskcc.post.1 = 1-mskcc_post.12mo.glasgow, mskcc.post.2 = 1-mskcc_post.24mo.glasgow, mskcc.post.3 = 1
```

```
# invisible(stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", predictors = c("gg.1", "cph.1",
# invisible(stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", predictors = c("gg.2", "cph.2",
# invisible(stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", predictors = c("gg.3", "cph.3",
invisible(stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", predictors = c("gg.1", "mskcc.pre

## Error in stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", :  Number of probabilities
specified must be the same as the number of predictors being checked.

invisible(stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", predictors = c("gg.2", "mskcc.pre

## Error in stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", :  Number of probabilities
specified must be the same as the number of predictors being checked.

invisible(stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", predictors = c("gg.3", "mskcc.pre

## Error in stdca(data = temp.data, outcome = "DSD", ttoutcome = "Time", :  Number of probabilities
specified must be the same as the number of predictors being checked.
```

4.6 Brier score

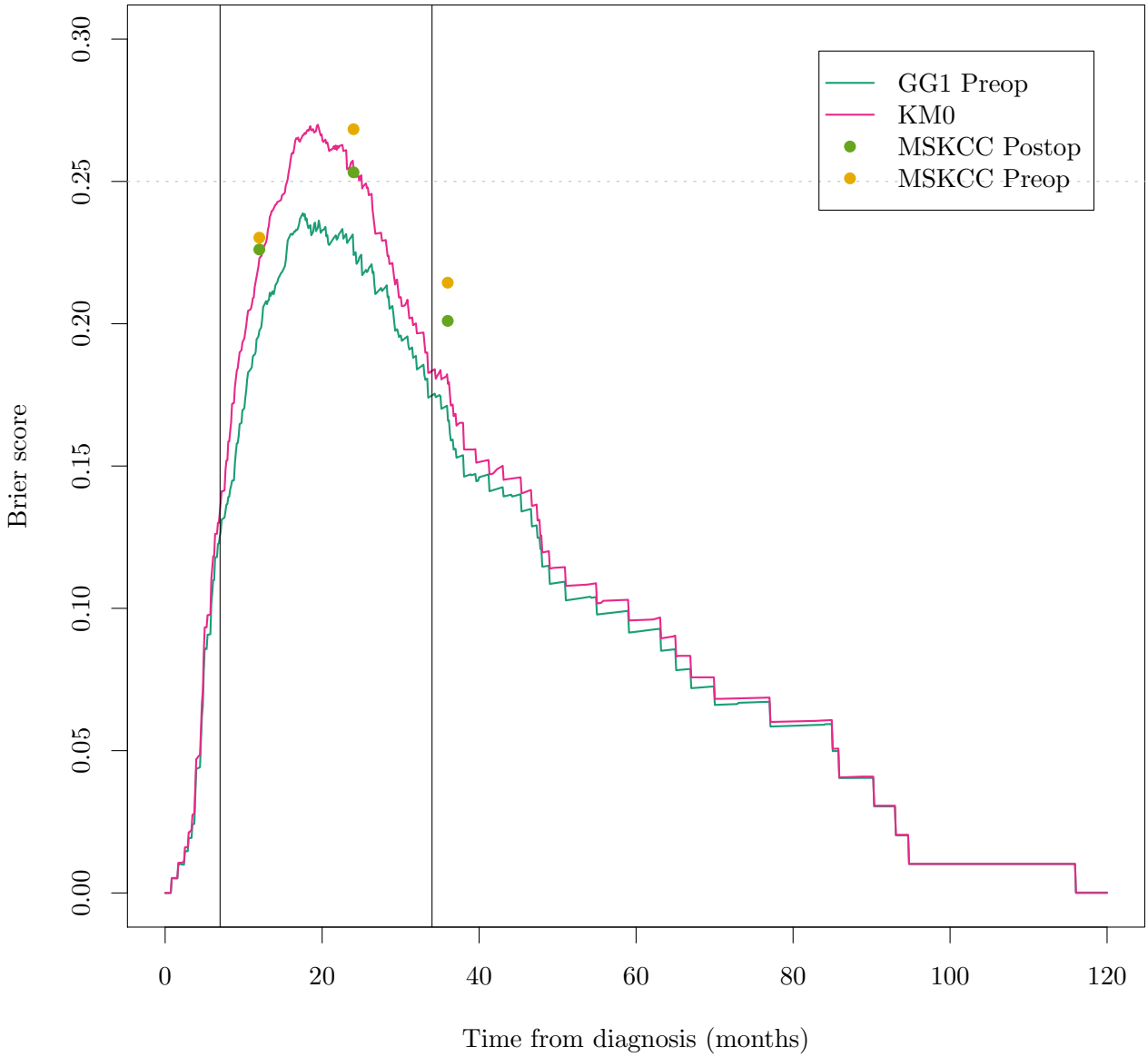
```
calcIBS = function(surv, pred, pred_times, max_time, min_time = 0)
{
        stopifnot(nrow(surv) == nrow(pred) && length(pred_times) == ncol(pred))

        n = nrow(surv)
        marg_survfit = survfit(surv ~ 1)
        marg_censfit = survfit(Surv(surv[,1], !surv[,2]) ~ 1)
        marg_surv_func = approxfun(marg_survfit$time, marg_survfit$surv, method = "constant", yleft = 1,
        marg_cens_func = approxfun(marg_censfit$time, marg_censfit$surv, method = "constant", yleft = 1,

        pred_funcs = apply(pred, 1, function(pat_preds) approxfun(pred_times, pat_preds, yleft = 1, yrig

        indiv_patient_bsc = function(pat_i, tstars)
        {
                observed_time = surv[pat_i, 1]
                observed_event = surv[pat_i, 2]
                pred_func = pred_funcs[[pat_i]]
                category = 1*(observed_time <= tstars & observed_event) + 2*(observed_time > tstars) + 3
                bsc = rep(NA, length(tstars))
                bsc[category == 1] = pred_func(tstars[category == 1])^2 / marg_cens_func(observed_time)
                bsc[category == 2] = (1 - pred_func(tstars[category == 2]))^2 / marg_cens_func(tstars[ca
                bsc[category == 3] = 0
                bsc
        }

        bsc_func = function(tstars) { rowMeans(sapply(1:n, function(pat_i) indiv_patient_bsc(pat_i, tsta

        weight_func = function(tstars) { (1 - marg_surv_func(tstars)) / (1 - marg_surv_func(max_time)) }

        # Be slack and do trapezoidal int. with a fine grid.  It should be possible
        # to calulate the int. exactly but I cbfed.
        int_grid = seq(min_time, max_time, length.out = 1e3)
        bsc_vals = bsc_func(int_grid)
```

```
        weight_vals = weight_func(int_grid)
        int_vals = bsc_vals * weight_vals
        ibsc = (2*sum(int_vals) - int_vals[1] - int_vals[length(int_vals)]) * (diff(range(int_grid))) /

        return(list(bsc = bsc_vals, weights = weight_vals, eval_times = int_grid, ibsc = ibsc))
}

calcBSsingle = function(surv, pred, pred_time)
{
        n = nrow(surv)
        obs_time = surv[,1]
        obs_event = surv[,2]
        marg_censfit = survfit(Surv(obs_time, !obs_event) ~ 1)
        marg_cens_func = approxfun(marg_censfit$time, marg_censfit$surv, method = "constant", yleft = 1,

        brier_val = rep(NA, n)
        cat = 1*I(obs_time <= pred_time & obs_event) + 2*I(obs_time > pred_time) + 3*I(obs_time <= pred_
        brier_val[cat == 1] = (pred[cat == 1])^2 / marg_cens_func(obs_time[cat == 1])
        brier_val[cat == 2] = (1-pred[cat == 2])^2 / marg_cens_func(pred_time)
        brier_val[cat == 3] = 0

        mean(brier_val)
}
```

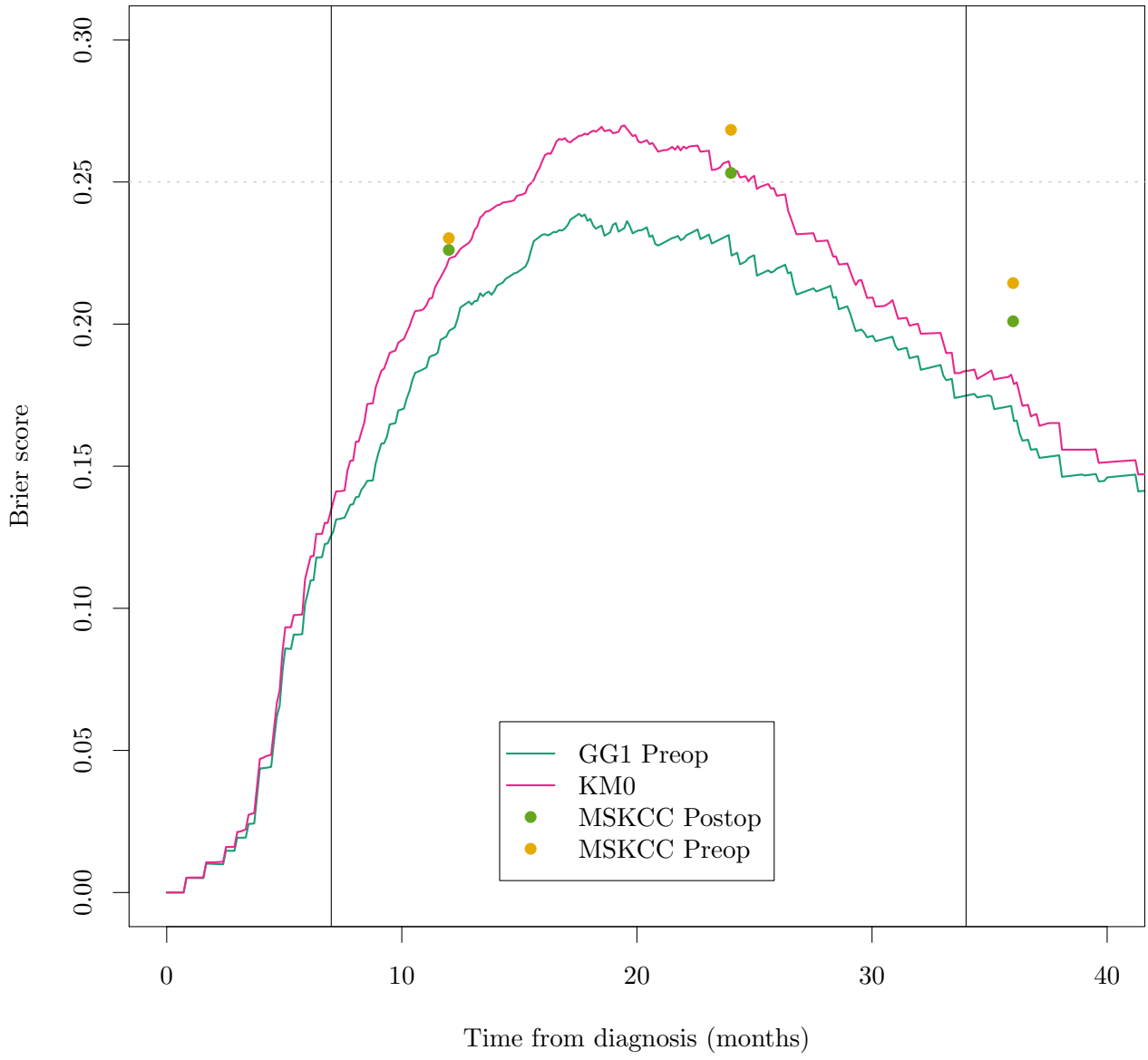
```
mskcc_post.12mo.glasgow.brier = calcBSsingle(Surv(data.glasgow$Time, data.glasgow$DSD), mskcc_post.12mo.
mskcc_post.24mo.glasgow.brier = calcBSsingle(Surv(data.glasgow$Time, data.glasgow$DSD), mskcc_post.24mo.
mskcc_post.36mo.glasgow.brier = calcBSsingle(Surv(data.glasgow$Time, data.glasgow$DSD), mskcc_post.36mo.
mskcc_pre.12mo.glasgow.brier = calcBSsingle(Surv(data.glasgow$Time, data.glasgow$DSD), mskcc_pre.12mo.gl
mskcc_pre.24mo.glasgow.brier = calcBSsingle(Surv(data.glasgow$Time, data.glasgow$DSD), mskcc_pre.24mo.gl
mskcc_pre.36mo.glasgow.brier = calcBSsingle(Surv(data.glasgow$Time, data.glasgow$DSD), mskcc_pre.36mo.gl
gg.path.glasgow.brier = calcIBS(Surv(data.glasgow$Time, data.glasgow$DSD), t(sapply(gg.path.glasgow, fun

km0.path.glasgow.brier = calcIBS(Surv(data.glasgow$Time, data.glasgow$DSD), matrix(fit.km0$surv, nrow =

temp.cph.pred = survfit(fit.cph, newdata = data.glasgow)
temp.cph.pred.expanded_strata = rep(names(temp.cph.pred$strata), temp.cph.pred$strata)
temp.cph.pred_funcs = sapply(rownames(data.glasgow), function(pat_id) {
        approxfun(temp.cph.pred$time[temp.cph.pred.expanded_strata == pat_id], temp.cph.pred$surv[temp.c
})
temp.brier.times = unique(sort(c(seq(0, 10*365.25, 1), c(12, 24, 36)/12*365.25)))
cph.path.glasgow.brier = calcIBS(Surv(data.glasgow$Time, data.glasgow$DSD),
        t(sapply(temp.cph.pred_funcs[rownames(data.glasgow)], function(f) f(temp.brier.times))), temp.br

temp.rsf.pred = predict(fit.rsf, newdata = data.glasgow)
rsf.path.glasgow.brier = calcIBS(Surv(data.glasgow$Time, data.glasgow$DSD), t(apply(temp.rsf.pred$surviv
```

```
plot(gg.path.glasgow.brier$eval_times/365.25*12, gg.path.glasgow.brier$bsc, col = pal["gg"], type = "l",
lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc, col = pal["km0"], lwd = 2)
# lines(gg.path.glasgow.brier$eval_times/365.25*12, cph.path.glasgow.brier$bsc, col = pal["cph"], lwd =
# lines(gg.path.glasgow.brier$eval_times/365.25*12, rsf.path.glasgow.brier$bsc, col = pal["rsf"], lwd =
points(c(12, 24, 36), c(mskcc_post.12mo.glasgow.brier, mskcc_post.24mo.glasgow.brier, mskcc_post.36mo.gl
points(c(12, 24, 36), c(mskcc_pre.12mo.glasgow.brier, mskcc_pre.24mo.glasgow.brier, mskcc_pre.36mo.glasg
```

```r
abline(h = 0.25, col = "grey", lty = "dotted")
abline(v = c(7, 34))
# legend("topright",
#  legend = c( "GG1 Preop",  "CP1 Preop",  "RSF Preop",  "KM0",  "MSKCC Postop",  "MSKCC Preop"),
#  pch = c( NA,  NA,  NA,  NA,  16,  16),
#  col = c( pal["gg"],  pal["cph"],  pal["rsf"],  pal["km0"], pal["mskcc.pre"],  pal["mskcc.post"]),
#  lty = c( "solid",  "solid",  "solid",  "solid",  NA,  NA),
#  inset = 0.05, lwd = 2)
legend("topright",
       legend = c(     "GG1 Preop",    "KM0",          "MSKCC Postop",         "MSKCC Preop"),
       pch = c(        NA,                     NA,             16,                             16),
       col = c(        pal["gg"],              pal["km0"], pal["mskcc.pre"],   pal["mskcc.post"]),
       lty = c(        "solid",                "solid",        NA,                             NA),
       inset = 0.05, lwd = 2)
```

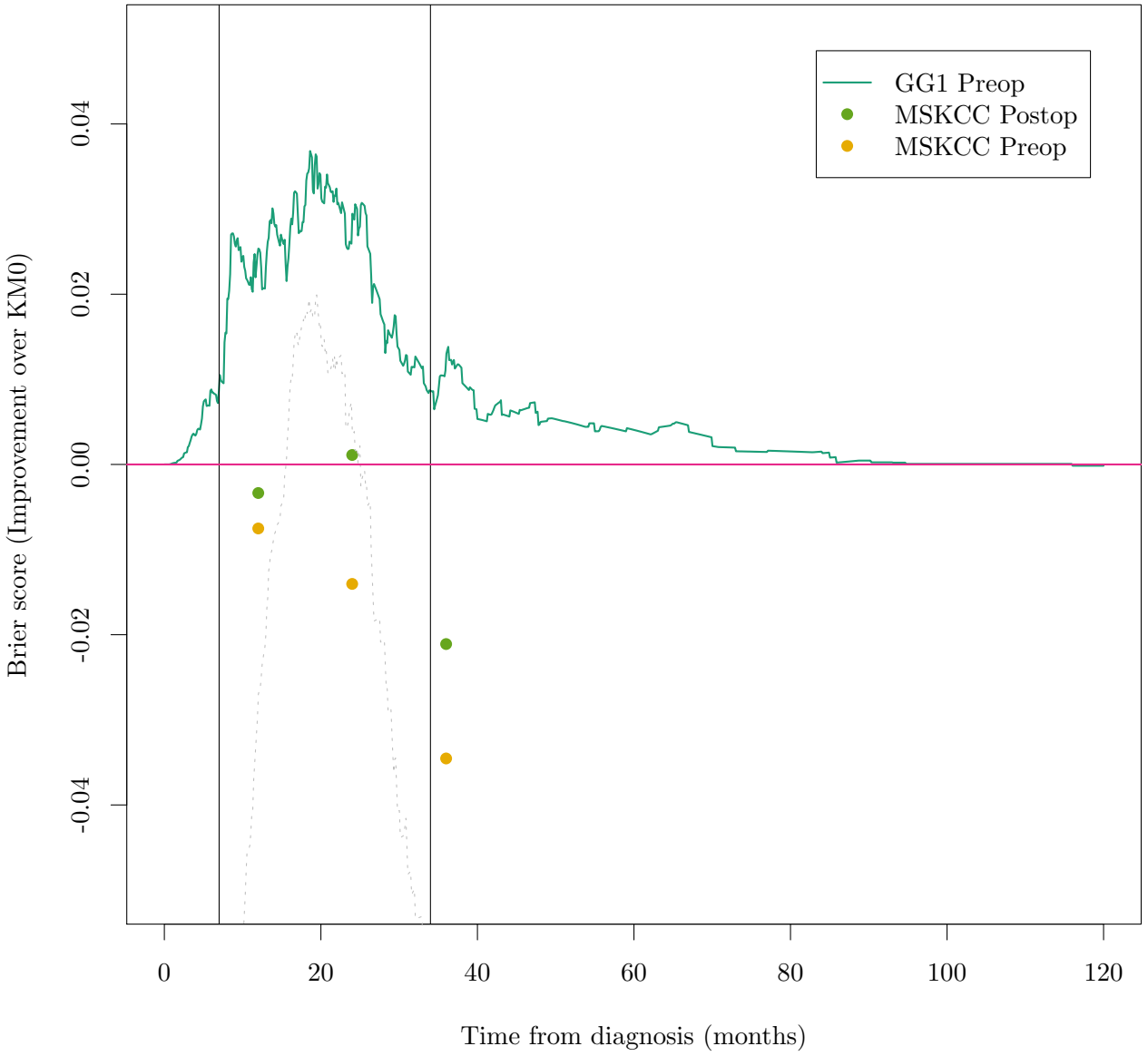


```r
plot(gg.path.glasgow.brier$eval_times/365.25*12, gg.path.glasgow.brier$bsc, col = pal["gg"], type = "l",
lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc, col = pal["km0"], lwd = 2)
# lines(gg.path.glasgow.brier$eval_times/365.25*12, cph.path.glasgow.brier$bsc, col = pal["cph"], lwd =
# lines(gg.path.glasgow.brier$eval_times/365.25*12, rsf.path.glasgow.brier$bsc, col = pal["rsf"], lwd =
points(c(12, 24, 36), c(mskcc_post.12mo.glasgow.brier, mskcc_post.24mo.glasgow.brier, mskcc_post.36mo.gl
points(c(12, 24, 36), c(mskcc_pre.12mo.glasgow.brier, mskcc_pre.24mo.glasgow.brier, mskcc_pre.36mo.glasg
abline(h = 0.25, col = "grey", lty = "dotted")
abline(v = c(7, 34))
# legend("bottom",
#  legend = c( "GG1 Preop",  "CP1 Preop",  "RSF Preop",  "KM0",  "MSKCC Postop",  "MSKCC Preop"),
#  pch = c( NA,  NA,  NA,  NA,  16,  16),
#  col = c( pal["gg"],  pal["cph"],  pal["rsf"],  pal["km0"], pal["mskcc.pre"],  pal["mskcc.post"]),
#  lty = c( "solid",  "solid",  "solid",  "solid",  NA,  NA),
#  inset = 0.05, lwd = 2)
legend("bottom",
       legend = c(     "GG1 Preop",    "KM0",          "MSKCC Postop",         "MSKCC Preop"),
       pch = c(        NA,                     NA,             16,                             16),
       col = c(        pal["gg"],              pal["km0"], pal["mskcc.pre"],   pal["mskcc.post"]),
       lty = c(        "solid",                "solid",        NA,                             NA),
       inset = 0.05, lwd = 2)
```



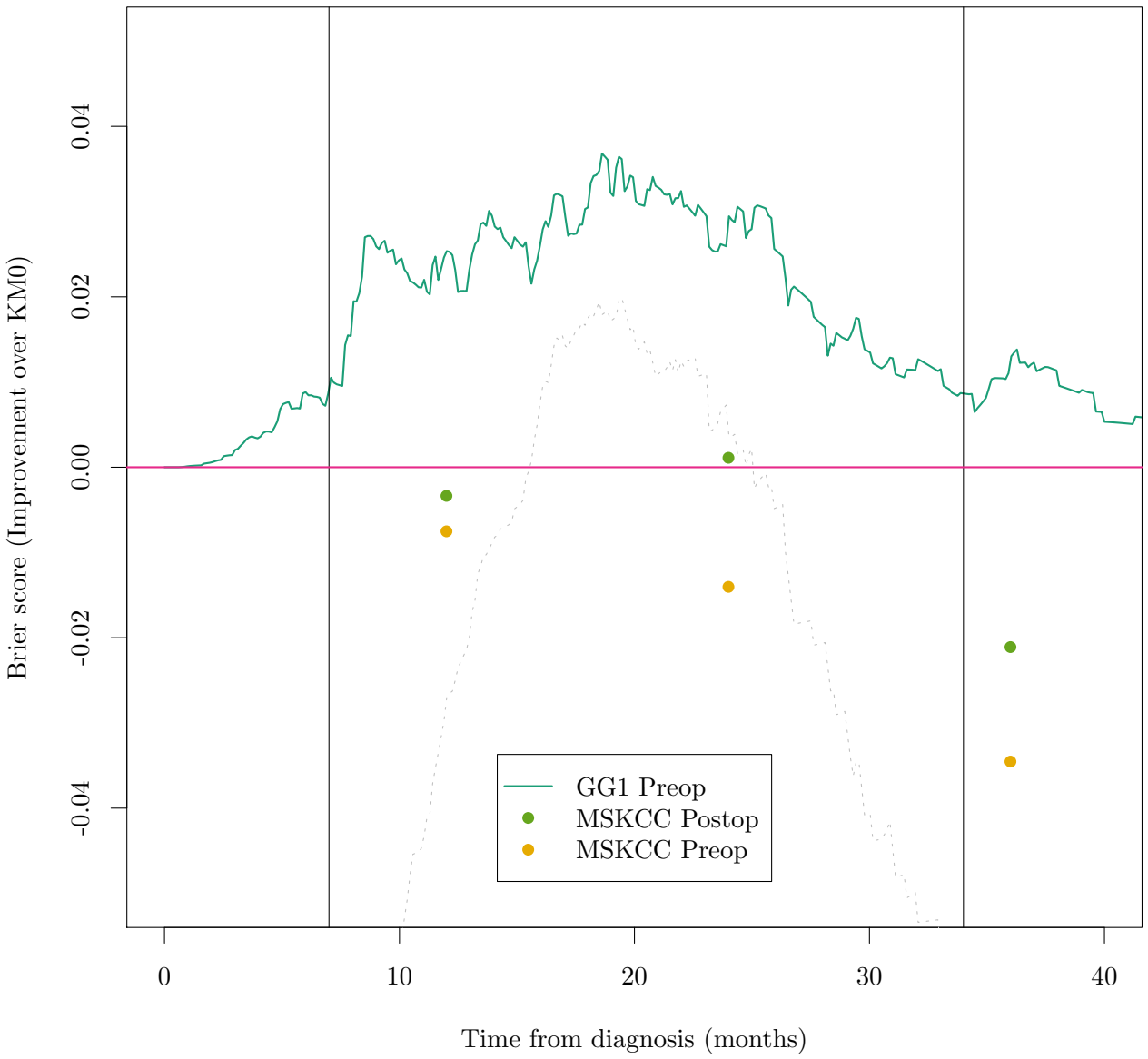

```r
plot(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - gg.path.glasgow.brier$bsc,
# lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - cph.path.glasgow.brier
# lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - rsf.path.glasgow.brier
points(c(12, 24, 36), approx(km0.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc, c(
points(c(12, 24, 36), approx(km0.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc, c(
lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - 0.25, col = "grey", lty =
abline(v = c(7, 34))
abline(h = 0, col = pal["km0"], lwd = 2)
# legend("topright",
#  legend = c( "GG1 Preop",  "CP1 Preop",  "RSF Preop",  "MSKCC Postop",  "MSKCC Preop"),
#  pch = c( NA,  NA,  NA,  16,  16),
#  col = c( pal["gg"],  pal["cph"],  pal["rsf"],  pal["mskcc.pre"],  pal["mskcc.post"]),
#  lty = c( "solid",  "solid",  "solid",  NA,  NA),
#  inset = 0.05, lwd = 2)
legend("topright",
       legend = c(     "GG1 Preop",    "MSKCC Postop",         "MSKCC Preop"),
       pch = c(        NA,                     16,                             16),
```

```
       col = c(        pal["gg"],              pal["mskcc.pre"],       pal["mskcc.post"]),
       lty = c(        "solid",                NA,                             NA),
       inset = 0.05, lwd = 2)
```




```
plot(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - gg.path.glasgow.brier$bsc,
# lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - cph.path.glasgow.brier
# lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - rsf.path.glasgow.brier
points(c(12, 24, 36), approx(km0.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc, c
points(c(12, 24, 36), approx(km0.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc, c
lines(gg.path.glasgow.brier$eval_times/365.25*12, km0.path.glasgow.brier$bsc - 0.25, col = "grey", lty =
abline(v = c(7, 34))
abline(h = 0, col = pal["km0"], lwd = 2)
# legend("bottom",
#  legend = c( "GG1 Preop",  "CP1 Preop",  "RSF Preop",  "MSKCC Postop",  "MSKCC Preop"),
#  pch = c( NA,  NA,  NA,  16,  16),
#  col = c( pal["gg"],  pal["cph"],  pal["rsf"],  pal["mskcc.pre"],  pal["mskcc.post"]),
#  lty = c( "solid",  "solid",  "solid",  NA,  NA),
```

```
#  inset = 0.05, lwd = 2)
legend("bottom",
        legend = c(     "GG1 Preop",    "MSKCC Postop",         "MSKCC Preop"),
        pch = c(        NA,                     16,                             16),
        col = c(        pal["gg"],              pal["mskcc.pre"],       pal["mskcc.post"]),
        lty = c(        "solid",                NA,                             NA),
        inset = 0.05, lwd = 2)
```




```
probs_bs_boot_func = function(d, i) {
        bs.mskcc.postop.12 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), mskcc_post.12mo.glasgow[i], 12/12*3
        bs.mskcc.postop.24 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), mskcc_post.24mo.glasgow[i], 24/12*3
        bs.mskcc.postop.36 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), mskcc_post.36mo.glasgow[i], 36/12*3
        bs.mskcc.preop.12 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), mskcc_pre.12mo.glasgow[i], 12/12*365
        bs.mskcc.preop.24 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), mskcc_pre.24mo.glasgow[i], 24/12*365
        bs.mskcc.preop.36 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), mskcc_pre.36mo.glasgow[i], 36/12*365

        bs.gg.vals = t(sapply(gg.path.glasgow[i], function(path) approx(path[,1], path[,2], c(12, 24, 36
```

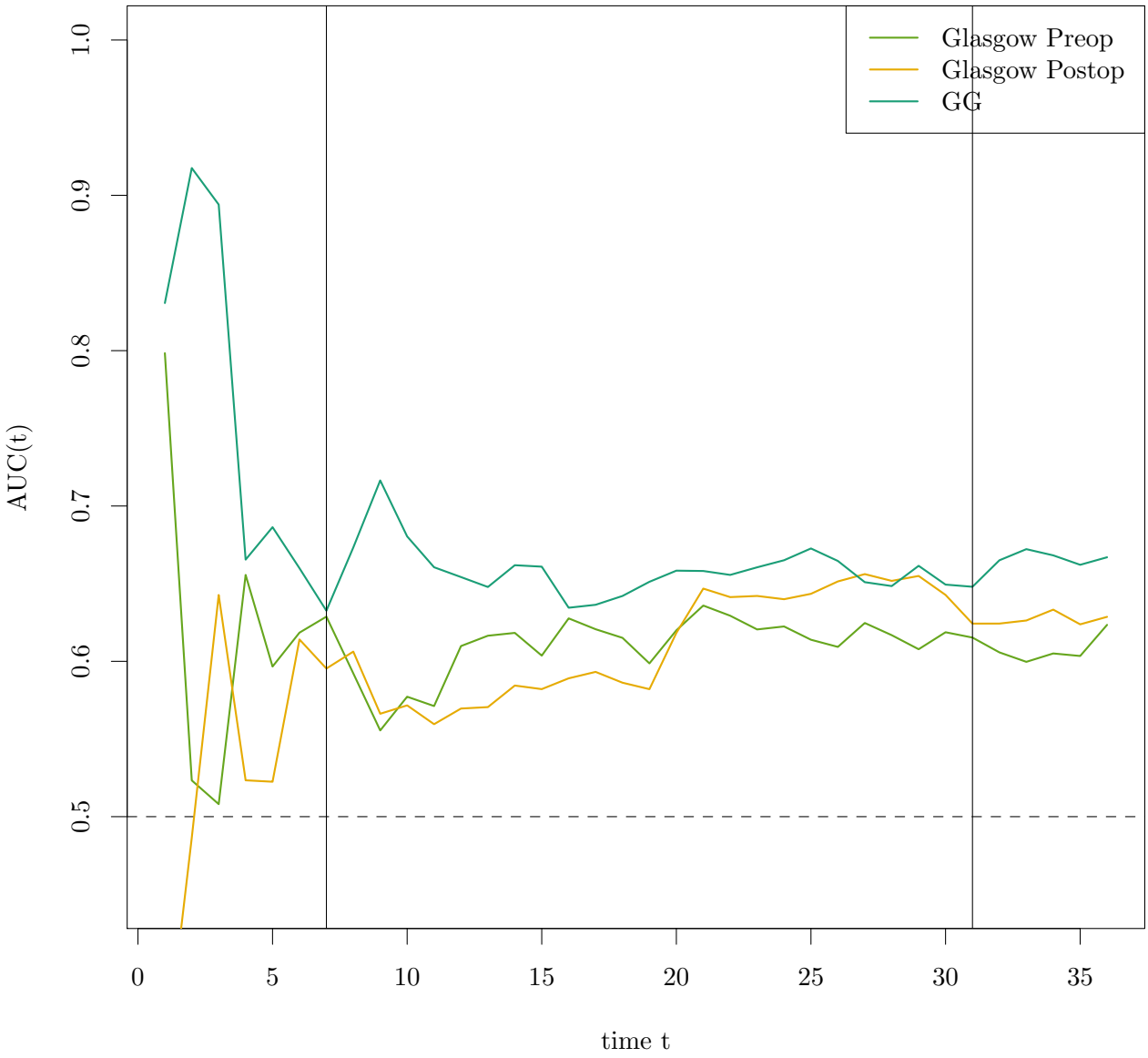
```
        rownames(bs.gg.vals) <- NULL
        bs.gg.12 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), bs.gg.vals[,1], 12/12*365.25)
        bs.gg.24 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), bs.gg.vals[,2], 24/12*365.25)
        bs.gg.36 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), bs.gg.vals[,3], 36/12*365.25)

        # cph.pred = survfit(fit.cph, newdata = d[i,])
        # cph.pred.expanded_strata = rep(names(cph.pred$strata), cph.pred$strata)
        # cph.pred_funcs = sapply(rownames(d)[i], function(pat_id) {
        #  approxfun(cph.pred$time[cph.pred.expanded_strata == pat_id], cph.pred$surv[cph.pred.expanded_
        # })
        # bs.cph.12 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), sapply(rownames(d)[i], function(pat_id) cp
        # bs.cph.24 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), sapply(rownames(d)[i], function(pat_id) cp
        # bs.cph.36 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), sapply(rownames(d)[i], function(pat_id) cp

        bs.km0.vals = approx(fit.km0$time, fit.km0$surv, c(12, 24, 36)/12*365.25)$y
        bs.km0.12 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), rep(bs.km0.vals[1], nrow(d[i,])), 12/12*365.
        bs.km0.24 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), rep(bs.km0.vals[2], nrow(d[i,])), 24/12*365.
        bs.km0.36 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), rep(bs.km0.vals[3], nrow(d[i,])), 36/12*365.

        # result = c(
        #  bs.cph.12 - bs.km0.12,  bs.gg.12 - bs.km0.12,  bs.mskcc.postop.12 - bs.km0.12,  bs.mskcc.pred
        #  bs.cph.12 - bs.mskcc.preop.12,  bs.gg.12 - bs.mskcc.preop.12,  bs.mskcc.postop.12 - bs.mskcc.
        #  bs.cph.12 - bs.mskcc.postop.12, bs.gg.12 - bs.mskcc.postop.12,
        #  bs.cph.12 - bs.gg.12,
        #  bs.cph.24 - bs.km0.24,  bs.gg.24 - bs.km0.24,  bs.mskcc.postop.24 - bs.km0.24,  bs.mskcc.pred
        #  bs.cph.24 - bs.mskcc.preop.24,  bs.gg.24 - bs.mskcc.preop.24,  bs.mskcc.postop.24 - bs.mskcc.
        #  bs.cph.24 - bs.mskcc.postop.24, bs.gg.24 - bs.mskcc.postop.24,
        #  bs.cph.24 - bs.gg.24,
        #  bs.cph.36 - bs.km0.36,  bs.gg.36 - bs.km0.36,  bs.mskcc.postop.36 - bs.km0.36,  bs.mskcc.pred
        #  bs.cph.36 - bs.mskcc.preop.36,  bs.gg.36 - bs.mskcc.preop.36,  bs.mskcc.postop.36 - bs.mskcc.
        #  bs.cph.36 - bs.mskcc.postop.36, bs.gg.36 - bs.mskcc.postop.36,
        #  bs.cph.36 - bs.gg.36)

        result = c(
                bs.gg.12 - bs.km0.12,                   bs.mskcc.preop.12 - bs.km0.12,
                bs.gg.12 - bs.mskcc.preop.12,
                bs.gg.24 - bs.km0.24,                   bs.mskcc.preop.24 - bs.km0.24,
                bs.gg.24 - bs.mskcc.preop.24,
                bs.gg.36 - bs.km0.36,                   bs.mskcc.preop.36 - bs.km0.36,
                bs.gg.36 - bs.mskcc.preop.36)

        names(result) <- NULL
        result
}

set.seed(20150208)
deltaBrier.boot.glasgow = boot(data.glasgow, probs_bs_boot_func, R = 500)
deltaBrier.boot.glasgow.cis = t(sapply(1:ncol(deltaBrier.boot.glasgow$t), function(i) boot.ci(deltaBrier
colnames(deltaBrier.boot.glasgow.cis) = c("level", "lowindex", "highindex", "lci", "uci")
# rownames(deltaBrier.boot.glasgow.cis) = c(
#  "12:cph-km0", "12:gg-km0", "12:post-km0", "12:pre-km0", "12:cph-pre", "12:gg-pre", "12:post-pre", "12
#  "24:cph-km0", "24:gg-km0", "24:post-km0", "24:pre-km0", "24:cph-pre", "24:gg-pre", "24:post-pre", "24
#  "36:cph-km0", "36:gg-km0", "36:post-km0", "36:pre-km0", "36:cph-pre", "36:gg-pre", "36:post-pre", "36
```

```
rownames(deltaBrier.boot.glasgow.cis) = c(
        "12:gg-km0", "12:pre-km0", "12:gg-pre",
        "24:gg-km0", "24:pre-km0", "24:gg-pre",
        "36:gg-km0", "36:pre-km0", "36:gg-pre")
deltaBrier.boot.glasgow

##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data.glasgow, statistic = probs_bs_boot_func, R = 500)
##
##
## Bootstrap Statistics :
##      original     bias    std. error
## t1* -0.025329  0.0001817    0.010606
## t2*  0.007328  0.0006720    0.014598
## t3* -0.032657 -0.0004902    0.018644
## t4* -0.029301 -0.0002576    0.011264
## t5*  0.015034  0.0006023    0.021414
## t6* -0.044335 -0.0008599    0.021193
## t7* -0.012862 -0.0001305    0.006532
## t8*  0.035751  0.0004482    0.018172
## t9* -0.048612 -0.0005786    0.017026

deltaBrier.boot.glasgow.cis

##            level lowindex highindex       lci       uci
## 12:gg-km0   0.95    16.68     491.8 -0.044114 -0.002791
## 12:pre-km0  0.95    11.63     487.5 -0.022014  0.036479
## 12:gg-pre   0.95    13.26     489.1 -0.069324  0.002733
## 24:gg-km0   0.95    11.13     486.9 -0.053063 -0.008089
## 24:pre-km0  0.95     8.67     483.4 -0.032969  0.054596
## 24:gg-pre   0.95    16.77     491.9 -0.082761  0.002474
## 36:gg-km0   0.95     9.75     485.4 -0.027364 -0.001529
## 36:pre-km0  0.95     9.11     484.4 -0.003833  0.067439
## 36:gg-pre   0.95    19.22     493.4 -0.079310 -0.009678
```

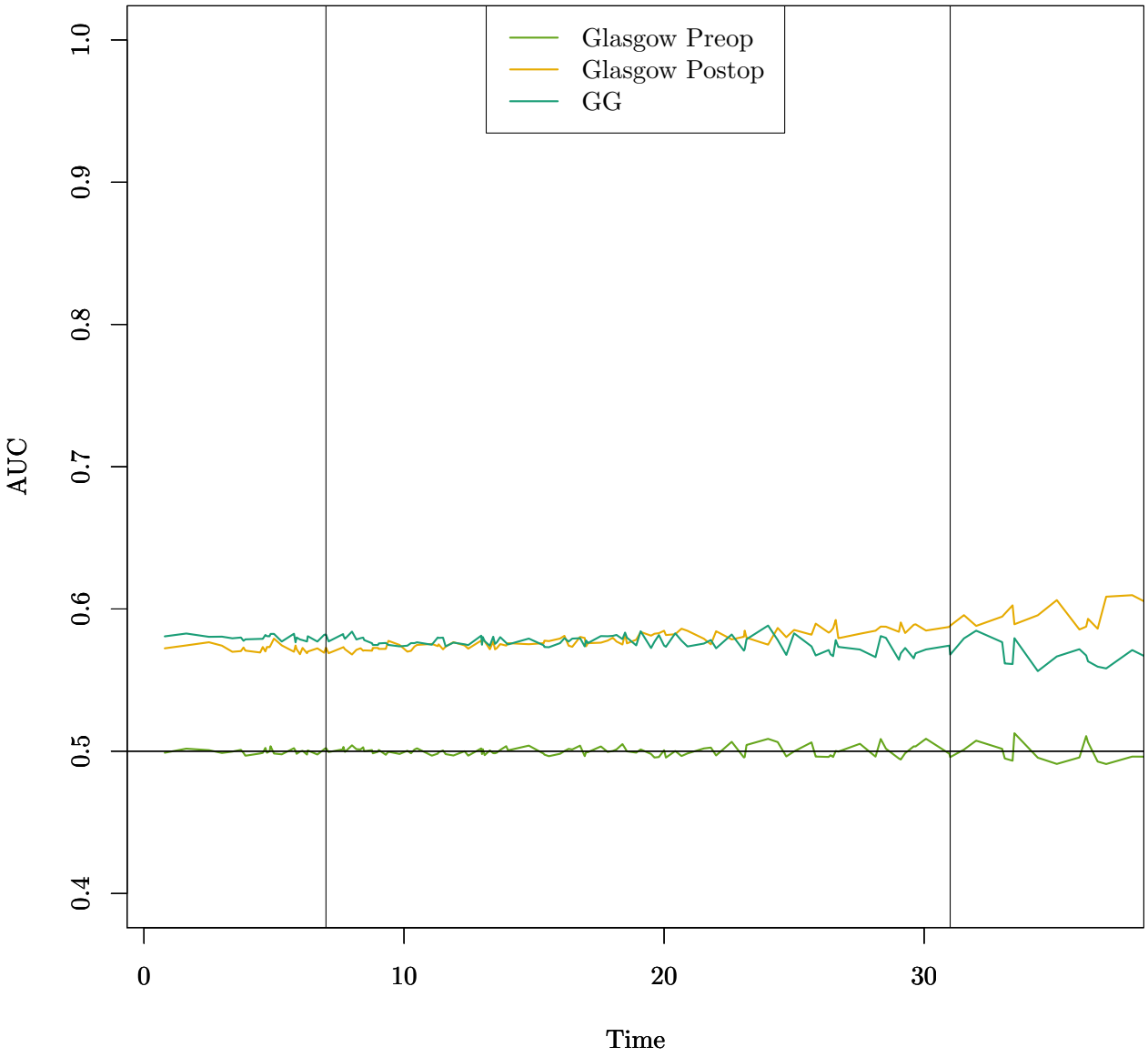
```
temp.time = gsub(":.*", "", rownames(deltaBrier.boot.glasgow.cis))
temp.methodpos = gsub(".*:", "", gsub("-.*", "", rownames(deltaBrier.boot.glasgow.cis)))
temp.methodneg = gsub(".*-", "", rownames(deltaBrier.boot.glasgow.cis))
temp.methods = sort(unique(c(temp.methodpos, temp.methodneg)))
tapply(1:length(temp.time), temp.time, function(is) {
        res = matrix(0, nrow = length(temp.methods), ncol = length(temp.methods))
        rownames(res) = temp.methods
        colnames(res) = temp.methods
        # Make res signed.  0 => NS.  +1 => row is better than col (BS_row - BS_col < 0).  -1 => row is
        res[cbind(temp.methodpos[is], temp.methodneg[is])] = (sign(deltaBrier.boot.glasgow.cis[is, "uci'
        res[cbind(temp.methodneg[is], temp.methodpos[is])] = (sign(deltaBrier.boot.glasgow.cis[is, "uci'
        res
})

## $`12`
```

```
##     gg km0 pre
## gg   0   1   0
## km0 -1   0   0
## pre  0   0   0
##
## $`24`
##     gg km0 pre
## gg   0   1   0
## km0 -1   0   0
## pre  0   0   0
##
## $`36`
##     gg km0 pre
## gg   0   1   1
## km0 -1   0   0
## pre -1   0   0
```

```
mskcc_pre.cdroc.glasgow = timeROC(data.glasgow$Time/365.25*12, data.glasgow$DSD, mskcc_pre.linpred.glasg
mskcc_post.cdroc.glasgow = timeROC(data.glasgow$Time/365.25*12, data.glasgow$DSD, mskcc_post.linpred.gla
gg.cdroc.glasgow = timeROC(data.glasgow$Time/365.25*12, data.glasgow$DSD, gg.linpred.glasgow, cause = 1,
# cph.cdroc.glasgow = timeROC(data.glasgow£Time/365.25*12, data.glasgow£DSD, cph.linpred.glasgow, cause
# rsf.cdroc.glasgow = timeROC(data.glasgow£Time/365.25*12, data.glasgow£DSD, rsf.linpred.glasgow, cause
plotAUCcurve(mskcc_pre.cdroc.glasgow, conf.int = FALSE, add = FALSE, col = pal["mskcc.pre"])
plotAUCcurve(mskcc_post.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = pal["mskcc.post"])
plotAUCcurve(gg.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = pal["gg"])
# plotAUCcurve(cph.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = pal["cph"])
# plotAUCcurve(rsf.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = pal["rsf"])
# legend("topright", legend = c("Glasgow Preop", "Glasgow Postop", "GG", "CPH", "RSF"), col = c(pal["msk
legend("topright", legend = c("Glasgow Preop", "Glasgow Postop", "GG"), col = c(pal["mskcc.pre"], pal["m
abline(v = c(7, 31))
```




```
# invisible(risksetROC(data.glasgow£Time/365.25*12, status = data.glasgow£DSD, marker = mskcc_pre.linpre
# invisible(risksetROC(data.glasgow£Time/365.25*12, status = data.glasgow£DSD, marker = mskcc_post.linpr
# invisible(risksetROC(data.glasgow£Time/365.25*12, status = data.glasgow£DSD, marker = gg.linpred.glasg
# invisible(risksetROC(data.glasgow£Time/365.25*12, status = data.glasgow£DSD, marker = cph.linpred.glas
# invisible(risksetROC(data.glasgow£Time/365.25*12, status = data.glasgow£DSD, marker = rsf.linpred.glas
invisible(risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = mskcc_pre.linpred.
par(new = TRUE)
invisible(risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = mskcc_post.linpred
par(new = TRUE)
invisible(risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = gg.linpred.glasgow
par(new = TRUE)
# invisible(risksetAUC(data.glasgow£Time/365.25*12, status = data.glasgow£DSD, marker = cph.linpred.glas
# par(new = TRUE)
# invisible(risksetAUC(data.glasgow£Time/365.25*12, status = data.glasgow£DSD, marker = rsf.linpred.glas
# legend("top", legend = c("Glasgow Preop", "Glasgow Postop", "GG", "CPH", "RSF"), col = c(pal["mskcc.pr
legend("top", legend = c("Glasgow Preop", "Glasgow Postop", "GG"), col = c(pal["mskcc.pre"], pal["mskcc.
abline(v = c(7, 31))
```


Glasgow Preop
Glasgow Postop
GG
AUC
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
10
20
30
Time