```r
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
```

# 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.

| 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 | |

So the preoperative MSKCC score would be:

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

```
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(
                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 ever
                # I'll never know.  The T34 coefficient of 11 was arrived at as (0.828*10+(1-0.037-0.119
                # 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$gg
fit.gg2 = temp$gg2
```

```r
fit.cph = temp$cph
fit.km0 = temp$km0
fit.rsf = temp$rsf
data.nswpcn = temp$data.train
```

```r
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

```r
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.Locatio

## 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.

```
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

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
```

And the GG2

```
gg2.path.glasgow = summary(fit.gg2, newdata = data.glasgow, ci = FALSE)
temp.coefs = coef(fit.gg2)
gg2.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))
        } })
gg2.linpred.glasgow = -rowSums(gg2.linpred.glasgow)     # Negate to bring into concordance with the dire

gg2.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)) {
```

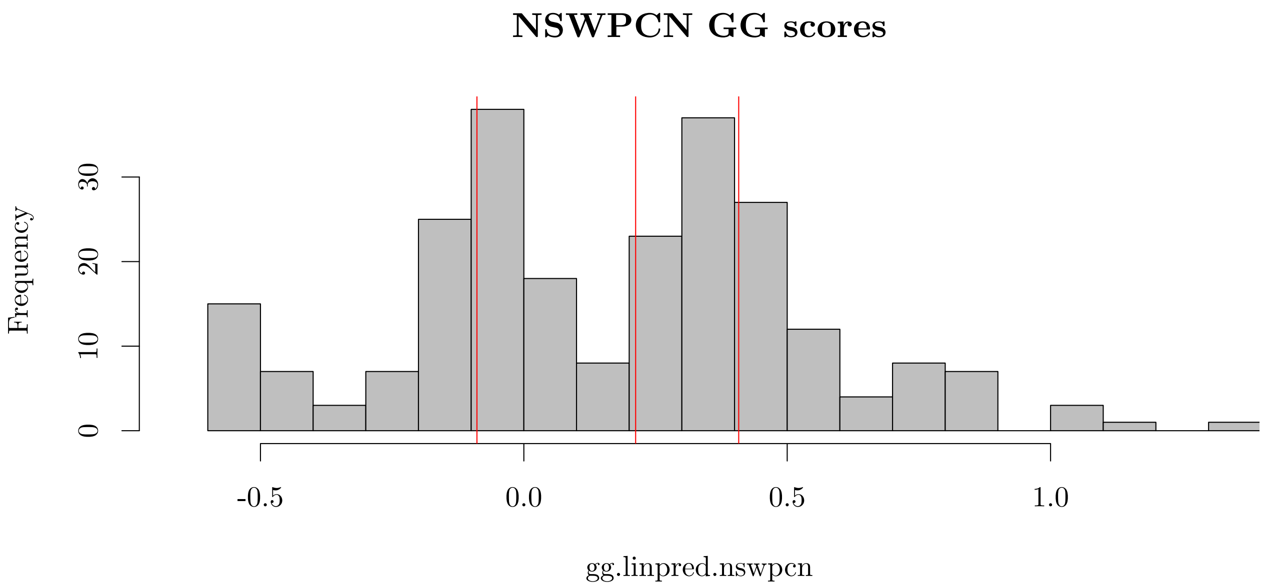
```
                temp.coefs[coef_i] * data.nswpcn[,gsub("TRUE$", "", names(temp.coefs)[coef_i])]
        } else {
                rep(0, nrow(data.nswpcn))
        } })
gg2.linpred.nswpcn = -rowSums(gg2.linpred.nswpcn)               # Negate to bring into concordance with
```

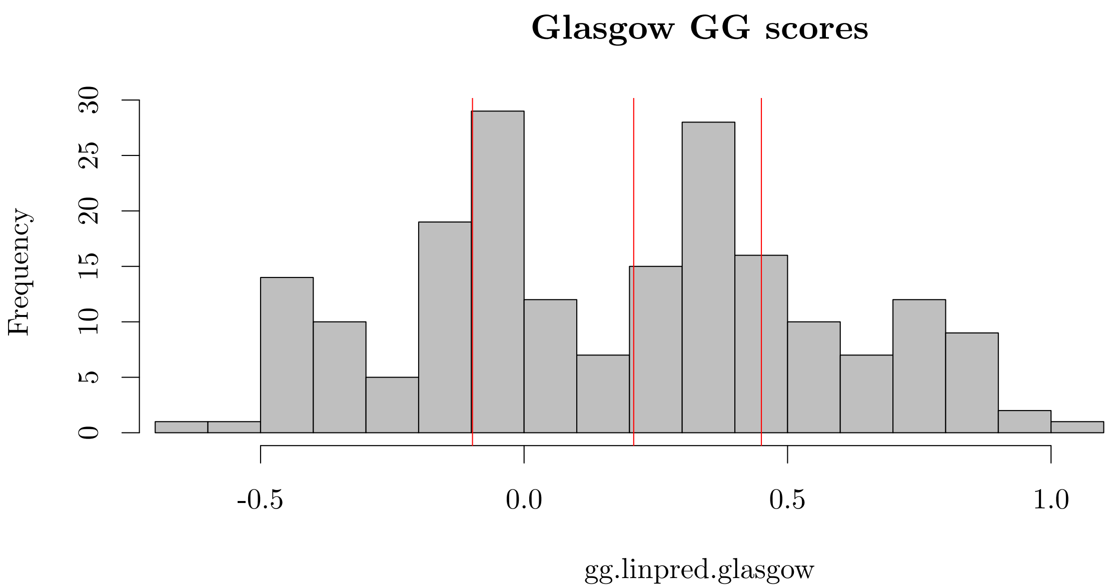
```
temp.coefs = coef(fit.cph)
cph.linpred.nswpcn = sapply(1:length(temp.coefs), function(coef_i) {
        if (names(temp.coefs)[coef_i] %in% colnames(data.nswpcn)) {
                temp.coefs[coef_i] * data.nswpcn[,names(temp.coefs)[coef_i]]
        } else if (gsub("TRUE$", "", names(temp.coefs)[coef_i]) %in% colnames(data.nswpcn)) {
                temp.coefs[coef_i] * data.nswpcn[,gsub("TRUE$", "", names(temp.coefs)[coef_i])]
        } else {
                rep(0, nrow(data.nswpcn))
        } })
cph.linpred.nswpcn = rowSums(cph.linpred.nswpcn)

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)

# 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)
```

# 4 Validation

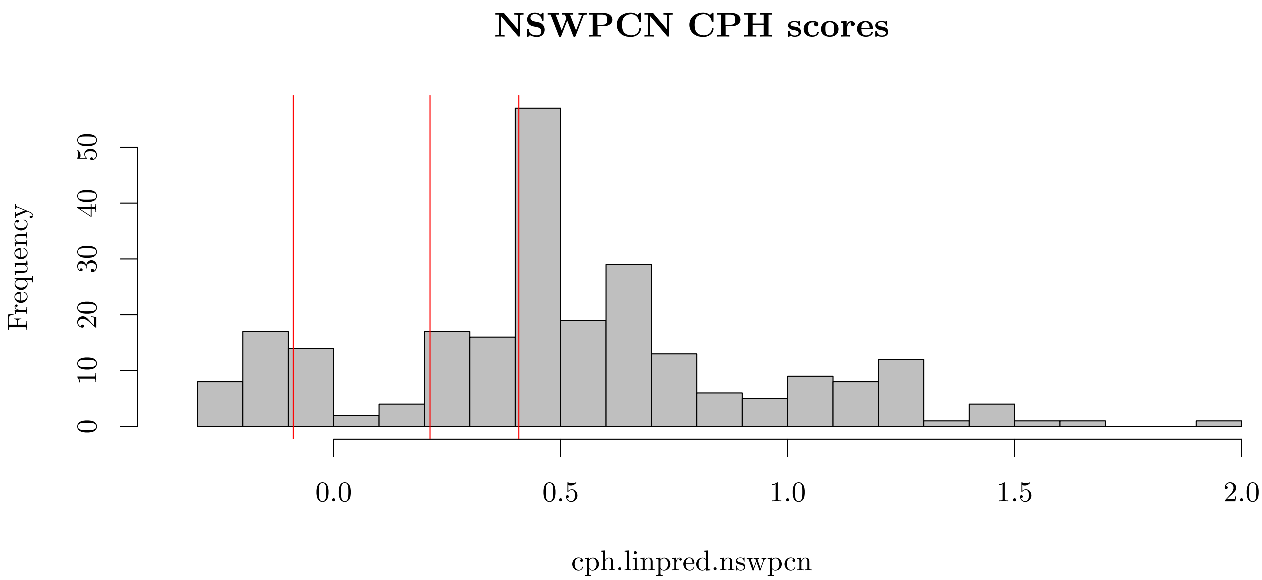
## 4.1 Altman diagnostic 1: score histograms

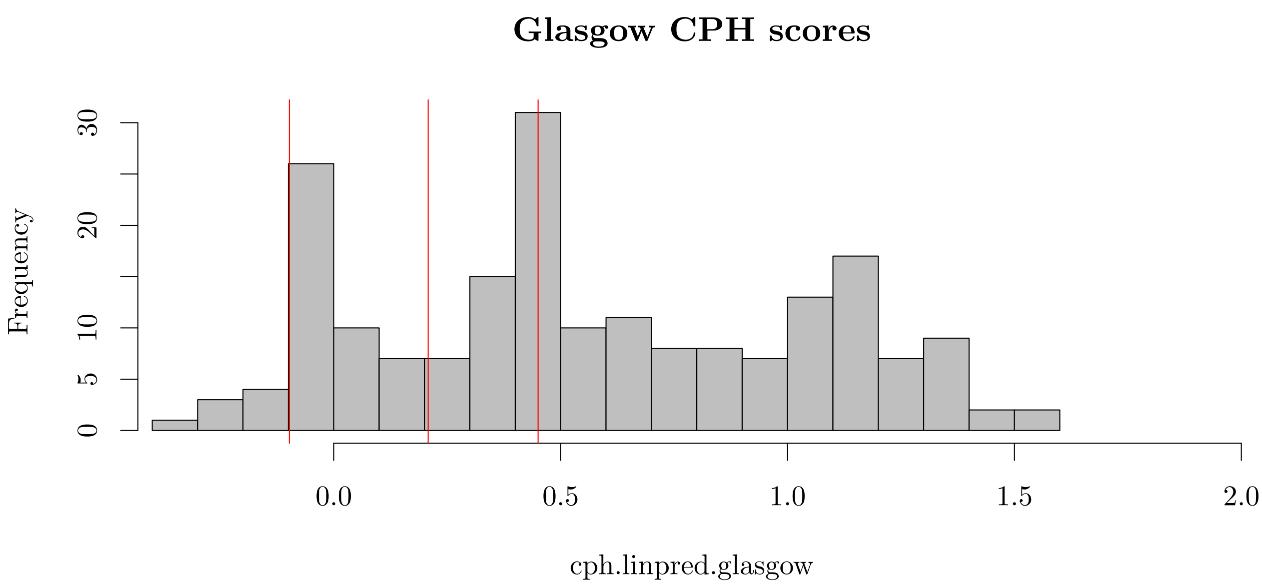
```
par(mfrow = c(2, 1))
hist(gg.linpred.nswpcn, main = "NSWPCN GG scores", xlim = range(c(gg.linpred.nswpcn, gg.linpred.glasgow)
abline(v = quantile(gg.linpred.nswpcn, probs = c(0.25, 0.5, 0.75)), col = "red")
hist(gg.linpred.glasgow, main = "Glasgow GG scores", xlim = range(c(gg.linpred.nswpcn, gg.linpred.glasgo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




```
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.glas
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.gl
abline(v = quantile(gg.linpred.glasgow, probs = c(0.25, 0.5, 0.75)), col = "red")
```




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

## 4.2 Altman method 1 (D,F)

```r
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 0.718     2.051    0.214 3.36  0.00078
##
##                    exp(coef) exp(-coef) lower .95 upper .95
## gg.linpred.glasgow      2.05      0.488      1.35      3.12
##
## Concordance= 0.602  (se = 0.025 )
## Rsquare= 0.056   (max possible= 0.999 )
## Likelihood ratio test= 11.3  on 1 df,   p=0.00077
## Wald test            = 11.3  on 1 df,   p=0.000779
## Score (logrank) test = 11.4  on 1 df,   p=0.000738
```

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

```
## Call:
## coxph(formula = Surv(Time, DSD) ~ cph.linpred.glasgow, data = data.glasgow)
##
```

```
##   n= 198, number of events= 170
##
##                        coef exp(coef) se(coef)    z Pr(>|z|)
## cph.linpred.glasgow 1.012     2.752    0.179 5.66  1.5e-08
##
##                       exp(coef) exp(-coef) lower .95 upper .95
## cph.linpred.glasgow      2.75      0.363      1.94      3.91
##
## Concordance= 0.658  (se = 0.025 )
## Rsquare= 0.148   (max possible= 0.999 )
## Likelihood ratio test= 31.6  on 1 df,   p=1.85e-08
## Wald test            = 32.1  on 1 df,   p=1.48e-08
## Score (logrank) test = 33.1  on 1 df,   p=8.54e-09
```

```
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  1.73  1       0.19
```

```
anova(coxph(Surv(Time, DSD) ~ offset(cph.linpred.glasgow) + cph.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                    -713
## cph.linpred.glasgow   -713     0  1       0.95
```

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  0.69598   2.00568  0.16160  4.31  1.7e-05
## SizeCent  0.02457   1.02487  0.00737  3.33  0.00086
## A2TRUE    0.31058   1.36422  0.17387  1.79  0.07406
## A4TRUE   -0.04240   0.95849  0.17723 -0.24  0.81093
##
##          exp(coef) exp(-coef) lower .95 upper .95
## AgeCent      0.968      1.033     0.952     0.984
## SexMTRUE     2.006      0.499     1.461     2.753
## SizeCent     1.025      0.976     1.010     1.040
## A2TRUE       1.364      0.733     0.970     1.918
## A4TRUE       0.958      1.043     0.677     1.357
##
## Concordance= 0.681  (se = 0.025 )
## Rsquare= 0.208   (max possible= 0.999 )
## Likelihood ratio test= 46.1  on 5 df,   p=8.58e-09
## Wald test            = 46.9  on 5 df,   p=5.86e-09
## Score (logrank) test = 49.1  on 5 df,   p=2.14e-09
```

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

```
## Call:
## coxph(formula = Surv(Time, DSD) ~ offset(cph.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  0.26736   1.30651  0.16160  1.65  0.09803
## SizeCent  0.01982   1.02002  0.00737  2.69  0.00719
## A2TRUE    0.10517   1.11090  0.17387  0.60  0.54526
## A4TRUE   -0.15400   0.85728  0.17723 -0.87  0.38489
##
##          exp(coef) exp(-coef) lower .95 upper .95
## AgeCent      0.968      1.033     0.952     0.984
## SexMTRUE     1.307      0.765     0.952     1.793
## SizeCent     1.020      0.980     1.005     1.035
## A2TRUE       1.111      0.900     0.790     1.562
## A4TRUE       0.857      1.166     0.606     1.213
##
## Concordance= 0.681  (se = 0.025 )
## Rsquare= 0.114   (max possible= 0.999 )
## Likelihood ratio test= 24.1  on 5 df,   p=0.000211
## Wald test            = 24.9  on 5 df,   p=0.000142
## Score (logrank) test = 25.5  on 5 df,   p=0.000112
```

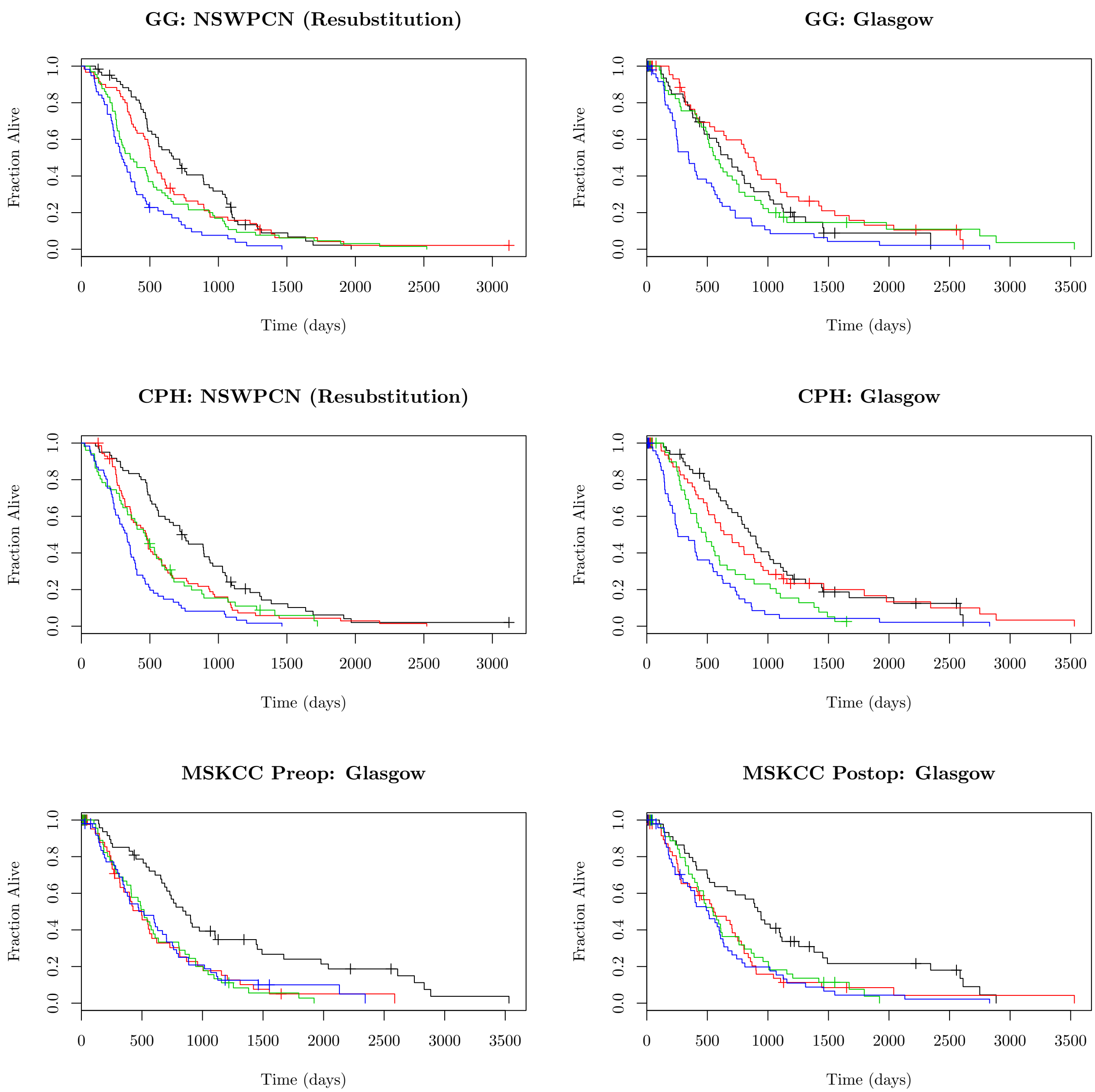
Still strong evidence of misspecification or poor fit. However, the above calibration slope was not significantly different from 1. Hmm. This doesn't necessarily sink the method, but will need checking as we go along.

### 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_quan
mskcc_post.groups.glasgow = cut(mskcc_post.linpred.glasgow, quantile(mskcc_post.linpred.glasgow, group_q
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))

par(mfrow = c(3, 2))
plot(survfit(Surv(data.nswpcn$Time, data.nswpcn$DSD) ~ gg.groups.nswpcn), col = 1:(length(group_quantile
plot(survfit(Surv(data.glasgow$Time, data.glasgow$DSD) ~ gg.groups.glasgow), col = 1:(length(group_quant
plot(survfit(Surv(data.nswpcn$Time, data.nswpcn$DSD) ~ cph.groups.nswpcn), col = 1:(length(group_quantil
plot(survfit(Surv(data.glasgow$Time, data.glasgow$DSD) ~ cph.groups.glasgow), col = 1:(length(group_quan
plot(survfit(Surv(data.glasgow$Time, data.glasgow$DSD) ~ mskcc_pre.groups.glasgow), col = 1:(length(grou
plot(survfit(Surv(data.glasgow$Time, data.glasgow$DSD) ~ mskcc_post.groups.glasgow), col = 1:(length(gro
```




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

# temp = survfit(Surv(data.nswpcn£Time, data.nswpcn£DSD) ~ gg.groups.nswpcn)
# plot(0 ~ 0, type = "n", xlim = c(0, max(data.nswpcn£Time)), ylim = c(0, 1))
# for (i in )
```

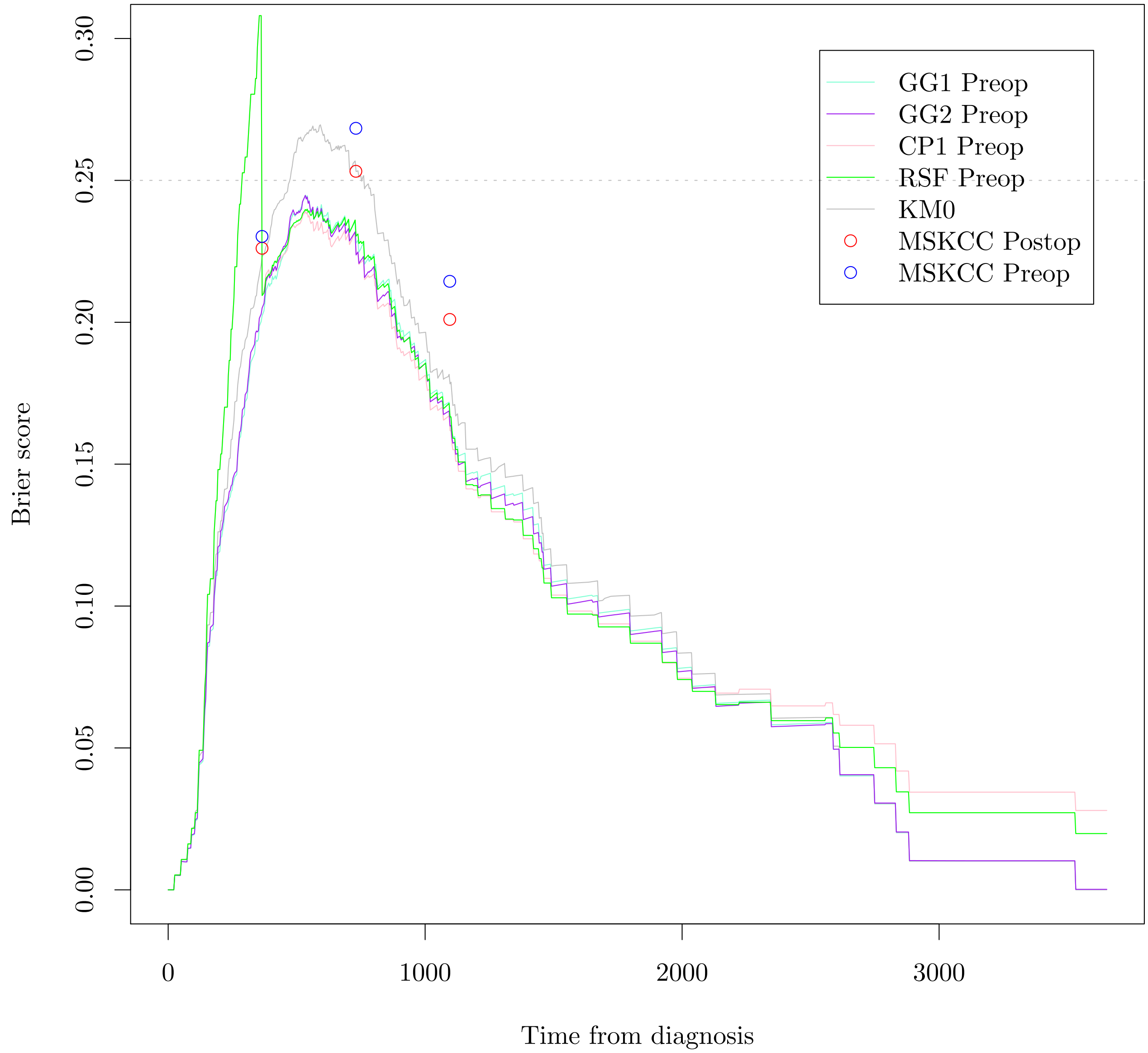
Weird. MSKCC somehow is still finding a subgroup, and it's somehow even clearer in preop! This is based on an approximation to GG only, but should be pretty close. It certainly does OK on resubstituted data, but not so well on the Glasgow patients.

## 4.6 Brier score

```
calcIBS = function(surv, pred, pred_times, max_time)
{
	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(0, 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
})
cph.path.glasgow.brier = calcIBS(Surv(data.glasgow$Time, data.glasgow$DSD),
        t(sapply(temp.cph.pred_funcs[rownames(data.glasgow)], function(f) f(c(12, 24, 36)/12*365.25))),

gg2.path.glasgow.brier = calcIBS(Surv(data.glasgow$Time, data.glasgow$DSD), t(sapply(gg2.path.glasgow, f

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$bsc ~ gg.path.glasgow.brier$eval_times, col = "aquamarine", type = "l", ylim
lines(km0.path.glasgow.brier$bsc ~ km0.path.glasgow.brier$eval_times, col = "grey")
lines(cph.path.glasgow.brier$bsc ~ cph.path.glasgow.brier$eval_times, col = "pink")
lines(gg2.path.glasgow.brier$bsc ~ gg2.path.glasgow.brier$eval_times, col = "purple")
lines(rsf.path.glasgow.brier$bsc ~ rsf.path.glasgow.brier$eval_times, col = "green")
points(c(12, 24, 36)/12*365.25, c(mskcc_post.12mo.glasgow.brier, mskcc_post.24mo.glasgow.brier, mskcc_po
points(c(12, 24, 36)/12*365.25, c(mskcc_pre.12mo.glasgow.brier, mskcc_pre.24mo.glasgow.brier, mskcc_pre.
abline(h = 0.25, col = "grey", lty = "dotted")
legend("topright",
        legend = c(     "GG1 Preop",    "GG2 Preop",    "CP1 Preop",    "RSF Preop",    "KM0",
        pch = c(        NA,                     NA,                     NA,                     NA,
        col = c(        "aquamarine",   "purple",               "pink",                 "green",
        lty = c(        "solid",                "solid",                "solid",                "solid",
        inset = 0.05)
```




```
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.expa
        })
        bs.cph.12 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), sapply(rownames(d)[i], function(pat_id) cph.
        bs.cph.24 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), sapply(rownames(d)[i], function(pat_id) cph.
        bs.cph.36 = calcBSsingle(Surv(d$Time[i], d$DSD[i]), sapply(rownames(d)[i], function(pat_id) cph.

        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
            bs.cph.12 - bs.mskcc.preop.12,  bs.gg.12 - bs.mskcc.preop.12,   bs.mskcc.postop.12 - bs.
            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
            bs.cph.24 - bs.mskcc.preop.24,  bs.gg.24 - bs.mskcc.preop.24,   bs.mskcc.postop.24 - bs.
            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
            bs.cph.36 - bs.mskcc.preop.36,  bs.gg.36 - bs.mskcc.preop.36,   bs.mskcc.postop.36 - bs.
            bs.cph.36 - bs.mskcc.postop.36, bs.gg.36 - bs.mskcc.postop.36,
            bs.cph.36 - bs.gg.36)
        names(result) <- NULL
        result
}

set.seed(20150113)
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'
        "24:cph-km0", "24:gg-km0", "24:post-km0", "24:pre-km0", "24:cph-pre", "24:gg-pre", "24:post-pre'
        "36:cph-km0", "36:gg-km0", "36:post-km0", "36:pre-km0", "36:cph-pre", "36:gg-pre", "36:post-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*  -0.0130382 -1.278e-03    0.010921
## t2*  -0.0208299 -1.331e-03    0.010856
## t3*   0.0030229 -1.048e-03    0.014649
## t4*   0.0071877 -6.540e-04    0.014936
## t5*  -0.0202259 -6.241e-04    0.020579
## t6*  -0.0280176 -6.772e-04    0.020104
```

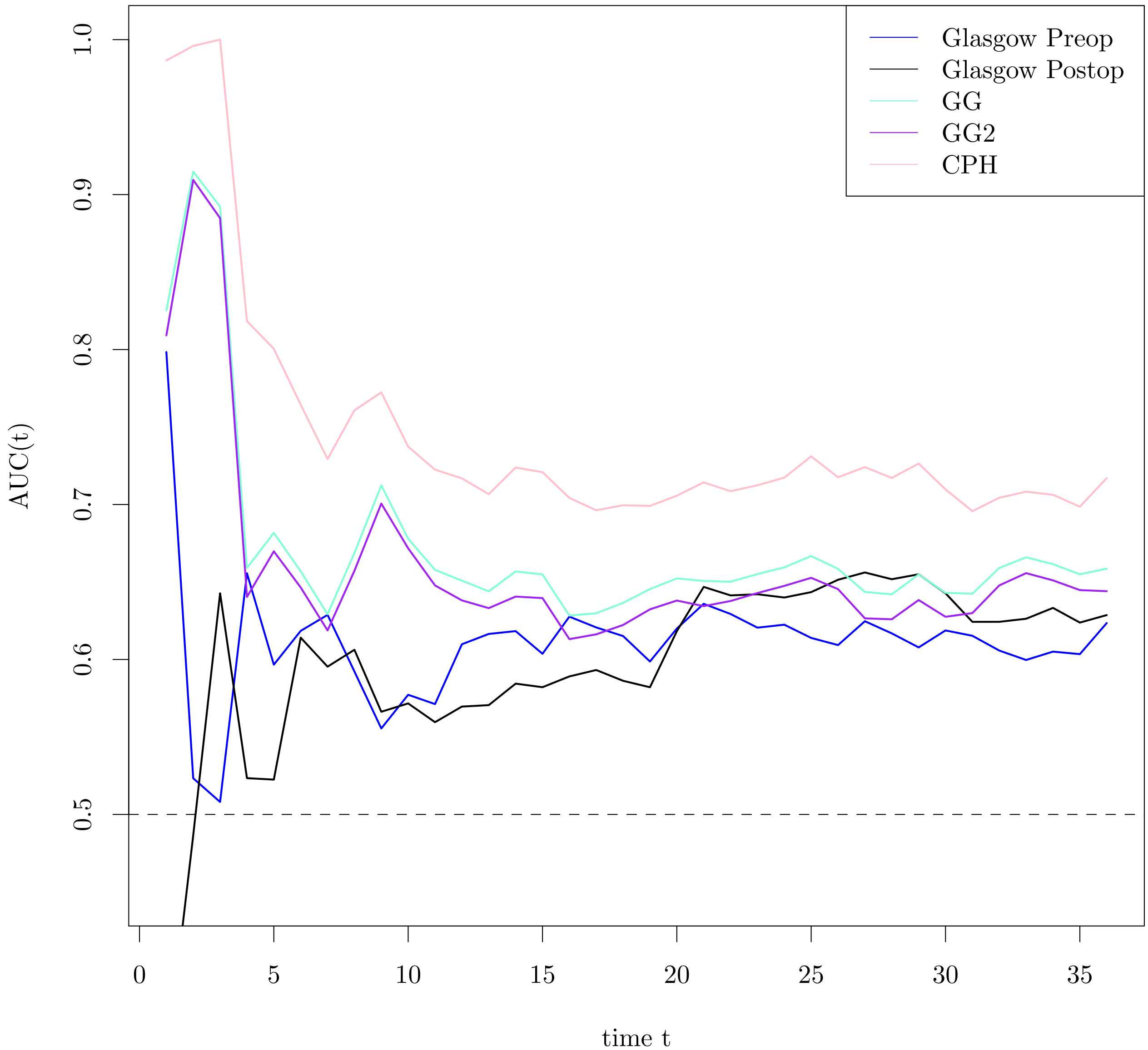
```
## t7*  -0.0041648 -3.935e-04    0.003150
## t8*  -0.0160610 -2.306e-04    0.020244
## t9*  -0.0238528 -2.837e-04    0.019807
## t10*  0.0077917  5.317e-05    0.002251
## t11* -0.0290212 -3.938e-04    0.010006
## t12* -0.0251333 -4.869e-04    0.010542
## t13*  0.0003272 -2.070e-03    0.020468
## t14*  0.0154723 -1.459e-03    0.020306
## t15* -0.0444935  1.065e-03    0.021024
## t16* -0.0406056  9.717e-04    0.021454
## t17* -0.0151451 -6.114e-04    0.005561
## t18* -0.0293483  1.676e-03    0.021050
## t19* -0.0254605  1.583e-03    0.021577
## t20* -0.0038878  9.305e-05    0.002469
## t21* -0.0163245 -5.644e-04    0.006933
## t22* -0.0116616 -4.838e-04    0.005960
## t23*  0.0228894 -2.116e-03    0.018865
## t24*  0.0363296 -1.423e-03    0.017841
## t25* -0.0526541  8.583e-04    0.016262
## t26* -0.0479912  9.390e-04    0.017138
## t27* -0.0134401 -6.928e-04    0.005662
## t28* -0.0392139  1.551e-03    0.017154
## t29* -0.0345511  1.632e-03    0.018066
## t30* -0.0046628 -8.062e-05    0.002300

deltaBrier.boot.glasgow.cis

##             level lowindex highindex        lci        uci
## 12:cph-km0   0.95    28.17     496.2 -0.0306390  0.0126239
## 12:gg-km0    0.95    27.07     495.9 -0.0386989  0.0035527
## 12:post-km0  0.95    21.80     494.4 -0.0233188  0.0366841
## 12:pre-km0   0.95    19.02     493.1 -0.0194999  0.0417737
## 12:cph-pre   0.95    11.12     487.0 -0.0659784  0.0196067
## 12:gg-pre    0.95    10.50     486.2 -0.0728125  0.0076840
## 12:post-pre  0.95    16.63     491.5 -0.0106143  0.0016693
## 12:cph-post  0.95    11.34     487.2 -0.0611593  0.0230695
## 12:gg-post   0.95    12.09     488.1 -0.0678988  0.0138256
## 12:cph-gg    0.95     8.50     483.1  0.0031365  0.0116653
## 24:cph-km0   0.95    16.86     491.9 -0.0496742 -0.0066401
## 24:gg-km0    0.95    14.09     489.9 -0.0463578 -0.0036625
## 24:post-km0  0.95    19.05     492.9 -0.0396312  0.0446312
## 24:pre-km0   0.95    16.51     491.6 -0.0237494  0.0585698
## 24:cph-pre   0.95     8.82     483.3 -0.0884392 -0.0059322
## 24:gg-pre    0.95     9.66     484.8 -0.0829245  0.0007140
## 24:post-pre  0.95    27.60     496.0 -0.0242163 -0.0011646
## 24:cph-post  0.95     8.92     483.5 -0.0719628  0.0116053
## 24:gg-post   0.95     9.78     485.0 -0.0682419  0.0166928
## 24:cph-gg    0.95    10.28     485.8 -0.0091586  0.0007611
## 36:cph-km0   0.95    20.08     493.2 -0.0291930 -0.0025001
## 36:gg-km0    0.95    15.48     490.8 -0.0235981  0.0004294
## 36:post-km0  0.95    18.10     492.3 -0.0149899  0.0608984
## 36:pre-km0   0.95    12.30     488.3  0.0007022  0.0701983
## 36:cph-pre   0.95    12.83     488.8 -0.0843222 -0.0196427
## 36:gg-pre    0.95    11.32     487.1 -0.0822648 -0.0128795
```

```
## 36:post-pre  0.95    22.31     494.5 -0.0242015 -0.0017397
## 36:cph-post  0.95    11.06     486.7 -0.0714961 -0.0032602
## 36:gg-post   0.95    10.48     485.9 -0.0687059  0.0006778
## 36:cph-gg    0.95    10.73     486.6 -0.0091047 -0.0005594
```

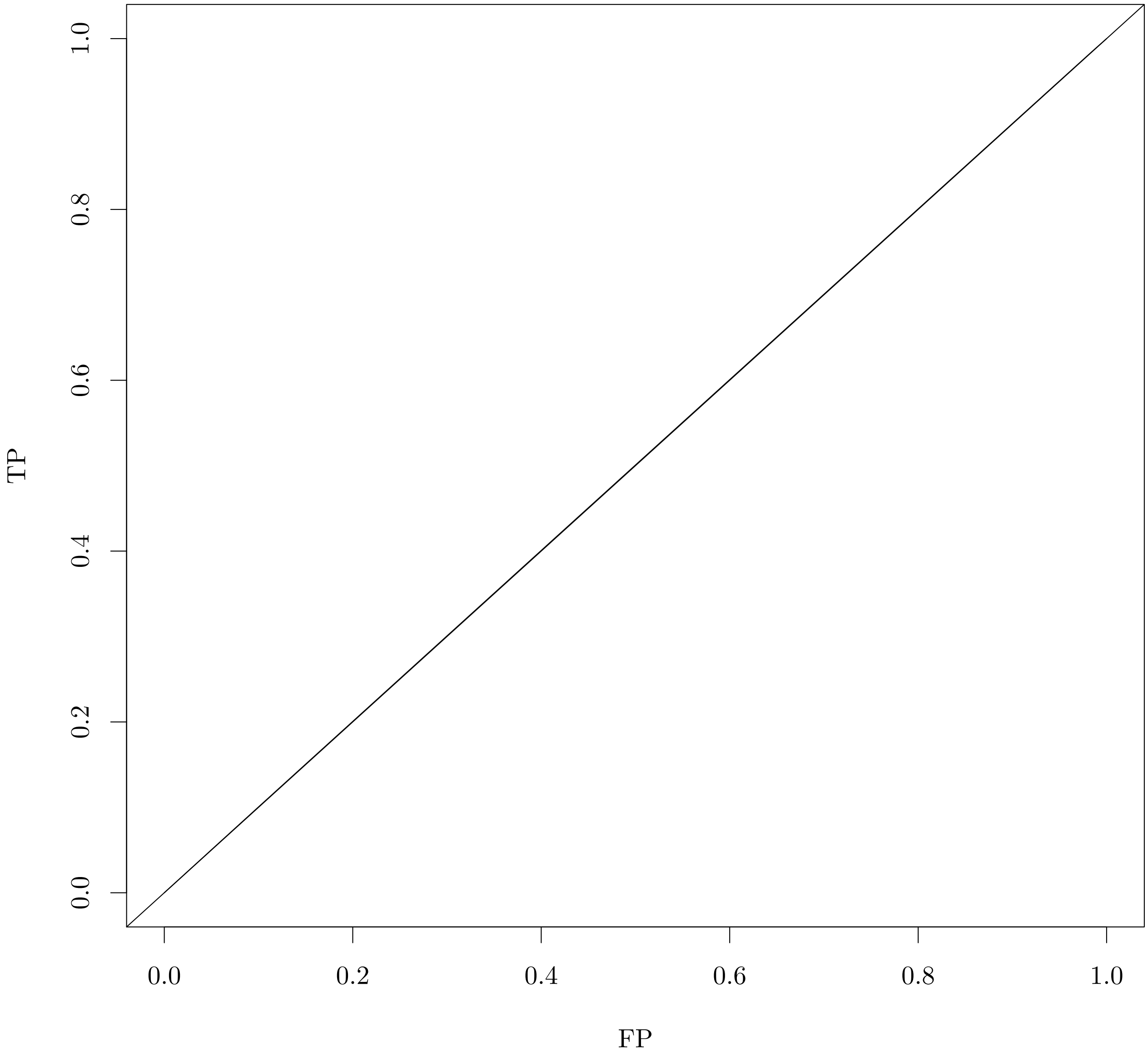
```
temp.time = gsub(":.*", "", rownames(deltaBrier.boot.glasgow.cis))
temp.methodpos = gsub(".*:", "", gsub("-.*", "", rownames(deltaBrier.boot.glasgow.cis)))
temp.methodneg = gsub(".*-", "", rownames(deltaBrier.boot.glasgow.cis))
temp.methods = sort(unique(c(temp.methodpos, temp.methodneg)))
tapply(1:length(temp.time), 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`
##      cph gg km0 post pre
## cph    0 -1   0    0   0
## gg     1  0   0    0   0
## km0    0  0   0    0   0
## post   0  0   0    0   0
## pre    0  0   0    0   0
##
## $`24`
##      cph gg km0 post pre
## cph    0  0   1    0   1
## gg     0  0   1    0   0
## km0   -1 -1   0    0   0
## post   0  0   0    0   1
## pre   -1  0   0   -1   0
##
## $`36`
##      cph gg km0 post pre
## cph    0  1   1    1   1
## gg    -1  0   0    0   1
## km0   -1  0   0    0   1
## post  -1  0   0    0   1
## pre   -1 -1  -1   -1   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,
gg2.cdroc.glasgow = timeROC(data.glasgow$Time/365.25*12, data.glasgow$DSD, gg2.linpred.glasgow, cause =
cph.cdroc.glasgow = timeROC(data.glasgow$Time/365.25*12, data.glasgow$DSD, cph.linpred.glasgow, cause =
plotAUCcurve(mskcc_pre.cdroc.glasgow, conf.int = FALSE, add = FALSE, col = "blue")
plotAUCcurve(mskcc_post.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = "black")
plotAUCcurve(gg.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = "aquamarine")
plotAUCcurve(gg2.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = "purple")
```

```r
plotAUCcurve(cph.cdroc.glasgow, conf.int = FALSE, add = TRUE, col = "pink")
legend("topright", legend = c("Glasgow Preop", "Glasgow Postop", "GG", "GG2", "CPH"), col = c("blue", "b
```

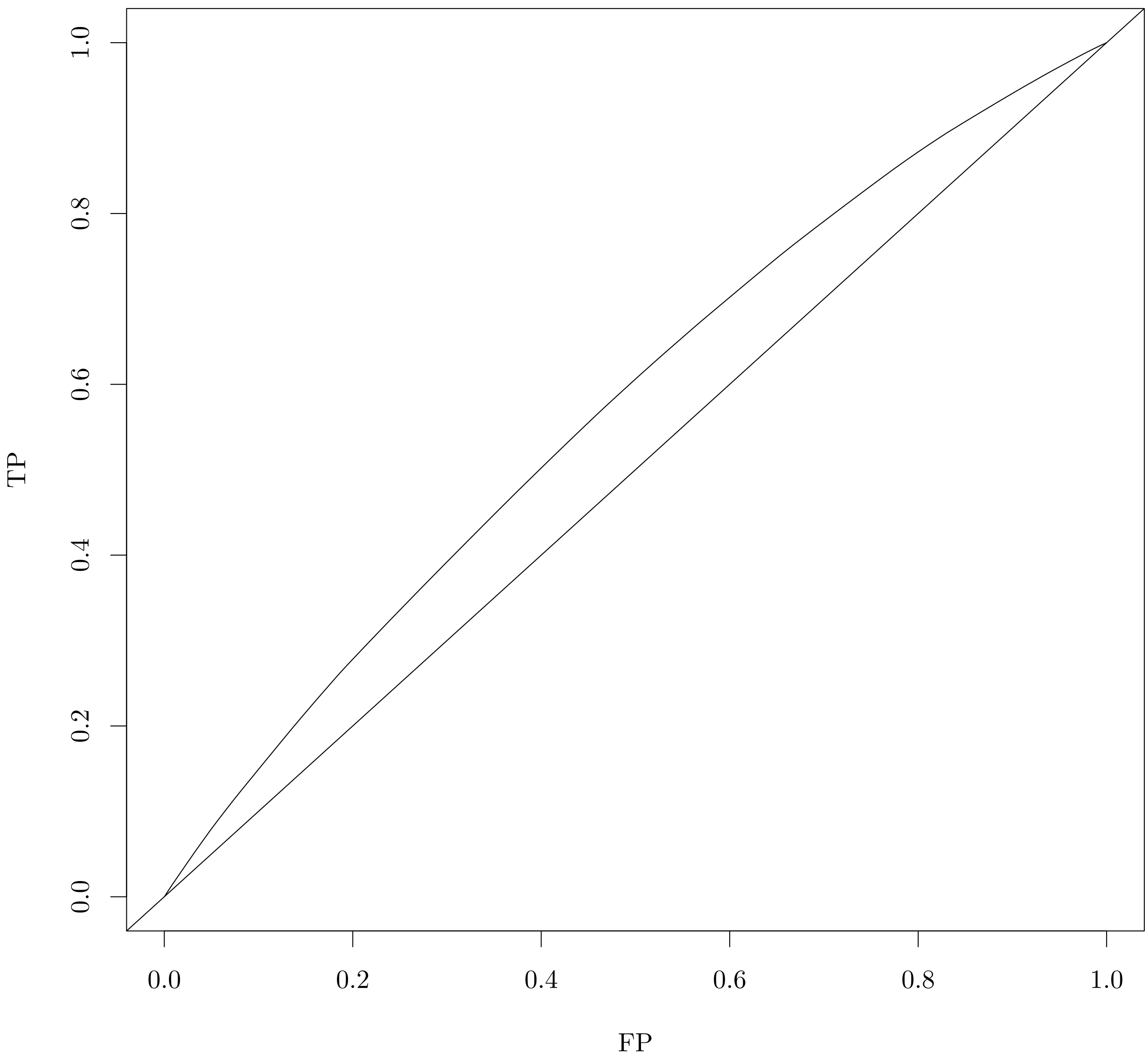



```r
risksetROC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = mskcc_pre.linpred.glasgow, p
```



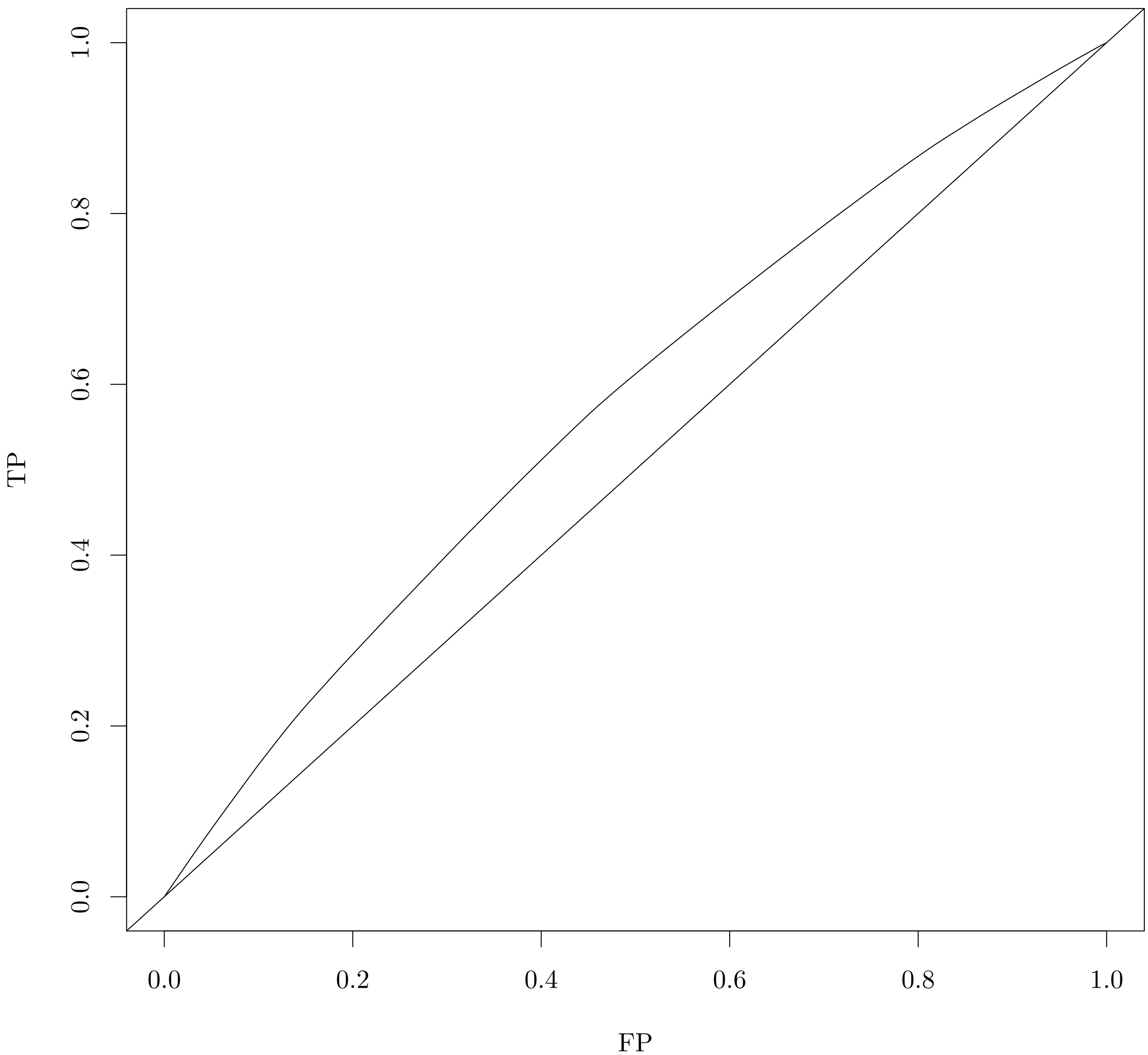

```
## $marker
##   [1] -0.09676 -0.08538 -0.07023 -0.07019 -0.06997 -0.06977 -0.06961
##   [8] -0.06958 -0.06955 -0.06950 -0.06947 -0.06928 -0.06917 -0.06916
##  [15] -0.06910 -0.06904 -0.06896 -0.06895 -0.06894 -0.06888 -0.06884
##  [22] -0.06884 -0.06881 -0.06876 -0.06867 -0.06867 -0.06865 -0.06864
##  [29] -0.06860 -0.06860 -0.06859 -0.06859 -0.06856 -0.06856 -0.06855
##  [36] -0.06854 -0.06852 -0.06851 -0.06851 -0.06850 -0.06848 -0.06844
##  [43] -0.06839 -0.06831 -0.06831 -0.06830 -0.06826 -0.06826 -0.06824
##  [50] -0.06823 -0.06823 -0.06823 -0.06823 -0.06823 -0.06822 -0.06821
##  [57] -0.06819 -0.06817 -0.06814 -0.06812 -0.06807 -0.06805 -0.06799
##  [64] -0.06797 -0.06797 -0.06797 -0.06790 -0.06787 -0.06787 -0.06778
##  [71] -0.06775 -0.06772 -0.06755 -0.06752 -0.06752 -0.06750 -0.06748
##  [78] -0.06748 -0.06746 -0.06744 -0.06743 -0.06743 -0.06731 -0.06725
##  [85] -0.06723 -0.06723 -0.06721 -0.06715 -0.06713 -0.06710 -0.06710
##  [92] -0.06709 -0.06704 -0.06704 -0.06703 -0.06703 -0.06703 -0.06703
##  [99] -0.06695 -0.06689 -0.06688 -0.06688 -0.06687 -0.06687 -0.06685
## [106] -0.06680 -0.06675 -0.06670 -0.06669 -0.06662 -0.06658 -0.06512
```

```
## [113] -0.06443 -0.06388 -0.06340 -0.06317 -0.06315 -0.06312 -0.06263
## [120] -0.06246 -0.06235 -0.06222 -0.06208 -0.06185
##
## $TP
##   [1] 1.000000 0.992165 0.984240 0.976194 0.968148 0.960100 0.952051
##   [8] 0.944000 0.935949 0.927898 0.919846 0.911794 0.903741 0.895686
##  [15] 0.887632 0.879577 0.871522 0.863466 0.855409 0.847353 0.839297
##  [22] 0.831240 0.823183 0.815125 0.807068 0.799009 0.790951 0.782893
##  [29] 0.774834 0.766775 0.758716 0.750657 0.742598 0.734539 0.726480
##  [36] 0.718420 0.710361 0.702301 0.694242 0.686182 0.678122 0.670062
##  [43] 0.662002 0.653942 0.645880 0.637819 0.629758 0.621696 0.613634
##  [50] 0.605573 0.597511 0.589449 0.581387 0.573325 0.565263 0.557201
##  [57] 0.549139 0.541077 0.533014 0.524952 0.516889 0.508826 0.500762
##  [64] 0.492698 0.484635 0.476571 0.468507 0.460442 0.452377 0.444312
##  [71] 0.436247 0.428181 0.420115 0.412048 0.403980 0.395912 0.387845
##  [78] 0.379777 0.371709 0.363641 0.355572 0.347504 0.339436 0.331366
##  [85] 0.323296 0.315226 0.307156 0.299086 0.291016 0.282945 0.274874
##  [92] 0.266803 0.258732 0.250660 0.242589 0.234517 0.226446 0.218374
##  [99] 0.210303 0.202230 0.194158 0.186085 0.178012 0.169939 0.161866
## [106] 0.153793 0.145720 0.137646 0.129572 0.121498 0.113423 0.105347
## [113] 0.097260 0.089168 0.081071 0.072970 0.064867 0.056764 0.048661
## [120] 0.040554 0.032445 0.024336 0.016225 0.008114 0.000000 0.000000
##
## $FP
##   [1] 1.000000 0.991935 0.983871 0.975806 0.967742 0.959677 0.951613
##   [8] 0.943548 0.935484 0.927419 0.919355 0.911290 0.903226 0.895161
##  [15] 0.887097 0.879032 0.870968 0.862903 0.854839 0.846774 0.838710
##  [22] 0.830645 0.822581 0.814516 0.806452 0.798387 0.790323 0.782258
##  [29] 0.774194 0.766129 0.758065 0.750000 0.741935 0.733871 0.725806
##  [36] 0.717742 0.709677 0.701613 0.693548 0.685484 0.677419 0.669355
##  [43] 0.661290 0.653226 0.645161 0.637097 0.629032 0.620968 0.612903
##  [50] 0.604839 0.596774 0.588710 0.580645 0.572581 0.564516 0.556452
##  [57] 0.548387 0.540323 0.532258 0.524194 0.516129 0.508065 0.500000
##  [64] 0.491935 0.483871 0.475806 0.467742 0.459677 0.451613 0.443548
##  [71] 0.435484 0.427419 0.419355 0.411290 0.403226 0.395161 0.387097
##  [78] 0.379032 0.370968 0.362903 0.354839 0.346774 0.338710 0.330645
##  [85] 0.322581 0.314516 0.306452 0.298387 0.290323 0.282258 0.274194
##  [92] 0.266129 0.258065 0.250000 0.241935 0.233871 0.225806 0.217742
##  [99] 0.209677 0.201613 0.193548 0.185484 0.177419 0.169355 0.161290
## [106] 0.153226 0.145161 0.137097 0.129032 0.120968 0.112903 0.104839
## [113] 0.096774 0.088710 0.080645 0.072581 0.064516 0.056452 0.048387
## [120] 0.040323 0.032258 0.024194 0.016129 0.008065 0.000000 0.000000
##
## $AUC
## [1] 0.5006

risksetROC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = mskcc_post.linpred.glasgow,
```



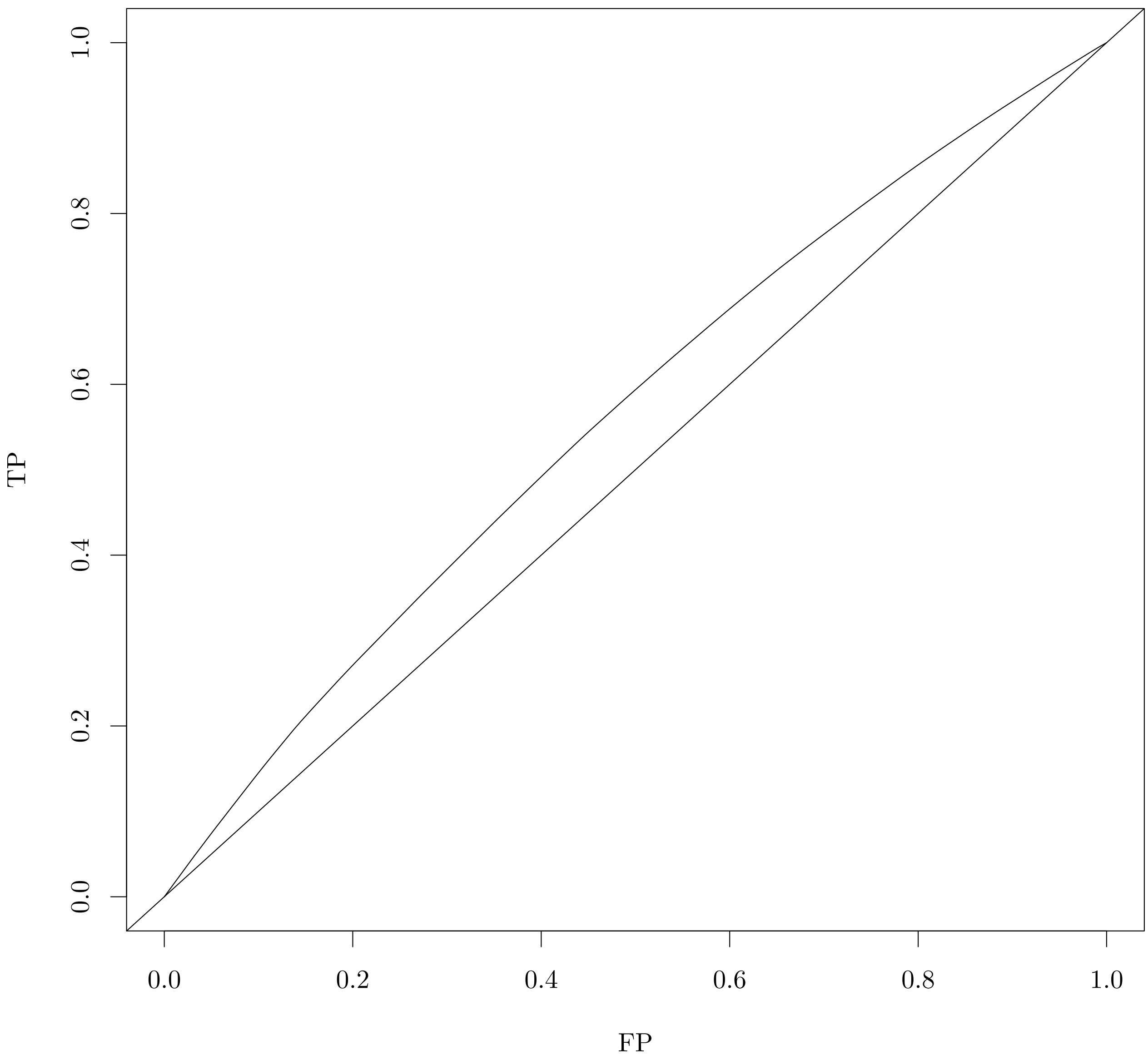

```
## $marker
##   [1] 1.734 1.808 1.847 1.877 1.890 1.899 1.901 1.933 1.945 1.953 1.955
##  [12] 1.980 1.984 1.990 2.001 2.009 2.009 2.012 2.016 2.032 2.033 2.086
##  [23] 2.099 2.113 2.136 2.152 2.165 2.182 2.208 2.210 2.224 2.225 2.227
##  [34] 2.229 2.233 2.240 2.245 2.248 2.252 2.259 2.261 2.286 2.295 2.320
##  [45] 2.324 2.331 2.335 2.337 2.341 2.341 2.342 2.347 2.348 2.355 2.379
##  [56] 2.379 2.382 2.384 2.388 2.403 2.404 2.415 2.425 2.426 2.427 2.437
##  [67] 2.451 2.464 2.471 2.474 2.477 2.481 2.485 2.491 2.493 2.495 2.496
##  [78] 2.499 2.515 2.515 2.515 2.521 2.524 2.524 2.527 2.527 2.529 2.531
##  [89] 2.533 2.538 2.541 2.545 2.548 2.548 2.555 2.558 2.564 2.567 2.572
## [100] 2.572 2.604 2.650 2.656 2.656 2.669 2.679 2.685 2.710 2.711 2.714
## [111] 2.717 2.718 2.721 2.726 2.742 2.766 2.779 2.806 2.850 2.860 2.883
## [122] 2.884 2.895 2.938
##
## $TP
##   [1] 1.00000 0.99594 0.99156 0.98701 0.98232 0.97757 0.97278 0.96798
##   [9] 0.96302 0.95801 0.95295 0.94788 0.94269 0.93747 0.93222 0.92691
```

```
##  [17] 0.92156 0.91621 0.91085 0.90546 0.89999 0.89451 0.88873 0.88288
##  [25] 0.87695 0.87087 0.86471 0.85845 0.85209 0.84557 0.83903 0.83240
##  [33] 0.82576 0.81911 0.81244 0.80575 0.79901 0.79224 0.78544 0.77862
##  [41] 0.77175 0.76487 0.75782 0.75070 0.74340 0.73607 0.72869 0.72127
##  [49] 0.71385 0.70640 0.69894 0.69148 0.68398 0.67648 0.66892 0.66118
##  [57] 0.65343 0.64566 0.63788 0.63007 0.62214 0.61419 0.60616 0.59806
##  [65] 0.58994 0.58181 0.57361 0.56529 0.55686 0.54837 0.53986 0.53132
##  [73] 0.52275 0.51413 0.50547 0.49679 0.48810 0.47939 0.47066 0.46179
##  [81] 0.45292 0.44405 0.43513 0.42618 0.41723 0.40825 0.39927 0.39027
##  [89] 0.38126 0.37223 0.36315 0.35404 0.34490 0.33573 0.32657 0.31734
##  [97] 0.30808 0.29875 0.28940 0.28001 0.27062 0.26092 0.25077 0.24056
## [105] 0.23035 0.22000 0.20955 0.19903 0.18825 0.17745 0.16663 0.15577
## [113] 0.14490 0.13400 0.12305 0.11191 0.10051 0.08895 0.07709 0.06469
## [121] 0.05217 0.03935 0.02651 0.01354 0.00000 0.00000
##
## $FP
##   [1] 1.000000 0.991935 0.983871 0.975806 0.967742 0.959677 0.951613
##   [8] 0.943548 0.935484 0.927419 0.919355 0.911290 0.903226 0.895161
##  [15] 0.887097 0.879032 0.870968 0.862903 0.854839 0.846774 0.838710
##  [22] 0.830645 0.822581 0.814516 0.806452 0.798387 0.790323 0.782258
##  [29] 0.774194 0.766129 0.758065 0.750000 0.741935 0.733871 0.725806
##  [36] 0.717742 0.709677 0.701613 0.693548 0.685484 0.677419 0.669355
##  [43] 0.661290 0.653226 0.645161 0.637097 0.629032 0.620968 0.612903
##  [50] 0.604839 0.596774 0.588710 0.580645 0.572581 0.564516 0.556452
##  [57] 0.548387 0.540323 0.532258 0.524194 0.516129 0.508065 0.500000
##  [64] 0.491935 0.483871 0.475806 0.467742 0.459677 0.451613 0.443548
##  [71] 0.435484 0.427419 0.419355 0.411290 0.403226 0.395161 0.387097
##  [78] 0.379032 0.370968 0.362903 0.354839 0.346774 0.338710 0.330645
##  [85] 0.322581 0.314516 0.306452 0.298387 0.290323 0.282258 0.274194
##  [92] 0.266129 0.258065 0.250000 0.241935 0.233871 0.225806 0.217742
##  [99] 0.209677 0.201613 0.193548 0.185484 0.177419 0.169355 0.161290
## [106] 0.153226 0.145161 0.137097 0.129032 0.120968 0.112903 0.104839
## [113] 0.096774 0.088710 0.080645 0.072581 0.064516 0.056452 0.048387
## [120] 0.040323 0.032258 0.024194 0.016129 0.008065 0.000000 0.000000
##
## $AUC
## [1] 0.5743

risksetROC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = gg.linpred.glasgow, predict.
```



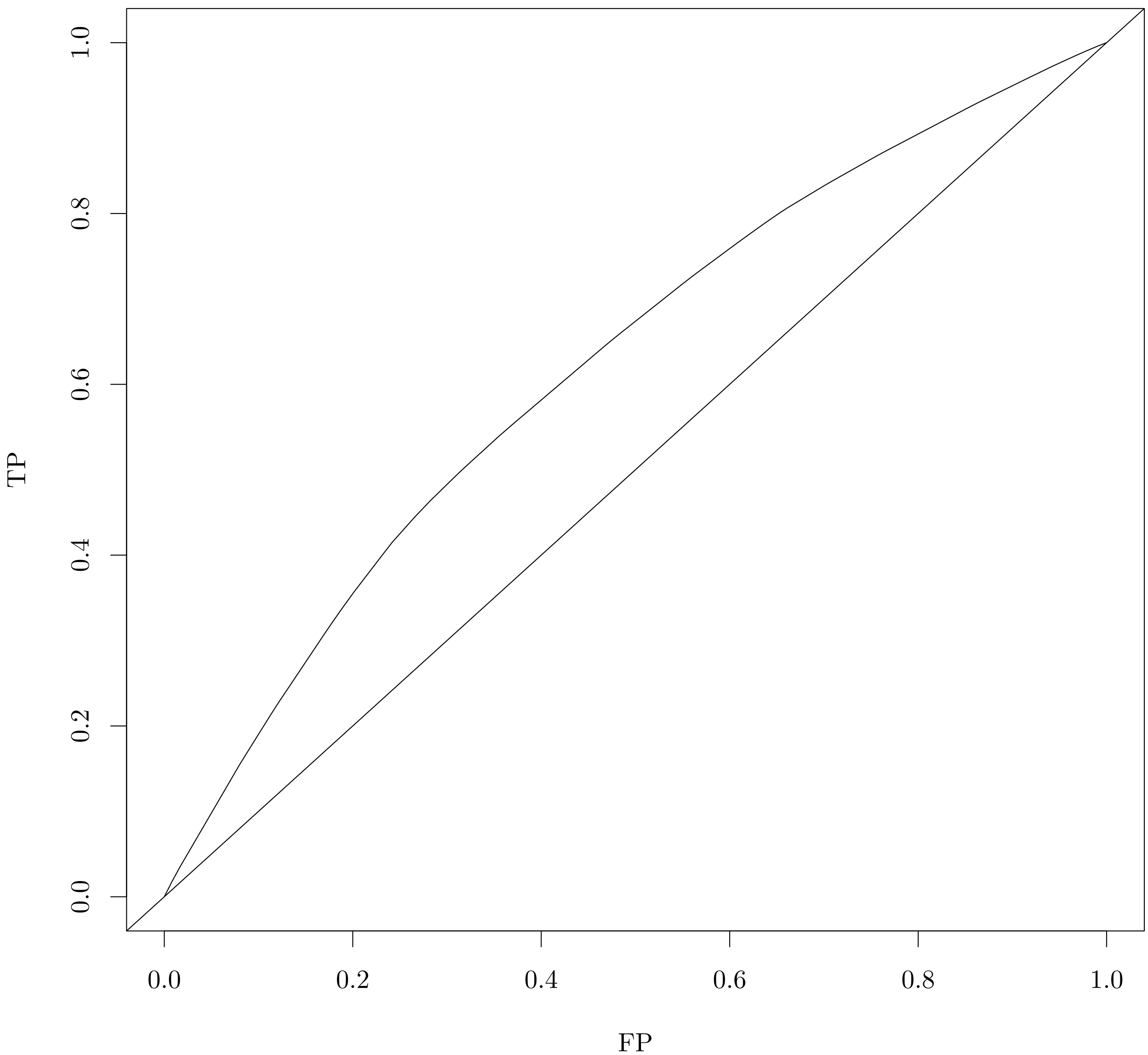

```
## $marker
##   [1] -0.467684 -0.371761 -0.346182 -0.346182 -0.346182 -0.339787 -0.339787
##   [8] -0.307812 -0.307812 -0.307812 -0.307812 -0.307812 -0.307812 -0.307812
##  [15] -0.275838 -0.275838 -0.263048 -0.243864 -0.243864 -0.243864 -0.243864
##  [22] -0.211889 -0.206711 -0.147941 -0.142762 -0.142762 -0.110788 -0.097998
##  [29] -0.095923 -0.095923 -0.083133 -0.078814 -0.078814 -0.078814 -0.078814
##  [36] -0.072419 -0.063949 -0.063949 -0.046839 -0.046839 -0.046839 -0.046839
##  [43] -0.046839 -0.040444 -0.034049 -0.031974 -0.031974 -0.031974 -0.031974
##  [50] -0.025580 -0.014865 -0.014865 -0.014865 -0.012790  0.000000  0.000000
##  [57]  0.000000  0.000000  0.006395  0.031974  0.031974  0.031974  0.031974
##  [64]  0.049084  0.049084  0.081058  0.081058  0.140687  0.153779  0.158957
##  [71]  0.184235  0.184235  0.190630  0.197024  0.197024  0.197024  0.203721
##  [78]  0.222604  0.222906  0.222906  0.228999  0.228999  0.228999  0.228999
##  [85]  0.248184  0.254880  0.254880  0.260973  0.260973  0.260973  0.260973
##  [92]  0.260973  0.260973  0.260973  0.269745  0.286855  0.286855  0.286855
##  [99]  0.286855  0.288930  0.292948  0.318829  0.324922  0.324922  0.324922
## [106]  0.350803  0.388871  0.429617  0.452820  0.466770  0.466770  0.492349
```

```
## [113]  0.524324  0.530718  0.530718  0.530718  0.530718  0.549903  0.562693
## [120]  0.562693  0.594667  0.594667  0.594667  0.594667
##
## $TP
##   [1] 1.00000 0.99552 0.99060 0.98554 0.98049 0.97543 0.97035 0.96526
##   [9] 0.96001 0.95476 0.94950 0.94425 0.93900 0.93375 0.92849 0.92307
##  [17] 0.91765 0.91216 0.90656 0.90096 0.89536 0.88976 0.88398 0.87817
##  [25] 0.87201 0.86581 0.85962 0.85322 0.84674 0.84025 0.83376 0.82718
##  [33] 0.82058 0.81398 0.80737 0.80077 0.79412 0.78742 0.78072 0.77390
##  [41] 0.76708 0.76026 0.75344 0.74662 0.73976 0.73286 0.72594 0.71902
##  [49] 0.71209 0.70517 0.69821 0.69117 0.68413 0.67709 0.67004 0.66289
##  [57] 0.65574 0.64860 0.64145 0.63426 0.62689 0.61951 0.61213 0.60475
##  [65] 0.59725 0.58974 0.58199 0.57425 0.56602 0.55769 0.54931 0.54072
##  [73] 0.53213 0.52348 0.51478 0.50608 0.49738 0.48862 0.47969 0.47076
##  [81] 0.46183 0.45285 0.44387 0.43488 0.42590 0.41674 0.40752 0.39830
##  [89] 0.38902 0.37975 0.37047 0.36120 0.35192 0.34264 0.33337 0.32401
##  [97] 0.31449 0.30497 0.29545 0.28593 0.27639 0.26682 0.25699 0.24710
## [105] 0.23721 0.22732 0.21718 0.20663 0.19565 0.18442 0.17302 0.16162
## [113] 0.14993 0.13786 0.12572 0.11357 0.10142 0.08927 0.07689 0.06434
## [121] 0.05180 0.03885 0.02590 0.01295 0.00000 0.00000
##
## $FP
##   [1] 1.000000 0.991935 0.983871 0.975806 0.967742 0.959677 0.951613
##   [8] 0.943548 0.935484 0.927419 0.919355 0.911290 0.903226 0.895161
##  [15] 0.887097 0.879032 0.870968 0.862903 0.854839 0.846774 0.838710
##  [22] 0.830645 0.822581 0.814516 0.806452 0.798387 0.790323 0.782258
##  [29] 0.774194 0.766129 0.758065 0.750000 0.741935 0.733871 0.725806
##  [36] 0.717742 0.709677 0.701613 0.693548 0.685484 0.677419 0.669355
##  [43] 0.661290 0.653226 0.645161 0.637097 0.629032 0.620968 0.612903
##  [50] 0.604839 0.596774 0.588710 0.580645 0.572581 0.564516 0.556452
##  [57] 0.548387 0.540323 0.532258 0.524194 0.516129 0.508065 0.500000
##  [64] 0.491935 0.483871 0.475806 0.467742 0.459677 0.451613 0.443548
##  [71] 0.435484 0.427419 0.419355 0.411290 0.403226 0.395161 0.387097
##  [78] 0.379032 0.370968 0.362903 0.354839 0.346774 0.338710 0.330645
##  [85] 0.322581 0.314516 0.306452 0.298387 0.290323 0.282258 0.274194
##  [92] 0.266129 0.258065 0.250000 0.241935 0.233871 0.225806 0.217742
##  [99] 0.209677 0.201613 0.193548 0.185484 0.177419 0.169355 0.161290
## [106] 0.153226 0.145161 0.137097 0.129032 0.120968 0.112903 0.104839
## [113] 0.096774 0.088710 0.080645 0.072581 0.064516 0.056452 0.048387
## [120] 0.040323 0.032258 0.024194 0.016129 0.008065 0.000000 0.000000
##
## $AUC
## [1] 0.5762

risksetROC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = gg2.linpred.glasgow, predict
```




```
## $marker
##   [1] -0.450187 -0.365769 -0.343257 -0.343257 -0.343257 -0.337630 -0.337630
##   [8] -0.309490 -0.309490 -0.309490 -0.309490 -0.309490 -0.309490 -0.309490
##  [15] -0.292441 -0.281351 -0.281351 -0.270095 -0.253212 -0.253212 -0.253212
##  [22] -0.253212 -0.236163 -0.236163 -0.225072 -0.208024 -0.196768 -0.179884
##  [29] -0.179884 -0.179884 -0.179884 -0.174256 -0.168794 -0.151745 -0.151745
##  [36] -0.151745 -0.151745 -0.151745 -0.146117 -0.140489 -0.123606 -0.123606
##  [43] -0.123606 -0.084418 -0.084418 -0.073162 -0.067327 -0.067327 -0.056279
##  [50] -0.056279 -0.039188 -0.039188 -0.028139 -0.028139 -0.028139 -0.028139
##  [57] -0.022511 -0.011256  0.000000  0.000000  0.000000  0.000000  0.005628
##  [64]  0.018497  0.028139  0.028139  0.028139  0.028139  0.057892  0.074776
##  [71]  0.074776  0.085866  0.090211  0.090211  0.095839  0.101467  0.101467
##  [78]  0.101467  0.102915  0.102915  0.123813  0.123978  0.129606  0.129606
##  [85]  0.129606  0.129606  0.131055  0.131055  0.131055  0.131055  0.146490
##  [92]  0.157745  0.157745  0.157745  0.157745  0.157745  0.157745  0.157745
##  [99]  0.159194  0.185885  0.187333  0.214024  0.214024  0.214024  0.226521
## [106]  0.243405  0.270302  0.326581  0.327988  0.327988  0.350499  0.367217
```

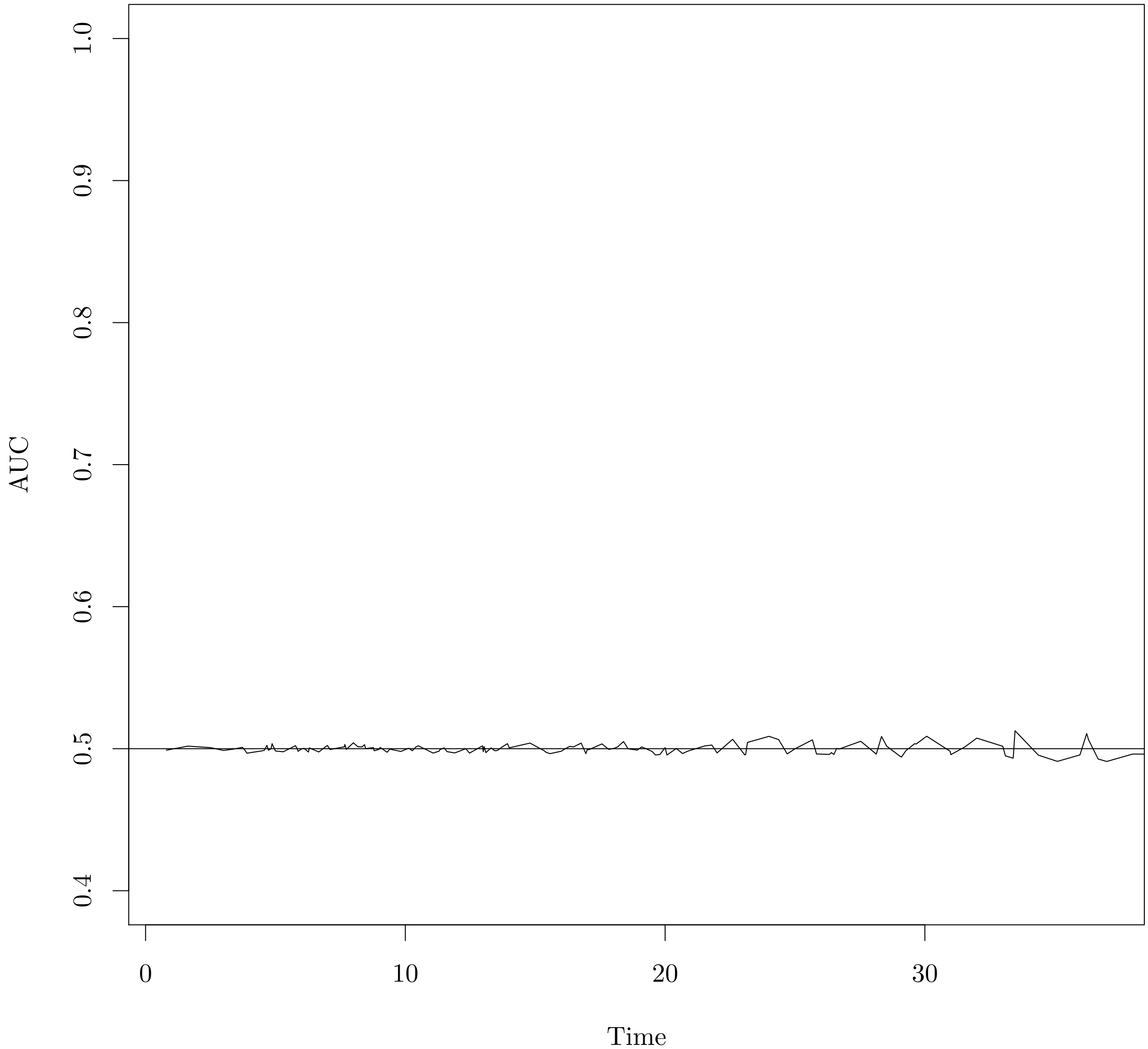
```
## [113]  0.378638  0.384266  0.384266  0.384266  0.384266  0.401150  0.412406
## [120]  0.412406  0.440545  0.440545  0.440545  0.440545
##
## $TP
##   [1] 1.00000 0.99504 0.98963 0.98411 0.97858 0.97306 0.96750 0.96194
##   [9] 0.95623 0.95051 0.94480 0.93908 0.93337 0.92765 0.92194 0.91612
##  [17] 0.91025 0.90437 0.89842 0.89238 0.88633 0.88029 0.87424 0.86809
##  [25] 0.86194 0.85572 0.84940 0.84300 0.83649 0.82999 0.82348 0.81698
##  [33] 0.81043 0.80385 0.79716 0.79047 0.78378 0.77709 0.77040 0.76367
##  [41] 0.75690 0.75002 0.74313 0.73625 0.72909 0.72194 0.71470 0.70742
##  [49] 0.70014 0.69277 0.68541 0.67792 0.67044 0.66286 0.65529 0.64772
##  [57] 0.64015 0.63253 0.62483 0.61704 0.60926 0.60147 0.59368 0.58585
##  [65] 0.57791 0.56990 0.56189 0.55388 0.54587 0.53762 0.52923 0.52083
##  [73] 0.51235 0.50382 0.49530 0.48673 0.47811 0.46949 0.46087 0.45224
##  [81] 0.44361 0.43479 0.42597 0.41711 0.40824 0.39938 0.39051 0.38163
##  [89] 0.37275 0.36388 0.35500 0.34598 0.33686 0.32774 0.31862 0.30950
##  [97] 0.30039 0.29127 0.28215 0.27302 0.26364 0.25424 0.24460 0.23495
## [105] 0.22530 0.21554 0.20560 0.19540 0.18460 0.17379 0.16298 0.15192
## [113] 0.14068 0.12930 0.11787 0.10643 0.09499 0.08356 0.07192 0.06016
## [121] 0.04840 0.03630 0.02420 0.01210 0.00000 0.00000
##
## $FP
##   [1] 1.000000 0.991935 0.983871 0.975806 0.967742 0.959677 0.951613
##   [8] 0.943548 0.935484 0.927419 0.919355 0.911290 0.903226 0.895161
##  [15] 0.887097 0.879032 0.870968 0.862903 0.854839 0.846774 0.838710
##  [22] 0.830645 0.822581 0.814516 0.806452 0.798387 0.790323 0.782258
##  [29] 0.774194 0.766129 0.758065 0.750000 0.741935 0.733871 0.725806
##  [36] 0.717742 0.709677 0.701613 0.693548 0.685484 0.677419 0.669355
##  [43] 0.661290 0.653226 0.645161 0.637097 0.629032 0.620968 0.612903
##  [50] 0.604839 0.596774 0.588710 0.580645 0.572581 0.564516 0.556452
##  [57] 0.548387 0.540323 0.532258 0.524194 0.516129 0.508065 0.500000
##  [64] 0.491935 0.483871 0.475806 0.467742 0.459677 0.451613 0.443548
##  [71] 0.435484 0.427419 0.419355 0.411290 0.403226 0.395161 0.387097
##  [78] 0.379032 0.370968 0.362903 0.354839 0.346774 0.338710 0.330645
##  [85] 0.322581 0.314516 0.306452 0.298387 0.290323 0.282258 0.274194
##  [92] 0.266129 0.258065 0.250000 0.241935 0.233871 0.225806 0.217742
##  [99] 0.209677 0.201613 0.193548 0.185484 0.177419 0.169355 0.161290
## [106] 0.153226 0.145161 0.137097 0.129032 0.120968 0.112903 0.104839
## [113] 0.096774 0.088710 0.080645 0.072581 0.064516 0.056452 0.048387
## [120] 0.040323 0.032258 0.024194 0.016129 0.008065 0.000000 0.000000
##
## $AUC
## [1] 0.5645

risksetROC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = cph.linpred.glasgow, predict
```




```
## $marker
##   [1] -0.34542 -0.20725 -0.20725 -0.17962 -0.13817 -0.13817 -0.13817
##   [8] -0.08290 -0.08290 -0.08290 -0.06908 -0.06908 -0.06908 -0.06908
##  [15] -0.06908 -0.06908 -0.05527 -0.02763  0.00000  0.00000  0.00000
##  [22]  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000
##  [29]  0.00000  0.01382  0.06908  0.06908  0.06908  0.06908  0.06908
##  [36]  0.06908  0.09672  0.13540  0.13817  0.13817  0.13817  0.13817
##  [43]  0.20725  0.27357  0.27357  0.30397  0.31502  0.31502  0.32883
##  [50]  0.34265  0.34265  0.34265  0.34265  0.34542  0.37028  0.39792
##  [57]  0.41173  0.41173  0.41173  0.41173  0.41173  0.41173  0.41173
##  [64]  0.41173  0.42555  0.45318  0.48082  0.48082  0.48082  0.48082
##  [71]  0.48082  0.48082  0.48082  0.48082  0.48082  0.48082  0.48082
##  [78]  0.48082  0.49463  0.50845  0.54990  0.54990  0.54990  0.54990
##  [85]  0.56413  0.60558  0.61898  0.61898  0.61898  0.68807  0.68807
##  [92]  0.75715  0.75715  0.75715  0.89532  0.90678  0.90678  0.90678
##  [99]  0.90955  0.96205  0.97863  1.00350  1.03113  1.04495  1.04495
## [106]  1.04495  1.04495  1.04495  1.04495  1.08640  1.11403  1.11403
```

```
## [113]  1.11403  1.11403  1.18311  1.18311  1.18311  1.18311  1.18311
## [120]  1.18311  1.18311  1.18311  1.25220  1.32128
##
## $TP
##   [1] 1.00000 0.99669 0.99289 0.98908 0.98518 0.98110 0.97703 0.97295
##   [9] 0.96865 0.96434 0.96004 0.95567 0.95130 0.94694 0.94257 0.93821
##  [17] 0.93384 0.92942 0.92487 0.92019 0.91551 0.91083 0.90615 0.90148
##  [25] 0.89680 0.89212 0.88744 0.88277 0.87809 0.87341 0.86867 0.86365
##  [33] 0.85864 0.85363 0.84862 0.84361 0.83859 0.83344 0.82808 0.82271
##  [41] 0.81734 0.81197 0.80660 0.80085 0.79470 0.78855 0.78221 0.77580
##  [49] 0.76939 0.76289 0.75630 0.74971 0.74312 0.73653 0.72992 0.72315
##  [57] 0.71618 0.70912 0.70206 0.69500 0.68794 0.68088 0.67382 0.66676
##  [65] 0.65970 0.65254 0.64518 0.63761 0.63005 0.62248 0.61492 0.60735
##  [73] 0.59978 0.59222 0.58465 0.57709 0.56952 0.56195 0.55439 0.54672
##  [81] 0.53894 0.53083 0.52273 0.51462 0.50651 0.49829 0.48972 0.48103
##  [89] 0.47234 0.46366 0.45435 0.44504 0.43507 0.42509 0.41512 0.40367
##  [97] 0.39208 0.38050 0.36892 0.35730 0.34506 0.33261 0.31985 0.30673
## [105] 0.29343 0.28013 0.26683 0.25353 0.24023 0.22693 0.21307 0.19882
## [113] 0.18457 0.17031 0.15606 0.14079 0.12552 0.11025 0.09498 0.07971
## [121] 0.06444 0.04917 0.03390 0.01753 0.00000 0.00000
##
## $FP
##   [1] 1.000000 0.991935 0.983871 0.975806 0.967742 0.959677 0.951613
##   [8] 0.943548 0.935484 0.927419 0.919355 0.911290 0.903226 0.895161
##  [15] 0.887097 0.879032 0.870968 0.862903 0.854839 0.846774 0.838710
##  [22] 0.830645 0.822581 0.814516 0.806452 0.798387 0.790323 0.782258
##  [29] 0.774194 0.766129 0.758065 0.750000 0.741935 0.733871 0.725806
##  [36] 0.717742 0.709677 0.701613 0.693548 0.685484 0.677419 0.669355
##  [43] 0.661290 0.653226 0.645161 0.637097 0.629032 0.620968 0.612903
##  [50] 0.604839 0.596774 0.588710 0.580645 0.572581 0.564516 0.556452
##  [57] 0.548387 0.540323 0.532258 0.524194 0.516129 0.508065 0.500000
##  [64] 0.491935 0.483871 0.475806 0.467742 0.459677 0.451613 0.443548
##  [71] 0.435484 0.427419 0.419355 0.411290 0.403226 0.395161 0.387097
##  [78] 0.379032 0.370968 0.362903 0.354839 0.346774 0.338710 0.330645
##  [85] 0.322581 0.314516 0.306452 0.298387 0.290323 0.282258 0.274194
##  [92] 0.266129 0.258065 0.250000 0.241935 0.233871 0.225806 0.217742
##  [99] 0.209677 0.201613 0.193548 0.185484 0.177419 0.169355 0.161290
## [106] 0.153226 0.145161 0.137097 0.129032 0.120968 0.112903 0.104839
## [113] 0.096774 0.088710 0.080645 0.072581 0.064516 0.056452 0.048387
## [120] 0.040323 0.032258 0.024194 0.016129 0.008065 0.000000 0.000000
##
## $AUC
## [1] 0.6232
```

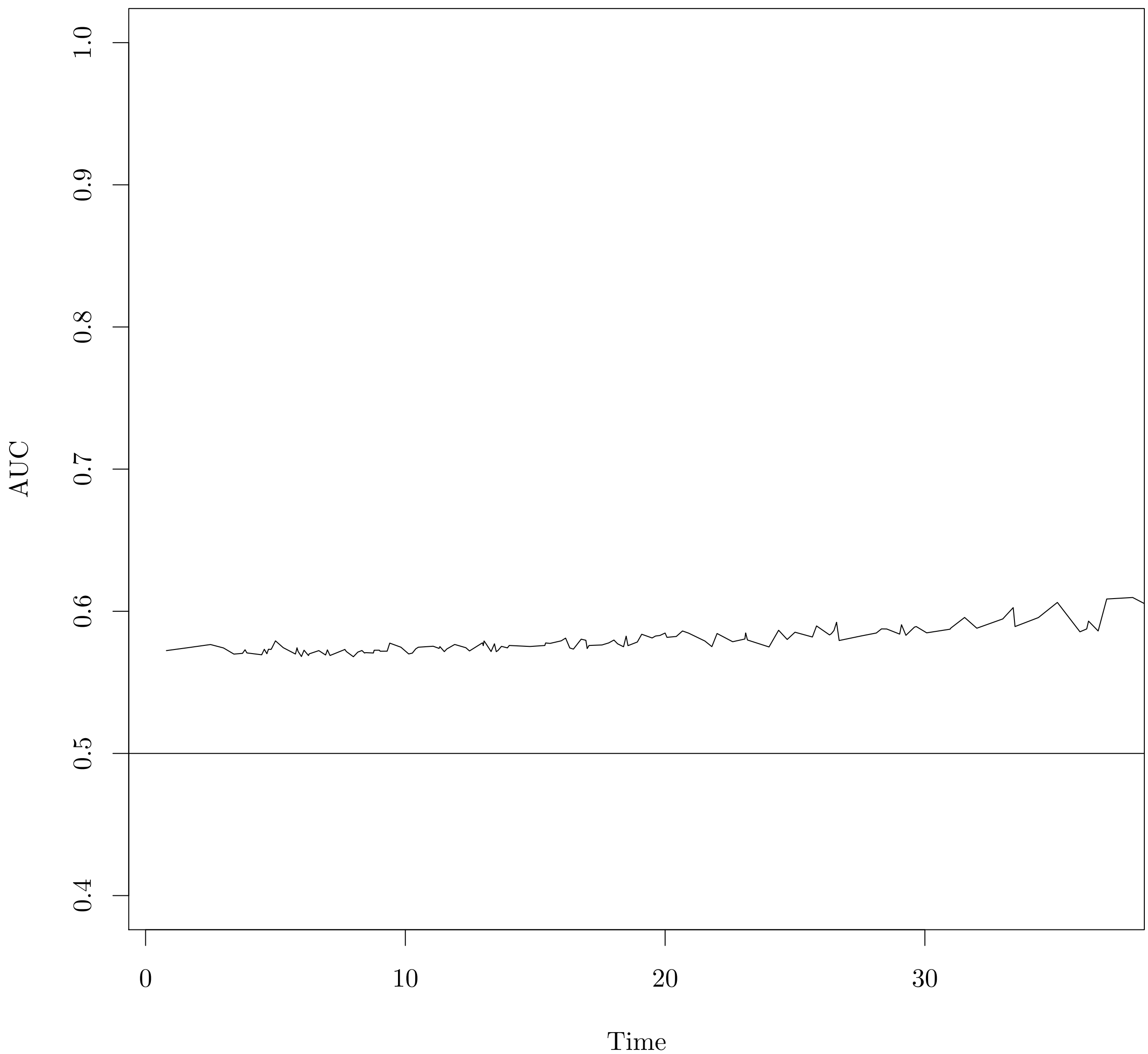
```r
risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = mskcc_pre.linpred.glasgow, t
```




```
## $utimes
##   [1]   0.80   1.63   2.50   3.00   3.40   3.73   3.83   3.90   4.47   4.57
##  [11]   4.67   4.73   4.83   4.87   5.00   5.30   5.77   5.83   5.87   6.00
##  [21]   6.10   6.27   6.30   6.67   6.93   7.00   7.10   7.66   7.67   7.73
##  [31]   8.00   8.17   8.33   8.43   8.47   8.77   8.80   9.00   9.03   9.30
##  [41]   9.40   9.83  10.13  10.27  10.40  10.50  11.07  11.30  11.33  11.50
##  [51]  11.60  11.90  12.33  12.47  12.97  13.00  13.03  13.10  13.30  13.43
##  [61]  13.50  13.57  13.70  13.93  14.00  14.80  15.37  15.40  15.57  16.00
##  [71]  16.17  16.33  16.47  16.77  16.95  17.00  17.07  17.57  17.83  18.03
##  [81]  18.17  18.40  18.50  18.57  18.93  19.10  19.50  19.63  19.80  20.00
##  [91]  20.07  20.43  20.67  20.90  21.53  21.80  22.00  22.60  23.07  23.10
## [101]  23.17  24.00  24.37  24.70  25.00  25.67  25.83  26.33  26.40  26.50
## [111]  26.60  26.70  27.53  28.13  28.33  28.53  29.03  29.10  29.27  29.60
## [121]  29.67  30.07  30.97  31.00  31.53  32.00  33.00  33.10  33.40  33.47
## [131]  34.37  35.10  35.97  36.23  36.30  36.67  37.00  38.00  39.60  41.23
## [141]  43.07  45.37  46.67  47.43  47.73  48.00  49.00  51.00  54.90  59.00
## [151]  63.13  65.00  67.00  70.00  77.00  85.00  85.80  90.33  93.00  94.77
```

```
## [161] 116.00
##
## $St
##   [1] 0.99476 0.98930 0.98383 0.97834 0.97284 0.96734 0.96185 0.95086
##   [9] 0.93986 0.93437 0.92887 0.92337 0.91788 0.90689 0.89589 0.89040
##  [17] 0.88490 0.87940 0.87391 0.86841 0.86291 0.85742 0.85192 0.84643
##  [25] 0.84093 0.83543 0.82994 0.82444 0.81894 0.81345 0.80246 0.79696
##  [33] 0.79146 0.78597 0.78047 0.77497 0.76948 0.76398 0.75845 0.75291
##  [41] 0.74737 0.74184 0.73630 0.73077 0.72523 0.71969 0.71416 0.70862
##  [49] 0.70308 0.69755 0.69201 0.68648 0.68094 0.67540 0.66987 0.66433
##  [57] 0.65880 0.65326 0.64772 0.64219 0.63665 0.63112 0.62558 0.62004
##  [65] 0.61451 0.60892 0.60333 0.59775 0.59216 0.58658 0.58099 0.57540
##  [73] 0.56982 0.56423 0.55864 0.55306 0.54747 0.54188 0.53630 0.53071
##  [81] 0.52512 0.51954 0.51395 0.50837 0.50278 0.49719 0.49161 0.48602
##  [89] 0.48043 0.46926 0.46367 0.45809 0.45250 0.44691 0.44133 0.43574
##  [97] 0.43016 0.42457 0.41898 0.41340 0.40781 0.39664 0.39105 0.38546
## [105] 0.37429 0.36870 0.36312 0.35753 0.35195 0.34636 0.34077 0.33519
## [113] 0.32960 0.32401 0.31843 0.31284 0.30725 0.30167 0.29608 0.29049
## [121] 0.28491 0.27932 0.27374 0.26815 0.26256 0.25698 0.25139 0.24580
## [129] 0.24022 0.23463 0.22904 0.22332 0.21759 0.21187 0.20614 0.20041
## [137] 0.19469 0.18289 0.17679 0.17048 0.16416 0.15760 0.15103 0.14446
## [145] 0.13790 0.13133 0.12442 0.11751 0.10967 0.10184 0.09401 0.08617
## [153] 0.07834 0.07050 0.06169 0.05141 0.04113 0.03085 0.02056 0.01028
## [161] 0.00000
##
## $AUC
##   [1] 0.4989 0.5018 0.5007 0.4988 0.4998 0.5010 0.4988 0.4968 0.4984 0.4989
##  [11] 0.5023 0.4989 0.5000 0.5035 0.4984 0.4978 0.5021 0.5003 0.4981 0.4998
##  [21] 0.5003 0.4976 0.5006 0.4977 0.5014 0.5022 0.4994 0.5014 0.5030 0.4995
##  [31] 0.5041 0.5014 0.5013 0.5028 0.5001 0.5008 0.4985 0.4997 0.5009 0.4975
##  [41] 0.4997 0.4981 0.5004 0.4986 0.5012 0.5020 0.4969 0.4983 0.4993 0.5005
##  [51] 0.4980 0.4970 0.5000 0.4969 0.5020 0.4979 0.5014 0.4972 0.5006 0.4986
##  [61] 0.4986 0.4990 0.5009 0.5036 0.5006 0.5039 0.4983 0.4976 0.4965 0.4982
##  [71] 0.5001 0.5017 0.5012 0.5039 0.4965 0.4993 0.4994 0.5034 0.4995 0.5002
##  [81] 0.5013 0.5050 0.5021 0.4999 0.4990 0.5013 0.4980 0.4956 0.4959 0.5007
##  [91] 0.4955 0.5002 0.4966 0.4984 0.5020 0.5026 0.4970 0.5066 0.4956 0.4959
## [101] 0.5044 0.5087 0.5064 0.4963 0.4999 0.5062 0.4963 0.4960 0.4972 0.4960
## [111] 0.5002 0.4997 0.5052 0.4962 0.5086 0.5018 0.4949 0.4941 0.4986 0.5034
## [121] 0.5034 0.5088 0.4983 0.4958 0.5012 0.5074 0.5017 0.4949 0.4933 0.5127
## [131] 0.4955 0.4911 0.4956 0.5107 0.5063 0.4927 0.4910 0.4962 0.4960 0.4900
## [141] 0.4942 0.4897 0.4932 0.5232 0.5198 0.4847 0.5216 0.4930 0.5204 0.4821
## [151] 0.4758 0.5357 0.4924 0.4517 0.4380 0.4171 0.4504 0.6255 0.5000 0.7500
## [161] 0.0000
##
## $Cindex
## [1] 0.5
```

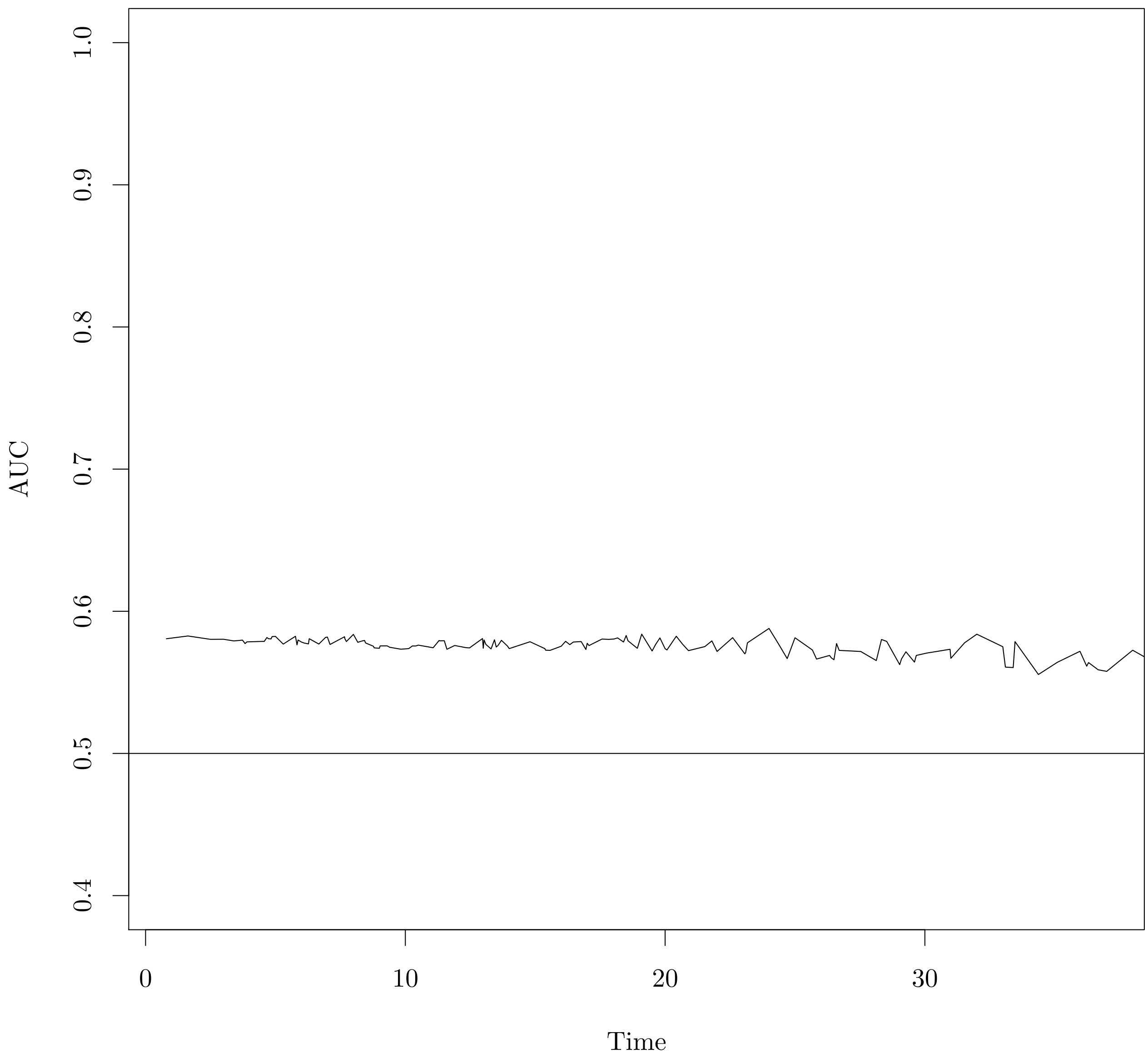
```
risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = mskcc_post.linpred.glasgow,
```




```
## $utimes
##   [1]  0.80  1.63  2.50  3.00  3.40  3.73  3.83  3.90  4.47  4.57
##  [11]  4.67  4.73  4.83  4.87  5.00  5.30  5.77  5.83  5.87  6.00
##  [21]  6.10  6.27  6.30  6.67  6.93  7.00  7.10  7.66  7.67  7.73
##  [31]  8.00  8.17  8.33  8.43  8.47  8.77  8.80  9.00  9.03  9.30
##  [41]  9.40  9.83 10.13 10.27 10.40 10.50 11.07 11.30 11.33 11.50
##  [51] 11.60 11.90 12.33 12.47 12.97 13.00 13.03 13.10 13.30 13.43
##  [61] 13.50 13.57 13.70 13.93 14.00 14.80 15.37 15.40 15.57 16.00
##  [71] 16.17 16.33 16.47 16.77 16.95 17.00 17.07 17.57 17.83 18.03
##  [81] 18.17 18.40 18.50 18.57 18.93 19.10 19.50 19.63 19.80 20.00
##  [91] 20.07 20.43 20.67 20.90 21.53 21.80 22.00 22.60 23.07 23.10
## [101] 23.17 24.00 24.37 24.70 25.00 25.67 25.83 26.33 26.40 26.50
## [111] 26.60 26.70 27.53 28.13 28.33 28.53 29.03 29.10 29.27 29.60
## [121] 29.67 30.07 30.97 31.00 31.53 32.00 33.00 33.10 33.40 33.47
## [131] 34.37 35.10 35.97 36.23 36.30 36.67 37.00 38.00 39.60 41.23
## [141] 43.07 45.37 46.67 47.43 47.73 48.00 49.00 51.00 54.90 59.00
## [151] 63.13 65.00 67.00 70.00 77.00 85.00 85.80 90.33 93.00 94.77
```

```
## [161] 116.00
##
## $St
##   [1] 0.99476 0.98930 0.98383 0.97834 0.97284 0.96734 0.96185 0.95086
##   [9] 0.93986 0.93437 0.92887 0.92337 0.91788 0.90689 0.89589 0.89040
##  [17] 0.88490 0.87940 0.87391 0.86841 0.86291 0.85742 0.85192 0.84643
##  [25] 0.84093 0.83543 0.82994 0.82444 0.81894 0.81345 0.80246 0.79696
##  [33] 0.79146 0.78597 0.78047 0.77497 0.76948 0.76398 0.75845 0.75291
##  [41] 0.74737 0.74184 0.73630 0.73077 0.72523 0.71969 0.71416 0.70862
##  [49] 0.70308 0.69755 0.69201 0.68648 0.68094 0.67540 0.66987 0.66433
##  [57] 0.65880 0.65326 0.64772 0.64219 0.63665 0.63112 0.62558 0.62004
##  [65] 0.61451 0.60892 0.60333 0.59775 0.59216 0.58658 0.58099 0.57540
##  [73] 0.56982 0.56423 0.55864 0.55306 0.54747 0.54188 0.53630 0.53071
##  [81] 0.52512 0.51954 0.51395 0.50837 0.50278 0.49719 0.49161 0.48602
##  [89] 0.48043 0.46926 0.46367 0.45809 0.45250 0.44691 0.44133 0.43574
##  [97] 0.43016 0.42457 0.41898 0.41340 0.40781 0.39664 0.39105 0.38546
## [105] 0.37429 0.36870 0.36312 0.35753 0.35195 0.34636 0.34077 0.33519
## [113] 0.32960 0.32401 0.31843 0.31284 0.30725 0.30167 0.29608 0.29049
## [121] 0.28491 0.27932 0.27374 0.26815 0.26256 0.25698 0.25139 0.24580
## [129] 0.24022 0.23463 0.22904 0.22332 0.21759 0.21187 0.20614 0.20041
## [137] 0.19469 0.18289 0.17679 0.17048 0.16416 0.15760 0.15103 0.14446
## [145] 0.13790 0.13133 0.12442 0.11751 0.10967 0.10184 0.09401 0.08617
## [153] 0.07834 0.07050 0.06169 0.05141 0.04113 0.03085 0.02056 0.01028
## [161] 0.00000
##
## $AUC
##   [1] 0.5723 0.5744 0.5766 0.5742 0.5699 0.5704 0.5729 0.5707 0.5694 0.5733
##  [11] 0.5701 0.5733 0.5732 0.5744 0.5792 0.5744 0.5699 0.5743 0.5718 0.5682
##  [21] 0.5726 0.5689 0.5701 0.5723 0.5692 0.5729 0.5689 0.5730 0.5732 0.5717
##  [31] 0.5680 0.5713 0.5724 0.5706 0.5709 0.5707 0.5726 0.5726 0.5719 0.5719
##  [41] 0.5776 0.5747 0.5700 0.5706 0.5736 0.5747 0.5754 0.5739 0.5752 0.5716
##  [51] 0.5735 0.5767 0.5744 0.5721 0.5778 0.5758 0.5791 0.5773 0.5717 0.5771
##  [61] 0.5716 0.5725 0.5753 0.5743 0.5760 0.5752 0.5759 0.5777 0.5774 0.5792
##  [71] 0.5811 0.5742 0.5734 0.5804 0.5796 0.5738 0.5759 0.5763 0.5778 0.5798
##  [81] 0.5771 0.5750 0.5825 0.5758 0.5784 0.5838 0.5812 0.5826 0.5830 0.5847
##  [91] 0.5817 0.5823 0.5862 0.5847 0.5791 0.5751 0.5844 0.5786 0.5805 0.5848
## [101] 0.5798 0.5749 0.5867 0.5802 0.5853 0.5819 0.5897 0.5834 0.5842 0.5865
## [111] 0.5923 0.5794 0.5826 0.5847 0.5876 0.5876 0.5839 0.5905 0.5831 0.5889
## [121] 0.5892 0.5848 0.5874 0.5882 0.5957 0.5881 0.5946 0.5966 0.6026 0.5892
## [131] 0.5956 0.6062 0.5856 0.5876 0.5931 0.5861 0.6086 0.6097 0.5945 0.5881
## [141] 0.6132 0.5807 0.5967 0.5913 0.5844 0.6111 0.5856 0.6234 0.6160 0.6236
## [151] 0.6229 0.5656 0.6111 0.6573 0.5917 0.5836 0.5309 0.4725 0.7439 0.2936
## [161] 0.0000
##
## $Cindex
## [1] 0.576
```

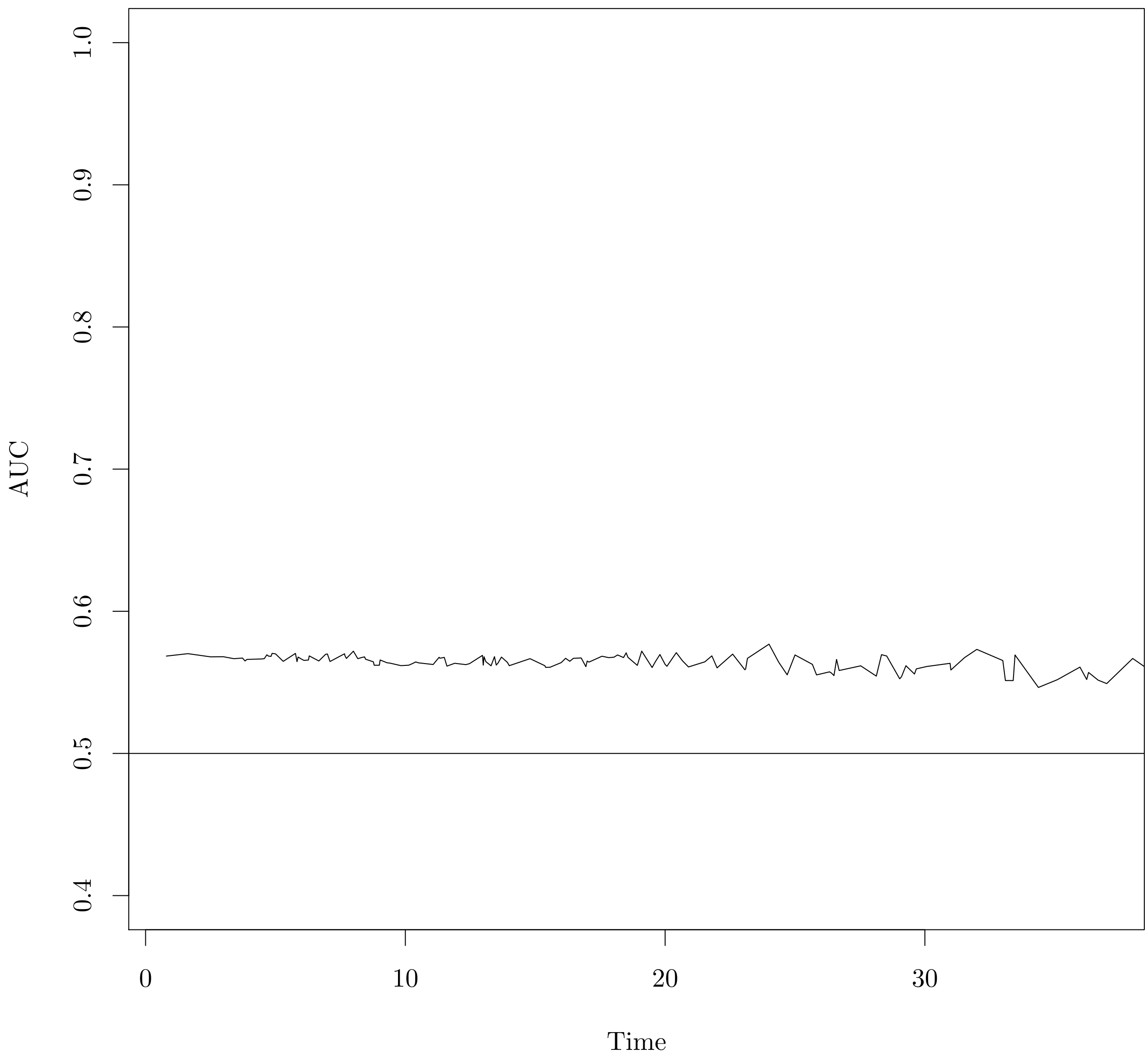
```r
risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = gg.linpred.glasgow, tmax = 3
```




```
## $utimes
##   [1]   0.80   1.63   2.50   3.00   3.40   3.73   3.83   3.90   4.47   4.57
##  [11]   4.67   4.73   4.83   4.87   5.00   5.30   5.77   5.83   5.87   6.00
##  [21]   6.10   6.27   6.30   6.67   6.93   7.00   7.10   7.66   7.67   7.73
##  [31]   8.00   8.17   8.33   8.43   8.47   8.77   8.80   9.00   9.03   9.30
##  [41]   9.40   9.83  10.13  10.27  10.40  10.50  11.07  11.30  11.33  11.50
##  [51]  11.60  11.90  12.33  12.47  12.97  13.00  13.03  13.10  13.30  13.43
##  [61]  13.50  13.57  13.70  13.93  14.00  14.80  15.37  15.40  15.57  16.00
##  [71]  16.17  16.33  16.47  16.77  16.95  17.00  17.07  17.57  17.83  18.03
##  [81]  18.17  18.40  18.50  18.57  18.93  19.10  19.50  19.63  19.80  20.00
##  [91]  20.07  20.43  20.67  20.90  21.53  21.80  22.00  22.60  23.07  23.10
## [101]  23.17  24.00  24.37  24.70  25.00  25.67  25.83  26.33  26.40  26.50
## [111]  26.60  26.70  27.53  28.13  28.33  28.53  29.03  29.10  29.27  29.60
## [121]  29.67  30.07  30.97  31.00  31.53  32.00  33.00  33.10  33.40  33.47
## [131]  34.37  35.10  35.97  36.23  36.30  36.67  37.00  38.00  39.60  41.23
## [141]  43.07  45.37  46.67  47.43  47.73  48.00  49.00  51.00  54.90  59.00
## [151]  63.13  65.00  67.00  70.00  77.00  85.00  85.80  90.33  93.00  94.77
```

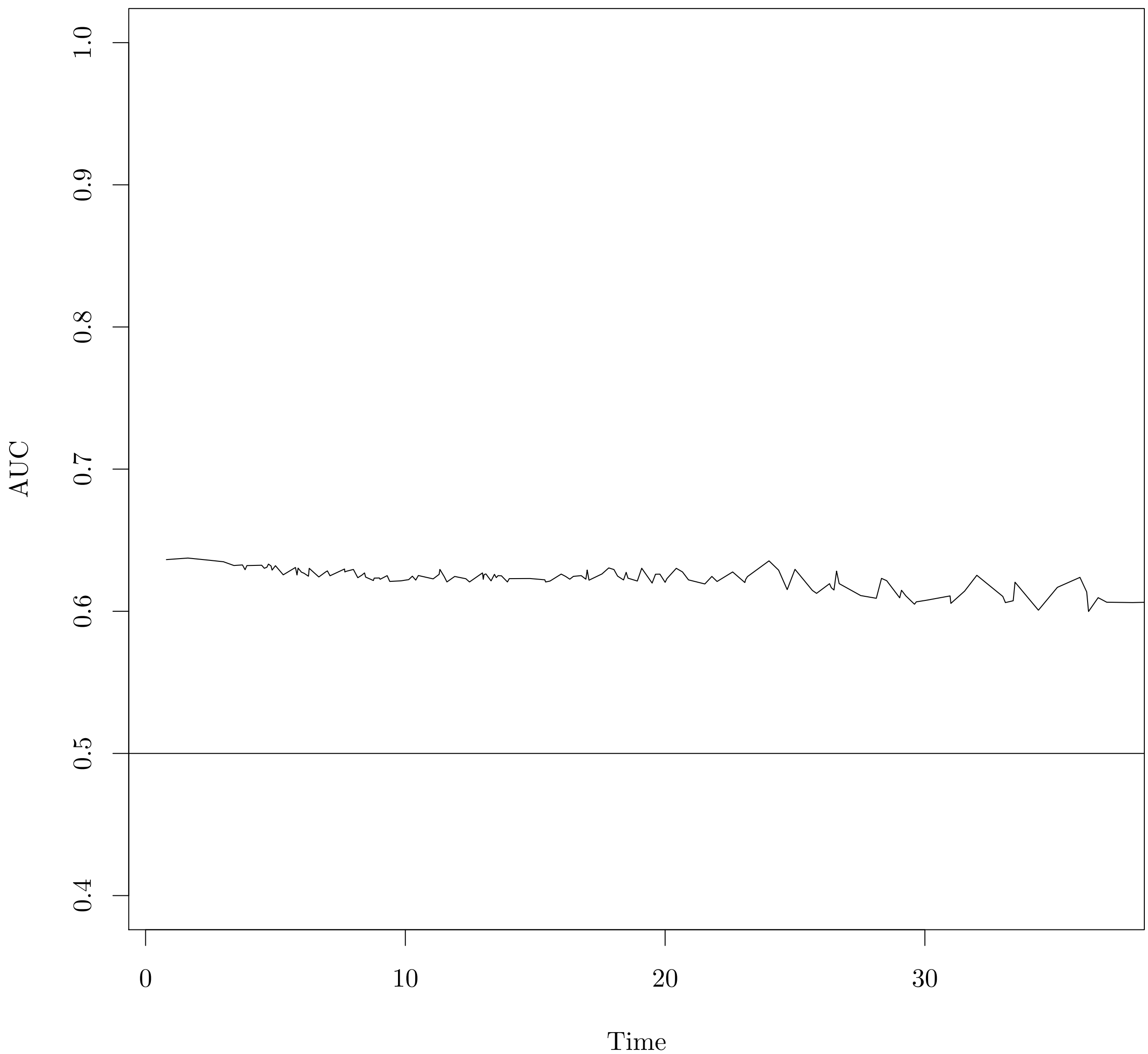
```
## [161] 116.00
##
## $St
##   [1] 0.99476 0.98930 0.98383 0.97834 0.97284 0.96734 0.96185 0.95086
##   [9] 0.93986 0.93437 0.92887 0.92337 0.91788 0.90689 0.89589 0.89040
##  [17] 0.88490 0.87940 0.87391 0.86841 0.86291 0.85742 0.85192 0.84643
##  [25] 0.84093 0.83543 0.82994 0.82444 0.81894 0.81345 0.80246 0.79696
##  [33] 0.79146 0.78597 0.78047 0.77497 0.76948 0.76398 0.75845 0.75291
##  [41] 0.74737 0.74184 0.73630 0.73077 0.72523 0.71969 0.71416 0.70862
##  [49] 0.70308 0.69755 0.69201 0.68648 0.68094 0.67540 0.66987 0.66433
##  [57] 0.65880 0.65326 0.64772 0.64219 0.63665 0.63112 0.62558 0.62004
##  [65] 0.61451 0.60892 0.60333 0.59775 0.59216 0.58658 0.58099 0.57540
##  [73] 0.56982 0.56423 0.55864 0.55306 0.54747 0.54188 0.53630 0.53071
##  [81] 0.52512 0.51954 0.51395 0.50837 0.50278 0.49719 0.49161 0.48602
##  [89] 0.48043 0.46926 0.46367 0.45809 0.45250 0.44691 0.44133 0.43574
##  [97] 0.43016 0.42457 0.41898 0.41340 0.40781 0.39664 0.39105 0.38546
## [105] 0.37429 0.36870 0.36312 0.35753 0.35195 0.34636 0.34077 0.33519
## [113] 0.32960 0.32401 0.31843 0.31284 0.30725 0.30167 0.29608 0.29049
## [121] 0.28491 0.27932 0.27374 0.26815 0.26256 0.25698 0.25139 0.24580
## [129] 0.24022 0.23463 0.22904 0.22332 0.21759 0.21187 0.20614 0.20041
## [137] 0.19469 0.18289 0.17679 0.17048 0.16416 0.15760 0.15103 0.14446
## [145] 0.13790 0.13133 0.12442 0.11751 0.10967 0.10184 0.09401 0.08617
## [153] 0.07834 0.07050 0.06169 0.05141 0.04113 0.03085 0.02056 0.01028
## [161] 0.00000
##
## $AUC
##   [1] 0.5807 0.5826 0.5803 0.5803 0.5791 0.5797 0.5773 0.5785 0.5788 0.5788
##  [11] 0.5816 0.5807 0.5806 0.5822 0.5823 0.5769 0.5824 0.5764 0.5798 0.5784
##  [21] 0.5777 0.5771 0.5807 0.5770 0.5816 0.5819 0.5766 0.5821 0.5808 0.5787
##  [31] 0.5838 0.5782 0.5791 0.5795 0.5777 0.5754 0.5743 0.5740 0.5756 0.5757
##  [41] 0.5747 0.5733 0.5737 0.5756 0.5756 0.5762 0.5743 0.5795 0.5792 0.5793
##  [51] 0.5733 0.5759 0.5744 0.5742 0.5808 0.5740 0.5798 0.5767 0.5735 0.5799
##  [61] 0.5748 0.5760 0.5796 0.5755 0.5737 0.5786 0.5737 0.5726 0.5725 0.5754
##  [71] 0.5789 0.5765 0.5784 0.5787 0.5732 0.5775 0.5759 0.5805 0.5802 0.5805
##  [81] 0.5813 0.5784 0.5830 0.5793 0.5739 0.5839 0.5721 0.5765 0.5812 0.5737
##  [91] 0.5728 0.5824 0.5769 0.5723 0.5752 0.5792 0.5717 0.5815 0.5701 0.5709
## [101] 0.5778 0.5879 0.5769 0.5667 0.5814 0.5728 0.5664 0.5689 0.5672 0.5659
## [111] 0.5773 0.5725 0.5717 0.5653 0.5801 0.5788 0.5625 0.5665 0.5715 0.5642
## [121] 0.5689 0.5706 0.5733 0.5669 0.5778 0.5839 0.5751 0.5607 0.5604 0.5787
## [131] 0.5555 0.5641 0.5719 0.5614 0.5639 0.5589 0.5577 0.5726 0.5561 0.5706
## [141] 0.5517 0.5854 0.5587 0.5572 0.5532 0.5497 0.5889 0.5654 0.5585 0.5384
## [151] 0.5967 0.5881 0.5616 0.5224 0.4994 0.5021 0.4551 0.4934 0.7081 0.7596
## [161] 0.0000
##
## $Cindex
## [1] 0.5773
```

```r
risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = gg2.linpred.glasgow, tmax =
```




```
## $utimes
##   [1]   0.80   1.63   2.50   3.00   3.40   3.73   3.83   3.90   4.47   4.57
##  [11]   4.67   4.73   4.83   4.87   5.00   5.30   5.77   5.83   5.87   6.00
##  [21]   6.10   6.27   6.30   6.67   6.93   7.00   7.10   7.66   7.67   7.73
##  [31]   8.00   8.17   8.33   8.43   8.47   8.77   8.80   9.00   9.03   9.30
##  [41]   9.40   9.83  10.13  10.27  10.40  10.50  11.07  11.30  11.33  11.50
##  [51]  11.60  11.90  12.33  12.47  12.97  13.00  13.03  13.10  13.30  13.43
##  [61]  13.50  13.57  13.70  13.93  14.00  14.80  15.37  15.40  15.57  16.00
##  [71]  16.17  16.33  16.47  16.77  16.95  17.00  17.07  17.57  17.83  18.03
##  [81]  18.17  18.40  18.50  18.57  18.93  19.10  19.50  19.63  19.80  20.00
##  [91]  20.07  20.43  20.67  20.90  21.53  21.80  22.00  22.60  23.07  23.10
## [101]  23.17  24.00  24.37  24.70  25.00  25.67  25.83  26.33  26.40  26.50
## [111]  26.60  26.70  27.53  28.13  28.33  28.53  29.03  29.10  29.27  29.60
## [121]  29.67  30.07  30.97  31.00  31.53  32.00  33.00  33.10  33.40  33.47
## [131]  34.37  35.10  35.97  36.23  36.30  36.67  37.00  38.00  39.60  41.23
## [141]  43.07  45.37  46.67  47.43  47.73  48.00  49.00  51.00  54.90  59.00
## [151]  63.13  65.00  67.00  70.00  77.00  85.00  85.80  90.33  93.00  94.77
```

```
## [161] 116.00
##
## $St
##   [1] 0.99476 0.98930 0.98383 0.97834 0.97284 0.96734 0.96185 0.95086
##   [9] 0.93986 0.93437 0.92887 0.92337 0.91788 0.90689 0.89589 0.89040
##  [17] 0.88490 0.87940 0.87391 0.86841 0.86291 0.85742 0.85192 0.84643
##  [25] 0.84093 0.83543 0.82994 0.82444 0.81894 0.81345 0.80246 0.79696
##  [33] 0.79146 0.78597 0.78047 0.77497 0.76948 0.76398 0.75845 0.75291
##  [41] 0.74737 0.74184 0.73630 0.73077 0.72523 0.71969 0.71416 0.70862
##  [49] 0.70308 0.69755 0.69201 0.68648 0.68094 0.67540 0.66987 0.66433
##  [57] 0.65880 0.65326 0.64772 0.64219 0.63665 0.63112 0.62558 0.62004
##  [65] 0.61451 0.60892 0.60333 0.59775 0.59216 0.58658 0.58099 0.57540
##  [73] 0.56982 0.56423 0.55864 0.55306 0.54747 0.54188 0.53630 0.53071
##  [81] 0.52512 0.51954 0.51395 0.50837 0.50278 0.49719 0.49161 0.48602
##  [89] 0.48043 0.46926 0.46367 0.45809 0.45250 0.44691 0.44133 0.43574
##  [97] 0.43016 0.42457 0.41898 0.41340 0.40781 0.39664 0.39105 0.38546
## [105] 0.37429 0.36870 0.36312 0.35753 0.35195 0.34636 0.34077 0.33519
## [113] 0.32960 0.32401 0.31843 0.31284 0.30725 0.30167 0.29608 0.29049
## [121] 0.28491 0.27932 0.27374 0.26815 0.26256 0.25698 0.25139 0.24580
## [129] 0.24022 0.23463 0.22904 0.22332 0.21759 0.21187 0.20614 0.20041
## [137] 0.19469 0.18289 0.17679 0.17048 0.16416 0.15760 0.15103 0.14446
## [145] 0.13790 0.13133 0.12442 0.11751 0.10967 0.10184 0.09401 0.08617
## [153] 0.07834 0.07050 0.06169 0.05141 0.04113 0.03085 0.02056 0.01028
## [161] 0.00000
##
## $AUC
##   [1] 0.5685 0.5702 0.5679 0.5680 0.5666 0.5671 0.5650 0.5661 0.5665 0.5667
##  [11] 0.5694 0.5685 0.5683 0.5704 0.5701 0.5648 0.5704 0.5646 0.5678 0.5664
##  [21] 0.5654 0.5657 0.5686 0.5651 0.5697 0.5700 0.5647 0.5701 0.5690 0.5668
##  [31] 0.5719 0.5667 0.5675 0.5679 0.5663 0.5644 0.5620 0.5620 0.5657 0.5638
##  [41] 0.5636 0.5618 0.5621 0.5632 0.5644 0.5637 0.5626 0.5676 0.5670 0.5676
##  [51] 0.5614 0.5634 0.5625 0.5632 0.5690 0.5621 0.5680 0.5645 0.5616 0.5681
##  [61] 0.5621 0.5637 0.5678 0.5640 0.5617 0.5667 0.5617 0.5605 0.5606 0.5639
##  [71] 0.5669 0.5648 0.5669 0.5672 0.5610 0.5650 0.5643 0.5684 0.5674 0.5677
##  [81] 0.5693 0.5674 0.5708 0.5676 0.5620 0.5720 0.5604 0.5647 0.5696 0.5626
##  [91] 0.5613 0.5709 0.5651 0.5608 0.5644 0.5686 0.5602 0.5699 0.5589 0.5594
## [101] 0.5669 0.5769 0.5643 0.5553 0.5693 0.5627 0.5552 0.5574 0.5566 0.5548
## [111] 0.5661 0.5583 0.5616 0.5544 0.5695 0.5685 0.5525 0.5540 0.5617 0.5558
## [121] 0.5595 0.5612 0.5634 0.5587 0.5674 0.5732 0.5654 0.5513 0.5513 0.5693
## [131] 0.5464 0.5519 0.5607 0.5520 0.5570 0.5516 0.5491 0.5668 0.5466 0.5533
## [141] 0.5429 0.5764 0.5440 0.5500 0.5459 0.5409 0.5783 0.5370 0.5470 0.5256
## [151] 0.5821 0.5759 0.5498 0.5086 0.4861 0.4878 0.4420 0.4879 0.7018 0.7584
## [161] 0.0000
##
## $Cindex
## [1] 0.5656

risksetAUC(data.glasgow$Time/365.25*12, status = data.glasgow$DSD, marker = cph.linpred.glasgow, tmax =
```




```
## $utimes
##   [1]   0.80   1.63   2.50   3.00   3.40   3.73   3.83   3.90   4.47   4.57
##  [11]   4.67   4.73   4.83   4.87   5.00   5.30   5.77   5.83   5.87   6.00
##  [21]   6.10   6.27   6.30   6.67   6.93   7.00   7.10   7.66   7.67   7.73
##  [31]   8.00   8.17   8.33   8.43   8.47   8.77   8.80   9.00   9.03   9.30
##  [41]   9.40   9.83  10.13  10.27  10.40  10.50  11.07  11.30  11.33  11.50
##  [51]  11.60  11.90  12.33  12.47  12.97  13.00  13.03  13.10  13.30  13.43
##  [61]  13.50  13.57  13.70  13.93  14.00  14.80  15.37  15.40  15.57  16.00
##  [71]  16.17  16.33  16.47  16.77  16.95  17.00  17.07  17.57  17.83  18.03
##  [81]  18.17  18.40  18.50  18.57  18.93  19.10  19.50  19.63  19.80  20.00
##  [91]  20.07  20.43  20.67  20.90  21.53  21.80  22.00  22.60  23.07  23.10
## [101]  23.17  24.00  24.37  24.70  25.00  25.67  25.83  26.33  26.40  26.50
## [111]  26.60  26.70  27.53  28.13  28.33  28.53  29.03  29.10  29.27  29.60
## [121]  29.67  30.07  30.97  31.00  31.53  32.00  33.00  33.10  33.40  33.47
## [131]  34.37  35.10  35.97  36.23  36.30  36.67  37.00  38.00  39.60  41.23
## [141]  43.07  45.37  46.67  47.43  47.73  48.00  49.00  51.00  54.90  59.00
## [151]  63.13  65.00  67.00  70.00  77.00  85.00  85.80  90.33  93.00  94.77
```

```
## [161] 116.00
##
## $St
##   [1] 0.99476 0.98930 0.98383 0.97834 0.97284 0.96734 0.96185 0.95086
##   [9] 0.93986 0.93437 0.92887 0.92337 0.91788 0.90689 0.89589 0.89040
##  [17] 0.88490 0.87940 0.87391 0.86841 0.86291 0.85742 0.85192 0.84643
##  [25] 0.84093 0.83543 0.82994 0.82444 0.81894 0.81345 0.80246 0.79696
##  [33] 0.79146 0.78597 0.78047 0.77497 0.76948 0.76398 0.75845 0.75291
##  [41] 0.74737 0.74184 0.73630 0.73077 0.72523 0.71969 0.71416 0.70862
##  [49] 0.70308 0.69755 0.69201 0.68648 0.68094 0.67540 0.66987 0.66433
##  [57] 0.65880 0.65326 0.64772 0.64219 0.63665 0.63112 0.62558 0.62004
##  [65] 0.61451 0.60892 0.60333 0.59775 0.59216 0.58658 0.58099 0.57540
##  [73] 0.56982 0.56423 0.55864 0.55306 0.54747 0.54188 0.53630 0.53071
##  [81] 0.52512 0.51954 0.51395 0.50837 0.50278 0.49719 0.49161 0.48602
##  [89] 0.48043 0.46926 0.46367 0.45809 0.45250 0.44691 0.44133 0.43574
##  [97] 0.43016 0.42457 0.41898 0.41340 0.40781 0.39664 0.39105 0.38546
## [105] 0.37429 0.36870 0.36312 0.35753 0.35195 0.34636 0.34077 0.33519
## [113] 0.32960 0.32401 0.31843 0.31284 0.30725 0.30167 0.29608 0.29049
## [121] 0.28491 0.27932 0.27374 0.26815 0.26256 0.25698 0.25139 0.24580
## [129] 0.24022 0.23463 0.22904 0.22332 0.21759 0.21187 0.20614 0.20041
## [137] 0.19469 0.18289 0.17679 0.17048 0.16416 0.15760 0.15103 0.14446
## [145] 0.13790 0.13133 0.12442 0.11751 0.10967 0.10184 0.09401 0.08617
## [153] 0.07834 0.07050 0.06169 0.05141 0.04113 0.03085 0.02056 0.01028
## [161] 0.00000
##
## $AUC
##   [1] 0.6364 0.6374 0.6358 0.6348 0.6322 0.6326 0.6292 0.6321 0.6324 0.6302
##  [11] 0.6310 0.6332 0.6319 0.6290 0.6321 0.6256 0.6308 0.6255 0.6304 0.6275
##  [21] 0.6268 0.6247 0.6302 0.6241 0.6276 0.6284 0.6250 0.6298 0.6277 0.6281
##  [31] 0.6294 0.6236 0.6255 0.6270 0.6241 0.6216 0.6233 0.6234 0.6225 0.6251
##  [41] 0.6210 0.6214 0.6223 0.6247 0.6219 0.6252 0.6228 0.6260 0.6294 0.6243
##  [51] 0.6206 0.6245 0.6229 0.6206 0.6270 0.6225 0.6258 0.6263 0.6214 0.6260
##  [61] 0.6237 0.6251 0.6249 0.6206 0.6230 0.6230 0.6221 0.6206 0.6213 0.6262
##  [71] 0.6245 0.6226 0.6246 0.6251 0.6226 0.6290 0.6219 0.6263 0.6305 0.6294
##  [81] 0.6247 0.6220 0.6274 0.6233 0.6213 0.6303 0.6198 0.6261 0.6261 0.6204
##  [91] 0.6230 0.6302 0.6277 0.6221 0.6192 0.6245 0.6210 0.6277 0.6202 0.6223
## [101] 0.6244 0.6355 0.6289 0.6152 0.6295 0.6147 0.6126 0.6194 0.6166 0.6150
## [111] 0.6283 0.6195 0.6110 0.6091 0.6231 0.6215 0.6095 0.6148 0.6108 0.6050
## [121] 0.6066 0.6078 0.6108 0.6056 0.6142 0.6253 0.6105 0.6061 0.6074 0.6204
## [131] 0.6008 0.6168 0.6239 0.6136 0.5999 0.6096 0.6064 0.6061 0.6068 0.6287
## [141] 0.6039 0.6359 0.6261 0.6045 0.6015 0.6116 0.6403 0.6405 0.6143 0.6478
## [151] 0.6759 0.6226 0.5906 0.5641 0.5829 0.5572 0.5140 0.5359 0.7544 0.7707
## [161] 0.0000
##
## $Cindex
## [1] 0.6255
```