Blood Xpert-Ultra on biobanked KDHTB samples: analysis script

Linda Boloko, David Barr, KDHTB study team

2019-2021

Contents

1 Introduction

This Rmd consolidates analysis from previous blood_Xpert.Rmd after comments from coauthors and is point of reference for revised manuscript final analysis.

2 Objectives and aims

Objectives are to:

- report the diagnostic utility of a blood Xpert-ultra testing protocol in people living with HIV (PLHIV) admitted to hospital with suspected tuberculosis, and
- test the hypothesis that presence and magnitude of *M. tuberculosis* bloodstream infection (MTBBSI) is robustly assoociated with diseases phenotype and outcome.

Translating into aims:

1. Estimate sensitivity and specificity, and diagnostic yield, of a blood Xpert-ultra testing protocol for diagnosing tuberculosis.
2. Compare diagnostic utility of blood Xpert-ultra to other rapid TB diagnostic tests.
3. Assess the relationship between patient characteristics and diagnostic performance of blood Xpert-ultra.
4. Test if blood Xpert-ultra positivity is associated with established mediators of pathophysiology, clinical phenotype, and mortality in HIV-associated tuberculosis.
5. Within the strata of patients with positive blood Xpert-ultra, test for a dose-response relationship between blood Xpert-ultra cycle threshold (Ct) value and established mediators of pathophysiology, clinical phenotype, and mortality risk.

3 Definitions

Sensitivity = number of patients who have positive result on the index test divided by total number of patients who had:

1. A valid index test performed i.e. unable to obtain sample or technical problem with processing are excluded; AND
2. TB diagnosis confirmed by a *strict microbiological reference standard*: any positive TB culture result (sputum, blood or any other site) and/or positive Xpert from sputum urine or other site (blood Xpert not used in this reference standard as unvalidated). urine-LAM is also excluded.

Diagnostic yield = number of patients who have positive result on the index test divided by total number of patients who had TB diagnosis confirmed by *any TB diagnostic* including any positive TB culture from any site, any positive Xpert result (sputum, urine, blood, other), and/or positive urine LAM (Alere). Patients with a missing test result due to inability to obtain sample or technical failure of the index test are included as negative results in the numerator.

Specificity = number of patients who have a valid negative result on the index test divided by total number of patients who had:

1. A valid index test performed i.e. unable to obtain sample or technical problem with processing are excluded; AND
2. Two or more negative and valid reference TB diagnostic tests (culture from any site, Xpert results from any site other than blood, urine-LAM).

4 Overall study numbers for CONSORT diagram

```
# Data frame includes only patienst meeting global KDHTB inclusion criteria:
(N_kdhtb <- nrow(df))
```

```
## [1] 659
```

```
# excluding patients for whom blood sample not available
# (used elsewhere, MCAR) - note have confirmed this manually
# with the raw data files - those coded NA there are no
# samples processed for.
(sum(is.na(df$blood_Xpert_MTB)))
```

```
## [1] 77
```

```
# Which leaves
df <- filter(df, !is.na(blood_Xpert_MTB))
(n_inclusion <- nrow(df))
```

```
## [1] 582
```

```
# Numbers from this n=582 meeting the 2 TB diagnosis ref standards
df %>%
  mutate(strict_micro_ref =
           (!is.na(df$sputumCulture1_cultureID) & df$sputumCulture1_cultureID=="MTB") |
           (!is.na(df$sputumCulture2_cultureID) & df$sputumCulture2_cultureID=="MTB") |
           (!is.na(df$sputumCulture3_cultureID) & df$sputumCulture3_cultureID=="MTB") |

           (!is.na(df$sputumGXP1_GeneXpert) &  df$sputumGXP1_GeneXpert=="MTB") |
           (!is.na(df$sputumGXP2_GeneXpert) & df$sputumGXP2_GeneXpert=="MTB") |
           (!is.na(df$sputumGXP3_GeneXpert) & df$sputumGXP3_GeneXpert=="MTB") |

           (!is.na(df$MBC1_cultureID) & df$MBC1_cultureID == "MTB") |
           (!is.na(df$MBC2_cultureID) & df$MBC2_cultureID == "MTB") |
           (!is.na(df$MBC3_cultureID) & df$MBC3_cultureID == "MTB") |

           (!is.na(df$uMTBculture) & df$uMTBculture == "MTB") |
           (!is.na(df$otherCul1_cultureID) & df$otherCul1_cultureID=="MTB") |
           (!is.na(df$otherCul2_cultureID) & df$otherCul2_cultureID=="MTB") |

           (!is.na(df$uGXP) & df$uGXP=="MTB") |
           (!is.na(df$otherGXP) & df$otherGXP=="MTB"),

         anyTBtest_pos =
           (strict_micro_ref==TRUE) |

           (!is.na(df$ALERE_FC) & df$ALERE_FC==1) |
```

```
    #       (!is.na(df$FUJISAI_FC) & df$FUJISAI_FC==1) |
           (!is.na(df$blood_Xpert_MTB) &
              df$blood_Xpert_MTB!="Negative" & df$blood_Xpert_MTB!="Error"),

         bld_xpert_pos = (!is.na(df$blood_Xpert_MTB) &
              df$blood_Xpert_MTB!="Negative" & df$blood_Xpert_MTB!="Error")
  ) -> df

# TB cases by strict micro reference
(n_strict <- sum(df$strict_micro_ref))
```

```
## [1] 424
```

```
# TB cases by extended micro reference
(n_any <- sum(df$anyTBtest_pos))
```

```
## [1] 447
```

```
# specificity variables
df %>%
  select( # all TB diagnostics
    UID,
    sputumCulture1_cultureID,
    sputumCulture2_cultureID,
    sputumCulture3_cultureID,
    sputumGXP1_GeneXpert,
    sputumGXP2_GeneXpert,
    sputumGXP3_GeneXpert,
    MBC1_cultureID,
    MBC2_cultureID,
    MBC3_cultureID,
    uMTBculture,
    otherCul1_cultureID,
    otherCul2_cultureID,
    uGXP,
    otherGXP,
    ALERE_FC,
    FUJISAI_FC
  ) %>%
  mutate(
    ALERE_FC = case_when(
      ALERE_FC==1 ~ "MTB",
      ALERE_FC==0 ~ "negative"),
    FUJISAI_FC = case_when(
      FUJISAI_FC==1 ~ "MTB",
      FUJISAI_FC==0 ~ "negative")
    ) %>%
  pivot_longer(2:17) %>%
  mutate(
    value = na_if(value, "contaminated"),
    value = na_if(value, "AFB"),
    value = na_if(value, "NTM"),
    value = recode(value, NEG = "negative")
  ) %>%
  group_by(UID) %>%
```

```
  summarise(
    valid_test_n = sum(!is.na(value)),
    negative_test_n = sum(value=="negative", na.rm = TRUE),
    positive_test_n = sum(value=="MTB", na.rm = TRUE)
  ) -> tests_df

df %>%
  left_join(
    tests_df,
    by = "UID"
  ) -> df

# patients meeting "no TB" reference definition for specificity analysis
(n_no_tb <- sum(df$negative_test_n>1 & df$positive_test_n<1))
```

```
## [1] 126
```

```
# 2 patients who were blood xpert +ve but "no TB" by micro definition used in specificity
q_FP_bldxpt <- df$UID[df$negative_test_n>1 & df$positive_test_n<1 & df$bld_xpert_pos]

# They are:

# UID81: 25F CD4 31 VL 29 PC: weight loss, drencing NS, diarrhoea,
# inability to walk unaided. Fever, peripheral lymphadenopathy, weight 38Kg, oedema,
# unable to produce sputum. Anaemia, neutrophilia, metabolic acidosis with AKI,
# miliary lesions on CXR, splenic microabscess, evidence of renal TB and
# pyelonephritis on USS. Started TB treatment day before recruitment, also treated for
# bacterial sepsis secondary to pyelonephritis, died 6 days after admission.

# UID371 40M CD4 10 VL 195 PC: cough, LOW, drenching night sweats, diarrhoea.
# R side pleural effusion, ascities, and uveitis. No diagnostic aspirate performed,
# treated empirically for TB with improvement. survived.
```

```
# Distribution of non-TB diagnoses:

df %>%
  filter(anyTBtest_pos == FALSE) %>%
  group_by(TBcat) %>%
  count() %>% kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

TBcat	n
alt.bloodpathogen	6
alt.cap	31
alt.crypto	5
alt.other	12
alt.PJP	5
clinical.TB	68
possible.TB	8

5 Cohort description table

```
mycontrols <- tableby.control(numeric.test = "kwt",
                              cat.test="fe",
                              cat.simplify = FALSE,
                              numeric.stats = c("median", "q1q3"))

tab1 <- tableby(bld_xpert_pos ~ age + Sex + CD4 + ARTstatus +
                  HR + lactate + Haemoglobin +
                  creatinine + CRP + Sodium +
                  Cough + LossOfAppetite + DrenchingNightSweats + LossOfWeight +
                  survival.12weeks,
                data = df[df$blood_Xpert_MTB!="Error",], control=mycontrols)
```

```
summary(tab1, text=TRUE)
```

	FALSE (N=413)	TRUE (N=165)	Total (N=578)	p value
age				0.849
- Median	36.307	36.003	36.221	
- Q1, Q3	30.964, 44.008	30.928, 43.761	30.947, 43.859	
Sex				0.065
- F	227 (55.0%)	76 (46.1%)	303 (52.4%)	
- M	186 (45.0%)	89 (53.9%)	275 (47.6%)	
CD4				< 0.001
- Median	86.000	25.000	62.000	
- Q1, Q3	34.000, 160.000	8.000, 60.000	22.000, 133.000	
ARTstatus				< 0.001
- N-Miss	6	1	7	
- Defaulted	81 (19.9%)	49 (29.9%)	130 (22.8%)	
- Naive	150 (36.9%)	70 (42.7%)	220 (38.5%)	
- On_ART	176 (43.2%)	45 (27.4%)	221 (38.7%)	
HR				< 0.001
- Median	102.500	111.000	104.000	
- Q1, Q3	92.000, 117.000	98.000, 123.000	94.000, 120.000	
lactate				< 0.001
- Median	1.700	2.200	1.800	
- Q1, Q3	1.200, 2.300	1.500, 3.200	1.300, 2.500	
Haemoglobin				< 0.001
- Median	9.300	7.900	8.800	
- Q1, Q3	7.600, 10.800	6.700, 9.200	7.225, 10.500	
creatinine				< 0.001
- Median	76.000	96.000	78.500	
- Q1, Q3	58.000, 105.000	66.000, 170.000	59.000, 118.000	
CRP				< 0.001
- Median	137.000	196.000	154.000	
- Q1, Q3	75.275, 225.075	142.000, 251.000	86.550, 231.800	
Sodium				< 0.001
- Median	130.000	127.000	129.000	
- Q1, Q3	126.000, 132.000	124.000, 130.000	125.000, 132.000	
Cough				0.544
- N-Miss	13	8	21	
- N	122 (30.5%)	52 (33.1%)	174 (31.2%)	
- Y	278 (69.5%)	105 (66.9%)	383 (68.8%)	

	FALSE (N=413)	TRUE (N=165)	Total (N=578)	p value
LossOfAppetite				0.190
- N-Miss	14	12	26	
- N	140 (35.1%)	44 (28.8%)	184 (33.3%)	
- Y	259 (64.9%)	109 (71.2%)	368 (66.7%)	
DrenchingNightSweats				0.848
- N-Miss	18	12	30	
- N	178 (45.1%)	67 (43.8%)	245 (44.7%)	
- Y	217 (54.9%)	86 (56.2%)	303 (55.3%)	
LossOfWeight				0.759
- N-Miss	16	11	27	
- N	43 (10.8%)	15 (9.7%)	58 (10.5%)	
- Y	354 (89.2%)	139 (90.3%)	493 (89.5%)	
survival.12weeks				< 0.001
- Died	70 (16.9%)	51 (30.9%)	121 (20.9%)	
- LTFU	10 (2.4%)	2 (1.2%)	12 (2.1%)	
- Survived	333 (80.6%)	112 (67.9%)	445 (77.0%)	

6 Sensitivity & diagnostic yield

6.1 Which patients had sputum samples collected?

We want the sensitivity of a single sputum Xpert to compare against the sesnitivity of a single blood xpert etc. The database includes all sputum samples from study and from the national lab (NHLS) system ie routine care samples. We select only sputum Xperts collected between 28 days before and 5 days after day of study recruitment, and if there are more than one, select the one closest to day of recruitment, and if >1 at that timepoint select one of these at random.

All results

```
df %>%
    mutate(
    sptmxpert1_day = as.numeric(
      as.Date(df$sputumGXP1_Date, format = "%d/%m/%Y") -
        as.Date(df$StudyDate, format = "%d/%m/%Y")),
    sptmxpert2_day = as.numeric(
      as.Date(df$sputumGXP2_Date, format = "%d/%m/%Y") -
        as.Date(df$StudyDate, format = "%d/%m/%Y")),
    sptmxpert3_day = as.numeric(
      as.Date(df$sputumGXP3_Date, format = "%d/%m/%Y") -
        as.Date(df$StudyDate, format = "%d/%m/%Y")),
    # same for 2 blood cultures?
    mfl1_day = as.numeric(
      as.Date(df$MBC1_Date, format = "%d/%m/%Y") -
        as.Date(df$StudyDate, format = "%d/%m/%Y")),
    mfl2_day = as.numeric(
      as.Date(df$MBC2_Date, format = "%d/%m/%Y") -
        as.Date(df$StudyDate, format = "%d/%m/%Y"))) -> df

# set up some new variables which are just copies
# of the orginal GXP variables
df$sputumGXP1 <- df$sputumGXP1_GeneXpert
```

```r
df$sputumGXP2 <- df$sputumGXP2_GeneXpert
df$sputumGXP3 <- df$sputumGXP3_GeneXpert

# remove the results outside our date range
df$sputumGXP1[!is.na(df$sptmxpert1_day) &
  (df$sptmxpert1_day <= -29 | df$sptmxpert1_day>5)] <- NA
df$sputumGXP2[!is.na(df$sptmxpert2_day) &
  (df$sptmxpert2_day <= -29 | df$sptmxpert2_day>5)] <- NA
df$sputumGXP3[!is.na(df$sptmxpert3_day) &
  (df$sptmxpert3_day <= -29 | df$sptmxpert3_day>5)] <- NA

# also remove the day of collection from those samples
# so it doesn't mess with later for loop
df$sptmxpert1_day[is.na(df$sputumGXP1)] <- NA
df$sptmxpert2_day[is.na(df$sputumGXP2)] <- NA
df$sptmxpert3_day[is.na(df$sputumGXP3)] <- NA

# set seed to make random picking of the results reproducable
set.seed(123)

# create a new variable which will be our final sputum Xpert result
df$sputum_xpert <- rep("foo", nrow(df))

# This for loop now populates that new sputum variable so that it is:
## NA if all 3 sputum Xperts are NA
## gets result of single Xpert result if only one available
## picks closest to recruitment date or 'samples' one at random
## if 2 or 3 are available on same day

for(i in 1:nrow(df)){

  if(is.na(df$sputumGXP1[i]) &
     is.na(df$sputumGXP2[i]) &
     is.na(df$sputumGXP3[i])){
    df$sputum_xpert[i] <- NA  # If all 3 NA then result is NA
     } else

  if(!is.na(df$sputumGXP1[i]) &
     is.na(df$sputumGXP2[i]) &
     is.na(df$sputumGXP3[i])){
    df$sputum_xpert[i] <- df$sputumGXP1[i]
     } else

  if(is.na(df$sputumGXP1[i]) &
     !is.na(df$sputumGXP2[i]) &
     is.na(df$sputumGXP3[i])){
    df$sputum_xpert[i] <- df$sputumGXP2[i]
     } else

  if(is.na(df$sputumGXP1[i]) &
     is.na(df$sputumGXP2[i]) &
     !is.na(df$sputumGXP3[i])){
    df$sputum_xpert[i] <- df$sputumGXP3[i]
```

```
    } else
                    # If only 1/3 recorded then result is that one
if(!is.na(df$sputumGXP1[i]) &
   !is.na(df$sputumGXP2[i]) &
   is.na(df$sputumGXP3[i])){
  if(
    (abs(df$sptmxpert1_day[i])<abs(df$sptmxpert2_day[i]) &
       !is.na(abs(df$sptmxpert1_day[i])<abs(df$sptmxpert2_day[i])))
  ){
    df$sputum_xpert[i] <- df$sputumGXP1[i]
    }else
      if(
    (abs(df$sptmxpert1_day[i])>abs(df$sptmxpert2_day[i]) &
       !is.na(abs(df$sptmxpert1_day[i])>abs(df$sptmxpert2_day[i])))
  ){
    df$sputum_xpert[i] <- df$sputumGXP2[i]
    } else
      if(
        (abs(df$sptmxpert1_day[i])==abs(df$sptmxpert2_day[i]) &
       !is.na(abs(df$sptmxpert1_day[i])==abs(df$sptmxpert2_day[i])))
      ){
        df$sputum_xpert[i] <-
    sample(c(df$sputumGXP1[i],
             df$sputumGXP2[i]), 1)
      }
           } else
                  # if 2 result available sample 1 closest to recruitment
                  # and if both same day select between them at random;
                  # now same for other combos of 2

  if(is.na(df$sputumGXP1[i]) &
   !is.na(df$sputumGXP2[i]) &
   !is.na(df$sputumGXP3[i])){
  if(
    (abs(df$sptmxpert2_day[i])<abs(df$sptmxpert3_day[i]) &
       !is.na(abs(df$sptmxpert2_day[i])<abs(df$sptmxpert3_day[i])))
  ){
    df$sputum_xpert[i] <- df$sputumGXP2[i]
    } else
      if(
    (abs(df$sptmxpert2_day[i])>abs(df$sptmxpert3_day[i]) &
       !is.na(abs(df$sptmxpert2_day[i])>abs(df$sptmxpert3_day[i])))
  ){
    df$sputum_xpert[i] <- df$sputumGXP3[i]
    } else
      if(
        (abs(df$sptmxpert2_day[i])==abs(df$sptmxpert3_day[i]) &
       !is.na(abs(df$sptmxpert2_day[i])==abs(df$sptmxpert3_day[i])))
      ){
        df$sputum_xpert[i] <-
    sample(c(df$sputumGXP2[i],
             df$sputumGXP3[i]), 1)
      }
```

```
            } else

   if(!is.na(df$sputumGXP1[i]) &
    is.na(df$sputumGXP2[i]) &
    !is.na(df$sputumGXP3[i])){
   if(
     (abs(df$sptmxpert1_day[i])<abs(df$sptmxpert3_day[i]) &
        !is.na(abs(df$sptmxpert1_day[i])<abs(df$sptmxpert3_day[i])))
   ){
     df$sputum_xpert[i] <- df$sputumGXP1[i]
     } else
       if(
     (abs(df$sptmxpert1_day[i])>abs(df$sptmxpert3_day[i]) &
        !is.na(abs(df$sptmxpert1_day[i])>abs(df$sptmxpert3_day[i])))
   ){
     df$sputum_xpert[i] <- df$sputumGXP3[i]
     } else
       if(
         (abs(df$sptmxpert1_day[i])==abs(df$sptmxpert3_day[i]) &
        !is.na(abs(df$sptmxpert1_day[i])==abs(df$sptmxpert3_day[i])))
       ){
         df$sputum_xpert[i] <-
     sample(c(df$sputumGXP1[i],
              df$sputumGXP3[i]), 1)
       }
            } else
# now for the times when all 3 results are available...
 if(!is.na(df$sputumGXP1[i]) &
    !is.na(df$sputumGXP2[i]) &
    !is.na(df$sputumGXP3[i])){

# one sample of 3 is closest to recruitment:
     if(
       (abs(df$sptmxpert1_day[i])<abs(df$sptmxpert2_day[i])) &
       (abs(df$sptmxpert1_day[i])<abs(df$sptmxpert3_day[i]))){
         df$sputum_xpert[i] <- df$sputumGXP1[i]}else
     if(
       (abs(df$sptmxpert2_day[i])<abs(df$sptmxpert1_day[i])) &
       (abs(df$sptmxpert2_day[i])<abs(df$sptmxpert3_day[i]))){
         df$sputum_xpert[i] <- df$sputumGXP2[i]}else
     if(
       (abs(df$sptmxpert3_day[i])<abs(df$sptmxpert2_day[i])) &
       (abs(df$sptmxpert3_day[i])<abs(df$sptmxpert1_day[i]))){
         df$sputum_xpert[i] <- df$sputumGXP3[i]}else

# now cases where 2 of 3 available are same day

     if(
       (abs(df$sptmxpert1_day[i])<abs(df$sptmxpert2_day[i])) &
       (abs(df$sptmxpert1_day[i])==abs(df$sptmxpert3_day[i]))){
         df$sputum_xpert[i] <- sample(
           c(df$sputumGXP1[i], df$sputumGXP3[i]), 1)}else
     if(
```

```
        (abs(df$sptmxpert1_day[i])<abs(df$sptmxpert3_day[i])) &
        (abs(df$sptmxpert1_day[i])==abs(df$sptmxpert2_day[i]))){
          df$sputum_xpert[i] <- sample(
            c(df$sputumGXP1[i], df$sputumGXP2[i]), 1)}else
      if(
        (abs(df$sptmxpert2_day[i])<abs(df$sptmxpert1_day[i])) &
        (abs(df$sptmxpert2_day[i])==abs(df$sptmxpert3_day[i]))){
          df$sputum_xpert[i] <- sample(
            c(df$sputumGXP2[i], df$sputumGXP3[i]), 1)}else

 # all 3 are same day, sample one at random
      if(
        (abs(df$sptmxpert1_day[i])==abs(df$sptmxpert2_day[i])) &
        (abs(df$sptmxpert2_day[i])==abs(df$sptmxpert3_day[i]))
        ){
        df$sputum_xpert[i] <- sample(
          c(df$sputumGXP1[i], df$sputumGXP2[i], df$sputumGXP3[i]), 1)}
  }
}
```

6.2 Sensitivity

% positive index tests from denominator of proven TB by strict micro reference and valid test performed.

6.2.1 In whole cohort

With specificity tacked on here.

```
# make a function which collects the statistics we want for sensitivity:
 ## number with valid test for index test (eg sputum xpert);
 ## number with proven TB;
 ## number of these that tested positive (true positives);
 ## sesnitivity, 95% CI for sensitivity

sens_function <- function(reference_std, index_test){

  dat <- data.frame(reference_std, index_test)
  dat <- dat[!is.na(index_test),]
  n_valid_test = nrow(dat)
  n_provenTB = sum(dat$reference_std==TRUE)
  n_true_positive = sum(dat$reference_std==TRUE &
                          dat$index_test=="MTB")
  sens = round(n_true_positive / n_provenTB, 2)
  CI_95 = paste0(
    round(
    prop.test(
      n_true_positive, n_provenTB)$conf.int[1],
    2),
    " to ",
    round(
    prop.test(
      n_true_positive, n_provenTB)$conf.int[2],
```

```
    2))

  return(data.frame(n_valid_test, n_provenTB,
           n_true_positive, sens, CI_95))
}

# standardise how the results are coded in these variables:
df$ALERE_FC[df$ALERE_FC==1] <- "MTB"
df$ALERE_FC[df$ALERE_FC=="0"] <- "NEG"
df$bld_xpert_pos[df$bld_xpert_pos==TRUE] <- "MTB"
df$bld_xpert_pos[df$bld_xpert_pos=="FALSE"] <- "NEG"
df$bld_xpert_pos[df$blood_Xpert_MTB=="Error"] <- NA

# can now apply the function to each variable of interest
# and combine them in an data frame
bind_rows(
  sens_function(df$strict_micro_ref, df$ALERE_FC),
  sens_function(df$strict_micro_ref, df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> sens_table1

sens_table1 %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	n_valid_test	n_provenTB	n_true_positive	sens	CI_95
Alere_LAM	519	375	171	0.46	0.4 to 0.51
blood_Xpert	578	423	161	0.38	0.33 to 0.43

```
# don't want this by subgroups so no need to encode in a function

df %>%
  filter(negative_test_n > 1 & positive_test_n<1) %>%
  select(
    bld_xpert_pos
  ) %>%
  summarise(
    n_valid_index = sum(!is.na(bld_xpert_pos)),
    n_FP = sum(bld_xpert_pos=="MTB", na.rm = TRUE)
  ) %>%
  mutate(
    n_TN = n_valid_index - n_FP,
    specificity = n_TN / n_valid_index,
    CI_95 = paste0(
    round(
    prop.test(
      n_TN, n_valid_index)$conf.int[1],
    2),
```

```
    " to ",
    round(
    prop.test(
      n_TN, n_valid_index)$conf.int[2],
    2)
  )
  ) -> spec_bldxpt_table
```

6.2.1.1 specificty in whole cohort

Specificity whole cohort:

```
spec_bldxpt_table %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

n_valid_index	n_FP	n_TN	specificity	CI_95
123	2	121	0.9837398	0.94 to 1

6.2.2 Sensitivity in pre-specified sub-groups

```
# make dataframes which are sub-groups of interest

df[df$CD4<100,] -> cd4_df
df[df$lactate>2.5 & !is.na(df$lactate),] -> lact_df
df[df$Haemoglobin<8,] -> hb_df
df[df$survival.12weeks=="Died",] -> died_df

# re-use our function from above
bind_rows(
  sens_function(cd4_df$strict_micro_ref, cd4_df$ALERE_FC),
  sens_function(cd4_df$strict_micro_ref, cd4_df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> sens_table_cd4

bind_rows(
  sens_function(lact_df$strict_micro_ref, lact_df$ALERE_FC),
  sens_function(lact_df$strict_micro_ref, lact_df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> sens_table_lact

bind_rows(
  sens_function(hb_df$strict_micro_ref, hb_df$ALERE_FC),
  sens_function(hb_df$strict_micro_ref, hb_df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> sens_table_hb

bind_rows(
  sens_function(died_df$strict_micro_ref, died_df$ALERE_FC),
  sens_function(died_df$strict_micro_ref, died_df$bld_xpert_pos)) %>%
```

```
  mutate(index_test =
           c("Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> sens_table_died
```

6.2.2.1 CD4 < 100

In patients with CD4 < 100 cells/mm3:

```
sens_table_cd4 %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	n_valid_test	n_provenTB	n_true_positive	sens	CI_95
Alere_LAM	329	254	145	0.57	0.51 to 0.63
blood_Xpert	372	288	145	0.50	0.44 to 0.56

6.2.2.2 Haemoglobin < 8

In patients with Hb < 8:

```
sens_table_hb %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	n_valid_test	n_provenTB	n_true_positive	sens	CI_95
Alere_LAM	175	144	89	0.62	0.53 to 0.7
blood_Xpert	206	170	80	0.47	0.39 to 0.55

6.2.2.3 Lactate > 2.5

In patients with lactate > 2.5:

```
sens_table_lact %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	n_valid_test	n_provenTB	n_true_positive	sens	CI_95
Alere_LAM	119	97	49	0.51	0.4 to 0.61
blood_Xpert	139	115	59	0.51	0.42 to 0.61

6.2.2.4 By mortality

In those who died by 12 weeks:

```
sens_table_died %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

6.3 Diagnostic yield

Reference standard is any positive TB test; those with missing index test result are included as negative test.

index_test	n_valid_test	n_provenTB	n_true_positive	sens	CI_95
Alere_LAM	101	74	39	0.53	0.41 to 0.64
blood_Xpert	121	89	49	0.55	0.44 to 0.65

6.3.1 In whole cohort

```
yield_function <- function(reference_std, index_test){
  dat <- data.frame(reference_std, index_test,
                    stringsAsFactors = FALSE)
  dat$index_test[is.na(dat$index_test)] <- "neg"
  # dat <- dat[!is.na(index_test),] # keep these in
  N = nrow(dat) # this is now all the patients
  n_TB = sum(dat$reference_std==TRUE)
  n_true_positive = sum(dat$reference_std==TRUE &
                          dat$index_test=="MTB")
  diag_yield = round(n_true_positive / n_TB, 2)
  CI_95 = paste0(
    round(
    prop.test(
      n_true_positive, n_TB)$conf.int[1],
    2),
    " to ",
    round(
    prop.test(
      n_true_positive, n_TB)$conf.int[2],
    2))

  return(data.frame(N, n_TB,
           n_true_positive, diag_yield, CI_95))
}

bind_rows(
  yield_function(df$anyTBtest_pos, df$sputum_xpert),
  yield_function(df$anyTBtest_pos, df$ALERE_FC),
  yield_function(df$anyTBtest_pos, df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("sputum_Xpert", "Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> yield_table1

bind_rows(
  yield_function(cd4_df$anyTBtest_pos, cd4_df$sputum_xpert),
  yield_function(cd4_df$anyTBtest_pos, cd4_df$ALERE_FC),
  yield_function(cd4_df$anyTBtest_pos, cd4_df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("sputum_Xpert", "Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> yield_table_cd4

bind_rows(
  yield_function(hb_df$anyTBtest_pos, hb_df$sputum_xpert),
  yield_function(hb_df$anyTBtest_pos, hb_df$ALERE_FC),
  yield_function(hb_df$anyTBtest_pos, hb_df$bld_xpert_pos)) %>%
```

```
  mutate(index_test =
           c("sputum_Xpert", "Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> yield_table_hb

bind_rows(
  yield_function(lact_df$anyTBtest_pos, lact_df$sputum_xpert),
  yield_function(lact_df$anyTBtest_pos, lact_df$ALERE_FC),
  yield_function(lact_df$anyTBtest_pos, lact_df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("sputum_Xpert", "Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> yield_table_lact

bind_rows(
  yield_function(died_df$anyTBtest_pos, died_df$sputum_xpert),
  yield_function(died_df$anyTBtest_pos, died_df$ALERE_FC),
  yield_function(died_df$anyTBtest_pos, died_df$bld_xpert_pos)) %>%
  mutate(index_test =
           c("sputum_Xpert", "Alere_LAM", "blood_Xpert")) %>%
  select(index_test, everything()) -> yield_table_died
```

Diagnostic yield in whole cohort:

```
yield_table1 %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	N	n_TB	n_true_positive	diag_yield	CI_95
sputum_Xpert	582	447	277	0.62	0.57 to 0.66
Alere_LAM	582	447	190	0.43	0.38 to 0.47
blood_Xpert	582	447	165	0.37	0.32 to 0.42

6.3.1.1 CD4 < 100

Diagnostic yield in CD4 < 100:

```
yield_table_cd4 %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	N	n_TB	n_true_positive	diag_yield	CI_95
sputum_Xpert	375	301	187	0.62	0.56 to 0.68
Alere_LAM	375	301	153	0.51	0.45 to 0.57
blood_Xpert	375	301	149	0.50	0.44 to 0.55

6.3.1.2 Haemoglobin < 8

Diagnostic yield in Hb < 8:

```
yield_table_hb %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	N	n_TB	n_true_positive	diag_yield	CI_95
sputum_Xpert	207	179	115	0.64	0.57 to 0.71
Alere_LAM	207	179	94	0.53	0.45 to 0.6
blood_Xpert	207	179	84	0.47	0.39 to 0.55

6.3.1.3 Lactate > 2.5

Diagnostic yield in lactate > 2.5:

```
yield_table_lact %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

index_test	N	n_TB	n_true_positive	diag_yield	CI_95
sputum_Xpert	142	124	70	0.56	0.47 to 0.65
Alere_LAM	142	124	54	0.44	0.35 to 0.53
blood_Xpert	142	124	62	0.50	0.41 to 0.59

6.3.1.4 By mortality

Diagnostic yield in those who died:

```
yield_table_died %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

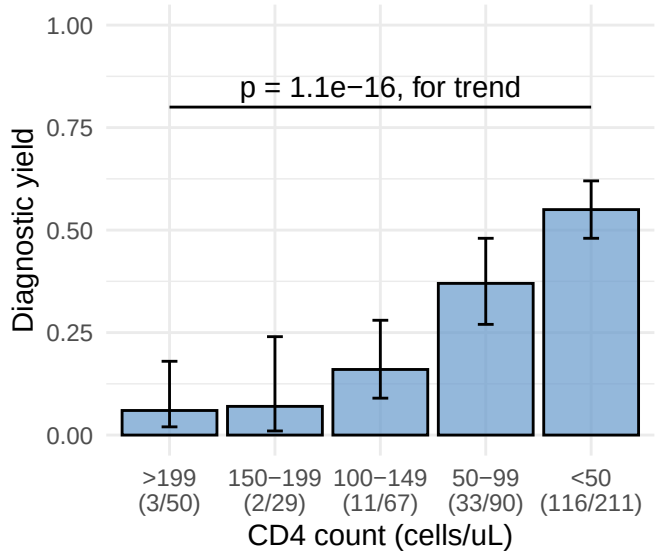
index_test	N	n_TB	n_true_positive	diag_yield	CI_95
sputum_Xpert	123	96	54	0.56	0.46 to 0.66
Alere_LAM	123	96	43	0.45	0.35 to 0.55
blood_Xpert	123	96	51	0.53	0.43 to 0.63

6.4 Diagnostic yield figures

6.4.1 Re-creating the Steve Lawn figure

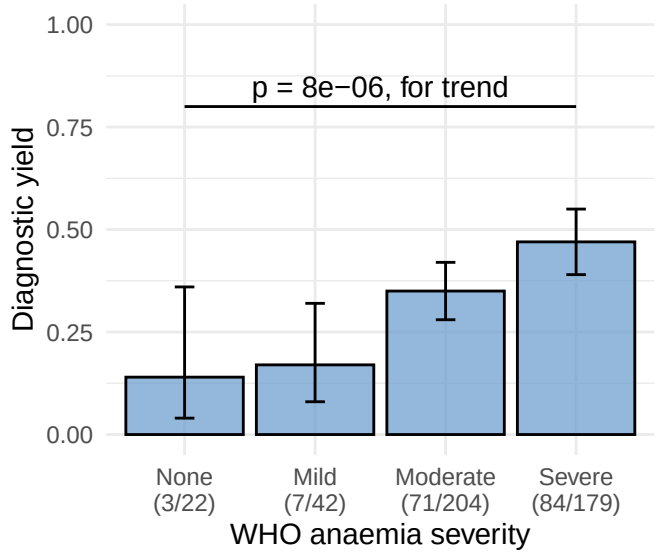
```
# We can re-use the "yield function" from above, but apply it to data sub-setted
# by CD4 count
# have run the function 5 times on each CD4 subset (bin)
# and bind teh resulst together as rows of a new data frame
# at the end also add ("mutate") a few new variables which will help make the later plot
bind_rows(
  yield_function(df$anyTBtest_pos[df$CD4>=200],
                 df$bld_xpert_pos[df$CD4>=200]),

  yield_function(df$anyTBtest_pos[df$CD4<200 & df$CD4>=150],
                 df$bld_xpert_pos[df$CD4<200 & df$CD4>=150]),

  yield_function(df$anyTBtest_pos[df$CD4<150 & df$CD4>=100],
                 df$bld_xpert_pos[df$CD4<150 & df$CD4>=100]),
```

```
  yield_function(df$anyTBtest_pos[df$CD4<100 & df$CD4>=50],
                 df$bld_xpert_pos[df$CD4<100 & df$CD4>=50]),

  yield_function(df$anyTBtest_pos[df$CD4<50],
                 df$bld_xpert_pos[df$CD4<50])) %>%
  mutate(CD4_bin = c(">199", "150-199", "100-149", "50-99", "<50"),
         no_obs = paste0("(",n_true_positive,"/",n_TB,")"),
         CI_95 = as.character(CI_95),
         lwr_95 = as.numeric(sapply(strsplit(CI_95, " to "), '[', 1)),
         upr_95 = as.numeric(sapply(strsplit(CI_95, " to "), '[', 2))) -> cd4_tbl

# pull out the x axis tick labels and format them same as the lawn figure
x_axis_labels <- paste0(cd4_tbl$CD4_bin, "\n", cd4_tbl$no_obs)

# get the p value for the Chi squared test for trend
pvalue <- paste0(
  "p = ",
  signif(prop.trend.test(x = cd4_tbl$n_true_positive,
                 n = cd4_tbl$n_TB)$p.value, digits=2),
  ", for trend")

# make sure R knows we want these in the correct order on the plot
cd4_tbl$CD4_bin <- factor(cd4_tbl$CD4_bin,
                          levels = c(">199", "150-199", "100-149", "50-99", "<50"))

# plot
ggplot(cd4_tbl, aes(CD4_bin, diag_yield)) +
  geom_bar(stat = "identity",
           colour="black", fill="#6699CC", alpha=0.7) +
  geom_errorbar(aes(ymin=lwr_95, ymax=upr_95),
                width=0.15) +
  theme_minimal() +
  scale_x_discrete(labels=x_axis_labels) +
  xlab("CD4 count (cells/uL)") +
  ylab("Diagnostic yield") +
  ylim(0,1) +
  annotate("text", x=3, y=0.85, label=pvalue) +
  annotate("segment", x = 1, xend = 5, y = 0.8, yend = 0.8)
```




```
### Now repeating for haemoglobin

# first make a haemoglobin classification as per WHO
df$anaemia <- "foo"
df$anaemia[df$Sex=="M" & df$Haemoglobin>=13] <- "None"
df$anaemia[df$Sex=="F" & df$Haemoglobin>=12] <- "None"
df$anaemia[df$Sex=="M" & df$Haemoglobin<13 & df$Haemoglobin>=11] <- "Mild"
df$anaemia[df$Sex=="F" & df$Haemoglobin<12 & df$Haemoglobin>=11] <- "Mild"
df$anaemia[df$Haemoglobin<11 & df$Haemoglobin>=8] <- "Moderate" #both sexes
df$anaemia[df$Haemoglobin<8] <- "Severe" #both sexes

bind_rows(
  yield_function(df$anyTBtest_pos[df$anaemia=="None"],
                 df$bld_xpert_pos[df$anaemia=="None"]),

  yield_function(df$anyTBtest_pos[df$anaemia=="Mild"],
                 df$bld_xpert_pos[df$anaemia=="Mild"]),

  yield_function(df$anyTBtest_pos[df$anaemia=="Moderate"],
                 df$bld_xpert_pos[df$anaemia=="Moderate"]),

  yield_function(df$anyTBtest_pos[df$anaemia=="Severe"],
                 df$bld_xpert_pos[df$anaemia=="Severe"])) %>%

  mutate(hb_bin = c("None", "Mild", "Moderate", "Severe"),
         no_obs = paste0("(",n_true_positive,"/",n_TB,")"),
         CI_95 = as.character(CI_95),
         lwr_95 = as.numeric(sapply(strsplit(CI_95, " to "), '[', 1)),
         upr_95 = as.numeric(sapply(strsplit(CI_95, " to "), '[', 2))) -> hb_tbl

# pull out the x axis tick labels and format them same as the lawn figure
x_axis_labels <- paste0(hb_tbl$hb_bin, "\n", hb_tbl$no_obs)

# get the p value for the Chi squared test for trend
pvalue <- paste0(
```

```
  "p = ",
  signif(prop.trend.test(x = hb_tbl$n_true_positive,
                n = hb_tbl$n_TB)$p.value, digits=2),
  ", for trend")

# make sure R knows we want these in the correct order on the plot
hb_tbl$hb_bin <- factor(hb_tbl$hb_bin,
                        levels = c("None", "Mild", "Moderate", "Severe"))

# plot
ggplot(hb_tbl, aes(hb_bin, diag_yield)) +
  geom_bar(stat = "identity",
           colour="black", fill="#6699CC", alpha=0.7) +
  geom_errorbar(aes(ymin=lwr_95, ymax=upr_95),
                width=0.15) +
  theme_minimal() +
  scale_x_discrete(labels=x_axis_labels) +
  xlab("WHO anaemia severity") +
  ylab("Diagnostic yield") +
  ylim(0,1) +
  annotate("text", x=2.5, y=0.85, label=pvalue) +
  annotate("segment", x = 1, xend = 4, y = 0.8, yend = 0.8)
```



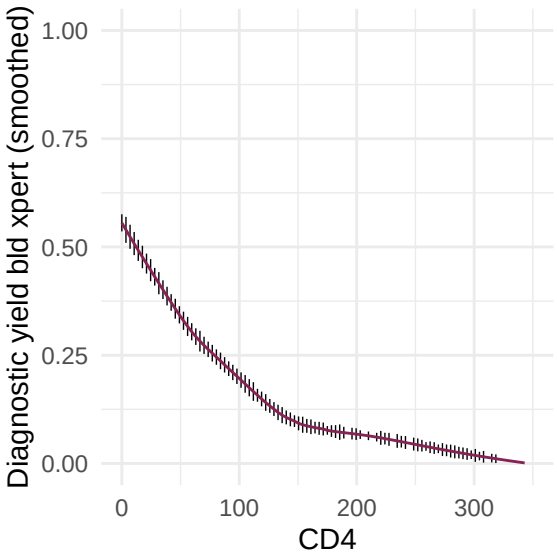
6.4.2 Alternative plots

The categories in plots above are arbitrary, and imbalanced (some have few patients, others many). This is inefficient use of the data for estimating precision and shape of relationship between predictor (CD4, Hb...) and diagnostic yield. Also, if you want to show more than one test (eg compare sputum and blood Xpert, or combinations), this requires more plots. Alternative is to model the realtionship as two continuous variables, rather than binning the predictor variable into ordered categories.

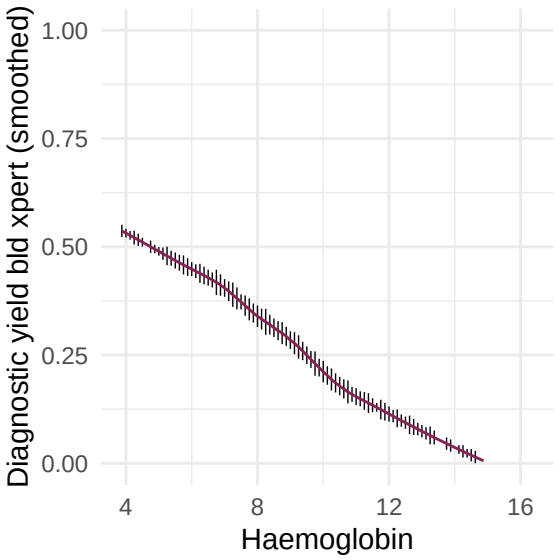
6.4.2.1 Frank Harrell's Hmisc package has these "spike histogram" plots The red line is the smoothed (Loess) relationship between CD4 (or Hb) and diagnostic yield of blood Xpert. The little back

verticle lines “spikes” are histogram of frequencies at different CD4 counts, giving an idea ho wwell the line is supported by data in agiven range.

```
df$bld_xpert_diagnosed <- df$bld_xpert_pos=="MTB" & !is.na(df$bld_xpert_pos)

ggplot(df, aes(CD4, bld_xpert_diagnosed, colour=as.factor(1))) +
  histSpikeg(bld_xpert_diagnosed ~ CD4, lowess=TRUE,
             data=df) +
  ylim(0,1) +
  theme_minimal() +
  scale_colour_manual(values = "#882255") +
  theme(legend.position = "none") +
  ylab("Diagnostic yield bld xpert (smoothed)")
```




```
ggplot(df, aes(Haemoglobin, bld_xpert_diagnosed, colour=as.factor(1))) +
  histSpikeg(bld_xpert_diagnosed ~ Haemoglobin, lowess=TRUE,
             data=df) +
  ylim(0,1) +
  theme_minimal() +
  scale_colour_manual(values = "#882255") +
  theme(legend.position = "none") +
  ylab("Diagnostic yield bld xpert (smoothed)")
```



6.4.2.2 Similar idea but replace the spikes with conf intervals for the model Takes us a step further from the raw data but gives more flexibility in presentation. Still using a Loess smoothing function for the model, similar to the Harrell plots.

For each of the three diagnostic tests (left column) and four test combinations (right column) the diagnostic yield is modelled by a a dependent variable (CD4, haemoglobin, lactate; top, middle and bottom row) with a loess smoothing function. 95% confidence intervals derived from 1000 bootstraps of each model ((3 tests + 4 combos) * 3 dependent variables = 21 models, each bootstrapped 1000 times).

```
# a dataframe with just the TB patients (as per diagnostic yield analysis definition)
dftb <- df[df$anyTBtest_pos, ] # n=447

# set up the dummy variables
dftb$sputum_xpert_diagnosed <- dftb$sputum_xpert=="MTB" & !is.na(dftb$sputum_xpert)
dftb$urine_LAM_diagnosed <- dftb$ALERE_FC=="MTB" & !is.na(dftb$ALERE_FC)
# COMBINATIONS
dftb$sputum_or_ulam <-
  dftb$sputum_xpert_diagnosed | dftb$urine_LAM_diagnosed
dftb$sputum_or_bldx <-
  dftb$sputum_xpert_diagnosed | dftb$bld_xpert_diagnosed
dftb$bldx_or_ulam <-
  dftb$urine_LAM_diagnosed | dftb$bld_xpert_diagnosed
dftb$sputum_ulam_bldx <-
  dftb$sputum_xpert_diagnosed |
  dftb$urine_LAM_diagnosed |
  dftb$bld_xpert_diagnosed


### BOOTSTRAPPING RESULTS

# a new data frame to get predictions on
newdata <- data.frame(CD4 = seq(0,300,length.out = 20),
                      lactate = seq(1, 10, length.out = 20),
                      Haemoglobin = seq(3, 12, length.out = 20))
```

```r
# Funcion to apply in the bootstrap
f1 <- function(data, indicies, formula, span = 0.8, newdata){
  d <- data[indicies,]
  m <- loess(formula, data=d, span = span)
  preds <- predict(m, newdata=newdata, type = "response")
  return(preds)
}

# function to summaries the bootstrap results
sumBoot <- function(boot_data) {
  return(
    data.frame(lwr = apply(boot_data, 2,
                           function(x) as.numeric(
                             quantile(x, probs=0.025, na.rm = TRUE))),
               fit = apply(boot_data, 2,
                           function(x) as.numeric(
                             quantile(x, probs=0.5, na.rm = TRUE))),
               upr = apply(boot_data, 2,
                           function(x) as.numeric(
                             quantile(x, probs=0.975, na.rm = TRUE)))
    )
  )
}


set.seed(2212)

# run and summarise boot for each model we need,
# bind them all into one dataframe

### CD4

bind_rows(

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = bld_xpert_diagnosed ~ CD4,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = sputum_xpert_diagnosed ~ CD4,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = urine_LAM_diagnosed ~ CD4,
     newdata=newdata,
     R=10)$t),

  sumBoot(
```

```
  boot(data=dftb, statistic = f1,
     formula = sputum_or_ulam ~ CD4,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = sputum_or_bldx ~ CD4,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = bldx_or_ulam ~ CD4,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = sputum_ulam_bldx ~ CD4,
     newdata=newdata,
     R=10)$t)

  ) -> boot_cd4

boot_cd4$value <- rep(seq(0,300,length.out = 20), 7)

boot_cd4$diagnostic <- rep(
  c("Bld Xpert",
  "Sptm Xpert",
  "uLAM",
  "Sptm Xpert + uLAM",
  "Sptm Xpert + Bld Xpert",
  "Bld Xpert + uLAM",
  "All 3 tests"),
  each=20
)

boot_cd4$var <- rep("CD4", nrow(boot_cd4))

boot_cd4$panel <- factor(
  c(rep("Single tests", 3*20), rep("Combinations", 4*20)),
  levels = c("Single tests", "Combinations"))


### hb

bind_rows(

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = bld_xpert_diagnosed ~ Haemoglobin,
     newdata=newdata,
```

```
    R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
    formula = sputum_xpert_diagnosed ~ Haemoglobin,
    newdata=newdata,
    R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
    formula = urine_LAM_diagnosed ~ Haemoglobin,
    newdata=newdata,
    R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
    formula = sputum_or_ulam ~ Haemoglobin,
    newdata=newdata,
    R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
    formula = sputum_or_bldx ~ Haemoglobin,
    newdata=newdata,
    R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
    formula = bldx_or_ulam ~ Haemoglobin,
    newdata=newdata,
    R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
    formula = sputum_ulam_bldx ~ Haemoglobin,
    newdata=newdata,
    R=10)$t)

  ) -> boot_Haemoglobin

boot_Haemoglobin$value <- rep(seq(3,12,length.out = 20), 7)

boot_Haemoglobin$diagnostic <- rep(
  c("Bld Xpert",
  "Sptm Xpert",
  "uLAM",
  "Sptm Xpert + uLAM",
  "Sptm Xpert + Bld Xpert",
  "Bld Xpert + uLAM",
  "All 3 tests"),
  each=20
)
```

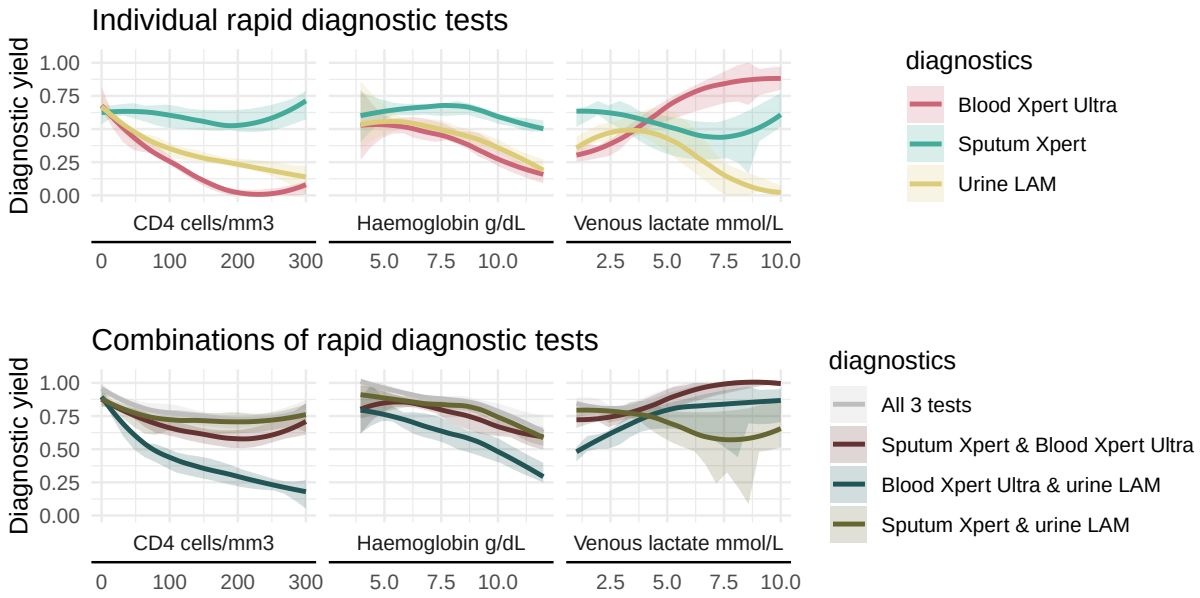
```r
boot_Haemoglobin$var <- rep("Haemoglobin", nrow(boot_Haemoglobin))

boot_Haemoglobin$panel <- factor(
  c(rep("Single tests", 3*20), rep("Combinations", 4*20)),
  levels = c("Single tests", "Combinations"))

### LACTATE

bind_rows(

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = bld_xpert_diagnosed ~ lactate,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = sputum_xpert_diagnosed ~ lactate,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = urine_LAM_diagnosed ~ lactate,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = sputum_or_ulam ~ lactate,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = sputum_or_bldx ~ lactate,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = bldx_or_ulam ~ lactate,
     newdata=newdata,
     R=10)$t),

  sumBoot(
  boot(data=dftb, statistic = f1,
     formula = sputum_ulam_bldx ~ lactate,
     newdata=newdata,
     R=10)$t)

  ) -> boot_lactate
```

```r
boot_lactate$value <- rep(seq(1,10,length.out = 20), 7)


boot_lactate$diagnostic <- rep(
  c("Bld Xpert",
  "Sptm Xpert",
  "uLAM",
  "Sptm Xpert + uLAM",
  "Sptm Xpert + Bld Xpert",
  "Bld Xpert + uLAM",
  "All 3 tests"),
  each=20
)

boot_lactate$var <- rep("lactate", nrow(boot_lactate))

boot_lactate$panel <- factor(
  c(rep("Single tests", 3*20), rep("Combinations", 4*20)),
  levels = c("Single tests", "Combinations"))

boot_df <- bind_rows(boot_cd4, boot_Haemoglobin, boot_lactate)
```

```r
boot_df %>%
  mutate(diagnostics = case_when(
    diagnostic == "Bld Xpert" ~ "Blood Xpert Ultra",
    diagnostic == "Sptm Xpert"~ "Sputum Xpert",
    diagnostic == "uLAM" ~ "Urine LAM",
    diagnostic == "Sptm Xpert + uLAM" ~ "Sputum Xpert & urine LAM",
    diagnostic == "Sptm Xpert + Bld Xpert" ~ "Sputum Xpert & Blood Xpert Ultra",
    diagnostic == "Bld Xpert + uLAM" ~ "Blood Xpert Ultra & urine LAM",
    diagnostic == "All 3 tests" ~ "All 3 tests"
  )) %>%
  mutate(var = case_when(
    var == "CD4" ~ "CD4 cells/mm3",
    var == "Haemoglobin" ~ "Haemoglobin g/dL",
    var == "lactate" ~ "Venous lactate mmol/L"
  )) %>%
  mutate(
    lwr = ifelse(lwr<0, 0, lwr),
    fit = ifelse(fit>1, 1, fit),
    upr = ifelse(upr>1, 1, upr)
  )-> boot_df

boot_df %>%
  filter(panel=="Single tests") %>%
  ggplot(aes(value, fit, colour=diagnostics)) +
  geom_ribbon(aes(ymin=lwr, ymax=upr, fill=diagnostics),
              alpha=0.2, linetype=0) +
  geom_smooth(se=FALSE) +
  theme_minimal() +
  theme(axis.line.x.bottom = element_line(colour = "black")) +
  ylim(0,1.05) +
  scale_colour_manual(values = c(
```

```
    "#CC6677", "#44AA99", "#DDCC77"
    )) +
  scale_fill_manual(values = c(
    "#CC6677", "#44AA99", "#DDCC77"
  )) +
  facet_wrap(~var, scales = "free_x", nrow = 1, switch = "x") +
  ylab("Diagnostic yield") + xlab("") +
  ggtitle("Individual rapid diagnostic tests") -> g_singles

boot_df %>%
  filter(panel=="Combinations") %>%
  mutate(
    diagnostics = factor(
      diagnostics,
      levels = c(
        "All 3 tests",
        "Sputum Xpert & Blood Xpert Ultra",
        "Blood Xpert Ultra & urine LAM",
        "Sputum Xpert & urine LAM"
      )
    )
  ) %>%
  ggplot(aes(value, fit, colour=diagnostics)) +
  geom_ribbon(aes(ymin=lwr, ymax=upr, fill=diagnostics),
              alpha=0.2, linetype=0) +
  geom_smooth(se=FALSE) +
  theme_minimal() +
  theme(axis.line.x.bottom = element_line(colour = "black")) +
  ylim(0,1.05) +
  scale_colour_manual(values = c(
    "grey", "#663333", "#225555", "#666633"
    )) +
  scale_fill_manual(values = c(
    "grey", "#663333", "#225555", "#666633"
  )) +
  facet_wrap(~var, scales = "free_x", nrow = 1, switch = "x") +
  ylab("Diagnostic yield") + xlab("") +
  ggtitle("Combinations of rapid diagnostic tests") -> g_combos
```

```
g_singles / g_combos
```



7 Blood Xpert cycle threshold & mortality risk

This analysis is limited to patients with confirmed TB, defined using the "any TB test positive" variable.

7.1 Imputing Ct values for trace positive samples

To determine the semi-quantitative readout result ("very low", "low", "medium", "high"), Xpert software uses the minimum Ct value from the 4 rpoB probes when reporting a positive sample. Trace positive samples are those where the IS1081_IS6110 probe was positive but all rpoB probes negative. Since IS1081_IS6110 is multi-copy per genome, it may not be reliable as rpoB Ct values to quantify bacilli. However, as shown below correlation between minimum rpoB Ct value and IS1081_IS6110 Ct value is quite strong: within sample, IS1081_IS6110 Ct is a reliable predictor of rpoB CT; this may not be true across different settings or studies. Therefore, we use IS1081_IS6110 Ct value to impute the unobserved minimum rpoB Ct value in 'trace' positive samples.

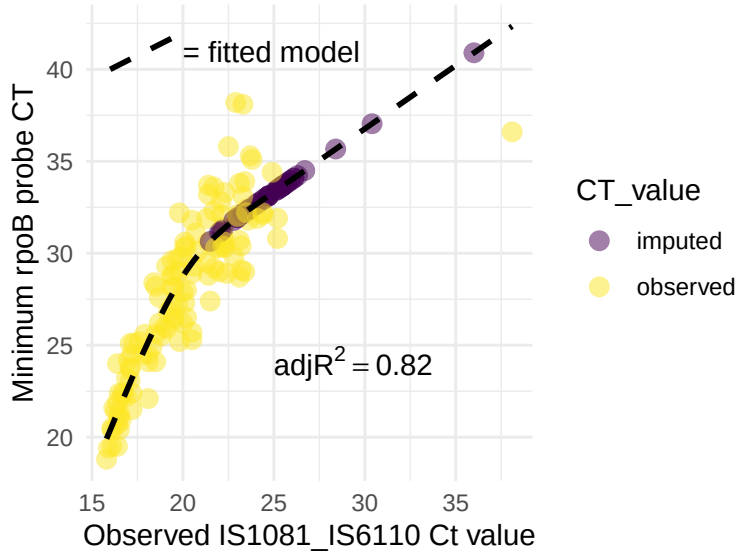
```
df %>%
  select(blood_Xpert_MTB) %>%
  group_by(blood_Xpert_MTB) %>%
  mutate(blood_Xpert_MTB = factor(
    blood_Xpert_MTB,
    levels = c("High", "Medium", "Low",
               "Very low", "Trace",
               "Negative", "Error")
  )) %>%
  count() %>%
  arrange(blood_Xpert_MTB) %>%
  kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

blood_Xpert_MTB	n
High	1
Medium	32
Low	34
Very low	57
Trace	41
Negative	413
Error	4

```
# missing values are coded as zero - fix this
df$rpoB1[df$rpoB1==0] <- NA
df$rpoB2[df$rpoB2==0] <- NA
df$rpoB3[df$rpoB3==0] <- NA
df$rpoB4[df$rpoB4==0] <- NA
df$IS1081_IS6110[df$IS1081_IS6110==0] <- NA

# get the minimum rpoB probe Ct value
df %>%
  rowwise() %>%
  mutate(min_rpoB_CT =
           min(rpoB1, rpoB2, rpoB3, rpoB4, na.rm = TRUE)) -> df
# quick fix the 'infinite' values which NAs have been turned into
df$min_rpoB_CT[is.infinite(df$min_rpoB_CT)] <- NA

# exclude extreme outliers for the imputation model
foo <- df[df$IS1081_IS6110<35,]
# fit restricted cubic spline model
fit1 <- lm(min_rpoB_CT ~ rcs(IS1081_IS6110,c(16,20,24)), data=foo)
rm(foo)
# get the model fit statistic for later use
R2 <- round(summary(fit1)$adj.r.squared, 2)
# use teh model to get predicted values of minimum rpoB CT
imputed_CT <- predict(fit1, newdata = df)

# call this something simpler
df$blood_Xpert_CT <- df$min_rpoB_CT

# this is just to help with the later graph
df$CT_value <- NA
df$CT_value[is.na(df$min_rpoB_CT) & !is.na(df$IS1081_IS6110)] <- "imputed"
df$CT_value[!is.na(df$min_rpoB_CT)] <- "observed"

# samples which dont have a rpoB Ct value but do have an IS1081_IS6110 Ct value
# (which are trace positive samples) get an imputed Ct value, to make our final
# Ct value "blood_xpert_CT"
df$blood_Xpert_CT[df$CT_value=="imputed" &
                    !is.na(df$CT_value)] <- imputed_CT[df$CT_value=="imputed" &
                                                         !is.na(df$CT_value)]

df$imputed_CT <- imputed_CT
```

Here are the observed IS1081_IS6110 Ct values versus the minimum rpoB Ct values, either oberved (yellow points) or imputed (purple) using the model. The model fit is shown as dashed line (all imputed points therefore lie on this line).

```
ggplot(df[!is.na(df$IS1081_IS6110), ],
       aes(IS1081_IS6110, blood_Xpert_CT)) +
  geom_point(aes(colour=CT_value), alpha=0.5, size=3) +
  geom_line(aes(IS1081_IS6110, imputed_CT), size=1, linetype=2) +
  theme_minimal() +
  scale_colour_viridis_d() +
  xlab("Observed IS1081_IS6110 Ct value") +
  ylab("Minimum rpoB probe CT") +
  annotate("segment", x = 16, xend = 20, y = 40, yend = 42, size=1, linetype=2) +
  annotate("text", x=20, y=41, label="= fitted model", hjust=0) +
  annotate("text", x=25, y=24, label=paste0( "adjR^2 == ", R2 ), parse = TRUE, hjust=0)
```



Using these values for all subsequent analysis.

7.2 Visualising blood Xpert Ct v mortality risk

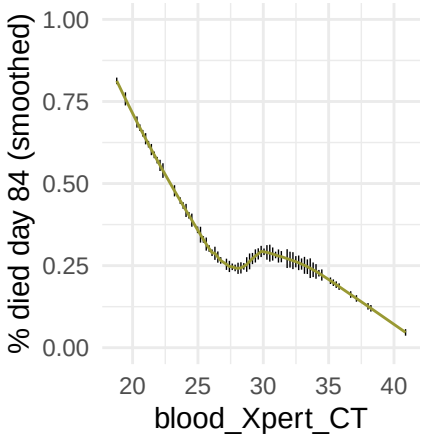
7.2.1 Ct values treated as continuous variable

```
### Making a clean df for TB cases only

# Set up "end date" correctly for KM plots etc
df$dateDeath.x <- as.Date(df$dateDeath.x, format="%d/%m/%Y")
df$LTFU.censor.date <- as.Date(df$LTFU.censor.date, format="%d/%m/%Y")
df$StudyDate  <- as.Date(df$StudyDate, format="%d/%m/%Y")

# these are miscoded as mssing - not sure why - manually pulled from KDHTB database
df$dateDeath.x[df$UID==480] <- "2016-01-26"
df$dateDeath.x[df$UID==485] <- "2016-03-15"
df$dateDeath.x[df$UID==490] <- "2016-02-14"
df$dateDeath.x[df$UID==498] <- "2016-02-09"
df$dateDeath.x[df$UID==263] <- "2015-06-02"
```

```
df$endDate <- as.Date("1900-01-01")
df$endDate[df$survival.12weeks=="LTFU"] <-
  df$LTFU.censor.date[df$survival.12weeks=="LTFU"]

df$endDate[df$survival.12weeks=="Survived"] <-
  df$StudyDate[df$survival.12weeks=="Survived"] + 84

df$endDate[df$survival.12weeks=="Died"] <-
  df$dateDeath.x[df$survival.12weeks=="Died"]

# follow up time variable
df$time <- as.numeric(df$endDate - df$StudyDate)

df$day84outcome <- df$survival.12weeks
df$day84death <- df$day84outcome=="Died"
df$day84death[df$day84outcome=="LTFU"] <- NA

# make dataframe with only the confirmed Tb cases

tbdf <- df[df$anyTBtest_pos==TRUE, ]
```

Proportion of patients dying as a function of blood Xpert Ct value. The coloured line is a loess smoothing function; number of patients at each Ct value shown by the height of the little verticle black lines. These plots are a way of seeing the 'shape' of the relationship between Ct value and risk of death - not forcing the relationship to be linear.

```
ggplot(tbdf, aes(blood_Xpert_CT, day84death, colour=as.factor(1))) +
  histSpikeg(day84death ~ blood_Xpert_CT, lowess=TRUE,
             data=tbdf) +
  ylim(0,1) +
  theme_minimal() +
  scale_colour_manual(values = "#999933") +
  theme(legend.position = "none") +
  ylab("% died day 84 (smoothed)")
```



Borrowing the idea of the harrell style plot here, will fit LOESS models to continuous data (Ct or TTP) versus mortality, and generate 95%CI for these using bootstraps.

```r
# Funcion to apply in the bootstrap
f1 <- function(data, indicies, formula, span = 1.2, newdata){
  d <- data[indicies,]
  m <- loess(formula, data=d, span = span)
  preds <- predict(m, newdata=newdata, type = "response")
  return(preds)
}

# function to summaries the bootstrap results
sumBoot <- function(boot_data) {
  return(
    data.frame(lwr = apply(boot_data, 2,
                           function(x) as.numeric(
                             quantile(x, probs=0.025, na.rm = TRUE))),
               fit = apply(boot_data, 2,
                           function(x) as.numeric(
                             quantile(x, probs=0.5, na.rm = TRUE))),
               upr = apply(boot_data, 2,
                           function(x) as.numeric(
                             quantile(x, probs=0.975, na.rm = TRUE)))
    )
  )
}

set.seed(2212)

### Blood Xpert CT value
# a new data frame to get predictions on
newdata <- data.frame(
  blood_Xpert_CT = seq(min(tbdf$blood_Xpert_CT, na.rm = TRUE),
                       max(tbdf$blood_Xpert_CT, na.rm = TRUE),
                       length.out = 500))
# run and summarise boot for model
sumBoot(
  boot(
    data = tbdf,
    statistic = f1,
    formula = day84death ~ blood_Xpert_CT,
    newdata = newdata,
    R = 1000
  )$t
) %>% bind_cols(newdata) -> bx_loess_df

bx_loess_df %>%
  mutate(
    lwr = ifelse(lwr<0, 0, lwr),
    upr = ifelse(upr>1, 1, upr)
  ) %>%
  ggplot(
    aes(blood_Xpert_CT, fit,
        ymin=lwr, ymax=upr)
  ) +
  geom_ribbon(aes(ymin=lwr, ymax=upr),
```

```
              alpha=0.2, linetype=0, fill="#453781FF") +
  geom_smooth(se=FALSE, colour="#453781FF") +
  theme_minimal() +
  theme(axis.line.x.bottom = element_line(colour = "black")) +
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) +
  xlab("Blood Xpert-ultra Ct") +
  ylab("Proportion died") -> g_bxct_loess


### urine Xpert CT value
# a new data frame to get predictions on
newdata <- data.frame(
  min.ct_urineGXP = seq(min(tbdf$min.ct_urineGXP, na.rm = TRUE),
                        max(tbdf$min.ct_urineGXP, na.rm = TRUE),
                        length.out = 500))
# run and summarise boot for model
sumBoot(
  boot(
    data = tbdf,
    statistic = f1,
    formula = day84death ~ min.ct_urineGXP,
    newdata = newdata,
    R = 1000
  )$t
) %>% bind_cols(newdata) -> ux_loess_df

ux_loess_df %>%
  mutate(
    lwr = ifelse(lwr<0, 0, lwr),
    upr = ifelse(upr>1, 1, upr)
  ) %>%
  ggplot(
    aes(min.ct_urineGXP, fit,
        ymin=lwr, ymax=upr)
  ) +
  geom_ribbon(aes(ymin=lwr, ymax=upr),
              alpha=0.2, linetype=0, fill="#3CBB75FF") +
  geom_smooth(se=FALSE, colour="#3CBB75FF") +
  theme_minimal() +
  theme(axis.line.x.bottom = element_line(colour = "black")) +
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) +
  xlab("Urine Xpert Ct") +
  ylab("Proportion died") -> g_uxct_loess


### blood culture TTP value
# a new data frame to get predictions on
newdata <- data.frame(
  MBC1_TTP = seq(12,
                 42,
                 length.out = 500))
# run and summarise boot for model
sumBoot(
```

```
  boot(
    data = tbdf,
    statistic = f1,
    formula = day84death ~ MBC1_TTP,
    newdata = newdata,
    R = 1000
  )$t
) %>% bind_cols(newdata) -> bc_loess_df

bc_loess_df %>%
  mutate(
    lwr = ifelse(lwr<0, 0, lwr),
    upr = ifelse(upr>1, 1, upr)
  ) %>%
  ggplot(
    aes(MBC1_TTP, fit,
        ymin=lwr, ymax=upr)
  ) +
  geom_ribbon(aes(ymin=lwr, ymax=upr),
              alpha=0.2, linetype=0, fill="#287D8EFF") +
  geom_smooth(se=FALSE, colour="#287D8EFF") +
  theme_minimal() +
  theme(axis.line.x.bottom = element_line(colour = "black")) +
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) +
  xlab("TB blood culture TTP") +
  ylab("Proportion died") -> g_bcttp_loess


### sputum Xpert CT value
# a new data frame to get predictions on
newdata <- data.frame(
  min.ct_sptmGXP = seq(min(tbdf$min.ct_sptmGXP, na.rm = TRUE),
                       max(tbdf$min.ct_sptmGXP, na.rm = TRUE),
                       length.out = 500))
# run and summarise boot for model
sumBoot(
  boot(
    data = tbdf,
    statistic = f1,
    formula = day84death ~ min.ct_sptmGXP,
    newdata = newdata,
    R = 1000
  )$t
) %>% bind_cols(newdata) -> sx_loess_df

sx_loess_df %>%
  mutate(
    lwr = ifelse(lwr<0, 0, lwr),
    upr = ifelse(upr>1, 1, upr)
  ) %>%
  ggplot(
    aes(min.ct_sptmGXP, fit,
        ymin=lwr, ymax=upr)
```

```
  ) +
  geom_ribbon(aes(ymin=lwr, ymax=upr),
              alpha=0.2, linetype=0, fill="#DCE319FF") +
  geom_smooth(se=FALSE, colour="#DCE319FF") +
  theme_minimal() +
  theme(axis.line.x.bottom = element_line(colour = "black")) +
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) +
  xlab("Sputum Xpert Ct") +
  ylab("Proportion died") -> g_sxct_loess
```

To go with these LOESS plots, make mosaic plots comparing qualitative results of the rapid diagnostic tests (positive or negative) versus mortality. Do this in ggplot but is equivalent to plot(table(x,y)).

```
tbdf %>%
  filter(day84outcome!="LTFU" & blood_Xpert_MTB!="Error") %>%
  select(day84outcome, bld_xpert_pos) %>%
  group_by(bld_xpert_pos, day84outcome) %>%
  summarise(count = n()) %>%
  ungroup() %>% group_by(bld_xpert_pos) %>%
  mutate(
    test_n = sum(count),
    prop = count/sum(count),
    bld_xpert_pos = factor(bld_xpert_pos,
                           levels = c("NEG", "MTB"),
                           labels = c("Neg", "Pos")),
    day84outcome = factor(day84outcome,
                          levels = c("Survived", "Died"))
    ) %>%
  ungroup() %>% { . ->> tmp } %>%
  ggplot(
    aes(x=bld_xpert_pos, y=prop,
        width=test_n, fill=day84outcome)
    ) +
  geom_bar(stat = "identity", position = "fill", colour = "white") +
  facet_grid(~bld_xpert_pos, scales = "free_x", space = "free_x") +
  theme_minimal() +
  theme(legend.position = "none",
        panel.spacing.x = unit(0, "npc"),
        strip.background = element_blank(),
        strip.text.x = element_blank(),
        axis.line.x.bottom = element_line(colour = "black")) +
  ylab("Proportion died") +
  xlab("Blood Xpert-ultra") +
  scale_fill_manual(values = c("#4537814C", "black")) +
  geom_text(aes(label=count, colour=day84outcome),
            position = position_stack(vjust = 0.5), size=3.5) +
  scale_colour_manual(values = c("black", "white")) +
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) -> g_bxct_2x2



tbdf %>%
  filter(day84outcome!="LTFU" & !is.na(uGXP)) %>%
  select(day84outcome, uGXP) %>%
```

```
  group_by(uGXP, day84outcome) %>%
  summarise(count = n()) %>%
  ungroup() %>% group_by(uGXP) %>%
  mutate(
    test_n = sum(count),
    prop = count/sum(count),
    uGXP = factor(uGXP,
                          levels = c("NEG", "MTB"),
                          labels = c("Neg", "Pos")),
    day84outcome = factor(day84outcome,
                          levels = c("Survived", "Died"))
    ) %>%
  ungroup() %>%
  ggplot(
    aes(x=uGXP, y=prop,
        width=test_n, fill=day84outcome)
    ) +
  geom_bar(stat = "identity", position = "fill", colour = "white") +
  facet_grid(~uGXP, scales = "free_x", space = "free_x") +
  theme_minimal() +
  theme(legend.position = "none",
        panel.spacing.x = unit(0, "npc"),
        strip.background = element_blank(),
        strip.text.x = element_blank(),
        axis.line.x.bottom = element_line(colour = "black")) +
  ylab("Proportion died") +
  xlab("Urine Xpert") +
  scale_fill_manual(values = c("#3CBB754C", "black")) +
  geom_text(aes(label=count, colour=day84outcome),
            position = position_stack(vjust = 0.5), size=3.5) +
  scale_colour_manual(values = c("black", "white")) +
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) -> g_uxct_2x2


tbdf %>%
  filter(day84outcome!="LTFU" &
           (MBC1_cultureID=="MTB" | MBC1_cultureID=="negative")) %>%
  select(day84outcome, MBC1_cultureID) %>%
  group_by(MBC1_cultureID, day84outcome) %>%
  summarise(count = n()) %>%
  ungroup() %>% group_by(MBC1_cultureID) %>%
  mutate(
    test_n = sum(count),
    prop = count/sum(count),
    MBC1_cultureID = factor(MBC1_cultureID,
                          levels = c("negative", "MTB"),
                          labels = c("Neg", "Pos")),
    day84outcome = factor(day84outcome,
                          levels = c("Survived", "Died"))
    ) %>%
  ungroup() %>%
  ggplot(
    aes(x=MBC1_cultureID, y=prop,
```

```
        width=test_n, fill=day84outcome)
    ) +
  geom_bar(stat = "identity", position = "fill", colour = "white") +
  facet_grid(~MBC1_cultureID, scales = "free_x", space = "free_x") +
  theme_minimal() +
  theme(legend.position = "none",
        panel.spacing.x = unit(0, "npc"),
        strip.background = element_blank(),
        strip.text.x = element_blank(),
        axis.line.x.bottom = element_line(colour = "black")) +
  ylab("Proportion died") +
  xlab("TB blood culture") +
  scale_fill_manual(values = c("#287D8E4C", "black")) +
  geom_text(aes(label=count, colour=day84outcome),
            position = position_stack(vjust = 0.5), size=3.5) +
  scale_colour_manual(values = c("black", "white")) +
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) -> g_bcttp_2x2


tbdf %>%
  filter(day84outcome!="LTFU" & !is.na(sputum_xpert)) %>%
  select(day84outcome, sputum_xpert) %>%
  group_by(sputum_xpert, day84outcome) %>%
  summarise(count = n()) %>%
  ungroup() %>% group_by(sputum_xpert) %>%
  mutate(
    test_n = sum(count),
    prop = count/sum(count),
    sputum_xpert = factor(sputum_xpert,
                          levels = c("NEG", "MTB"),
                          labels = c("Neg", "Pos")),
    day84outcome = factor(day84outcome,
                          levels = c("Survived", "Died"))
    ) %>%
  ungroup() %>%
  ggplot(
    aes(x=sputum_xpert, y=prop,
        width=test_n, fill=day84outcome)
    ) +
  geom_bar(stat = "identity", position = "fill", colour = "white") +
  facet_grid(~sputum_xpert, scales = "free_x", space = "free_x") +
  theme_minimal() +
  theme(legend.position = "none",
        panel.spacing.x = unit(0, "npc"),
        strip.background = element_blank(),
        strip.text.x = element_blank(),
        axis.line.x.bottom = element_line(colour = "black")) +
  ylab("Proportion died") +
  xlab("Sputum Xpert") +
  scale_fill_manual(values = c("#DCE3194C", "black")) +
  geom_text(aes(label=count, colour=day84outcome),
            position = position_stack(vjust = 0.5), size=3.5) +
  scale_colour_manual(values = c("black", "white")) +
```

```
  scale_y_continuous(breaks = seq(0,1,by=0.25), limits = c(0,1)) -> g_sxct_2x2
```

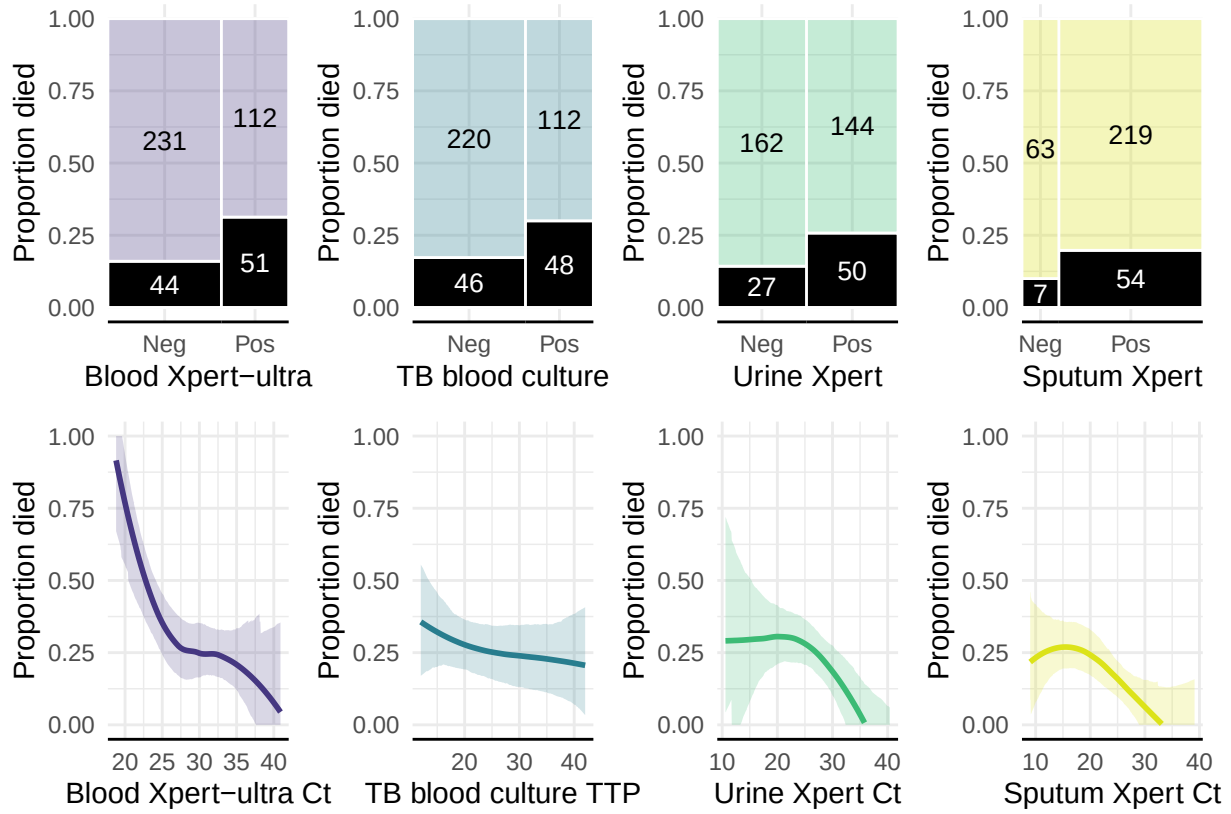
Perform statistical tests (based on linear glms) to compliment the bootstrapped LOESS and mosaic plots made above:

```
# A function to fit glm model for binary outcome ~ cpntinuous predictor
# and extract the beta coefficient (effect size) estimate corresponding
# to a one IQR change in teh continuous predictor (here coded as a one
# IQR *decrease* in ttp or Ct)

iqr_or = function(var){
  x <- tbdf[[var]]
  iqr = IQR(x, na.rm = T)
  m = glm(day84death ~ x, data=tbdf, family=binomial)
  c(
    exp(iqr * (-1 * summary(m)$coefficients[2])),
    summary(m)$coefficients[8]
    )
}

# apply this function to each predictor and store them in a df
data.frame(
  diagnostic = rep(
    c(
      "blood_Xpert_CT",
      "MBC1_TTP",
      "min.ct_urineGXP",
      "min.ct_sptmGXP"
    ),
    each = 2
  ),
  name = rep(c("iqr_OR", "p_value"), 4),
  value = c(
    iqr_or(var = "blood_Xpert_CT"),
    iqr_or(var = "MBC1_TTP"),
    iqr_or(var = "min.ct_urineGXP"),
    iqr_or(var = "min.ct_sptmGXP")
  )
) %>%
  pivot_wider(names_from = name, values_from = value) -> ct_ttp_iqr_or_table

# A function to fit glm model for binary outcome ~ binary (dummy) predictor
# and extract the beta coefficient (effect size) estimate
dummy_or = function(var){
  x <- tbdf[[var]]=="MTB"
  m = glm(day84death ~ x, data=tbdf, family=binomial)
  c(
    exp(summary(m)$coefficients[2]),
    summary(m)$coefficients[8]
    )
}

# apply this function to each predictor and store them in a df
data.frame(
```

```
  diagnostic = rep(
    c(
      "blood_Xpert_ultra",
      "TB_blood_culture",
      "urine_xpert",
      "sputum_xpert"
    ),
    each = 2
  ),
  name = rep(c("OR", "p_value"), 4),
  value = c(
    dummy_or(var = "bld_xpert_pos"),
    dummy_or(var = "MBC1_cultureID"),
    dummy_or(var = "uGXP"),
    dummy_or(var = "sputum_xpert")
  )
) %>%
  pivot_wider(names_from = name, values_from = value) -> diagnostic_or_table



#g_bxct_loess +
#  ggtitle(
#    paste0("iqrOR = ",
#           round(as.numeric(ct_ttp_iqr_or_table[1,2]), 1),
#           "; p = ", round(as.numeric(ct_ttp_iqr_or_table[1,3]), 3))
#  ) +
#  theme(plot.title = element_text(size = 10, face = "plain"))

#col2rgb("#DCE319FF")
#rgb(220, 227, 25,
#    max = 255,
#    alpha = (100 - 70) * 255 / 100,
#    names = NULL)
```

Combine all these plots into one figure showing all bootstrapped loess regressions, also with OR for binary outcome presented in mosaic plot:

```
(g_bxct_2x2 | g_bcttp_2x2 | g_uxct_2x2 | g_sxct_2x2 ) /
  ( g_bxct_loess | g_bcttp_loess | g_uxct_loess | g_sxct_loess )
```



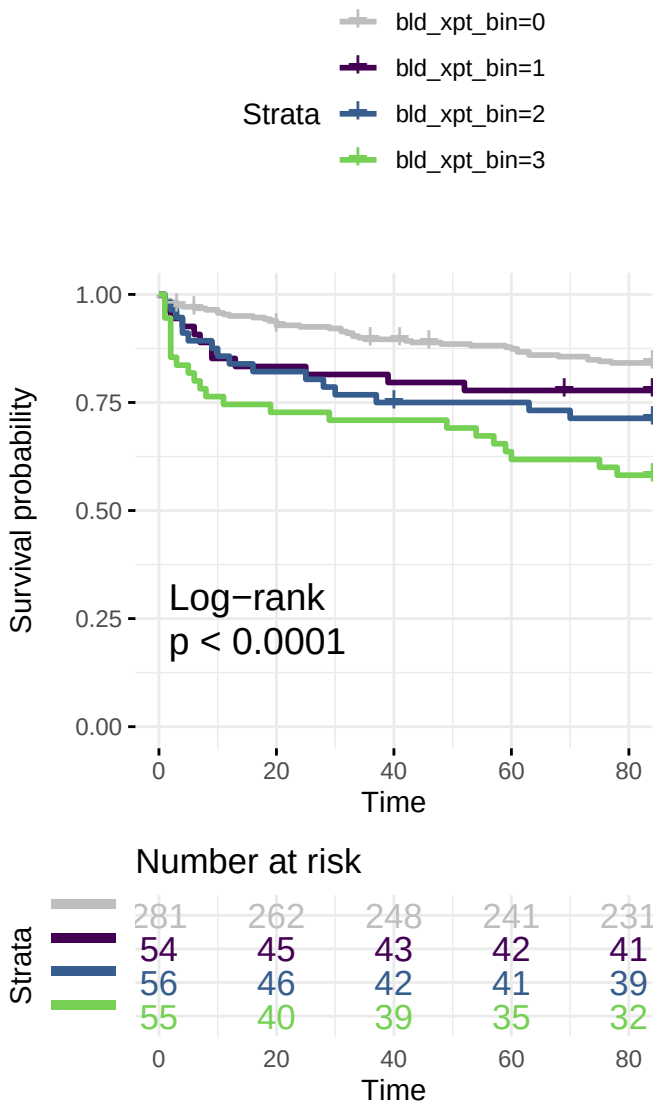
7.2.2 KM and proportional hazards regression using blood Xpert results on ordinal scale as predictor variable

Each diagnostic binned into 4 strata; 0 = negative, 1 to 3 = lowest to highest bacilli load by tertile for patients with positive result (ie highest to lowest CT or TTP). Giving an ordinal scale combining all information from each rapid diagnostic test.

KM plots, raw data (complete case analysis):

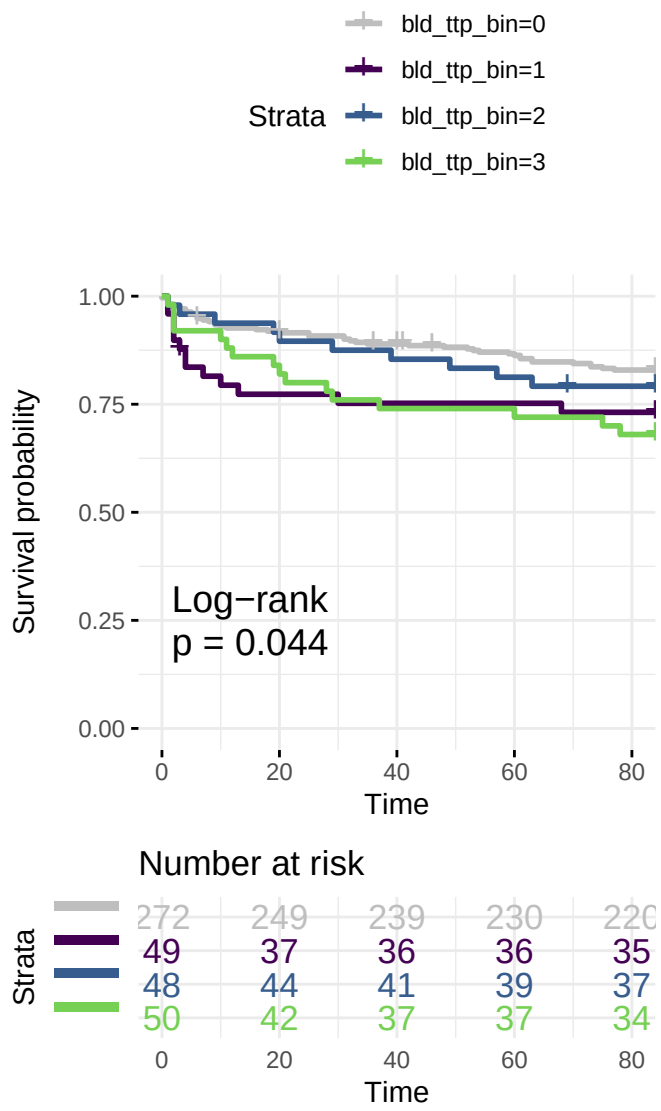
```
# make ordinal scale for each

n = 4

tbdf$bld_xpt_bin <- n - as.numeric(cut2(tbdf$blood_Xpert_CT, g = n-1))
tbdf$bld_xpt_bin[tbdf$blood_Xpert_MTB=="Negative"] <- 0

tbdf$bld_ttp_bin <- n - as.numeric(cut2(tbdf$MBC1_TTP, g = n-1))
tbdf$bld_ttp_bin[tbdf$MBC1_cultureID=="negative"] <- 0

tbdf$urn_xpt_bin <- n - as.numeric(cut2(tbdf$min.ct_urineGXP, g = n-1))
tbdf$urn_xpt_bin[tbdf$uGXP=="NEG"] <- 0

tbdf$spm_xpt_bin <- n - as.numeric(cut2(tbdf$min.ct_sptmGXP, g = n-1))
tbdf$spm_xpt_bin[tbdf$sputum_xpert=="NEG"] <- 0

tbdf$day84death[is.na(tbdf$day84death)] <- 0 # teh LTFU patients
```

```r
# KM plots

y <- Surv(tbdf$time, tbdf$day84death)

ggsurvplot(survfit(y ~ tbdf$bld_xpt_bin),
           data = tbdf,
           risk.table = TRUE,
           palette = c("grey", "#440154FF", "#365D8DFF", "#75D054FF"),
           pval = TRUE, pval.method = TRUE,
           ggtheme = theme_minimal(),
           risk.table.col="strata",
           risk.table.y.text=FALSE) +
  guides(colour = guide_legend(nrow = 5))
```

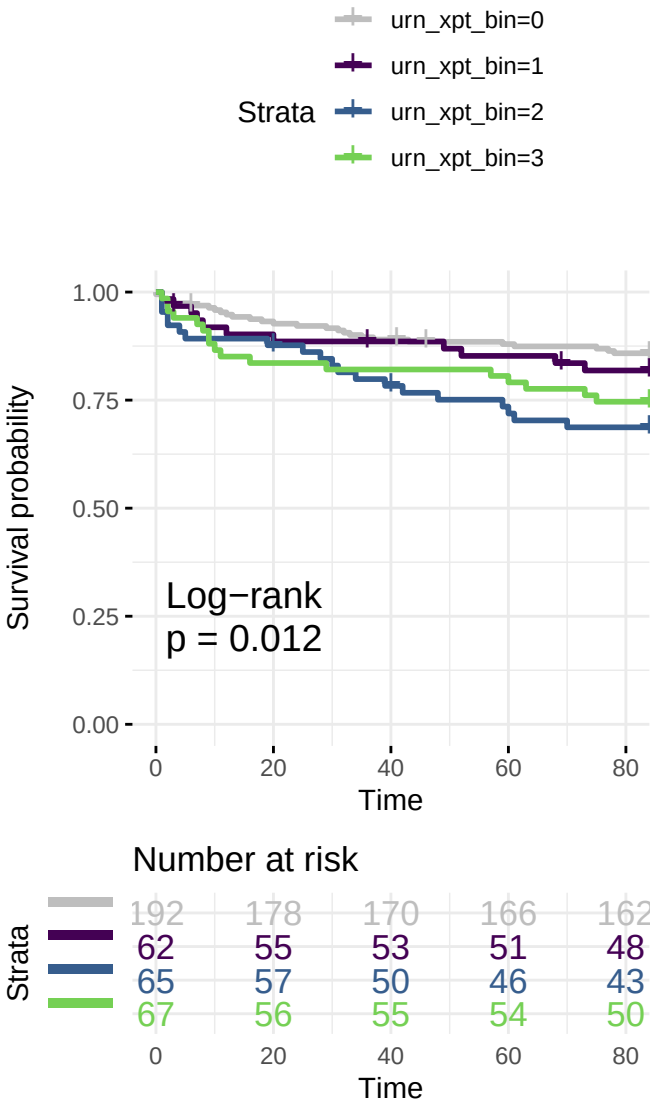



```r
ggsurvplot(survfit(y ~ tbdf$bld_ttp_bin),
           data = tbdf,
```

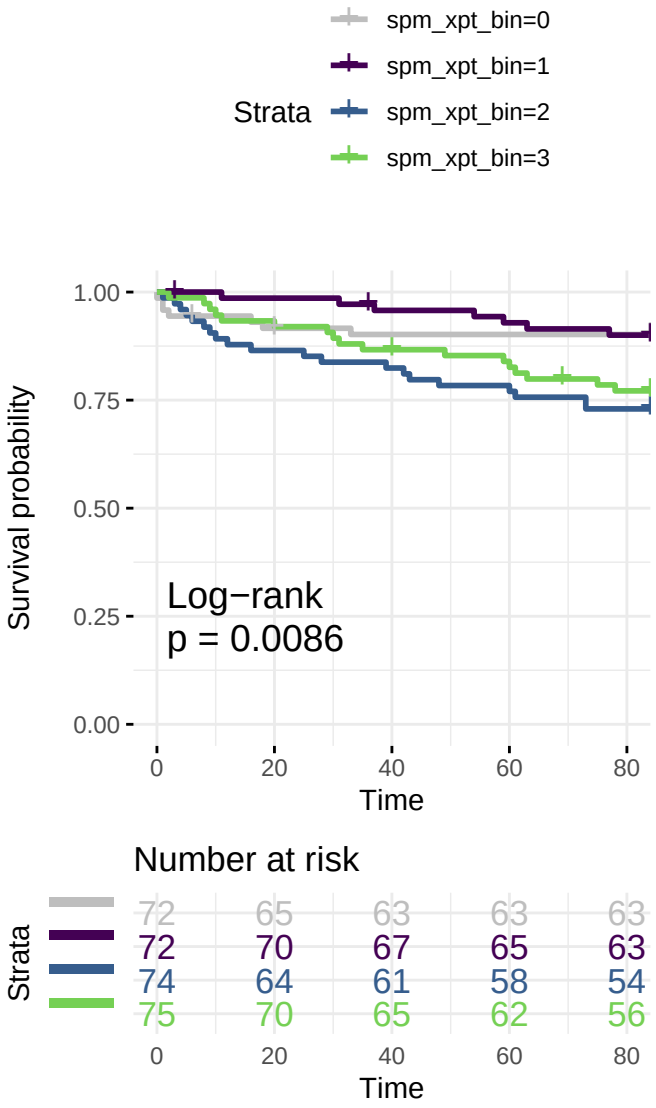
```
          risk.table = TRUE,
          palette = c("grey", "#440154FF", "#365D8DFF", "#75D054FF"),
          pval = TRUE, pval.method = TRUE,
          ggtheme = theme_minimal(),
          risk.table.col="strata",
          risk.table.y.text=FALSE) +
  guides(colour = guide_legend(nrow = 5))
```




```
ggsurvplot(survfit(y ~ tbdf$urn_xpt_bin),
          data = tbdf,
          risk.table = TRUE,
          palette = c("grey", "#440154FF", "#365D8DFF", "#75D054FF"),
          pval = TRUE, pval.method = TRUE,
          ggtheme = theme_minimal(),
          risk.table.col="strata",
          risk.table.y.text=FALSE) +
  guides(colour = guide_legend(nrow = 5))
```




```
ggsurvplot(survfit(y ~ tbdf$spm_xpt_bin),
           data = tbdf,
           risk.table = TRUE,
           palette = c("grey", "#440154FF", "#365D8DFF", "#75D054FF"),
           pval = TRUE, pval.method = TRUE,
           ggtheme = theme_minimal(),
           risk.table.col="strata",
           risk.table.y.text=FALSE) +
  guides(colour = guide_legend(nrow = 5))
```



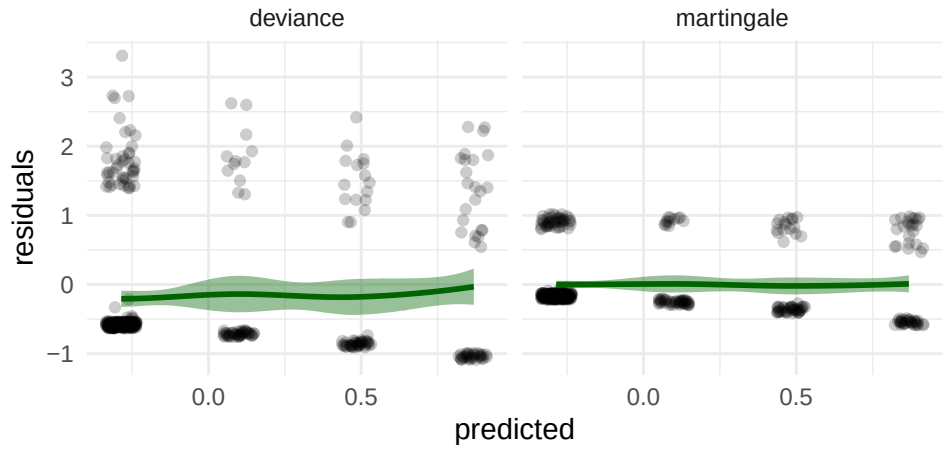

```
#ggsurvplot(survfit(y ~ tbdf$bld_xpt_bin),
#           data = tbdf,
#           risk.table = TRUE,
#           palette = c("grey", "#4B2991", "#952EA0", "#D44292"),
#           pval = FALSE, pval.method = FALSE,
#           ggtheme = theme_pubclean(),
#           risk.table.col="strata",
#           risk.table.y.text=FALSE) +
#  guides(colour = guide_legend(nrow = 4))

# this has a better ar risk table with censored and events n
#ts <- seq(0, 80, by=20)
#fit <- survival::survfit(
#  survival::Surv(time, as.numeric(day84death)) ~ bld_xpt_bin,
#  data = tbdf)
#KMunicate::KMunicate(fit=fit, time_scale = ts)
```

7.2.3 CPH model of blood Xpert ultra on ordinal scale

Cox proportional hazards model fit for blood Xpert-Ultra on ordinal scale.

```
# cph model for blood xpert ordinal scale HR
phbx <- coxph(Surv(time, day84death) ~ bld_xpt_bin, data = tbdf)
```

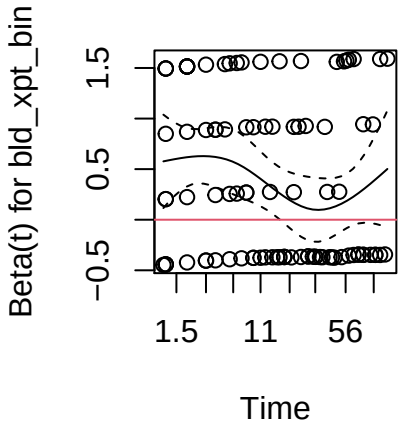
```
# some model diagnostics: residuals
data.frame(
  predicted = predict(phbx),
  martingale = residuals(phbx, type="martingale"),
  deviance = residuals(phbx, type="deviance")
  ) %>%
  pivot_longer(names_to = "type", values_to = "residuals", 2:3) %>%
  ggplot(
    aes(predicted, residuals)
  ) +
  geom_point(
    position = position_jitter(width=0.05, height=0.05), alpha=0.2) +
  geom_smooth(colour="darkgreen", fill="darkgreen") +
  facet_wrap(~type) +
  theme_minimal()
```




```
# some model diagnostics: PH assumption
cox.zph(phbx)
```

```
##             chisq df     p
## bld_xpt_bin  3.02  1 0.082
## GLOBAL       3.02  1 0.082
```

```
plot(cox.zph(phbx)); abline(h=0, col=2)
```




```
# model summary

summary(phbx)
```

```
## Call:
## coxph(formula = Surv(time, day84death) ~ bld_xpt_bin, data = tbdf)
##
##   n= 446, number of events= 95
##    (1 observation deleted due to missingness)
##
##                coef exp(coef) se(coef)     z Pr(>|z|)
## bld_xpt_bin 0.38468   1.46914  0.08244 4.666 3.07e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##             exp(coef) exp(-coef) lower .95 upper .95
## bld_xpt_bin     1.469     0.6807      1.25     1.727
##
## Concordance= 0.618  (se = 0.027 )
## Likelihood ratio test= 19.97  on 1 df,   p=8e-06
## Wald test            = 21.77  on 1 df,   p=3e-06
## Score (logrank) test = 23.21  on 1 df,   p=1e-06
```

7.2.4 Formal testing of blood Xpert ultra versus other variables for prediction of day 84 mortality

Missing observations of urine and sputum diagnostics imputed so that comparisons can be nested. Imputation by CART in mice, with 10 imuted dfs created.

Now a function that fits models using specified predictor variable sets with eth imputed datasets, extracts and returns the average Likelihood Chi squared stats for the model across all imputed versions. This function is then run over a set of predictor sets we want to compare. These nested models are then compared using Chi squared distribution, as follows:

1. All predictors from the set {TB blood culture (ordinal scale); urine Xpert (ordinal scale); urine Xpert (ordinal scale); urine-LAM; qSOFA score} individually and combined in a multivariable model (6

models total) are compared to the same model refitted *plus* blood Xpert-Ultra (ordinal scale) as an additional predictor. This formally tests *if blood Xpert-ultra adds predictive value to each of these base models.*

2. Each of the 6 models with blood Xpert-Ultra added as an additional predictor is compared to a model with blood Xpert-Ultra alone. This formally tests *if each predictor set adds predictive value to blood Xpert-Ultra.*

If the answr to all in 1. is "yes" and all in 2. in "no" (at alpha<0.05) we can conclude that blood Xpert-Ultra is teh more valuable prognostic indicator. (Ref Harrell RMS chapter 9)

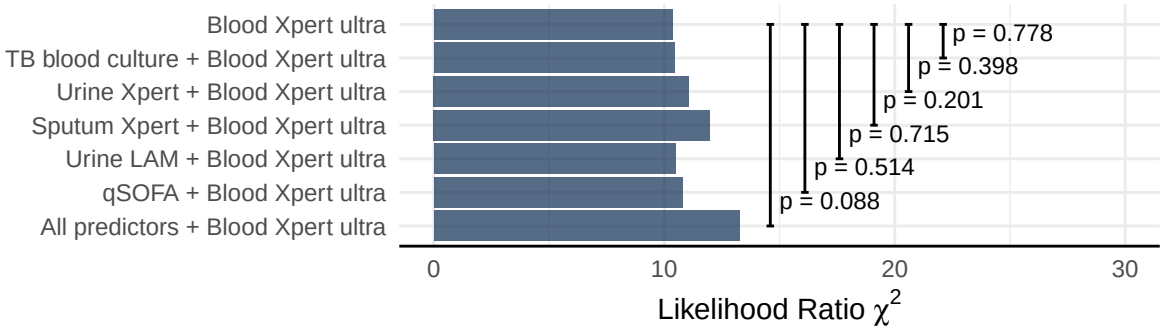
```
# a function to extract LR chi squared from models
# built with each imputed dataset and avaerage them to summary
LRX2_extract <- function(predictor){
  formula = paste0(
    "Surv(time, day84death) ~ ", predictor
    )
  with(imp, coxph(formula = as.formula(formula)))$analyses %>%
    map(3) %>% map(diff) %>% unlist() %>% mean() -> lrx
  return(lrx)
}

# predictor sets set we want to compare
predictor_set <- c(
  "bld_xpt_bin",
  "bld_ttp_bin",
  "bld_xpt_bin + bld_ttp_bin",
  "urn_xpt_bin",
  "bld_xpt_bin + urn_xpt_bin",
  "spm_xpt_bin",
  "bld_xpt_bin + spm_xpt_bin",
  "ALERE_FC",
  "bld_xpt_bin + ALERE_FC",
  "qSOFA",
  "bld_xpt_bin + qSOFA",
  "bld_ttp_bin + urn_xpt_bin + spm_xpt_bin + ALERE_FC + qSOFA",
  "bld_xpt_bin + bld_ttp_bin + urn_xpt_bin + spm_xpt_bin + ALERE_FC + qSOFA"
)

# map this function over each predictor set we want
LRX2 <- predictor_set %>%
  map_dbl(LRX2_extract)

# nice format model labels
model_labels <- c(
  "Blood Xpert ultra",
  "TB blood culture",
  "TB blood culture + Blood Xpert ultra",
  "Urine Xpert",
  "Urine Xpert + Blood Xpert ultra",
  "Sputum Xpert",
  "Sputum Xpert + Blood Xpert ultra",
  "Urine LAM",
  "Urine LAM + Blood Xpert ultra",
  "qSOFA",
  "qSOFA + Blood Xpert ultra",
```

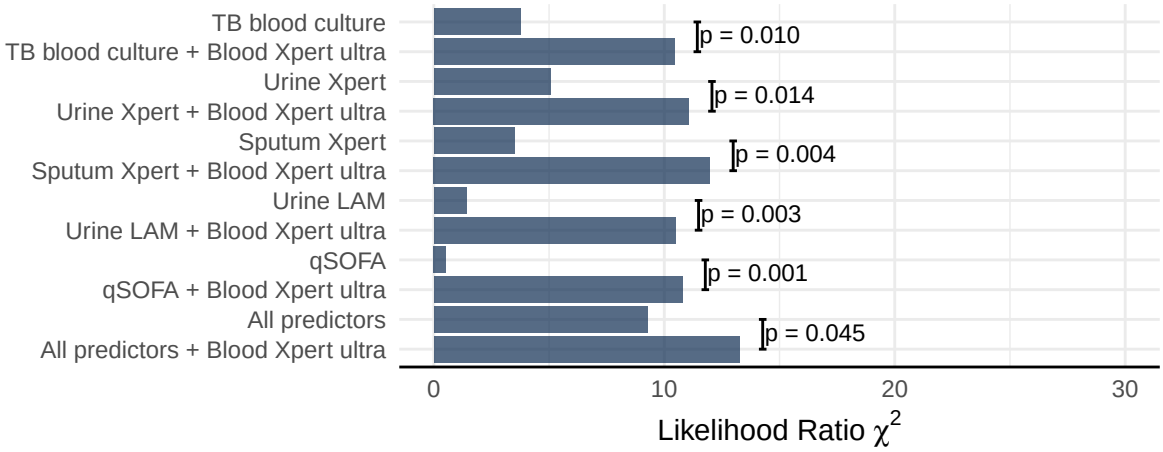
```
  "All predictors",
  "All predictors + Blood Xpert ultra"
)

# LRTs all models versus blood xpert ultra as single predictor
vrs_phbx = pchisq(q=LRX2-LRX2[1], df = 1, lower.tail = FALSE)
vrs_phbx[vrs_phbx==1] <- NA

# LRTs other models v nested version
vrs_singe_predictor = c(
  NA,
  NA, pchisq(q=LRX2[3]-LRX2[2], df = 1, lower.tail = FALSE), NA,
  pchisq(q=LRX2[5]-LRX2[4], df = 1, lower.tail = FALSE), NA,
  pchisq(q=LRX2[7]-LRX2[6], df = 1, lower.tail = FALSE), NA,
  pchisq(q=LRX2[9]-LRX2[8], df = 1, lower.tail = FALSE), NA,
  pchisq(q=LRX2[11]-LRX2[10], df = 1, lower.tail = FALSE), NA,
  pchisq(q=LRX2[13]-LRX2[12], df = 1, lower.tail = FALSE)
)

# combine in df
data.frame(model_labels,
           LRX2,
           vrs_phbx, vrs_singe_predictor) %>%
  mutate(
    model_labels = factor(
      model_labels,
      levels = rev(model_labels)
    ),
    vrs_phbx = ifelse(!is.na(vrs_phbx),
                      paste0("p = ", round(vrs_phbx, 3)),
                      NA),
    vrs_singe_predictor = ifelse(!is.na(vrs_singe_predictor),
                                 paste0("p = ",
                                        format(
                                          round(vrs_singe_predictor, 3),
                                          nsmall = 3
                                 )),
                                 NA)
  ) -> nested_lrt_df

# make plots as per Harrell suggested approach
nested_lrt_df[c(1,3,5,7,9,11,13),] %>%
  mutate(y1 = seq(24,15, length.out = 7)) %>%
    ggplot(aes(model_labels, LRX2)) +
    geom_bar(stat="identity", alpha=0.7, fill="#0c2a50ff") +
    coord_flip() +
    theme_minimal() +
    ylim(0, 30) +
    geom_text(
        aes(label = vrs_phbx,
            y=y1),
        vjust=-1, hjust=0, size=3) +
  annotate("segment", x=7, xend=6, y=22.1, yend=22.1,
```

```
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=7, xend=5, y=20.6, yend=20.6,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=7, xend=4, y=19.1, yend=19.1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=7, xend=3, y=17.6, yend=17.6,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=7, xend=2, y=16.1, yend=16.1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=7, xend=1, y=14.6, yend=14.6,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  xlab("") + ylab(expression(paste("Likelihood Ratio ", chi^2))) +
  theme(axis.line.x = element_line(colour = "black")) -> g1



nested_lrt_df[2:13,] %>%
    ggplot(aes(model_labels, LRX2)) +
    geom_bar(stat="identity", alpha=0.7, fill="#0c2a50ff") +
    coord_flip() +
    theme_minimal() +
    ylim(0, 30) +
    geom_text(
        aes(label = vrs_singe_predictor,
            ),
        vjust=-0.5, hjust=-0.25, size=3) +
  annotate("segment", x=12, xend=11, y=LRX2[3]+1, yend=LRX2[3]+1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=10, xend=9, y=LRX2[5]+1, yend=LRX2[5]+1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=8, xend=7, y=LRX2[7]+1, yend=LRX2[7]+1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=6, xend=5, y=LRX2[9]+1, yend=LRX2[9]+1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=4, xend=3, y=LRX2[11]+1, yend=LRX2[11]+1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  annotate("segment", x=2, xend=1, y=LRX2[13]+1, yend=LRX2[13]+1,
          arrow = arrow(ends = "both", angle = 90, length = unit(.05,"cm"))) +
  xlab("") + ylab(expression(paste("Likelihood Ratio ", chi^2))) +
  theme(axis.line.x = element_line(colour = "black")) -> g2
```

Blood Xpert-Ultra adds value to every other predictor (or even all of them in combination) - lower panel. No set of predictors combined with blood Xpert-Ultra does better than blood Xpert-ultra alone - top panel.

```
g1/g2 + plot_layout(heights = c(1,1.5))
```





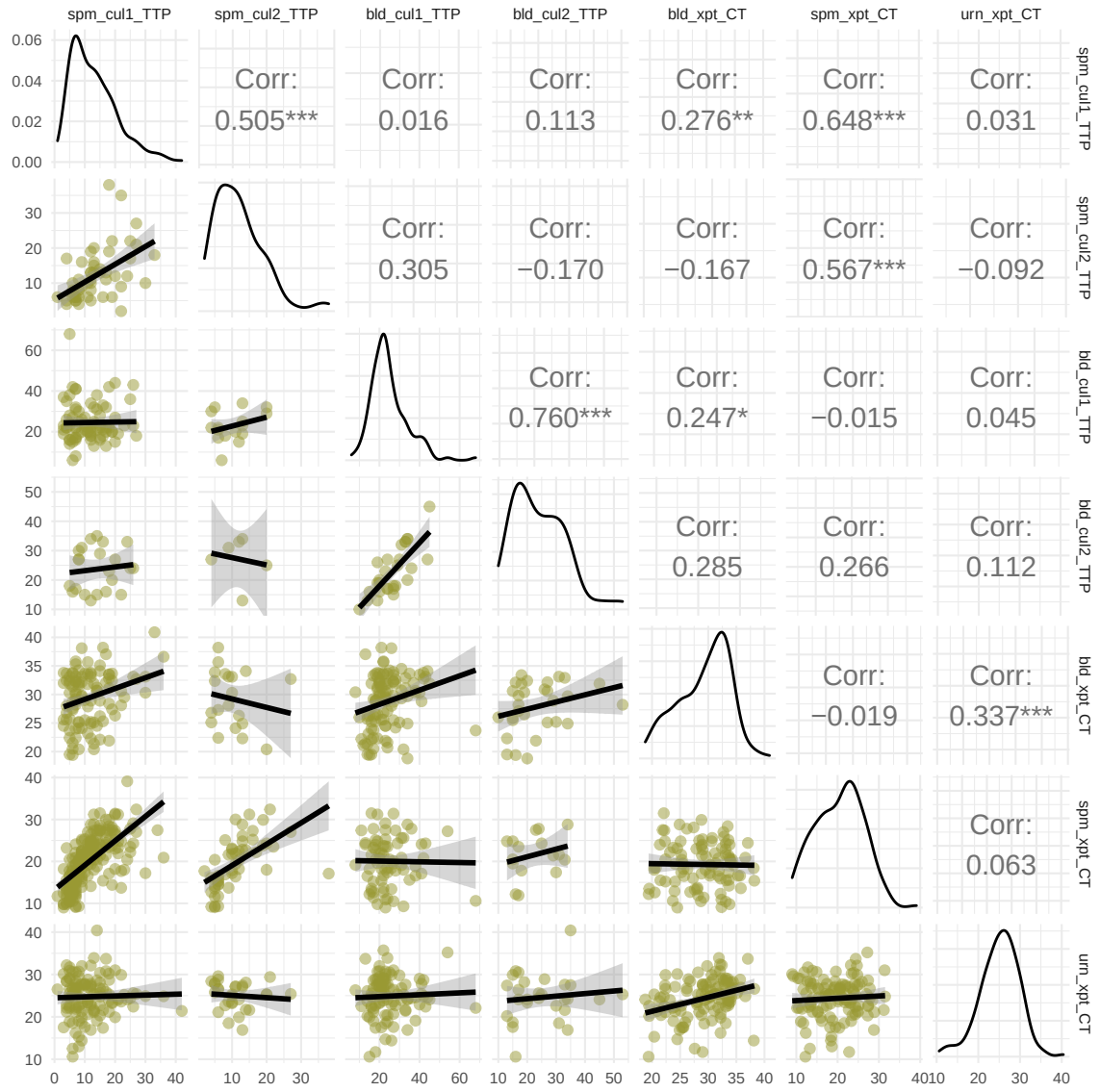
8 Correlation between (semi) quantitative measures of bacilli number

8.1 Pairwise comparisons, all samples we have readouts for

8.1.1 pairwise scatter plots

With Pearson's correlation coefficients

```
tbdf %>%
  select(spm_cul1_TTP=sputumCulture1_TTP,
         spm_cul2_TTP=sputumCulture2_TTP,
         bld_cul1_TTP=MBC1_TTP,
         bld_cul2_TTP=MBC2_TTP,
         bld_xpt_CT=blood_Xpert_CT,
         spm_xpt_CT=min.ct_sptmGXP,
         urn_xpt_CT=min.ct_urineGXP) %>%
  ggpairs(lower=list(
    continuous = wrap("smooth",
                      alpha=0.5,
                      colour="#999933"))) +
  theme_minimal(base_size = 8)
```

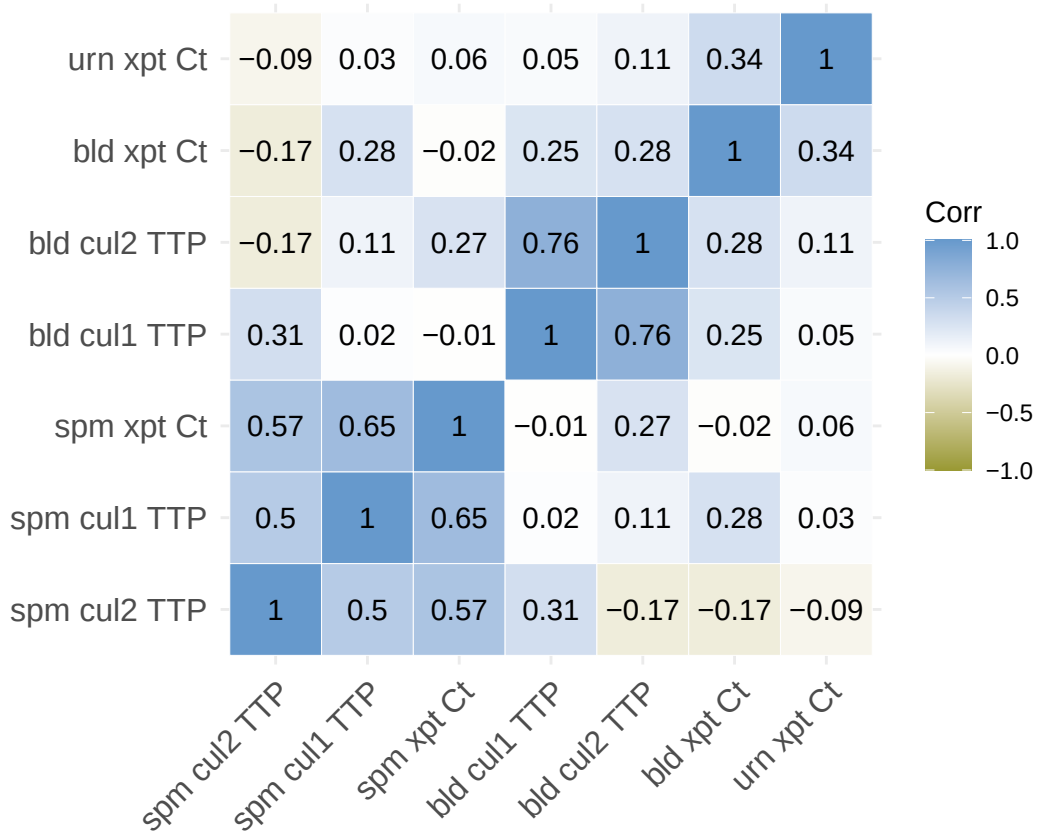


8.1.2 Same correlations but in a correlation matrix

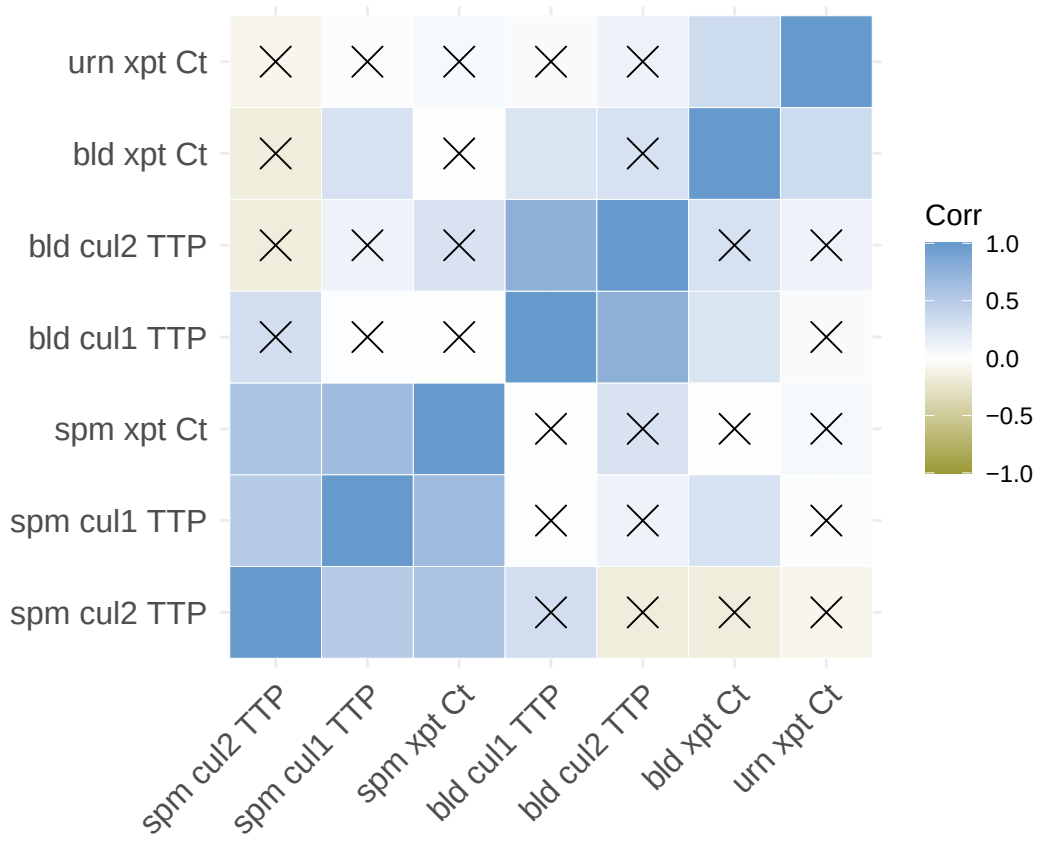
- A : with Pearson's r coefficients shown by numbers in the squares
- B : same plot, but with "non-significant" (p>0.05) correlations indicated by an X in the squares

```
m <- cor(tbdf %>%
  select(`spm cul1 TTP`=sputumCulture1_TTP,
         `spm cul2 TTP`=sputumCulture2_TTP,
         `bld cul1 TTP`=MBC1_TTP,
         `bld cul2 TTP`=MBC2_TTP,
         `bld xpt Ct`=blood_Xpert_CT,
         `spm xpt Ct`=min.ct_sptmGXP,
         `urn xpt Ct`=min.ct_urineGXP),
  use = "pairwise")
p.mat <- cor_pmat(tbdf %>%
  select(`spm cul1 TTP`=sputumCulture1_TTP,
```

```
          `spm cul2 TTP`=sputumCulture2_TTP,
          `bld cul1 TTP`=MBC1_TTP,
          `bld cul2 TTP`=MBC2_TTP,
          `bld xpt Ct`=blood_Xpert_CT,
          `spm xpt Ct`=min.ct_sptmGXP,
          `urn xpt Ct`=min.ct_urineGXP),
  use = "pairwise")

ggcorrplot(m,
  hc.order = TRUE, outline.color = "white", lab = TRUE,
  colors = c("#999933", "white", "#6699CC"))
```

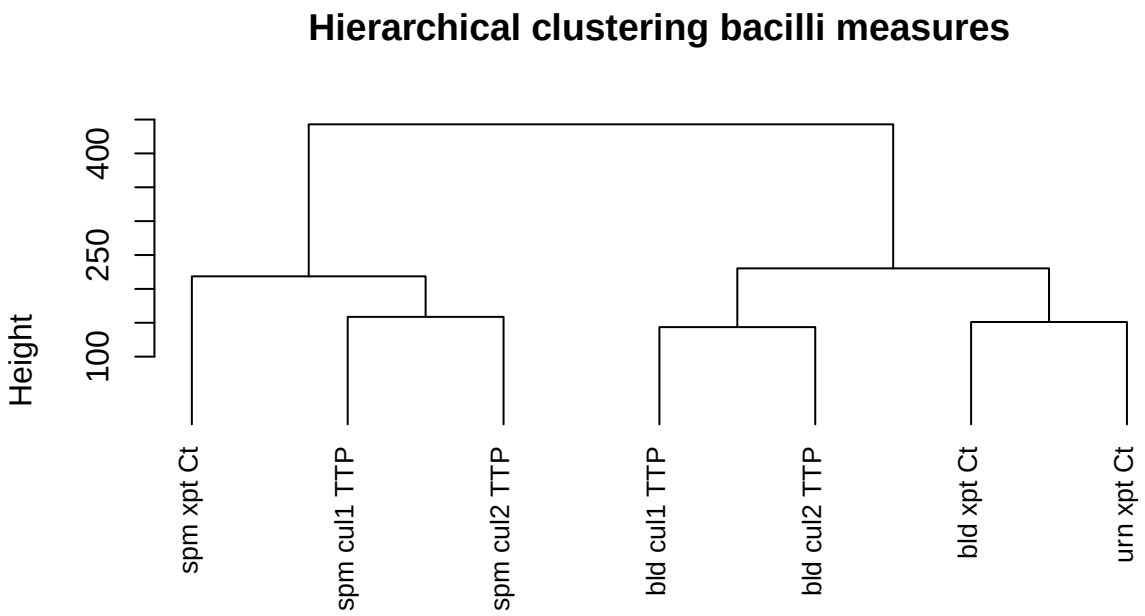



```
ggcorrplot(m,
  hc.order = TRUE, outline.color = "white", p.mat=p.mat,
  colors = c("#999933", "white", "#6699CC"))
```



8.1.3 Clustering the same variables

```
m <- tbdf %>%
  select(`spm cul1 TTP`=sputumCulture1_TTP,
         `spm cul2 TTP`=sputumCulture2_TTP,
         `bld cul1 TTP`=MBC1_TTP,
         `bld cul2 TTP`=MBC2_TTP,
         `bld xpt Ct`=blood_Xpert_CT,
         `spm xpt Ct`=min.ct_sptmGXP,
         `urn xpt Ct`=min.ct_urineGXP)

d <- dist(t(as.matrix(m)), method = "euclidean")
fit.average <- hclust(d, method="complete")
plot(fit.average, hang=-1, cex=.8,
     main="Hierarchical clustering bacilli measures", xlab="", sub="")
```



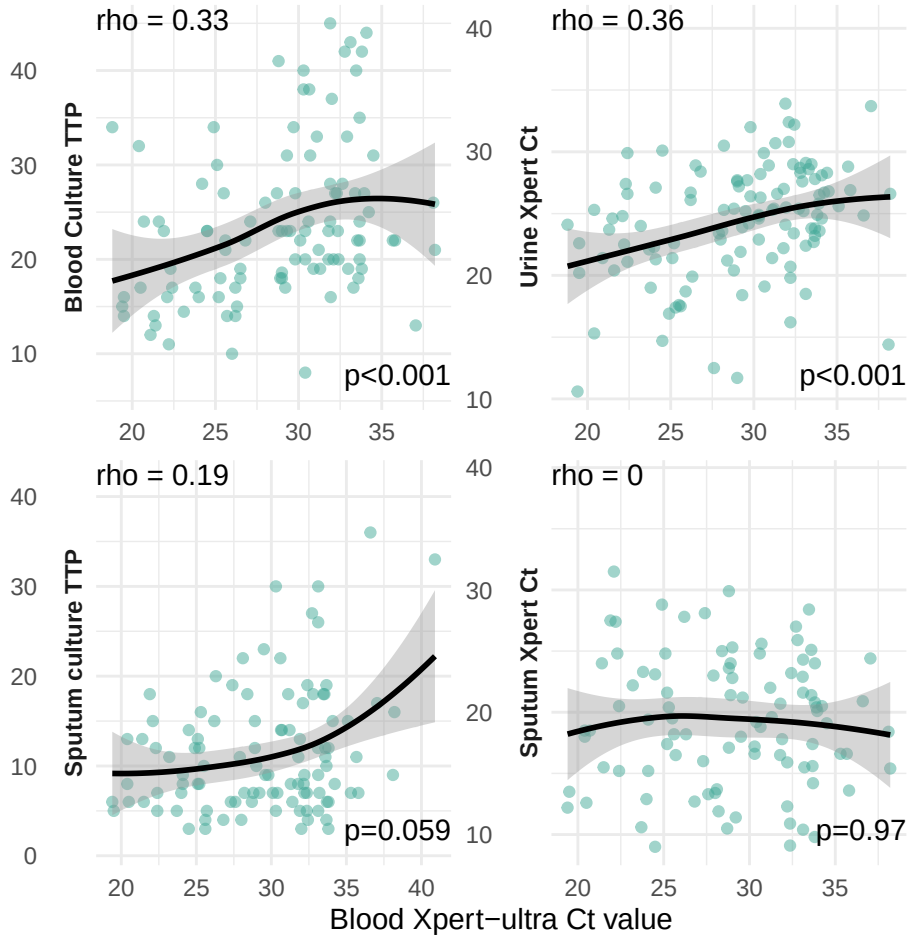
8.2 Plot focusing on blood Xpert Ct values

Spearman's Rho with p value shown.

```
df.cor <- function(x, y) {
  round(cor(x[y<50], y[y<50], use = "complete.obs", method = "spear"), 2)
}

df.p <- function(x, y){
  formatC(cor.test(x[y<50], y[y<50],
                   method = "spear",
                   use="complete.cases")$p.value,
          format="e", digits=1)
}


tbdf %>%
  select(
    bld_xpt_CT=blood_Xpert_CT,
    spm_cul1_TTP=sputumCulture1_TTP,
#    spm_cul2_TTP=sputumCulture2_TTP,
    bld_cul1_TTP=MBC1_TTP,
#    bld_cul2_TTP=MBC2_TTP,
    spm_xpt_CT=min.ct_sptmGXP,
    urn_xpt_CT=min.ct_urineGXP) -> foo
```

```
foo %>% map(df.cor, y=foo$bld_xpt_CT) -> rdf
names(rdf) -> var
data.frame(var,
           rho = as.numeric(rdf)) %>%
  filter(var!="bld_xpt_CT")%>%
  mutate(var = factor(var,
                      levels =
                        c("bld_cul1_TTP",
                          "urn_xpt_CT",
                          "spm_cul1_TTP",
                          "spm_xpt_CT"),
                      labels =
                        c("Blood Culture TTP",
                          "Urine Xpert Ct",
                          "Sputum culture TTP",
                          "Sputum Xpert Ct"))) -> rdf

foo %>% map(df.p, y=foo$bld_xpt_CT) -> pdf
names(pdf) -> var
data.frame(var,
           p = as.numeric(pdf)) %>%
  filter(var!="bld_xpt_CT") %>%
  mutate(var = factor(var,
                      levels =
                        c("bld_cul1_TTP",
                          "urn_xpt_CT",
                          "spm_cul1_TTP",
                          "spm_xpt_CT"),
                      labels =
                        c("Blood Culture TTP",
                          "Urine Xpert Ct",
                          "Sputum culture TTP",
                          "Sputum Xpert Ct"))) %>%
  mutate(
    p = ifelse(
      p<0.001,
      "p<0.001",
      paste0("p=", p)
    )
  )-> pdf

foo %>%
  gather(key=var, value = value, 2:5) %>%
  filter(value<50) %>%
  mutate(var = factor(var,
                      levels =
                        c("bld_cul1_TTP",
                          "urn_xpt_CT",
                          "spm_cul1_TTP",
                          "spm_xpt_CT"),
                      labels =
                        c("Blood Culture TTP",
                          "Urine Xpert Ct",
```

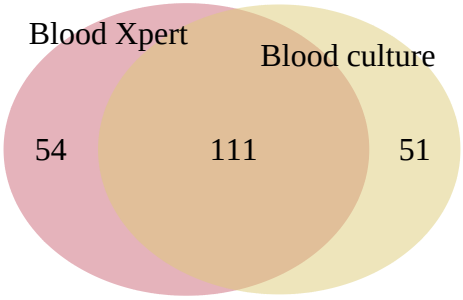
```
                          "Sputum culture TTP",
                          "Sputum Xpert Ct"))) %>%
  ggplot(aes(bld_xpt_CT, value)) +
  geom_point(colour="#44AA99", alpha=0.5) +
  geom_smooth(span=2, colour="black") +
  geom_text(data=rdf,
            aes(label = paste0("rho = ", rho)),
            x=-Inf, y=Inf, hjust=0, vjust=1.2) +
  geom_text(data=pdf,
            aes(label = p),
            x=Inf, y=-Inf, hjust=1, vjust=-1.2) +
  facet_wrap(~var, scales = "free", strip.position = "left") +
  theme_minimal() +
  theme(strip.text = element_text(face = "bold")) +
  ylab("") + xlab("Blood Xpert-ultra Ct value")
```



9 Venn/Euler and Venn type figures

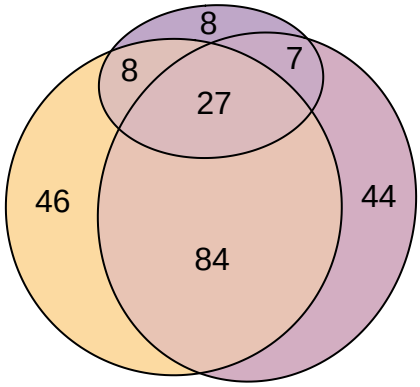
```
area1 = sum(tbdf$bld_xpert_diagnosed)
area2 = sum(tbdf$MBC1_cultureID=="MTB" &
```

```
              !is.na(tbdf$MBC1_cultureID))
n12 = sum(tbdf$bld_xpert_diagnosed &
            tbdf$MBC1_cultureID=="MTB" &
            !is.na(tbdf$MBC1_cultureID))

draw.pairwise.venn(area1 = area1, area2 = area2, cross.area = n12,
                   category = c("Blood Xpert", "Blood culture"),
                   lty="blank",
                   fill = c("#CC6677", "#DDCC77"),
                   cat.just = list(c(0,0), c(1,1)))
```




```
## (polygon[GRID.polygon.4775], polygon[GRID.polygon.4776], polygon[GRID.polygon.4777], polygon[GRID.pol
```

```
Blood_Xpt <- tbdf$bld_xpert_diagnosed
Blood_Cul_1 <- tbdf$MBC1_cultureID=="MTB" & !is.na(tbdf$MBC1_cultureID)
Blood_Cul_2 <- tbdf$MBC2_cultureID=="MTB" & !is.na(tbdf$MBC2_cultureID)
Blood_Cul_1_f <- tbdf$MBC1_cultureID=="MTB"
Blood_Cul_2_f <- tbdf$MBC2_cultureID=="MTB"

euldf <- data.frame(Blood_Xpt, Blood_Cul_1, Blood_Cul_2, Blood_Cul_1_f, Blood_Cul_2_f)


plot( euler(euldf[,c(1:3)], shape="ellipse"),
      quantities = TRUE,
      fills = list(fill = c("#f9b641ff", "#a65c85ff", "#7e4e90ff"), alpha=0.5),
      labels = list(alpha=0))
```




```
c1 <- cohen.kappa(table(Blood_Xpt, Blood_Cul_1_f))
c2 <- cohen.kappa(table(Blood_Xpt, Blood_Cul_2_f))
```

```
c3 <- cohen.kappa(table(Blood_Cul_1_f, Blood_Cul_2_f))

ck <- data.frame(
  contrast = c("Xpt v Cul1", "Xpt v Cul2", "Cul1 v Cul2"),
  n = c(c1$n.obs, c2$n.obs, c3$n.obs),
  kappa = round(c(c1$kappa, c2$kappa, c3$kappa), 2),
  CI = c(
    paste0(round(c1$confid[1,1],2), "-", round(c1$confid[1,3],2)),
    paste0(round(c2$confid[1,1],2), "-", round(c2$confid[1,3],2)),
    paste0(round(c3$confid[1,1],2), "-", round(c3$confid[1,3],2))
  )
)

kable(ck,"latex", booktabs=T)
```

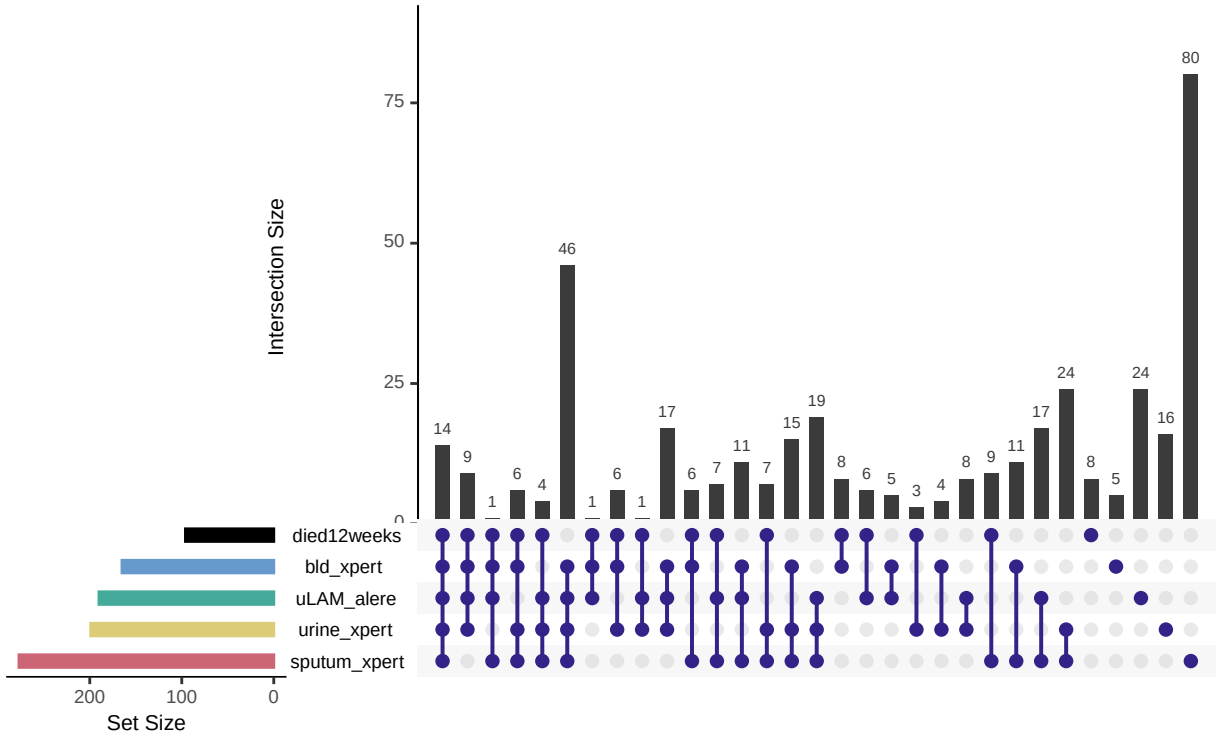
contrast	n	kappa	CI
Xpt v Cul1	438	0.49	0.41-0.58
Xpt v Cul2	113	0.38	0.21-0.55
Cul1 v Cul2	113	0.46	0.29-0.62

9.1 "UpSet" plot

As described by Lex and Gehlenborg in http://www.nature.com/nmeth/journal/v11/n8/abs/nmeth.3033.html

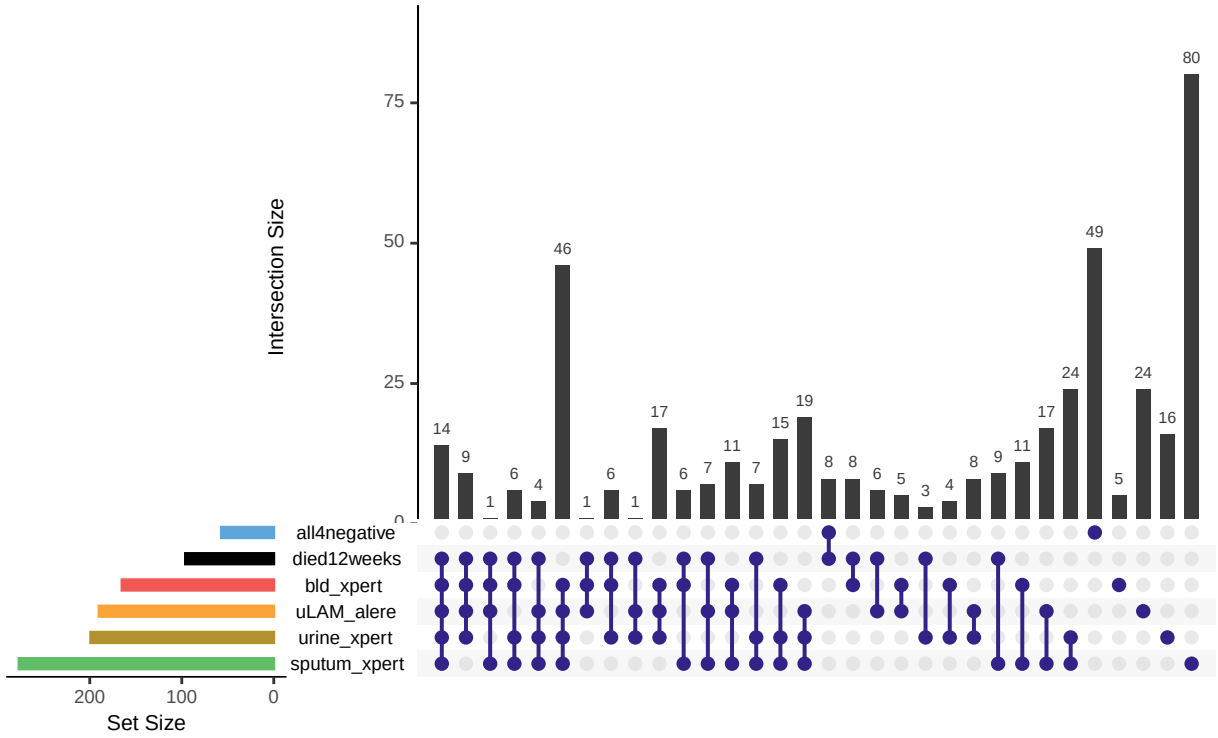
Five variables are shown: 4 rapid diagnostics and 12 week mortality. The horizontal coloured bars show the number positive for each of these 5 variables. The vertical bars show the size of the intersections between these variables indicated by the dots below the bar, e.g. 14 patients are postive for all 5 variables (the first bar) and 79 patients were positive by sputum xpert only and didn't die (the last bar).

```
bld_xpert <- as.numeric(tbdf$bld_xpert_diagnosed)
urine_xpert <- as.numeric(tbdf$uGXP=="MTB" & !is.na(tbdf$uGXP))
sputum_xpert <- as.numeric(tbdf$sputum_xpert=="MTB" & !is.na(tbdf$sputum_xpert))
uLAM_alere <- as.numeric(tbdf$ALERE_FC=="MTB" & !is.na(tbdf$ALERE_FC))
died12weeks <- as.numeric(tbdf$survival.12weeks=="Died" & !is.na(tbdf$survival.12weeks))


updf <- data.frame(bld_xpert, urine_xpert, sputum_xpert, uLAM_alere, died12weeks)

UpSetR::upset(updf,
              sets = c("bld_xpert", "urine_xpert",
                       "sputum_xpert", "uLAM_alere",
                       "died12weeks"),
              order.by = "degree", matrix.color = "#332288",
              sets.bar.color = c("#CC6677", "#DDCC77",
                                 "#44AA99", "#6699CC",
                                 "black"))
```

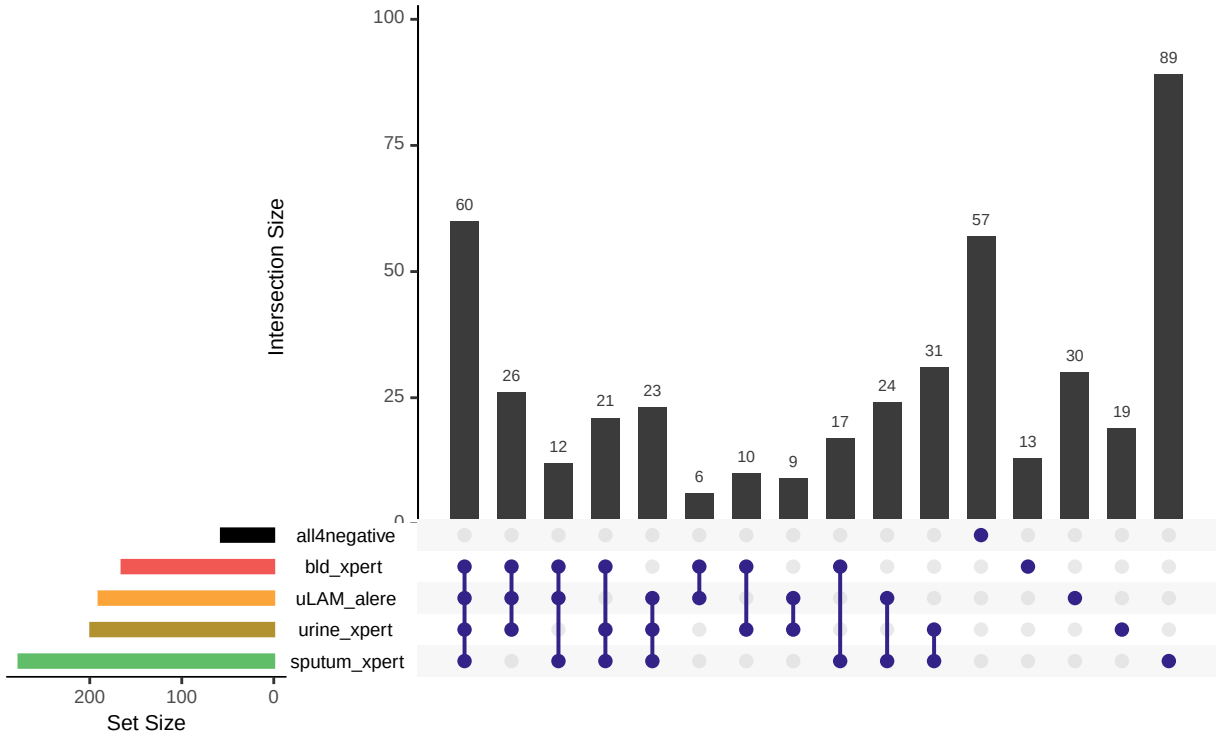


Unlike Venn this is scalable - we could add blood culture for example. Or can add set "no rapid diagnostic test positive" - cases missed by the 4 rapid diagnostics (but diagnosed TB by another test eg culture):

```
all4negative <- as.numeric(bld_xpert==0 & urine_xpert==0 & sputum_xpert==0 & uLAM_alere==0)

updf <- data.frame(bld_xpert, urine_xpert, sputum_xpert, uLAM_alere, died12weeks, all4negative)

UpSetR::upset(updf,
              sets = c("bld_xpert", "urine_xpert",
                       "sputum_xpert", "uLAM_alere",
                       "died12weeks", "all4negative"),
              order.by = "degree", matrix.color = "#332288",
              sets.bar.color = c("#60BD68", "#B2912F",
                                 "#FAA43A", "#F15854",
                                 "black", "#5DA5DA"))
```




```
updf <- data.frame(bld_xpert, urine_xpert, sputum_xpert, uLAM_alere, all4negative)

UpSetR::upset(updf,
              sets = c("bld_xpert", "urine_xpert",
                       "sputum_xpert", "uLAM_alere",
                       "all4negative"),
              order.by = "degree", matrix.color = "#332288",
              sets.bar.color = c("#60BD68", "#B2912F",
                                 "#FAA43A", "#F15854",
                                 "black"))
```



58 patients were missed by the 4 rapid diagnostics, of these 8 died before 12 weeks.

10 Clinical phenotype correlation with blood Xpert Ct values

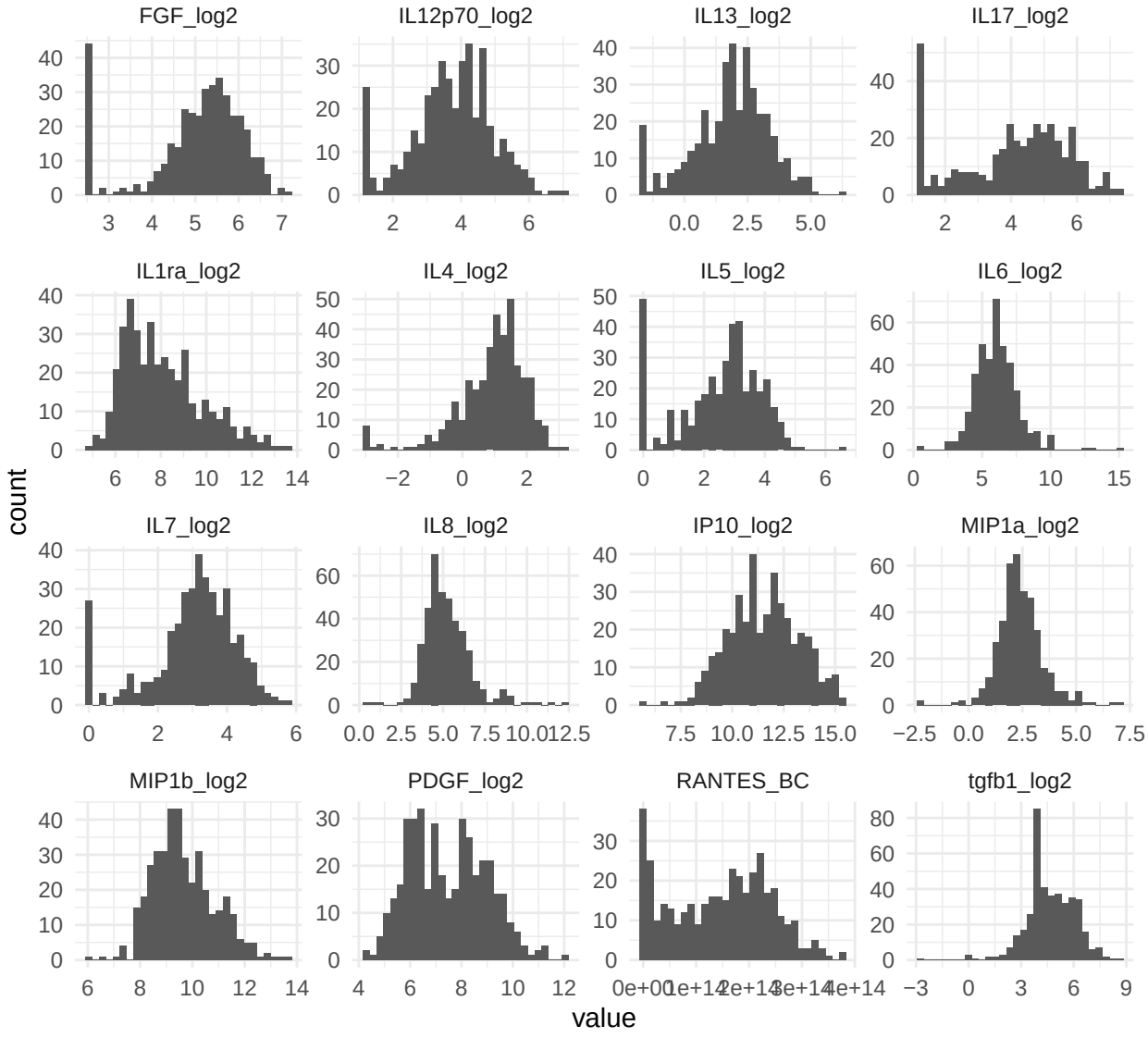
Aim here is to assess for "dose-response" relationship between blood bacilli burden as measured by blood Xpert positivity *and* Ct value, and markers of clinical and immunological phenotype, in particular variables we know to be associated with mortality.

10.1 Univariate and bivariate distributions for individual markers

The 16 soluble immune mediators Charlotte Schutz identified as being most strongly associated with mortality are considered. They are transformed to be approximately normally distributed (in most cases with log transformation). q-values are given where p values are "corrected" for multiple comparison by Benjamini-Hochberg procedure for limiting false discovery rate.

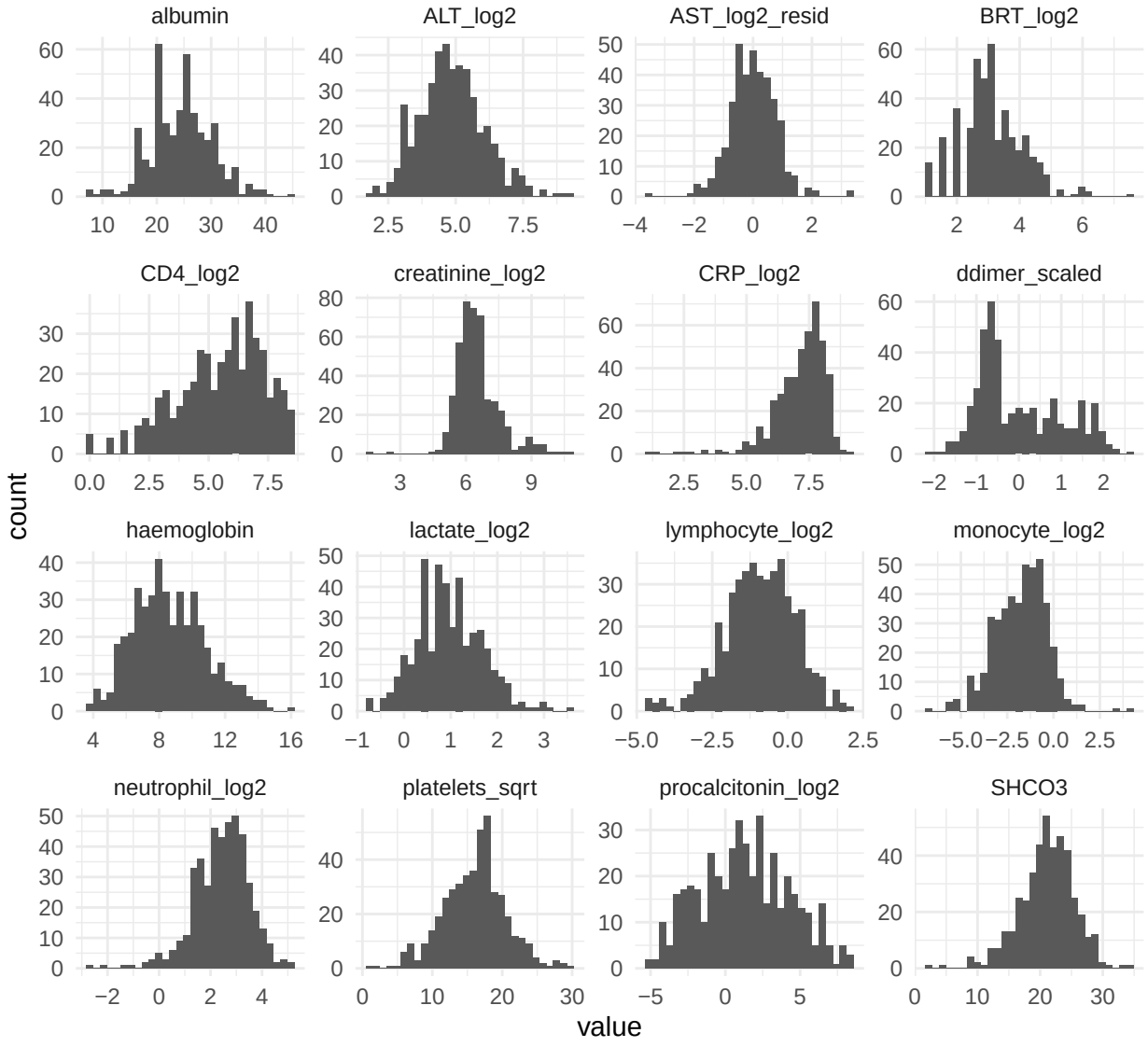
```
tbdf %>%
  dplyr::transmute(blood_Xpert_CT,
       IL8_log2 = log2(Hu.IL_8), #innate and chemotaxis
       MIP1a_log2 = log2(Hu.MIP_1a),
       MIP1b_log2 = log2(Hu.MIP_1b_FI),
       IP10_log2 = log2(Hu.IP_10),

       IL6_log2 = log2(Hu.IL_6), # pro & anti inflam
       IL1ra_log2 = log2(Hu.IL_1ra_FI),

       IL17_log2 = log2(Hu.IL_17), # t cell
       IL4_log2 = log2(Hu.IL_4),
```

```
       RANTES_BC = Hu.RANTES_FI^3.4,
       IL7_log2 = log2(Hu.IL_7),
       IL12p70_log2 = log2(Hu.IL_12.p70),
       IL5_log2 = log2(Hu.IL_5),
       IL13_log2 = log2(Hu.IL_13),

       FGF_log2 = log2(Hu.FGF.basic), # growth factors
       PDGF_log2 = log2(Hu.PDGF_bb_FI),
       tgfb1_log2 = log2(tgfb1.pg.ml)
       ) -> immune_markers

# histograms
immune_markers %>%
  gather(key = var, value = value, 2:17) %>%
  ggplot(aes(value)) +
  geom_histogram() +
  facet_wrap(~var, scales = "free") +
  theme_minimal()
```




```
tbdf %>%
  mutate(
    ddimer.cpt.sqrt = sqrt(ddimer.cpt),
    ddimer.amst.sqrt = sqrt(ddimer.amst)) -> tbdf

u_cpt = mean(tbdf$ddimer.cpt.sqrt, na.rm=T)
sd_cpt = sd(tbdf$ddimer.cpt.sqrt, na.rm=T)
u_amst = mean(tbdf$ddimer.amst.sqrt, na.rm=T)
sd_amst = sd(tbdf$ddimer.amst.sqrt, na.rm=T)
ddmcpt <- (tbdf$ddimer.cpt.sqrt - u_cpt)/sd_cpt
ddmamst <- (tbdf$ddimer.amst.sqrt - u_amst)/sd_amst
tbdf$ddimer <- ifelse(!is.na(ddmcpt), ddmcpt, ddmamst)

tbdf %>%
  dplyr::transmute(blood_Xpert_CT,
                   albumin = Albumin,
                   ALT_log2 = log2(ALT),
```

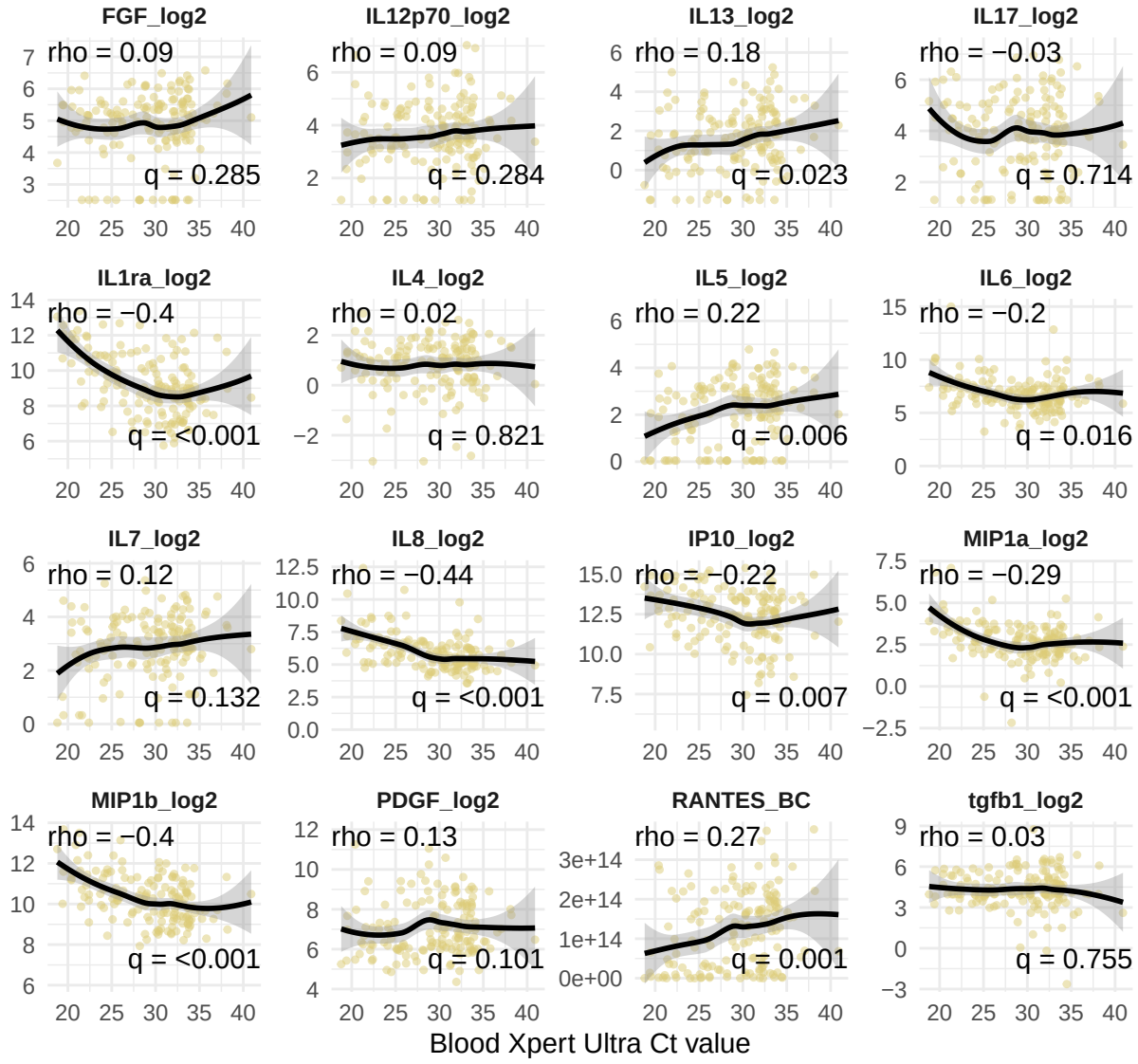
```
                    AST_log2 = log2(AST),
                    BRT_log2 = log2(BRT),
                    CD4_log2 = log2(CD4+1),
                    creatinine_log2 = log2(creatinine),
                    CRP_log2 = log2(CRP),
                    ddimer_scaled = ddimer,
                    haemoglobin = Haemoglobin,
                    lactate_log2 = log2(lactate),
                    lymphocyte_log2 = log2(AbsLymphocyte),
                    monocyte_log2 = log2(AbsMonocyte),
                    neutrophil_log2 = log2(AbsNeutrophil),
                    platelets_sqrt = sqrt(Platelets),
                    procalcitonin_log2 = log2(ProCalcitonin),
                    SHCO3 = SHCO3) -> clin_markers

ast_m <- lm(AST_log2 ~ ALT_log2, data=clin_markers)

clin_markers %>%
  add_residuals(ast_m) %>%
  mutate(AST_log2_resid = resid) %>%
  select(-AST_log2, -resid) -> clin_markers


# histograms
clin_markers %>%
  gather(key = var, value = value, 2:17) %>%
  ggplot(aes(value)) +
  geom_histogram() +
  facet_wrap(~var, scales = "free") +
  theme_minimal()
```



Correlation with blood Xpert Ct values for each of the 16 immune markers:

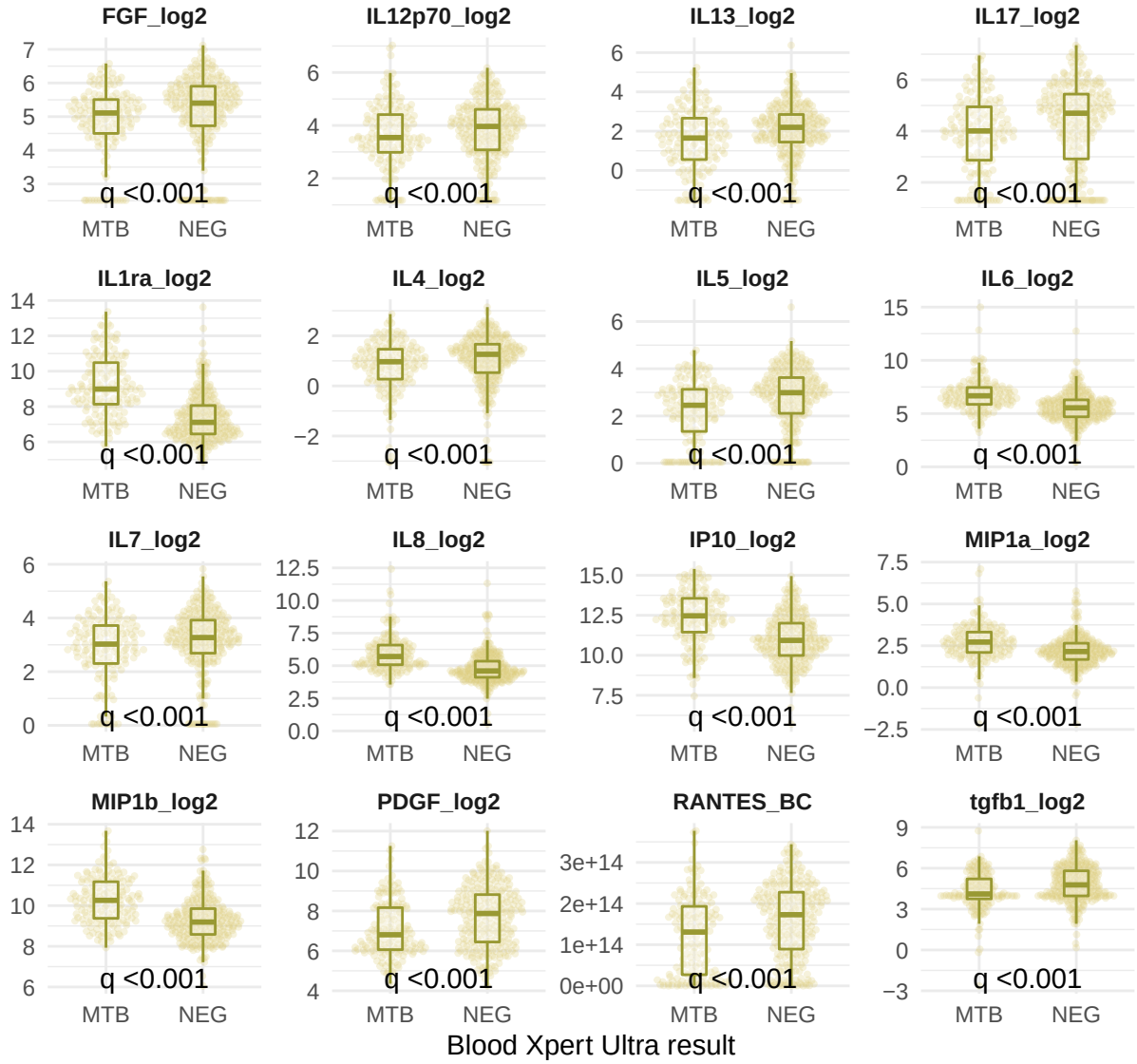
```
immune_markers %>%
  filter(blood_Xpert_CT<50) %>%
  pivot_longer(-blood_Xpert_CT) %>%
  group_by(name) %>%
  do(tidy(cor.test(.$blood_Xpert_CT, .$value,
                   use="complete.cases", method = "spear"))) %>%
  select(-method, -alternative) %>%
  mutate(q = round(p.adjust(p.value, method="BH"), digits=3),
         q_value = ifelse(q<0.001, "<0.001", as.character(q)),
         estimate = round(estimate, 2)) %>%
  mutate(
    cytokine_type =
      case_when(
        name %in% c("IL8_log2", "IP10_log2",
                    "MIP1a_log2", "MIP1b_log2") ~ "innate/chemotax",
```

```
        name %in% "IL6_log2" ~ "acute inflam",
        name %in% "IL1ra_log2" ~ "anti-inflam",
        name %in% c("FGF_log2", "IL12p70_log2", "IL13_log2",
                    "IL17_log2", "IL4_log2", "IL5_log2", "IL7_log2",
                    "PDGF_log2", "RANTES_BC", "tgfb1_log2") ~ "t-cell")) %>%
  rename(var = name) -> rdf

rdf_imm <- rdf

# scatter
immune_markers %>%
  gather(key = var, value = value, 2:17) %>%
  ggplot(aes(blood_Xpert_CT, value)) +
  geom_point(colour="#DDCC77", alpha=0.5, size=0.9) +
  geom_smooth(colour="black") +
  facet_wrap(~var, scales = "free") +
  theme_minimal() +
  geom_text(data=rdf,
            aes(label = paste0("rho = ", estimate)),
            x=-Inf, y=Inf, hjust=0, vjust=1.2) +
  geom_text(data=rdf,
            aes(label = paste0("q = ", q_value)),
            x=Inf, y=-Inf, hjust=1, vjust=-1.2) +
  theme(strip.text = element_text(face = "bold")) +
  ylab("") + xlab("Blood Xpert Ultra Ct value")
```




```
immune_markers %>%
  mutate(xpt = ifelse(!is.na(blood_Xpert_CT),
                      "MTB", "NEG")) %>%
  select(- blood_Xpert_CT) %>% pivot_longer(-xpt) %>%
  group_by(name) %>%
  do(tidy(t.test(.$value, .$xpt=="MTB"))) %>%
  select(-method, -alternative) %>%
  mutate(q = round(p.adjust(p.value, method="BH"), digits=3),
         q_value = ifelse(q<0.001, "<0.001", as.character(q)),
         estimate = round(estimate, 2)) %>%
  mutate(
    cytokine_type =
      case_when(
        name %in% c("IL8_log2", "IP10_log2",
                    "MIP1a_log2", "MIP1b_log2") ~ "innate/chemotax",
        name %in% "IL6_log2" ~ "acute inflam",
        name %in% "IL1ra_log2" ~ "anti-inflam",
```

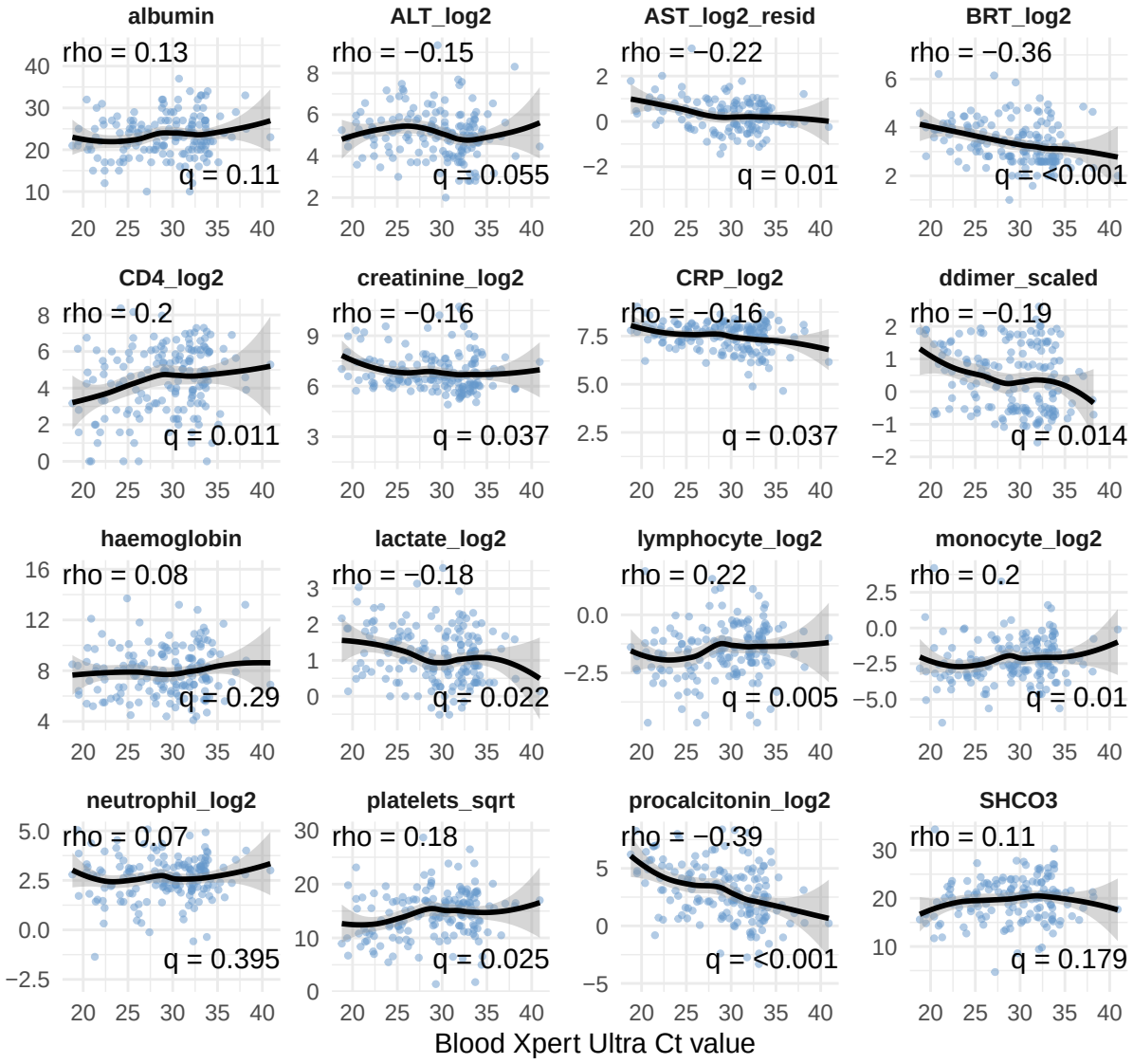
```
        name %in% c("FGF_log2", "IL12p70_log2", "IL13_log2",
                    "IL17_log2", "IL4_log2", "IL5_log2", "IL7_log2",
                    "PDGF_log2", "RANTES_BC", "tgfb1_log2") ~ "t-cell")) -> tdf

immune_markers %>%
    mutate(xpt = ifelse(!is.na(blood_Xpert_CT),
                        "MTB", "NEG")) %>%
    select(- blood_Xpert_CT) %>% pivot_longer(-xpt) %>%
    ggplot(aes(xpt, value)) +
    geom_quasirandom(colour="#DDCC77", alpha=0.25, size=0.7) +
    geom_boxplot(width=0.25, colour="#999933", fill="white", alpha=0.3,
                 outlier.alpha = 0) +
    facet_wrap(~name, scales = "free") +
    theme_minimal() +
    geom_text(data=tdf,
              aes(label = paste0("q ", q_value)),
              x=1.5, y=-Inf, hjust=0.5, vjust=-0.3) +
  theme(strip.text = element_text(face = "bold")) +
  xlab("Blood Xpert Ultra result") + ylab("")
```



Correlation with blood Xpert Ct values for each of the 16 clinical markers:

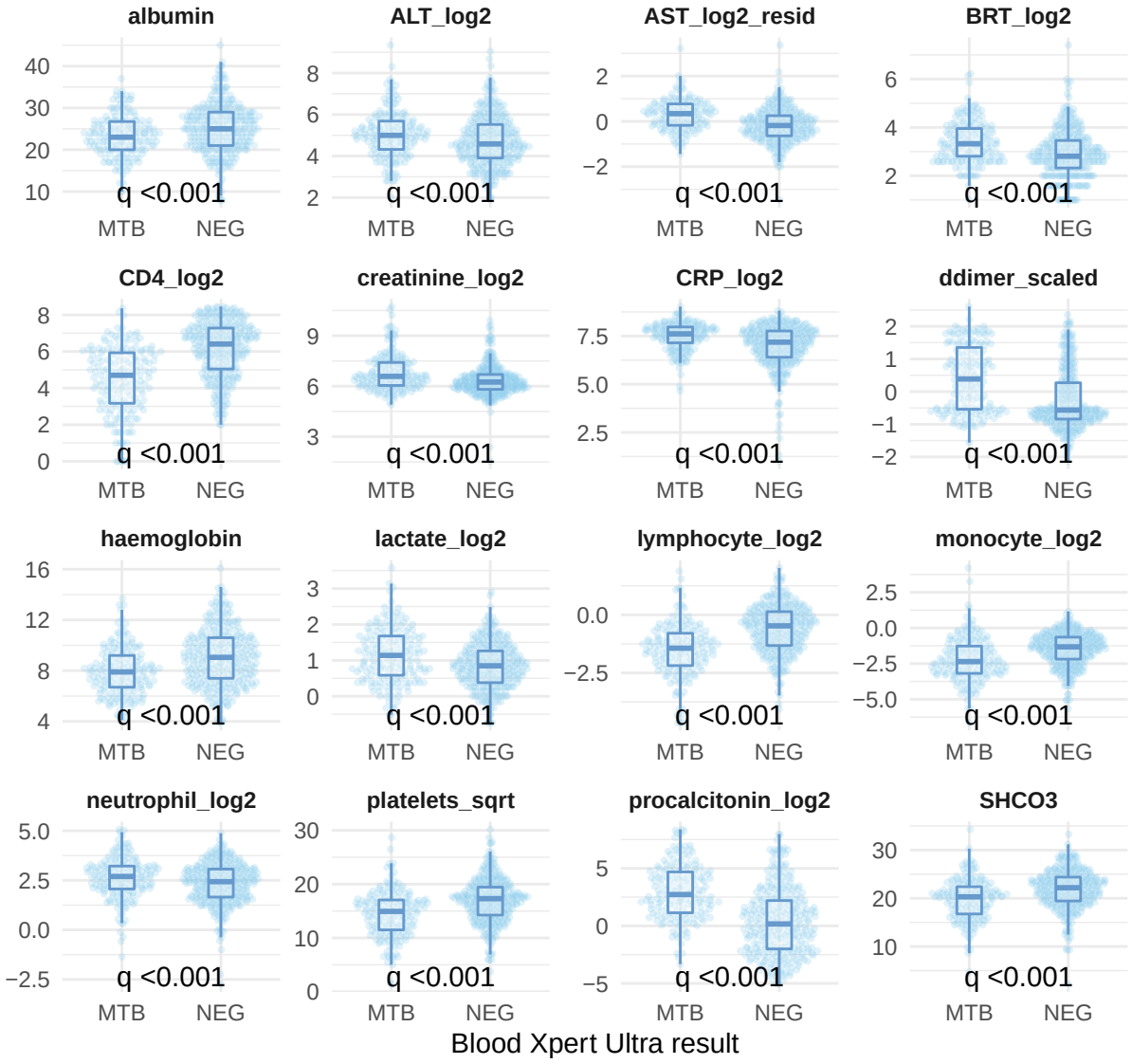
```
clin_markers %>%
  filter(blood_Xpert_CT<50) %>%
  pivot_longer(-blood_Xpert_CT) %>%
  group_by(name) %>%
  do(tidy(cor.test(.$blood_Xpert_CT, .$value,
                   use="complete.cases", method = "spear"))) %>%
  select(-method, -alternative) %>%
  mutate(q = round(p.adjust(p.value, method="BH"), digits=3),
         q_value = ifelse(q<0.001, "<0.001", as.character(q)),
         estimate = round(estimate, 2)) %>%
  mutate(
    marker_type =
      case_when(
        name %in% c("CRP_log2", "lactate_log2",
                    "procalcitonin_log2") ~ "acute inflam",
```

```
        name %in% c("albumin", "haemoglobin") ~ "chronic inflam",
        name %in% c("CD4_log2", "lymphocyte_log2", "monocyte_log2", "neutrophil_log2") ~ "WBC",
        name %in% c("ALT_log2", "BRT_log2") ~ "liver",
        name %in% c("ddimer_scaled", "platelets_sqrt") ~ "coagulation",
        name %in% c("creatinine_log2","SHCO3") ~ "renal",
        name %in% "AST_log2_resid" ~ "mitochondrial")) -> rdf

rdf_clin <- rdf

# scatter
clin_markers %>%
  gather(key = name, value = value, 2:17) %>%
  ggplot(aes(blood_Xpert_CT, value)) +
  geom_point(colour="#6699CC", alpha=0.5, size=0.8) +
  geom_smooth(colour="black") +
  facet_wrap(~name, scales = "free") +
  theme_minimal() +
  geom_text(data=rdf,
            aes(label = paste0("rho = ", estimate)),
            x=-Inf, y=Inf, hjust=0, vjust=1.2) +
  geom_text(data=rdf,
            aes(label = paste0("q = ", q_value)),
            x=Inf, y=-Inf, hjust=1, vjust=-1.2) +
  theme(strip.text = element_text(face = "bold")) +
  ylab("") + xlab("Blood Xpert Ultra Ct value")
```




```
clin_markers %>%
  mutate(xpt = ifelse(!is.na(blood_Xpert_CT),
                      "MTB", "NEG")) %>%
  select(- blood_Xpert_CT) %>% pivot_longer(-xpt) %>%
  group_by(name) %>%
  do(tidy(t.test(.$value, .$xpt=="MTB"))) %>%
  select(-method, -alternative) %>%
  mutate(q = round(p.adjust(p.value, method="BH"), digits=3),
         q_value = ifelse(q<0.001, "<0.001", as.character(q)),
         estimate = round(estimate, 2)) %>%
  mutate(
    marker_type =
      case_when(
        name %in% c("CRP_log2", "lactate_log2",
                    "procalcitonin_log2") ~ "acute inflam",
        name %in% c("albumin", "haemoglobin") ~ "chronic inflam",
        name %in% c("CD4_log2", "lymphocyte_log2", "monocyte_log2", "neutrophil_log2") ~ "WBC",
```

```
        name %in% c("ALT_log2", "BRT_log2") ~ "liver",
        name %in% c("ddimer_scaled", "platelets_sqrt") ~ "coagulation",
        name %in% c("creatinine_log2","SHCO3") ~ "renal",
        name %in% "AST_log2_resid" ~ "mitochondrial")) -> tdf

clin_markers %>%
    mutate(xpt = ifelse(!is.na(blood_Xpert_CT),
                        "MTB", "NEG")) %>%
    select(- blood_Xpert_CT) %>% pivot_longer(-xpt) %>%
    ggplot(aes(xpt, value)) +
    geom_quasirandom(colour="#88CCEE", alpha=0.25, size=0.7) +
    geom_boxplot(width=0.25, colour="#6699CC", fill="white", alpha=0.3,
                 outlier.alpha = 0) +
    facet_wrap(~name, scales = "free") +
    theme_minimal() +
    geom_text(data=tdf,
              aes(label = paste0("q ", q_value)),
              x=1.5, y=-Inf, hjust=0.5, vjust=-0.3) +
  xlab("Blood Xpert Ultra result") + ylab("") +
  theme(strip.text = element_text(face = "bold"))
```

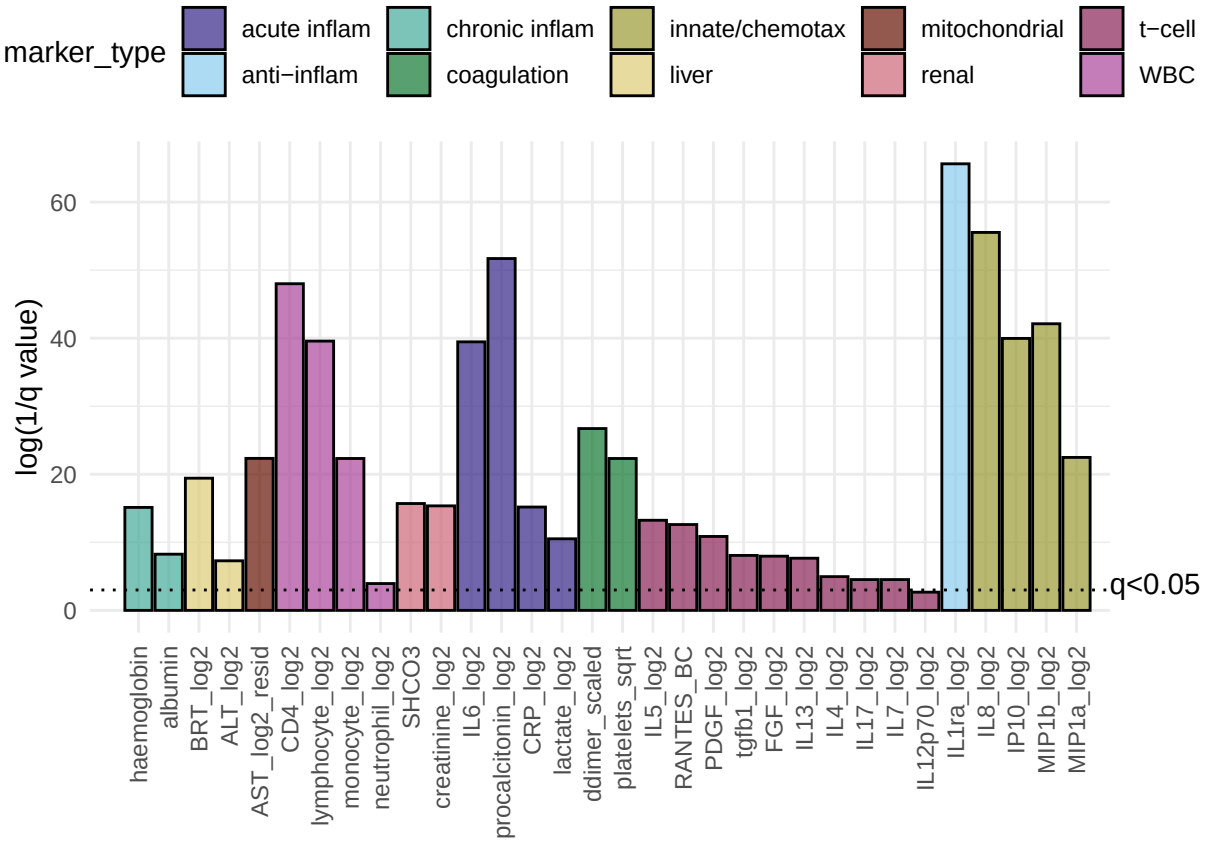


Manhattan plot showing strength of association with blood Xpert Ct values, cytokines grouped by approxiamte function.

```
### immune markers and ordinal scale bld xpert

immune_markers$bld_xpt_bin <- tbdf$bld_xpt_bin

immune_markers %>%
  select(-blood_Xpert_CT) %>%
  pivot_longer(-bld_xpt_bin) %>%
  group_by(name) %>%
  do(tidy(cor.test(.$bld_xpt_bin, .$value,
                   use="complete.cases", method = "spear"))) %>%
  select(-method, -alternative) %>%
  mutate(
    cytokine_type =
      case_when(
        name %in% c("IL8_log2", "IP10_log2",
```

```
                    "MIP1a_log2", "MIP1b_log2") ~ "innate/chemotax",
        name %in% "IL6_log2" ~ "acute inflam",
        name %in% "IL1ra_log2" ~ "anti-inflam",
        name %in% c("FGF_log2", "IL12p70_log2", "IL13_log2",
                    "IL17_log2", "IL4_log2", "IL5_log2", "IL7_log2",
                    "PDGF_log2", "RANTES_BC", "tgfb1_log2") ~ "t-cell")) %>%
  rename(var = name) -> rdf_imm

### clin markers and ordinal scale bld xpert

clin_markers$bld_xpt_bin <- tbdf$bld_xpt_bin

clin_markers %>%
  select(-blood_Xpert_CT) %>%
  pivot_longer(-bld_xpt_bin) %>%
  group_by(name) %>%
  do(tidy(cor.test(.$bld_xpt_bin, .$value,
                   use="complete.cases", method = "spear"))) %>%
  select(-method, -alternative) %>%
  mutate(
    marker_type =
      case_when(
        name %in% c("CRP_log2", "lactate_log2",
                    "procalcitonin_log2") ~ "acute inflam",
        name %in% c("albumin", "haemoglobin") ~ "chronic inflam",
        name %in% c("CD4_log2", "lymphocyte_log2", "monocyte_log2", "neutrophil_log2") ~ "WBC",
        name %in% c("ALT_log2", "BRT_log2") ~ "liver",
        name %in% c("ddimer_scaled", "platelets_sqrt") ~ "coagulation",
        name %in% c("creatinine_log2","SHCO3") ~ "renal",
        name %in% "AST_log2_resid" ~ "mitochondrial")) -> rdf_clin

### previously we multitle test adjusted these seperately but
### more reasonable to do together as one set so combine:

rdf_imm %>%
  rename(name = var, marker_type = cytokine_type) %>%
  mutate(set = "immuno") %>%
  bind_rows(rdf_clin %>% mutate(set = "clinico")) %>%
  mutate(marker_type = factor(marker_type)) -> foo

### BH correction
foo$q <- p.adjust(foo$p.value, method="BH")

### Manhattan plot:

foo %>%
  mutate(
    ordering_foo = q + as.numeric(as.factor(marker_type)),
    marker_type = factor(
      marker_type,
      levels = c("acute inflam", "anti-inflam", "chronic inflam",
                 "coagulation", "innate/chemotax", "liver",
                 "mitochondrial", "renal", "t-cell", "WBC")
```

```
    )
  ) %>%
ggplot(
  aes(reorder(name,  ordering_foo), log(1/q), fill=marker_type)
      ) +
geom_bar(colour="black", alpha=0.7, stat = "identity") +
theme_minimal() +
xlab("") + ylab("log(1/q value)") +
theme(axis.text.x =
        element_text(angle=90, hjust = 1, vjust=0.5),
      legend.position = "top") +
scale_fill_ptol() +
geom_hline(yintercept = log(1/(0.05)), linetype=3) +
annotate("text", x =Inf, y=4, hjust=0,
         label="q<0.05") +
coord_cartesian(xlim = c(0, 32.5),
                clip = 'off') +
theme(plot.margin = unit(c(1,3,1,1), "lines"))
```



10.2 Summarising clinico-immune phenotype by PCA dimension reduction

10.2.1 Imputation for PCA

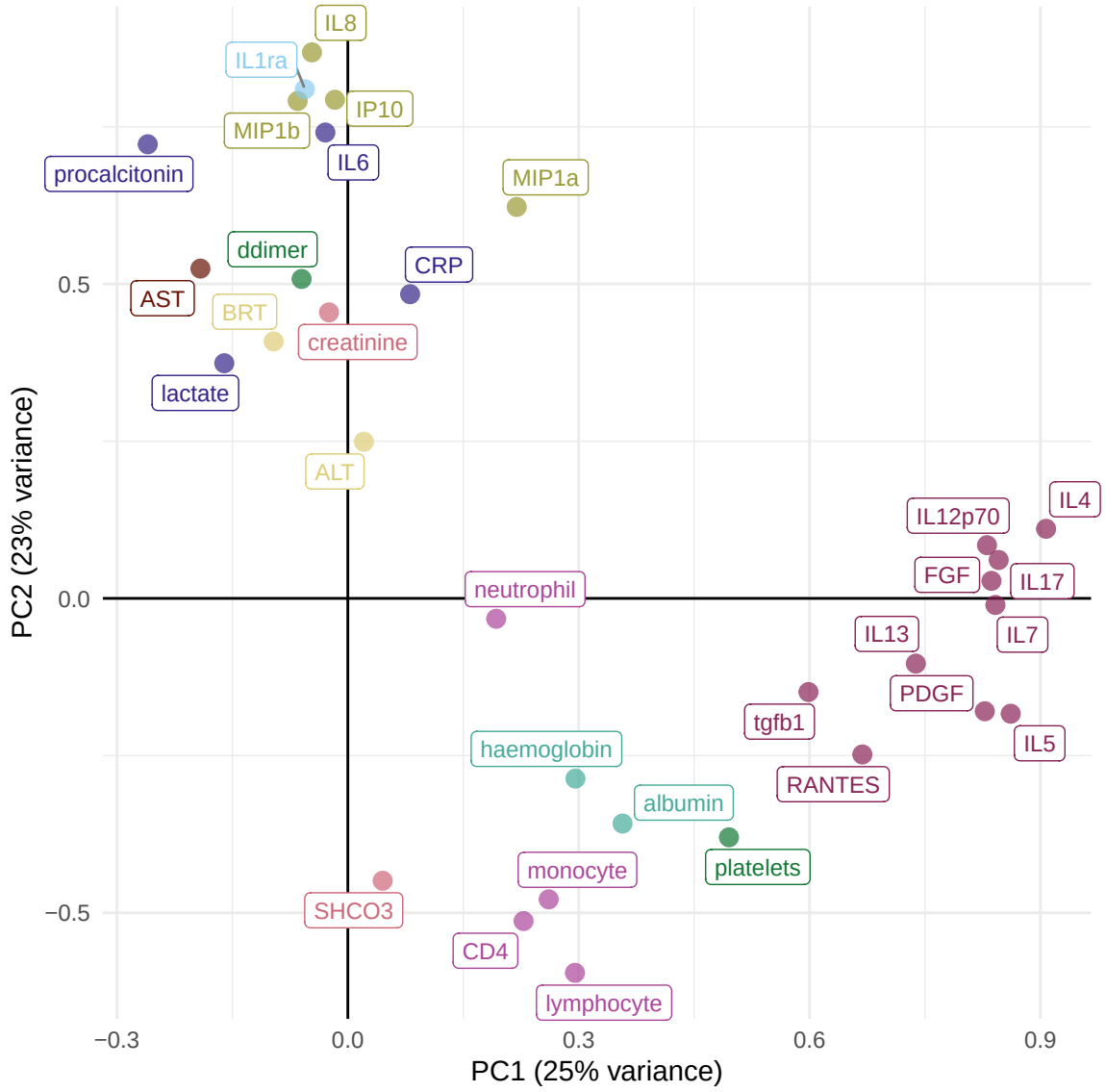
10.2.2 Varimax PCA

```
# varimax rotation PCA to summarise variance in two dimmensions
pc <- principal(phenotype_df[,-1], nfactors = 2, rotate="varimax")

pheno_pc <- data.frame(
  PC1 = pc$loadings[,1],
  PC2 =  pc$loadings[,2],
                  assay = names(pc$loadings[,1]))

# figure to show how markers load on these 2 PCs
pheno_pc %>%
  mutate(
    `Marker type` = case_when(
      assay %in% c("CRP_log2", "lactate_log2",
                   "procalcitonin_log2") ~ "Acute inflammation",
      assay %in% c("albumin", "haemoglobin") ~ "Chronic inflammation",
      assay %in% c(
        "CD4_log2",
        "lymphocyte_log2",
        "monocyte_log2",
        "neutrophil_log2"
      ) ~ "White cell counts",
      assay %in% c("ALT_log2", "BRT_log2") ~ "Liver",
      assay %in% c("ddimer_scaled", "platelets_sqrt") ~ "Coagulation",
      assay %in% c("creatinine_log2", "SHCO3") ~ "Renal",
      assay %in% "AST_log2_resid" ~ "Mitochondrial",
      assay %in% c("IL8_log2", "IP10_log2",
                   "MIP1a_log2", "MIP1b_log2") ~ "Innate/chemotaxis",
      assay %in% "IL6_log2" ~ "Acute inflammation",
      assay %in% "IL1ra_log2" ~ "Anti-inflammatory",
      assay %in% c(
        "FGF_log2",
        "IL12p70_log2",
        "IL13_log2",
        "IL17_log2",
        "IL4_log2",
        "IL5_log2",
        "IL7_log2",
        "PDGF_log2",
        "RANTES_BC",
        "tgfb1_log2"
      ) ~ "T-cell"
    )
  ) %>%
  mutate(
    assay = str_replace_all(assay,
                            c("_log2" = "",
                              "_BC" = "",
```

```
                              "_sqrt" = "",
                              "_scaled" = "",
                              "_resid" = ""))
  ) %>%
  ggplot(aes(PC1, PC2,
             colour = `Marker type`)) +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_point(size = 3, alpha = 0.7) +
  geom_label_repel(
    aes(PC1, PC2,
        label = assay),
    size=3,
#    box.padding = 0.35,
#    point.padding = 0.5,
    segment.color = 'grey50',
    show.legend = FALSE,
    xlim = c(-Inf, Inf), ylim = c(-Inf, Inf)
  ) +
  theme_minimal() + coord_cartesian(clip = "off") +
  theme(legend.position = "none") +
  scale_color_ptol() +
  xlab("PC1 (25% variance)") +
  ylab("PC2 (23% variance)") -> g_pca_loadings
```

10.2.3 Loadings of the variables on PCA

```
g_pca_loadings
```




```
# add each patients PC score to main TB dataframe
bind_cols(tbdf,
          data.frame(pc1 = pc$scores[, 1],
                     pc2 = pc$scores[, 2])) -> tbdf

# association of PCs with day84 death and blood xpert positivity
tdf <- data.frame(
  pc = c("PC1", "PC2"),
  auc =
  c(round(auc(roc(tbdf$day84death, tbdf$pc1)), 2),
    round(auc(roc(tbdf$day84death, tbdf$pc2)), 2),
    round(auc(roc(tbdf$bld_xpert_pos, tbdf$pc1)), 2),
    round(auc(roc(tbdf$bld_xpert_pos, tbdf$pc2)), 2)),
  p =
    c(signif(t.test(tbdf$pc1 ~ tbdf$day84death)$p.value, 2),
      signif(t.test(tbdf$pc2 ~ tbdf$day84death)$p.value, 2),
      signif(t.test(tbdf$pc1 ~ tbdf$bld_xpert_pos)$p.value, 2),
```

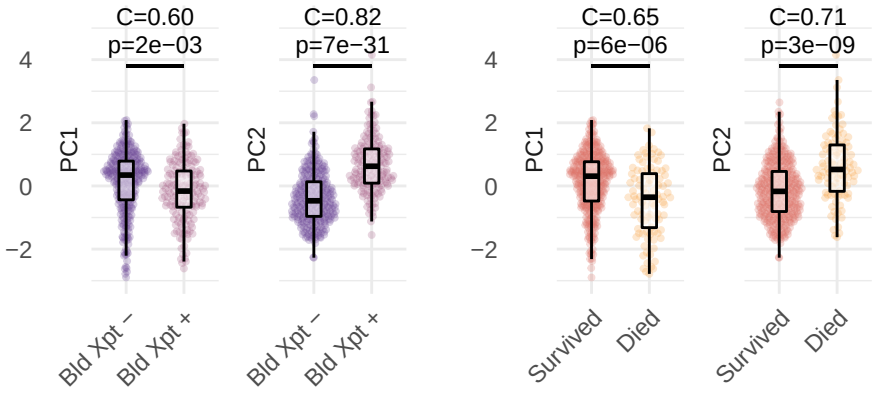
```
      signif(t.test(tbdf$pc2 ~ tbdf$bld_xpert_pos)$p.value, 2))
  ) %>%
  mutate(
    var = c("day84death", "day84death",
            "bld_xpert_pos", "bld_xpert_pos"),
    pval = paste0(
      "p=",
      formatC(p, format="e", digits=0)
    ),
    aucval = paste0(
      "C=",
      format(round(auc, 2), nsmall=2)
    )
  )


tbdf %>%
  filter(!is.na(day84death)) %>%
  transmute(`Day 84 outcome` = ifelse(day84death,
                                      "Died", "Survived"),
            `Day 84 outcome` = factor(
              `Day 84 outcome`, levels = c("Survived", "Died")),
            PC1 = pc1, PC2 = pc2
            ) %>%
  pivot_longer(2:3, names_to = "pc", values_to = "PC score") %>%
  ggplot(
    aes(`Day 84 outcome`, `PC score`)
  ) +
  geom_quasirandom(
    aes(colour=`Day 84 outcome`),
    alpha=0.25, size=0.7
    ) +
  geom_boxplot(
    colour="black", width=0.25,
    fill="white", alpha=0.5,
    outlier.alpha = 0
  ) +
  facet_wrap(~pc, strip.position = "left") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(
          angle = 45, hjust=1, vjust = 1)) +
  scale_color_manual(values = c("#de7065ff", "#f9a242ff")) +
  xlab("") + ylab("") +
  geom_segment(x=1, xend=2, y=3.8, yend=3.8) +
  geom_text(
    data = tdf %>%  filter(var=="day84death"),
    aes(label=aucval),
    x=1.5, y=5.1, size=3, vjust=0
  ) +
  geom_text(
    data = tdf %>%  filter(var=="day84death"),
    aes(label=pval),
```

```
    x=1.5, y=4.2, size=3, vjust=0
  ) +
  ylim(-3, 5.3) -> g_d84


tbdf %>%
  filter(!is.na(bld_xpert_pos)) %>%
  transmute(`Bld Xpert-ultra` = ifelse(bld_xpert_pos=="MTB",
                                       "Bld Xpt +", "Bld Xpt -"),
            PC1 = pc1, PC2 = pc2
            ) %>%
  pivot_longer(2:3, names_to = "pc", values_to = "PC score") %>%
  ggplot(
    aes(`Bld Xpert-ultra`, `PC score`)
  ) +
  geom_quasirandom(
    aes(colour=`Bld Xpert-ultra`),
    alpha=0.25, size=0.7
    ) +
  geom_boxplot(
    colour="black", width=0.25,
    fill="white", alpha=0.5,
    outlier.alpha = 0
  ) +
  facet_wrap(~pc, strip.position = "left") +
  theme_minimal() +
  theme(legend.position = "none",
        axis.text.x = element_text(
          angle = 45, hjust=1, vjust = 1)) +
  scale_color_manual(values = c("#6b4596ff", "#a65c85ff")) +
  xlab("") + ylab("") +
  geom_segment(x=1, xend=2, y=3.8, yend=3.8) +
  geom_text(
    data = tdf %>%  filter(var=="bld_xpert_pos"),
    aes(label=aucval),
    x=1.5, y=5.1, size=3, vjust=0
  ) +
  geom_text(
    data = tdf %>%  filter(var=="bld_xpert_pos"),
    aes(label=pval),
    x=1.5, y=4.2, size=3, vjust=0
  ) +
  ylim(-3, 5.3) -> g_bxpt
```

10.2.4 Clinical phenotype defined by PCA relationship to mortality and blood Xpert-Ultra results

Distribution of PC1 and PC2 scores with (a) blood Xpert-Ultra positivity, and (b) day 84 mortality.

```
g_bxpt | g_d84
```



Two dimesnion PCA space relationship with mortality:

```
tbdf %>%
  filter(!is.na(day84death)) %>%
  transmute(day84death = day84death==1,
            ul = pc1<=0 & pc2>0,
            ur = pc1>0 & pc2>0,
            ll = pc1<=0 & pc2<=0,
            lr = pc1>0 & pc2<=0
            ) %>% ungroup() %>%
  summarise(
    ul = sum(ul & day84death) / sum(ul),
    ur = sum(ur & day84death) / sum(ur),
    ll = sum(ll & day84death) / sum(ll),
    lr = sum(lr & day84death) / sum(lr)
  ) %>%
  pivot_longer(1:4) %>%
  mutate(
    value = paste0(
      format(round(value*100, 0), nsmall = 0),
      "%"
    ),
    x = c(-2, 1.5, -2, 1.5),
    y = c(3.5, 3.5, -3, -3)
  ) -> quads_df

tbdf %>%
  filter(!is.na(day84death)) %>%
  transmute(day84death = day84death==1,
            ul = pc1<=0 & pc2>0,
            ur = pc1>0 & pc2>0,
            ll = pc1<=0 & pc2<=0,
            lr = pc1>0 & pc2<=0
            ) %>% ungroup() %>%
  mutate(
    quad = case_when(
```

```
      ul ~ "ul",
      ur ~ "ur",
      ll ~ "ll",
      lr ~ "lr"
    )
  ) -> foo

# comparing UL and LR quadrant of PCA space with mortality, Fisher's test:
foo <- filter(foo, quad=="ul"|quad=="lr")
fisher.test(table(foo$quad, foo$day84death))
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  table(foo$quad, foo$day84death)
## p-value = 6.792e-11
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##   4.680055 28.678046
## sample estimates:
## odds ratio
##   10.90952
```

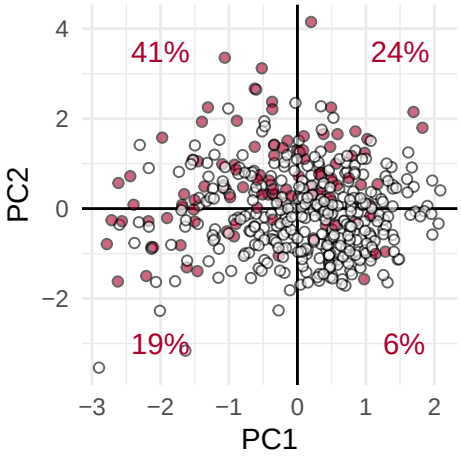
```
# comparing UL and LR quadrant of PCA space with mortality, GLM:
summary(glm(day84death ~ quad, family = "binomial", data=foo))
```

```
##
## Call:
## glm(formula = day84death ~ quad, family = "binomial", data = foo)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0273  -0.6884  -0.3495  -0.3495   2.3773
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.7647     0.3645  -7.585 3.33e-14 ***
## quadul        2.4008     0.4174   5.752 8.82e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 240.63  on 234  degrees of freedom
## Residual deviance: 196.10  on 233  degrees of freedom
## AIC: 200.1
##
## Number of Fisher Scoring iterations: 5
```

Visualising this with bivariate plot (patienst who died filled red, with % mortality by quadrant):

```
tbdf %>%
  filter(!is.na(day84death)) %>%
  transmute(`Day 84 outcome` = ifelse(day84death,
                                      "Died", "Survived"),
```

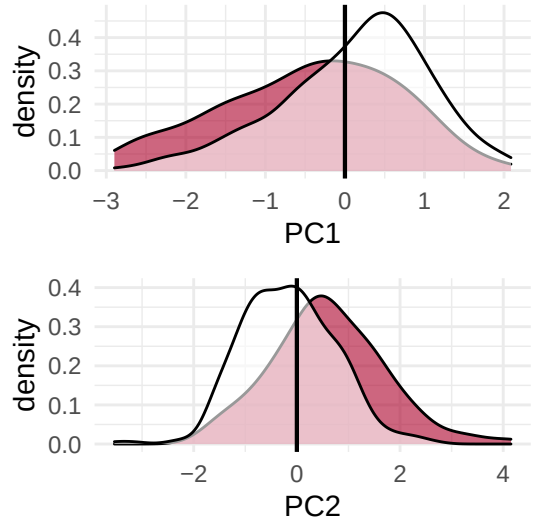
```
            PC1 = pc1, PC2 = pc2
            ) %>%
  ggplot(
    aes(PC1, PC2)
  ) +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_point(
    aes(fill=`Day 84 outcome`),
    shape=21, alpha=0.6
    ) +
  theme_minimal() + theme(legend.position = "none") +
  scale_fill_manual(values = c("#AD002AFF", "white")) +
  geom_text(
    data=quads_df, aes(x=x, y=y, label=value), colour="#AD002AFF") -> g_quads

g_quads
```



Distribution of PC1 and 2 by mortality again, but as density plot :

```
tbdf %>%
  filter(!is.na(day84death)) %>%
  transmute(`Day 84 outcome` = ifelse(day84death,
                                      "Died", "Survived"),
            PC1 = pc1, PC2 = pc2
            ) %>%
  ggplot(
    aes(PC1,
        fill=`Day 84 outcome`)
  ) +
  geom_density(colour="black",
               alpha=0.6,
               size=0.5
               ) +
  theme_minimal() + theme(legend.position = "none") +
  scale_fill_manual(values = c("#AD002AFF", "white")) +
  geom_vline(xintercept = 0, size=0.75) -> g_dens_pc1

tbdf %>%
  filter(!is.na(day84death)) %>%
```

```
  transmute(`Day 84 outcome` = ifelse(day84death,
                                      "Died", "Survived"),
            PC1 = pc1, PC2 = pc2
            ) %>%
  ggplot(
    aes(PC2,
        fill=`Day 84 outcome`)
  ) +
  geom_density(colour="black",
               alpha=0.6,
               size=0.5
               ) +
  theme_minimal() + theme(legend.position = "none") +
  scale_fill_manual(values = c("#AD002AFF", "white")) +
  geom_vline(xintercept = 0, size=0.75) -> g_dens_pc2

g_dens_pc1 / g_dens_pc2
```



Get odds ratios for moratlity for a 1 IQR change in PC 1 and PC 2:

```
iqr_or = function(var){
  x <- tbdf[[var]]
  iqr = IQR(x, na.rm = T)
  m = glm(day84death ~ x, data=tbdf, family="binomial")
  c(
    formatC( exp(iqr * (summary(m)$coefficients[2])), digits = 2 ),
    formatC( summary(m)$coefficients[8], format = "e", digits = 0 )
    )
}

iqr_or("pc1")
```

```
## [1] "0.47"  "1e-06"
```

```
iqr_or("pc2")
```

```
## [1] "  3"   "4e-10"
```

Linear correlation for blood Xpert-Ultra Ct value with pc1 and pc2:

```
rdf <- data.frame(
  pc = c("pc1", "pc2"),
  r = c(round(cor.test(tbdf$blood_Xpert_CT, tbdf$pc1)$estimate, 2),
      round(cor.test(tbdf$blood_Xpert_CT, tbdf$pc2)$estimate, 2)),
  p = c(signif(cor.test(tbdf$blood_Xpert_CT, tbdf$pc1)$p.value, 2),
      signif(cor.test(tbdf$blood_Xpert_CT, tbdf$pc2)$p.value, 2))
  )


#tbdf %>%
#  filter(!is.na(day84death) & !is.na(blood_Xpert_CT)) %>%
#  dplyr::select(day84death, blood_Xpert_CT, pc1, pc2) %>%
#  gather(key = pc, value = score, 3:4) %>%
#  ggplot(aes(blood_Xpert_CT, score)) +
#  geom_vline(xintercept = median(tbdf$blood_Xpert_CT, na.rm = TRUE),
#             linetype = 2) +
#  geom_hline(yintercept = 0, linetype = 2) +#
#  geom_point(
#    colour="grey", alpha = 0.5
#  ) +
#  geom_smooth(colour = "#AD002AFF", fill = "#AD002AFF", span=1, alpha=0.5) +
#  theme_minimal() +
#  xlab("Blood Xpert Ct value") +
#  ylab("") +
#  geom_text(data = rdf,
#            aes(label = paste0("r = ", r)),
#            x = 35,
#            y = 2) +
#  geom_text(data = rdf,
#            aes(label = paste0("p = ", p)),
#            x = 35,
#            y = 1.5) +
#  theme(strip.text = element_text(face = "bold")) +
#  facet_wrap(~ pc, strip.position = "left", scales = "free") -> g_pc_Ct


tbdf %>%
  filter(!is.na(blood_Xpert_CT)) %>%
  ggplot(aes(blood_Xpert_CT, pc1)) +
  geom_vline(xintercept = median(tbdf$blood_Xpert_CT, na.rm = TRUE),
             linetype = 2) +
  geom_hline(yintercept = 0, linetype = 2) +
  geom_point(
    colour="grey20", alpha = 0.5,  fill="grey", shape=21
  ) +
  geom_smooth(colour = "#AD002AFF", fill = "#AD002AFF", span=1, alpha=0.5) +
  theme_minimal() +
  xlab("") +
  ylab("") +
  theme(axis.line.x.bottom = element_line(colour = "black")) + ylim(-3, 2.25) -> g_pc1ct
```

```
tbdf %>%
  filter(bld_xpert_pos=="NEG") %>%
  ggplot(aes(bld_xpert_pos, pc1)) +
  geom_hline(yintercept = 0, linetype = 2) +
  geom_quasirandom(
    colour="grey20", alpha = 0.25,  fill="grey", shape=21
  ) +
  geom_boxplot(colour = "#AD002AFF",
               fill = "#AD002AFF",
               width=0.3, alpha=0.3, outlier.alpha = 0) +
  theme_minimal() +
  xlab("") +
  ylab("") +
  theme(axis.line.x.bottom = element_line(colour = "black")) + ylim(-3, 2.25) -> g_bxpc1

tbdf %>%
  filter(!is.na(blood_Xpert_CT)) %>%
  ggplot(aes(blood_Xpert_CT, pc2)) +
  geom_vline(xintercept = median(tbdf$blood_Xpert_CT, na.rm = TRUE),
             linetype = 2) +
  geom_hline(yintercept = 0, linetype = 2) +
  geom_point(
    colour="grey20", alpha = 0.5, fill="grey", shape=21
  ) +
  geom_smooth(colour = "#AD002AFF", fill = "#AD002AFF", span=1, alpha=0.5) +
  theme_minimal() +
  xlab("") +
  ylab("") +
  theme(axis.line.x.bottom = element_line(colour = "black")) + ylim(-2.1, 4.2) -> g_pc2ct

tbdf %>%
  filter(bld_xpert_pos=="NEG") %>%
  ggplot(aes(bld_xpert_pos, pc2)) +
  geom_hline(yintercept = 0, linetype = 2) +
  geom_quasirandom(
    colour="grey20", alpha = 0.25, fill="grey", shape=21
  ) +
  geom_boxplot(colour = "#AD002AFF",
               fill = "#AD002AFF",
               width=0.3, alpha=0.3, outlier.alpha = 0) +
  theme_minimal() +
  xlab("") +
  ylab("") +
  theme(axis.line.x.bottom = element_line(colour = "black")) + ylim(-2.1, 4.2) -> g_bxpc2
```

Pearson's correllation for Ct value and pc1 and pc2:

```
rdf %>% kable(booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

```
(g_pc1ct | g_bxpc1 | g_pc2ct | g_bxpc2) + plot_layout(widths = c(1,0.3, 1, 0.3))
```

pc	r	p
pc1	0.11	0.15
pc2	-0.48	0.00