ABG-VBG Analysis

Brian Locke, Anila Mehta

Table of contents

1 Data Pre-processing

This code pulls in the master database (a STATA file) and does some initial cleaning - this will only need to be run once, and then the data can be accessed in the usual way.

1.1 1) Seed escrow (reproducibility anchors)

Table 1: Seed escrow for MI, GBM, and SHAP runs

component	seed
Multiple imputation (mice)	20251206
ABG propensity GBM (non-MI)	42
VBG propensity GBM (non-MI)	42
SHAP (fastshap/shapviz)	123
MI GBM seeds (ABG per imputation)	20251206 + imputation index
MI GBM seeds (VBG per imputation)	30251206 + imputation index

Chunk seed-escrow runtime: 0.00 s

Chunk gt-pdf-helper runtime: 0.00 s

Converts the data from a STATA format to rdata if the rdata file does not exist. If it does already exist, it just loads that.

```
# data_dir_name <- '/Users/blocke/Box Sync/Residency Personal Files/Scholarly Work/Locke Research Projects/abg-vbg-project/data'
data_dir_name <- '/Users/reblocke/Research/abg-vbg-project/data'

rdata_file <- file.path(data_dir_name, "full_trinetx.rdata")
stata_file <- file.path(data_dir_name, "full_db.dta")

if (!dir.exists(data_dir_name)) {
  dir.create(data_dir_name)
  message("Directory 'data' created.")
} else {
  message("Directory 'data' already exists.")
}
```

```
Directory 'data' already exists.
```

```r
if (file.exists(rdata_file)) {
  load(rdata_file)
  message("Loaded existing dataset from 'full_trinetx.rdata'.")
} else {
  message("RData file not found. Reading Stata dataset...")
  stata_data <- read_dta(stata_file)

  message("Extracting variable labels...")
  var_label(stata_data)

  message("Extracting value labels...")
  sapply(stata_data, function(x) if (is.labelled(x)) val_labels(x))

  save(stata_data, file = rdata_file)
  message("Dataset saved as 'full_trinetx.rdata'.")

  load(rdata_file)
  message("Loaded newly saved dataset from 'full_trinetx.rdata'.")
}
```

```
Loaded existing dataset from 'full_trinetx.rdata'.
```

Chunk load-trinetx-data runtime: 7.55 s

```r
covars_gbm <- c(
  "age_at_encounter","sex","race_ethnicity","curr_bmi",
  "copd","asthma","osa","chf","acute_nmd","phtn","ckd","dm",
  "location","encounter_type","temp_new","sbp","dbp","hr","spo2",
  "sodium","serum_cr","serum_hco3","serum_cl","serum_lac","serum_k",
  "wbc","plt","bnp","serum_phos","serum_ca"
)

gbm_params <- list(
  n.trees           = 1500,
  interaction.depth = 3,
```

```
  shrinkage          = 0.01,
  bag.fraction       = 0.8,
  cv.folds           = 5,
  stop.method        = "es.mean",
  n.cores            = parallel::detectCores()
)

formula_abg      <- reformulate(covars_gbm, response = "has_abg")
formula_vbg      <- reformulate(covars_gbm, response = "has_vbg")
```

Chunk propensity-config runtime: 0.01 s

Creating subset_data

```
set.seed(123)
rows_to_keep <- round(nrow(stata_data) * 0.01) #1 for real run
subset_data <- stata_data[sample(nrow(stata_data), rows_to_keep), ]

subset_data <- subset_data %>%
  filter(encounter_type != 1)

table(subset_data$encounter_type)
```

```
   2    3
1754 3357
```

```
dim(subset_data)
```

```
[1] 5111  546
```

Chunk sample-subset-data runtime: 1.05 s

Generating Codebook for the Full Dataset

```r
message("Generating codebook for the dataset...")
```

```
Generating codebook for the dataset...
```

```r
study_codebook <- codebookr::codebook(
  stata_data,
  title = "Full TrinetX",
  subtitle = "Dataset Documentation",
  description = "This dataset contains patient-level records from the TrinetX database.
                 It has been processed and converted from the original Stata file."
)
codebook_file <- file.path(data_dir_name, "codebookr.docx")
print(study_codebook, codebook_file)
message("Codebook saved as 'codebookr.docx' in the data directory.")
```

```
Codebook saved as 'codebookr.docx' in the data directory.
```

Chunk codebook-export-full runtime: 103.44 s

New Variable - Death at 60 days

```r
subset_data <- subset_data %>%
  mutate(
    ## 1. Did the patient die?
    died = if_else(!is.na(death_date), 1L, 0L),

    ## 2. Absolute death date (if death_date is an offset)
    death_abs = if_else(!is.na(death_date),
                        encounter_date + death_date,
                        as.Date(NA)),

    ## 3. Year month (YM) for encounter and death
    enc_ym   = floor_date(encounter_date, unit = "month"),
    death_ym = floor_date(death_abs      , unit = "month"),
```

```
    ## 4. Reference censoring date: 1 Jun 2024
    ref_ym = ymd("2024-06-01"),

    ## 5. Months from encounter to death or censoring
    months_death_or_cens = case_when(
      !is.na(death_ym) ~ interval(enc_ym, death_ym) %/% months(1),
      TRUE             ~ interval(enc_ym, ref_ym)   %/% months(1)
    ),

    ## 6. Remove impossible values
    months_death_or_cens = if_else(
      months_death_or_cens < 0 | months_death_or_cens > 16,
      NA_integer_, months_death_or_cens
    ),

    ## 7. Death within one or two months
    died_1mo = if_else(died == 1 & months_death_or_cens <  1, 1L, 0L),
    died_2mo = if_else(died == 1 & months_death_or_cens <= 1, 1L, 0L),

    ## 8. Month of death (missing if censored)
    death_time = if_else(died == 1, months_death_or_cens, NA_integer_),

    ## 9. Death within 60 days (new variable)
    death_60d = if_else(died == 1 & death_abs <= (encounter_date + days(60)), 1L, 0L)
  ) %>%
  select(-enc_ym, -death_ym)

subset_data <- subset_data %>%
  mutate(
    death_60d = if_else(died == 1 & death_abs <= (encounter_date + days(60)), 1L, 0L)
  )
```

Chunk derive-death-60d runtime: 0.05 s

```r
table(subset_data$death_60d, useNA = "ifany")
```

```
   0    1
4569  542
```

```r
prop.table(table(subset_data$death_60d, useNA = "ifany"))
```

```
        0         1
0.8939542 0.1060458
```

```r
summary(subset_data$death_60d)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.000   0.000   0.000   0.106   0.000   1.000
```

Chunk death-60d-summary runtime: 0.00 s

1.2 2) Baseline tables

1.2.1 2.1 Table 1A and 1B:

```r
# Robust derivation of analysis variables + helper for Table 1 production
# ------------------------------------------------------------------------

# helper: label binary 0/1 → "No"/"Yes"
bin_lab <- function(x) factor(x, levels = c(0, 1), labels = c("No", "Yes"))

subset_data <- subset_data %>%
  mutate(
```

```r
    ## ensure 0/1 numerics (avoids factor-level coercion)
    across(c(has_abg, has_vbg, hypercap_on_abg, hypercap_on_vbg),
           ~ as.numeric(as.character(.))),

    ## derive ABG / VBG hypercapnia groups
    abg_group
    = case_when(
      has_abg == 0                          ~ "No ABG",
      has_abg == 1 & hypercap_on_abg == 0   ~ "ABG_NoHypercapnia",
      has_abg == 1 & hypercap_on_abg == 1   ~ "ABG_Hypercapnia",
      TRUE                                  ~ "Missing"
    ),
    vbg_group = case_when(
      has_vbg == 0                          ~ "No VBG",
      has_vbg == 1 & hypercap_on_vbg == 0   ~ "VBG_NoHypercapnia",
      has_vbg == 1 & hypercap_on_vbg == 1   ~ "VBG_Hypercapnia",
      TRUE                                  ~ "Missing"
    ),

    ## factorise groups with explicit NA/Missing level
    abg_group = factor(
      abg_group,
      levels = c("No ABG", "ABG_NoHypercapnia", "ABG_Hypercapnia", "Missing")
    ),
    vbg_group = factor(
      vbg_group,
      levels = c("No VBG", "VBG_NoHypercapnia", "VBG_Hypercapnia", "Missing")
    ),

    ## labelled covariates
    sex_label      = factor(sex, levels = c(0, 1), labels = c("Female", "Male")),
    race_ethnicity_label     = factor(
      race_ethnicity,
      levels = c(0, 1, 2, 3, 4, 5, 6),
      labels = c("White", "Black or African American", "Hispanic",
                 "Asian", "American Indian", "Pacific Islander", "Unknown")
```

```
    ), location_label     = factor(
      location,
      levels = c(0, 1, 2, 3),
      labels = c("South", "Northeast" ,"Midwest", "West")
    ), encounter_type_label = factor(
      encounter_type,
      levels = c(2, 3),
      labels = c("Emergency", "Inpatient")
    ),
    osa_label      = bin_lab(osa),
    asthma_label   = bin_lab(asthma),
    copd_label     = bin_lab(copd),
    chf_label      = bin_lab(chf),
    nmd_label      = bin_lab(nmd),
    phtn_label     = bin_lab(phtn),
    ckd_label      = bin_lab(ckd),
    diabetes_label = bin_lab(dm)
  )

# variables to summarise
vars <- c(
  "age_at_encounter", "curr_bmi", "sex_label", "race_ethnicity_label", "location_label",
  "osa_label", "asthma_label", "copd_label", "chf_label", "nmd_label",
  "phtn_label", "ckd_label", "diabetes_label", "encounter_type_label", "vbg_co2", "paco2"
)

# Table 1 constructor
make_table1 <- function(data, group_var, caption = "") {
  group_sym <- rlang::sym(group_var)

  data %>%
    filter(!is.na(!!group_sym),                    # drop explicit NA
           !!group_sym != "Missing") %>%           # drop "Missing" cohort
    droplevels() %>%                               # trim empty factor levels
    select(all_of(c(group_var, vars))) %>%
    gtsummary::tbl_summary(
```

```
      by   = !!group_sym,
      type = list(sex_label ~ "categorical"),
      statistic = list(
        gtsummary::all_continuous()  ~ "{mean} ± {sd}; {N_miss}/{N_obs} missing ({p_miss}%)",
        gtsummary::all_categorical() ~ "{n} ({p}%)"
      ),
      digits   = list(gtsummary::all_continuous() ~ 1),
      missing  = "no"                                  # no gtsummary missing column/row
    ) %>%
    gtsummary::modify_header(label = "**Variable**") %>%
    gtsummary::modify_caption(caption)
}

# build tables
table1A <- make_table1(subset_data, "abg_group", caption = "Table 1A: ABG cohorts")
table1B <- make_table1(subset_data, "vbg_group", caption = "Table 1B: VBG cohorts")

table1A
```

```
table1B
```

Chunk derive-table1-cohorts runtime: 1.57 s

Generating Word Doc for Table 1A & 1B

```
ft_table1A <- as_flex_table(table1A)
ft_table1B <- as_flex_table(table1B)

doc <- read_docx() %>%
  body_add_par("Table 1A. Baseline Characteristics by ABG Group", style = "heading 1") %>%
  body_add_flextable(ft_table1A) %>%
  body_add_par("Table 1B. Baseline Characteristics by VBG Group", style = "heading 1") %>%
  body_add_flextable(ft_table1B)

print(doc, target = "Table1_ABG_VBG.docx")
```

Chunk export-table1a-table1b-word runtime: 0.58 s

Variable	**No ABG** N = 3,321[1]	**ABG_NoHypercapnia** N = 1,245[1]	**ABG_Hypercapnia** N = 545[1]
Age (years)	57.5 ± 18.3; 0.0/3,321.0 missing (0.0%)	60.2 ± 17.4; 0.0/1,245.0 missing (0.0%)	61.5 ± 16.2; 0.0/545.0 missing (0.0%)
Current BMI kg/m2	32.7 ± 8.8; 1,802.0/3,321.0 missing (54.3%)	28.2 ± 6.8; 728.0/1,245.0 missing (58.5%)	29.6 ± 7.8; 332.0/545.0 missing (60.9%)
sex_label			
Female	1,728 (52%)	557 (45%)	255 (47%)
Male	1,593 (48%)	688 (55%)	290 (53%)
race_ethnicity_label			
White	2,000 (60%)	750 (60%)	366 (67%)
Black or African American	648 (20%)	207 (17%)	88 (16%)
Hispanic	246 (7.4%)	75 (6.0%)	29 (5.3%)
Asian	48 (1.4%)	25 (2.0%)	4 (0.7%)
American Indian	22 (0.7%)	17 (1.4%)	4 (0.7%)
Pacific Islander	2 (<0.1%)	2 (0.2%)	2 (0.4%)
Unknown	355 (11%)	169 (14%)	52 (9.5%)
location_label			
South	1,403 (42%)	684 (55%)	298 (55%)
Northeast	906 (27%)	227 (18%)	139 (26%)
Midwest	235 (7.1%)	97 (7.8%)	50 (9.2%)
West	777 (23%)	237 (19%)	58 (11%)
osa_label	606 (18%)	148 (12%)	114 (21%)
asthma_label	480 (14%)	120 (9.6%)	78 (14%)
copd_label	610 (18%)	185 (15%)	181 (33%)
chf_label	584 (18%)	243 (20%)	147 (27%)
nmd_label	124 (3.7%)	61 (4.9%)	20 (3.7%)
phtn_label	239 (7.2%)	90 (7.2%)	61 (11%)
ckd_label	606 (18%)	251 (20%)	109 (20%)
diabetes_label	952 (29%)	351 (28%)	162 (30%)
encounter_type_label			
Emergency	1,485 (45%)	186 (15%)	83 (15%)
Inpatient	1,836 (55%)	1,059 (85%)	462 (85%)
VBG PCO2	45.0 ± 10.9; 2,420.0/3,321.0 missing (72.9%)	41.9 ± 10.9; 882.0/1,245.0 missing (70.8%)	59.0 ± 21.3; 377.0/545.0 missing (69.2%)
Arterial PCO2	NA ± NA; 3,321.0/3,321.0 missing (100.0%)	35.4 ± 6.1; 0.0/1,245.0 missing (0.0%)	60.7 ± 24.4; 0.0/545.0 missing (0.0%)

[1]Mean ± SD; N Missing/No. obs. missing (% Missing); n (%)

Variable	**No VBG** N = 3,679[1]	**VBG_NoHypercapnia** N = 1,036[1]	**VBG_Hypercapnia** N = 396[1]
Age (years)	58.7 ± 18.2; 0.0/3,679.0 missing (0.0%)	57.6 ± 17.3; 0.0/1,036.0 missing (0.0%)	59.8 ± 17.2; 0.0/396.0 missing (0.0%)
Current BMI kg/m2	32.2 ± 8.6; 1,915.0/3,679.0 missing (52.1%)	28.2 ± 7.1; 675.0/1,036.0 missing (65.2%)	29.3 ± 8.2; 272.0/396.0 missing (68.7%)
sex_label			
Female	1,851 (50%)	499 (48%)	190 (48%)
Male	1,828 (50%)	537 (52%)	206 (52%)
race_ethnicity_label			
White	2,409 (65%)	492 (47%)	215 (54%)
Black or African American	651 (18%)	206 (20%)	86 (22%)
Hispanic	238 (6.5%)	86 (8.3%)	26 (6.6%)
Asian	38 (1.0%)	31 (3.0%)	8 (2.0%)
American Indian	21 (0.6%)	18 (1.7%)	4 (1.0%)
Pacific Islander	5 (0.1%)	1 (<0.1%)	0 (0%)
Unknown	317 (8.6%)	202 (19%)	57 (14%)
location_label			
South	1,962 (53%)	283 (27%)	140 (35%)
Northeast	652 (18%)	453 (44%)	167 (42%)
Midwest	258 (7.0%)	80 (7.7%)	44 (11%)
West	807 (22%)	220 (21%)	45 (11%)
osa_label	638 (17%)	150 (14%)	80 (20%)
asthma_label	490 (13%)	132 (13%)	56 (14%)
copd_label	721 (20%)	141 (14%)	114 (29%)
chf_label	677 (18%)	189 (18%)	108 (27%)
nmd_label	160 (4.3%)	30 (2.9%)	15 (3.8%)
phtn_label	284 (7.7%)	67 (6.5%)	39 (9.8%)
ckd_label	668 (18%)	216 (21%)	82 (21%)
diabetes_label	1,020 (28%)	320 (31%)	125 (32%)
encounter_type_label			
Emergency	1,276 (35%)	348 (34%)	130 (33%)
Inpatient	2,403 (65%)	688 (66%)	266 (67%)
VBG PCO2	NA ± NA; 3,679.0/3,679.0 missing (100.0%)	39.8 ± 6.6; 0.0/1,036.0 missing (0.0%)	61.5 ± 14.4; 0.0/396.0 missing (0.0%)
Arterial PCO2	42.6 ± 17.7; 2,420.0/3,679.0 missing (65.8%)	39.9 ± 17.4; 670.0/1,036.0 missing (64.7%)	54.6 ± 22.0; 231.0/396.0 missing (58.3%)

[1]Mean ± SD; N Missing/No. obs. missing (% Missing); n (%)

1.2.2 2.2 Table 1 (Overall ABG/VBG status)

```
# Status factors (column labels are taken from factor levels)
subset_data <- subset_data %>%
  mutate(
    abg_status = factor(has_abg, levels = c(0, 1),
                        labels = c("Did not get ABG", "Did get ABG")),
    vbg_status = factor(has_vbg, levels = c(0, 1),
                        labels = c("Did not get VBG", "Did get VBG"))
  )

# ABG table with "Everyone" column first
tbl1_abg <- subset_data %>%
  select(all_of(vars), abg_status) %>%
  gtsummary::tbl_summary(
    by = abg_status,
    type = list(sex_label ~ "categorical"),
    statistic = list(
      gtsummary::all_continuous()  ~ "{mean} ± {sd}; {N_miss}/{N_obs} missing ({p_miss}%)",
      gtsummary::all_categorical() ~ "{n} ({p}%)"
    ),
    digits   = list(gtsummary::all_continuous() ~ 1),
    missing  = "no"
  ) %>%
  gtsummary::add_overall(last = FALSE, col_label = "Everyone") %>%
  gtsummary::modify_header(label = "**Variable**")

# VBG table (no "Everyone" here)
tbl1_vbg <- subset_data %>%
  select(all_of(vars), vbg_status) %>%
  gtsummary::tbl_summary(
    by = vbg_status,
    type = list(sex_label ~ "categorical"),
    statistic = list(
      gtsummary::all_continuous()  ~ "{mean} ± {sd}; {N_miss}/{N_obs} missing ({p_miss}%)",
```

```
      gtsummary::all_categorical() ~ "{n} ({p}%)"
    ),
    digits   = list(gtsummary::all_continuous() ~ 1),
    missing  = "no"
  ) %>%
  gtsummary::modify_header(label = "**Variable**")

library(gtsummary)

tbl1 <- tbl_merge(
  tbls = list(tbl1_abg, tbl1_vbg)
) %>%
  modify_caption("**Table 1. Baseline summary: Everyone, ABG status, and VBG status**")

tbl1
```

Chunk table1-everyone-abg-vbg runtime: 1.72 s

1.2.3 2.3 Table 2 (Hypercapnia within cohorts)

```
# Hypercapnia factors within measured cohorts
subset_data <- subset_data %>%
  mutate(
    hyper_abg = factor(hypercap_on_abg, levels = c(1, 0),
                       labels = c("Got ABG & Hypercapnia", "Got ABG & No hypercapnia")),
    hyper_vbg = factor(hypercap_on_vbg, levels = c(1, 0),
                       labels = c("Got VBG & Hypercapnia", "Got VBG & No hypercapnia"))
  )

# ABG cohort (has_abg == 1)
tbl2_abg <- subset_data %>%
  filter(has_abg == 1) %>%
  select(all_of(vars), hyper_abg) %>%
  gtsummary::tbl_summary(
```

Table 1

Variable	Everyone[1]	**Did not get ABG** N = 3,321[1]	**Did get ABG** N = 1,790[1]
Age (years)	58.6 ± 17.9; 0.0/5,111.0 missing (0.0%)	57.5 ± 18.3; 0.0/3,321.0 missing (0.0%)	60.6 ± 17.0; 0.0/1,790.0 missing (0.0
Current BMI kg/m2	31.4 ± 8.5; 2,862.0/5,111.0 missing (56.0%)	32.7 ± 8.8; 1,802.0/3,321.0 missing (54.3%)	28.6 ± 7.1; 1,060.0/1,790.0 missing (59
sex_label			
Female	2,540 (50%)	1,728 (52%)	812 (45%)
Male	2,571 (50%)	1,593 (48%)	978 (55%)
race_ethnicity_label			
White	3,116 (61%)	2,000 (60%)	1,116 (62%)
Black or African American	943 (18%)	648 (20%)	295 (16%)
Hispanic	350 (6.8%)	246 (7.4%)	104 (5.8%)
Asian	77 (1.5%)	48 (1.4%)	29 (1.6%)
American Indian	43 (0.8%)	22 (0.7%)	21 (1.2%)
Pacific Islander	6 (0.1%)	2 (<0.1%)	4 (0.2%)
Unknown	576 (11%)	355 (11%)	221 (12%)
location_label			
South	2,385 (47%)	1,403 (42%)	982 (55%)
Northeast	1,272 (25%)	906 (27%)	366 (20%)
Midwest	382 (7.5%)	235 (7.1%)	147 (8.2%)
West	1,072 (21%)	777 (23%)	295 (16%)
osa_label	868 (17%)	606 (18%)	262 (15%)
asthma_label	678 (13%)	480 (14%)	198 (11%)
copd_label	976 (19%)	610 (18%)	366 (20%)
chf_label	974 (19%)	584 (18%)	390 (22%)
nmd_label	205 (4.0%)	124 (3.7%)	81 (4.5%)
phtn_label	390 (7.6%)	239 (7.2%)	151 (8.4%)
ckd_label	966 (19%)	606 (18%)	360 (20%)
diabetes_label	1,465 (29%)	952 (29%)	513 (29%)
encounter_type_label			
Emergency	1,754 (34%)	1,485 (45%)	269 (15%)
Inpatient	3,357 (66%)	1,836 (55%)	1,521 (85%)
VBG PCO2	45.8 ± 13.5; 3,679.0/5,111.0 missing (72.0%)	45.0 ± 10.9; 2,420.0/3,321.0 missing (72.9%)	47.3 ± 17.0; 1,259.0/1,790.0 missing (70
Arterial PCO2	43.1 ± 18.5; 3,321.0/5,111.0 missing (65.0%)	NA ± NA; 3,321.0/3,321.0 missing (100.0%)	43.1 ± 18.5; 0.0/1,790.0 missing (0.0

[1]Mean ± SD; N Missing/No. obs. missing (% Missing); n (%)

```
    by = hyper_abg,
    type = list(sex_label ~ "categorical"),
    statistic = list(
      gtsummary::all_continuous()  ~ "{mean} ± {sd}; {N_miss}/{N_obs} missing ({p_miss}%)",
      gtsummary::all_categorical() ~ "{n} ({p}%)"
    ),
    digits   = list(gtsummary::all_continuous() ~ 1),
    missing  = "no"
  ) %>%
  gtsummary::modify_header(
    label  = "**Variable**",
    stat_1 = "**Got ABG & Hypercapnia**",
    stat_2 = "**Got ABG & No hypercapnia**"
  )

# VBG cohort (has_vbg == 1)
tbl2_vbg <- subset_data %>%
  filter(has_vbg == 1) %>%
  select(all_of(vars), hyper_vbg) %>%
  gtsummary::tbl_summary(
    by = hyper_vbg,
    type = list(sex_label ~ "categorical"),
    statistic = list(
      gtsummary::all_continuous()  ~ "{mean} ± {sd}; {N_miss}/{N_obs} missing ({p_miss}%)",
      gtsummary::all_categorical() ~ "{n} ({p}%)"
    ),
    digits   = list(gtsummary::all_continuous() ~ 1),
    missing  = "no"
  ) %>%
  gtsummary::modify_header(
    label  = "**Variable**",
    stat_1 = "**Got VBG & Hypercapnia**",
    stat_2 = "**Got VBG & No hypercapnia**"
  )

# Merge side-by-side (no spanners; 4 requested columns)
```

```r
table2 <- gtsummary::tbl_merge(
  tbls = list(tbl2_abg, tbl2_vbg),
  tab_spanner = c(NULL, NULL)
) %>%
  gtsummary::modify_caption("**Table 2. Baseline summary by hypercapnia within ABG and VBG cohorts**")

table2
```

Chunk table2-hypercapnia-cohorts runtime: 1.30 s

1.2.4 2.4 Generating Word Docs for New Table 1 and 2

```r
library(gtsummary)
library(flextable)
library(officer)

# gtsummary objects (example: table1, table2)
ft1 <- as_flex_table(tbl1)
ft2 <- as_flex_table(table2)

doc <- read_docx() %>%
  body_add_par("Table 1", style = "heading 1") %>%
  body_add_flextable(ft1) %>%
  body_add_par("Table 2", style = "heading 1") %>%
  body_add_flextable(ft2)

print(doc, target = "Tables.docx")
```

Chunk export-table1-table2-word runtime: 0.77 s

2 Unweighted Binary Logistic Regressions

Unweighted, Hypercapnia (binary yes/no) Simple (1 predictor) Regressions:

	Table 1		Table
Variable	**Got ABG & Hypercapnia**[1]	**Got ABG & No hypercapnia**[1]	**Got VBG & Hypercapnia**[1]
Age (years)	61.5 ± 16.2; 0.0/545.0 missing (0.0%)	60.2 ± 17.4; 0.0/1,245.0 missing (0.0%)	59.8 ± 17.2; 0.0/396.0 missing (0.0%)
Current BMI kg/m2	29.6 ± 7.8; 332.0/545.0 missing (60.9%)	28.2 ± 6.8; 728.0/1,245.0 missing (58.5%)	29.3 ± 8.2; 272.0/396.0 missing (68.7%)
sex_label			
Female	255 (47%)	557 (45%)	190 (48%)
Male	290 (53%)	688 (55%)	206 (52%)
race_ethnicity_label			
White	366 (67%)	750 (60%)	215 (54%)
Black or African American	88 (16%)	207 (17%)	86 (22%)
Hispanic	29 (5.3%)	75 (6.0%)	26 (6.6%)
Asian	4 (0.7%)	25 (2.0%)	8 (2.0%)
American Indian	4 (0.7%)	17 (1.4%)	4 (1.0%)
Pacific Islander	2 (0.4%)	2 (0.2%)	0 (0%)
Unknown	52 (9.5%)	169 (14%)	57 (14%)
location_label			
South	298 (55%)	684 (55%)	140 (35%)
Northeast	139 (26%)	227 (18%)	167 (42%)
Midwest	50 (9.2%)	97 (7.8%)	44 (11%)
West	58 (11%)	237 (19%)	45 (11%)
osa_label	114 (21%)	148 (12%)	80 (20%)
asthma_label	78 (14%)	120 (9.6%)	56 (14%)
copd_label	181 (33%)	185 (15%)	114 (29%)
chf_label	147 (27%)	243 (20%)	108 (27%)
nmd_label	20 (3.7%)	61 (4.9%)	15 (3.8%)
phtn_label	61 (11%)	90 (7.2%)	39 (9.8%)
ckd_label	109 (20%)	251 (20%)	82 (21%)
diabetes_label	162 (30%)	351 (28%)	125 (32%)
encounter_type_label			
Emergency	83 (15%)	186 (15%)	130 (33%)
Inpatient	462 (85%)	1,059 (85%)	266 (67%)
VBG PCO2	59.0 ± 21.3; 377.0/545.0 missing (69.2%)	41.9 ± 10.9; 882.0/1,245.0 missing (70.8%)	61.5 ± 14.4; 0.0/396.0 missing (0.0%)
Arterial PCO2	60.7 ± 24.4; 0.0/545.0 missing (0.0%)	35.4 ± 6.1; 0.0/1,245.0 missing (0.0%)	54.6 ± 22.0; 231.0/396.0 missing (58.3%)

[1]Mean ± SD; N Missing/No. obs. missing (% Missing); n (%)

Unweighted, ABG Group: hypercapnia treated as a binary (yes/no) predictor

2.0.1 3.1 ABG: Binary hypercapnia models

```
logit_intubated_abg <- glm(imv_proc ~ hypercap_on_abg, data = subset_data, family = binomial)
summary(logit_intubated_abg)
```

```
Call:
glm(formula = imv_proc ~ hypercap_on_abg, family = binomial,
    data = subset_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -2.32422    0.05194  -44.75   <2e-16 ***
hypercap_on_abg  1.38339    0.10856   12.74   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3532.8  on 5110  degrees of freedom
Residual deviance: 3391.6  on 5109  degrees of freedom
AIC: 3395.6

Number of Fisher Scoring iterations: 5
```

```
tidy(logit_intubated_abg,
     exponentiate = TRUE,   # turns log-odds → OR
     conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.0978601	0.0519369	-44.75075	0	0.0882607	0.1081953
hypercap_on_abg	3.9884107	0.1085560	12.74358	0	3.2182933	4.9265465

```r
logit_niv_abg <- glm(niv_proc ~ hypercap_on_abg, data = subset_data, family = binomial)
summary(logit_niv_abg)
```

```
Call:
glm(formula = niv_proc ~ hypercap_on_abg, family = binomial,
    data = subset_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -2.78687    0.06329 -44.031  < 2e-16 ***
hypercap_on_abg  1.04144    0.13604   7.655 1.93e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2531.4  on 5110  degrees of freedom
Residual deviance: 2481.2  on 5109  degrees of freedom
AIC: 2485.2

Number of Fisher Scoring iterations: 5
```

```r
tidy(logit_niv_abg,
     exponentiate = TRUE,   # turns log-odds → OR
     conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.0616136	0.0632928	-44.031437	0	0.0542991	0.0695965

term	estimate	std.error	statistic	p.value	conf.low	conf.high
hypercap_on_abg	2.8332872	0.1360398	7.655388	0	2.1594194	3.6831150

```r
logit_death_abg <- glm(death_60d ~ hypercap_on_abg, data = subset_data, family = binomial)
summary(logit_death_abg)
```

```
Call:
glm(formula = death_60d ~ hypercap_on_abg, family = binomial,
    data = subset_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -2.25858    0.05056 -44.668  < 2e-16 ***
hypercap_on_abg  0.88371    0.11810   7.483 7.26e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3456.7  on 5110  degrees of freedom
Residual deviance: 3407.3  on 5109  degrees of freedom
AIC: 3411.3

Number of Fisher Scoring iterations: 5
```

```r
tidy(logit_death_abg,
     exponentiate = TRUE,   # turns log-odds → OR
     conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.1044993	0.0505639	-44.667712	0	0.0945116	0.1152356
hypercap_on_abg	2.4198595	0.1180952	7.483028	0	1.9131029	3.0405992

```r
logit_icd_abg <- glm(hypercap_resp_failure ~ hypercap_on_abg, data = subset_data, family = binomial)
summary(logit_icd_abg)
```

```
Call:
glm(formula = hypercap_resp_failure ~ hypercap_on_abg, family = binomial,
    data = subset_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      -3.3687     0.0825  -40.83   <2e-16 ***
hypercap_on_abg   2.2676     0.1288   17.60   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2216.1  on 5110  degrees of freedom
Residual deviance: 1945.6  on 5109  degrees of freedom
AIC: 1949.6

Number of Fisher Scoring iterations: 6
```

```r
tidy(logit_icd_abg,
     exponentiate = TRUE,   # turns log-odds → OR
     conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.0344359	0.0824954	-40.83447	0	0.0291724	0.0403191
hypercap_on_abg	9.6561575	0.1288543	17.59814	0	7.4981130	12.4306031

Chunk abg-binary-logit-models runtime: 0.26 s

Display the regression coefficients for the binary (hypercapnia yes/no) predictor logistic regressions

	Intubated	NIV	Death	ICD Hyper
hypercap_on_abg	3.99 (3.22, 4.93)	2.83 (2.16, 3.68)	2.42 (1.91, 3.04)	9.66 (7.50, 12.43)

```
modelsummary(
  list("Intubated" = logit_intubated_abg,
       "NIV"       = logit_niv_abg,
       "Death"     = logit_death_abg,
       "ICD Hyper" = logit_icd_abg),
  exponentiate = TRUE,
  conf_level   = 0.95,
  estimate     = "{estimate}",
  statistic    = "({conf.low}, {conf.high})",
  coef_omit    = "(Intercept)",
  gof_omit     = ".*",                      # drop all goodness-of-fit rows
  fmt          = 2,                         # 2 decimal places everywhere
  output       = "gt"
) |>
  gt_pdf(title = "Odds Ratios for ABG Hypercapnia (>45 mmHg)'s association with...")
```

Chunk abg-binary-or-table runtime: 0.72 s

Unweighted VBG Group

2.0.2 3.2 VBG: Binary hypercapnia models

```
logit_intubated_vbg <- glm(imv_proc ~ hypercap_on_vbg, data = subset_data, family = binomial)
summary(logit_intubated_vbg)
```

```
Call:
glm(formula = imv_proc ~ hypercap_on_vbg, family = binomial,
    data = subset_data)
```

```
Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -2.15209    0.04768 -45.138  < 2e-16 ***
hypercap_on_vbg  0.59624    0.14080   4.235 2.29e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3532.8  on 5110  degrees of freedom
Residual deviance: 3516.7  on 5109  degrees of freedom
AIC: 3520.7

Number of Fisher Scoring iterations: 4
```

```r
tidy(logit_intubated_vbg,
     exponentiate = TRUE,   # turns log-odds → OR
     conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.1162405	0.0476777	-45.138392	0.00e+00	0.1057455	0.1274825
hypercap_on_vbg	1.8152805	0.1407978	4.234726	2.29e-05	1.3682145	2.3779294

```r
logit_niv_vbg <- glm(niv_proc ~ hypercap_on_vbg, data = subset_data, family = binomial)
summary(logit_niv_vbg)
```

```
Call:
glm(formula = niv_proc ~ hypercap_on_vbg, family = binomial,
    data = subset_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -2.70331    0.06004 -45.026  < 2e-16 ***
```

```
hypercap_on_vbg  0.76889    0.16277   4.724 2.32e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2531.4  on 5110  degrees of freedom
Residual deviance: 2512.1  on 5109  degrees of freedom
AIC: 2516.1

Number of Fisher Scoring iterations: 5
```

```r
tidy(logit_niv_vbg,
     exponentiate = TRUE,   # turns log-odds → OR
     conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.0669835	0.0600386	-45.026216	0.0e+00	0.0594246	0.0751992
hypercap_on_vbg	2.1573778	0.1627722	4.723738	2.3e-06	1.5526989	2.9427376

```r
logit_death_vbg <- glm(death_60d ~ hypercap_on_vbg, data = subset_data, family = binomial)
summary(logit_death_vbg)
```

```
Call:
glm(formula = death_60d ~ hypercap_on_vbg, family = binomial,
    data = subset_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -2.17272    0.04807 -45.200  < 2e-16 ***
hypercap_on_vbg  0.44996    0.14817   3.037  0.00239 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3456.7  on 5110  degrees of freedom
Residual deviance: 3448.3  on 5109  degrees of freedom
AIC: 3452.3

Number of Fisher Scoring iterations: 4
```

```r
tidy(logit_death_vbg,
     exponentiate = TRUE,   # turns log-odds → OR
     conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.1138672	0.0480690	-45.200090	0.0000000	0.1035043	0.1249726
hypercap_on_vbg	1.5682424	0.1481671	3.036812	0.0023909	1.1633628	2.0815317

```r
logit_icd_vbg <- glm(hypercap_resp_failure ~ hypercap_on_vbg, data = subset_data, family = binomial)
summary(logit_icd_vbg)
```

```
Call:
glm(formula = hypercap_resp_failure ~ hypercap_on_vbg, family = binomial,
    data = subset_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -3.18137    0.07443  -42.74   <2e-16 ***
hypercap_on_vbg  2.09618    0.13754   15.24   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 2216.1  on 5110  degrees of freedom
Residual deviance: 2027.5  on 5109  degrees of freedom
AIC: 2031.5

Number of Fisher Scoring iterations: 6
```

```
tidy(logit_icd_vbg,
     exponentiate = TRUE,   # turns log-odds → OR
      conf.int     = TRUE)   # adds 95 % CI
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	0.0415286	0.0744315	-42.74230	0	0.0357713	0.0478982
hypercap_on_vbg	8.1350633	0.1375443	15.24007	0	6.1984088	10.6330855

Chunk vbg-binary-logit-models runtime: 0.26 s

2.0.3 3.3 Display model coefficients for binary hypercapnia on VBG logistic regression

```
modelsummary(
  list("Intubated" = logit_intubated_vbg,
       "NIV"       = logit_niv_vbg,
       "Death"     = logit_death_vbg,
       "ICD Hyper" = logit_icd_vbg),
  exponentiate = TRUE,
  conf_level   = 0.95,
  estimate     = "{estimate}",
  statistic    = "({conf.low}, {conf.high})",
  coef_omit    = "(Intercept)",
  gof_omit     = ".*",                    # drop all goodness-of-fit rows
  fmt          = 2,                       # 2 decimal places everywhere
  output       = "gt"
) |>
  gt_pdf(title = "Odds ratios for VBG hypercapnia (>45 mmHg) on outcomes")
```

	Intubated	NIV	Death	ICD Hyper
hypercap_on_vbg	1.82 (1.37, 2.38)	2.16 (1.55, 2.94)	1.57 (1.16, 2.08)	8.14 (6.20, 10.63)

Chunk vbg-binary-or-table runtime: 0.50 s

2.1 3) Three-level PCO2 categories (unweighted)

Now doing 3 groups instead of binary (above, normal and below)

```
subset_data <- subset_data %>%
  mutate(
    pco2_cat_abg = case_when(
      !is.na(paco2) & paco2 < 35 ~ "Hypocapnia",
      !is.na(paco2) & paco2 > 45 ~ "Hypercapnia",
      !is.na(paco2)              ~ "Eucapnia"
    ),
    pco2_cat_vbg = case_when(
      !is.na(vbg_co2) & vbg_co2 < 40 ~ "Hypocapnia",
      !is.na(vbg_co2) & vbg_co2 > 50 ~ "Hypercapnia",
      !is.na(vbg_co2)                ~ "Eucapnia"
    )
  ) %>%
  mutate(
    across(c(pco2_cat_abg, pco2_cat_vbg),
           ~factor(.x, levels = c("Eucapnia", "Hypocapnia", "Hypercapnia")))
  )

library(broom)
library(dplyr)

run_logit <- function(data, outcome, exposure, group_name) {
  f <- as.formula(paste(outcome, "~", exposure))
  glm(f, data = data, family = binomial) %>%
    tidy(exponentiate = TRUE, conf.int = TRUE) %>%
    filter(term != "(Intercept)") %>%
```

```
    mutate(
      outcome = outcome,
      group   = group_name
    )
}

outcomes <- c("imv_proc", "niv_proc", "death_60d", "hypercap_resp_failure")

results <- bind_rows(
  lapply(outcomes, function(o) run_logit(subset_data, o, "pco2_cat_abg", "ABG")),
  lapply(outcomes, function(o) run_logit(subset_data, o, "pco2_cat_vbg", "VBG"))
)

combined_or_df <- results %>%
  mutate(
    exposure = recode(term,
                      "pco2_cat_abgHypocapnia"   = "Hypocapnia",
                      "pco2_cat_abgHypercapnia"  = "Hypercapnia",
                      "pco2_cat_vbgHypocapnia"   = "Hypocapnia",
                      "pco2_cat_vbgHypercapnia"  = "Hypercapnia"),
    outcome = recode(outcome,
                     imv_proc = "Intubation",
                     niv_proc = "NIV",
                     death_60d = "Death (60d)",
                     hypercap_resp_failure = "Hypercapnic RF")
  ) %>%
  select(outcome, group, exposure, estimate, conf.low, conf.high)
```

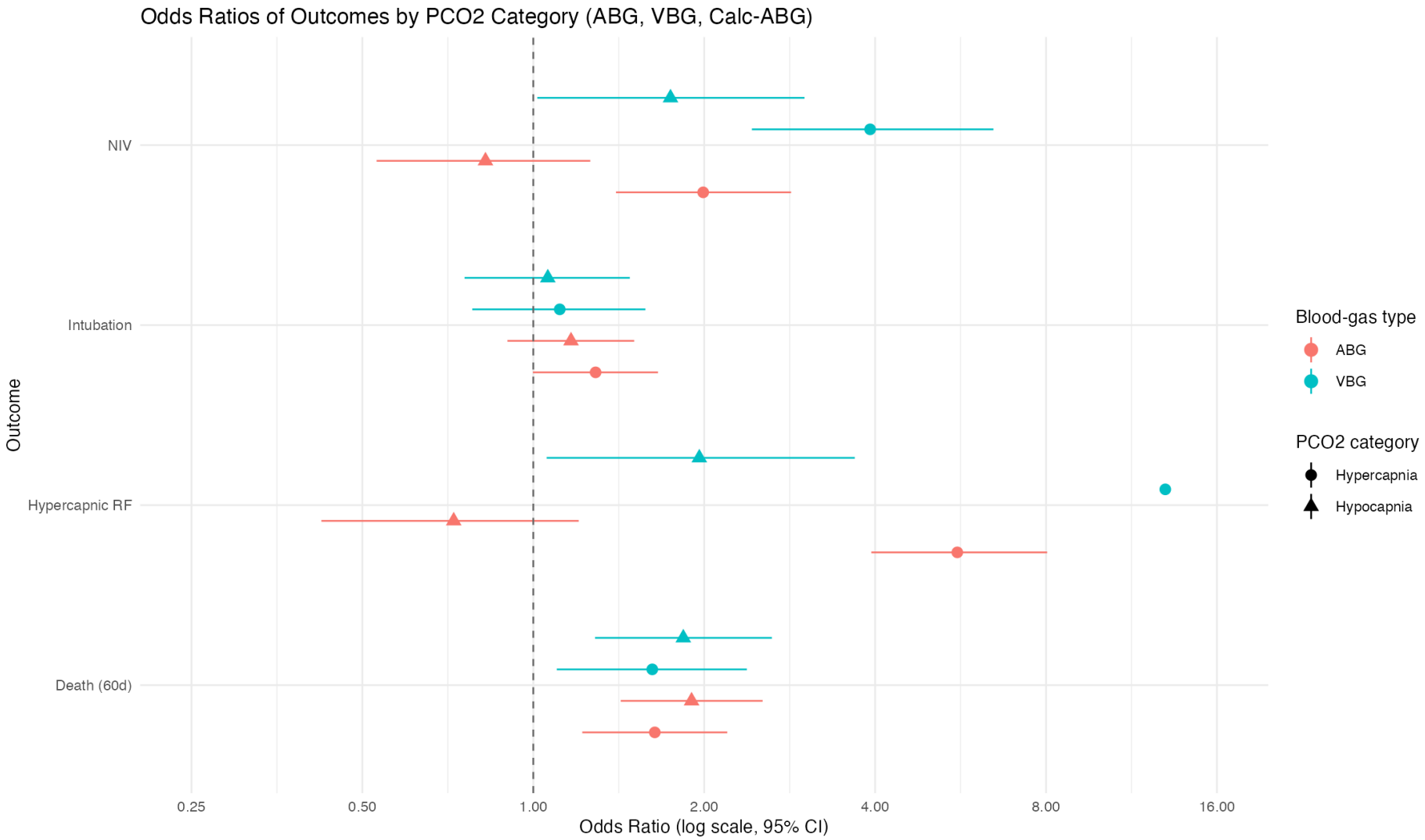
Chunk or-data-three-level-unweighted runtime: 0.45 s

```
library(scales)

combined_or_df$group <- factor(
  combined_or_df$group,
  levels = c("ABG", "VBG")
)
```

```r
ggplot(
  combined_or_df,
  aes(
    x      = outcome,
    y      = estimate,
    ymin   = conf.low,
    ymax   = conf.high,
    color  = group,
    shape  = exposure
  )
) +
  geom_pointrange(
    position = position_dodge(width = 0.7),
    size     = 0.6
  ) +
  geom_hline(yintercept = 1, linetype = "dashed", colour = "grey40") +
  scale_y_log10(
    breaks = c(0.25, 0.5, 1, 2, 4, 8, 16),
    limits = c(0.25, 16),
    labels = number_format(accuracy = 0.01)
  ) +
  coord_flip() +
  labs(
    title  = "Odds Ratios of Outcomes by PCO2 Category (ABG, VBG, Calc-ABG)",
    x      = "Outcome",
    y      = "Odds Ratio (log scale, 95% CI)",
    color  = "Blood-gas type",
    shape  = "PCO2 category",
    caption = paste(
      "Odds ratios are computed within each blood-gas cohort.",
      "Reference = patients in the normal PCO2 range.",
      "Hypocapnia: <35 mmHg (ABG/Calc) or <40 mmHg (VBG); Hypercapnia: >45 mmHg (ABG/Calc) or >50 mmHg (VBG).",
      "Because the underlying cohorts differ (ABG, VBG), denominators are not identical across groups.",
      sep = "\n"
    )
```

```
  ) +
  theme_minimal(base_size = 10) +
  theme(plot.caption = element_text(hjust = 0))
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_segment()`).
```



Odds ratios are computed within each blood-gas cohort.
Reference = patients in the normal PCO2 range.
Hypocapnia: <35 mmHg (ABG/Calc) or <40 mmHg (VBG); Hypercapnia: >45 mmHg (ABG/Calc) or >50 mmHg (VBG).
Because the underlying cohorts differ (ABG, VBG), denominators are not identical across groups.

Chunk or-plot-three-level-unweighted runtime: 0.27 s

2.2 4) Restricted cubic spline regressions (unweighted)

```
# ABG spline dataset
subset_data_abg <- subset_data %>%
  select(paco2, imv_proc, niv_proc, death_60d, hypercap_resp_failure) %>%
  filter(!is.na(paco2))

dd_abg <- datadist(subset_data_abg)
options(datadist = "dd_abg")
```

Chunk rcs-abg-data-prep runtime: 0.00 s

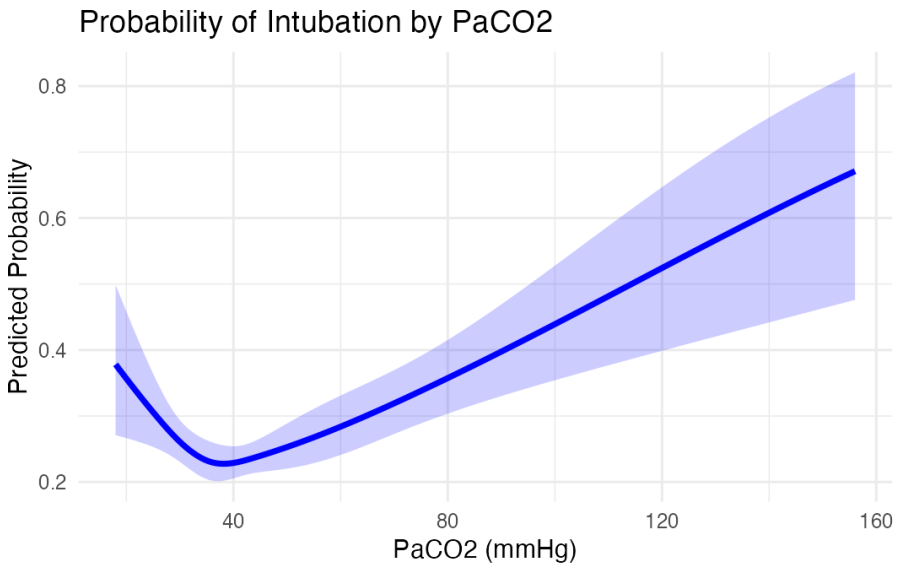
2.2.1 4.1 Unweighted, Restricted Cubic Spline Regression - ABG by PaCO2

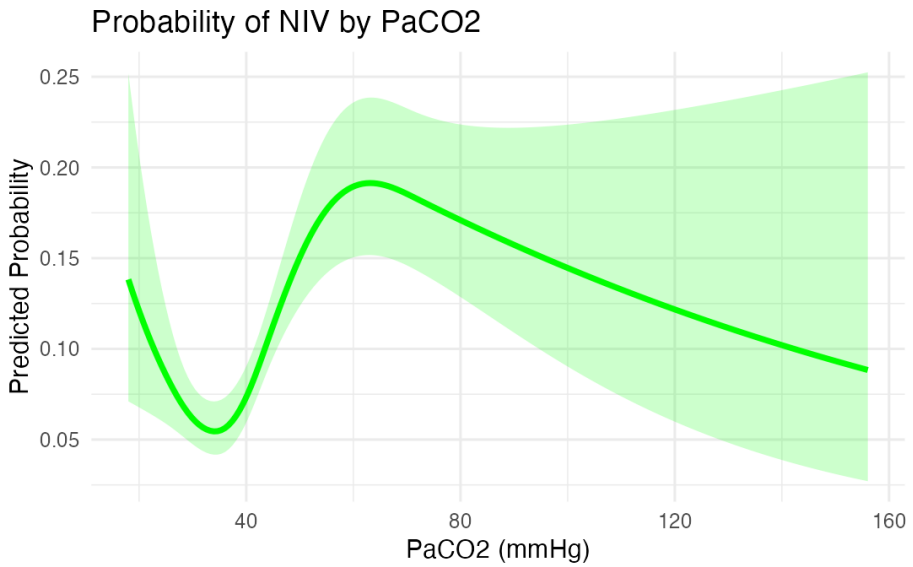
```
fit_imv <- lrm(imv_proc ~ rcs(paco2, 4), data = subset_data_abg)
pred_imv <- as.data.frame(Predict(fit_imv, paco2, fun = plogis))

plot_imv <- ggplot(pred_imv, aes(x = paco2, y = yhat)) +
  geom_line(color = "blue", size = 1.2) +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "blue", alpha = 0.2) +
labs(title = "Probability of Intubation by PaCO2",
     x = "PaCO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()
```

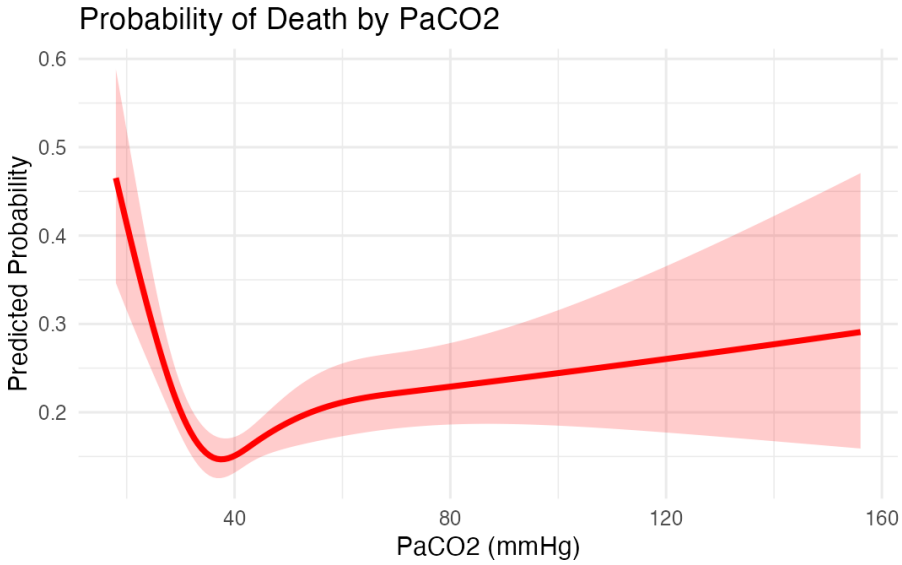
```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```

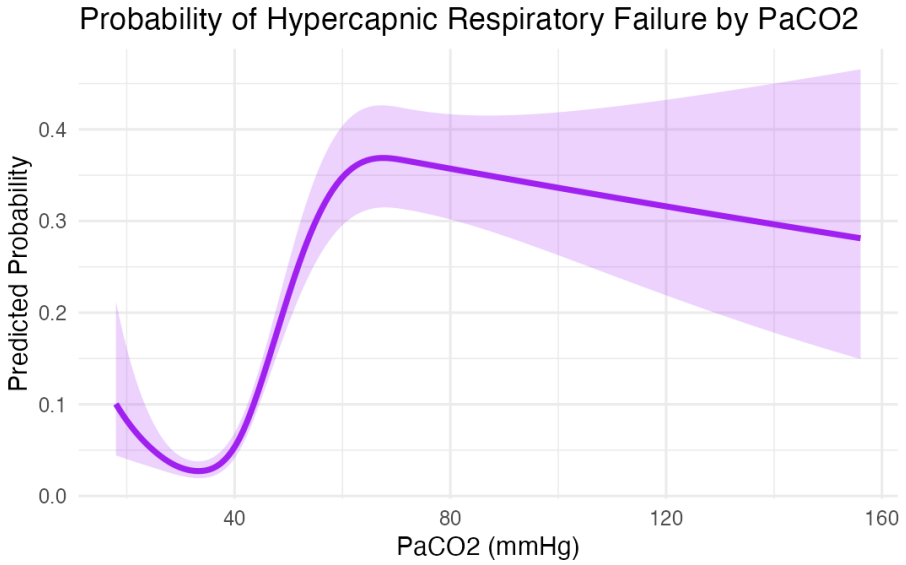
```
fit_niv <- lrm(niv_proc ~ rcs(paco2, 4), data = subset_data_abg)
pred_niv <- as.data.frame(Predict(fit_niv, paco2, fun = plogis))

plot_niv <- ggplot(pred_niv, aes(x = paco2, y = yhat)) +
  geom_line(color = "green", size = 1.2) +
```

```
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "green", alpha = 0.2) +
labs(title = "Probability of NIV by PaCO2",
     x = "PaCO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()

fit_death <- lrm(death_60d ~ rcs(paco2, 4), data = subset_data_abg)
pred_death <- as.data.frame(Predict(fit_death, paco2, fun = plogis))

plot_death <- ggplot(pred_death, aes(x = paco2, y = yhat)) +
  geom_line(color = "red", size = 1.2) +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "red", alpha = 0.2) +
labs(title = "Probability of Death by PaCO2",
     x = "PaCO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()

fit_hcrf <- lrm(hypercap_resp_failure ~ rcs(paco2, 4), data = subset_data_abg)
pred_hcrf <- as.data.frame(Predict(fit_hcrf, paco2, fun = plogis))

plot_hcrf <- ggplot(pred_hcrf, aes(x = paco2, y = yhat)) +
  geom_line(color = "purple", size = 1.2) +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "purple", alpha = 0.2) +
labs(title = "Probability of Hypercapnic Respiratory Failure by PaCO2",
     x = "PaCO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()

(plot_imv | plot_niv) / (plot_death | plot_hcrf)
```









Chunk rcs-abg-unweighted-models runtime: 0.48 s

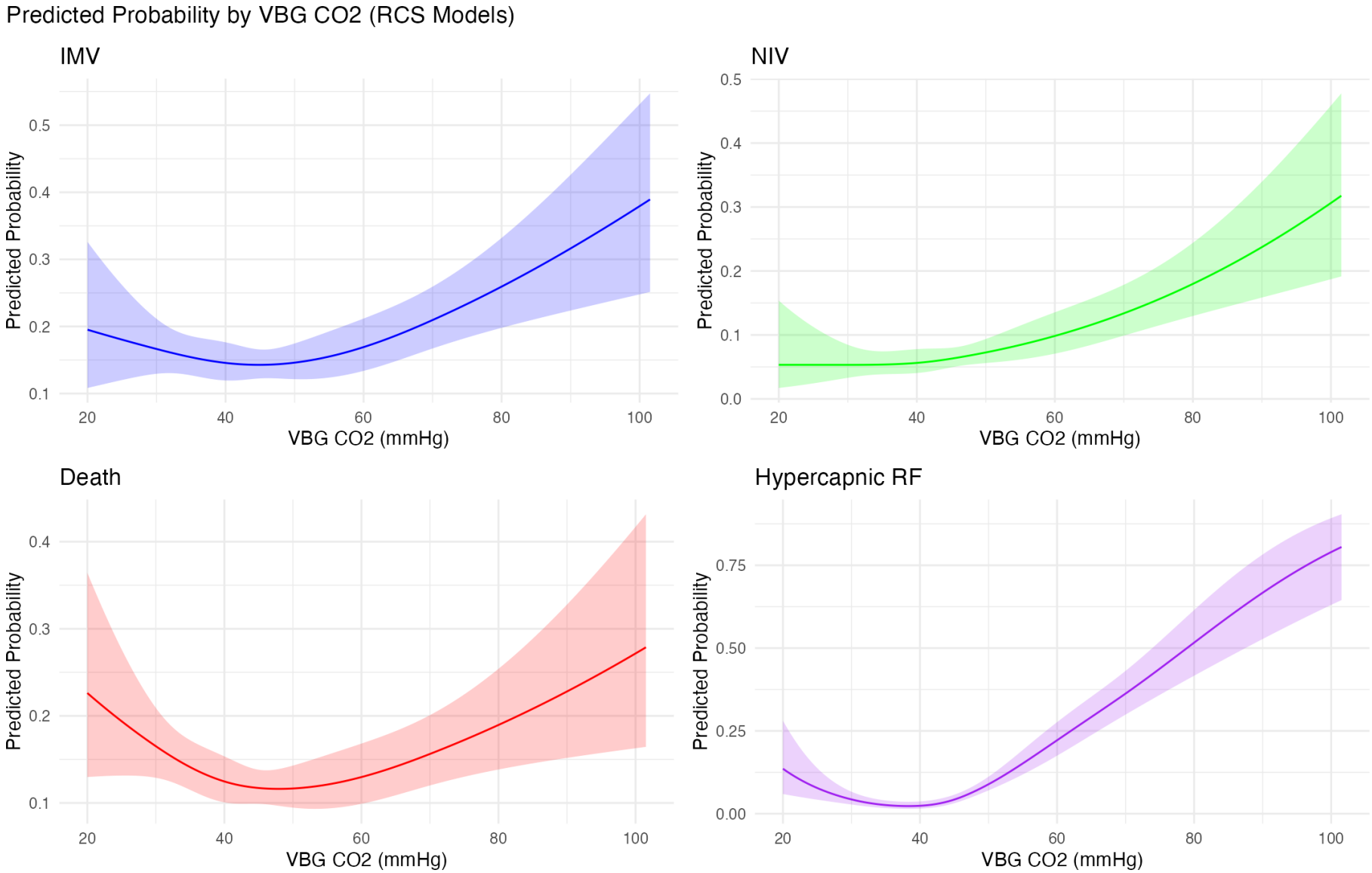
2.2.2 4.2 Unweighted, Restricted Cubic Spline - VBG

```
# --- VBG dataset ---
subset_data_vbg <- subset_data %>%
  dplyr::select(vbg_co2, imv_proc, niv_proc, death_60d, hypercap_resp_failure) %>%
  dplyr::filter(!is.na(vbg_co2) & complete.cases(.))

dd_vbg <- datadist(subset_data_vbg)   # create datadist for VBG
# activate when doing VBG models:
options(datadist = "dd_vbg")
```

Chunk rcs-vbg-data-prep runtime: 0.00 s

```
subset_data_vbg <- subset_data %>%
  select(vbg_co2, imv_proc, niv_proc, death_60d, hypercap_resp_failure) %>%
  filter(!is.na(vbg_co2) & complete.cases(.))

dd <- datadist(subset_data_vbg)
options(datadist = "dd")

fit_imv_vbg <- lrm(imv_proc ~ rcs(vbg_co2, 4), data = subset_data_vbg)
fit_niv_vbg <- lrm(niv_proc ~ rcs(vbg_co2, 4), data = subset_data_vbg)
fit_death_vbg <- lrm(death_60d ~ rcs(vbg_co2, 4), data = subset_data_vbg)
fit_hcrf_vbg <- lrm(hypercap_resp_failure ~ rcs(vbg_co2, 4), data = subset_data_vbg)

pred_imv_vbg <- as.data.frame(Predict(fit_imv_vbg, vbg_co2, fun = plogis))
pred_niv_vbg <- as.data.frame(Predict(fit_niv_vbg, vbg_co2, fun = plogis))
pred_death_vbg <- as.data.frame(Predict(fit_death_vbg, vbg_co2, fun = plogis))
pred_hcrf_vbg <- as.data.frame(Predict(fit_hcrf_vbg, vbg_co2, fun = plogis))

plot_imv_vbg <- ggplot(pred_imv_vbg, aes(x = vbg_co2, y = yhat)) +
  geom_line(color = "blue") +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "blue", alpha = 0.2) +
  labs(title = "IMV", x = "VBG CO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()
```

```r
plot_niv_vbg <- ggplot(pred_niv_vbg, aes(x = vbg_co2, y = yhat)) +
  geom_line(color = "green") +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "green", alpha = 0.2) +
  labs(title = "NIV", x = "VBG CO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()

plot_death_vbg <- ggplot(pred_death_vbg, aes(x = vbg_co2, y = yhat)) +
  geom_line(color = "red") +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "red", alpha = 0.2) +
  labs(title = "Death", x = "VBG CO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()

plot_hcrf_vbg <- ggplot(pred_hcrf_vbg, aes(x = vbg_co2, y = yhat)) +
  geom_line(color = "purple") +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "purple", alpha = 0.2) +
  labs(title = "Hypercapnic RF", x = "VBG CO2 (mmHg)", y = "Predicted Probability") +
  theme_minimal()

((plot_imv_vbg | plot_niv_vbg) /
 (plot_death_vbg | plot_hcrf_vbg)) +
 plot_annotation(title = "Predicted Probability by VBG CO2 (RCS Models)")
```



Chunk rcs-vbg-unweighted-models runtime: 0.39 s

3 Inverse Propensity Weighting

IPW done using Gradient Boosting Methods (GBM) - a type of decision-tree based machine learning. "***Random forests and GBM are designed to automatically include relevant interactions for variables included in the model.*** As such, using a GBM to estimate the PS model, can reduce model misspecification, since ***the analyst is not required to identify relevant interactions or nonlinearities.***" from this citation: PMID: 39947224https://pmc.ncbi.nlm.nih.gov/articles/PMC11825193/

Current propensity score uses **age_at_encounter + sex + race_ethnicity** (remember - have to specify to use this as a factor variable) **+ curr_bmi + copd + asthma + osa + chf + acute_nmd + phtn + location (as a factor variable)**

Note: for all these, I suggested new GBM adjustments that accomplish the following:

1. Smaller GBM & stopping rule → faster fit, avoids over-fitting, lighter tails (which lead to extreme weights that are problematic).
2. bal.tab() documents balance; aim is to adjust spec until standard mean difference (SMD) < 0.1.
3. Weight stabilization (divide by mean) mitigates a few huge weights. I also winsorized, which is a way to avoid very extreme weights (ie you set <1st percentile to the 1st percentile value, and >99th percentile to 99th percentile.
4. Uses robust variance estimation (e.g. allows the variances to change by PaCO2) for IP-weighted GLM; works with splines via rcs(). This is a bit nuanced but I think good to change even though it adds complexity
5. Deterministic seed ensures result replication.

3.0.1 5.1 ABG IPW weighting and diagnostics

```
subset_data$encounter_type <- factor(subset_data$encounter_type,
                                     levels = c(2, 3),
                                     labels = c("Emergency", "Inpatient"))
```

Chunk encode-encounter-type runtime: 0.00 s

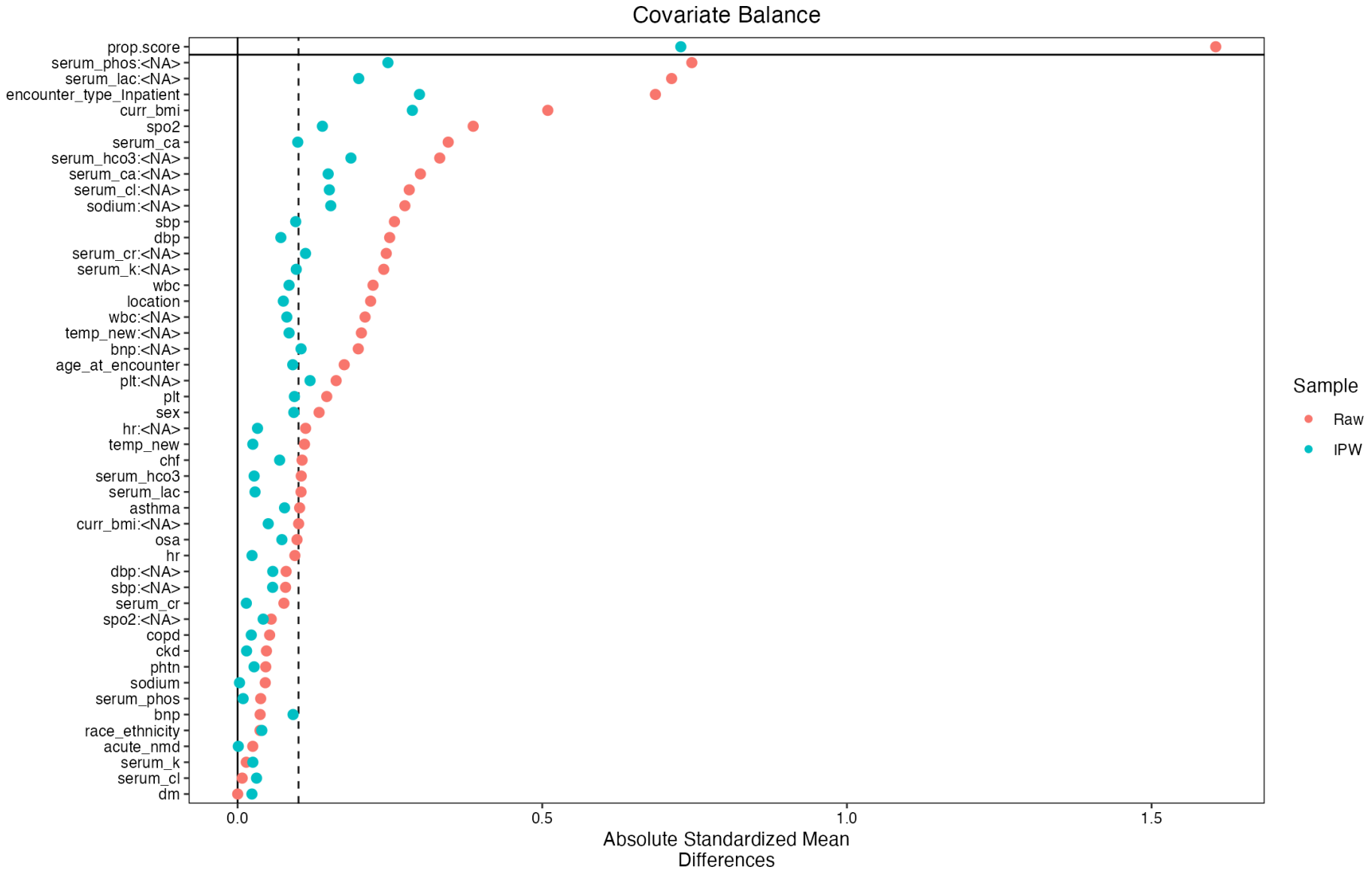
**Removed lactate from weights, decreased n.trees, increased bagging

```r
#   1. fit GBM propensity model, ABG
set.seed(42)

weight_model <- do.call(
  weightit,
  c(
    list(
      formula_abg,
      data        = subset_data,
      method      = "gbm",
      estimand    = "ATE",
      missing    = "ind",
      include.obj = TRUE        # ← REQUIRED for importance/SHAP
    ),
    gbm_params
  )
)
w_abg <- weight_model  # Canonical alias so later code can use `w_abg`

#   2. Winsorise / stabilise weights (two-sided)
w <- weight_model$weights            # original GBM weights
w <- w / mean(w)                     # stabilise
cut <- quantile(w, c(0.01, 0.99), na.rm = TRUE)
w   <- pmin(pmax(w, cut[1]), cut[2]) # two-tail Winsorisation
w <- w / mean(w)                     # re-stabilise so E[w]=1

# overwrite inside the object and attach to data
weight_model$weights <- w
subset_data$w_abg    <- w

#   3. balance diagnostics (only raw vs. IPW)
bal  <- bal.tab(
  weight_model,
  un = TRUE,
  m.threshold = 0.1,
  binary = "std",
```

```
  s.d.denom = "pooled"
)
```

```
Warning: Missing values exist in the covariates. Displayed values omit these
observations.
```

```
love.plot(
  bal,
  stats        = "m",          # standardized mean differences only
  abs          = TRUE,
  var.order    = "unadjusted",
  sample.names = c("Raw", "IPW")
)
```




```r
#   4. survey design with the same weights
design <- svydesign(ids = ~1, weights = ~w_abg, data = subset_data)

#   5. outcome models (examples)
```

```r
fit_niv   <- svyglm(niv_proc   ~ has_abg, design = design, family = quasibinomial())
fit_imv   <- svyglm(imv_proc   ~ has_abg, design = design, family = quasibinomial())
fit_death <- svyglm(death_60d       ~ has_abg, design = design, family = quasibinomial())
fit_icd   <- svyglm(hypercap_resp_failure ~ has_abg, design = design, family = quasibinomial())

# quick effect estimates
lapply(list(IMV = fit_imv, NIV = fit_niv, Death = fit_death, ICD = fit_icd), function(m) {
  c(OR  = exp(coef(m)[2]),
    LCL = exp(confint(m)[2,1]),
    UCL = exp(confint(m)[2,2]))
})
```

```
$IMV
OR.has_abg        LCL        UCL
  6.953209   5.383502   8.980607

$NIV
OR.has_abg        LCL        UCL
  1.733444   1.358505   2.211864

$Death
OR.has_abg        LCL        UCL
  2.340690   1.914698   2.861457

$ICD
OR.has_abg        LCL        UCL
  3.186309   2.390923   4.246297
```

Chunk ipw-abg-weighting runtime: 5.87 s

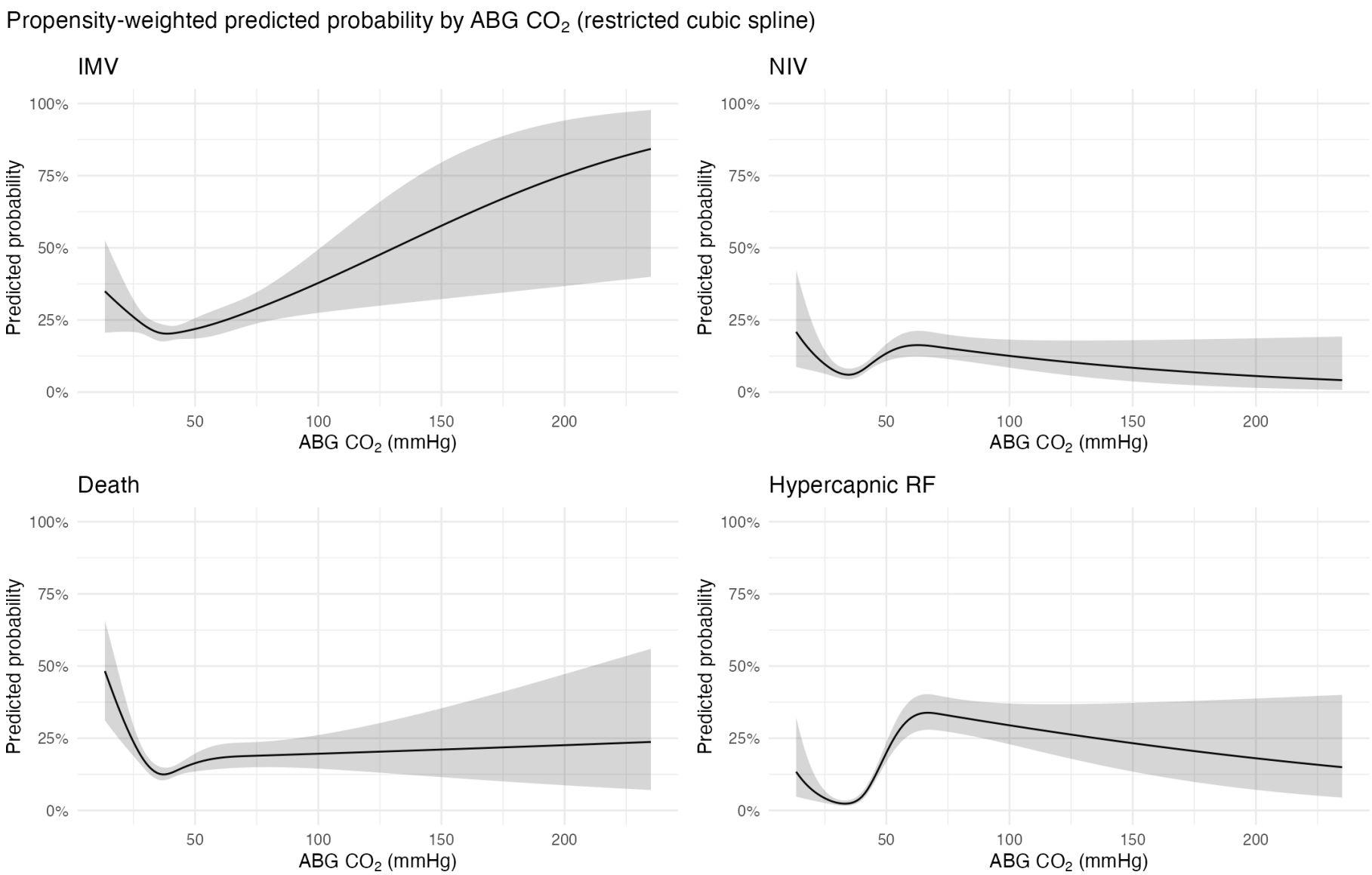
Inverse Propensity-Weighted Logistic Regressions with CO2 predictor represented as a restricted cubic spline.

3.0.2 5.2 ABG IPW spline models

```
# set.seed(42)  # reproducible GBM fit
#
# #   1. inverse-probability weights for receiving an ABG
#
# # done in the last block, so not needed
#

#   2. analysis sample: rows with a measured PaCO2
subset_data_abg <- subset_data %>%
  filter(!is.na(paco2)) %>%                    # implies has_abg == 1
  select(paco2, imv_proc, niv_proc, death_60d,
         hypercap_resp_failure, w_abg) %>%
  filter(complete.cases(.))


#   3. weighted logistic spline models with robust SEs
dd <- datadist(subset_data_abg); options(datadist = "dd")

fitfun <- function(formula)
  svyglm(
    formula,
    design = svydesign(ids = ~1, weights = ~w_abg, data = subset_data_abg),
    family = quasibinomial()
  )

fit_imv_abg   <- fitfun(imv_proc              ~ rcs(paco2, 4))
fit_niv_abg   <- fitfun(niv_proc              ~ rcs(paco2, 4))
fit_death_abg <- fitfun(death_60d                  ~ rcs(paco2, 4))
fit_hcrf_abg  <- fitfun(hypercap_resp_failure ~ rcs(paco2, 4))

#   4. prediction helper
mkpred <- function(fit, data_ref) {
  # 1. Grid of PaCO2 values
```

```r
  newd <- data.frame(
    paco2 = seq(min(data_ref$paco2, na.rm = TRUE),
                max(data_ref$paco2, na.rm = TRUE),
                length.out = 200)
  )

  # 2. Design (model) matrix for the new data
  mm <- model.matrix(delete.response(terms(fit)),  # drop outcome
                     data = newd)

  # 3. Linear predictor and its standard error
  eta  <- mm %*% coef(fit)                         #  'x
  vcov <- vcov(fit)                                # robust VCOV from svyglm
  se   <- sqrt(rowSums((mm %*% vcov) * mm))        # √diag(X Σ X )

  # 4. Transform to probability scale
  transform(
    newd,
    yhat  = plogis(eta),
    lower = plogis(eta - 1.96 * se),
    upper = plogis(eta + 1.96 * se)
  )
}

pred_imv_abg   <- mkpred(fit_imv_abg,   subset_data_abg)
pred_niv_abg   <- mkpred(fit_niv_abg,   subset_data_abg)
pred_death_abg <- mkpred(fit_death_abg, subset_data_abg)
pred_hcrf_abg  <- mkpred(fit_hcrf_abg,  subset_data_abg)

#   5. plotting
xlab <- expression(paste("ABG CO"[2], " (mmHg)"))

plt <- function(dat, title)
  ggplot(dat, aes(paco2, yhat)) +
    geom_line() +
    geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2) +
```

```
    scale_y_continuous(limits = c(0, 1), labels = percent_format(accuracy = 1)) +
    labs(title = title, x = xlab, y = "Predicted probability") +
    theme_minimal()

(patchwork::wrap_plots(
    plt(pred_imv_abg,   "IMV"),
    plt(pred_niv_abg,   "NIV"),
    plt(pred_death_abg, "Death"),
    plt(pred_hcrf_abg,  "Hypercapnic RF"),
    ncol = 2
  )
) +
  plot_annotation(
    title = expression(
      paste("Propensity-weighted predicted probability by ABG CO"[2],
            " (restricted cubic spline)")
    )
)
```



Chunk ipw-abg-rcs-models runtime: 0.41 s

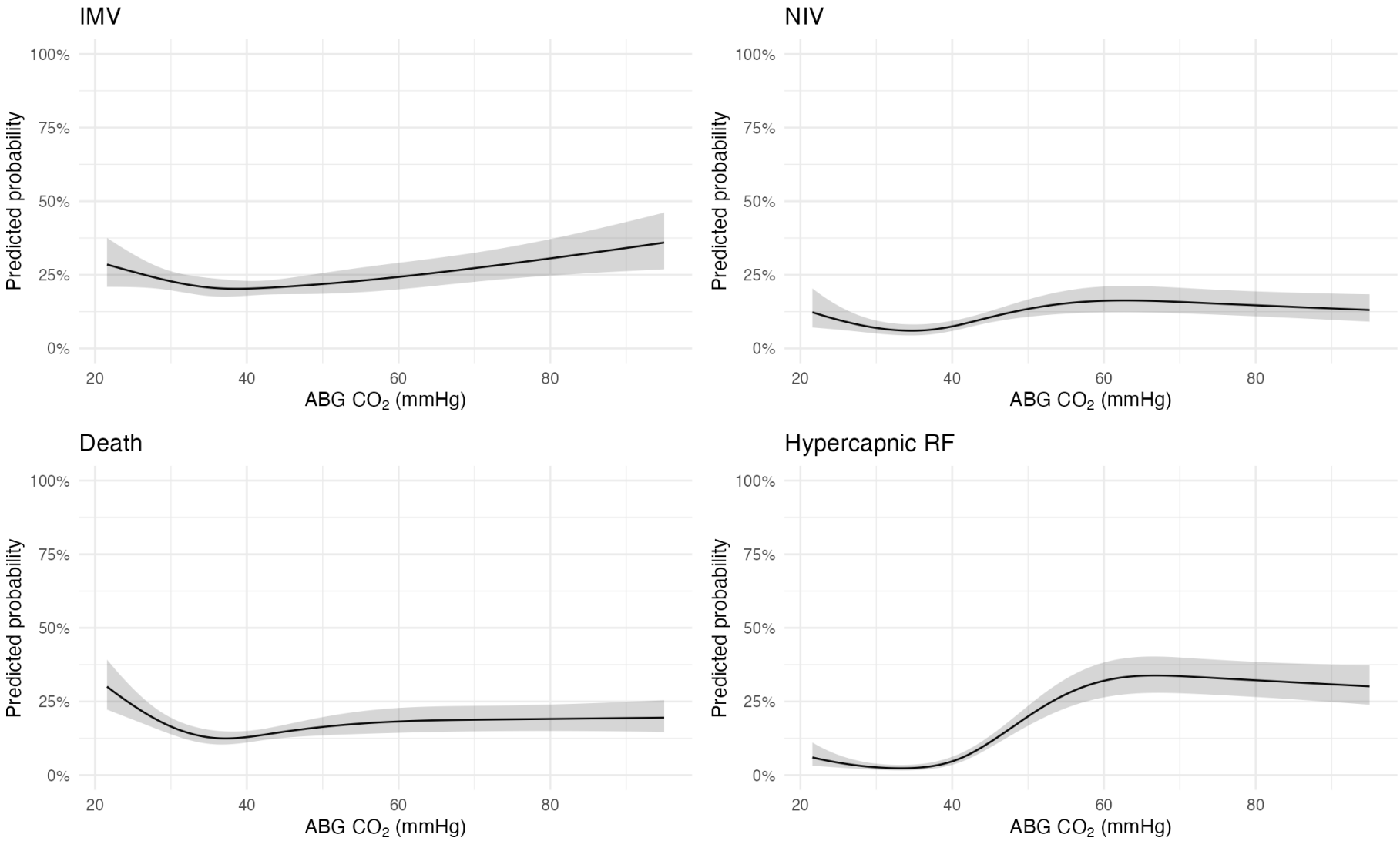
Restricting plots bewtween 0.02 and 0.98

3.0.3 5.3 ABG IPW spline models (2–98th percentile)

```
subset_data_abg <- subset_data %>%
  filter(!is.na(paco2)) %>%                    # implies has_abg == 1
  select(paco2, imv_proc, niv_proc, death_60d,
         hypercap_resp_failure, w_abg) %>%
  filter(complete.cases(.))


#   3. weighted logistic spline models with robust SEs
dd <- datadist(subset_data_abg); options(datadist = "dd")

fitfun <- function(formula)
  svyglm(
    formula,
    design = svydesign(ids = ~1, weights = ~w_abg, data = subset_data_abg),
    family = quasibinomial()
  )

fit_imv_abg   <- fitfun(imv_proc              ~ rcs(paco2, 4))
fit_niv_abg   <- fitfun(niv_proc              ~ rcs(paco2, 4))
fit_death_abg <- fitfun(death_60d                 ~ rcs(paco2, 4))
fit_hcrf_abg  <- fitfun(hypercap_resp_failure ~ rcs(paco2, 4))

#   4. prediction helper
mkpred <- function(fit, data_ref) {
  # 1. Grid of PaCO2 values restricted to 2nd-98th percentile
  q <- quantile(data_ref$paco2, probs = c(0.02, 0.98), na.rm = TRUE)
  newd <- data.frame(
    paco2 = seq(q[1], q[2], length.out = 200)
  )

  # 2. Design (model) matrix for the new data
  mm <- model.matrix(delete.response(terms(fit)), data = newd)
```

```r
  # 3. Linear predictor and its standard error
  eta  <- mm %*% coef(fit)
  vcov <- vcov(fit)
  se   <- sqrt(rowSums((mm %*% vcov) * mm))

  # 4. Transform to probability scale
  transform(
    newd,
    yhat  = plogis(eta),
    lower = plogis(eta - 1.96 * se),
    upper = plogis(eta + 1.96 * se)
  )
}

pred_imv_abg   <- mkpred(fit_imv_abg,   subset_data_abg)
pred_niv_abg   <- mkpred(fit_niv_abg,   subset_data_abg)
pred_death_abg <- mkpred(fit_death_abg, subset_data_abg)
pred_hcrf_abg  <- mkpred(fit_hcrf_abg,  subset_data_abg)

#   5. plotting
xlab <- expression(paste("ABG CO"[2], " (mmHg)"))

plt <- function(dat, title)
  ggplot(dat, aes(paco2, yhat)) +
    geom_line() +
    geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2) +
    scale_y_continuous(limits = c(0, 1), labels = percent_format(accuracy = 1)) +
    labs(title = title, x = xlab, y = "Predicted probability") +
    theme_minimal()

(patchwork::wrap_plots(
    plt(pred_imv_abg,   "IMV"),
    plt(pred_niv_abg,   "NIV"),
    plt(pred_death_abg, "Death"),
    plt(pred_hcrf_abg,  "Hypercapnic RF"),
    ncol = 2
```

```
  )
) +
  plot_annotation(
    title = expression(
      paste("Propensity-weighted predicted probability by ABG CO"[2],
            " (restricted cubic spline)")
    )
)
```



Chunk ipw-abg-rcs-trimmed runtime: 0.41 s

VBG - changed trees and bag fraction

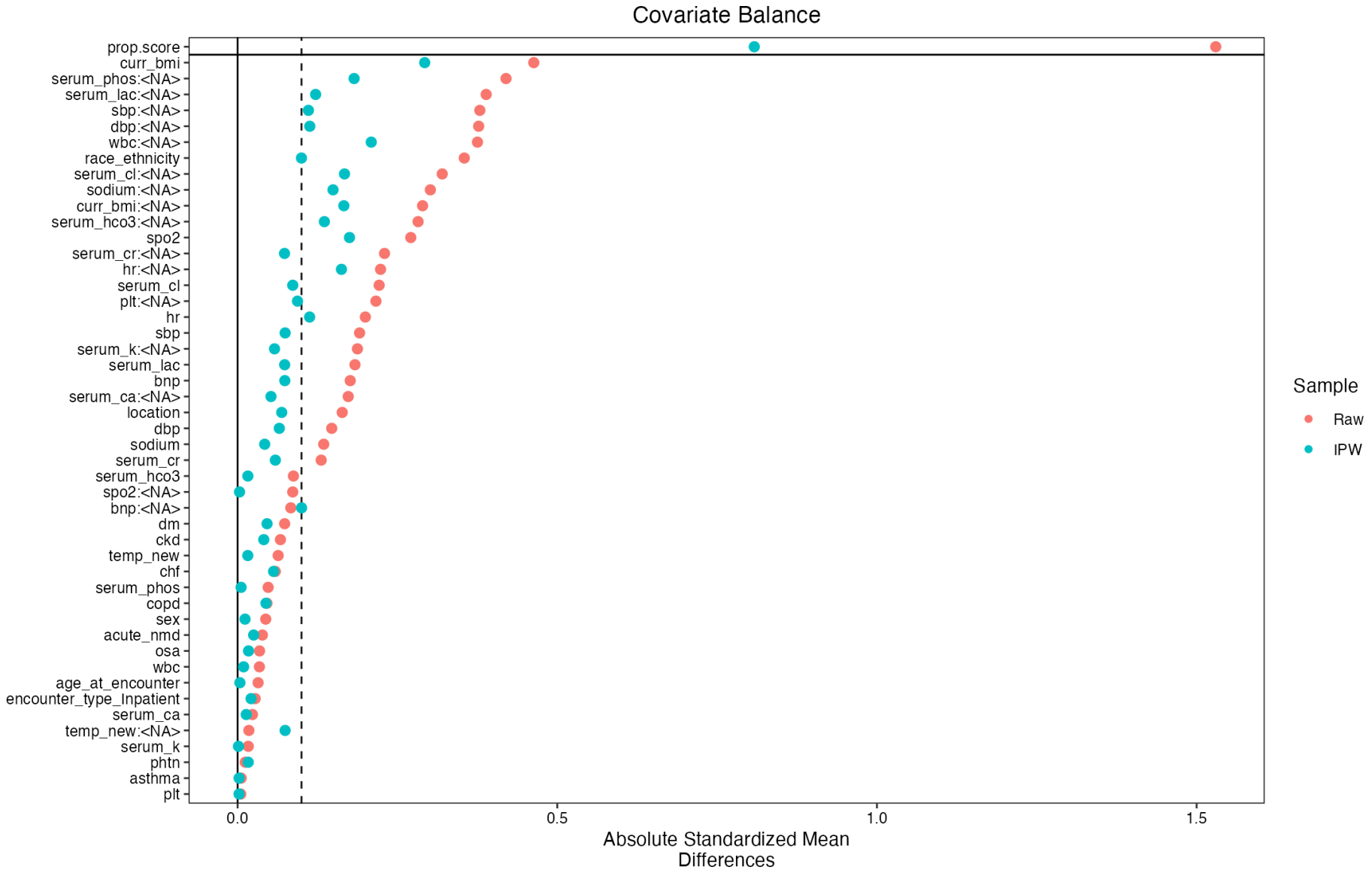
3.0.4 5.4 VBG IPW weighting and spline models

```
#  Inverse-propensity weighting & outcome modelling for **VBG** cohort
#    - mirrored 1-to-1 to the validated ABG workflow

set.seed(42)

# 1. IPW for VBG ---------------------------------------------------------------
set.seed(42)
w_vbg <- do.call(
  weightit,
  c(
    list(
      formula_vbg,
      data        = subset_data,
      method      = "gbm",
      estimand    = "ATE",
      missing     = "ind",
      include.obj = TRUE
    ),
    gbm_params
  )
)

# Stabilise & winsorise weights
w <- w_vbg$weights
w <- w / mean(w)
cut <- quantile(w, c(0.01, 0.99), na.rm = TRUE)
w   <- pmin(pmax(w, cut[1]), cut[2])
w <- w / mean(w)

w_vbg$weights   <- w
subset_data$w_vbg <- w

v_bal <- bal.tab(
```

```
  w_vbg,
  un = TRUE,
  m.threshold = 0.1,
  binary = "std",
  s.d.denom = "pooled"
)
```

```
Warning: Missing values exist in the covariates. Displayed values omit these
observations.
```

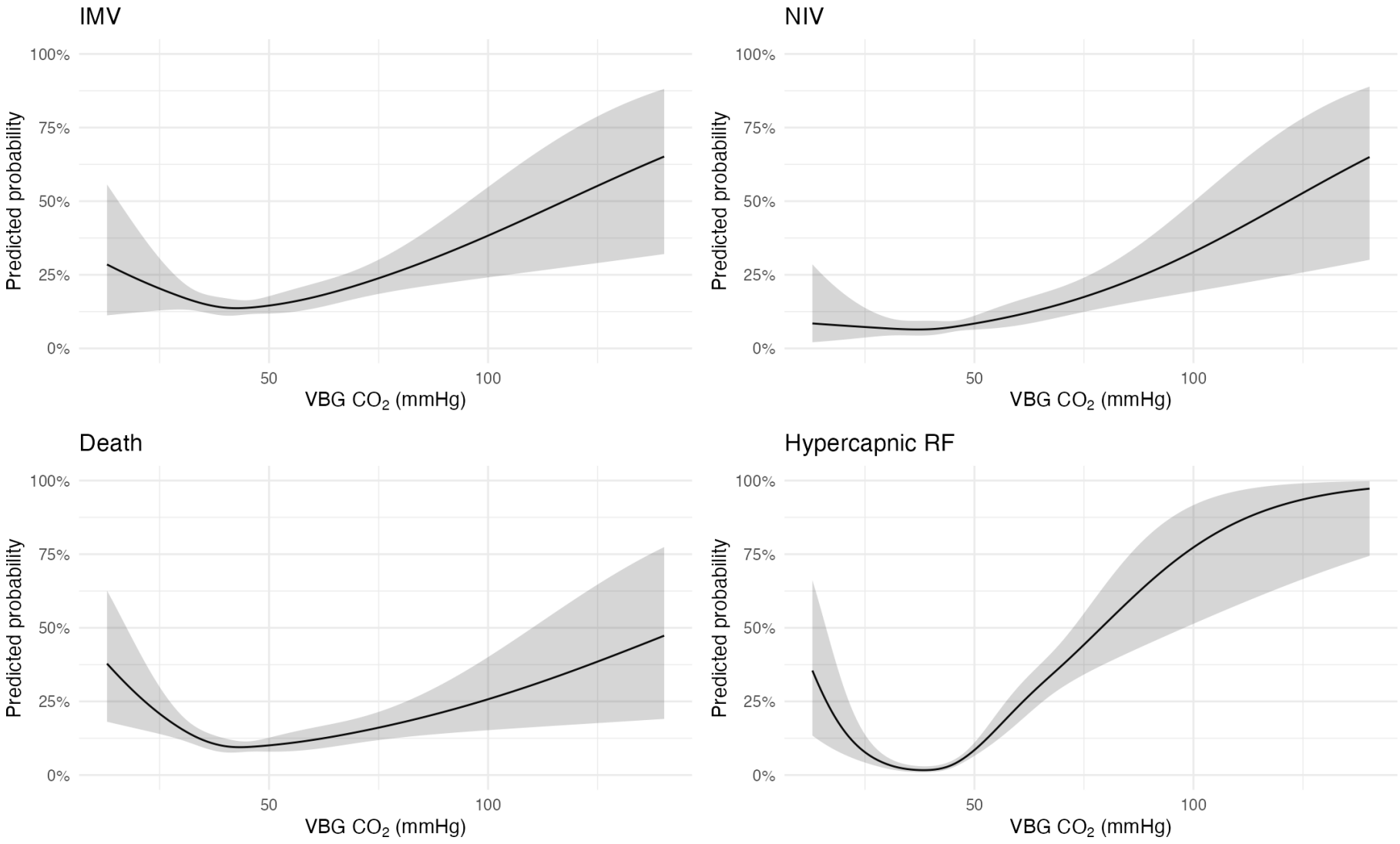
```
love.plot(
  v_bal,
  stats        = "m",          # standardized mean differences only
  abs          = TRUE,
  var.order    = "unadjusted",
  sample.names = c("Raw", "IPW")
)
```




```
# 2. Analysis set (VBG only) ------------------------------------------------
subset_data_vbg <- subset_data %>%
  filter(!is.na(vbg_co2)) %>%
  select(vbg_co2, imv_proc, niv_proc, death_60d,
```

```r
         hypercap_resp_failure, w_vbg) %>%
  filter(complete.cases(.))

# 3. Weighted spline models ------------------------------------------------
dd_vbg <- datadist(subset_data_vbg)
options(datadist = "dd_vbg")

fitfun <- function(formula)
  svyglm(
    formula,
    design = svydesign(ids = ~1, weights = ~w_vbg, data = subset_data_vbg),
    family = quasibinomial()
  )

fit_imv_vbg   <- fitfun(imv_proc              ~ rcs(vbg_co2, 4))
fit_niv_vbg   <- fitfun(niv_proc              ~ rcs(vbg_co2, 4))
fit_death_vbg <- fitfun(death_60d             ~ rcs(vbg_co2, 4))
fit_hcrf_vbg  <- fitfun(hypercap_resp_failure ~ rcs(vbg_co2, 4))

# 4. Prediction helper -----------------------------------------------------
mkpred <- function(fit, data_ref) {
  newd <- data.frame(
    vbg_co2 = seq(min(data_ref$vbg_co2, na.rm = TRUE),
                  max(data_ref$vbg_co2, na.rm = TRUE),
                  length.out = 200)
  )
  mm   <- model.matrix(delete.response(terms(fit)), newd)
  eta  <- mm %*% coef(fit)
  vcov <- vcov(fit)
  se   <- sqrt(rowSums((mm %*% vcov) * mm))
  transform(
    newd,
    yhat  = plogis(eta),
    lower = plogis(eta - 1.96 * se),
    upper = plogis(eta + 1.96 * se)
  )
```

```r
}

pred_imv_vbg   <- mkpred(fit_imv_vbg,   subset_data_vbg)
pred_niv_vbg   <- mkpred(fit_niv_vbg,   subset_data_vbg)
pred_death_vbg <- mkpred(fit_death_vbg, subset_data_vbg)
pred_hcrf_vbg  <- mkpred(fit_hcrf_vbg,  subset_data_vbg)

# 5. Plotting (gray scheme) ----------------------------------------------------
xlab <- expression(paste("VBG CO"[2], " (mmHg)"))

plt <- function(dat, title)
  ggplot(dat, aes(vbg_co2, yhat)) +
    geom_line() +
    geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2) +
    scale_y_continuous(limits = c(0, 1), labels = percent_format(accuracy = 1)) +
    labs(title = title, x = xlab, y = "Predicted probability") +
    theme_minimal()

(patchwork::wrap_plots(
    plt(pred_imv_vbg,   "IMV"),
    plt(pred_niv_vbg,   "NIV"),
    plt(pred_death_vbg, "Death"),
    plt(pred_hcrf_vbg,  "Hypercapnic RF"),
    ncol = 2
  )
) +
  plot_annotation(
    title = expression(
      paste("Propensity-weighted predicted probability by VBG CO"[2],
            " (restricted cubic spline)")
    )
)
```



Chunk ipw-vbg-workflow runtime: 6.30 s

Calculated VBG to ABG / Farkas

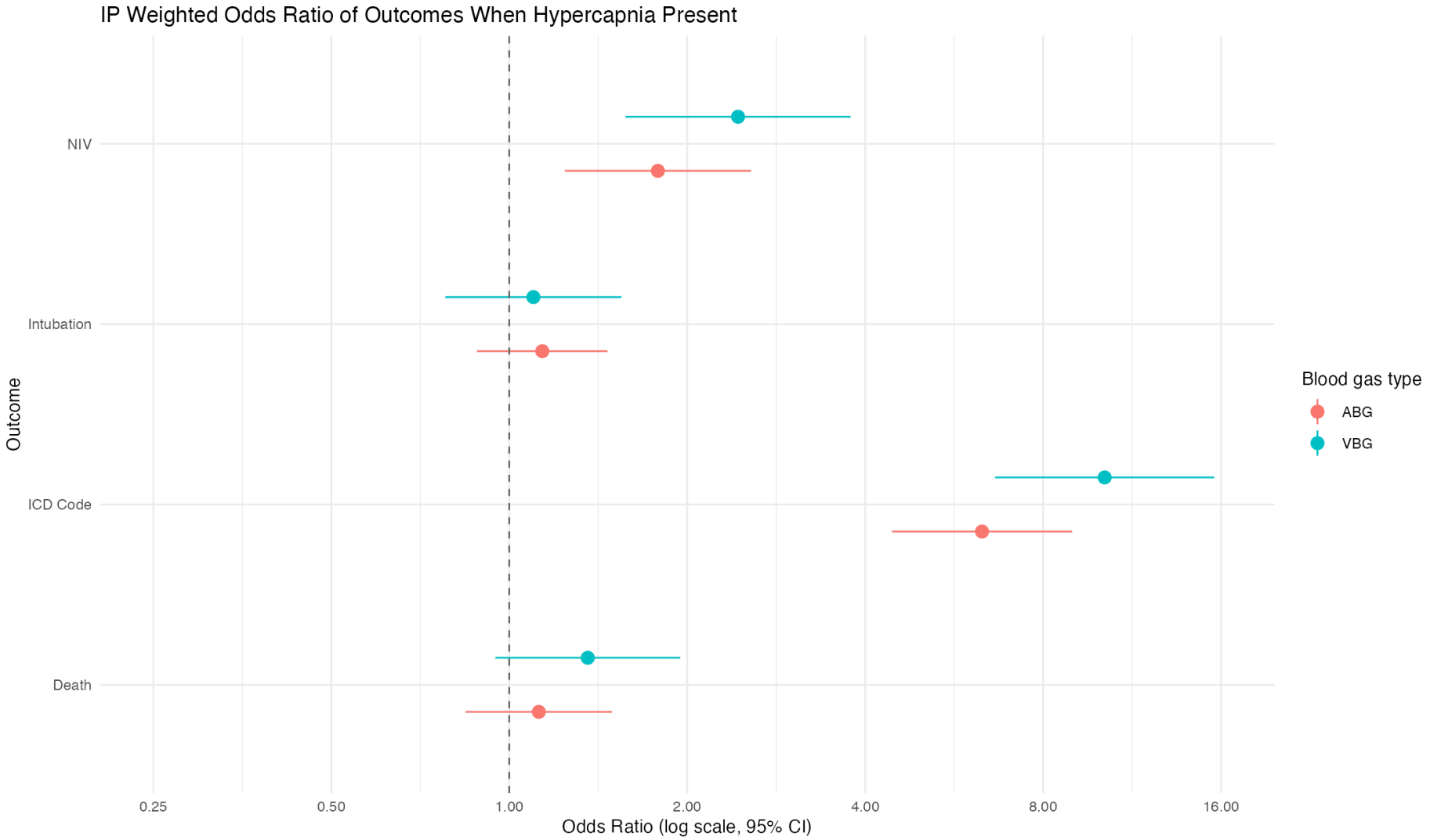
3.1 5) Weighted effect estimates

New weighted binary regression figures.

```
#  IP-weighted odds-ratio plot (ABG, VBG, Calculated-ABG)
#    - exact analogue of the un-weighted figure
#

# weights already attached earlier:
#   • w_abg         - propensity for *ABG*   (column in subset_data)
#   • w_vbg         - propensity for *VBG*   (column in subset_data)
#   •     - same weights, used for calculated ABG CO2

# 1. helper to fit an IP-weighted GLM and return tidy OR ----------------------
tidy_ipw <- function(data, outcome, exposure, weight_var,
                     group_label, outcome_label) {
  des <- svydesign(ids = ~1, weights = as.formula(paste0("~", weight_var)),
                   data = data)
  mod <- svyglm(
    as.formula(paste0(outcome, " ~ ", exposure)),
    design = des,
    family = quasibinomial()
  )

  tidy(mod, exponentiate = TRUE, conf.int = TRUE) %>%
    filter(term == exposure) %>%                # keep the exposure row
    mutate(group = group_label, outcome = outcome_label)
}

# 2. cohort-specific data frames ----------------------------------------------
abg_df   <- subset_data %>% filter(has_abg == 1)
vbg_df   <- subset_data %>% filter(has_vbg == 1)

# 3. fit models & collect estimates -------------------------------------------
ipw_estimates <- bind_rows(
  # ABG
```

```
  tidy_ipw(abg_df,  "imv_proc",              "hypercap_on_abg", "w_abg",      "ABG",          "Intubation"),
  tidy_ipw(abg_df,  "niv_proc",              "hypercap_on_abg", "w_abg",      "ABG",          "NIV"),
  tidy_ipw(abg_df,  "death_60d",                 "hypercap_on_abg", "w_abg",      "ABG",          "Death"),
  tidy_ipw(abg_df,  "hypercap_resp_failure", "hypercap_on_abg", "w_abg",      "ABG",          "ICD Code"),

  # VBG
  tidy_ipw(vbg_df,  "imv_proc",              "hypercap_on_vbg", "w_vbg",      "VBG",          "Intubation"),
  tidy_ipw(vbg_df,  "niv_proc",              "hypercap_on_vbg", "w_vbg",      "VBG",          "NIV"),
  tidy_ipw(vbg_df,  "death_60d",                 "hypercap_on_vbg", "w_vbg",      "VBG",          "Death"),
  tidy_ipw(vbg_df,  "hypercap_resp_failure", "hypercap_on_vbg", "w_vbg",      "VBG",          "ICD Code")
)

# 4. plotting ------------------------------------------------------------------
ipw_estimates$group <- factor(
  ipw_estimates$group,
  levels = c("ABG", "VBG")
)

ggplot(
  ipw_estimates,
  aes(
    x     = outcome,
    y     = estimate,
    ymin  = conf.low,
    ymax  = conf.high,
    color = group
  )
) +
  geom_pointrange(position = position_dodge(width = 0.6), size = 0.6) +
  geom_hline(yintercept = 1, linetype = "dashed", colour = "grey40") +
  scale_y_log10(
    breaks = c(0.25, 0.5, 1, 2, 4, 8, 16),
    limits = c(0.25, 16),
    labels = number_format(accuracy = 0.01)
  ) +
  coord_flip() +
```

```
  labs(
    title  = "IP Weighted Odds Ratio of Outcomes When Hypercapnia Present",
    x      = "Outcome",
    y      = "Odds Ratio (log scale, 95% CI)",
    color  = "Blood gas type",
    caption = paste(
      "Inverse-probability weights adjust for covariates associated with receiving each blood-gas.",
      "Models are fitted within their respective cohorts:",
      "ABG (weights = w_abg), VBG (w_vbg).",
      "Numerator = hypercapnic; denominator = normocapnic within cohort.",
      sep = "\n"
    )
  ) +
  theme_minimal(base_size = 10) +
  theme(plot.caption = element_text(hjust = 0))
```



Chunk ipw-binary-or-plot runtime: 0.37 s

3.1.1 5.5 Three-level PCO2 categories (weighted; ABG, VBG)

Three Groups with Weights

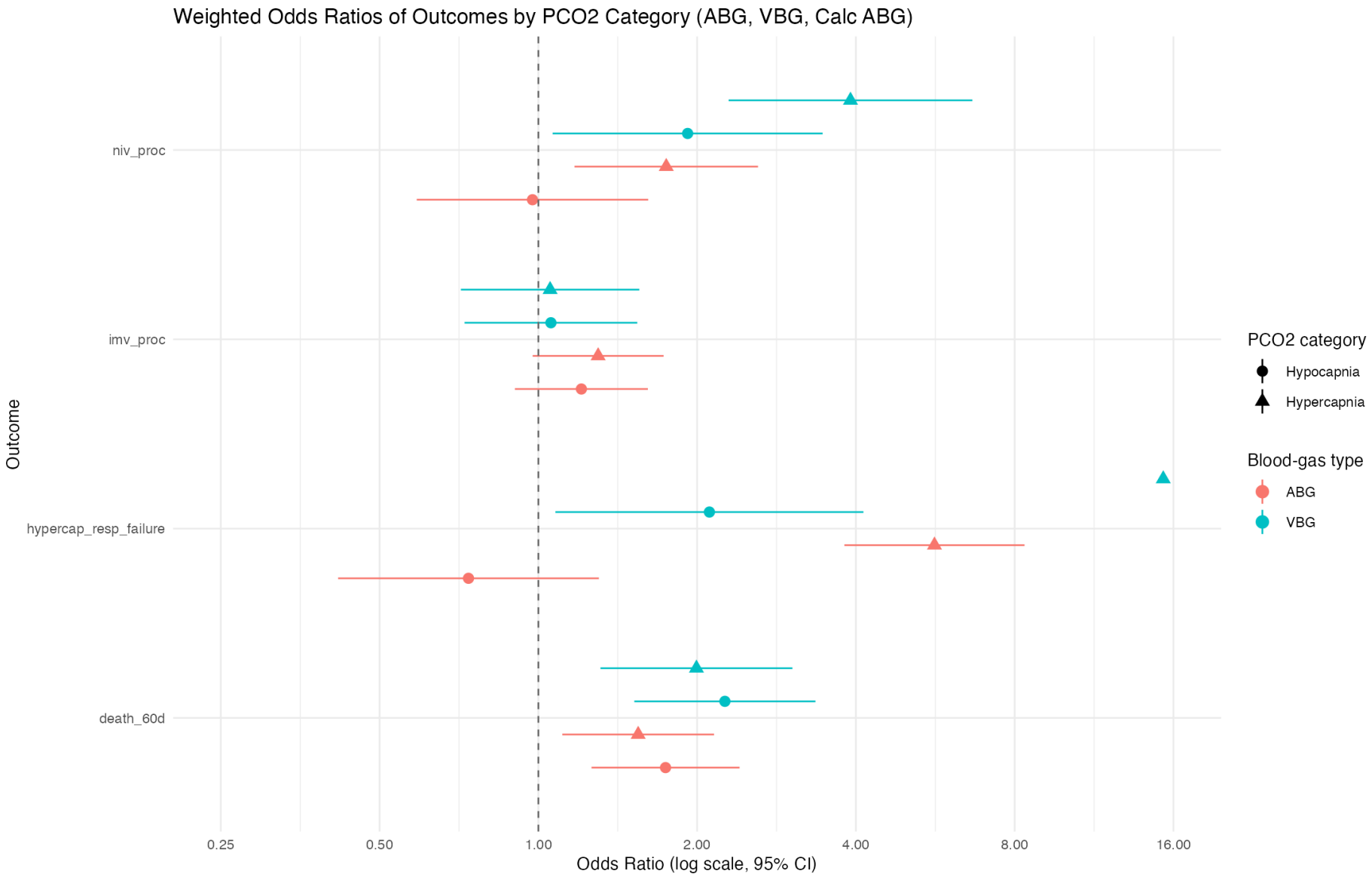
```
library(dplyr)
library(survey)
library(broom)
library(ggplot2)
library(scales)

#   1. Create PCO2 categories
subset_data <- subset_data %>%
  mutate(
    pco2_cat_abg = case_when(
      !is.na(paco2) & paco2 < 35 ~ "Hypocapnia",
      !is.na(paco2) & paco2 >= 35 & paco2 <= 45 ~ "Eucapnia",
      !is.na(paco2) & paco2 > 45 ~ "Hypercapnia",
      TRUE ~ NA_character_
    ),
    pco2_cat_vbg = case_when(
      !is.na(vbg_co2) & vbg_co2 < 40 ~ "Hypocapnia",
      !is.na(vbg_co2) & vbg_co2 >= 40 & vbg_co2 <= 50 ~ "Eucapnia",
      !is.na(vbg_co2) & vbg_co2 > 50 ~ "Hypercapnia",
      TRUE ~ NA_character_
    )
  )

#   2. Function: weighted logistic regression & OR extraction
run_weighted_or <- function(data, outcome, cat_var, weight_var, group_name) {
  dat <- data %>%
    filter(
      !is.na(.data[[cat_var]]),
      !is.na(.data[[outcome]]),
      !is.na(.data[[weight_var]]),
      .data[[weight_var]] > 0
    ) %>%
```

```
    mutate(
      !!cat_var := factor(.data[[cat_var]],
                          levels = c("Eucapnia", "Hypocapnia", "Hypercapnia"))
    ) %>%
    droplevels()

  design <- svydesign(
    ids = ~1,
    weights = as.formula(paste0("~", weight_var)),
    data = dat
  )

  fit <- svyglm(as.formula(paste(outcome, "~", cat_var)),
                design = design, family = quasibinomial())

  tidy(fit, exponentiate = TRUE, conf.int = TRUE) %>%
    filter(term != "(Intercept)") %>%
    mutate(
      group    = group_name,
      outcome  = outcome,
      exposure = gsub(paste0(cat_var), "", term) %>%
                   gsub("`", "", .)
    )
}

#   3. Run across outcomes & cohorts
outcomes <- c("imv_proc", "niv_proc", "death_60d", "hypercap_resp_failure")

combined_or_df <- bind_rows(
  lapply(outcomes, function(out)
    run_weighted_or(subset_data, out, "pco2_cat_abg",  "w_abg",      "ABG")),
  lapply(outcomes, function(out)
    run_weighted_or(subset_data, out, "pco2_cat_vbg",  "w_vbg",      "VBG"))
)

# Ensure nice ordering
```

```r
combined_or_df$group    <- factor(combined_or_df$group,
                                  levels = c("ABG", "VBG"))
combined_or_df$exposure <- factor(combined_or_df$exposure,
                                  levels = c("Eucapnia", "Hypocapnia", "Hypercapnia"))

#   4. Plot weighted odds ratios
ggplot(
  combined_or_df,
  aes(
    x      = outcome,
    y      = estimate,
    ymin   = conf.low,
    ymax   = conf.high,
    color  = group,
    shape  = exposure
  )
) +
  geom_pointrange(position = position_dodge(width = 0.7), size = 0.6) +
  geom_hline(yintercept = 1, linetype = "dashed", colour = "grey40") +
  scale_y_log10(
    breaks = c(0.25, 0.5, 1, 2, 4, 8, 16),
    limits = c(0.25, 16),
    labels = number_format(accuracy = 0.01)
  ) +
  coord_flip() +
  labs(
    title  = "Weighted Odds Ratios of Outcomes by PCO2 Category (ABG, VBG, Calc ABG)",
    x      = "Outcome",
    y      = "Odds Ratio (log scale, 95% CI)",
    color  = "Blood-gas type",
    shape  = "PCO2 category"
  ) +
  theme_minimal(base_size = 10) +
  theme(plot.caption = element_text(hjust = 0))
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
```

(`geom_segment()`).



Chunk ipw-three-level-pco2-all runtime: 0.51 s

3.1.2 5.6 Three-level PCO2 categories (weighted; ABG vs VBG only)

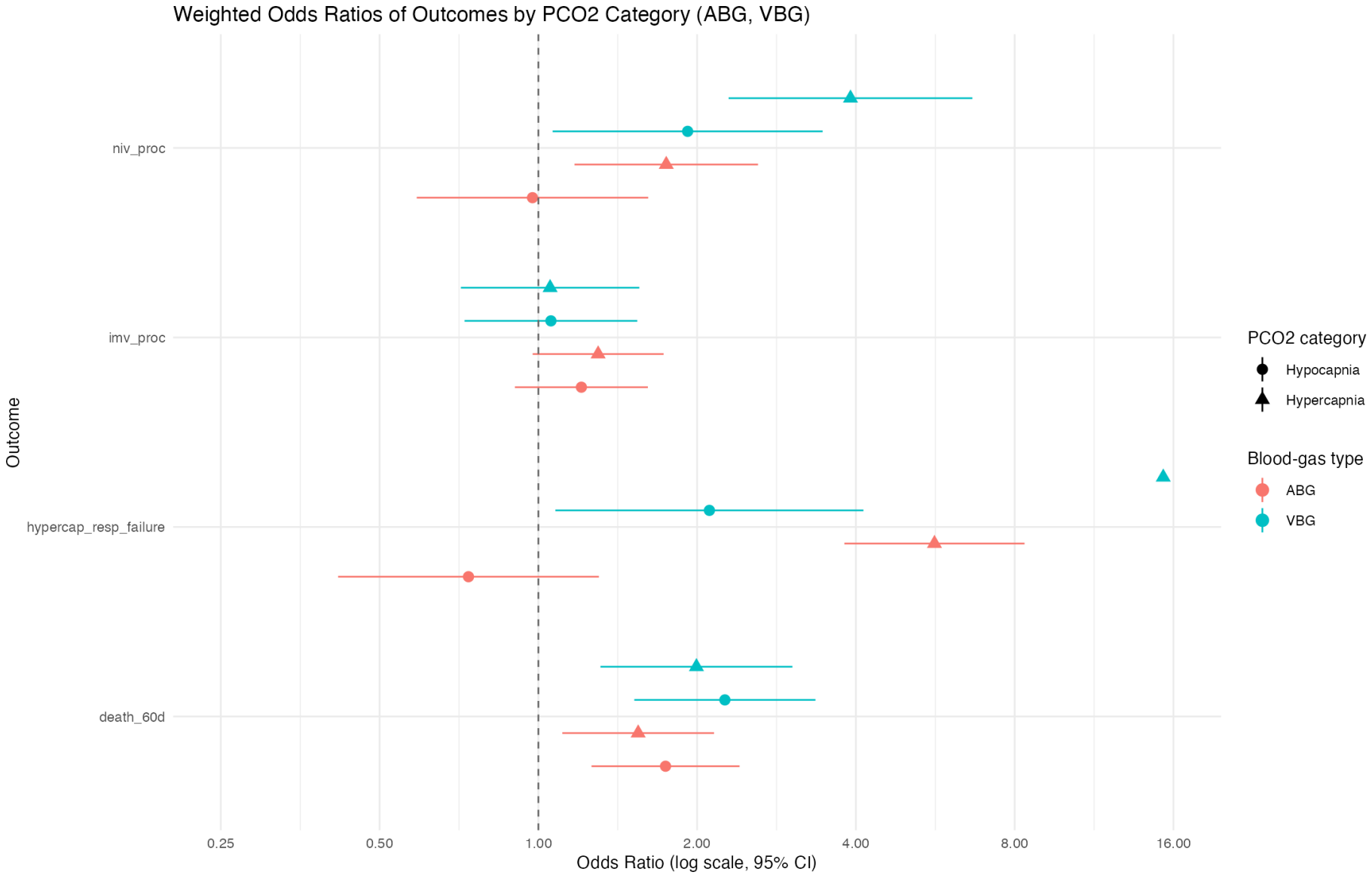
Three groups with weights: Just ABG and VBG

```
library(dplyr)
library(survey)
library(broom)
library(ggplot2)
library(scales)

#   2. Function: weighted logistic regression & OR extraction
run_weighted_or <- function(data, outcome, cat_var, weight_var, group_name) {
  dat <- data %>%
    filter(
      !is.na(.data[[cat_var]]),
      !is.na(.data[[outcome]]),
      !is.na(.data[[weight_var]]),
      .data[[weight_var]] > 0
    ) %>%
    mutate(
      !!cat_var := factor(.data[[cat_var]],
                          levels = c("Eucapnia", "Hypocapnia", "Hypercapnia"))
    ) %>%
    droplevels()

  design <- svydesign(
    ids = ~1,
    weights = as.formula(paste0("~", weight_var)),
    data = dat
  )

  fit <- svyglm(as.formula(paste(outcome, "~", cat_var)),
                design = design, family = quasibinomial())

  tidy(fit, exponentiate = TRUE, conf.int = TRUE) %>%
    filter(term != "(Intercept)") %>%
```

```
    mutate(
      group    = group_name,
      outcome  = outcome,
      exposure = gsub(paste0(cat_var), "", term) %>%
                   gsub("`", "", .)
    )
}

#   3. Run across outcomes & cohorts
outcomes <- c("imv_proc", "niv_proc", "death_60d", "hypercap_resp_failure")

combined_or_df <- bind_rows(
  lapply(outcomes, function(out)
    run_weighted_or(subset_data, out, "pco2_cat_abg",  "w_abg",      "ABG")),
  lapply(outcomes, function(out)
    run_weighted_or(subset_data, out, "pco2_cat_vbg",  "w_vbg",      "VBG"))
)

# Ensure nice ordering
combined_or_df$group    <- factor(combined_or_df$group,
                                  levels = c("ABG", "VBG"))
combined_or_df$exposure <- factor(combined_or_df$exposure,
                                  levels = c("Eucapnia", "Hypocapnia", "Hypercapnia"))

#   4. Plot weighted odds ratios
ggplot(
  combined_or_df,
  aes(
    x      = outcome,
    y      = estimate,
    ymin   = conf.low,
    ymax   = conf.high,
    color  = group,
    shape  = exposure
  )
) +
```

```
  geom_pointrange(position = position_dodge(width = 0.7), size = 0.6) +
  geom_hline(yintercept = 1, linetype = "dashed", colour = "grey40") +
  scale_y_log10(
    breaks = c(0.25, 0.5, 1, 2, 4, 8, 16),
    limits = c(0.25, 16),
    labels = number_format(accuracy = 0.01)
  ) +
  coord_flip() +
  labs(
    title  = "Weighted Odds Ratios of Outcomes by PCO2 Category (ABG, VBG)",
    x      = "Outcome",
    y      = "Odds Ratio (log scale, 95% CI)",
    color  = "Blood-gas type",
    shape  = "PCO2 category"
  ) +
  theme_minimal(base_size = 10) +
  theme(plot.caption = element_text(hjust = 0))
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_segment()`).
```



Chunk ipw-three-level-pco2-abg-vbg runtime: 0.42 s

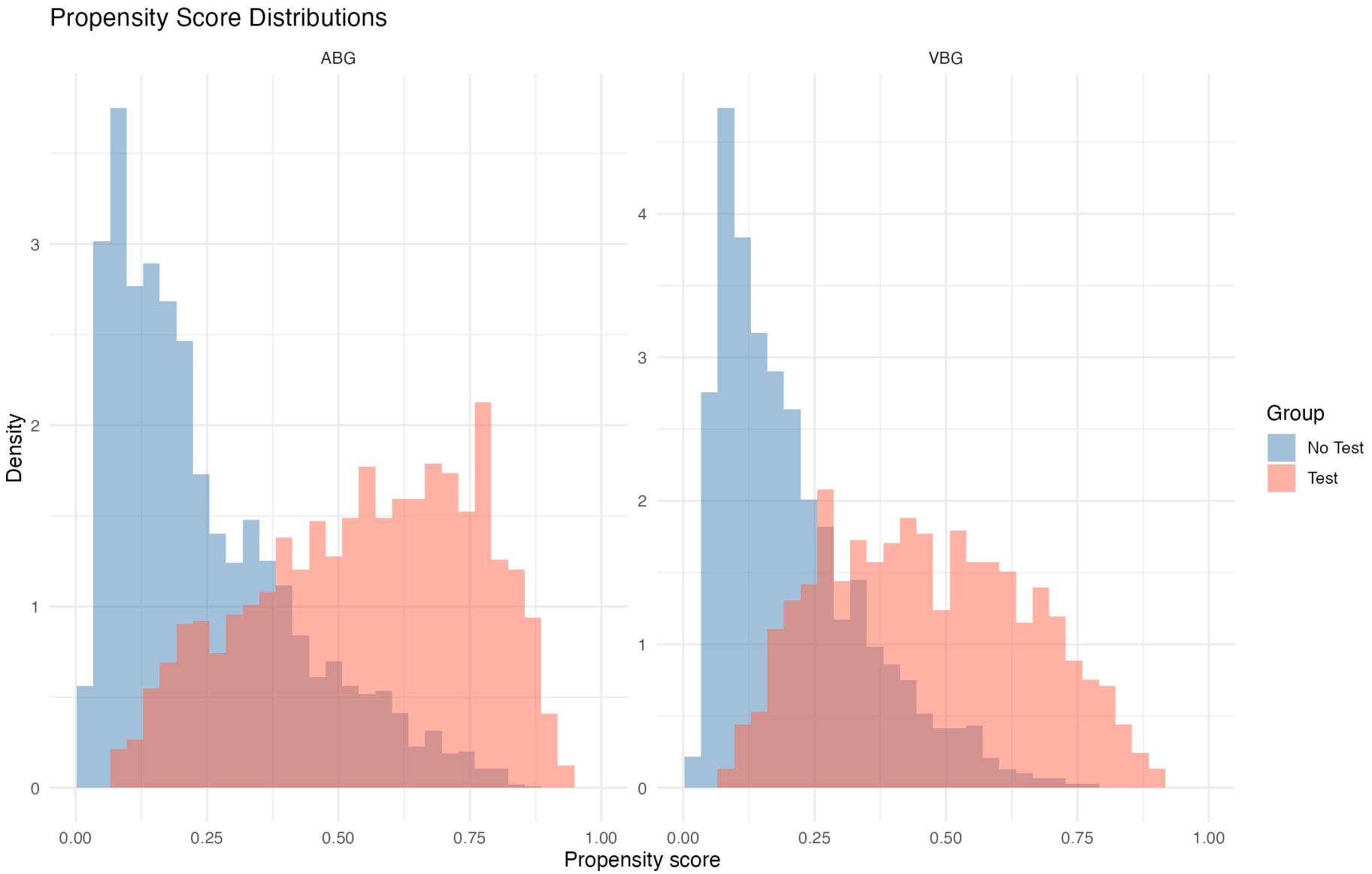
3.2 6) Propensity score diagnostics

Plotting propensity scores

```
# --- Propensity score histograms (ABG / VBG / Calculated-ABG) -----------------
# ABG = arterial blood gas; VBG = venous blood gas

library(dplyr)
library(ggplot2)
library(scales)

# Resolve WeightIt objects regardless of naming used upstream
w_abg_obj <- if (exists("w_abg")) w_abg else if (exists("weight_model")) weight_model else NULL
w_vbg_obj <- if (exists("w_vbg")) w_vbg else NULL

if (is.null(w_abg_obj)) stop("ABG WeightIt object not found. Define `w_abg` or `weight_model` before this block.")
if (!"has_abg" %in% names(subset_data)) stop("`subset_data` must contain `has_abg` for ABG PS plotting.")

# Build list of per-cohort PS data frames conditionally (so missing cohorts don't error)
ps_dfs <- list(
  ABG = data.frame(
    ps        = w_abg_obj$ps,
    treat     = subset_data$has_abg,
    ScoreType = "ABG"
  )
)

if (!is.null(w_vbg_obj) && "has_vbg" %in% names(subset_data)) {
  ps_dfs$VBG <- data.frame(
    ps        = w_vbg_obj$ps,
    treat     = subset_data$has_vbg,
    ScoreType = "VBG"
  )
} else if (is.null(w_vbg_obj)) {
  message("Note: VBG WeightIt object `w_vbg` not found; skipping VBG panel.")
}
```

```r
# Bind, clean, and factorize for plotting
df_ps <- bind_rows(ps_dfs) %>%
  filter(!is.na(ps), !is.na(treat)) %>%
  mutate(
    treat     = factor(treat, levels = c(0, 1), labels = c("No Test", "Test")),
    ScoreType = factor(ScoreType, levels = c("ABG", "VBG"))
  )

# Plot
ggplot(df_ps, aes(x = ps, fill = treat)) +
  geom_histogram(aes(y = ..density..), alpha = 0.5,
                 position = "identity", bins = 30) +
  scale_fill_manual(values = c("No Test" = "steelblue", "Test" = "tomato")) +
  facet_wrap(~ScoreType, scales = "free_y") +
  coord_cartesian(xlim = c(0, 1)) +
  labs(
    title = "Propensity Score Distributions",
    x     = "Propensity score",
    y     = "Density",
    fill  = "Group"
  ) +
  theme_minimal(base_size = 12)
```

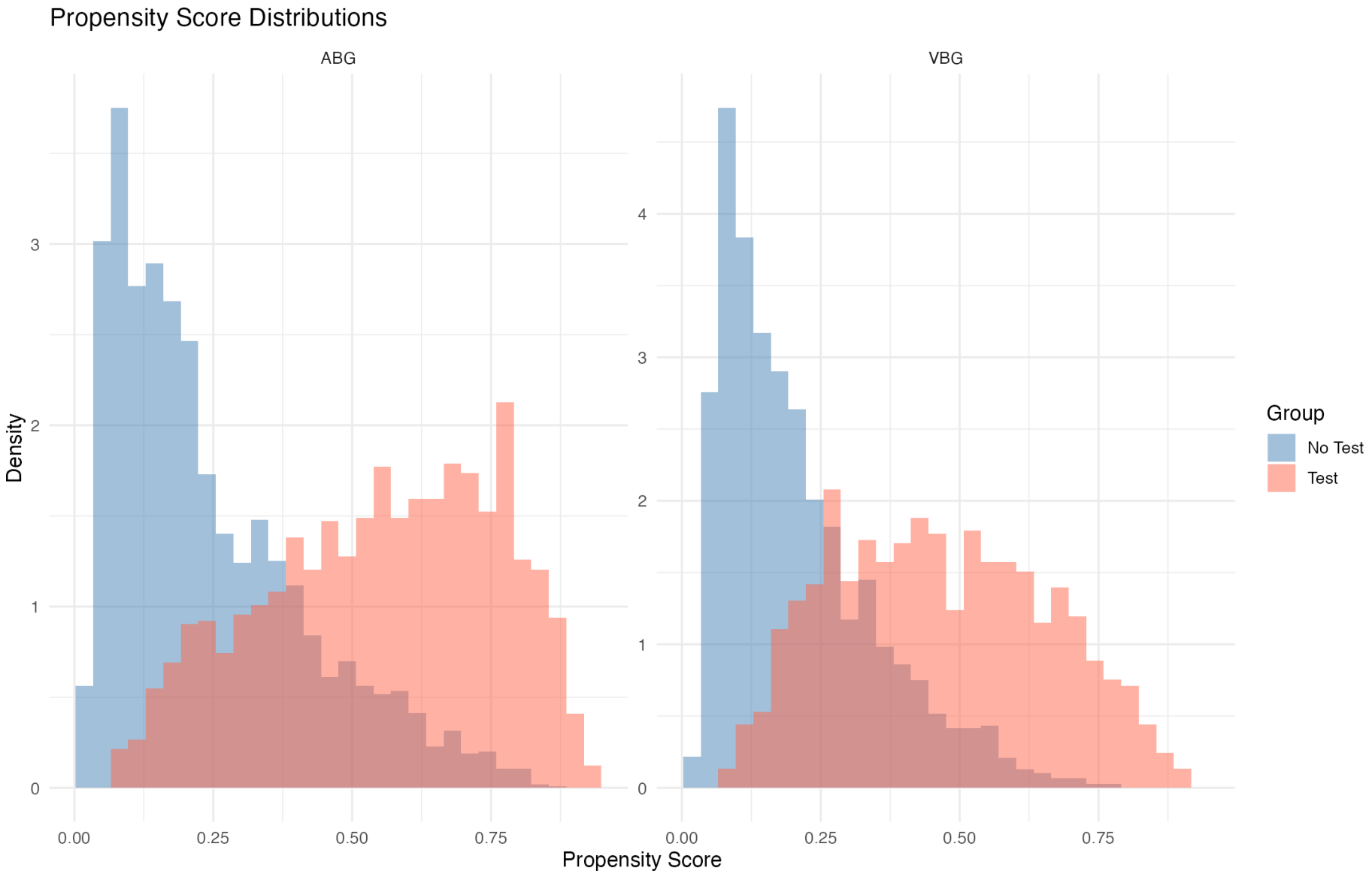
```
Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
i Please use `after_stat(density)` instead.
```



Chunk propensity-histograms-conditional runtime: 0.24 s

```
df_ps <- bind_rows(
  data.frame(
```

```
    ps        = w_abg$ps,
    treat     = subset_data$has_abg,
    ScoreType = "ABG"
  ),
  data.frame(
    ps        = w_vbg$ps,
    treat     = subset_data$has_vbg,
    ScoreType = "VBG"
  )
) %>%
  mutate(
    treat     = factor(treat, levels = c(0,1), labels = c("No Test", "Test")),
    ScoreType = factor(ScoreType, levels = c("ABG","VBG"))
  )

ggplot(df_ps, aes(x = ps, fill = treat)) +
  geom_histogram(aes(y = ..density..), alpha = 0.5,
                 position = "identity", bins = 30) +
  scale_fill_manual(values = c("No Test" = "steelblue", "Test" = "tomato")) +
  facet_wrap(~ScoreType, scales = "free_y") +
  labs(
    title = "Propensity Score Distributions",
    x = "Propensity Score",
    y = "Density",
    fill = "Group"
  ) +
  theme_minimal(base_size = 12)
```



Chunk propensity-histograms-all runtime: 0.18 s

4 Multiple Imputation Analysis

added 12/6/2025

4.1 7) Packages and reproducibility

```
# Core MI + diagnostics
library(mice)         # chained equations (MICE)
```

```
Attaching package: 'mice'

The following object is masked from 'package:stats':

    filter

The following objects are masked from 'package:base':

    cbind, rbind
```

```
library(miceadds)     # pooling helpers & utilities
```

```
* miceadds 3.18-36 (2025-09-12 09:54:54)
```

```
library(naniar)       # missingness summaries/plots
```

```
Attaching package: 'naniar'

The following object is masked from 'package:miceadds':

    prop_miss
```

```
library(visdat)       # quick type/missingness viz
library(skimr)        # data skim for large frames
```

```
Attaching package: 'skimr'
```

```
The following object is masked from 'package:naniar':

    n_complete
```

```
# Modeling
library(WeightIt)     # GBM propensity with weights
library(gbm)          # underlying GBM engine
library(survey)       # svyglm outcome models
library(cobalt)       # balance diagnostics
library(broom)        # tidy model outputs
library(dplyr)        # data manipulation
library(ggplot2)

# Pooling and MI bookkeeping
library(mitools)      # MIcombine for pooling (generic)
library(parallel)     # basic parallel where helpful

# Parallel + progress setup
library(future)
```

```
Attaching package: 'future'
```

```
The following object is masked from 'package:survival':

    cluster
```

```r
# setup
library(future.apply)
library(progressr)

workers <- max(1L, future::availableCores() - 1L)
future::plan(multisession, workers = workers)
on.exit(future::plan("sequential"), add = TRUE)

# choose a handler, but DO NOT make it global inside a knitted document
progressr::handlers(progressr::handler_rstudio)   # or handler_txtprogressbar
options(future.rng.onMisuse = "error")            # safer RNG with futures

set.seed(20251206)

# ensure a writable figure dir + stable device on macOS
if (!dir.exists("figs")) dir.create("figs", recursive = TRUE, showWarnings = FALSE)
knitr::opts_chunk$set(fig.path = "figs/", dev = "png", dpi = 144)
options(bitmapType = "cairo")  # prevents device issues on macOS
```

Chunk mi-packages runtime: 1.16 s

4.1.1 7.1 Missingness audit (what, where, how much)

```r
# High-level profiles
skimr::skim(subset_data)
```

Table 10: Data summary

Name	subset_data
Number of rows	5111
Number of columns	574
_____	
Column type frequency:	

character	13
Date	3
factor	19
numeric	539
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
encounter_id	0	1.00	4	6	0	5111	0
rfs	0	1.00	3	14	0	7	0
bnp_date	0	1.00	0	10	4130	348	0
serum_phos_date	0	1.00	0	10	2653	367	0
serum_ca_date	0	1.00	0	10	416	367	0
serum_albumin_date	0	1.00	0	10	1748	366	0
serum_tprot_date	0	1.00	0	10	1919	367	0
principal_diagnosis_indicator	0	1.00	0	1	4950	4	0
admitting_diagnosis	0	1.00	0	1	4950	2	0
reason_for_visit	0	1.00	0	1	4950	2	0
pat_enc_hash	0	1.00	9	12	0	5111	0
pco2_cat_abg	3321	0.35	8	11	0	3	0
pco2_cat_vbg	3679	0.28	8	11	0	3	0

Variable type: Date

skim_variable	n_missing	complete_rate	min	max	median	n_unique
encounter_date	0	1.00	2022-01-01	2022-12-31	2022-06-08	365
death_abs	4390	0.14	1995-06-01	2023-05-01	2022-07-01	41
ref_ym	0	1.00	2024-06-01	2024-06-01	2024-06-01	1

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
encounter_type	0	1	FALSE	2	Inp: 3357, Eme: 1754
abg_group	0	1	FALSE	3	No : 3321, ABG: 1245, ABG: 545, Mis: 0
vbg_group	0	1	FALSE	3	No : 3679, VBG: 1036, VBG: 396, Mis: 0
sex_label	0	1	FALSE	2	Mal: 2571, Fem: 2540
race_ethnicity_label	0	1	FALSE	7	Whi: 3116, Bla: 943, Unk: 576, His: 350
location_label	0	1	FALSE	4	Sou: 2385, Nor: 1272, Wes: 1072, Mid: 382
encounter_type_label	0	1	FALSE	2	Inp: 3357, Eme: 1754
osa_label	0	1	FALSE	2	No: 4243, Yes: 868
asthma_label	0	1	FALSE	2	No: 4433, Yes: 678
copd_label	0	1	FALSE	2	No: 4135, Yes: 976
chf_label	0	1	FALSE	2	No: 4137, Yes: 974
nmd_label	0	1	FALSE	2	No: 4906, Yes: 205
phtn_label	0	1	FALSE	2	No: 4721, Yes: 390
ckd_label	0	1	FALSE	2	No: 4145, Yes: 966
diabetes_label	0	1	FALSE	2	No: 3646, Yes: 1465
abg_status	0	1	FALSE	2	Did: 3321, Did: 1790
vbg_status	0	1	FALSE	2	Did: 3679, Did: 1432
hyper_abg	0	1	FALSE	2	Got: 4566, Got: 545
hyper_vbg	0	1	FALSE	2	Got: 4715, Got: 396

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
sex	0	1.00	0.50	0.50	0.00	0.00	1.00	1.00	1.00	
race	0	1.00	0.57	0.87	0.00	0.00	0.00	1.00	5.00	
ethnicity	0	1.00	0.41	0.76	0.00	0.00	0.00	0.00	2.00	
location	0	1.00	1.03	1.17	0.00	0.00	1.00	2.00	3.00	
age_at_encounter	0	1.00	58.59	17.92	18.00	46.00	61.00	73.00	90.00	
los	0	1.00	10.24	29.22	1.00	2.00	4.00	9.00	365.00	
curr_bmi	2862	0.44	31.40	8.51	11.50	24.70	29.92	38.78	49.78	
spo2	3632	0.29	94.36	6.28	52.00	93.00	96.00	98.00	99.90	
hr	1885	0.63	85.38	19.95	18.00	71.00	83.00	97.00	230.00	
curr_height	1413	0.72	66.80	4.19	56.00	64.00	67.00	70.00	81.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
vbg_temp	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
abg_temp	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
vbg_o2sat	4489	0.12	65.00	21.60	12.50	49.23	67.00	83.00	99.10	
abg_o2sat	4048	0.21	89.33	16.90	14.00	91.00	96.00	98.00	99.90	
sao2_blood	4863	0.05	79.36	24.56	11.00	67.00	90.50	97.00	99.90	
value_prev_weight	4670	0.09	197.18	59.63	82.00	154.00	190.00	232.00	398.00	
value_prev_height	4450	0.13	66.87	3.97	56.00	64.00	67.00	70.00	76.00	
value_prev_bmi	4779	0.06	30.14	8.17	15.00	24.00	29.00	36.00	49.00	
value_highest_20198	3916	0.23	48.02	20.49	14.00	37.00	43.60	52.00	198.00	
value_highest_115576	4640	0.09	47.43	15.89	20.00	38.00	44.00	52.25	139.00	
value_highest_327718	4890	0.04	68.05	52.70	22.00	39.00	45.00	68.00	249.00	
vbg_co2	3679	0.28	45.84	13.52	13.00	38.00	44.00	50.00	140.20	
highest_vbg_co2	3677	0.28	49.08	16.01	17.00	40.20	46.00	53.00	146.00	
pco2_nos	4564	0.11	41.65	11.46	14.60	34.00	41.00	47.00	98.00	
highest_pco2_nos	4563	0.11	47.79	15.57	20.00	38.38	45.00	53.00	139.00	
abg_ph	3295	0.36	7.37	0.11	6.83	7.32	7.38	7.43	7.74	
vbg_ph	3593	0.30	7.35	0.10	6.76	7.31	7.36	7.41	7.66	
abg_hco3	3977	0.22	23.47	5.93	4.00	20.20	23.00	26.00	51.80	
vbg_hco3	3843	0.25	24.83	6.10	3.10	22.00	25.00	28.00	56.00	
sodium	286	0.94	137.07	4.64	101.00	135.00	138.00	140.00	180.00	
serum_cr	454	0.91	1.44	1.74	0.20	0.76	0.97	1.32	19.40	
hgb	539	0.89	12.35	2.61	3.30	10.70	12.60	14.20	23.60	
serum_hco3	307	0.94	23.88	4.78	2.70	21.00	24.00	27.00	50.30	
serum_cl	319	0.94	102.00	5.73	66.00	99.00	103.00	106.00	134.00	
serum_lac	3081	0.40	2.13	2.19	0.30	1.00	1.50	2.30	24.00	
serum_k	397	0.92	4.08	0.66	1.80	3.70	4.00	4.40	8.90	
temp_cor_oxygen	5007	0.02	52.14	14.21	31.00	42.00	51.50	61.25	95.00	
vbg_ph_temp_cor	5011	0.02	7.35	0.09	6.96	7.30	7.36	7.41	7.55	
vbg_po2	3853	0.25	44.35	24.87	14.00	30.00	38.70	51.43	284.00	
vbg_lactate	4931	0.04	2.18	2.10	0.11	1.08	1.60	2.50	15.89	
vbg_hco3_calc	4965	0.03	25.93	6.87	8.00	22.00	26.00	30.00	47.00	
abg_po2	3966	0.22	135.52	93.25	20.00	73.50	101.00	170.00	495.00	
abg_po2_temp_cor	4886	0.04	110.65	98.12	20.00	41.10	82.00	137.00	497.00	
abg_ph_temp_cor	4911	0.04	7.37	0.09	7.02	7.32	7.37	7.43	7.63	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
abg_lactate	4943	0.03	2.01	2.13	0.42	0.80	1.20	2.52	18.00	
ph_blood	4618	0.10	7.31	0.15	6.88	7.28	7.36	7.41	7.60	
po2_blood	4654	0.09	76.95	70.74	11.00	36.60	55.00	85.40	481.00	
wbc	879	0.83	10.43	5.94	0.10	6.80	9.16	12.60	116.70	
plt	415	0.92	247.31	105.95	2.80	180.00	234.60	295.07	1159.00	
bnp	4161	0.19	966.36	3932.29	2.00	60.00	195.00	685.75	80400.00	
serum_phos	2715	0.47	3.66	1.41	0.80	2.80	3.40	4.10	14.10	
serum_ca	513	0.90	8.92	0.83	4.50	8.50	9.00	9.40	15.80	
serum_albumin	1834	0.64	3.57	0.72	1.10	3.10	3.70	4.10	5.60	
serum_tprot	1923	0.62	6.77	1.01	2.70	6.20	6.90	7.40	10.10	
has_j9612	0	1.00	0.00	0.07	0.00	0.00	0.00	0.00	1.00	
has_j9622	0	1.00	0.02	0.14	0.00	0.00	0.00	0.00	1.00	
has_j9602	0	1.00	0.03	0.18	0.00	0.00	0.00	0.00	1.00	
has_j9692	0	1.00	0.01	0.08	0.00	0.00	0.00	0.00	1.00	
ohs_code	0	1.00	0.01	0.09	0.00	0.00	0.00	0.00	1.00	
has_j9600	0	1.00	0.03	0.17	0.00	0.00	0.00	0.00	1.00	
has_j9601	0	1.00	0.19	0.39	0.00	0.00	0.00	0.00	1.00	
has_j961	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
has_j9610	0	1.00	0.01	0.08	0.00	0.00	0.00	0.00	1.00	
has_j9611	0	1.00	0.02	0.14	0.00	0.00	0.00	0.00	1.00	
has_j962	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
has_j9620	0	1.00	0.01	0.07	0.00	0.00	0.00	0.00	1.00	
has_j9621	0	1.00	0.04	0.21	0.00	0.00	0.00	0.00	1.00	
has_j9690	0	1.00	0.02	0.15	0.00	0.00	0.00	0.00	1.00	
has_j9691	0	1.00	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
other_abn_of_br	0	1.00	0.02	0.13	0.00	0.00	0.00	0.00	1.00	
cfdo	0	1.00	0.00	0.04	0.00	0.00	0.00	0.00	1.00	
has_i50_acute	0	1.00	0.09	0.28	0.00	0.00	0.00	0.00	1.00	
acute_nmd	0	1.00	0.00	0.04	0.00	0.00	0.00	0.00	1.00	
sepsis_dx	0	1.00	0.12	0.32	0.00	0.00	0.00	0.00	1.00	
stupor_dx	0	1.00	0.02	0.15	0.00	0.00	0.00	0.00	1.00	
cog_signs_dx	0	1.00	0.10	0.30	0.00	0.00	0.00	0.00	1.00	
mal_fat_dx	0	1.00	0.10	0.30	0.00	0.00	0.00	0.00	1.00	
resp_acid_dx	0	1.00	0.01	0.07	0.00	0.00	0.00	0.00	1.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
sleep_hypovent_dx	0	1.00	0.00	0.03	0.00	0.00	0.00	0.00	1.00	
cchs_dx	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
other_sleep_hypovent_dx	0	1.00	0.00	0.03	0.00	0.00	0.00	0.00	1.00	
acidosis_unspec	0	1.00	0.03	0.17	0.00	0.00	0.00	0.00	1.00	
headache_dx	0	1.00	0.03	0.18	0.00	0.00	0.00	0.00	1.00	
cpap	0	1.00	0.07	0.26	0.00	0.00	0.00	0.00	1.00	
tte_proc	0	1.00	0.12	0.33	0.00	0.00	0.00	0.00	1.00	
aero	0	1.00	0.12	0.33	0.00	0.00	0.00	0.00	1.00	
inh_teaching	0	1.00	0.03	0.16	0.00	0.00	0.00	0.00	1.00	
cxr1v	0	1.00	0.27	0.44	0.00	0.00	0.00	1.00	1.00	
cxr2v	0	1.00	0.03	0.17	0.00	0.00	0.00	0.00	1.00	
ctcnoncon	0	1.00	0.03	0.17	0.00	0.00	0.00	0.00	1.00	
ctccon	0	1.00	0.04	0.20	0.00	0.00	0.00	0.00	1.00	
cc_time	0	1.00	0.22	0.42	0.00	0.00	0.00	0.00	1.00	
meas_venous_o2_proc	0	1.00	0.01	0.12	0.00	0.00	0.00	0.00	1.00	
meas_arterial_gas_proc	0	1.00	0.01	0.09	0.00	0.00	0.00	0.00	1.00	
blood_cx_proc	0	1.00	0.14	0.35	0.00	0.00	0.00	0.00	1.00	
art_punct_proc	0	1.00	0.06	0.23	0.00	0.00	0.00	0.00	1.00	
ctabdpelv	0	1.00	0.05	0.22	0.00	0.00	0.00	0.00	1.00	
osa	0	1.00	0.17	0.38	0.00	0.00	0.00	0.00	1.00	
asthma	0	1.00	0.13	0.34	0.00	0.00	0.00	0.00	1.00	
copd	0	1.00	0.19	0.39	0.00	0.00	0.00	0.00	1.00	
chf	0	1.00	0.19	0.39	0.00	0.00	0.00	0.00	1.00	
stroke	0	1.00	0.07	0.25	0.00	0.00	0.00	0.00	1.00	
ckd	0	1.00	0.19	0.39	0.00	0.00	0.00	0.00	1.00	
pvd	0	1.00	0.09	0.29	0.00	0.00	0.00	0.00	1.00	
oud	0	1.00	0.05	0.22	0.00	0.00	0.00	0.00	1.00	
sedatives	0	1.00	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
phtn	0	1.00	0.08	0.27	0.00	0.00	0.00	0.00	1.00	
polycy	0	1.00	0.01	0.10	0.00	0.00	0.00	0.00	1.00	
po_steroid	0	1.00	0.45	0.50	0.00	0.00	0.00	1.00	1.00	
narcan	0	1.00	0.20	0.40	0.00	0.00	0.00	0.00	1.00	
inpt_inh	0	1.00	0.41	0.49	0.00	0.00	0.00	1.00	1.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
vasodilators	0	1.00	0.00	0.07	0.00	0.00	0.00	0.00	1.00	
ip_diuretics	0	1.00	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
ip_abx	0	1.00	0.02	0.14	0.00	0.00	0.00	0.00	1.00	
paralytic	0	1.00	0.01	0.10	0.00	0.00	0.00	0.00	1.00	
op_diuretics	0	1.00	0.37	0.48	0.00	0.00	0.00	1.00	1.00	
op_opiate	0	1.00	0.62	0.48	0.00	0.00	1.00	1.00	1.00	
op_mat	0	1.00	0.04	0.20	0.00	0.00	0.00	0.00	1.00	
op_nrt	0	1.00	0.12	0.32	0.00	0.00	0.00	0.00	1.00	
copd_med	0	1.00	0.48	0.50	0.00	0.00	0.00	1.00	1.00	
muscle_relax	0	1.00	0.23	0.42	0.00	0.00	0.00	0.00	1.00	
ABG_rfs	0	1.00	0.25	0.43	0.00	0.00	0.00	1.00	1.00	
is_amb	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
age_by_ten	0	1.00	5.86	1.79	1.80	4.60	6.10	7.30	9.00	
age_decade	0	1.00	4.32	1.78	1.00	3.00	5.00	6.00	7.00	
death_date	4390	0.14	-87.49	800.52	-10049.00	-12.00	4.00	59.00	444.00	
died	0	1.00	0.14	0.35	0.00	0.00	0.00	0.00	1.00	
months_death_or_cens	4420	0.14	1.91	2.75	0.00	0.00	1.00	3.00	15.00	
curr_weight	1625	0.68	197.88	62.02	80.00	151.83	189.60	236.20	444.80	
curr_weight_date	1625	0.68	-0.06	6.53	-93.00	0.00	0.00	0.00	59.00	
prev_weight_date	4670	0.09	-7.05	12.92	-93.00	-6.00	-2.00	-1.00	-1.00	
curr_height_date	1413	0.72	-0.37	5.93	-93.00	0.00	0.00	0.00	49.00	
prev_height_date	4450	0.13	-5.56	10.34	-93.00	-5.00	-2.00	-1.00	-1.00	
curr_bmi_date	2862	0.44	-0.52	6.45	-93.00	0.00	0.00	0.00	59.00	
prev_bmi_date	4779	0.06	-6.84	12.50	-93.00	-5.00	-2.00	-1.00	-1.00	
height	1410	0.72	66.81	4.19	56.00	64.00	67.00	70.00	81.00	
height_date	1410	0.72	-0.40	6.03	-93.00	0.00	0.00	0.00	49.00	
weight	1621	0.68	197.94	62.02	80.00	152.00	190.00	236.00	445.00	
weight_date	1621	0.68	-0.10	6.63	-93.00	0.00	0.00	0.00	59.00	
calc_bmi	2128	0.58	31.59	9.14	11.90	24.70	29.90	37.90	75.40	
calc_bmi_date	2128	0.58	-0.47	6.43	-93.00	0.00	0.00	0.00	49.00	
working_bmi	2859	0.44	31.40	8.51	11.50	24.70	29.95	38.80	49.80	
working_bmi_date	2859	0.44	-0.57	6.59	-93.00	0.00	0.00	0.00	59.00	
bmi	1738	0.66	31.63	9.10	11.50	24.70	29.90	38.10	75.40	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
bmi_date	1738	0.66	-0.44	6.44	-93.00	0.00	0.00	0.00	59.00	
bmi_int	1738	0.66	31.68	9.11	12.00	25.00	30.00	38.00	75.00	
bmi_by_five	1738	0.66	6.33	1.82	2.30	4.94	5.98	7.62	15.08	
rr	2796	0.45	18.09	4.97	3.00	16.00	18.00	20.00	56.00	
rr_date	2786	0.45	-0.80	3.73	-70.00	0.00	0.00	0.00	20.00	
temp_new	2488	0.51	97.87	1.77	47.00	97.38	98.00	98.43	105.80	
new_temp_date	2488	0.51	-0.81	4.48	-70.00	0.00	0.00	0.00	27.00	
sbp	1585	0.69	127.86	25.99	34.00	112.00	127.00	143.00	278.00	
sbp_date	1583	0.69	0.23	6.83	-93.00	0.00	0.00	0.00	64.00	
dbp	1601	0.69	72.45	16.05	21.00	62.00	72.00	83.00	163.00	
dbp_date	1594	0.69	0.22	6.83	-93.00	0.00	0.00	0.00	64.00	
spo2_date	3620	0.29	-0.46	2.98	-52.00	0.00	0.00	0.00	20.00	
hr_date	1885	0.63	-0.65	4.33	-70.00	0.00	0.00	0.00	40.00	
abg_ph_date	4012	0.22	0.11	1.79	-7.00	0.00	0.00	0.00	31.00	
vbg_ph_date	3726	0.27	0.18	3.96	-46.00	0.00	0.00	0.00	63.00	
abg_hco3_date	3955	0.23	0.13	2.80	-46.00	0.00	0.00	0.00	59.00	
abg_hco3_int	3977	0.22	23.49	5.94	4.00	20.00	23.00	26.00	52.00	
vbg_hco3_date	3814	0.25	0.21	3.71	-18.00	0.00	0.00	0.00	63.00	
sodium_date	285	0.94	-0.53	2.66	-52.00	0.00	0.00	0.00	53.00	
serum_k_date	397	0.92	-0.53	2.69	-52.00	0.00	0.00	0.00	53.00	
hgb_date	432	0.92	-0.49	2.65	-52.00	0.00	0.00	0.00	53.00	
wbc_date	781	0.85	-0.51	2.33	-32.00	0.00	0.00	0.00	40.00	
plt_date	359	0.93	-0.52	2.58	-52.00	0.00	0.00	0.00	40.00	
serum_hco3_date	302	0.94	-0.52	2.67	-52.00	0.00	0.00	0.00	53.00	
any_bicarb	255	0.95	24.04	4.77	2.70	21.40	24.00	26.60	56.00	
int_bicarb	255	0.95	24.08	4.77	3.00	21.00	24.00	27.00	56.00	
hco3_cat	307	0.94	0.89	0.98	0.00	0.00	1.00	2.00	3.00	
serum_cl_date	317	0.94	-0.53	2.67	-52.00	0.00	0.00	0.00	53.00	
serum_cr_date	422	0.92	-0.50	2.75	-52.00	0.00	0.00	0.00	53.00	
serum_lac_date	3048	0.40	-0.16	2.92	-31.00	0.00	0.00	0.00	38.00	
vbg_co2_date	3676	0.28	0.09	3.62	-46.00	0.00	0.00	0.00	63.00	
pco2_nos_date	4563	0.11	0.39	2.85	-17.00	0.00	0.00	0.00	27.00	
highest_vbg_co2_date	3676	0.28	0.74	5.19	-46.00	0.00	0.00	0.00	73.00	
highest_pco2_nos_date	4563	0.11	1.38	4.86	-16.00	0.00	0.00	0.00	44.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
paco2	3321	0.35	43.13	18.47	13.40	34.00	39.90	46.98	235.00	
paco2_date_1	3916	0.23	0.13	2.27	-7.00	0.00	0.00	0.00	63.00	
paco2_date_2	4640	0.09	0.31	2.33	-16.00	0.00	0.00	0.00	26.00	
paco2_date_3	4890	0.04	0.11	1.41	-6.00	0.00	0.00	0.00	16.00	
paco2_date	3318	0.35	0.15	2.15	-16.00	0.00	0.00	0.00	63.00	
paco2_int	3321	0.35	43.14	18.46	13.00	34.00	40.00	47.00	235.00	
highest_paco2	3318	0.35	50.59	26.86	14.00	37.90	44.00	53.00	249.00	
paco2_date_highest_1	3916	0.23	1.36	5.02	-5.00	0.00	0.00	0.00	79.00	
paco2_date_highest_2	4640	0.09	1.01	4.19	-16.00	0.00	0.00	0.00	44.00	
paco2_date_highest_3	4890	0.04	1.10	5.55	-6.00	0.00	0.00	0.00	73.00	
paco2_date_highest	3318	0.35	1.23	4.94	-16.00	0.00	0.00	0.00	79.00	
temp_cor_oxygen_date	4988	0.02	-0.01	3.03	-18.00	0.00	0.00	0.00	15.00	
temp_cor_vbg_ph_date	5011	0.02	-0.26	2.00	-11.00	0.00	0.00	0.00	4.00	
vbg_po2_date	3841	0.25	0.21	3.99	-46.00	0.00	0.00	0.00	63.00	
vbg_lactate_date	4930	0.04	0.69	5.89	-17.00	0.00	0.00	0.00	47.00	
vbg_hco3_calc_date	4963	0.03	0.32	2.95	-12.00	0.00	0.00	0.00	24.00	
abg_po2_date	3964	0.22	0.13	2.46	-20.00	0.00	0.00	0.00	63.00	
abg_po2_temp_cor_date	4884	0.04	0.10	1.01	-6.00	0.00	0.00	0.00	12.00	
abg_ph_temp_cor_date	4911	0.04	0.11	1.08	-6.00	0.00	0.00	0.00	12.00	
abg_lactate_date	4943	0.03	-0.08	1.94	-20.00	0.00	0.00	0.00	8.00	
ph_blood_date	4618	0.10	0.47	4.15	-16.00	0.00	0.00	0.00	74.00	
po2_blood_date	4649	0.09	0.31	2.35	-16.00	0.00	0.00	0.00	26.00	
vbg_temp_date	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
abg_temp_date	5097	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
vbg_o2sat_date	4488	0.12	0.04	3.07	-17.00	0.00	0.00	0.00	38.00	
abg_o2sat_date	3906	0.24	0.16	1.72	-7.00	0.00	0.00	0.00	31.00	
sao2_blood_date	4815	0.06	0.68	5.42	-31.00	0.00	0.00	1.00	46.00	
paco2_flag	3321	0.35	0.30	0.46	0.00	0.00	0.00	1.00	1.00	
highest_paco2_flag	3318	0.35	0.48	0.50	0.00	0.00	0.00	1.00	1.00	
paco2_52_flag	3321	0.35	0.16	0.37	0.00	0.00	0.00	0.00	1.00	
vbg_co2_flag	3679	0.28	0.28	0.45	0.00	0.00	0.00	1.00	1.00	
highest_vbg_co2_flag	3677	0.28	0.37	0.48	0.00	0.00	0.00	1.00	1.00	
miss_paco2_flag	0	1.00	0.11	0.31	0.00	0.00	0.00	0.00	1.00	
miss_vbg_co2_flag	0	1.00	0.08	0.27	0.00	0.00	0.00	0.00	1.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
miss_vbg_or_abg_co2_flag	0	1.00	0.16	0.37	0.00	0.00	0.00	0.00	1.00	
hco3_flag	307	0.94	0.25	0.43	0.00	0.00	0.00	1.00	1.00	
not_paco2_flag	3321	0.35	0.70	0.46	0.00	0.00	1.00	1.00	1.00	
not_hco3_flag	307	0.94	0.75	0.43	0.00	0.00	1.00	1.00	1.00	
k_cat	397	0.92	1.20	0.74	0.00	1.00	1.00	2.00	3.00	
acidemia	2357	0.54	0.32	0.47	0.00	0.00	0.00	1.00	1.00	
abg_sbe	4036	0.21	-1.97	6.60	-28.03	-5.42	-1.93	1.35	26.35	
vbg_sbe	3855	0.25	-0.93	6.84	-28.18	-4.13	-0.60	2.56	29.45	
cw_simple_acute_resp_acid	4295	0.16	0.08	0.27	0.00	0.00	0.00	0.00	1.00	
paco2_52_comp_flag	3321	0.35	0.04	0.20	0.00	0.00	0.00	0.00	1.00	
po_steroid_date	2832	0.45	1.28	6.61	-97.00	0.00	0.00	1.00	76.00	
narcan_date	4106	0.20	2.03	7.32	-109.00	0.00	0.00	3.00	70.00	
inpt_inh_date	2995	0.41	0.65	8.32	-233.00	0.00	0.00	1.00	74.00	
vasodilators_date	5086	0.00	1.28	5.26	-9.00	0.00	0.00	1.00	22.00	
ip_diuretics_date	5052	0.01	2.42	4.07	-12.00	0.00	2.00	5.00	12.00	
ip_abx_date	5014	0.02	0.74	3.80	-13.00	0.00	0.00	1.00	17.00	
paralytic_date	5064	0.01	1.70	4.58	-7.00	0.00	0.00	1.00	18.00	
inpt_inh_0	0	1.00	0.22	0.41	0.00	0.00	0.00	0.00	1.00	
ip_abx_0	0	1.00	0.01	0.10	0.00	0.00	0.00	0.00	1.00	
ip_diuretics_0	0	1.00	0.00	0.06	0.00	0.00	0.00	0.00	1.00	
narcan_0	0	1.00	0.10	0.30	0.00	0.00	0.00	0.00	1.00	
paralytic_0	0	1.00	0.01	0.08	0.00	0.00	0.00	0.00	1.00	
po_steroid_0	0	1.00	0.25	0.43	0.00	0.00	0.00	0.00	1.00	
vasodilators_0	0	1.00	0.00	0.05	0.00	0.00	0.00	0.00	1.00	
op_diuretics_first_date	3229	0.37	-1623.96	1422.77	-8424.00	-2400.50	-1328.50	-461.25	0.00	
op_diuretics_last_date	3234	0.37	-1620.49	1420.93	-8424.00	-2387.00	-1324.00	-461.00	0.00	
op_diuretics_365d	0	1.00	0.08	0.27	0.00	0.00	0.00	0.00	1.00	
op_opiate_first_date	1908	0.63	-1676.65	1513.12	-8863.00	-2613.50	-1373.00	-352.50	0.00	
op_opiate_last_date	1924	0.62	-1683.10	1511.90	-8863.00	-2618.00	-1376.00	-360.00	0.00	
op_opiate_365d	0	1.00	0.16	0.36	0.00	0.00	0.00	0.00	1.00	
op_mat_first_date	4900	0.04	-1301.62	1231.52	-6507.00	-1762.50	-966.00	-371.00	0.00	
op_mat_last_date	4901	0.04	-1304.41	1232.42	-6507.00	-1760.50	-966.00	-377.50	0.00	
op_mat_365d	0	1.00	0.01	0.10	0.00	0.00	0.00	0.00	1.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
op_nrt_first_date	4518	0.12	-1703.58	1497.27	-10350.00	-2493.00	-1357.00	-596.00	0.00	
op_nrt_last_date	4520	0.12	-1706.45	1496.14	-10350.00	-2493.50	-1356.00	-606.50	0.00	
op_nrt_365d	0	1.00	0.02	0.14	0.00	0.00	0.00	0.00	1.00	
copd_med_first_date	2659	0.48	-1662.43	1425.45	-9770.00	-2484.75	-1387.00	-508.75	0.00	
copd_med_last_date	2665	0.48	-1659.43	1423.35	-9770.00	-2471.75	-1388.00	-507.25	0.00	
copd_med_365d	0	1.00	0.10	0.30	0.00	0.00	0.00	0.00	1.00	
muscle_relax_first_date	3916	0.23	-1562.24	1366.05	-9679.00	-2339.00	-1285.00	-468.50	0.00	
muscle_relax_last_date	3919	0.23	-1562.49	1365.84	-9679.00	-2341.75	-1284.00	-473.50	0.00	
muscle_relax_365d	0	1.00	0.05	0.22	0.00	0.00	0.00	0.00	1.00	
tte_proc_first_date	4495	0.12	0.60	4.24	-45.00	0.00	1.00	2.00	26.00	
tte_proc_last_date	4495	0.12	2.18	6.98	-45.00	0.00	1.00	2.00	62.00	
vent_proc	0	1.00	0.16	0.37	0.00	0.00	0.00	0.00	1.00	
niv_proc	0	1.00	0.07	0.25	0.00	0.00	0.00	0.00	1.00	
imv_proc	0	1.00	0.11	0.31	0.00	0.00	0.00	0.00	1.00	
cpap_first_date	4753	0.07	1.37	4.17	-7.00	0.00	0.00	1.00	38.00	
cpap_last_date	4753	0.07	4.71	7.37	-6.00	0.00	2.00	6.00	49.00	
niv_proc_first_date	4765	0.07	1.36	4.77	-13.00	0.00	0.00	1.00	38.00	
niv_proc_last_date	4765	0.07	1.42	4.81	-13.00	0.00	0.00	1.00	38.00	
imv_proc_first_date	4551	0.11	0.93	5.24	-45.00	0.00	0.00	0.00	61.00	
imv_proc_last_date	4551	0.11	3.74	8.74	-45.00	0.00	0.50	4.00	92.00	
vent_proc_first_date	4277	0.16	0.74	4.44	-45.00	0.00	0.00	0.00	61.00	
vent_proc_last_date	4277	0.16	2.86	7.58	-45.00	0.00	0.00	3.00	92.00	
aero_first_date	4486	0.12	0.63	4.91	-46.00	0.00	0.00	1.00	76.00	
aero_last_date	4486	0.12	5.57	9.93	-18.00	0.00	3.00	7.00	103.00	
inh_teaching_first_date	4974	0.03	2.82	8.82	-45.00	0.00	1.00	3.00	81.00	
inh_teaching_last_date	4974	0.03	6.02	11.31	-2.00	1.00	2.00	7.00	102.00	
cxr1v_first_date	3744	0.27	-0.04	3.47	-46.00	0.00	0.00	0.00	38.00	
cxr1v_last_date	3744	0.27	2.96	7.65	-11.00	0.00	0.00	3.00	86.00	
cxr2v_first_date	4967	0.03	1.83	8.25	-20.00	0.00	0.00	2.00	59.00	
cxr2v_last_date	4967	0.03	2.72	8.08	-13.00	0.00	0.00	3.25	59.00	
ctcnoncon_first_date	4967	0.03	1.83	8.25	-20.00	0.00	0.00	2.00	59.00	
ctcnoncon_last_date	4967	0.03	2.72	8.08	-13.00	0.00	0.00	3.25	59.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
ctccon_first_date	4892	0.04	1.01	5.09	-14.00	0.00	0.00	0.00	33.00	
ctccon_last_date	4892	0.04	1.55	5.61	-10.00	0.00	0.00	1.00	38.00	
ctabdpelv_first_date	4843	0.05	1.73	8.20	-15.00	0.00	0.00	1.00	87.00	
ctabdpelv_last_date	4843	0.05	2.55	9.66	-15.00	0.00	0.00	1.00	87.00	
meas_venous_o2_proc_first_date	5036	0.01	-0.08	1.53	-9.00	0.00	0.00	0.00	4.00	
meas_venous_o2_proc_last_date	5036	0.01	2.01	5.46	-2.00	0.00	0.00	1.00	36.00	
meas_art_gas_proc_first_date	5066	0.01	0.44	1.18	0.00	0.00	0.00	0.00	5.00	
meas_art_gas_proc_last_date	5066	0.01	3.56	6.69	0.00	0.00	1.00	4.00	36.00	
blood_cx_proc_first_date	4403	0.14	0.09	4.10	-31.00	0.00	0.00	0.00	36.00	
blood_cx_proc_last_date	4403	0.14	2.15	8.05	-19.00	0.00	0.00	2.00	115.00	
art_punct_proc_first_date	4820	0.06	0.73	3.44	-7.00	0.00	0.00	0.00	31.00	
art_punct_proc_last_date	4820	0.06	2.81	7.61	-5.00	0.00	0.00	2.00	75.00	
cc_time_first_date	3977	0.22	0.16	3.70	-46.00	0.00	0.00	0.00	31.00	
cc_time_last_date	3977	0.22	3.78	8.46	-13.00	0.00	1.00	4.00	79.00	
aero_0	0	1.00	0.07	0.25	0.00	0.00	0.00	0.00	1.00	
blood_cx_proc_0	0	1.00	0.10	0.30	0.00	0.00	0.00	0.00	1.00	
cc_time_0	0	1.00	0.16	0.37	0.00	0.00	0.00	0.00	1.00	
cpap_0	0	1.00	0.04	0.20	0.00	0.00	0.00	0.00	1.00	
ctabdpelv_0	0	1.00	0.03	0.18	0.00	0.00	0.00	0.00	1.00	
ctccon_0	0	1.00	0.03	0.17	0.00	0.00	0.00	0.00	1.00	
ctcnoncon_0	0	1.00	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
cxr1v_0	0	1.00	0.21	0.40	0.00	0.00	0.00	0.00	1.00	
cxr2v_0	0	1.00	0.01	0.11	0.00	0.00	0.00	0.00	1.00	
imv_proc_0	0	1.00	0.08	0.28	0.00	0.00	0.00	0.00	1.00	
inh_teaching_0	0	1.00	0.01	0.09	0.00	0.00	0.00	0.00	1.00	
niv_proc_0	0	1.00	0.04	0.20	0.00	0.00	0.00	0.00	1.00	
tte_proc_0	0	1.00	0.04	0.20	0.00	0.00	0.00	0.00	1.00	
vent_proc_0	0	1.00	0.12	0.33	0.00	0.00	0.00	0.00	1.00	
aero_dur	4486	0.12	4.94	9.04	0.00	0.00	2.00	6.00	74.00	
blood_cx_proc_dur	4403	0.14	2.06	6.88	0.00	0.00	0.00	1.00	84.00	
cpap_dur	4753	0.07	3.34	5.81	0.00	0.00	1.00	4.00	50.00	
ctabdpelv_dur	4843	0.05	0.82	3.68	0.00	0.00	0.00	0.00	35.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
ctccon_dur	4892	0.04	0.54	2.99	0.00	0.00	0.00	0.00	32.00	
ctcnoncon_dur	4967	0.03	0.89	3.61	0.00	0.00	0.00	0.00	30.00	
cxr1v_dur	3744	0.27	3.00	7.65	0.00	0.00	0.00	3.00	81.00	
cxr2v_dur	4967	0.03	0.89	3.61	0.00	0.00	0.00	0.00	30.00	
imv_proc_dur	4551	0.11	2.81	6.67	0.00	0.00	0.00	2.00	58.00	
inh_teaching_dur	4974	0.03	3.20	8.52	0.00	0.00	0.00	2.00	69.00	
niv_proc_dur	4765	0.07	0.07	0.72	0.00	0.00	0.00	0.00	11.00	
tte_proc_dur	4495	0.12	1.57	5.55	0.00	0.00	0.00	0.00	55.00	
hypercap_resp_failure	0	1.00	0.06	0.23	0.00	0.00	0.00	0.00	1.00	
j9612_date	5087	0.00	0.71	4.32	-7.00	-0.25	0.00	0.00	12.00	
j9612_pcpl	5087	0.00	0.17	0.38	0.00	0.00	0.00	0.00	1.00	
j9612_adm	5087	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
j9612_vr	5087	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
j9622_date	5008	0.02	0.48	5.61	-15.00	-1.00	0.00	0.00	46.00	
j9622_pcpl	5012	0.02	0.11	0.32	0.00	0.00	0.00	0.00	1.00	
j9622_adm	5008	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
j9622_vr	5008	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
j9602_date	4941	0.03	0.79	8.26	-52.00	0.00	0.00	0.00	58.00	
j9602_pcpl	4944	0.03	0.07	0.25	0.00	0.00	0.00	0.00	1.00	
j9602_adm	4941	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
j9602_vr	4941	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
j9692_date	5079	0.01	2.53	8.33	-7.00	0.00	0.00	1.00	37.00	
j9692_pcpl	5081	0.01	0.07	0.25	0.00	0.00	0.00	0.00	1.00	
j9692_adm	5079	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
j9692_vr	5079	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
ohs_code_date	5068	0.01	-0.07	2.81	-6.00	-1.00	0.00	0.00	13.00	
e662_pcpl	5068	0.01	0.14	0.35	0.00	0.00	0.00	0.00	1.00	
e662_adm	5068	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
e662_vr	5068	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
hypercap_resp_failure_date	4823	0.06	0.68	7.27	-52.00	0.00	0.00	0.00	58.00	
other_resp_failure	0	1.00	0.26	0.44	0.00	0.00	0.00	1.00	1.00	
j9600_date	4950	0.03	2.11	10.16	-66.00	0.00	0.00	1.00	66.00	
j9601_date	4147	0.19	0.33	6.53	-46.00	0.00	0.00	0.00	125.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
j961_date	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
j9610_date	5076	0.01	4.40	17.96	-6.00	0.00	0.00	0.00	98.00	
j9611_date	5006	0.02	1.17	8.14	-7.00	0.00	0.00	0.00	75.00	
j962_date	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
j9620_date	5085	0.01	3.38	10.06	-7.00	0.00	0.00	3.00	40.00	
j9621_date	4883	0.04	0.38	7.59	-15.00	-0.25	0.00	0.00	74.00	
j9690_date	4998	0.02	4.47	13.29	-45.00	0.00	0.00	5.00	66.00	
j9691_date	5046	0.01	3.29	8.83	-19.00	0.00	0.00	3.00	43.00	
other_resp_failure_date	3791	0.26	0.17	5.49	-66.00	0.00	0.00	0.00	98.00	
sepsis_dx_date	4514	0.12	0.40	8.18	-46.00	0.00	0.00	0.00	125.00	
stupor_dx_date	4993	0.02	0.83	5.10	-15.00	0.00	0.00	0.00	36.00	
cog_signs_dx_date	4613	0.10	0.87	6.95	-52.00	0.00	0.00	0.00	54.00	
mal_fat_dx_date	4606	0.10	0.91	7.80	-22.00	0.00	0.00	0.00	125.00	
resp_acid_dx_date	5084	0.01	0.67	4.17	-3.00	0.00	0.00	0.00	21.00	
sleep_hypovent_dx_date	5106	0.00	1.80	3.03	0.00	0.00	0.00	2.00	7.00	
cchs_dx_date	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
other_sleep_hypovent_dx_date	5107	0.00	3.25	6.50	0.00	0.00	0.00	3.25	13.00	
acidosis_unspec_date	4950	0.03	0.19	4.48	-11.00	0.00	0.00	0.00	43.00	
headache_dx_date	4944	0.03	-0.07	2.96	-18.00	0.00	0.00	0.00	24.00	
dysp_dx	0	1.00	0.14	0.34	0.00	0.00	0.00	0.00	1.00	
dysp_dx_date	4420	0.14	0.68	5.84	-21.00	0.00	0.00	0.00	71.00	
symp_obs	0	1.00	0.01	0.08	0.00	0.00	0.00	0.00	1.00	
symp_obs_date	5080	0.01	0.45	2.06	-1.00	0.00	0.00	0.00	11.00	
abn_br_dx	0	1.00	0.00	0.03	0.00	0.00	0.00	0.00	1.00	
abn_br_dx_date	5105	0.00	5.33	11.22	0.00	0.00	0.00	3.00	28.00	
resp_abnormality	0	1.00	0.02	0.14	0.00	0.00	0.00	0.00	1.00	
r0689_date	5024	0.02	0.32	5.45	-22.00	-1.00	0.00	0.00	26.00	
resp_abnormality_date	5010	0.02	0.49	5.47	-22.00	-1.00	0.00	0.00	26.00	
other_abn_of_br_date	5024	0.02	0.32	5.45	-22.00	-1.00	0.00	0.00	26.00	
fast_br	0	1.00	0.00	0.07	0.00	0.00	0.00	0.00	1.00	
fast_br_date	5089	0.00	5.32	11.85	-11.00	0.00	0.00	7.25	40.00	
pulm_edema_dx	0	1.00	0.03	0.18	0.00	0.00	0.00	0.00	1.00	
pulm_edema_dx_date	4934	0.03	0.55	16.69	-181.00	0.00	0.00	0.00	74.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
pna_dx	0	1.00	0.14	0.35	0.00	0.00	0.00	0.00	1.00	
pna_dx_date	4379	0.14	0.14	6.56	-46.00	0.00	0.00	0.00	125.00	
acute_chf	0	1.00	0.09	0.28	0.00	0.00	0.00	0.00	1.00	
acute_old	0	1.00	0.08	0.27	0.00	0.00	0.00	0.00	1.00	
acute_old_date	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
resp_dep_compl	0	1.00	0.04	0.18	0.00	0.00	0.00	0.00	1.00	
resp_dep_compl_date	4930	0.04	-0.06	6.12	-22.00	0.00	0.00	0.00	75.00	
acute_nmd_date	5103	0.00	-0.25	0.71	-2.00	0.00	0.00	0.00	0.00	
osa_first_date	4241	0.17	-1388.50	1043.78	-6313.00	-2063.25	-1222.50	-566.75	0.00	
osa_last_date	4242	0.17	-1368.92	1020.41	-6106.00	-2043.00	-1206.00	-564.00	0.00	
asthma_first_date	4433	0.13	-1521.82	1204.57	-12433.00	-2172.75	-1410.00	-659.00	0.00	
asthma_last_date	4434	0.13	-1511.81	1179.92	-12433.00	-2172.00	-1405.00	-659.00	0.00	
copd_first_date	4133	0.19	-1290.86	1052.24	-9568.00	-1952.00	-1203.00	-452.75	0.00	
copd_last_date	4135	0.19	-1075.46	940.56	-9568.00	-1642.25	-935.50	-304.50	0.00	
chf_first_date	4127	0.19	-948.05	855.50	-5954.00	-1474.50	-729.00	-231.50	0.00	
chf_last_date	4127	0.19	-941.67	849.40	-5954.00	-1470.25	-728.00	-229.75	0.00	
stroke_first_date	4757	0.07	-972.77	1099.61	-7453.00	-1496.25	-647.00	-130.25	0.00	
stroke_last_date	4757	0.07	-971.08	1098.99	-7453.00	-1494.00	-644.00	-129.25	0.00	
ckd_first_date	4145	0.19	-1155.13	938.82	-5617.00	-1809.50	-1002.00	-339.25	0.00	
ckd_last_date	4151	0.19	-1151.99	937.24	-5617.00	-1807.25	-994.00	-336.50	0.00	
ctd	0	1.00	0.05	0.22	0.00	0.00	0.00	0.00	1.00	
ctd_first_date	4859	0.05	-1444.06	1107.35	-6297.00	-2165.50	-1283.50	-560.25	0.00	
ctd_last_date	4859	0.05	-1281.33	1036.67	-6297.00	-1988.50	-1125.50	-378.25	0.00	
dem	0	1.00	0.06	0.24	0.00	0.00	0.00	0.00	1.00	
dem_first_date	4788	0.06	-912.31	988.30	-8434.00	-1377.50	-647.00	-213.00	0.00	
dem_last_date	4789	0.06	-768.76	841.94	-5826.00	-1137.75	-494.50	-132.25	0.00	
dm	0	1.00	0.29	0.45	0.00	0.00	0.00	1.00	1.00	
dm_first_date	3644	0.29	-1537.96	1385.41	-22671.00	-2239.00	-1440.00	-516.50	0.00	
dm_last_date	3645	0.29	-1265.86	1013.13	-11032.00	-1960.50	-1176.00	-403.50	0.00	
pvd_first_date	4629	0.09	-891.06	911.64	-8011.00	-1443.25	-651.50	-94.50	0.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
pvd_last_date	4629	0.09	-890.98	911.62	-8011.00	-1443.25	-651.50	-94.50	0.00	
oud_first_date	4844	0.05	-1174.78	821.76	-4689.00	-1738.00	-1118.00	-458.00	0.00	
oud_last_date	4845	0.05	-1177.76	819.85	-4689.00	-1748.50	-1126.00	-460.75	0.00	
sedatives_first_date	5054	0.01	-1083.11	987.45	-6232.00	-1494.00	-1081.00	-304.00	0.00	
sedatives_last_date	5054	0.01	-1082.79	987.62	-6232.00	-1494.00	-1081.00	-304.00	0.00	
cfdo_first_date	5102	0.00	-2121.56	2038.18	-6061.00	-2261.00	-2080.00	-988.00	-1.00	
cfdo_last_date	5102	0.00	-1540.22	1429.30	-4600.00	-2109.00	-992.00	-829.00	-1.00	
phtn_first_date	4719	0.08	-974.33	777.01	-4323.00	-1548.50	-822.00	-314.75	0.00	
phtn_last_date	4720	0.08	-973.61	778.02	-4323.00	-1552.00	-811.00	-312.50	0.00	
polycy_first_date	5055	0.01	-1190.86	791.80	-2720.00	-1814.25	-1182.50	-483.00	0.00	
polycy_last_date	5055	0.01	-1184.91	790.00	-2720.00	-1814.25	-1165.50	-483.00	0.00	
nmd	0	1.00	0.04	0.20	0.00	0.00	0.00	0.00	1.00	
nmd_first_date	4905	0.04	-1096.67	1377.90	-13004.00	-1805.75	-680.50	-162.25	0.00	
nmd_last_date	4905	0.04	-1002.59	1288.23	-13004.00	-1580.75	-619.50	-139.00	0.00	
nic	0	1.00	0.24	0.43	0.00	0.00	0.00	0.00	1.00	
nic_first_date	3879	0.24	-1350.87	1178.14	-18865.00	-1957.75	-1295.50	-548.25	0.00	
nic_last_date	3879	0.24	-1345.84	1171.64	-18865.00	-1957.00	-1294.50	-537.00	0.00	
ovs	0	1.00	0.07	0.25	0.00	0.00	0.00	0.00	1.00	
ats_ohs_flag	4544	0.11	0.39	0.49	0.00	0.00	0.00	1.00	1.00	
pos_ohs_flag	0	1.00	0.40	0.49	0.00	0.00	0.00	1.00	1.00	
ats_copd_flag	4566	0.11	0.09	0.29	0.00	0.00	0.00	0.00	1.00	
guidelines	4566	0.11	0.55	0.66	0.00	0.00	0.00	1.00	2.00	
OBESITY_rfs	0	1.00	0.13	0.34	0.00	0.00	0.00	0.00	1.00	
_merge_amb_obes	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
PREDISPOSITION_rfs	0	1.00	0.54	0.50	0.00	0.00	1.00	1.00	1.00	
_merge_amb_predisp	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
RESPFAIL_rfs	0	1.00	0.22	0.41	0.00	0.00	0.00	0.00	1.00	
_merge_amb_respfail	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
VBG_rfs	0	1.00	0.33	0.47	0.00	0.00	0.00	1.00	1.00	
_merge_amb_vbg	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
VENTSUPPORT_rfs	0	1.00	0.19	0.39	0.00	0.00	0.00	0.00	1.00	
_merge_amb_ventsupp	5111	0.00	NaN	NA	NA	NA	NA	NA	NA	
is_emer	0	1.00	0.34	0.47	0.00	0.00	0.00	1.00	1.00	
_merge_emer_obes	4633	0.09	1.75	0.58	1.00	1.00	2.00	2.00	5.00	
_merge_emer_predisp	3810	0.25	2.15	1.17	1.00	2.00	2.00	2.00	5.00	
_merge_emer_respfail	3764	0.26	1.37	1.11	1.00	1.00	1.00	1.00	5.00	
_merge_emer_vbg	3396	0.34	1.59	1.17	1.00	1.00	1.00	2.00	5.00	
_merge_emer_ventsupp	3357	0.34	1.26	0.95	1.00	1.00	1.00	1.00	5.00	
_merge_emer	3357	0.34	2.00	0.00	2.00	2.00	2.00	2.00	2.00	
is_inp	1754	0.66	1.00	0.00	1.00	1.00	1.00	1.00	1.00	
_merge_inpat_obes	3693	0.28	1.38	0.90	1.00	1.00	1.00	1.00	5.00	
_merge_inpat_predisp	2561	0.50	2.42	1.53	1.00	1.00	2.00	2.00	5.00	
_merge_inpat_respfail	2285	0.55	2.05	1.68	1.00	1.00	1.00	2.00	5.00	
_merge_inpat_vbg	1894	0.63	2.06	1.66	1.00	1.00	1.00	2.00	5.00	
_merge_inpat_ventsupp	1754	0.66	1.83	1.59	1.00	1.00	1.00	1.00	5.00	
_merge_inpat	0	1.00	1.66	0.47	1.00	1.00	2.00	2.00	2.00	
patient_id	0	1.00	363497.02	213165.52	34.00	174410.00	365388.00	538369.50	748967.00	
rfsgroup	0	1.00	3.36	1.65	1.00	2.00	3.00	4.00	7.00	
first_encounter	0	1.00	0.87	0.33	0.00	1.00	1.00	1.00	1.00	
has_abg	0	1.00	0.35	0.48	0.00	0.00	0.00	1.00	1.00	
has_vbg	0	1.00	0.28	0.45	0.00	0.00	0.00	1.00	1.00	
max_hco3_or_its_met_alk	3321	0.35	30.47	26.38	-12.00	17.43	25.86	35.96	304.57	
prim_met_alk	4566	0.11	0.01	0.10	0.00	0.00	0.00	0.00	1.00	
max_hco3_or_its_comb_met_alk	3321	0.35	27.25	7.39	15.36	23.60	25.96	28.79	104.00	
combo_met_alk	4566	0.11	0.12	0.33	0.00	0.00	0.00	0.00	1.00	
paco2_50_flag	3321	0.35	0.19	0.39	0.00	0.00	0.00	0.00	1.00	
hypercap_dx_adm_flag	4823	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
hypercap_dx_pcpl_flag	4829	0.06	0.09	0.29	0.00	0.00	0.00	0.00	1.00	
hypercap_dx_vr_flag	4823	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
hypercap_on_abg	0	1.00	0.11	0.31	0.00	0.00	0.00	0.00	1.00	
hypercap_on_vbg	0	1.00	0.08	0.27	0.00	0.00	0.00	0.00	1.00	
has_both_abg_vbg	0	1.00	0.10	0.31	0.00	0.00	0.00	0.00	1.00	
has_neither_abg_vbg	0	1.00	0.47	0.50	0.00	0.00	0.00	1.00	1.00	
vbg_or_abg_co2_flag	0	1.00	0.16	0.37	0.00	0.00	0.00	0.00	1.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
dx_hypercap_on_abg	4823	0.06	0.47	0.50	0.00	0.00	0.00	1.00	1.00	
sugg_hypercap_dx_on_vbg	4823	0.06	0.35	0.48	0.00	0.00	0.00	1.00	1.00	
dx_hypercap_on_vbg	4823	0.06	0.16	0.37	0.00	0.00	0.00	0.00	1.00	
vbg_o2sat_calc	0	1.00	65.23	10.55	12.50	65.08	65.08	65.08	100.16	
abg_o2sat_calc	0	1.00	90.94	8.10	14.00	91.05	91.05	91.05	99.90	
corr_vbg_co2	3679	0.28	40.41	13.51	7.45	32.62	38.62	44.85	134.83	
corr_vbg_co2_flag	3679	0.28	0.25	0.43	0.00	0.00	0.00	0.00	1.00	
corr_hypercap_on_vbg	0	1.00	0.07	0.25	0.00	0.00	0.00	0.00	1.00	
race_ethnicity	0	1.00	1.08	1.91	0.00	0.00	0.00	1.00	6.00	
has_vbg_and_cat	0	1.00	0.47	0.88	0.00	0.00	0.00	1.00	3.00	
has_bmi_and_cat	0	1.00	2.46	2.16	0.00	0.00	2.00	4.00	6.00	
has_weight_and_cat	0	1.00	2.33	1.87	0.00	0.00	3.00	4.00	6.00	
has_height_and_cat	0	1.00	2.22	1.65	0.00	0.00	2.00	4.00	6.00	
has_hr_and_cat	0	1.00	1.28	1.04	0.00	0.00	2.00	2.00	3.00	
has_sbp_and_cat	0	1.00	1.54	1.12	0.00	0.00	2.00	2.00	3.00	
has_rr_and_cat	0	1.00	1.02	1.18	0.00	0.00	0.00	2.00	3.00	
has_temp_and_cat	0	1.00	0.94	0.97	0.00	0.00	1.00	2.00	3.00	
has_spo2_and_cat	0	1.00	0.54	0.86	0.00	0.00	0.00	1.00	2.00	
has_cl_and_cat	0	1.00	1.91	0.74	0.00	2.00	2.00	2.00	3.00	
has_k_and_cat	0	1.00	1.99	0.94	0.00	2.00	2.00	3.00	4.00	
has_hco3_and_cat	0	1.00	2.35	1.24	0.00	2.00	2.00	3.00	5.00	
has_lactate_and_cat	0	1.00	0.52	0.70	0.00	0.00	0.00	1.00	2.00	
has_na_and_cat	0	1.00	1.62	0.61	0.00	1.00	2.00	2.00	3.00	
has_cr_and_cat	0	1.00	1.25	0.70	0.00	1.00	1.00	2.00	3.00	
has_hgb_and_cat	0	1.00	2.99	1.35	0.00	2.00	3.00	4.00	5.00	
has_wbc_and_cat	0	1.00	2.00	1.11	0.00	2.00	2.00	3.00	4.00	
has_plt_and_cat	0	1.00	2.64	0.91	0.00	3.00	3.00	3.00	4.00	
has_bnp_and_cat	0	1.00	0.34	0.80	0.00	0.00	0.00	0.00	3.00	
has_phos_and_cat	0	1.00	0.76	0.96	0.00	0.00	0.00	1.00	3.00	
has_ca_and_cat	0	1.00	1.58	0.71	0.00	1.00	2.00	2.00	3.00	
has_alb_and_cat	0	1.00	1.55	1.29	0.00	0.00	2.00	3.00	3.00	
has_tprot_and_cat	0	1.00	1.14	0.96	0.00	0.00	1.00	2.00	3.00	
encounter_type_dummy1	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
encounter_type_dummy2	0	1.00	0.34	0.47	0.00	0.00	0.00	1.00	1.00	
encounter_type_dummy3	0	1.00	0.66	0.47	0.00	0.00	1.00	1.00	1.00	
female	0	1.00	0.50	0.50	0.00	0.00	0.00	1.00	1.00	
male	0	1.00	0.50	0.50	0.00	0.00	1.00	1.00	1.00	
white_race	0	1.00	0.64	0.48	0.00	0.00	1.00	1.00	1.00	
black_race	0	1.00	0.18	0.39	0.00	0.00	0.00	0.00	1.00	
unknown_race	0	1.00	0.15	0.36	0.00	0.00	0.00	0.00	1.00	
asian_race	0	1.00	0.02	0.12	0.00	0.00	0.00	0.00	1.00	
nat_am_race	0	1.00	0.01	0.09	0.00	0.00	0.00	0.00	1.00	
nhpi_race	0	1.00	0.00	0.03	0.00	0.00	0.00	0.00	1.00	
not_hisp_eth	0	1.00	0.76	0.43	0.00	1.00	1.00	1.00	1.00	
hisp_eth	0	1.00	0.07	0.26	0.00	0.00	0.00	0.00	1.00	
unknown_eth	0	1.00	0.17	0.37	0.00	0.00	0.00	0.00	1.00	
location_dummy1	0	1.00	0.47	0.50	0.00	0.00	0.00	1.00	1.00	
location_dummy2	0	1.00	0.25	0.43	0.00	0.00	0.00	0.00	1.00	
location_dummy3	0	1.00	0.07	0.26	0.00	0.00	0.00	0.00	1.00	
location_dummy4	0	1.00	0.21	0.41	0.00	0.00	0.00	0.00	1.00	
ps	0	1.00	0.36	0.25	0.00	0.14	0.31	0.56	0.97	
tx_pr_decile	0	1.00	7.23	2.00	1.00	6.00	7.00	9.00	10.00	
sumofweights	0	1.00	1593981.00	0.00	1593981.00	1593981.00	1593981.00	1593981.00	1593981.00	
ipw	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
approx_ipw_fweight	0	1.00	12.47	14.41	6.00	7.00	9.00	13.00	410.00	
inp_ps	0	1.00	0.40	0.22	0.02	0.21	0.36	0.57	0.95	
inp_sumofweights	0	1.00	1419945.50	0.00	1419945.50	1419945.50	1419945.50	1419945.50	1419945.50	
inp_ipw	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
approx_inp_ipw_fweight	0	1.00	13.36	9.24	7.00	9.00	10.00	14.00	184.00	
vbg_ps	0	1.00	0.29	0.23	0.00	0.10	0.22	0.44	0.97	
vbg_tx_pr_decile	0	1.00	7.24	1.98	1.00	6.00	7.00	9.00	10.00	
vbg_sumofweights	0	1.00	1542183.62	0.00	1542183.62	1542183.62	1542183.62	1542183.62	1542183.62	
vbg_ipw	0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
vbg_approx_ipw_fweight	0	1.00	12.73	16.31	7.00	7.00	8.00	12.00	526.00	
died_1mo	30	0.99	0.06	0.23	0.00	0.00	0.00	0.00	1.00	
died_2mo	30	0.99	0.09	0.28	0.00	0.00	0.00	0.00	1.00	
death_time	4420	0.14	1.91	2.75	0.00	0.00	1.00	3.00	15.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
death_60d	0	1.00	0.11	0.31	0.00	0.00	0.00	0.00	1.00	
w_abg	0	1.00	1.00	0.56	0.60	0.68	0.79	1.06	3.78	
w_vbg	0	1.00	1.00	0.57	0.62	0.68	0.78	1.05	3.68	

```
# Variable-wise % missing
naniar::miss_var_summary(subset_data)
```

variable	n_miss	pct_miss
vbg_temp	5111	100
abg_temp	5111	100
vbg_temp_date	5111	100
j961_date	5111	100
j962_date	5111	100
cchs_dx_date	5111	100
_merge_amb_obes	5111	100
_merge_amb_predisp	5111	100
_merge_amb_respfail	5111	100
_merge_amb_vbg	5111	100
_merge_amb_ventsupp	5111	100
other_sleep_hypovent_dx_date	5107	99.9
sleep_hypovent_dx_date	5106	99.9
abn_br_dx_date	5105	99.9
acute_nmd_date	5103	99.8
cfdo_first_date	5102	99.8
cfdo_last_date	5102	99.8
abg_temp_date	5097	99.7
fast_br_date	5089	99.6
j9612_date	5087	99.5
j9612_pcpl	5087	99.5
j9612_adm	5087	99.5
j9612_vr	5087	99.5
vasodilators_date	5086	99.5

variable	n_miss	pct_miss
j9620_date	5085	99.5
resp_acid_dx_date	5084	99.5
j9692_pcpl	5081	99.4
symp_obs_date	5080	99.4
j9692_date	5079	99.4
j9692_adm	5079	99.4
j9692_vr	5079	99.4
j9610_date	5076	99.3
ohs_code_date	5068	99.2
e662_pcpl	5068	99.2
e662_adm	5068	99.2
e662_vr	5068	99.2
meas_art_gas_proc_first_date	5066	99.1
meas_art_gas_proc_last_date	5066	99.1
paralytic_date	5064	99.1
polycy_first_date	5055	98.9
polycy_last_date	5055	98.9
sedatives_first_date	5054	98.9
sedatives_last_date	5054	98.9
ip_diuretics_date	5052	98.8
j9691_date	5046	98.7
meas_venous_o2_proc_first_date	5036	98.5
meas_venous_o2_proc_last_date	5036	98.5
r0689_date	5024	98.3
other_abn_of_br_date	5024	98.3
ip_abx_date	5014	98.1
j9622_pcpl	5012	98.1
vbg_ph_temp_cor	5011	98.0
temp_cor_vbg_ph_date	5011	98.0
resp_abnormality_date	5010	98.0
j9622_date	5008	98.0
j9622_adm	5008	98.0
j9622_vr	5008	98.0
temp_cor_oxygen	5007	98.0
j9611_date	5006	97.9

variable	n_miss	pct_miss
j9690_date	4998	97.8
stupor_dx_date	4993	97.7
temp_cor_oxygen_date	4988	97.6
inh_teaching_first_date	4974	97.3
inh_teaching_last_date	4974	97.3
inh_teaching_dur	4974	97.3
cxr2v_first_date	4967	97.2
cxr2v_last_date	4967	97.2
ctcnoncon_first_date	4967	97.2
ctcnoncon_last_date	4967	97.2
ctcnoncon_dur	4967	97.2
cxr2v_dur	4967	97.2
vbg_hco3_calc	4965	97.1
vbg_hco3_calc_date	4963	97.1
j9600_date	4950	96.8
acidosis_unspec_date	4950	96.8
j9602_pcpl	4944	96.7
headache_dx_date	4944	96.7
abg_lactate	4943	96.7
abg_lactate_date	4943	96.7
j9602_date	4941	96.7
j9602_adm	4941	96.7
j9602_vr	4941	96.7
pulm_edema_dx_date	4934	96.5
vbg_lactate	4931	96.5
vbg_lactate_date	4930	96.5
resp_dep_compl_date	4930	96.5
abg_ph_temp_cor	4911	96.1
abg_ph_temp_cor_date	4911	96.1
nmd_first_date	4905	96.0
nmd_last_date	4905	96.0
op_mat_last_date	4901	95.9
op_mat_first_date	4900	95.9
ctccon_first_date	4892	95.7
ctccon_last_date	4892	95.7

variable	n_miss	pct_miss
ctccon_dur	4892	95.7
value_highest_327718	4890	95.7
paco2_date_3	4890	95.7
paco2_date_highest_3	4890	95.7
abg_po2_temp_cor	4886	95.6
abg_po2_temp_cor_date	4884	95.6
j9621_date	4883	95.5
sao2_blood	4863	95.1
ctd_first_date	4859	95.1
ctd_last_date	4859	95.1
oud_last_date	4845	94.8
oud_first_date	4844	94.8
ctabdpelv_first_date	4843	94.8
ctabdpelv_last_date	4843	94.8
ctabdpelv_dur	4843	94.8
hypercap_dx_pcpl_flag	4829	94.5
hypercap_resp_failure_date	4823	94.4
hypercap_dx_adm_flag	4823	94.4
hypercap_dx_vr_flag	4823	94.4
dx_hypercap_on_abg	4823	94.4
sugg_hypercap_dx_on_vbg	4823	94.4
dx_hypercap_on_vbg	4823	94.4
art_punct_proc_first_date	4820	94.3
art_punct_proc_last_date	4820	94.3
sao2_blood_date	4815	94.2
dem_last_date	4789	93.7
dem_first_date	4788	93.7
value_prev_bmi	4779	93.5
prev_bmi_date	4779	93.5
niv_proc_first_date	4765	93.2
niv_proc_last_date	4765	93.2
niv_proc_dur	4765	93.2
stroke_first_date	4757	93.1
stroke_last_date	4757	93.1
cpap_first_date	4753	93.0

variable	n_miss	pct_miss
cpap_last_date	4753	93.0
cpap_dur	4753	93.0
phtn_last_date	4720	92.3
phtn_first_date	4719	92.3
value_prev_weight	4670	91.4
prev_weight_date	4670	91.4
po2_blood	4654	91.1
po2_blood_date	4649	91.0
value_highest_115576	4640	90.8
paco2_date_2	4640	90.8
paco2_date_highest_2	4640	90.8
_merge_emer_obes	4633	90.6
pvd_first_date	4629	90.6
pvd_last_date	4629	90.6
ph_blood	4618	90.4
ph_blood_date	4618	90.4
cog_signs_dx_date	4613	90.3
mal_fat_dx_date	4606	90.1
ats_copd_flag	4566	89.3
guidelines	4566	89.3
prim_met_alk	4566	89.3
combo_met_alk	4566	89.3
pco2_nos	4564	89.3
highest_pco2_nos	4563	89.3
pco2_nos_date	4563	89.3
highest_pco2_nos_date	4563	89.3
imv_proc_first_date	4551	89.0
imv_proc_last_date	4551	89.0
imv_proc_dur	4551	89.0
ats_ohs_flag	4544	88.9
op_nrt_last_date	4520	88.4
op_nrt_first_date	4518	88.4
sepsis_dx_date	4514	88.3
tte_proc_first_date	4495	87.9
tte_proc_last_date	4495	87.9

variable	n_miss	pct_miss
tte_proc_dur	4495	87.9
vbg_o2sat	4489	87.8
vbg_o2sat_date	4488	87.8
aero_first_date	4486	87.8
aero_last_date	4486	87.8
aero_dur	4486	87.8
value_prev_height	4450	87.1
prev_height_date	4450	87.1
asthma_last_date	4434	86.8
asthma_first_date	4433	86.7
months_death_or_cens	4420	86.5
dysp_dx_date	4420	86.5
death_time	4420	86.5
blood_cx_proc_first_date	4403	86.1
blood_cx_proc_last_date	4403	86.1
blood_cx_proc_dur	4403	86.1
death_date	4390	85.9
death_abs	4390	85.9
pna_dx_date	4379	85.7
cw_simple_acute_resp_acid	4295	84.0
vent_proc_first_date	4277	83.7
vent_proc_last_date	4277	83.7
osa_last_date	4242	83.0
osa_first_date	4241	83.0
bnp	4161	81.4
ckd_last_date	4151	81.2
j9601_date	4147	81.1
ckd_first_date	4145	81.1
copd_last_date	4135	80.9
copd_first_date	4133	80.9
chf_first_date	4127	80.7
chf_last_date	4127	80.7
narcan_date	4106	80.3
abg_o2sat	4048	79.2
abg_sbe	4036	79.0

variable	n_miss	pct_miss
abg_ph_date	4012	78.5
abg_hco3	3977	77.8
abg_hco3_int	3977	77.8
cc_time_first_date	3977	77.8
cc_time_last_date	3977	77.8
abg_po2	3966	77.6
abg_po2_date	3964	77.6
abg_hco3_date	3955	77.4
muscle_relax_last_date	3919	76.7
value_highest_20198	3916	76.6
paco2_date_1	3916	76.6
paco2_date_highest_1	3916	76.6
muscle_relax_first_date	3916	76.6
abg_o2sat_date	3906	76.4
nic_first_date	3879	75.9
nic_last_date	3879	75.9
vbg_sbe	3855	75.4
vbg_po2	3853	75.4
vbg_hco3	3843	75.2
vbg_po2_date	3841	75.2
vbg_hco3_date	3814	74.6
_merge_emer_predisp	3810	74.5
other_resp_failure_date	3791	74.2
_merge_emer_respfail	3764	73.6
cxr1v_first_date	3744	73.3
cxr1v_last_date	3744	73.3
cxr1v_dur	3744	73.3
vbg_ph_date	3726	72.9
_merge_inpat_obes	3693	72.3
vbg_co2	3679	72.0
vbg_co2_flag	3679	72.0
corr_vbg_co2	3679	72.0
corr_vbg_co2_flag	3679	72.0
pco2_cat_vbg	3679	72.0
highest_vbg_co2	3677	71.9

variable	n_miss	pct_miss
highest_vbg_co2_flag	3677	71.9
vbg_co2_date	3676	71.9
highest_vbg_co2_date	3676	71.9
dm_last_date	3645	71.3
dm_first_date	3644	71.3
spo2	3632	71.1
spo2_date	3620	70.8
vbg_ph	3593	70.3
_merge_emer_vbg	3396	66.4
_merge_emer_ventsupp	3357	65.7
_merge_emer	3357	65.7
paco2	3321	65.0
paco2_int	3321	65.0
paco2_flag	3321	65.0
paco2_52_flag	3321	65.0
not_paco2_flag	3321	65.0
paco2_52_comp_flag	3321	65.0
max_hco3_or_its_met_alk	3321	65.0
max_hco3_or_its_comb_met_alk	3321	65.0
paco2_50_flag	3321	65.0
pco2_cat_abg	3321	65.0
paco2_date	3318	64.9
highest_paco2	3318	64.9
paco2_date_highest	3318	64.9
highest_paco2_flag	3318	64.9
abg_ph	3295	64.5
op_diuretics_last_date	3234	63.3
op_diuretics_first_date	3229	63.2
serum_lac	3081	60.3
serum_lac_date	3048	59.6
inpt_inh_date	2995	58.6
curr_bmi	2862	56.0
curr_bmi_date	2862	56.0
working_bmi	2859	55.9
working_bmi_date	2859	55.9

variable	n_miss	pct_miss
po_steroid_date	2832	55.4
rr	2796	54.7
rr_date	2786	54.5
serum_phos	2715	53.1
copd_med_last_date	2665	52.1
copd_med_first_date	2659	52.0
_merge_inpat_predisp	2561	50.1
temp_new	2488	48.7
new_temp_date	2488	48.7
acidemia	2357	46.1
_merge_inpat_respfail	2285	44.7
calc_bmi	2128	41.6
calc_bmi_date	2128	41.6
op_opiate_last_date	1924	37.6
serum_tprot	1923	37.6
op_opiate_first_date	1908	37.3
_merge_inpat_vbg	1894	37.1
hr	1885	36.9
hr_date	1885	36.9
serum_albumin	1834	35.9
is_inp	1754	34.3
_merge_inpat_ventsupp	1754	34.3
bmi	1738	34.0
bmi_date	1738	34.0
bmi_int	1738	34.0
bmi_by_five	1738	34.0
curr_weight	1625	31.8
curr_weight_date	1625	31.8
weight	1621	31.7
weight_date	1621	31.7
dbp	1601	31.3
dbp_date	1594	31.2
sbp	1585	31.0
sbp_date	1583	31.0
curr_height	1413	27.6

variable	n_miss	pct_miss
curr_height_date	1413	27.6
height	1410	27.6
height_date	1410	27.6
wbc	879	17.2
wbc_date	781	15.3
hgb	539	10.5
serum_ca	513	10.0
serum_cr	454	8.88
hgb_date	432	8.45
serum_cr_date	422	8.26
plt	415	8.12
serum_k	397	7.77
serum_k_date	397	7.77
k_cat	397	7.77
plt_date	359	7.02
serum_cl	319	6.24
serum_cl_date	317	6.20
serum_hco3	307	6.01
hco3_cat	307	6.01
hco3_flag	307	6.01
not_hco3_flag	307	6.01
serum_hco3_date	302	5.91
sodium	286	5.60
sodium_date	285	5.58
any_bicarb	255	4.99
int_bicarb	255	4.99
died_1mo	30	0.587
died_2mo	30	0.587
encounter_id	0	0
rfs	0	0
sex	0	0
race	0	0
ethnicity	0	0
location	0	0
age_at_encounter	0	0

variable	n_miss	pct_miss
los	0	0
bnp_date	0	0
serum_phos_date	0	0
serum_ca_date	0	0
serum_albumin_date	0	0
serum_tprot_date	0	0
has_j9612	0	0
has_j9622	0	0
has_j9602	0	0
has_j9692	0	0
ohs_code	0	0
has_j9600	0	0
principal_diagnosis_indicator	0	0
admitting_diagnosis	0	0
reason_for_visit	0	0
has_j9601	0	0
has_j961	0	0
has_j9610	0	0
has_j9611	0	0
has_j962	0	0
has_j9620	0	0
has_j9621	0	0
has_j9690	0	0
has_j9691	0	0
other_abn_of_br	0	0
cfdo	0	0
has_i50_acute	0	0
acute_nmd	0	0
sepsis_dx	0	0
stupor_dx	0	0
cog_signs_dx	0	0
mal_fat_dx	0	0
resp_acid_dx	0	0
sleep_hypovent_dx	0	0
cchs_dx	0	0

variable	n_miss	pct_miss
other_sleep_hypovent_dx	0	0
acidosis_unspec	0	0
headache_dx	0	0
cpap	0	0
tte_proc	0	0
aero	0	0
inh_teaching	0	0
cxr1v	0	0
cxr2v	0	0
ctcnoncon	0	0
ctccon	0	0
cc_time	0	0
meas_venous_o2_proc	0	0
meas_arterial_gas_proc	0	0
blood_cx_proc	0	0
art_punct_proc	0	0
ctabdpelv	0	0
osa	0	0
asthma	0	0
copd	0	0
chf	0	0
stroke	0	0
ckd	0	0
pvd	0	0
oud	0	0
sedatives	0	0
phtn	0	0
polycy	0	0
po_steroid	0	0
narcan	0	0
inpt_inh	0	0
vasodilators	0	0
ip_diuretics	0	0
ip_abx	0	0
paralytic	0	0

variable	n_miss	pct_miss
op_diuretics	0	0
op_opiate	0	0
op_mat	0	0
op_nrt	0	0
copd_med	0	0
muscle_relax	0	0
pat_enc_hash	0	0
ABG_rfs	0	0
is_amb	0	0
encounter_date	0	0
age_by_ten	0	0
age_decade	0	0
died	0	0
miss_paco2_flag	0	0
miss_vbg_co2_flag	0	0
miss_vbg_or_abg_co2_flag	0	0
inpt_inh_0	0	0
ip_abx_0	0	0
ip_diuretics_0	0	0
narcan_0	0	0
paralytic_0	0	0
po_steroid_0	0	0
vasodilators_0	0	0
op_diuretics_365d	0	0
op_opiate_365d	0	0
op_mat_365d	0	0
op_nrt_365d	0	0
copd_med_365d	0	0
muscle_relax_365d	0	0
vent_proc	0	0
niv_proc	0	0
imv_proc	0	0
aero_0	0	0
blood_cx_proc_0	0	0
cc_time_0	0	0

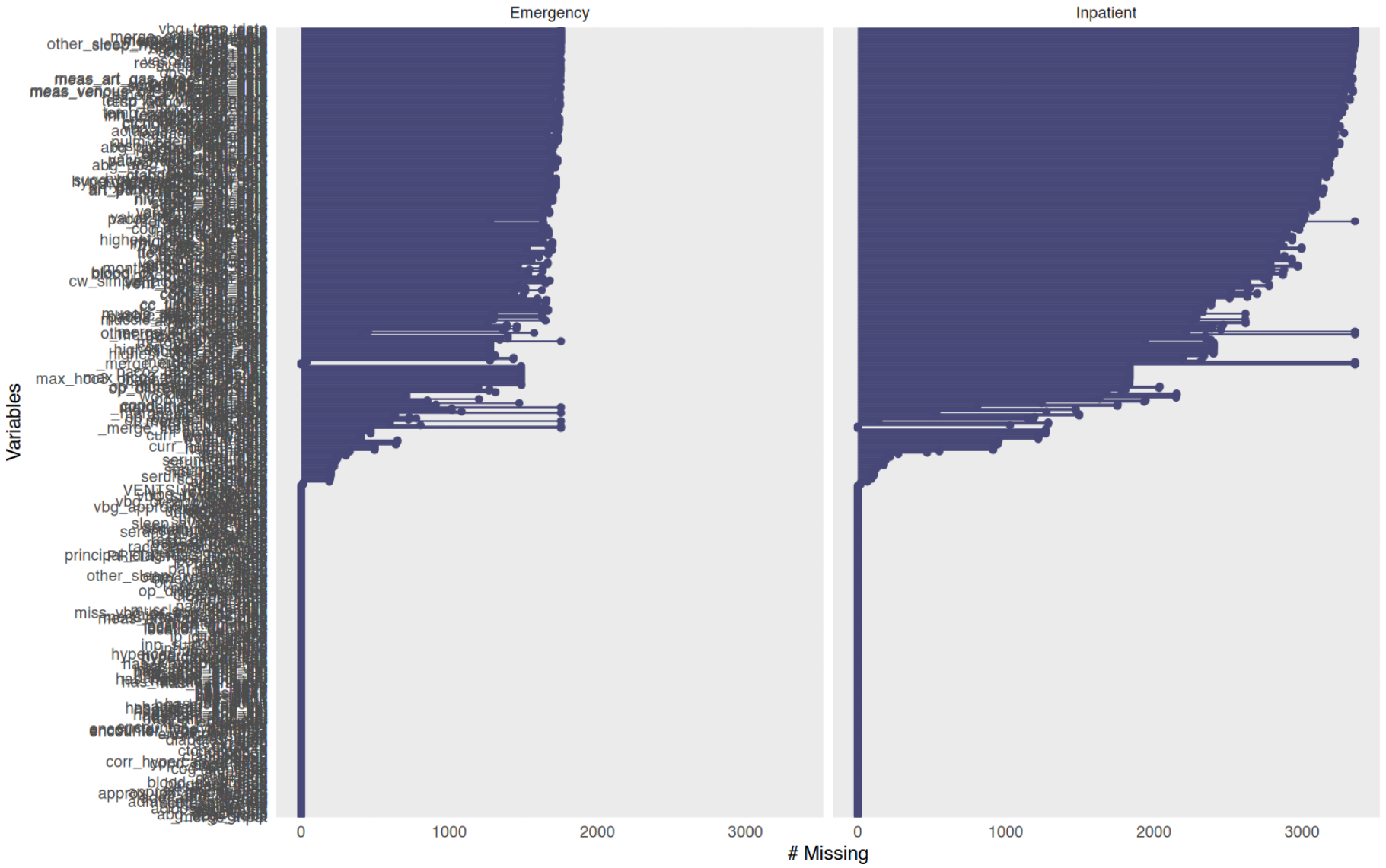
variable	n_miss	pct_miss
cpap_0	0	0
ctabdpelv_0	0	0
ctccon_0	0	0
ctcnoncon_0	0	0
cxr1v_0	0	0
cxr2v_0	0	0
imv_proc_0	0	0
inh_teaching_0	0	0
niv_proc_0	0	0
tte_proc_0	0	0
vent_proc_0	0	0
hypercap_resp_failure	0	0
other_resp_failure	0	0
dysp_dx	0	0
symp_obs	0	0
abn_br_dx	0	0
resp_abnormality	0	0
fast_br	0	0
pulm_edema_dx	0	0
pna_dx	0	0
acute_chf	0	0
acute_old	0	0
acute_old_date	0	0
resp_dep_compl	0	0
ctd	0	0
dem	0	0
dm	0	0
nmd	0	0
nic	0	0
ovs	0	0
pos_ohs_flag	0	0
OBESITY_rfs	0	0
PREDISPOSITION_rfs	0	0
RESPFAIL_rfs	0	0
VBG_rfs	0	0

variable	n_miss	pct_miss
VENTSUPPORT_rfs	0	0
is_emer	0	0
_merge_inpat	0	0
patient_id	0	0
rfsgroup	0	0
encounter_type	0	0
first_encounter	0	0
has_abg	0	0
has_vbg	0	0
hypercap_on_abg	0	0
hypercap_on_vbg	0	0
has_both_abg_vbg	0	0
has_neither_abg_vbg	0	0
vbg_or_abg_co2_flag	0	0
vbg_o2sat_calc	0	0
abg_o2sat_calc	0	0
corr_hypercap_on_vbg	0	0
race_ethnicity	0	0
has_vbg_and_cat	0	0
has_bmi_and_cat	0	0
has_weight_and_cat	0	0
has_height_and_cat	0	0
has_hr_and_cat	0	0
has_sbp_and_cat	0	0
has_rr_and_cat	0	0
has_temp_and_cat	0	0
has_spo2_and_cat	0	0
has_cl_and_cat	0	0
has_k_and_cat	0	0
has_hco3_and_cat	0	0
has_lactate_and_cat	0	0
has_na_and_cat	0	0
has_cr_and_cat	0	0
has_hgb_and_cat	0	0
has_wbc_and_cat	0	0

variable	n_miss	pct_miss
has_plt_and_cat	0	0
has_bnp_and_cat	0	0
has_phos_and_cat	0	0
has_ca_and_cat	0	0
has_alb_and_cat	0	0
has_tprot_and_cat	0	0
encounter_type_dummy1	0	0
encounter_type_dummy2	0	0
encounter_type_dummy3	0	0
female	0	0
male	0	0
white_race	0	0
black_race	0	0
unknown_race	0	0
asian_race	0	0
nat_am_race	0	0
nhpi_race	0	0
not_hisp_eth	0	0
hisp_eth	0	0
unknown_eth	0	0
location_dummy1	0	0
location_dummy2	0	0
location_dummy3	0	0
location_dummy4	0	0
ps	0	0
tx_pr_decile	0	0
sumofweights	0	0
ipw	0	0
approx_ipw_fweight	0	0
inp_ps	0	0
inp_sumofweights	0	0
inp_ipw	0	0
approx_inp_ipw_fweight	0	0
vbg_ps	0	0
vbg_tx_pr_decile	0	0

variable	n_miss	pct_miss
vbg_sumofweights	0	0
vbg_ipw	0	0
vbg_approx_ipw_fweight	0	0
ref_ym	0	0
death_60d	0	0
abg_group	0	0
vbg_group	0	0
sex_label	0	0
race_ethnicity_label	0	0
location_label	0	0
encounter_type_label	0	0
osa_label	0	0
asthma_label	0	0
copd_label	0	0
chf_label	0	0
nmd_label	0	0
phtn_label	0	0
ckd_label	0	0
diabetes_label	0	0
abg_status	0	0
vbg_status	0	0
hyper_abg	0	0
hyper_vbg	0	0
w_abg	0	0
w_vbg	0	0

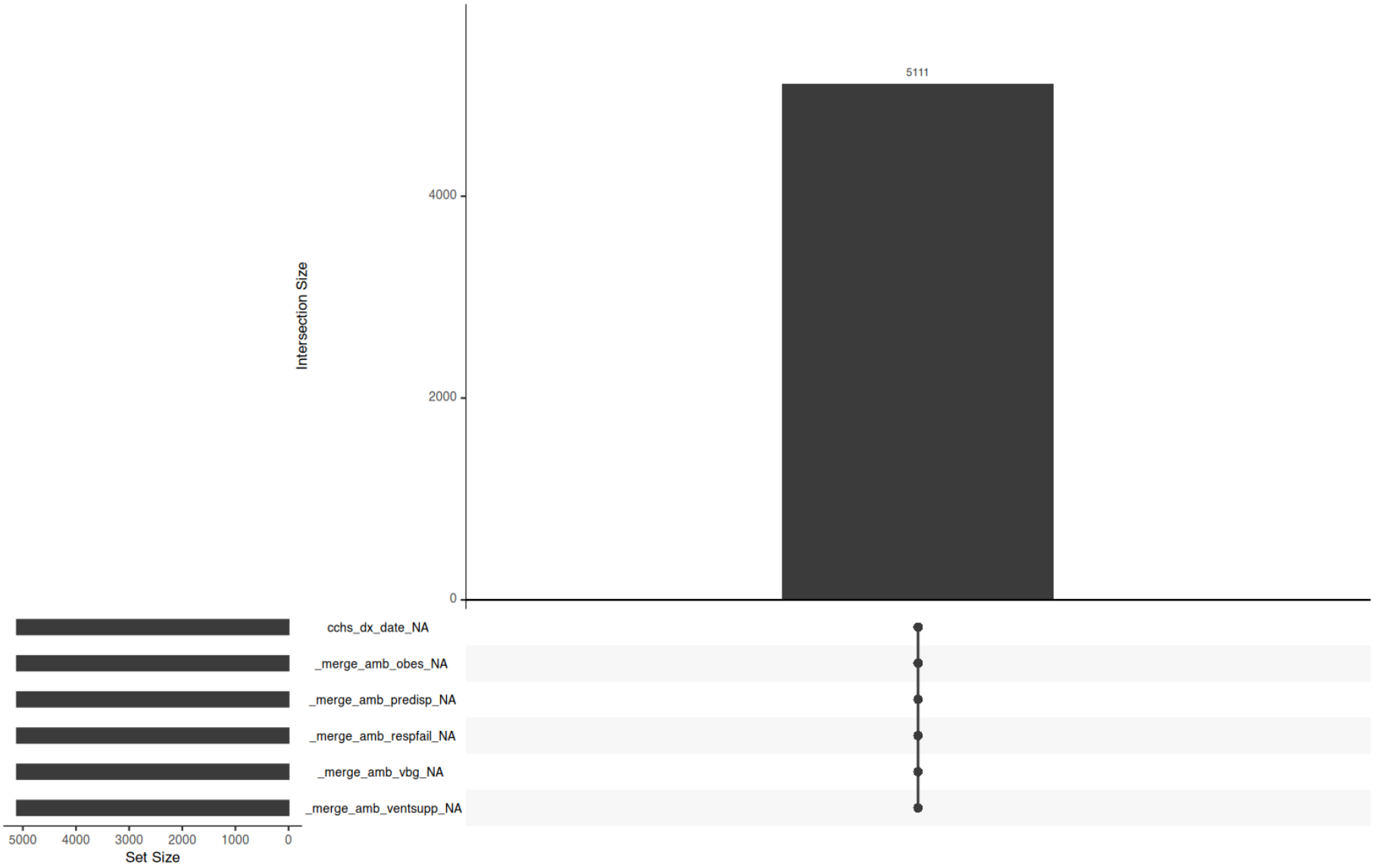
```
# Patterns
naniar::gg_miss_var(subset_data, facet = encounter_type)
```



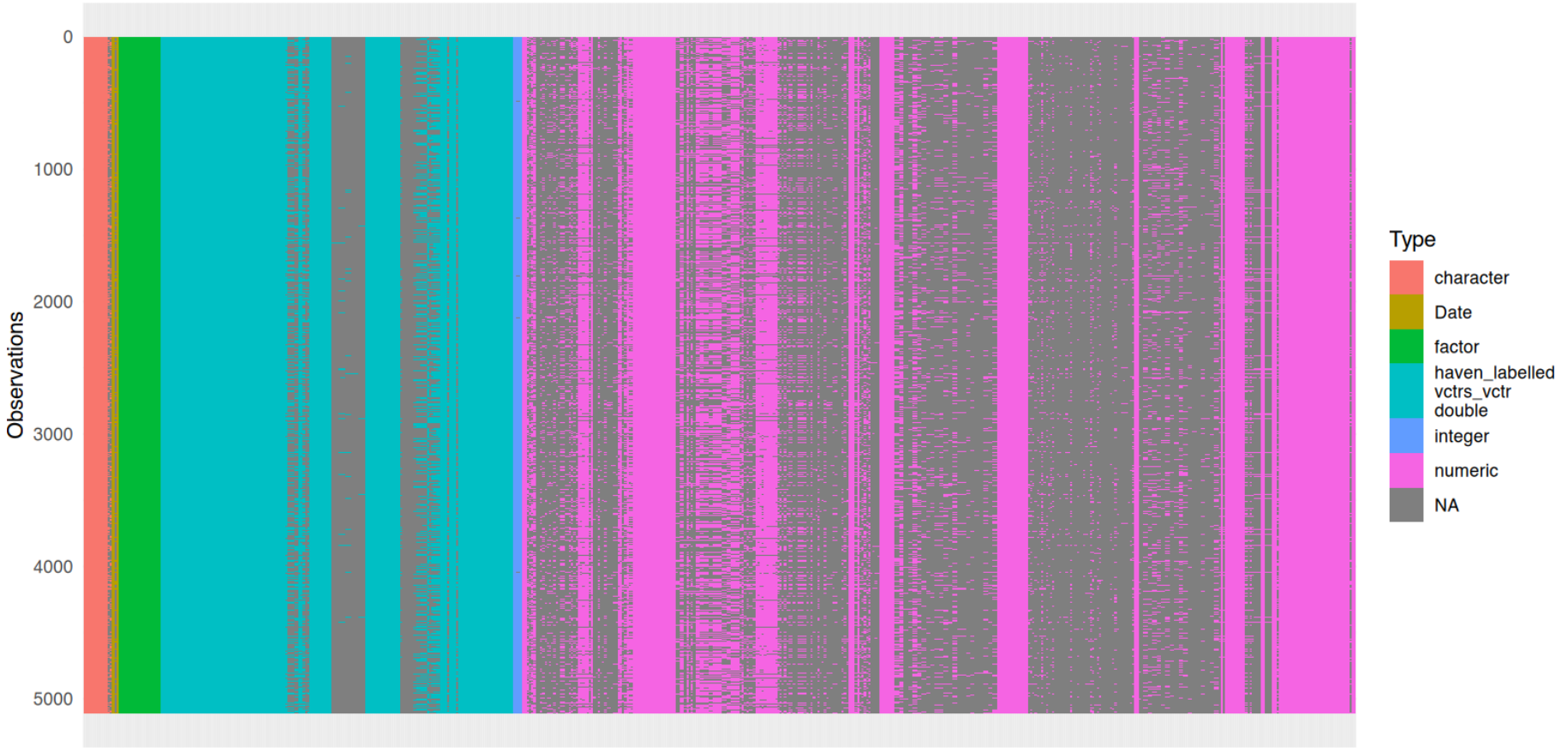

```
naniar::gg_miss_upset(subset_data, nsets = 6)
```

```
Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
i Please use tidy evaluation idioms with `aes()`.
```

```
i See also `vignette("ggplot2-in-packages")` for more information.
i The deprecated feature was likely used in the UpSetR package.
  Please report the issue to the authors.

Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
i Please use the `linewidth` argument instead.
i The deprecated feature was likely used in the UpSetR package.
  Please report the issue to the authors.
```




```
# Types & potential issues
visdat::vis_dat(subset_data, warn_large_data = FALSE)
```



Chunk mi-missing-audit runtime: 13.32 s

4.2 8) Pre-imputation data prep (consistent types & predictors)

Why: MI models need coherent types; using exactly the same covariates as the propensity score models avoids model drift.

```
# Ensure intended factor/numeric types for imputation
# --- Inspect current encounter_type ----------------------------------------
cat("encounter_type class:", paste(class(subset_data$encounter_type), collapse = ", "), "\n")
```

```
encounter_type class: factor
```

```
print(utils::head(unique(subset_data$encounter_type), 20))
```

```
[1] Emergency Inpatient
Levels: Emergency Inpatient
```

```
# Keep a raw copy for debugging if mapping fails
encounter_type_raw <- subset_data$encounter_type

# --- Helpers ---------------------------------------------------------------

# Map various encodings to strict 0/1 integer (for sex + 0/1 indicators)
to01 <- function(x) {
  if (is.logical(x)) return(as.integer(x))
  if (is.factor(x))  x <- as.character(x)

  out <- rep(NA_integer_, length(x))
  xs  <- suppressWarnings(as.numeric(x))
  is_num <- !is.na(xs)

  # numeric 0/1
  out[is_num & xs %in% c(0, 1)] <- as.integer(xs[is_num & xs %in% c(0, 1)])

  # character encodings (case/space-insensitive)
  if (any(!is_num)) {
    s <- trimws(tolower(as.character(x[!is_num])))
```

```r
    out[!is_num][s %in% c("0","no","false","female","f")] <- 0L
    out[!is_num][s %in% c("1","yes","true","male","m")]   <- 1L
  }
  out
}

# Robust normalizer for encounter_type:
# - accepts numeric 2/3, digit-strings "2", "3", "2 - ED", etc.
# - accepts common synonyms with word-boundary protection
normalize_encounter_type <- function(x) {
  # to character once
  s_chr <- trimws(tolower(as.character(x)))

  # try to pull numeric code from any digits present
  num_from_text <- suppressWarnings(as.numeric(gsub("[^0-9]+", "", s_chr)))

  lab <- rep(NA_character_, length(s_chr))

  # numeric path
  lab[!is.na(num_from_text) & num_from_text == 2] <- "Emergency"
  lab[!is.na(num_from_text) & num_from_text == 3] <- "Inpatient"

  # synonym path (fill only still-NA)
  is_na <- is.na(lab)

  # Emergency synonyms: "emergency", "emerg", exact "ed", "a&e", "emergency dept"
  is_em <- grepl("\\bemerg(?:ency)?\\b", s_chr) |
           grepl("(^|[^a-z])ed([^a-z]|$)", s_chr) |
           grepl("\\ba&e\\b", s_chr) |
           grepl("\\bemergency\\s+dept\\b", s_chr)

  # Inpatient synonyms: "inpatient", "inpt", "inpat", exact "ip"
  is_ip <- grepl("\\binpatient\\b", s_chr) |
           grepl("\\binpt\\b",     s_chr) |
           grepl("\\binpat\\b",    s_chr) |
           grepl("(^|[^a-z])ip([^a-z]|$)", s_chr)
```

```
  lab[is_na & is_em] <- "Emergency"
  lab[is_na & is_ip] <- "Inpatient"

  factor(lab, levels = c("Emergency", "Inpatient"))
}

# --- Coerce analysis types (including encounter_type) -------------------------
subset_data <- subset_data |>
  mutate(
    sex               = factor(to01(sex), levels = c(0L, 1L), labels = c("Female", "Male")),
    race_ethnicity    = if (is.factor(race_ethnicity)) race_ethnicity else factor(race_ethnicity),
    location          = if (is.factor(location))       location       else factor(location),
    encounter_type    = normalize_encounter_type(encounter_type),
    has_abg           = to01(has_abg),
    has_vbg           = to01(has_vbg),
    hypercap_on_abg   = to01(hypercap_on_abg),
    hypercap_on_vbg   = to01(hypercap_on_vbg)
  )

# immediately drop unused levels and assert exactly two levels in the observed data used by MI
subset_data$encounter_type <- droplevels(subset_data$encounter_type)
stopifnot(nlevels(subset_data$encounter_type) == 2L)

# --- Diagnostics for encounter_type -------------------------------------------
tab_enc <- table(subset_data$encounter_type, useNA = "ifany")
print(tab_enc)
```

```
Emergency Inpatient
     1754      3357
```

```
if (sum(!is.na(subset_data$encounter_type)) == 0) {
  message("All encounter_type values are NA after normalization. Showing top raw values:")
  s_raw <- trimws(tolower(as.character(encounter_type_raw)))
  print(utils::head(sort(table(s_raw), decreasing = TRUE), 20))
```

```
  stop("normalize_encounter_type produced all NA; extend the synonym map to your raw values.")
}

# Must have at least one observed value and (after droplevels) exactly two levels
stopifnot(sum(!is.na(subset_data$encounter_type)) > 0)
stopifnot(nlevels(droplevels(subset_data$encounter_type)) == 2)
```

Chunk mi-prep runtime: 0.07 s

Chunk type-invariants runtime: 0.02 s

4.3 9) Imputation model specification (MICE)

4.3.1 9.1 Predictor matrix & methods. Run MICE (moderate settings for scale)

```
# --- variables for GBM propensity (kept identical to main analysis) ---
# ---- MICE setup (Option A: include PaCO2 for ABG + keep VBG CO2/O2Sat) ----
library(mice)
library(dplyr)

# --- add analysis targets and CO2 measures explicitly -------------------------
co2_vars <- c("paco2", "vbg_co2", "vbg_o2sat")

mi_vars <- unique(c(
  covars_gbm,
  "has_abg","has_vbg",                                  # treatments (NOT imputed)
  "imv_proc","niv_proc","death_60d","hypercap_resp_failure",  # outcomes (NOT imputed)
  co2_vars
))

mi_df <- subset_data[, mi_vars, drop = FALSE]

# Make binary comorbids factors so "logreg" is used (and stays binary)
bin_covars <- c("copd","asthma","osa","chf","acute_nmd","phtn","ckd","dm")
```

```r
mi_df[bin_covars] <- lapply(mi_df[bin_covars], function(z) {
  if (is.factor(z)) return(droplevels(z))
  zz <- suppressWarnings(as.integer(z))
  factor(zz, levels = c(0L,1L), labels = c("0","1"))
})

# Force CO2 variables to be numeric BEFORE we convert any leftover characters to factor
coerce_num <- function(x) suppressWarnings(as.numeric(as.character(x)))
for (nm in intersect(co2_vars, names(mi_df))) mi_df[[nm]] <- coerce_num(mi_df[[nm]])

# For MICE: convert any remaining characters → factors (after CO2 numeric coercion)
mi_df <- dplyr::mutate(mi_df, across(where(is.character), ~ factor(.x)))

# --- methods & predictor matrix aligned to *mi_df* ----------------------------
meth <- mice::make.method(mi_df)

is_fac      <- vapply(mi_df, is.factor,  logical(1))
is_num      <- vapply(mi_df, is.numeric, logical(1))
is_bin_fac  <- vapply(mi_df, function(x) is.factor(x)  && nlevels(x) == 2, logical(1))
is_multicat <- vapply(mi_df, function(x) is.factor(x)  && nlevels(x) >  2, logical(1))

# robust defaults
meth[is_num]      <- "pmm"      # numerics: predictive mean matching
meth[is_multicat] <- "polyreg"  # unordered multicategory
meth[is_bin_fac]  <- "logreg"   # binary factors: logistic regression

# never impute treatments or outcomes
no_imp <- c("has_abg","has_vbg","imv_proc","niv_proc","death_60d","hypercap_resp_failure")
meth[intersect(names(meth), no_imp)] <- ""

# predictor matrix; forbid treatments/outcomes as predictors
pred <- mice::quickpred(mi_df, mincor = 0.05, minpuc = 0.25)
pred[, intersect(colnames(pred), no_imp)] <- 0
pred[intersect(rownames(pred), no_imp), ] <- 0

# MI integrity: treatments/outcomes excluded; binaries use logreg
```

```r
stopifnot(all(meth[no_imp] == ""))
stopifnot(all(colSums(pred[, intersect(colnames(pred), no_imp), drop = FALSE]) == 0))
stopifnot(all(rowSums(pred[intersect(rownames(pred), no_imp), , drop = FALSE]) == 0))
bin_fac <- names(which(vapply(mi_df, function(x) is.factor(x) && nlevels(x) == 2, logical(1))))
stopifnot(all(meth[bin_fac] == "logreg"))

# integrity checks
stopifnot(
  ncol(pred) == ncol(mi_df),
  nrow(pred) == ncol(mi_df),
  length(meth) == ncol(mi_df),
  identical(names(meth), colnames(mi_df))
)

# --- run MICE ----------------------------------------------------------------
set.seed(20251206)
imp <- mice::mice(
  data            = mi_df,
  m               = 5,
  maxit           = 10,
  predictorMatrix = pred,
  method          = meth,
  printFlag       = TRUE,
  seed            = 20251206
)
```

```
 iter imp variable
  1   1  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  1   2  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  1   3  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  1   4  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  1   5  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  2   1  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  2   2  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  2   3  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
```

2	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
2	5	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
3	1	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
3	2	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
3	3	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
3	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
3	5	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
4	1	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
4	2	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
4	3	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
4	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
4	5	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
5	1	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
5	2	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
5	3	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
5	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
5	5	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
6	1	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
6	2	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
6	3	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
6	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
6	5	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
7	1	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
7	2	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
7	3	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
7	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
7	5	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
8	1	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
8	2	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
8	3	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
8	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
8	5	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
9	1	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
9	2	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
9	3	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru
9	4	curr_bmi	temp_new	sbp	dbp	hr	spo2	sodium	serum_cr	serum_hco3	serum_cl	serum_lac	serum_k	wbc	plt	bnp	seru

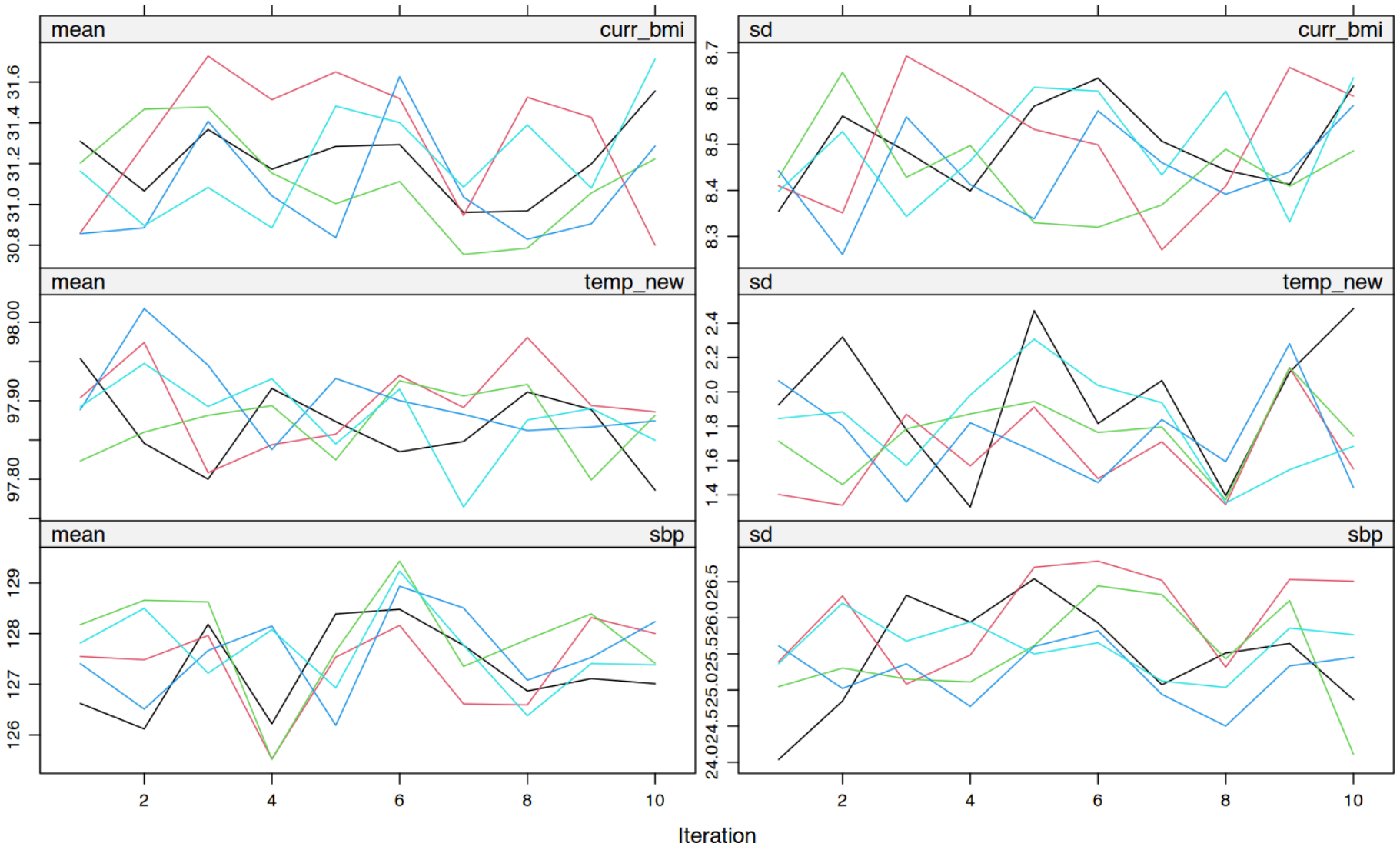
```
  9   5  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  10   1  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  10   2  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  10   3  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  10   4  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
  10   5  curr_bmi  temp_new  sbp  dbp  hr  spo2  sodium  serum_cr  serum_hco3  serum_cl  serum_lac  serum_k  wbc  plt  bnp  seru
```

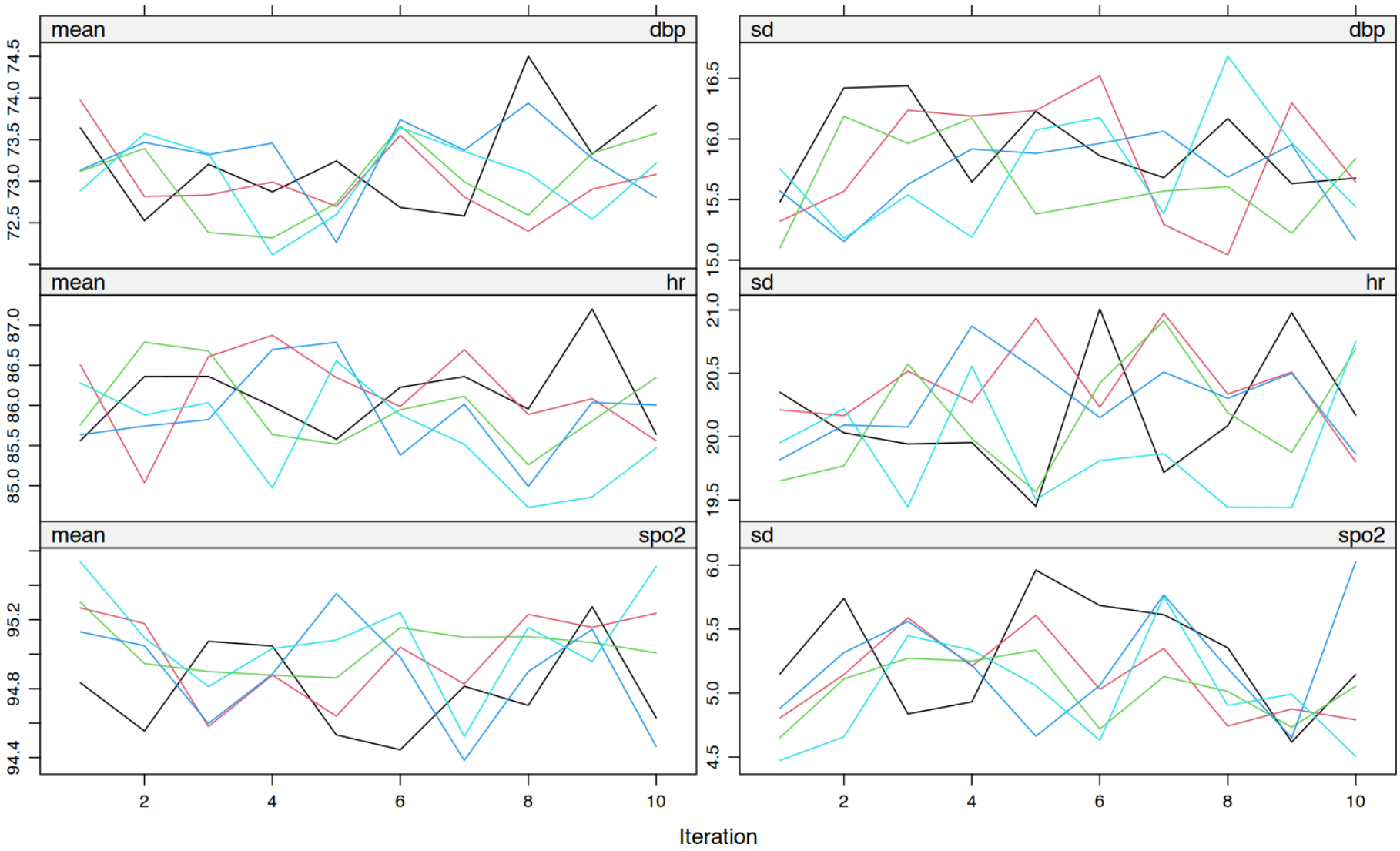
```r
saveRDS(imp, file = "mi_abg_vbg_mids.rds")

# quick sanity: these must exist and be numeric in completed data
dlist <- mice::complete(imp, action = "all")
stopifnot(all(c("paco2","vbg_co2") %in% names(dlist[[1]])))
stopifnot(is.numeric(dlist[[1]]$paco2), is.numeric(dlist[[1]]$vbg_co2))
```

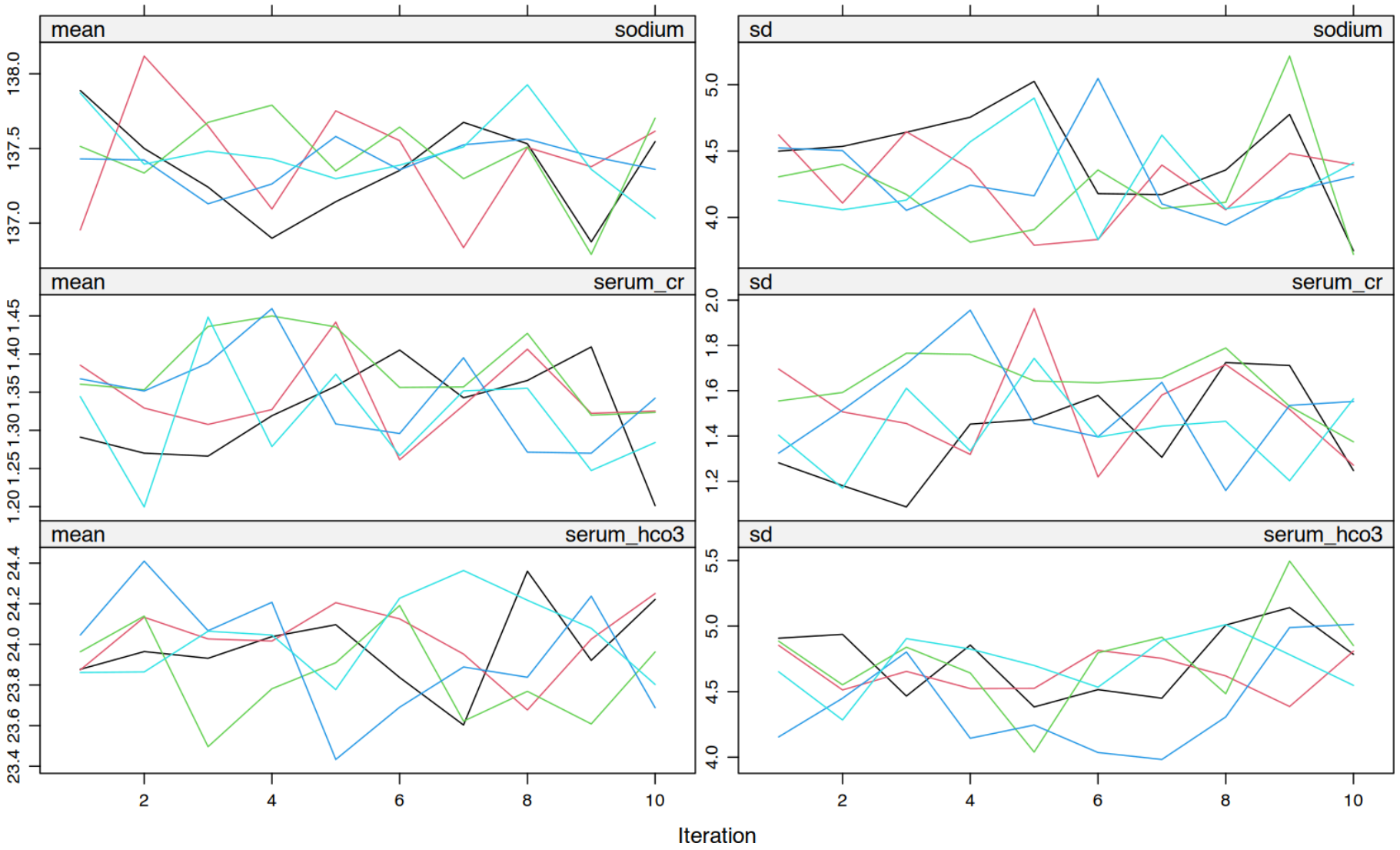
Chunk mi-exec runtime: 7.61 s

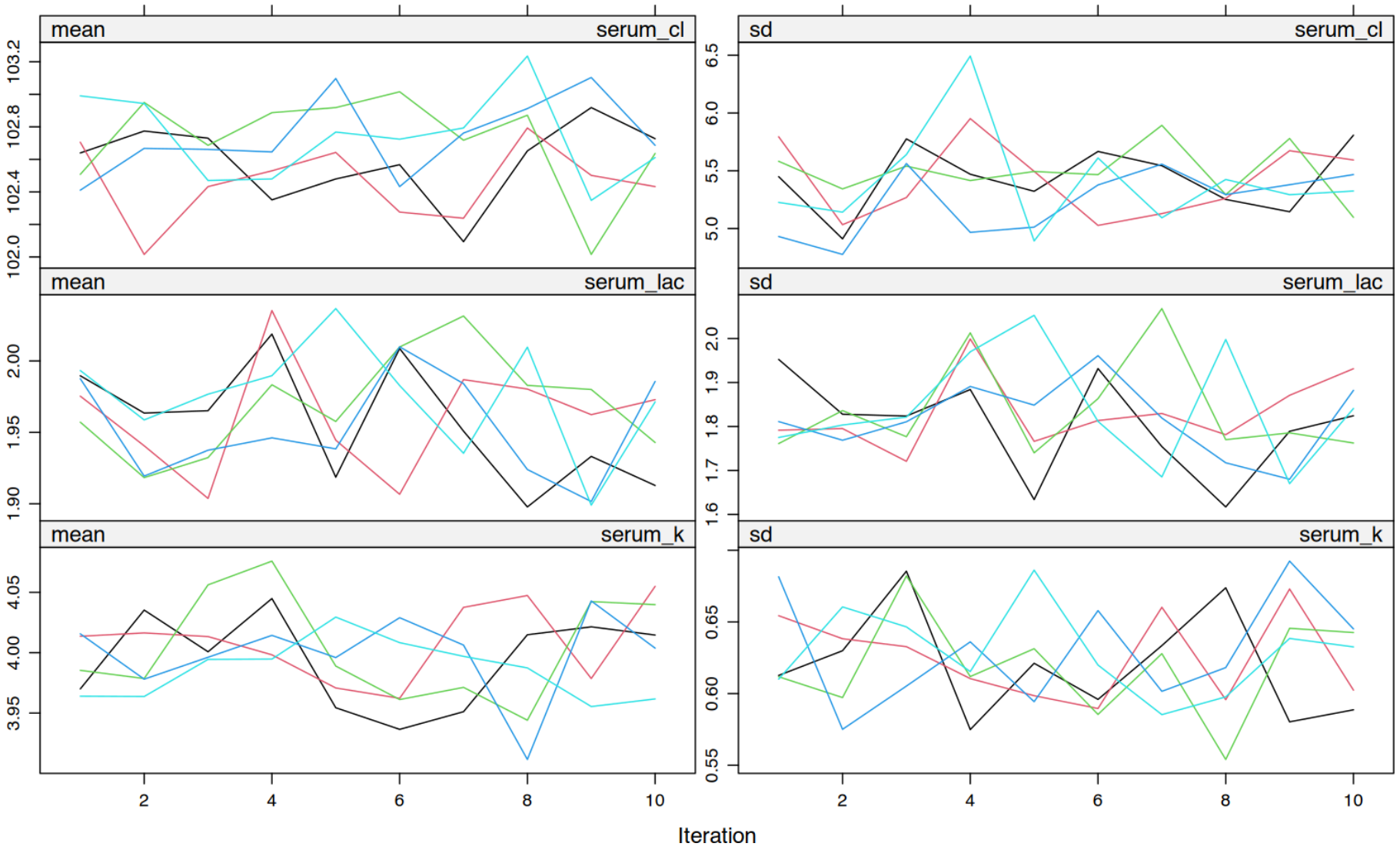
4.3.2 9.2 Convergence & plausibility checks

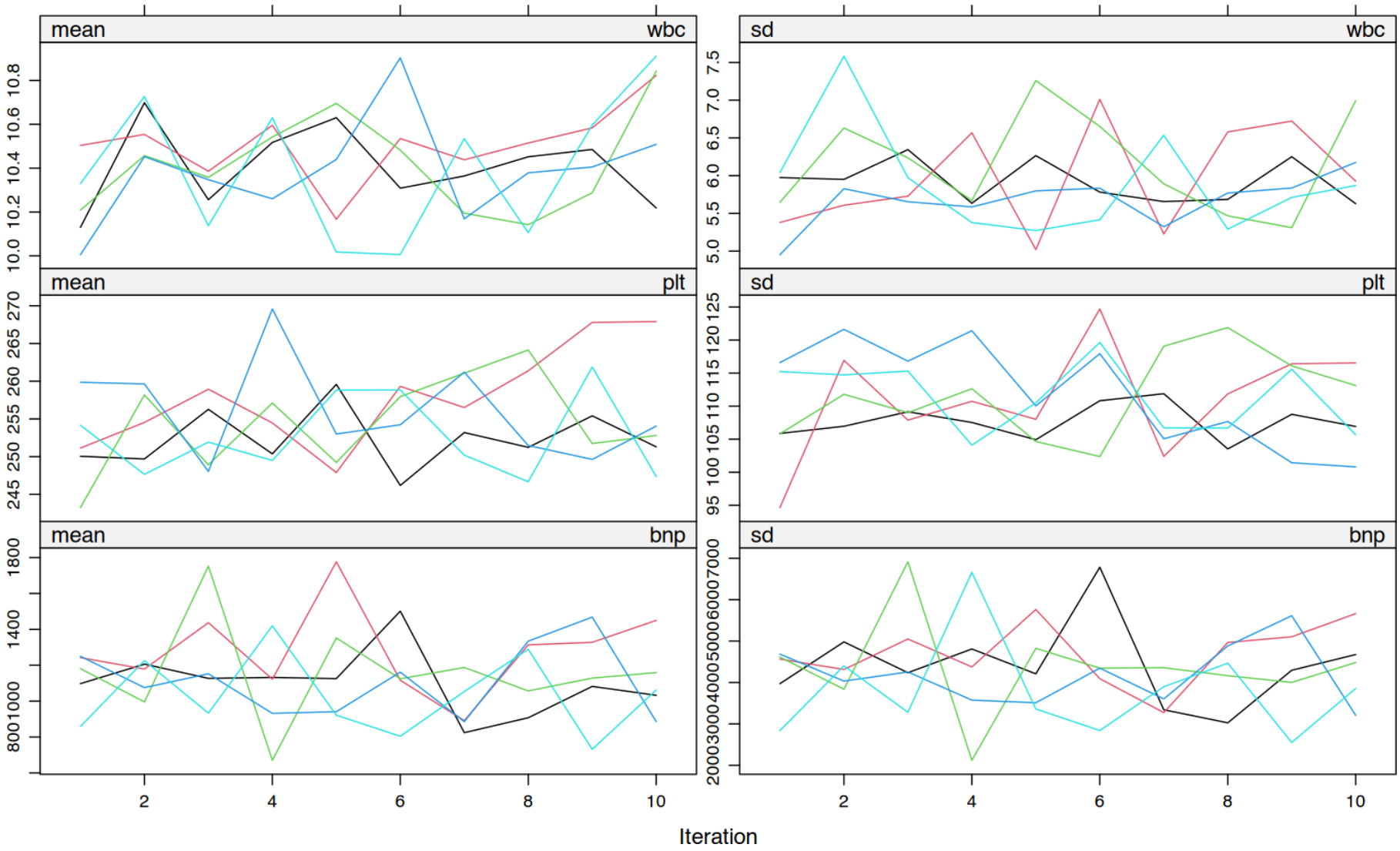
```r
imp <- readRDS("mi_abg_vbg_mids.rds")

plot(imp)                                    # trace plots
```

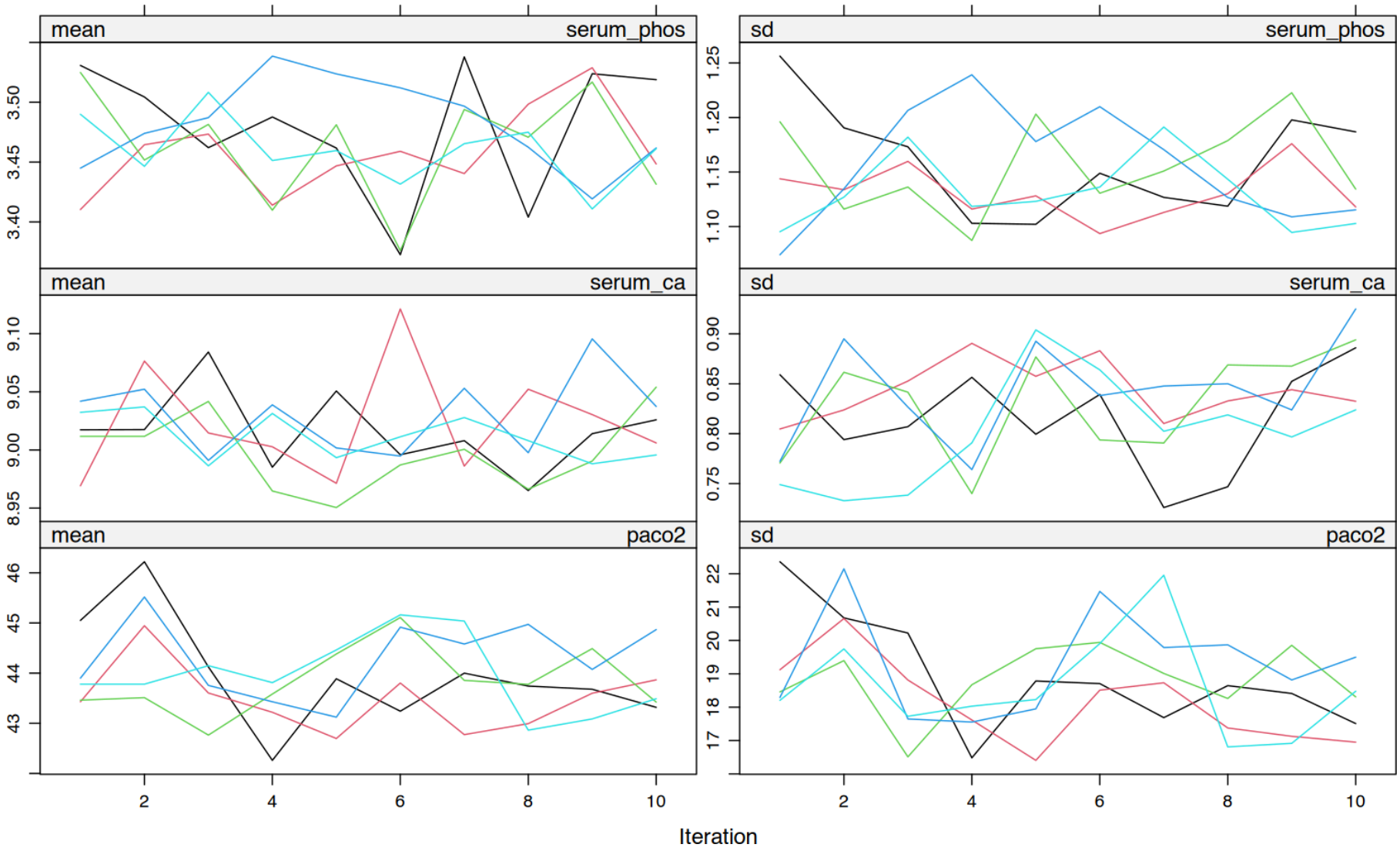

mean
curr_bmi
sd
curr_bmi
mean
temp_new
sd
temp_new
mean
sbp
sd
sbp
30.8 31.0 31.2 31.4 31.6
8.3 8.4 8.5 8.6 8.7
97.80 97.90 98.00
1.4 1.6 1.8 2.0 2.2 2.4
126 127 128 129
24.0 24.5 25.0 25.5 26.0 26.5
2 4 6 8 10
2 4 6 8 10
Iteration

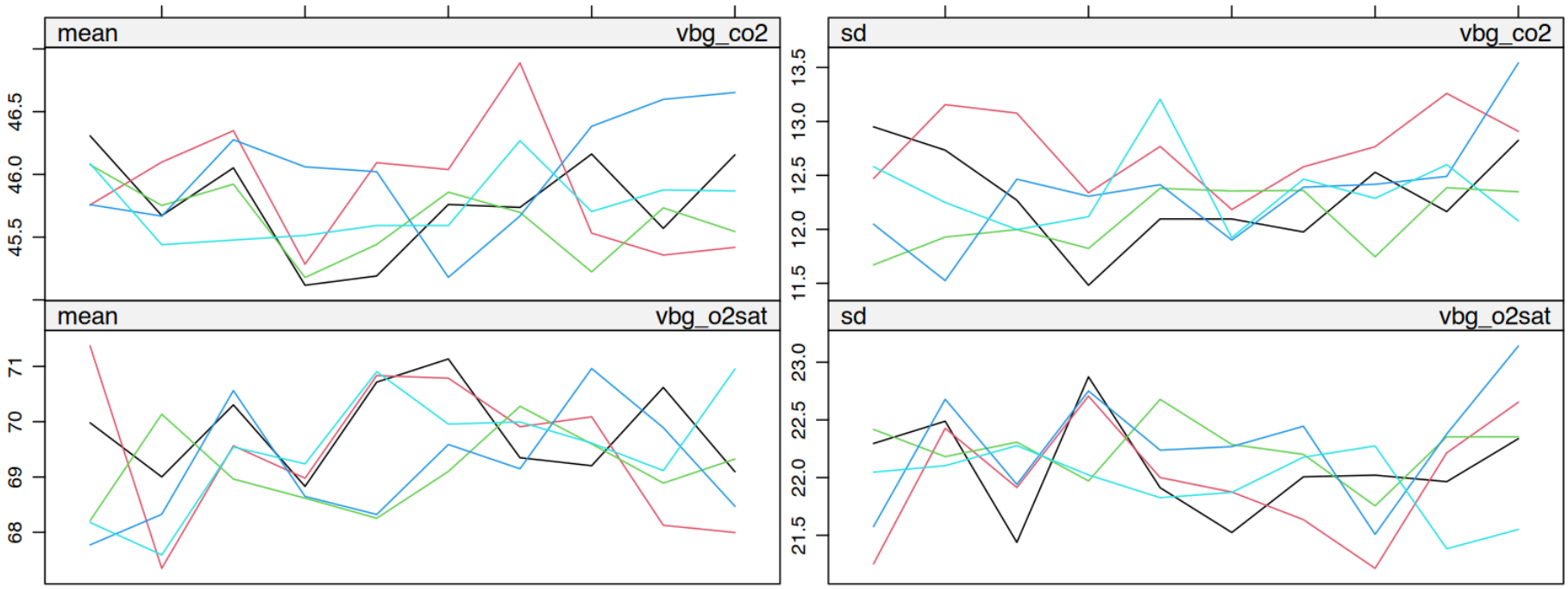

mean
dbp
sd
dbp
mean
hr
sd
hr
mean
spo2
sd
spo2
72.5 73.0 73.5 74.0 74.5
85.0 85.5 86.0 86.5 87.0
94.4 94.8 95.2
15.0 15.5 16.0 16.5
19.5 20.0 20.5 21.0
4.5 5.0 5.5 6.0
2 4 6 8 10
Iteration


mean
sodium
sd
sodium
mean
serum_cr
sd
serum_cr
mean
serum_hco3
sd
serum_hco3
Iteration

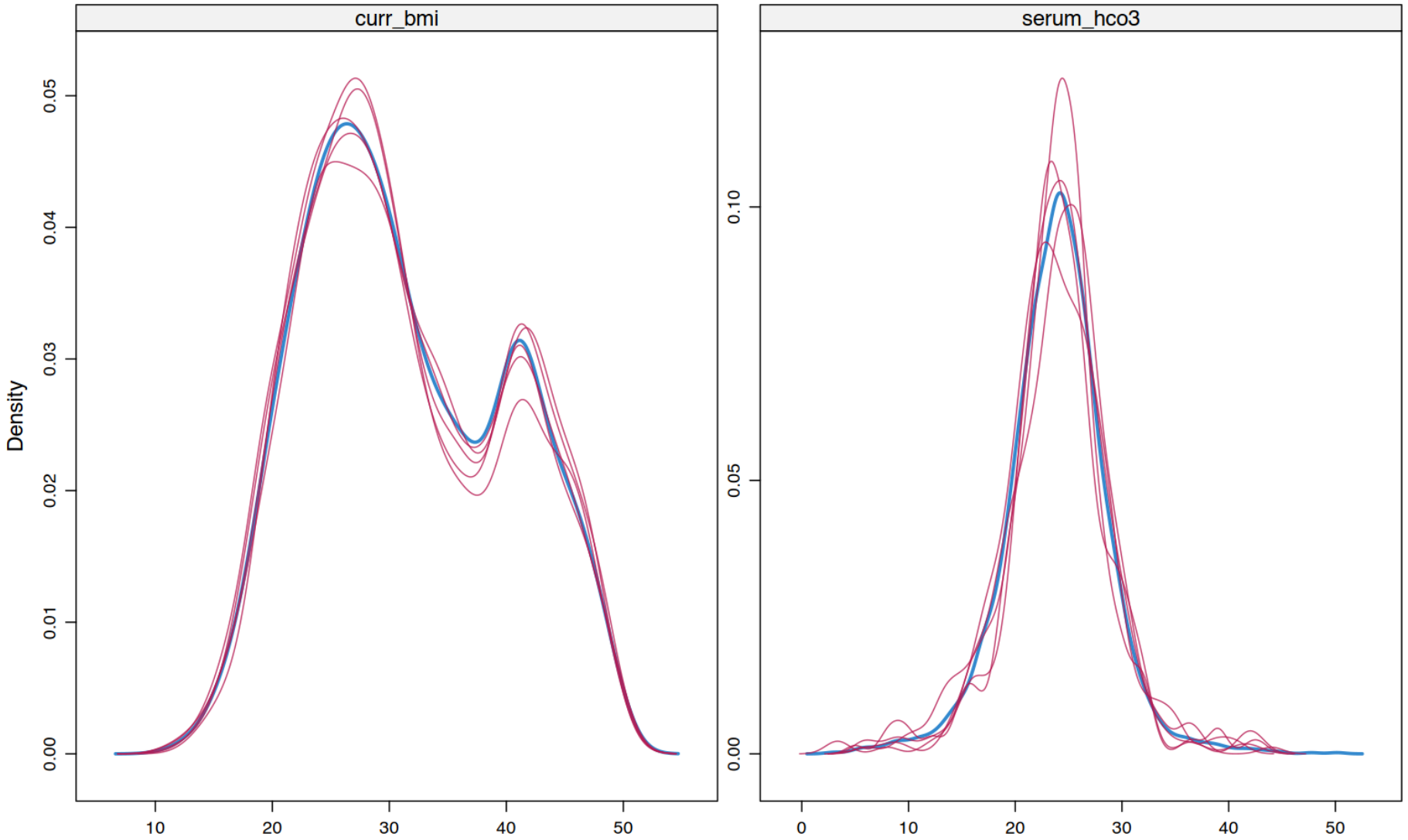

mean
serum_cl
sd
serum_cl
mean
serum_lac
sd
serum_lac
mean
serum_k
sd
serum_k
102.0
102.4
102.8
103.2
5.0
5.5
6.0
6.5
1.90
1.95
2.00
1.6
1.7
1.8
1.9
2.0
3.95
4.00
4.05
0.55
0.60
0.65
2
4
6
8
10
Iteration


mean
wbc
sd
wbc
mean
plt
sd
plt
mean
bnp
sd
bnp
Iteration

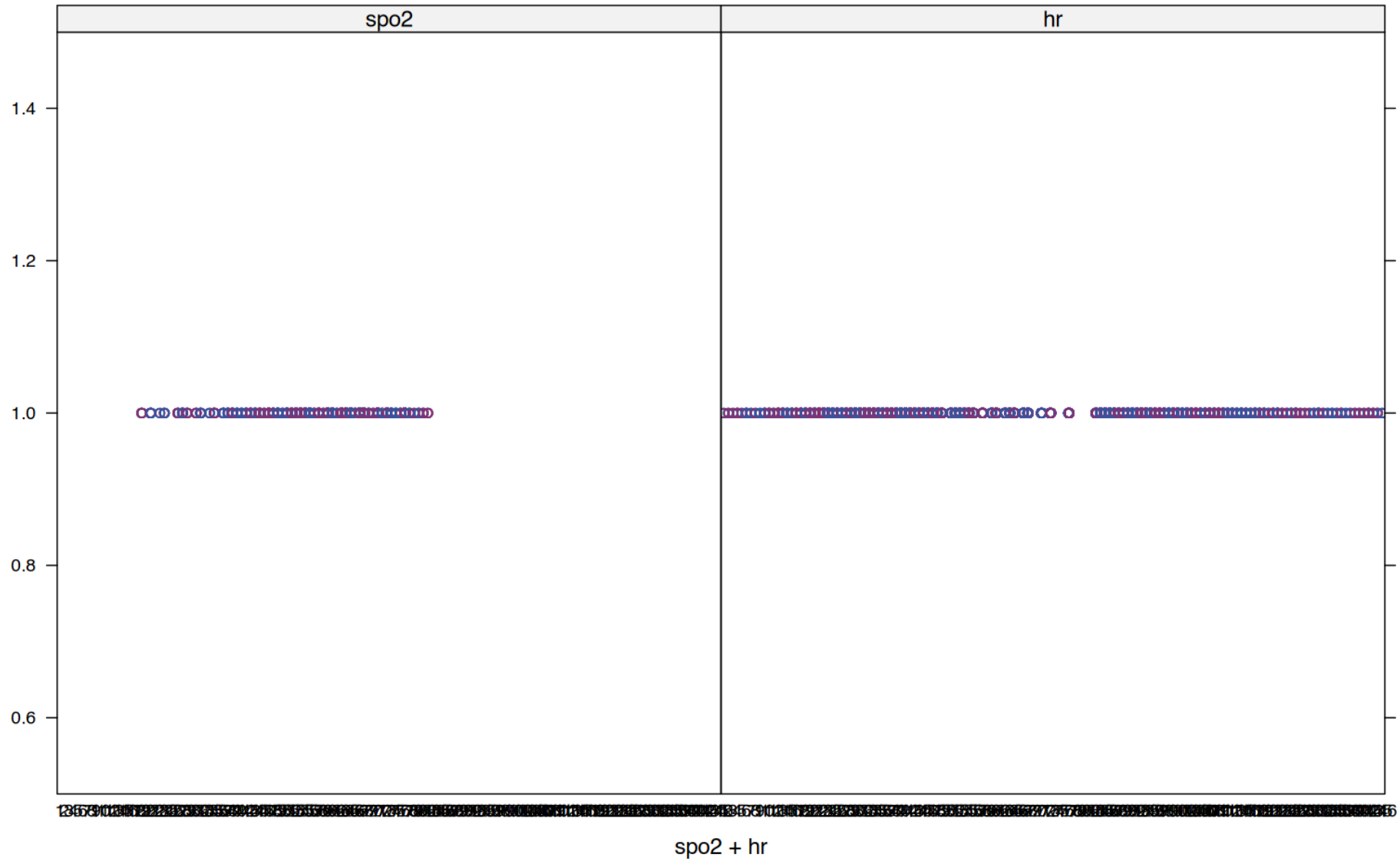

mean
serum_phos
sd
serum_phos
mean
serum_ca
sd
serum_ca
mean
paco2
sd
paco2
Iteration




```
densityplot(imp, ~ curr_bmi + serum_hco3)     # imputed vs observed
```



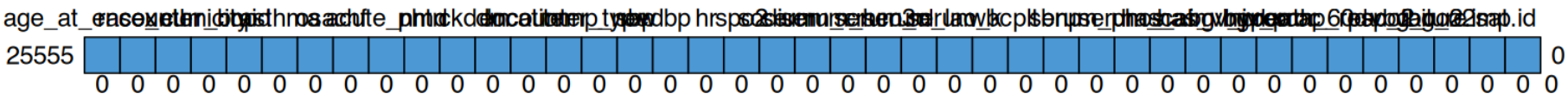

```
stripplot(imp,  ~ spo2 + hr)                # distribution overlap
```




```
md.pattern(complete(imp, "long"))            # pattern after MI - should be complete
```

```
 /\     /\
{  `---'  }
```

```
{  0   0  }
==>  V <==  No need for mice. This data set is completely observed.
 \  \|/  /
  `-----'
```




```
      age_at_encounter sex race_ethnicity curr_bmi copd asthma osa chf
25555                1   1              1        1    1      1   1   1
                     0   0              0        0    0      0   0   0
      acute_nmd phtn ckd dm location encounter_type temp_new sbp dbp hr spo2
25555         1    1   1  1        1              1        1   1   1  1    1
              0    0   0  0        0              0        0   0   0  0    0
      sodium serum_cr serum_hco3 serum_cl serum_lac serum_k wbc plt bnp
25555      1        1          1        1         1       1   1   1   1
           0        0          0        0         0       0   0   0   0
      serum_phos serum_ca has_abg has_vbg imv_proc niv_proc death_60d
25555          1        1       1       1        1        1         1
               0        0       0       0        0        0         0
      hypercap_resp_failure paco2 vbg_co2 vbg_o2sat .imp .id
25555                     1     1       1         1    1   1 0
                          0     0       0         0    0   0 0
```

Chunk mi-diagnostics runtime: 3.44 s

4.4 10) Refit propensity models within each imputation

We keep the GBM recipe close to the non-MI run, but with lighter CV (cv.folds = 3) and slightly fewer trees.

4.4.1 10.1 ABG propensity (has_abg)

```r
# Create completed datasets
dlist <- mice::complete(imp, action = "all")

# Fit ABG propensity weights in each imputation
fit_abg_one <- function(d) {
  w <- weightit(
    formula_abg,
    data   = d[, c("has_abg", covars_gbm)],
    method = "gbm",
    estimand    = "ATE",
    include.obj = TRUE,
    n.trees           = gbm_params$n.trees,
    interaction.depth = gbm_params$interaction.depth,
    shrinkage         = gbm_params$shrinkage,
    bag.fraction      = gbm_params$bag.fraction,
    cv.folds          = gbm_params$cv.folds,
    stop.method       = gbm_params$stop.method,
    n.cores           = gbm_params$n.cores
  )
  # stabilise + two-sided Winsorization
  ww <- w$weights; ww <- ww/mean(ww)
  cut <- stats::quantile(ww, c(.01,.99), na.rm = TRUE)
  ww <- pmin(pmax(ww, cut[1]), cut[2]); ww <- ww/mean(ww)
  w$weights <- ww
  w
}

with_progress({
  p <- progressor(along = seq_along(dlist))
  W_abg_list <- future_lapply(
    X = seq_along(dlist),
    FUN = function(i) {
      p(sprintf("Fitting ABG on imputation %d", i))
      set.seed(20251206 + i)           # per-imputation seed for reproducibility
      fit_abg_one(dlist[[i]])
```

```
    },
    future.seed = TRUE                  # reproducible RNG across workers
  )
})

saveRDS(W_abg_list, "mi_W_abg_list.rds")
```

Chunk mi-propensity-abg runtime: 29.66 s

```
        n   min p99.99%   max      ess
[1,] 5111 0.595   3.457 3.460 4145.181
[2,] 5111 0.593   3.369 3.372 4180.831
[3,] 5111 0.592   3.598 3.603 4122.118
[4,] 5111 0.593   3.533 3.534 4116.794
[5,] 5111 0.591   3.496 3.498 4152.556
```

Chunk mi-weight-diagnostics-abg runtime: 0.01 s

4.4.2 10.2 Balance diagnostics across imputations

```
# Vars you intended to use (from your earlier code)
vars0 <- covars_gbm

# Which factors collapse to 1 level AFTER complete-case filtering (per imputation, per arm)?
find_offenders_post_cc <- function(d, treat_var, vars) {
  keep <- c(treat_var, vars)
  dd   <- d[, keep, drop = FALSE]
  dd   <- dd[stats::complete.cases(dd), , drop = FALSE]  # mimic cobalt's CC
  if (!nrow(dd)) return(character(0))

  # factor with <2 levels in either arm
  bad <- vapply(vars, function(v) {
    x <- dd[[v]]
```

```
    if (!is.factor(x)) return(FALSE)
    by_arm <- tapply(x, dd[[treat_var]], function(z) nlevels(droplevels(z)))
    any(is.na(by_arm)) || any(by_arm < 2)
  }, logical(1))

  names(bad)[bad]
}

off_by_imp <- lapply(dlist, find_offenders_post_cc, treat_var = "has_abg", vars = vars0)
to_drop    <- Reduce(union, off_by_imp)  # union across imputations
message("Offenders (post CC): ", if (length(to_drop)) paste(to_drop, collapse = ", ") else "<none>")
```

```
Offenders (post CC): <none>
```

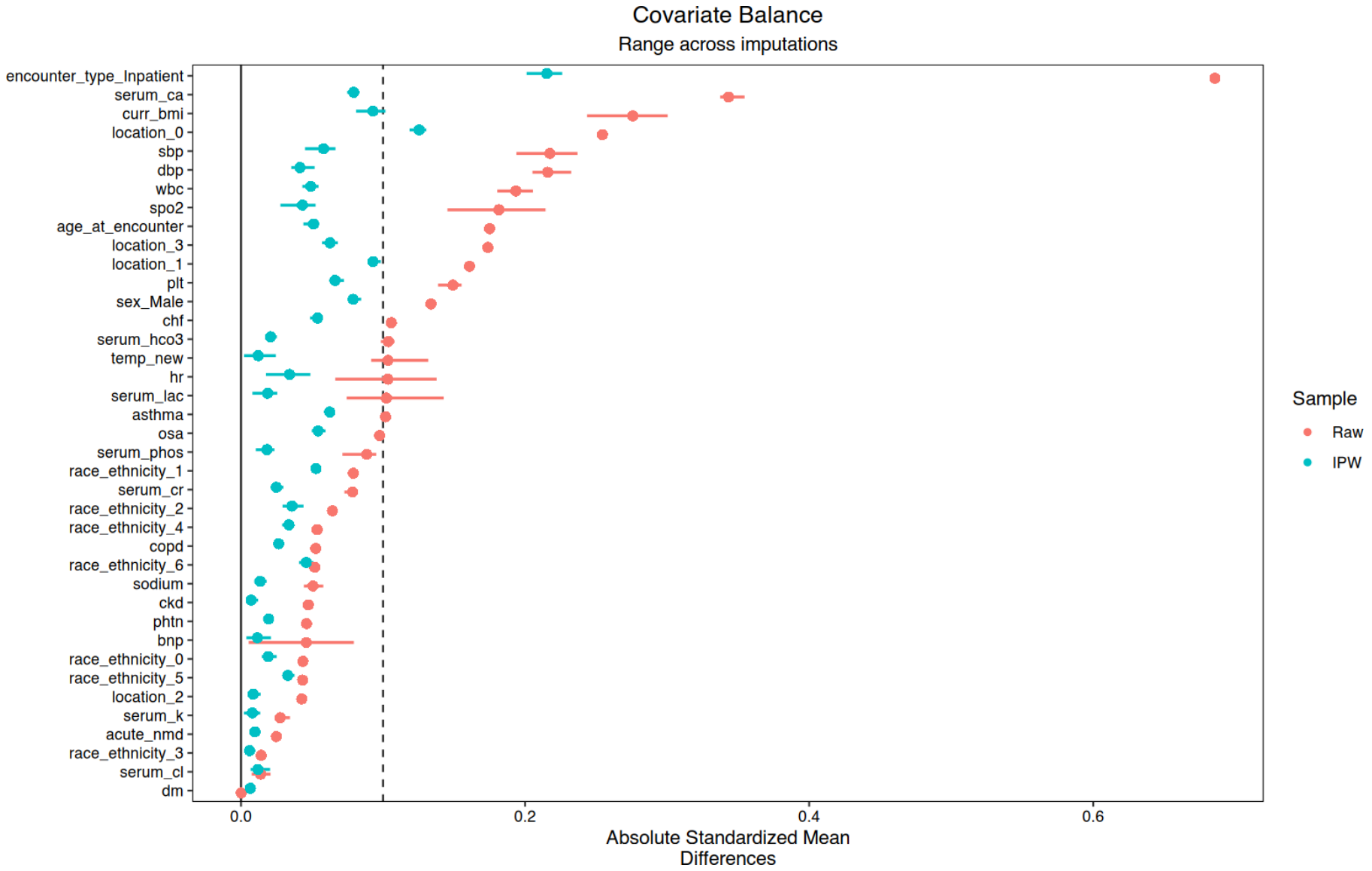
```
# Keep only variables that never collapse post-CC
vars_keep2 <- setdiff(vars0, to_drop)
stopifnot(length(vars_keep2) > 0)
```

Chunk unnamed-chunk-1 runtime: 0.05 s

```
# Build a variable set that has  2 levels in *every* imputation (prevents contrasts errors)
vary_ok <- function(z) {
  nz <- z[!is.na(z)]
  if (is.factor(nz)) nlevels(droplevels(nz)) > 1 else dplyr::n_distinct(nz) > 1
}
vars_keep <- Reduce(intersect, lapply(dlist, function(d) {
  keep <- vapply(d[, covars_gbm, drop = FALSE], vary_ok, logical(1))
  names(keep)[keep]
}))

# Long data for cobalt with weights and imputation id
make_long_for_cobalt <- function(dlist, W_list, treat_var, covars) {
  stopifnot(length(dlist) == length(W_list))
  do.call(rbind, lapply(seq_along(dlist), function(i) {
    di <- dlist[[i]][, c(treat_var, covars), drop = FALSE]
```

```
    di$.imp <- i
    di$.w   <- W_list[[i]]$weights
    di
  }))
}
dlong_abg <- make_long_for_cobalt(dlist, W_abg_list, "has_abg", vars_keep)

# removes empty levels introduced by the per-imputation slicing and prevents spurious contrast errors.
dlong_abg <- droplevels(dlong_abg)

# Final guard: drop any factor that is 1-level in the long frame (should be none after vars_keep)
one_level_factors <- names(Filter(function(x) is.factor(x) && nlevels(droplevels(x)) < 2,
                                  dlong_abg[vars_keep]))
if (length(one_level_factors)) {
  message("Dropping 1-level factors in long data: ", paste(one_level_factors, collapse = ", "))
  vars_keep <- setdiff(vars_keep, one_level_factors)
}

# Balance with imputation identifiers
fml_abg <- reformulate(termlabels = vars_keep, response = "has_abg")
bal_abg <- cobalt::bal.tab(
  fml_abg,
  data        = dlong_abg,
  weights     = dlong_abg$.w,
  imp         = dlong_abg$.imp,      # vector of imputation IDs
  estimand    = "ATE",
  un          = TRUE,
  m.threshold = 0.1,
  binary      = "std",
  s.d.denom   = "pooled"
)

# Optional plot
cobalt::love.plot(
  bal_abg,
  var.order    = "unadjusted",
```

```
  thresholds   = c(m = .1),
  sample.names = c("Raw", "IPW"),
  abs          = TRUE
)
```



Chunk mi-balance-abg runtime: 0.77 s

4.4.3 10.3 VBG propensity (has_vbg)

```
fit_vbg_one <- function(d) {
  w <- weightit(
    formula_vbg,
    data   = d[, c("has_vbg", covars_gbm)],
    method = "gbm",
    estimand    = "ATE",
    include.obj = TRUE,
    n.trees           = gbm_params$n.trees,
    interaction.depth = gbm_params$interaction.depth,
    shrinkage         = gbm_params$shrinkage,
    bag.fraction      = gbm_params$bag.fraction,
    cv.folds          = gbm_params$cv.folds,
    stop.method       = gbm_params$stop.method,
    n.cores           = gbm_params$n.cores
  )
  ww <- w$weights; ww <- ww/mean(ww); cut <- stats::quantile(ww, c(.01,.99), na.rm=TRUE)
  ww <- pmin(pmax(ww, cut[1]), cut[2]); ww <- ww/mean(ww)
  w$weights <- ww
  w
}

with_progress({
  p <- progressor(along = seq_along(dlist))
  W_vbg_list <- future_lapply(
    X = seq_along(dlist),
    FUN = function(i) {
      p(sprintf("Fitting VBG on imputation %d", i))
      set.seed(30251206 + i)
      fit_vbg_one(dlist[[i]])
    },
    future.seed = TRUE
  )
})
```

```
saveRDS(W_vbg_list, "mi_W_vbg_list.rds")
```

Chunk mi-propensity-vbg runtime: 28.98 s

```
        n   min p99.99%   max      ess
[1,] 5111 0.601   4.218 4.219 3538.192
[2,] 5111 0.598   4.152 4.153 3548.530
[3,] 5111 0.598   4.153 4.157 3523.277
[4,] 5111 0.599   4.105 4.105 3555.931
[5,] 5111 0.600   4.069 4.070 3570.105
```

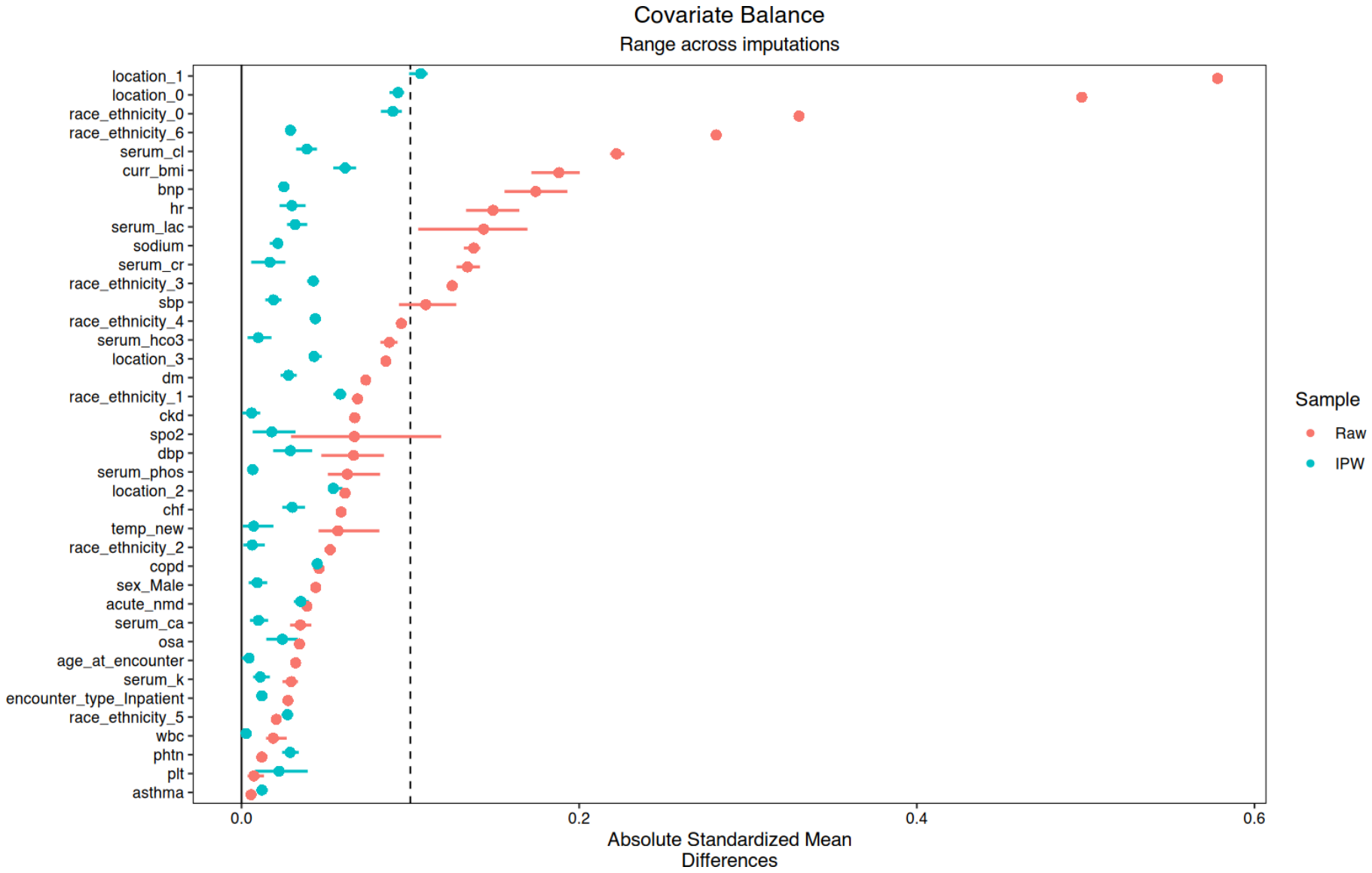
Chunk mi-weight-diagnostics-vbg runtime: 0.01 s

4.4.4 10.4 VBG balance

```
# --- VBG: balance across imputations (parallel to ABG) ------------------------

# Preconditions: dlist (m completed data sets), W_vbg_list (weights per imputation),
#                covars_gbm (covariate set). If W_vbg_list not in memory, load it.
if (!exists("W_vbg_list", inherits = TRUE)) {
  W_vbg_list <- readRDS("mi_W_vbg_list.rds")
}

# Helper(s) if not already defined above
if (!exists("vary_ok", inherits = TRUE)) {
  vary_ok <- function(z) {
    nz <- z[!is.na(z)]
    if (is.factor(nz)) nlevels(droplevels(nz)) > 1 else dplyr::n_distinct(nz) > 1
  }
}
if (!exists("make_long_for_cobalt", inherits = TRUE)) {
  make_long_for_cobalt <- function(dlist, W_list, treat_var, covars) {
    stopifnot(length(dlist) == length(W_list))
```

```
    do.call(rbind, lapply(seq_along(dlist), function(i) {
      di <- dlist[[i]][, c(treat_var, covars), drop = FALSE]
      di$.imp <- i
      di$.w   <- W_list[[i]]$weights
      di
    }))
  }
}

# 1) Keep only covariates that vary ( 2 levels for factors) in every imputation
vars_keep_vbg <- Reduce(intersect, lapply(dlist, function(d) {
  keep <- vapply(d[, covars_gbm, drop = FALSE], vary_ok, logical(1))
  names(keep)[keep]
}))

# 2) Long data (stack imputations), attach weights and imputation id
dlong_vbg <- make_long_for_cobalt(dlist, W_vbg_list, "has_vbg", vars_keep_vbg)
dlong_vbg <- droplevels(dlong_vbg)  # remove empty factor levels from stacking

# 3) Final guard: drop any factor that is 1-level in the stacked frame
one_level_factors_vbg <- names(Filter(function(x) is.factor(x) && nlevels(x) < 2,
                                      dlong_vbg[vars_keep_vbg]))
if (length(one_level_factors_vbg)) {
  message("Dropping 1-level factors in long VBG data: ",
          paste(one_level_factors_vbg, collapse = ", "))
  vars_keep_vbg <- setdiff(vars_keep_vbg, one_level_factors_vbg)
}

# 4) Balance with imputation identifiers
fml_vbg <- reformulate(termlabels = vars_keep_vbg, response = "has_vbg")

bal_vbg <- cobalt::bal.tab(
  fml_vbg,
  data        = dlong_vbg,
  weights     = dlong_vbg$.w,
  imp         = dlong_vbg$.imp,   # vector of imputation IDs
```

```
  estimand    = "ATE",
  un          = TRUE,
  m.threshold = 0.1,
  binary      = "std",
  s.d.denom   = "pooled"
)

# 5) Optional Love plot
cobalt::love.plot(
  bal_vbg,
  var.order    = "unadjusted",
  thresholds   = c(m = .1),
  sample.names = c("Raw", "IPW"),
  abs          = TRUE
)
```



Chunk mi-balance-vbg runtime: 0.53 s

4.5 11) Weighted outcome models within each imputation + pooling

Pool via Rubin's rules using mitools::MIcombine. Extract the coefficient and its variance for the treatment effect from each imputation.

4.5.1 11.1 Helper: fit + extract log-OR and SE from svyglm

```
# --- Robust coefficient extraction from svyglm (handles factor-coded terms) ---
coef_var_from_svy <- function(fit, treat_name) {
  cn <- names(coef(fit))
  idx <- match(treat_name, cn)
  if (is.na(idx)) {
    # handle factor encodings like has_abg1, has_abgYes, etc.
    pat <- paste0("^", gsub("([\\W])", "\\\\$1", treat_name), "(1|Yes|TRUE|Male)?$")
    idx <- grep(pat, cn, perl = TRUE)[1]
  }
  if (is.na(idx)) stop("Treatment coefficient '", treat_name, "' not found in model.")
  b <- unname(coef(fit)[idx])
  V <- unname(vcov(fit)[idx, idx, drop = TRUE])
  if (!is.finite(b) || !is.finite(V)) stop("Non-finite estimate/variance.")
  list(b = b, V = V)
}

# --- Fit one weighted GLM on one completed dataset and extract log-OR + var ---
fit_and_extract <- function(data, weights, formula, treat_name) {
  stopifnot(length(weights) == nrow(data))
  # basic guards: variation in outcome and treatment
  yname <- all.vars(formula)[1L]
  if (dplyr::n_distinct(na.omit(data[[yname]]))   < 2L) stop("Outcome '", yname, "' has one level.")
  if (dplyr::n_distinct(na.omit(data[[treat_name]])) < 2L) stop("Treatment '", treat_name, "' has one level.")
  # design + fit
  data$w <- as.numeric(weights)
  des <- survey::svydesign(ids = ~1, weights = ~w, data = data)
  fit <- survey::svyglm(formula, design = des, family = quasibinomial())
  cv  <- coef_var_from_svy(fit, treat_name)
  list(coef = cv$b, vcov = cv$V)
```

```
}

# --- Pool across imputations; tolerate failures; report how many used ----------
pool_logOR <- function(est_list, term) {
  ok <- vapply(est_list, function(x) is.list(x) && is.finite(x$coef) && is.finite(x$vcov), logical(1))
  est_list <- est_list[ok]
  m_ok  <- length(est_list); m_tot <- length(ok)
  if (m_ok == 0L) {
    return(data.frame(term = term, logOR = NA_real_, SE = NA_real_, OR = NA_real_,
                      LCL = NA_real_, UCL = NA_real_, m_used = 0L, m_total = m_tot))
  }
  results   <- lapply(est_list, function(x) setNames(c(x$coef), term))
  variances <- lapply(est_list, function(x) { M <- matrix(x$vcov, 1, 1); dimnames(M) <- list(term, term); M })
  pooled <- mitools::MIcombine(results = results, variances = variances)
  est <- as.numeric(coef(pooled))
  se  <- sqrt(diag(pooled$variance))
  data.frame(term = term, logOR = est, SE = se, OR = exp(est),
             LCL = exp(est - 1.96 * se), UCL = exp(est + 1.96 * se),
             m_used = m_ok, m_total = m_tot, row.names = NULL)
}
```

Chunk mi-pool-helpers runtime: 0.00 s

4.5.2 11.2 ABG: outcomes = IMV, NIV, Death(60d), Hypercapnic RF

```
# Parallel ABG outcome fits and pooling across imputations
library(future.apply)
library(progressr)

# Inputs assumed present: imp, W_abg_list
stopifnot(exists("imp"), exists("W_abg_list"))
dlist <- mice::complete(imp, action = "all")
stopifnot(length(W_abg_list) == length(dlist))
```

```r
outcomes_abg <- list(
  imv_proc = imv_proc ~ has_abg,
  niv_proc = niv_proc ~ has_abg,
  death    = death_60d ~ has_abg,
  hcrf     = hypercap_resp_failure ~ has_abg
)

old_plan <- future::plan()
workers  <- max(1L, as.integer(future::availableCores() - 1L))
future::plan(multisession, workers = workers)
on.exit(future::plan(old_plan), add = TRUE)

abg_results <- NULL
progressr::with_progress({
  p <- progressr::progressor(steps = length(outcomes_abg) * length(dlist))
  abg_results <- lapply(names(outcomes_abg), function(nm) {
    fml <- outcomes_abg[[nm]]
    ests <- future.apply::future_lapply(
      X = seq_along(dlist),
      FUN = function(i) {
        p(sprintf("ABG: %s (imp %d)", nm, i))
        tryCatch(
          fit_and_extract(dlist[[i]], W_abg_list[[i]]$weights, fml, "has_abg"),
          error = function(e) NULL
        )
      },
      future.seed = TRUE
    )
    out <- pool_logOR(ests, term = "has_abg")
    out$outcome <- nm
    out
  })
})
abg_results <- dplyr::bind_rows(abg_results)[, c("outcome","term","logOR","SE","OR","LCL","UCL","m_used","m_total")]
abg_results
```

outcome	term	logOR	SE	OR	LCL	UCL	m_used	m_total
imv_proc	has_abg	2.0878950	0.1247049	8.067914	6.318446	10.301779	5	5
niv_proc	has_abg	0.5067368	0.1215693	1.659866	1.307950	2.106468	5	5
death	has_abg	0.8956671	0.1016408	2.448969	2.006619	2.988833	5	5
hcrf	has_abg	1.1779851	0.1436109	3.247823	2.451029	4.303645	5	5

Chunk mi-abg-outcomes runtime: 1.33 s

4.5.3 11.3 Repeat for VBG

```
# --- VBG outcomes ------------------------------------------------------------
# Parallel VBG outcome fits and pooling across imputations
library(future.apply)
library(progressr)

# Inputs assumed present: imp, W_vbg_list
stopifnot(exists("imp"), exists("W_vbg_list"))
dlist <- mice::complete(imp, action = "all")
stopifnot(length(W_vbg_list) == length(dlist))

outcomes_vbg <- list(
  imv_proc = imv_proc ~ has_vbg,
  niv_proc = niv_proc ~ has_vbg,
  death    = death_60d ~ has_vbg,
  hcrf     = hypercap_resp_failure ~ has_vbg
)

old_plan <- future::plan()
workers  <- max(1L, as.integer(future::availableCores() - 1L))
future::plan(multisession, workers = workers)
on.exit(future::plan(old_plan), add = TRUE)

vbg_results <- NULL
```

```
progressr::with_progress({
  p <- progressr::progressor(steps = length(outcomes_vbg) * length(dlist))
  vbg_results <- lapply(names(outcomes_vbg), function(nm) {
    fml <- outcomes_vbg[[nm]]
    ests <- future.apply::future_lapply(
      X = seq_along(dlist),
      FUN = function(i) {
        p(sprintf("VBG: %s (imp %d)", nm, i))
        tryCatch(
          fit_and_extract(dlist[[i]], W_vbg_list[[i]]$weights, fml, "has_vbg"),
          error = function(e) NULL
        )
      },
      future.seed = TRUE
    )
    out <- pool_logOR(ests, term = "has_vbg")
    out$outcome <- nm
    out
  })
})
vbg_results <- dplyr::bind_rows(vbg_results)[, c("outcome","term","logOR","SE","OR","LCL","UCL","m_used","m_total")]
vbg_results
```

outcome	term	logOR	SE	OR	LCL	UCL	m_used	m_total
imv_proc	has_vbg	0.5572607	0.1019906	1.745883	1.4295491	2.132217	5	5
niv_proc	has_vbg	0.3183479	0.1320000	1.374855	1.0614415	1.780809	5	5
death	has_vbg	0.1170636	0.1051072	1.124191	0.9148946	1.381367	5	5
hcrf	has_vbg	0.8619008	0.1353847	2.367657	1.8158377	3.087170	5	5

Chunk mi-vbg-outcomes runtime: 1.32 s

4.6 12) Explainability on one representative imputation

To manage runtime, compute SHAP/PDP/ALE on the first imputed dataset and its fitted GBM(s).

Chunk shap-axis-labels runtime: 0.00 s

```
# --- Fast SHAP for WeightIt GBM fits (works with MI) -------------------------

library(fastshap)
```

```
Attaching package: 'fastshap'

The following object is masked from 'package:dplyr':

    explain
```

```
library(shapviz)
# --- Robust SHAP backend for WeightIt GBM fits -------------------------------
# Works whether gbm$var.names are raw feature names (factors ok) or one-hot names
# like "sexFemale", "race_ethnicity3", "encounter_typeInpatient", etc.

# Map of factor levels observed at training time (for raw-factor path)
.train_levels <- function(df) {
  f <- vapply(df, is.factor, logical(1))
  lapply(df[f], levels)
}

# Align factors in 'df' to a stored levels map (keeps order, avoids new/ambiguous levels)
.align_to_levels <- function(df, levels_map) {
  out <- as.data.frame(df, stringsAsFactors = FALSE)
  for (nm in intersect(names(levels_map), names(out))) {
    out[[nm]] <- factor(as.character(out[[nm]]), levels = levels_map[[nm]])
  }
  out
}

# Build a design with columns EXACTLY equal to 'varnames' by deriving indicators
# from the raw covariate frame 'X_raw'. This covers one-hot names like "sexMale",
```

```r
# "race_ethnicity2", "encounter_typeInpatient", etc.
.design_from_varnames <- function(X_raw, varnames) {
  X_raw <- as.data.frame(X_raw, stringsAsFactors = FALSE)

  # normalize odd classes; turn characters into factors (stable level strings)
  for (nm in names(X_raw)) {
    if (inherits(X_raw[[nm]], "haven_labelled")) X_raw[[nm]] <- as.character(X_raw[[nm]])
  }
  X_raw[] <- lapply(X_raw, function(z) if (is.character(z)) factor(z) else z)

  cn    <- colnames(X_raw)
  cn_s  <- make.names(cn)
  vn    <- varnames
  vn_s  <- make.names(vn)

  out <- matrix(NA_real_, nrow = nrow(X_raw), ncol = length(vn))
  colnames(out) <- vn

  for (i in seq_along(vn)) {
    v   <- vn[i]
    v_s <- vn_s[i]

    # Case 1: exact column present
    if (v %in% cn) {
      z <- X_raw[[v]]
      out[, i] <- if (is.factor(z)) as.numeric(z) else as.numeric(z)
      next
    }

    # Case 2: derive dummy = 1{ base == level } from a base column prefix
    # Find the longest raw name that is a prefix of v (sanitized comparison)
    cand <- which(startsWith(v_s, cn_s))
    if (length(cand)) {
      j <- cand[which.max(nchar(cn_s[cand]))]
      base <- cn[j]
      # level is the suffix of v after the base name (unsanitized; preserves case)
```

```
      lev  <- sub(paste0("^", base), "", v)
      x    <- X_raw[[base]]

      # Compare as strings to match labels like "Female", "Inpatient", "0","1",...
      x_chr <- if (is.factor(x)) as.character(x) else as.character(x)
      out[, i] <- as.numeric(x_chr == lev)
      out[is.na(x_chr), i] <- NA_real_
      next
    }

    stop("Cannot construct design column for '", v, "'. ",
         "No matching base variable found in raw covariates.")
  }

  as.data.frame(out, check.names = FALSE)
}

# Unified backend:
#  - If gbm$var.names are raw feature names, pass factors directly (aligning levels).
#  - Otherwise, build a one-hot design with those exact column names and predict on it.
make_shap_backend_any <- function(W) {
  gbm_fit <- if (!is.null(W$obj)) W$obj else if (!is.null(W$info$obj)) W$info$obj else W$info$model.obj
  stopifnot(inherits(gbm_fit, "gbm"))
  best_tree <- if (!is.null(W$info$best.tree)) W$info$best.tree else gbm_fit$n.trees

  X_raw <- as.data.frame(W$covs, stringsAsFactors = FALSE)
  # tidy odd classes; keep factors where they already are
  for (nm in names(X_raw)) {
    if (inherits(X_raw[[nm]], "haven_labelled")) X_raw[[nm]] <- as.character(X_raw[[nm]])
  }
  X_raw[] <- lapply(X_raw, function(z) if (is.character(z)) factor(z) else z)
  X_raw[] <- lapply(X_raw, function(z) if (is.factor(z)) droplevels(z) else z)

  vn <- gbm_fit$var.names
  levels_map <- .train_levels(X_raw)
```

```
  # Path A: raw-factor training (names line up directly)
  if (all(vn %in% colnames(X_raw))) {
    X_use <- .align_to_levels(X_raw[, vn, drop = FALSE], levels_map)
    pred  <- function(object, newdata) {
      nd <- .align_to_levels(newdata, levels_map)
      predict(object, newdata = nd, n.trees = best_tree, type = "link")
    }
    return(list(gbm = gbm_fit, X = X_use, pred = pred, best_tree = best_tree))
  }

  # Path B: dummy-coded training (var.names are one-hot)
  X_use <- .design_from_varnames(X_raw, vn)
  pred  <- function(object, newdata) {
    # If caller already supplies the dummy design, use it; else derive it
    if (all(vn %in% colnames(newdata))) {
      nd <- as.data.frame(newdata)[, vn, drop = FALSE]
    } else {
      nd <- .design_from_varnames(newdata, vn)
    }
    predict(object, newdata = nd, n.trees = best_tree, type = "link")
  }

  list(gbm = gbm_fit, X = X_use, pred = pred, best_tree = best_tree)
}
```

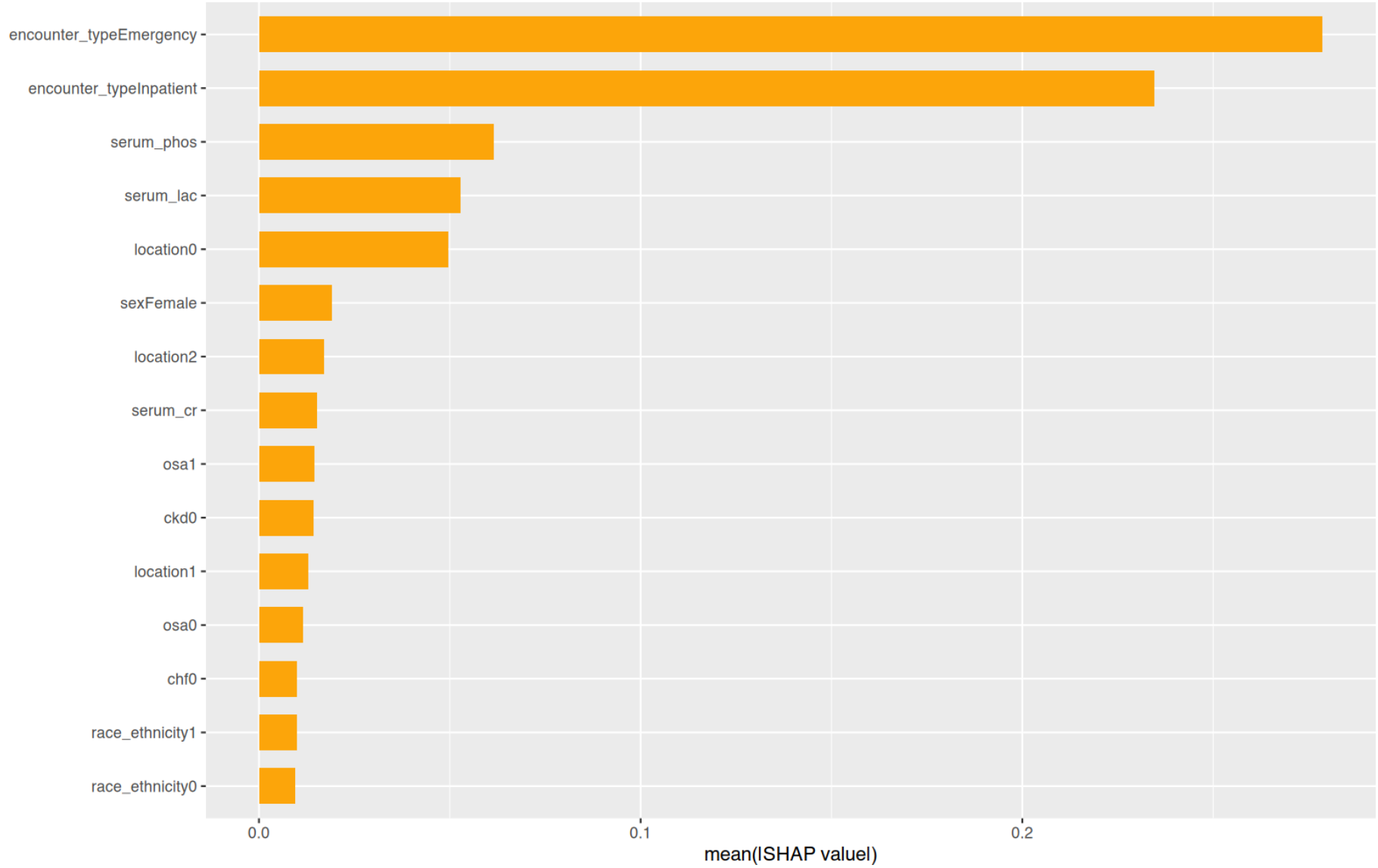
Chunk unnamed-chunk-2 runtime: 0.02 s

```
# Choose one completed dataset's fit
# Device safety (once per doc)
if (!dir.exists("figs")) dir.create("figs", recursive = TRUE, showWarnings = FALSE)
knitr::opts_chunk$set(fig.path = "figs/", dev = "ragg_png", dpi = 200)

# SHAP throttle knobs (single imputation, subsample rows, fewer sims)
shap_cfg <- list(
  frac_rows = 0.15,
  nsim      = 8,
```

```
  top_k     = 20,
  seed      = 123
)

# --- ABG explainability on imputation 1 ------------------------------------
stopifnot(exists("W_abg_list"), length(W_abg_list) >= 1)
W1_abg <- W_abg_list[[1]]
bk_abg <- make_shap_backend_any(W1_abg)

# Equivalence: wrapper vs direct gbm predict on the same design matrix
bk <- make_shap_backend_any(W_abg_list[[1]])
p_wrap <- bk$pred(bk$gbm, bk$X)
p_direct <- predict(
  bk$gbm,
  newdata = bk$X[, bk$gbm$var.names, drop = FALSE],
  n.trees = bk$best_tree,
  type    = "link"
)
stopifnot(mean(abs(p_wrap - p_direct), na.rm = TRUE) < 1e-8)

# Subsample rows for SHAP speed
set.seed(shap_cfg$seed)
ix_abg <- sample.int(nrow(bk_abg$X), max(50L, floor(shap_cfg$frac_rows * nrow(bk_abg$X))))
X_abg  <- bk_abg$X[ix_abg, , drop = FALSE]

# Fast SHAP on link (logit) scale
S_abg <- fastshap::explain(
  object       = bk_abg$gbm,
  X            = X_abg,
  pred_wrapper = bk_abg$pred,
  nsim         = shap_cfg$nsim,
  adjust       = FALSE
)

# shapviz object (X can be data.frame or matrix; keep column names)
sv_abg <- shapviz::shapviz(as.matrix(S_abg), X = as.matrix(X_abg))
```

```r
# Bar importance (top K)
ord_abg   <- order(colMeans(abs(S_abg), na.rm = TRUE), decreasing = TRUE)
topK_abg  <- colnames(S_abg)[ord_abg[1:min(shap_cfg$top_k, ncol(S_abg))]]
p_bar_abg <- shapviz::sv_importance(sv_abg, kind = "bar", v = topK_abg)
p_bar_abg
```

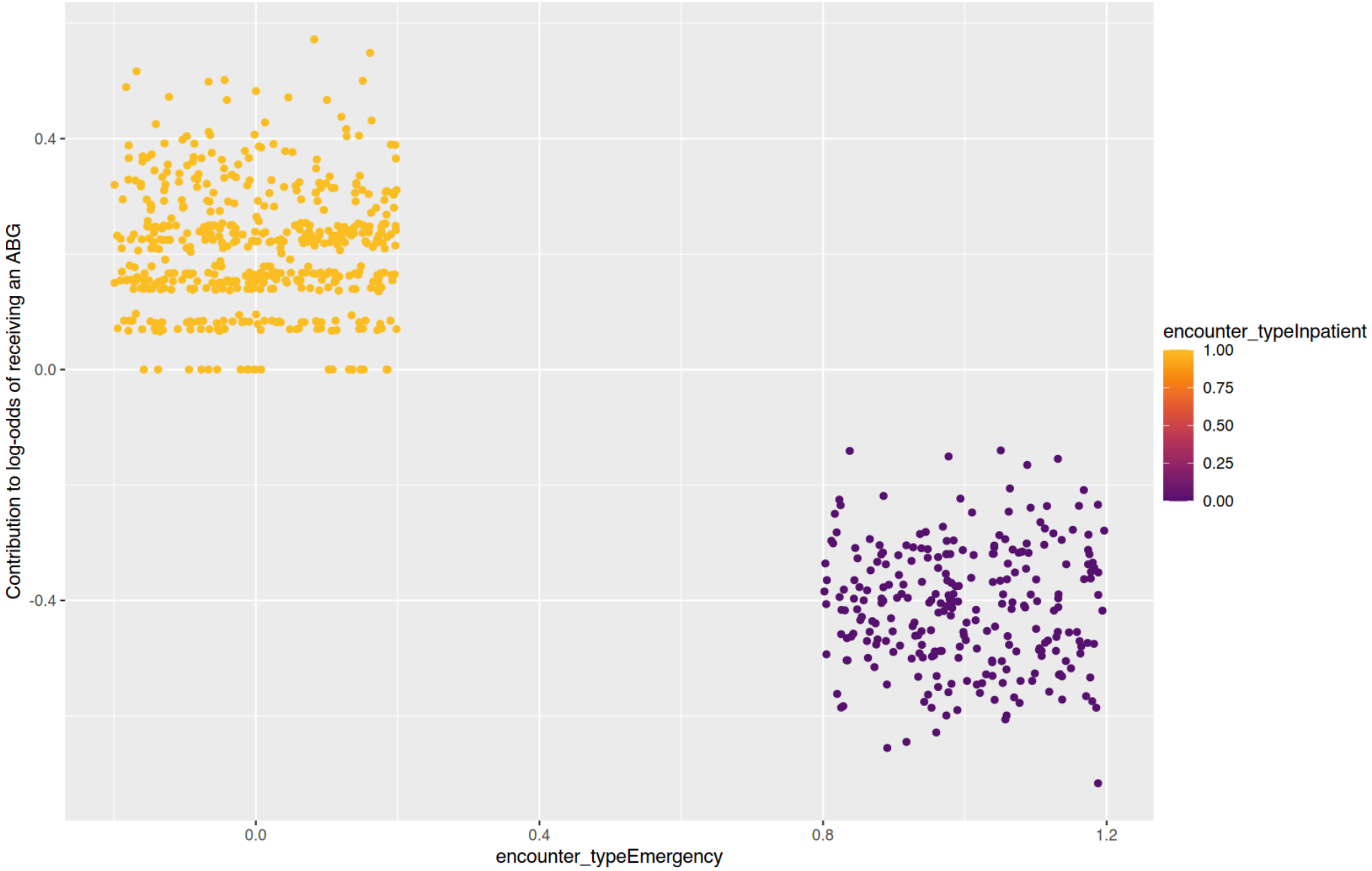
```
Warning: `label` cannot be a <ggplot2::element_blank> object.
```




```
# Dependence: top feature colored by second (add smoother explicitly)
pri_abg <- topK_abg[1]
aux_abg <- topK_abg[2]
# dependence: replace loess with GAM to avoid near-singularity
```

```r
p_dep_abg <- shapviz::sv_dependence(sv_abg, v = pri_abg, color_var = aux_abg) +
  ggplot2::geom_smooth(method = "gam", formula = y ~ s(x, bs = "cs", k = 4),
                       se = FALSE, linewidth = 0.5) +
  ggplot2::labs(y = shap_y_abg, x = pri_abg)

# for importance: don't set label = element_blank() on shapviz's plot; let it draw defaults
# if you see "aesthetics dropped: colour", avoid mapping `colour` in a stat
p_dep_abg
```

```
Warning: Failed to fit group -1.
Caused by error in `smooth.construct.cr.smooth.spec()`:
! x has insufficient unique values to support 4 knots: reduce k.
```

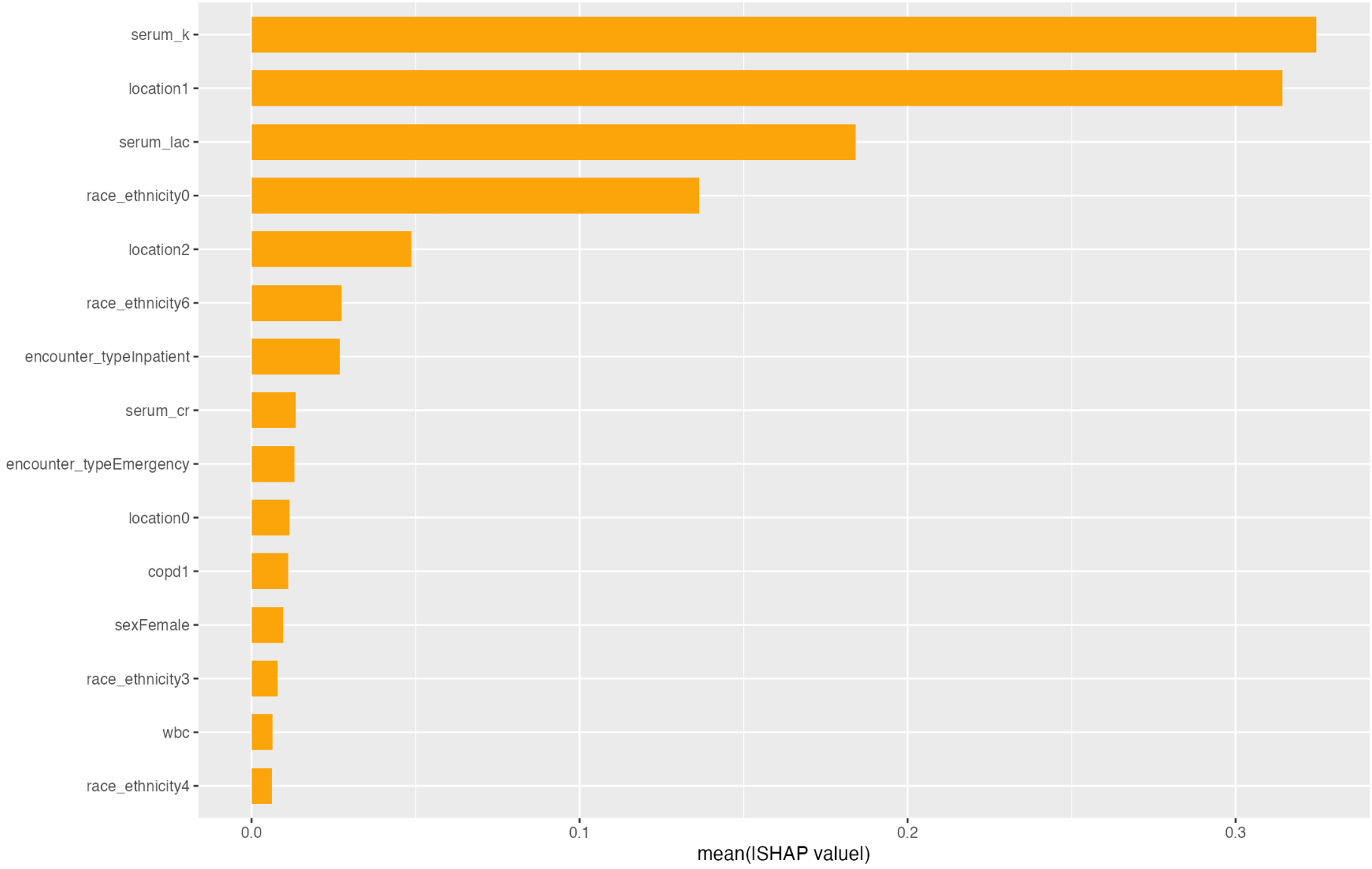


Chunk shap-abg-vbg runtime: 12.19 s

```
# Repeat for VBG
```

```r
# --- VBG explainability on imputation 1 ------------------------------------
stopifnot(exists("W_vbg_list"), length(W_vbg_list) >= 1)
W1_vbg <- W_vbg_list[[1]]

bk_vbg <- make_shap_backend_any(W1_vbg)

set.seed(shap_cfg$seed)
ix_vbg <- sample.int(nrow(bk_vbg$X), max(50L, floor(shap_cfg$frac_rows * nrow(bk_vbg$X))))
X_vbg  <- bk_vbg$X[ix_vbg, , drop = FALSE]

S_vbg <- fastshap::explain(
  object       = bk_vbg$gbm,
  X            = X_vbg,
  pred_wrapper = bk_vbg$pred,
  nsim         = shap_cfg$nsim,
  adjust       = FALSE
)

sv_vbg <- shapviz::shapviz(as.matrix(S_vbg), X = as.matrix(X_vbg))

ord_vbg   <- order(colMeans(abs(S_vbg), na.rm = TRUE), decreasing = TRUE)
topK_vbg  <- colnames(S_vbg)[ord_vbg[1:min(shap_cfg$top_k, ncol(S_vbg))]]
p_bar_vbg <- shapviz::sv_importance(sv_vbg, kind = "bar", v = topK_vbg)
p_bar_vbg
```

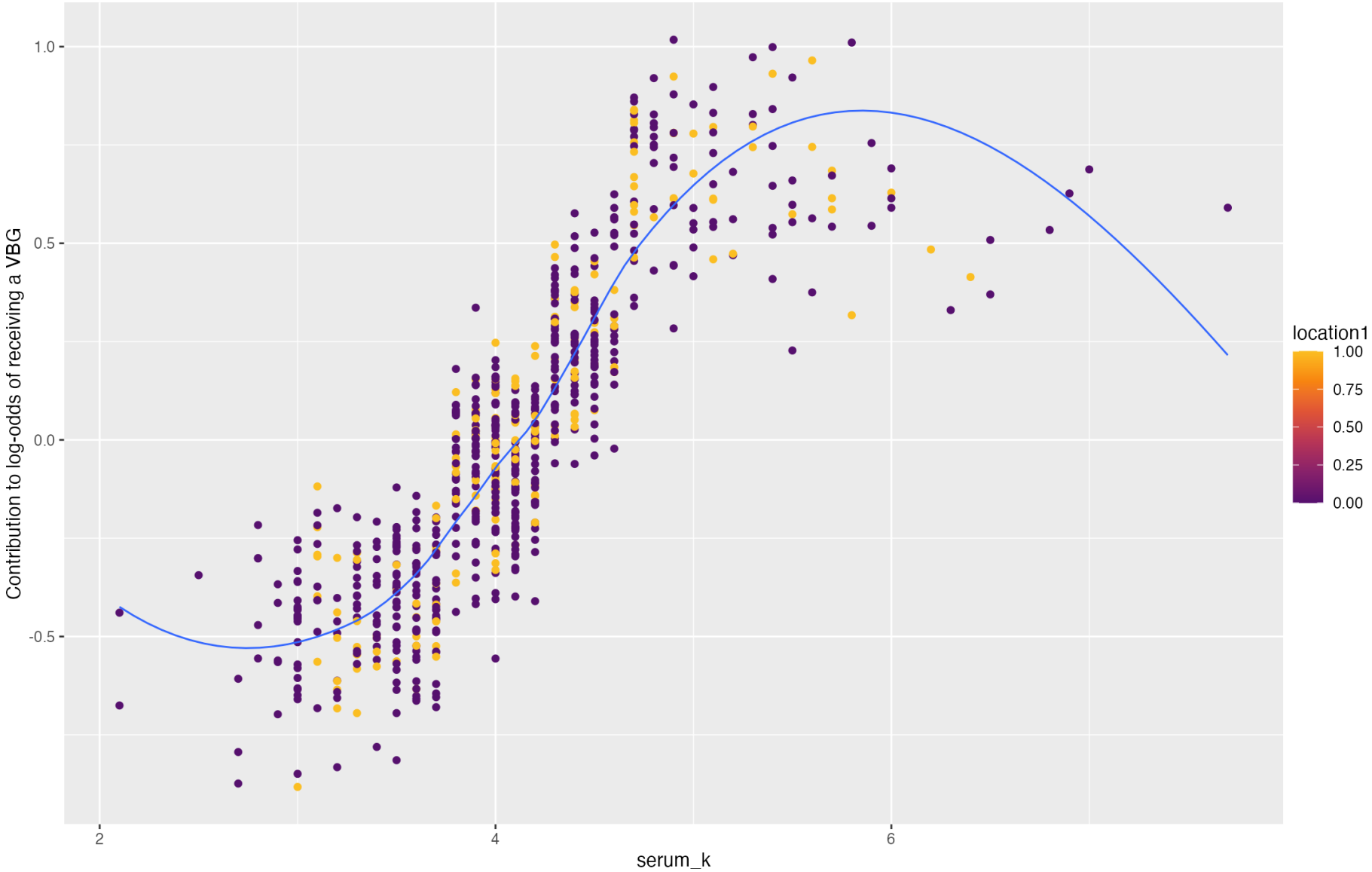
```
Warning: `label` cannot be a <ggplot2::element_blank> object.
```




```
pri_vbg <- topK_vbg[1]
aux_vbg <- topK_vbg[2]
p_dep_vbg <- shapviz::sv_dependence(sv_vbg, v = pri_vbg, color_var = aux_vbg) +
  ggplot2::geom_smooth(se = FALSE, linewidth = 0.5, method = "loess", formula = y ~ x) +
```

```
  ggplot2::labs(y = shap_y_vbg, x = pri_vbg)
p_dep_vbg
```

```
Warning: The following aesthetics were dropped during statistical transformation:
colour.
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?
```



Chunk unnamed-chunk-3 runtime: 11.27 s

4.7 13) Imputed, weighted, three-level PCO2 (ABG & VBG)

```
# --- helpers ------------------------------------------------------------
library(splines)
library(mitools)
library(survey)
library(dplyr)

# Pool (Rubin) any subset of coefficients across imputed fits (glm/svyglm)
pool_terms <- function(fits, term_prefix = NULL, term_pattern = NULL) {
  fits <- Filter(Negate(is.null), fits)
  if (!length(fits)) return(
    data.frame(term = character(), logOR = numeric(), SE = numeric(),
               OR = numeric(), LCL = numeric(), UCL = numeric())
  )

  coef_names <- lapply(fits, function(f) names(stats::coef(f)))
  common <- Reduce(intersect, coef_names)
  if (!is.null(term_prefix))  common <- common[startsWith(common, term_prefix)]
  if (!is.null(term_pattern)) common <- common[grepl(term_pattern, common)]
  if (!length(common)) return(
    data.frame(term = character(), logOR = numeric(), SE = numeric(),
               OR = numeric(), LCL = numeric(), UCL = numeric())
  )

  rows <- lapply(common, function(tt) {
    results <- lapply(fits, function(f) setNames(c(stats::coef(f)[tt]), tt))
    variances <- lapply(fits, function(f) {
      v <- stats::vcov(f)[tt, tt, drop = TRUE]
      m <- matrix(v, 1, 1); dimnames(m) <- list(tt, tt); m
    })
    pooled <- mitools::MIcombine(results = results, variances = variances)
    est <- as.numeric(coef(pooled))
    se  <- sqrt(diag(pooled$variance))
    data.frame(term = tt,
```

```
                logOR = est, SE = se,
                OR  = exp(est),
                LCL = exp(est - 1.96*se),
                UCL = exp(est + 1.96*se),
                row.names = NULL)
  })
  dplyr::bind_rows(rows)
}

# 3-level CO2 category maker; can use fixed clinical cutpoints or data-driven
# --- CO2 category helper (3-level) ------------------------------------------
make_co2_cat <- function(x,
                         fixed_breaks = NULL,
                         labels = c("Eucapnia", "Hypocapnia", "Hypercapnia"),
                         normal = NULL) {
  x <- suppressWarnings(as.numeric(x))
  x_ok <- x[is.finite(x)]
  if (length(x_ok) < 10) {
    return(factor(rep(NA_character_, length(x)), levels = labels))
  }

  brks <- NULL
  # priority: explicit breaks → "normal" window → quantile fallback
  if (!is.null(fixed_breaks)) {
    brks <- fixed_breaks
  } else if (!is.null(normal)) {
    n <- sort(unique(as.numeric(normal)))
    if (length(n) >= 2 && all(is.finite(n)) && (n[2] > n[1])) {
      brks <- c(-Inf, n[1], n[2], Inf)
    }
  }
  if (is.null(brks)) {
    brks <- stats::quantile(x_ok, probs = c(0, 1 / 3, 2 / 3, 1), na.rm = TRUE)
  }

  brks <- unique(brks)
```

```r
  if (length(brks) < 4 || any(diff(brks) <= 0)) {
    return(factor(rep(NA_character_, length(x)), levels = labels))
  }
  cut(x, breaks = brks, include.lowest = TRUE, labels = labels, right = TRUE)
}

# defaults (safe if you didn't predefine)
use_fixed_abg   <- if (exists("use_fixed_abg")) isTRUE(use_fixed_abg) else FALSE
co2_breaks_abg  <- if (exists("co2_breaks_abg")) co2_breaks_abg else NULL
co2_labels_abg  <- if (exists("co2_labels_abg")) co2_labels_abg else c("Eucapnia","Hypocapnia","Hypercapnia")
ref_label_abg   <- if (exists("ref_label_abg")) ref_label_abg else "Eucapnia"

# (Optional) explicit clinical cutpoints - override by setting use_fixed_* = TRUE
use_fixed_abg <- TRUE
co2_breaks_abg   <- c(-Inf, 45, 55, Inf)
co2_labels_abg   <- c("Eucapnia", "Hypocapnia", "Hypercapnia")
ref_label_abg    <- "Eucapnia"

use_fixed_vbg <- TRUE
co2_breaks_vbg   <- c(-Inf, 50, 60, Inf)
co2_labels_vbg   <- c("Eucapnia", "Hypocapnia", "Hypercapnia")
ref_label_vbg    <- "Eucapnia"

# Fixed spline knots & boundary knots (shared across imputations)
# Use 2-98th percentile boundaries; 2 internal knots (  df=4 total)
make_knots <- function(x) {
  x <- x[is.finite(x)]
  B <- stats::quantile(x, probs = c(0.02, 0.98), na.rm = TRUE)
  K <- stats::quantile(x[x >= B[1] & x <= B[2]], probs = c(1/3, 2/3), na.rm = TRUE)
  list(boundary = unname(B), knots = unname(K))
}
```

Chunk mi_pco2_threelevel runtime: 0.01 s

Chunk mi-co2-cat-checks runtime: 0.03 s

Chunk pool-spline-helpers runtime: 0.00 s

4.8 14) MI + IPW three-level PCO2 (ABG & VBG)

4.8.1 14.1 ABG: MI + IPW, three-level PCO2 outcomes

```
# --- ABG: outcome ~ CO2 category, IPW by W_abg_list --------------------------
# Assumes: dlist, W_abg_list, make_co2_cat(), use_fixed_abg, co2_breaks_abg,
#          co2_labels_abg, ref_label_abg are defined.

fit_abg_cat <- function(outcome_var) {
  fits <- vector("list", length(dlist))
  for (i in seq_along(dlist)) {
    d <- dlist[[i]]
    if (!("paco2" %in% names(d))) { fits[[i]] <- NULL; next }
    d$paco2 <- suppressWarnings(as.numeric(d$paco2))

    g <- with(d, has_abg == 1 & is.finite(paco2))
    if (!any(g)) { fits[[i]] <- NULL; next }

    d2 <- d[g, , drop = FALSE]
    d2$co2_cat <- make_co2_cat(
      d2$paco2,
      fixed_breaks = if (isTRUE(use_fixed_abg)) co2_breaks_abg else NULL,
      labels       = co2_labels_abg,
      normal       = if (exists("co2_breaks_abg", inherits = TRUE)) co2_breaks_abg else c(35, 45)
    )
    d2$co2_cat <- base::droplevels(d2$co2_cat)
    d2$co2_cat <- stats::relevel(d2$co2_cat, ref = ref_label_abg)
    if (nlevels(d2$co2_cat) < 2) { fits[[i]] <- NULL; next }

    w <- W_abg_list[[i]]$weights[g]
    des <- survey::svydesign(ids = ~1, weights = ~w, data = d2)
    fml <- stats::as.formula(sprintf("%s ~ co2_cat", outcome_var))
    fits[[i]] <- survey::svyglm(fml, design = des, family = quasibinomial())
  }
  pool_terms(fits, term_prefix = "co2_cat")
```

```
}

abg_cat_results <- dplyr::bind_rows(
  dplyr::mutate(fit_abg_cat("imv_proc"),              outcome = "IMV"),
  dplyr::mutate(fit_abg_cat("niv_proc"),              outcome = "NIV"),
  dplyr::mutate(fit_abg_cat("death_60d"),             outcome = "Death60d"),
  dplyr::mutate(fit_abg_cat("hypercap_resp_failure"), outcome = "HCRF")
) |>
  dplyr::relocate(outcome)
```

Chunk unnamed-chunk-4 runtime: 0.22 s

4.8.2 14.2 VBG: MI + IPW, three-level PCO2 outcomes

```
# Assumes: dlist, W_vbg_list, make_co2_cat(), use_fixed_vbg, co2_breaks_vbg,
#          co2_labels_vbg, ref_label_vbg are defined.

fit_vbg_cat <- function(outcome_var) {
  fits <- vector("list", length(dlist))
  for (i in seq_along(dlist)) {
    d <- dlist[[i]]
    if (!("vbg_co2" %in% names(d))) { fits[[i]] <- NULL; next }
    d$vbg_co2 <- suppressWarnings(as.numeric(d$vbg_co2))

    g <- with(d, has_vbg == 1 & is.finite(vbg_co2))
    if (!any(g)) { fits[[i]] <- NULL; next }

    d2 <- d[g, , drop = FALSE]
    d2$co2_cat <- make_co2_cat(
      d2$vbg_co2,
      fixed_breaks = if (isTRUE(use_fixed_vbg)) co2_breaks_vbg else NULL,
      labels       = co2_labels_vbg,
      normal       = if (exists("co2_breaks_vbg", inherits = TRUE)) co2_breaks_vbg else c(40, 50)
    )
```

```
    d2$co2_cat <- base::droplevels(d2$co2_cat)
    d2$co2_cat <- stats::relevel(d2$co2_cat, ref = ref_label_vbg)
    if (nlevels(d2$co2_cat) < 2) { fits[[i]] <- NULL; next }

    w <- W_vbg_list[[i]]$weights[g]
    des <- survey::svydesign(ids = ~1, weights = ~w, data = d2)
    fml <- stats::as.formula(sprintf("%s ~ co2_cat", outcome_var))
    fits[[i]] <- survey::svyglm(fml, design = des, family = quasibinomial())
  }
  pool_terms(fits, term_prefix = "co2_cat")
}

vbg_cat_results <- dplyr::bind_rows(
  dplyr::mutate(fit_vbg_cat("imv_proc"),              outcome = "IMV"),
  dplyr::mutate(fit_vbg_cat("niv_proc"),              outcome = "NIV"),
  dplyr::mutate(fit_vbg_cat("death_60d"),             outcome = "Death60d"),
  dplyr::mutate(fit_vbg_cat("hypercap_resp_failure"), outcome = "HCRF")
) |>
  dplyr::relocate(outcome)
```

Chunk unnamed-chunk-5 runtime: 0.19 s

```
# After re-running MICE:
dlist <- mice::complete(imp, action = "all")

# 1) must exist and be numeric
stopifnot(all(c("paco2","vbg_co2") %in% names(dlist[[1]])))
stopifnot(is.numeric(dlist[[1]]$paco2), is.numeric(dlist[[1]]$vbg_co2))

# 2) confirm at least two PaCO2 levels among those with ABG in each imputation
table(vapply(dlist, function(d) dplyr::n_distinct(d$paco2[d$has_abg == 1 & is.finite(d$paco2)]), integer(1)) > 1)
```

```
TRUE
   5
```

```r
# 3) smoke test the ABG category fit on the first imputation
tmp <- fit_abg_cat("imv_proc"); print(tmp)
```

```
                term      logOR        SE        OR       LCL      UCL
1  co2_catHypocapnia -0.1755016 0.1676418 0.8390361 0.6040614 1.165414
2 co2_catHypercapnia  0.5573834 0.1694546 1.7460977 1.2526387 2.433948
```

Chunk unnamed-chunk-6 runtime: 0.06 s

4.8.3 14.3 Visualization: pooled three-level ORs

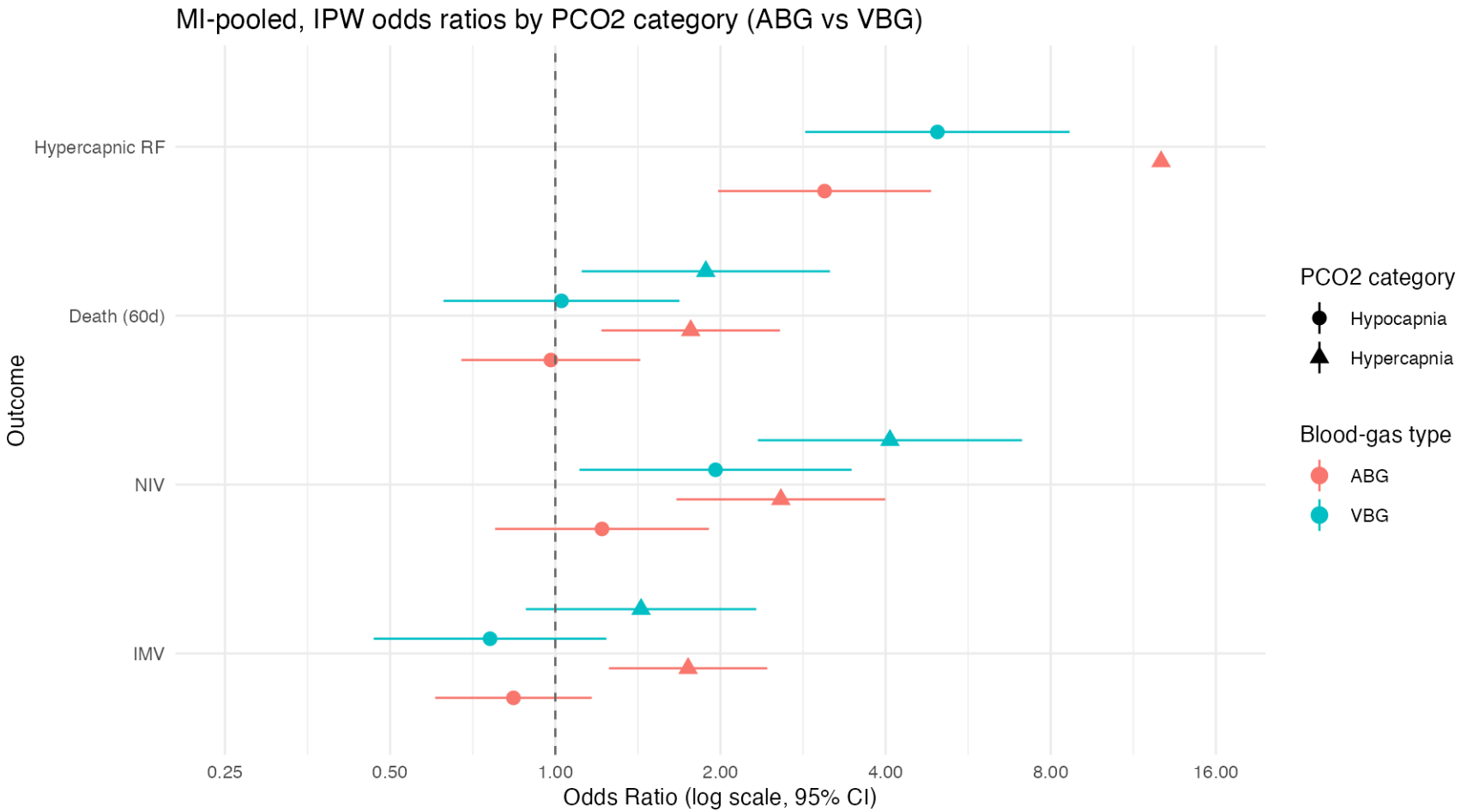
```r
library(dplyr)
library(survey)
library(ggplot2)
library(scales)
library(purrr)
library(mitools)

# --- Pre-flight ------------------------------------------------------------
if (!exists("dlist")) dlist <- mice::complete(imp, action = "all")
stopifnot(length(W_abg_list) == length(dlist),
          length(W_vbg_list) == length(dlist))

# --- Pooling helper for term-level log-ORs across imputations --------------
pool_terms <- function(fits, term_prefix) {
  fits <- Filter(Negate(is.null), fits)
  if (length(fits) == 0L) {
    return(tibble::tibble(term=character(), logOR=numeric(), SE=numeric(),
                          OR=numeric(), LCL=numeric(), UCL=numeric()))
  }
  terms_list <- lapply(fits, function(f) names(stats::coef(f)))
  common     <- Reduce(intersect, terms_list)
  keep_terms <- grep(paste0("^", term_prefix), common, value = TRUE)
```

```r
  if (!length(keep_terms)) {
    return(tibble::tibble(term=character(), logOR=numeric(), SE=numeric(),
                          OR=numeric(), LCL=numeric(), UCL=numeric()))
  }

  purrr::map_dfr(keep_terms, function(term) {
    res <- lapply(fits, function(f) setNames(c(stats::coef(f)[term]), term))
    vars <- lapply(fits, function(f) {
      V <- stats::vcov(f)
      m <- matrix(V[term, term], 1, 1); dimnames(m) <- list(term, term); m
    })
    pooled <- mitools::MIcombine(results = res, variances = vars)
    est <- as.numeric(stats::coef(pooled))
    se  <- sqrt(diag(pooled$variance))
    tibble::tibble(
      term  = term,
      logOR = est,
      SE    = se,
      OR    = exp(est),
      LCL   = exp(est - 1.96 * se),
      UCL   = exp(est + 1.96 * se)
    )
  })
}

# --- Per-group runner over imputations (ABG/VBG) -------------------------------
fit_cat_group <- function(group = c("ABG", "VBG"), outcome) {
  group <- match.arg(group)
  fits <- vector("list", length(dlist))

  for (i in seq_along(dlist)) {
    d <- dlist[[i]]

    if (group == "ABG") {
      if (!("paco2" %in% names(d))) { fits[[i]] <- NULL; next }
      d$paco2 <- suppressWarnings(as.numeric(d$paco2))
```

```
    g <- with(d, has_abg == 1 & is.finite(paco2))
    if (!any(g)) { fits[[i]] <- NULL; next }
    d2 <- d[g, , drop = FALSE]
    d2$co2_cat <- make_co2_cat(
      d2$paco2,
      fixed_breaks = if (exists("use_fixed_abg", inherits = TRUE) && isTRUE(use_fixed_abg)) co2_breaks_abg else NULL,
      labels       = if (exists("co2_labels_abg", inherits = TRUE)) co2_labels_abg else c("Eucapnia","Hypocapnia","Hypercapnia
      normal       = if (exists("co2_breaks_abg", inherits = TRUE)) co2_breaks_abg else c(35, 45)
    )
    w <- W_abg_list[[i]]$weights[g]

  } else {
    if (!("vbg_co2" %in% names(d))) { fits[[i]] <- NULL; next }
    d$vbg_co2 <- suppressWarnings(as.numeric(d$vbg_co2))
    g <- with(d, has_vbg == 1 & is.finite(vbg_co2))
    if (!any(g)) { fits[[i]] <- NULL; next }
    d2 <- d[g, , drop = FALSE]
    d2$co2_cat <- make_co2_cat(
      d2$vbg_co2,
      fixed_breaks = if (exists("use_fixed_vbg", inherits = TRUE) && isTRUE(use_fixed_vbg)) co2_breaks_vbg else NULL,
      labels       = if (exists("co2_labels_vbg", inherits = TRUE)) co2_labels_vbg else c("Eucapnia","Hypocapnia","Hypercapnia
      normal       = if (exists("co2_breaks_vbg", inherits = TRUE)) co2_breaks_vbg else c(40, 50)
    )
    w <- W_vbg_list[[i]]$weights[g]
  }

  ref_lab <- if (group == "ABG") {
    if (exists("ref_label_abg", inherits = TRUE)) ref_label_abg else levels(d2$co2_cat)[1]
  } else {
    if (exists("ref_label_vbg", inherits = TRUE)) ref_label_vbg else levels(d2$co2_cat)[1]
  }
  if (!ref_lab %in% levels(d2$co2_cat)) ref_lab <- levels(d2$co2_cat)[1]
  d2$co2_cat <- stats::relevel(base::droplevels(d2$co2_cat), ref = ref_lab)
  if (nlevels(d2$co2_cat) < 2) { fits[[i]] <- NULL; next }

  d2$w <- w
```

```
    des  <- survey::svydesign(ids = ~1, weights = ~w, data = d2)
    fml  <- stats::as.formula(paste0(outcome, " ~ co2_cat"))
    fit  <- try(survey::svyglm(fml, design = des, family = quasibinomial()), silent = TRUE)
    fits[[i]] <- if (!inherits(fit, "try-error")) fit else NULL
  }

  out <- pool_terms(fits, term_prefix = "co2_cat")
  out
}

# --- Run & plot ------------------------------------------------------------
outs <- c("imv_proc", "niv_proc", "death_60d", "hypercap_resp_failure")

abg_df <- purrr::map_dfr(outs, ~ dplyr::mutate(fit_cat_group("ABG", .x),
                                               outcome = .x, group = "ABG"))
vbg_df <- purrr::map_dfr(outs, ~ dplyr::mutate(fit_cat_group("VBG", .x),
                                               outcome = .x, group = "VBG"))
combined <- dplyr::bind_rows(abg_df, vbg_df)

# decode "co2_cat<level>" → exposure
combined <- combined |>
  dplyr::mutate(exposure = gsub("^co2_cat", "", term),
                exposure = factor(exposure, levels = c("Eucapnia","Hypocapnia","Hypercapnia")),
                outcome  = factor(outcome,
                                  levels = c("imv_proc","niv_proc","death_60d","hypercap_resp_failure"),
                                  labels = c("IMV","NIV","Death (60d)","Hypercapnic RF")),
                group    = factor(group, levels = c("ABG","VBG")))

ggplot(
  combined,
  aes(x = outcome, y = OR, ymin = LCL, ymax = UCL, color = group, shape = exposure)
) +
  geom_pointrange(position = position_dodge(width = 0.7), size = 0.6) +
  geom_hline(yintercept = 1, linetype = "dashed", colour = "grey40") +
  scale_y_log10(
    breaks = c(0.25, 0.5, 1, 2, 4, 8, 16),
```

```
  limits = c(0.25, 16),
  labels = scales::number_format(accuracy = 0.01)
) +
coord_flip() +
labs(
  title = "MI-pooled, IPW odds ratios by PCO2 category (ABG vs VBG)",
  x = "Outcome",
  y = "Odds Ratio (log scale, 95% CI)",
  color = "Blood-gas type",
  shape = "PCO2 category"
) +
theme_minimal(base_size = 10)
```



Chunk ipw-three-level-pco2-mi-abg-vbg runtime: 0.51 s

4.9 15) Imputed, weighted spline PCO2 (ABG & VBG)

4.9.1 15.1 ABG, imputed, weighted, spline outcome

```
# Use pooled per-group helpers defined above (fit_cat_group + pool_terms)
outs <- c("imv_proc", "niv_proc", "death_60d", "hypercap_resp_failure")
```

```
abg_cat_results <- purrr::map_dfr(outs, ~ dplyr::mutate(fit_cat_group("ABG", .x),
                                                        outcome = .x, group = "ABG")) |>
  dplyr::relocate(outcome)
abg_cat_results
```

outcome	term	logOR	SE	OR	LCL	UCL	group
imv_proc	co2_catHypocapnia	-0.1755016	0.1676418	0.8390361	0.6040614	1.165414	ABG
imv_proc	co2_catHypercapnia	0.5573834	0.1694546	1.7460977	1.2526387	2.433948	ABG
niv_proc	co2_catHypocapnia	0.1959942	0.2289090	1.2165198	0.7767262	1.905331	ABG
niv_proc	co2_catHypercapnia	0.9468259	0.2235439	2.5775152	1.6630942	3.994713	ABG
death_60d	co2_catHypocapnia	-0.0186159	0.1914436	0.9815563	0.6744584	1.428484	ABG
death_60d	co2_catHypercapnia	0.5683536	0.1911837	1.7653582	1.2136516	2.567862	ABG
hypercap_resp_failure	co2_catHypocapnia	1.1302675	0.2279728	3.0964848	1.9806811	4.840869	ABG
hypercap_resp_failure	co2_catHypercapnia	2.5426350	0.2049342	12.7131255	8.5076340	18.997474	ABG

Chunk unnamed-chunk-7 runtime: 0.20 s

4.9.2 15.2 VBG, imputed, weighted, spline outcome

```
outs <- c("imv_proc", "niv_proc", "death_60d", "hypercap_resp_failure")

vbg_cat_results <- purrr::map_dfr(outs, ~ dplyr::mutate(fit_cat_group("VBG", .x),
                                                        outcome = .x, group = "VBG")) |>
  dplyr::relocate(outcome)
vbg_cat_results
```

outcome	term	logOR	SE	OR	LCL	UCL	group
imv_proc	co2_catHypocapnia	-0.2737668	0.2491733	0.7605094	0.4666637	1.239382	VBG
imv_proc	co2_catHypercapnia	0.3600126	0.2466934	1.4333475	0.8838158	2.324562	VBG
niv_proc	co2_catHypocapnia	0.6724674	0.2914485	1.9590652	1.1065293	3.468445	VBG

outcome	term	logOR	SE	OR	LCL	UCL	group
niv_proc	co2_catHypercapnia	1.4045482	0.2828213	4.0736857	2.3401579	7.091366	VBG
death_60d	co2_catHypocapnia	0.0259946	0.2524763	1.0263355	0.6257159	1.683455	VBG
death_60d	co2_catHypercapnia	0.6318045	0.2658379	1.8810017	1.1171292	3.167196	VBG
hypercap_resp_failure	co2_catHypocapnia	1.6037332	0.2831293	4.9715574	2.8542233	8.659583	VBG
hypercap_resp_failure	co2_catHypercapnia	3.1272827	0.2629414	22.8119096	13.6251543	38.192831	VBG

Chunk unnamed-chunk-8 runtime: 0.14 s

4.9.3 15.3 Visualization

```
library(dplyr)
library(ggplot2)
library(patchwork)
library(splines)
library(survey)
library(purrr)

if (!exists("dlist")) dlist <- mice::complete(imp, action = "all")
stopifnot(length(W_abg_list) == length(dlist),
          length(W_vbg_list) == length(dlist))

# CO2 support sanity: ensure trimmed ranges exist for both groups
rng_abg <- quantile(
  unlist(lapply(dlist, function(d) as.numeric(d$paco2[d$has_abg == 1]))),
  c(.02, .98), na.rm = TRUE
)
rng_vbg <- quantile(
  unlist(lapply(dlist, function(d) as.numeric(d$vbg_co2[d$has_vbg == 1]))),
  c(.02, .98), na.rm = TRUE
)
stopifnot(rng_abg[1] < rng_abg[2], rng_vbg[1] < rng_vbg[2])
```

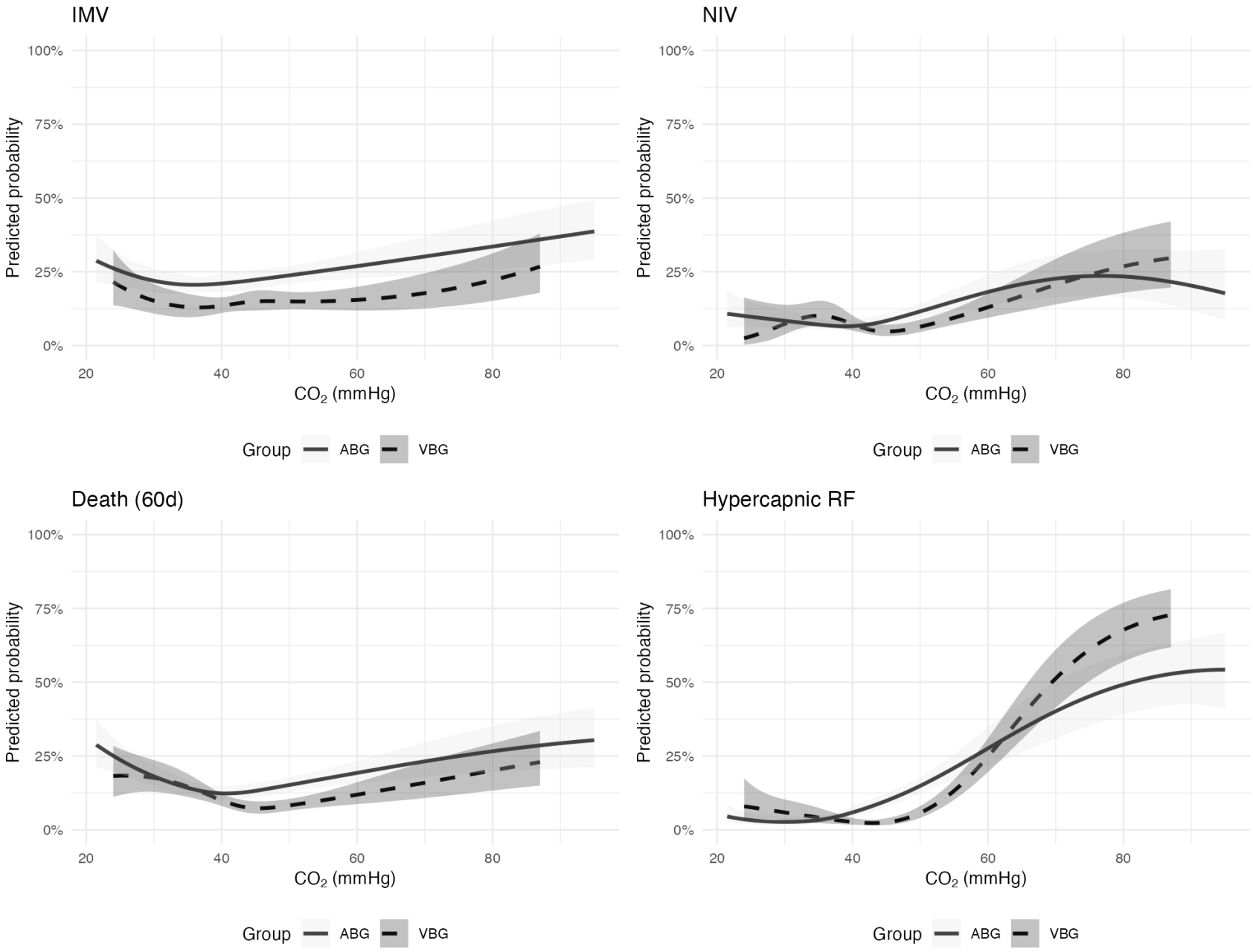
```r
# Predict link for svyglm; fallback to X  and delta method if predict() returns a vector
predict_link_svyglm <- function(fit, newdata) {
  pr <- try(stats::predict(fit, newdata = newdata, type = "link", se.fit = TRUE), silent = TRUE)
  if (!inherits(pr, "try-error") && is.list(pr) && !is.null(pr$fit)) {
    return(list(fit = as.numeric(pr$fit), se.fit = as.numeric(pr$se.fit)))
  }
  # manual:   = X ; Var( ) = X V X^T
  X <- stats::model.matrix(stats::delete.response(stats::terms(fit)), newdata)
  beta <- stats::coef(fit)
  eta  <- drop(X %*% beta)
  V    <- stats::vcov(fit)
  se   <- sqrt(rowSums((X %*% V) * X))
  list(fit = eta, se.fit = se)
}

# Pool predicted curves on the link scale, then transform with logistic
pool_rcs_curve <- function(group = c("ABG","VBG"), outcome,
                           df = 4, grid_n = 220, trim = c(0.02, 0.98)) {
  group <- match.arg(group)

  # global trimmed range for this group
  co2_all <- unlist(lapply(seq_along(dlist), function(i) {
    d <- dlist[[i]]
    if (group == "ABG") as.numeric(d$paco2[d$has_abg == 1])
    else as.numeric(d$vbg_co2[d$has_vbg == 1])
  }), use.names = FALSE)
  co2_all <- co2_all[is.finite(co2_all)]
  stopifnot(length(co2_all) > 0)
  rng  <- as.numeric(stats::quantile(co2_all, probs = trim, na.rm = TRUE))
  if (!is.finite(rng[1]) || !is.finite(rng[2]) || rng[2] <= rng[1]) {
    med <- stats::median(co2_all); rng <- c(med - 1, med + 1)
  }
  xseq <- seq(rng[1], rng[2], length.out = grid_n)

  eta_list <- list()
  var_list <- list()
```

```r
for (i in seq_along(dlist)) {
  d <- dlist[[i]]

  if (group == "ABG") {
    d$paco2 <- suppressWarnings(as.numeric(d$paco2))
    g <- with(d, has_abg == 1 & is.finite(paco2))
    if (!any(g)) next
    d2 <- d[g, , drop = FALSE]
    d2$co2 <- d2$paco2
    w  <- W_abg_list[[i]]$weights[g]
  } else {
    d$vbg_co2 <- suppressWarnings(as.numeric(d$vbg_co2))
    g <- with(d, has_vbg == 1 & is.finite(vbg_co2))
    if (!any(g)) next
    d2 <- d[g, , drop = FALSE]
    d2$co2 <- d2$vbg_co2
    w  <- W_vbg_list[[i]]$weights[g]
  }

  d2$w <- w
  des  <- survey::svydesign(ids = ~1, weights = ~w, data = d2)
  fml  <- stats::as.formula(paste0(outcome, " ~ splines::ns(co2, ", df, ")"))
  fit  <- try(survey::svyglm(fml, design = des, family = quasibinomial()), silent = TRUE)
  if (inherits(fit, "try-error")) next

  newd <- data.frame(co2 = xseq)
  pr   <- predict_link_svyglm(fit, newd)
  eta_list[[length(eta_list) + 1L]] <- pr$fit
  var_list[[length(var_list) + 1L]] <- pr$se.fit^2
}

m <- length(eta_list)
if (m == 0L) stop("No successful fits to pool for ", group, " / ", outcome)

ETA <- do.call(cbind, eta_list)   # ngrid × m
```

```
  VAR <- do.call(cbind, var_list)   # ngrid × m

  if (m == 1L) {
    etaBar <- as.numeric(ETA)
    Tvar   <- as.numeric(VAR)
  } else {
    etaBar <- rowMeans(ETA)
    Wbar   <- rowMeans(VAR)
    B      <- apply(ETA, 1, stats::var)
    Tvar   <- Wbar + (1 + 1/m) * B
  }
  seBar <- sqrt(pmax(Tvar, 0))

  tibble::tibble(
    co2   = xseq,
    yhat  = plogis(etaBar),
    lower = plogis(etaBar - 1.96 * seBar),
    upper = plogis(etaBar + 1.96 * seBar),
    group = group
  )
}

mk_curves <- function(outcome)
  dplyr::bind_rows(
    pool_rcs_curve("ABG", outcome, df = 4, grid_n = 220, trim = c(0.02, 0.98)),
    pool_rcs_curve("VBG", outcome, df = 4, grid_n = 220, trim = c(0.02, 0.98))
  )

cur_imv   <- mk_curves("imv_proc")
cur_niv   <- mk_curves("niv_proc")
cur_death <- mk_curves("death_60d")
cur_hcrf  <- mk_curves("hypercap_resp_failure")

plt_gray <- function(dat, title) {
  ggplot(dat, aes(x = co2, y = yhat, linetype = group)) +
    geom_line(color = "black", linewidth = 1) +
```

```
    geom_ribbon(aes(ymin = lower, ymax = upper, fill = group), alpha = 0.3, color = NA) +
    scale_fill_manual(values = c("ABG" = "gray90", "VBG" = "gray20")) +
    scale_linetype_manual(values = c("ABG" = "solid", "VBG" = "dashed")) +
    scale_y_continuous(limits = c(0, 1),
                       labels = scales::percent_format(accuracy = 1)) +
    labs(title = title,
         x = expression(CO[2]~"(mmHg)"),
         y = "Predicted probability",
         fill = "Group", linetype = "Group") +
    theme_minimal(base_size = 10) +
    theme(legend.position = "bottom")
}

(patchwork::wrap_plots(
   plt_gray(cur_imv,   "IMV"),
   plt_gray(cur_niv,   "NIV"),
   plt_gray(cur_death, "Death (60d)"),
   plt_gray(cur_hcrf,  "Hypercapnic RF"),
   ncol = 2
 ) +
  plot_annotation(
    title = expression(
      paste("MI-pooled, propensity-weighted predicted probability: ABG vs VBG CO"[2],
            " (restricted cubic splines, 2-98% range)")
    )
  ))
```

MI-pooled, propensity-weighted predicted probability: ABG vs VBG CO_2 (restricted cubic splines, 2–98% range)



Chunk ipw-rcs-overlay-mi-abg-vbg runtime: 1.01 s

4.10 16) Save, export, and session info

```
saveRDS(list(abg = abg_results, vbg = vbg_results), "mi_pooled_results.rds")
```

Chunk mi-save-exports runtime: 0.00 s

```
sessionInfo()
```

```
R version 4.5.0 (2025-04-11)
Platform: aarch64-apple-darwin20
Running under: macOS 26.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;  LAPACK version 3.12.1

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

time zone: America/Denver
tzcode source: internal

attached base packages:
 [1] splines   grid      parallel  stats     graphics  grDevices utils
 [8] datasets  methods   base

other attached packages:
 [1] shapviz_0.10.2      fastshap_0.1.1      progressr_0.18.0
 [4] future.apply_1.20.0 future_1.67.0       mitools_2.4
 [7] skimr_2.2.1         visdat_0.6.0        naniar_1.1.0
[10] miceadds_3.18-36    mice_3.18.0         sensitivitymw_2.1
[13] lubridate_1.9.4     tibble_3.3.0        survey_4.4-8
```

```
[16] survival_3.8-3      Matrix_1.7-4        rms_8.0-0
[19] Hmisc_5.2-3         patchwork_1.3.2     officer_0.7.0
[22] modelsummary_2.5.0  scales_1.4.0        labelled_2.15.0
[25] haven_2.5.5         gt_1.1.0            ggplot2_4.0.0
[28] gbm_2.2.2           flextable_0.9.10    dplyr_1.1.4
[31] codebookr_0.1.8     cobalt_4.6.1        broom_1.0.11
[34] WeightIt_1.5.0      purrr_1.2.0         gtsummary_2.4.0
[37] kableExtra_1.4.0

loaded via a namespace (and not attached):
  [1] polspline_1.1.25        datawizard_1.2.0        rpart_4.1.24
  [4] lifecycle_1.0.4         Rdpack_2.6.4            globals_0.18.0
  [7] lattice_0.22-7          MASS_7.3-65             insight_1.4.2
 [10] backports_1.5.0         magrittr_2.0.4          rmarkdown_2.30
 [13] yaml_2.3.10             zip_2.3.3               askpass_1.2.1
 [16] DBI_1.2.3               minqa_1.2.8             RColorBrewer_1.1-3
 [19] multcomp_1.4-28         nnet_7.3-20             TH.data_1.1-4
 [22] sandwich_3.1-1          gdtools_0.4.3           listenv_0.9.1
 [25] cards_0.7.0             MatrixModels_0.5-4      performance_0.15.1
 [28] parallelly_1.45.1       svglite_2.2.1           commonmark_2.0.0
 [31] codetools_0.2-20        xml2_1.4.0              tidyselect_1.2.1
 [34] shape_1.4.6.1           farver_2.1.2            lme4_1.1-38
 [37] effectsize_1.0.1        base64enc_0.1-3         jsonlite_2.0.0
 [40] mitml_0.4-5             Formula_1.2-5           iterators_1.0.14
 [43] emmeans_1.11.2-8        systemfonts_1.2.3       foreach_1.5.2
 [46] tools_4.5.0             ragg_1.5.0              Rcpp_1.1.0
 [49] glue_1.8.0              gridExtra_2.3           pan_1.9
 [52] mgcv_1.9-3              xfun_0.53               chk_0.10.0
 [55] withr_3.0.2             fastmap_1.2.0           boot_1.3-32
 [58] SparseM_1.84-2          openssl_2.3.3           litedown_0.7
 [61] digest_0.6.37           timechange_0.3.0        R6_2.6.1
 [64] estimability_1.5.1      textshaping_1.0.3       colorspace_2.1-2
 [67] markdown_2.0            UpSetR_1.4.0            tidyr_1.3.1
 [70] generics_0.1.4          fontLiberation_0.1.0    data.table_1.17.8
 [73] htmlwidgets_1.6.4       parameters_0.28.2       pkgconfig_2.0.3
 [76] gtable_0.3.6            lmtest_0.9-40           S7_0.2.0
```

```
 [79] htmltools_0.5.8.1       fontBitstreamVera_0.1.1 reformulas_0.4.2
 [82] knitr_1.50              rstudioapi_0.17.1       uuid_1.2-1
 [85] coda_0.19-4.1           checkmate_2.3.3         nlme_3.1-168
 [88] nloptr_2.2.1            repr_1.1.7              zoo_1.8-14
 [91] stringr_1.6.0           foreign_0.8-90          pillar_1.11.1
 [94] vctrs_0.6.5             jomo_2.7-6              xtable_1.8-4
 [97] cluster_2.1.8.1         htmlTable_2.4.3         evaluate_1.0.5
[100] tinytex_0.57            magick_2.9.0            mvtnorm_1.3-3
[103] cli_3.6.5               compiler_4.5.0          rlang_1.1.6
[106] crayon_1.5.3            labeling_0.4.3          plyr_1.8.9
[109] forcats_1.0.1           fs_1.6.6                stringi_1.8.7
[112] viridisLite_0.4.2       tables_0.9.31           glmnet_4.1-10
[115] bayestestR_0.17.0       quantreg_6.1            fontquiver_0.2.1
[118] hms_1.1.4               rbibutils_2.4           xgboost_1.7.11.1
```

Chunk mi-session runtime: 0.06 s