Simulating Operating Characteristics of Longitudinal Markov Ordinal Outcome Trials

Frank Harrell
Department of Biostatistics
Vanderbilt University School of Medicine

2021-02-24

Contents

Overview

In a clinical trial comparing two treatments where only a single type of event is of interest and that event is a terminating event (e.g., death, also called an *absorbing state*), time-to-event comparison of treatments (e.g., using a Cox proportional hazards model) can provide the highest powered analysis. When multiple types of events can occur, or one event can recur, this is not the case. In a COVID-19 therapeutic trial, the events/outcome states of interest may include being at home, being in hospital, being in hospital requiring a ventilator because of respiratory failure, or dying. The majority of clinical trials involving multiple outcome states use a derived quantity as the outcome, such as trials of hospitalized patients using days until the patient was able to go home or using ventilator-free days alive. Derived outcome indexes do not retain the information in the component variables. Greater statistical power, or lower sample sizes for the same power, can be achieved by using all the underlying information used to obtain the derived quantity. A longitudinal statistical model will provide the most efficient treatment comparison while (1) allowing one to estimate the effect of treatment as a function of time, (2) handling aborbing states such as death exactly, and (3) providing derived estimates that are more efficient than computing simple summary statistics on individually patient-derived quantities. For example, in a 28-day study with repeated measurements one can use a longitudinal model to estimate the cumulative incidence of death as a function of time and treatment, the probability of being on a ventilator for more than five days, or the probability of being at home on day 14.

The primary outcome data element is the patient's status on a given day, week, or month. This assessment can be considered to be categorical or ordinal. Taking the outcome as ordinal reduces the number of treatment effect parameters and increases power. To use an ordinal outcome we take a hierarchical approach, i.e., on a given assessment time the outcome is the worst state the patient occupied in that time period. A proportional odds (PO) model is used to model these outcomes. Even when the proportional odds model is strongly violated for treatment, a simple function of the odds ratio (OR) from the model almost exactly reproduces the Wilcoxon statistic or *concordance probability*. The Wilcoxon statistic, when scaled to 0-1 (i.e., the concordance probability) is to within an average absolute error of 0.002 equal to $\frac{\text{OR}^{0.66}}{1+\text{OR}^{0.66}}$. As detailed below, the most important non-PO effect to model is due to the possible change in the mix of outcomes over follow-up time—an effect not necessarily involving treatment. This is modeled using the Peterson and Harrell (1990) *partial proportional odds model*.

To be able to study the performance of univariate, time-to-event, and longitudinal treatment comparisons one must develop a longitudinal model that matches the study design and disease course. As shown here and here, random effects ordinal models cannot adequately fit because of their constraint that the correlation between two outcomes within patient is the same no matter how far apart the two outcomes are assessed. Random effects also do not allow for a patient to stay in the same state for many days in a row. A Markov state transition model provides an excellent fit to the correlation structure. With careful specification, a Markov PO model can be used to simulate realistic clinical trials that have longitudinal ordinal outcomes.

In the simulations shown below, the power gains from using the full information in the data are impressive. For example, in a study with assessments at days 1,2,3,4,7,14,28 the power to detect a transition odds ratio of 0.6 with $n = 600$ was 0.7 whereas the power from comparing time until home from a Cox model was only 0.22 using the same simulated data. For a simulated trial with 28 consecutive days of outcome assessment the powers were 1.0 and 0.77, respectively. For the 7-day design the power of comparing ordinal outcomes on a single day was at best 0.44.

A frequently-used outcome in critical care medicine is ventilator-free days with an override for death, e.g., setting days to -1. For simulating off of VIOLET 2 we use ventilator and ARDS-free days and compare treatment groups using a Wilcoxon test on these strangely-distributed outcome measures. Like time to recovery, ventilator/ARDS-free days is an information-losing one-number summary of a complex process, and loses power. For an odds ratio of 1.3 in the proportional odds Markov model (OR > 1 indicates treatment benefit on the VIOLET 2 outcome scale) and for $n = 600$, the power of the Wilcoxon test on this one-number summary was only 0.30, whereas the power of a Cox test for time to recovery was 0.8. The power of the Markov ordinal model for detecting an odds ratio of 1.3 was 0.94.

Simulations also provide good estimates of relative power and efficiency of longitudinal treatment comparisons

as a function of the number of assessment days. One can also take an outcome-rich study such as VIOLET 2 which followed all alive patients for 28 consecutive days, and selectively remove observations. In an example below it is seen that 28 days of outcome measurements per patient effectively increased the sample size of the study by more than a factor of 5 over assessing the ordinal outcome at a single day per patient.

Introduction

As opposed to time-to-event analysis or comparing outcomes at a single follow-up time, longitudinal ordinal analysis makes optimum use of available data from multiple time periods, resulting in greater Bayesian and frequentist statistical power or smaller sample sizes to achieve the same power as information-losing methods. This report provides a template for simulating operating characteristics for treatment comparisons in a longitudinal study with an ordinal outcome Y where within-patient serial correlation is modeled using a first-order Markov process. This within-patient correlation model is specified by conditioning on the ordinal outcome at the previous time point just as though it were any covariate, with one exception. When time gaps between measurements (which are made on days 1, 3, 7, 14, 28 in one example) are not constant, an interaction term is added to the model so that the influence of the previous state wanes as the time gap increases.

Results for both a frequentist analysis and a Bayesian analysis are provided. Only one data look is taken, at the final sample size. For the Bayesian calculations, the simulations presented here assume that a normal distribution is an adequate approximation to the distribution of the treatment effect estimate, here a treatment B : treatment A transition log odds ratio. The primary data model is a partial proportional odds ordinal logistic model analyzed with the R `VGAM` package `vgam` function[1]. Simulations are done by the `Hmisc` package `simMarkovOrd` and `estSeqMarkovOrd` functions.

The Markov proportional odds ordinal logistic model used here is stated in terms of the probability of being in state y or worse given baseline covariates (including treatment) and the previous state. This state transition model has the following form: $\Pr(Y(t) \geq y | X, Y(t-\delta)) = \text{expit}(\alpha_y + X\beta + f(t, \delta, Y(t-\delta)))$ where $t - \delta$ is the most recent measurement time before current time t, f is a linear model involving a number of regression coefficients that are fitted in addition to α and β, and expit is the logistic cumulative distribution function (inverse logit). When t is the first post-time-zero measurement, $Y(t-\delta)$ is the baseline state.

State transition probabilities are easy to understand, but for study planning and final interpretation we are more used to seeing state occupancy probabilities (SOP). When a state is an absorbing state (e.g., death), the SOP is the cumulative incidence of that state. In general, a SOP is the probability of being in a state $Y = y$ **at** a given time t, whether or not the patient was in a certain state previously. To compute SOPs we un-condition on previous states so that the result is conditional only on the pre-study state and baseline covariates X. In this context, the `Hmisc` function `soprobMarkovOrd` is exemplified. This function computes exact SOPs from the simulation model. The `Hmisc` `intMarkovOrd` function is used to compute the proportional odds model intercepts and other parameters that satisfy specified SOPs for Y=1, 2, 3, 4 at day 28. These parameter values are used in the simulations.

The new functions are in `Hmisc` version 4.4-3. Until this is available from CRAN, source, Linux binary, and Windows binary versions are available at hbiostat.org/R/Hmisc.

As implemented in the R `Hmisc` package `gbayes` function, the Bayesian posterior distribution used to approximate (thus avoiding another simulation loop for MCMC posterior draws) the real posterior distribution is normal, making for very quick approximate posterior probability calculations when the prior distribution used for the log OR is also Gaussian as used here. Three priors are used:

- a skeptical prior for assessing evidence for efficacy
- a flat prior for assessing evidence for efficacy
- a flat prior for studying posterior probabilities of inefficacy/harm

[1] Partial proportional odds is needed for many longitudinal multi-state models because the mix of event types can cross over as time on study progresses. This is handled by adding partial proportional odds terms for time, representing a time $\times Y$ interaction just as when time-dependent covariates are added to a Cox model.

- an optimistic prior for inefficacy/harm

See hbiostat.org/proj/covid19/bayesplan.html and hbiostat.org/stat/irreg.html for more simulations and graphical presentations of them.

The `estSeqMarkovOrd` function also optionally provides, at the last data look only, the Cox proportional hazards χ^2 statistic for treatment for each simulated clinical trial, and the χ^2 statistic for testing proportional hazards in this unadjusted Cox model. The Cox test for treatment is done after the simulated serial ordinal responses for a patient are summarized with the time until achieving Y=1, with this time right-censored at 28 if the patient never achieved Y=1 within 28d, or right-censored earlier if an absorbing state occurs. The purpose of this additional simulation is to compare the frequentist power of a two-sample comparison of time until Y=1 with the frequentist power of the transition odds ratio at day 28 in the Markov proportional odds model, and to gauge the extent to which the time-to-event variable induced by the Markov model operates in proportional hazards across the two treatment groups.

A simulation of 28 consecutive days of data is run, and the effect of sampling only some of the days on statistical efficiency is shown.

This report presents a simulation of frequentist power for comparing univariate ordinal outcomes at a single day of follow-up, for a single odds ratio. Finally, a Markov model is fitted to a previous study with daily ordinal assessments, and the model estimates are translated into a simulation model for creating new repetitions of the study, but at any sample size.

When viewing the `html` version of the report, most of the graphics are interactive in that hovering the pointer over a graphic element will cause details about that element to appear. When viewing pdf output, more output is printed to make up for the lack of interactive drill-down.

Service Functions

This report illustrates several R programming techniques including parallel computing to hasten simulations, and using numerical optimization to reach best compromises to meet design characteristics, here state occupancy and transition probabilities. Also illustrated is the use of a many-to-one hash to sense whether source code or parameters changed since the last time a specific simulation was run, and to only run the simulation if something changed. This can be more effective than the use of the R markdown/`knitr` `cache=TRUE` chunk option because it produces much smaller cache files and gives the user control over which functions should have their source code examined for changes since the last run.

Here we define a number of functions to make repetitive tasks easier and to factor out some formatting processes we may change our mind about. The code for these service functions may be found at hbiostat.org/R/Hmisc/markov/funs.r.

```
outfmt <- if(knitr::is_html_output ()) 'html'  else 'pdf'
markup <- if(knitr::is_latex_output()) 'latex' else 'html'
require(rms)          # automatically engages rms(Hmisc)
```

```
require(data.table)   # used to compute various summary measures
```

```
if(outfmt == 'html') require(plotly)
fdev <- switch(outfmt, html='png', pdf='pdf')
knitrSet(lang='markdown', w=7, h=7, dev=fdev, fig.path=paste0(fdev, '/sim-'))
options(prType=markup)
## If producing html, ggplot2 graphics are converted to plotly graphics
## so that hovertext will show extra information
## Assumes aes(..., label=txt) used
ggp <- if(outfmt != 'html')
  function(ggobject, ...) ggobject else
  function(ggobject, tooltip='label', remove='txt: ', ...) {
```

```
    # Get around a bug in tooltip construction with ggplotly
    # See https://stackoverflow.com/questions/66316337
    g <- ggplotly(ggobject, tooltip=tooltip)
        if(! length(remove) || remove == '') return(g)
      d <- g$x$data
      for(i in 1 : length(d)) {
        w <- d[[i]]$text
      if(length(w)) d[[i]]$text <- gsub(remove, '', w)
          }
    g$x$data <- d
    g
    }

# When outputting to pdf, make LaTeX place figures right at the point
# in which they are created; requires float.sty (see yaml header)
if(outfmt == 'pdf') knitr::opts_chunk$set(fig.pos = 'H', out.extra='')
source('funs.r')
# Some of the functions in funs.r are also found in Hmisc::markupSpecs$markdown:
# md <- markupSpecs$markdown
# squote <- md$squote  # pull functions from markupSpecs$markdown$squote etc.
# equote <- md$equote
# pr     <- md$pr
```

Simulation Parameters

The treatment effect on transition probabilities, on the log odds ratio scale, is taken to be linear in time, i.e., the treatment effect is zero on day 1 and linearly increases to its maximum on day 28. This maximum value is used in much of the graphical output and in specifying the underlying true treatment effect. For each true OR we simulate 1000 clinical trials. ORs will vary from benefit to harm over this sequence: 0.4, 0.5, 0.6, ..., 1.0, 1.25. The sample size is 600 patients. Frequentist power is computed at $\alpha = 0.05$. Posterior probabilities will be computed for the following assertion and prior distribution combinations:

- Efficacy: P(OR < 1) with a skeptical prior P(OR < 0.5) $= 0.05$
- Efficacy: P(OR < 1) with a flat prior (log(OR) has mean 0 and SD 100)
- Inefficacy/harm: P(OR > 1) with a flat prior (log(OR) has mean 0 and SD 100)
- Inefficacy/harm: P(OR > 1) with an optimistic prior (log(OR) has mean log(0.85) = -0.1625 and SD of 0.5)

Simulation Model Specification

Simulation Model Components

This report addresses a 4-level ordinal outcome Y with levels $y = 1, 2, 3, 4$. These levels are numeric for simplicity but the numeric values are never used. We might just as well have coded $y = a, b, c, d$. We initially use measurement times $t = 1, 2, 3, 7, 14, 28$.

Careful specification of longitudinal simulation models results in realistic simulated data that are useful for computing sample sizes and operating characteristics. For Markov transition models there are several components to consider:

- Aspects which are in common with all longitudinal models: choice of measurement times, treatment allocation ratio, and baseline covariate distributions.
- The distribution of a special baseline covariate: the initial pre-randomization patient state $Y(0)$

- The intercepts α_y in the proportional odds model corresponding to $y = 2, 3, 4$ (there is no intercept for $\Pr(Y \geq 1)$ since this probability is 1.
- The treatment effect β which is under our control, and is the log odds ratio for state transition probabilities. Our simulation model assumes that the treatment effect (log odds ratio for group 2 : group 1) is zero at the first follow-up measurement $t = 1$ and that linearly increases to the last day of follow-up, day 28. So the treatment effect is $\beta\frac{t-1}{27}$ for simulating data. For model fitting with real data we would estimate two parameters: the treatment main effect and the treatment $\times t$ interaction effect. Were the treatment $\times t$ interaction not linear there would be more terms. Simulations below explore the power loss from adding an unnecessary quadratic term to this part of the model.
- Imposition of some states being absorbing so that the patient remains in that state forever with probability 1. Our examples take $y = 4$ to be an absorbing state (e.g., death).
- The effects of the previous state on the log odds that the current state is $\geq y$; this is how within-subject dependence is modeled. There are three possible previous states: $y = 1, 2, 3$ since once $y = 4$ is reached there are no more records for the patient. On the log odds scale we write this part of the model as $\tau_1[y' = 2] + \tau_2[y' = 3]$, where $y' = Y(t - \delta)$, the τs are to be determined and $[c]$ denotes 1 if c holds, 0 otherwise.
- When the spacing being measurement times is not constant, we need to model how the influence of the previous state may wane as the time gap from that measurement increases. We chose the simulation model to assume no discounting when the gap is 1 (the gap is 1 when transitioning from the baseline state to $t = 1$) or 2. The gap discounting is assumed to be linear in $\max(\text{gap} - 2, 0)$.
- One state may be semi-absorbing. Here we take $y = 1$ to be such a state. If $y = 1$ corresponds to a patient being back home, we may consider it unlikely that the patient will get sick again and return to $y > 1$. To account for this, we place an additional constraint that the probability of staying at $y = 1$ when the previous state was 1 is 0.9 for treatment group 1.
- A general time trend indicating that the majority of patients tend to get more well regardless of treatment or initial state. This trend is assumed to be linear in our simulations so is specified only as $\kappa(t - 1)$ were proportional odds (PO) to hold. From a Markov analysis of the ORCHID COVID-19 clinical trial, there is moderately strong non-PO with respect to the general time trend. Some events tend to occur early while others tend to occur late, and events may be shuffled out of order. A partial proportional odds model with respect to time will accommodate this.

The only covariate X in our model is treatment. Putting all this together, our Markov transition model is

$$\begin{aligned}\Pr(Y(t) \geq y | Y(t-\delta), X) = \text{expit}(&\alpha_y + \beta\frac{t-1}{27}[X = 2] + \kappa_1(t-1) + \kappa_2(t-1)[y = 3] + \kappa_3(t-1)[y = 4] + \\ &\tau_1[Y(t-\delta) = 2] + \tau_2[Y(t-\delta) = 3] + \\ &\gamma_1 \max(\delta - 2, 0)[Y(t-\delta) = 2] + \gamma_2 \max(\delta - 2, 0)[Y(t-\delta) = 3]\end{aligned}$$

where expit is the inverse logit transformation or $\frac{1}{1+\exp(-x)}$. κ_2 and κ_3 represent non-proportional odds with respect to follow-up time t. With 4 levels of Y there are two possible linear-in-t departures from proportional odds.

Setting Simulation Model Parameter Values

The next step is to specify data characteristics to achieve with the simulation model. We start the process by setting SOPs at $t = 1$ to constants when a patient is in treatment group 1 and starts at a given initial state. We use as our reference an initial state of $y = 2$ which is taken to be the most common state at baseline. For group 1 we ignore β. For $t = 1, \delta = 1, X = 1, Y(0) = 2$ our model reduces to $\text{expit}(\alpha_y + \tau_1)$ If we temporarily assume $\tau_1 = 0$, and desire SOPs at $t = 1$ for group 1 to be $[0.05, 0.70, 0.24, 0.01]$ corresponding to $y = 1, 2, 3, 4$, the cumulative probabilities for $Y \geq y$ for $y = 2, 3, 4$ are 0.95, 0.25, 0.01 and their logits are 2.94, -1.1, -4.6. We will take these as are starting values for α when $Y(0) = 2$.

We must constrain the model using other time points in order to solve for the state dependency parameters.

We take a second time to be $t = 28$, i.e., the last planned follow-up time. Aside from the three αs, these parameters need to be solved for: $\kappa_1, \kappa_2, \kappa_3, \gamma_1, \gamma_2, \tau_1, \tau_2$.

Finally, let's deal with the semi-absorbing nature of $Y = 1$ by forcing $\Pr(Y(14) = 1|Y(7) = 1, X = 1) = 0.9$. We temporarily set $\kappa_2 = \kappa_3 = 0$ and let α_1 denote the first intercept (for $y \geq 2$), $\alpha_1 = \text{logit}(0.95) = 2.944$. Since $\Pr(Y = 1) = 1 - \Pr(Y > 1)$ we set $1 - \text{expit}(\alpha_1 + 13\kappa_1)$ to 0.9, resulting in $\kappa_1 = (\text{logit}(0.1) - \alpha_1)/13$ or -0.39551.

We add the constraint for $y = 1$ transition probabilities by specifying the `ftarget` argument to `intMarkovOrd` as shown below.

Understanding Markov Model Components

To understand how the time and state dependency parameter choices influence the SOPs, let's fix α at the values above and vary the other parameter types one at a time. Define a function g that computes the linear predictor for non-intercept terms. The `h` function defines the default settings for parameters not currently being varied. Note in the output the day 7 to 14 $Y = 1$ transition probability for each combination of parameter values. Also presented for each parameter combination are transition probabilities $\Pr(Y(7) = y|Y(3) = y', X)$ for all 3 values of y' and 4 values of y. The `showtrans` function helps with this. The treatment effect is set to OR=0.1 so that an effect can be seen as early as day 7.

```
times <- c(1, 3, 7, 14, 28)
g <- function(yprev, t, gap, X, parameter=-0.5, extra) {
    tau   <- extra[1:2]
    gamma <- extra[3:4]
    kappa <- extra[5:7]
    lp <- matrix(0., nrow=length(yprev), ncol=3,
                            dimnames=list(as.character(yprev), c('2','3','4')))
    # 3 columns = no. distinct y - 1 = length of intercepts
    # lp[yp, ] is a 3-vector because the kappa components are split out into a 3-vector
    gap <- pmax(gap - 2, 0.)
    for(yp in yprev)
      lp[as.character(yp), ] <- tau[1] * (yp == 2) + tau[2] * (yp == 3) +
          gamma[1] * gap * (yp == 2) + gamma[2] * gap * (yp == 3) +
        (t - 1) * (kappa[1] + c(0., kappa[2], kappa[3])) +
            parameter * (X == 2) * (t - 1) / 27
            lp
}

alpha  <- qlogis(c(0.95, 0.25, 0.01))
kappa1 <- (qlogis(0.1) - alpha[1])/ 13.

h <- function(intercepts=alpha,
              tau=c(-4, 2), gamma=c(1, -1), kappa=c(-0.1, 0, 0),
              t1=3, t2=7, i=1) {
    extra <- c(tau=tau, gamma=gamma, kappa=kappa)
    chkints(times, intercepts, extra, c(kappa=paste(kappa, collapse=' ')))

    # Compute P(Y(14) = 1 | Y(7) = 1, X=1)
    xb <- intercepts + as.vector(g(1, 14, 7, X=1, extra=extra))
    stay1 <- 1. - plogis(xb[1])
    # pr('Extra', extra)
    # pr('P(Y(14)=1 | Y(7)=1)', inline=round(stay1, 3))
    z <- soprobMarkovOrd(1:4, times, initial=2, absorb=4,
                    intercepts=alpha, g=g, X=1, extra=extra)
```

```
    # Convert from matrix to tall and thin data frame
    z <- data.frame(p=as.vector(z),
                    t=as.numeric(rownames(z)[as.vector(row(z))]),
                    y=as.vector(col(z)), i=i)

    ## For each previous state 1-3 at time t1 and for each group compute the
    ## probability that Y=1,2,3,4 at time t2
    r <- NULL
    for(X in 1 : 2) {
        xb <- g(1:3, t2, t2 - t1, X, parameter=log(0.1), extra=extra)
        for(yp in 1 : 3) {
            p <- plogis(intercepts + xb[yp, ])
            p <- c(1., p) - c(p, 0.)
            w <- data.frame(X=X, yp=yp, y=1:4, p=p, i=i)
            r <- rbind(r, w)
        }
    }
    list(sop=z, tprob=r, stay1=stay1)
    }

# Vary one parameter at a time, compute and graph results
vary1 <- function(..., cap) {
    vary <- list(...)
    nam <- names(vary)
    vary <- vary[[1]]
    if(! is.list(vary)) vary <- list(vary)
    labs <- paste0(nam, '=', sapply(vary, pst))
    w <- u <- stay1 <- NULL
    for(i in 1 : length(vary)) {
        j <- structure(list(vary[[i]], i), names=c(nam, 'i'))
        z <- do.call(h, j)
        w <- rbind(w, z$sop)
        u <- rbind(u, z$tprob)
        stay1 <- c(stay1, z$stay1)
    }
    w$label <- factor(w$i, labels=labs)
    u$label <- factor(u$i, labels=labs)
    w$txt <- with(w, paste0(label, '<br>State:', y, '<br>t=', t,
                            '<br>p=', round(p, 3)))

    # u$txt <- with(u, paste0('Group:', X, '<br>', label,
    #               '<br>Y(3):', yp,
    #               '<br>Y(7):',  y, '<br>p=', round(p, 3)))
    # Graph would not simultaneously handle color and linetype
  # with ggplotly.  Use regular ggplot object.

  stayc <- paste0('. P(Y(14)=1 | Y(7)=1) = ',
                  paste(round(stay1, 2), collapse=', '), '.')
    caps <- paste(c('State transition probabilities',
                    'State occupancy probabilities'), cap, stayc)

  g1 <- ggplot(u, aes(x=yp, y=p,
                      color=factor(y), linetype=factor(X))) +
```

```
      geom_line() + facet_wrap(~ label) +
      guides(color    = guide_legend(title='Y'),
                    linetype = guide_legend(title='Group')) +
                xlab('State on Day 3') + ylab('Probability on Day 7')
   g2 <- ggplot(w, aes(x=factor(t), y=p, fill=factor(y),
                label=txt)) +
               geom_col() + facet_wrap(~ label) +
               guides(fill=guide_legend(title='Y')) +
           xlab('t') + ylab('Occupancy Probability')
   r <- list(g1, ggp(g2))
   # Create unique chunk names even if vary1 called multiple times
  # If omit chunk names, LaTeX -> pdf screwed up all the figures
  cnames <- paste0('studyparmsvary1', letters[(nvar1 + 1) : (nvar1 + 2)])
   nvar1  <<- nvar1 + 2
   markupSpecs$html$mdchunk(robj=r, caption=caps, cnames=cnames)
   invisible()
}

nvar1 <<- 0
cap <- 'varying $\\\\gamma$, with $\\\\tau=-4, 2$, $\\\\kappa=-0.1, 0, 0$'
gams <- list(c(-1, 0), c(-1, 1), c(0, 1), c(1, 1))
vary1(gamma=gams, cap=cap)
```

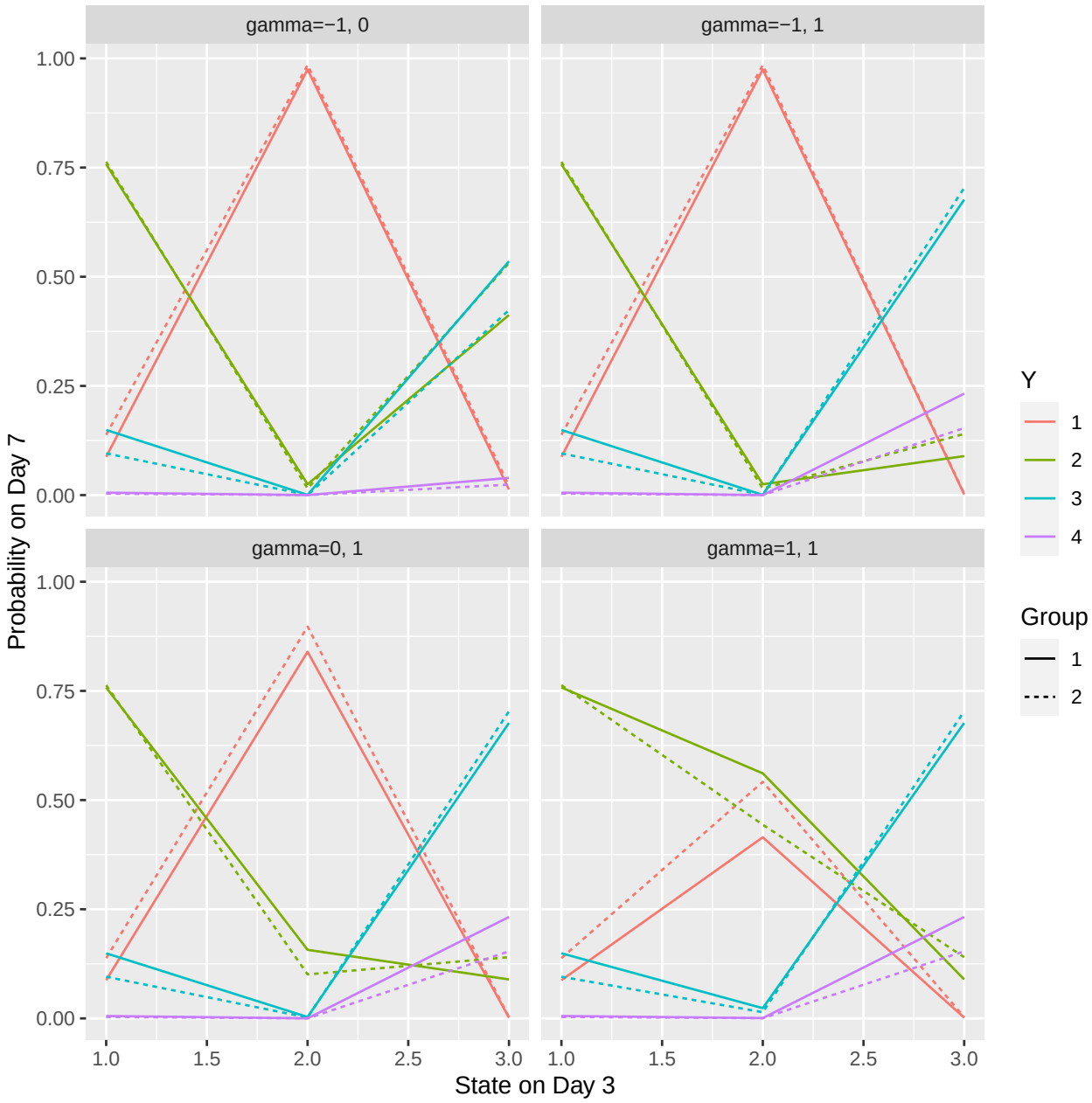

Figure 1: State transition probabilities varying γ, with $\tau = -4, 2$, $\kappa = -0.1, 0, 0$. P(Y(14)=1 | Y(7)=1) = 0.16, 0.16, 0.16, 0.16.

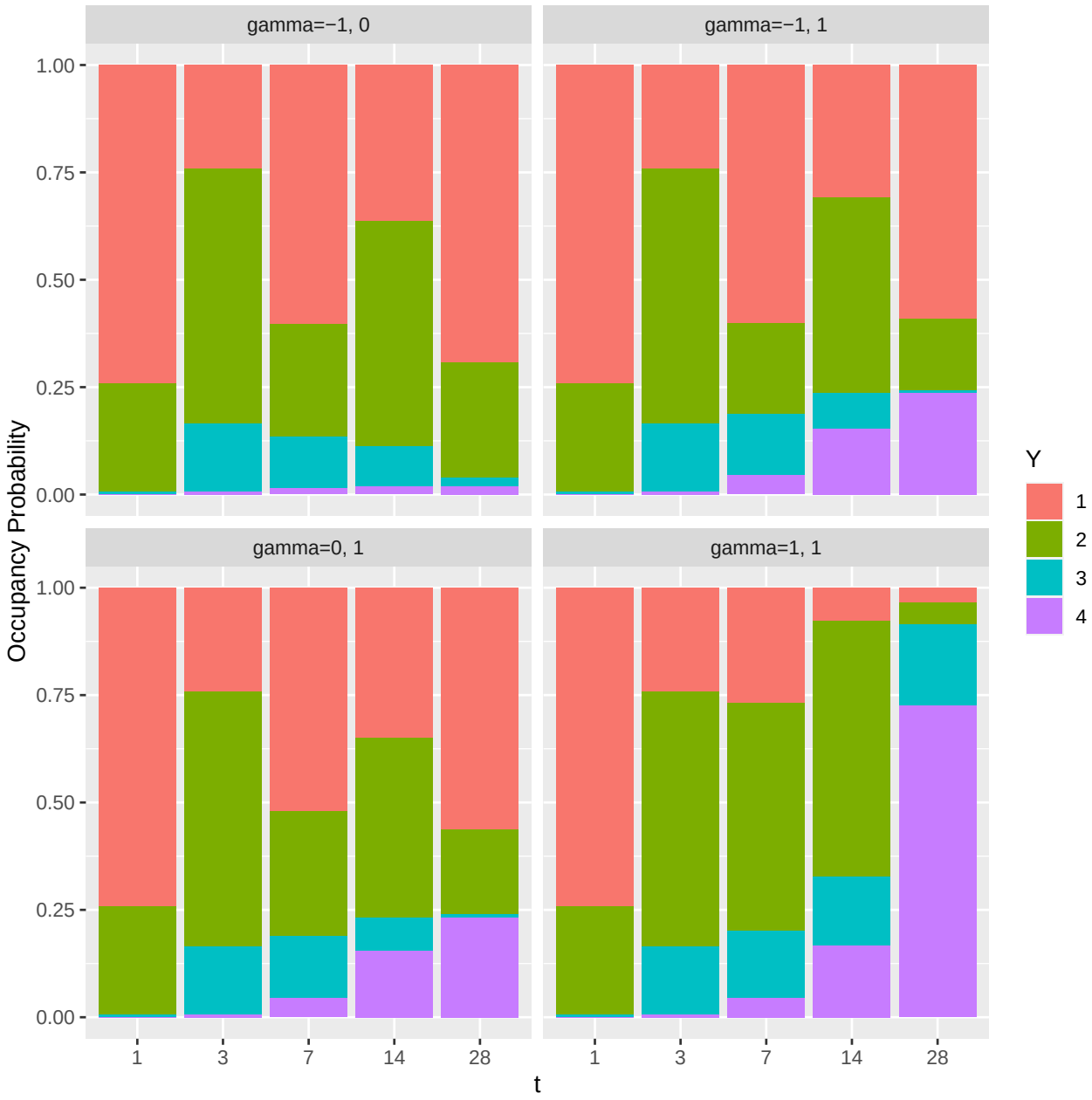

Figure 2: State occupancy probabilities varying γ, with $\tau = -4, 2$, $\kappa = -0.1, 0, 0$. P(Y(14)=1 | Y(7)=1) = 0.16, 0.16, 0.16, 0.16.

```
cap <- 'varying $\\\\kappa$, with $\\\\tau=-4, 2$, $\\\\gamma=1, -1$'
kappas <- list(c(0, 0, 0), c(0.15, 0, 0), c(-.15, 0, 0), c(.15, .1, 0),
                           c(.15, .1, .1))
vary1(kappa=kappas, cap=cap)
```

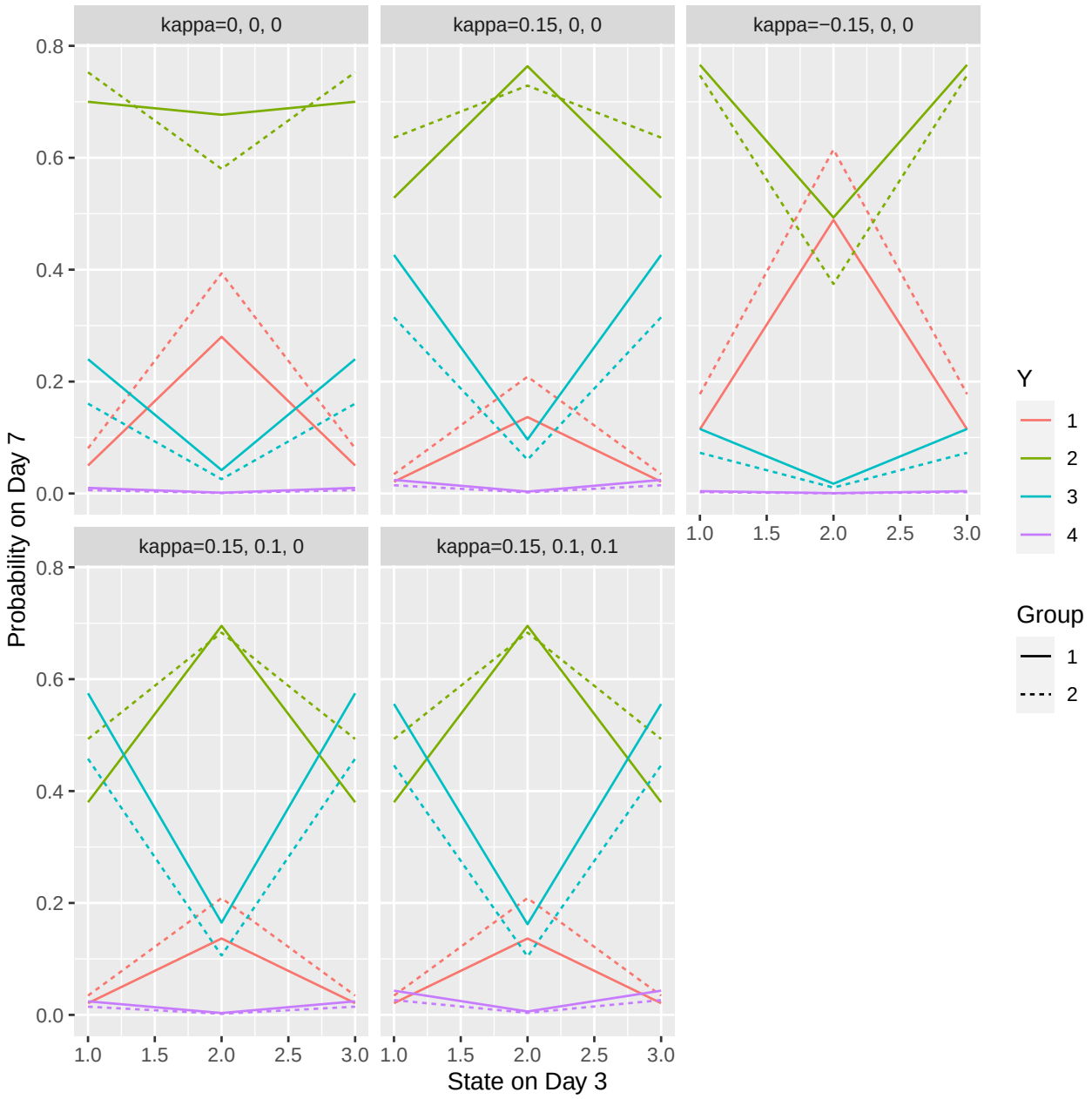

Figure 3: State transition probabilities varying κ, with $\tau = -4, 2$, $\gamma = 1, -1$. P(Y(14)=1 | Y(7)=1) = 0.05, 0.01, 0.27, 0.01, 0.01.

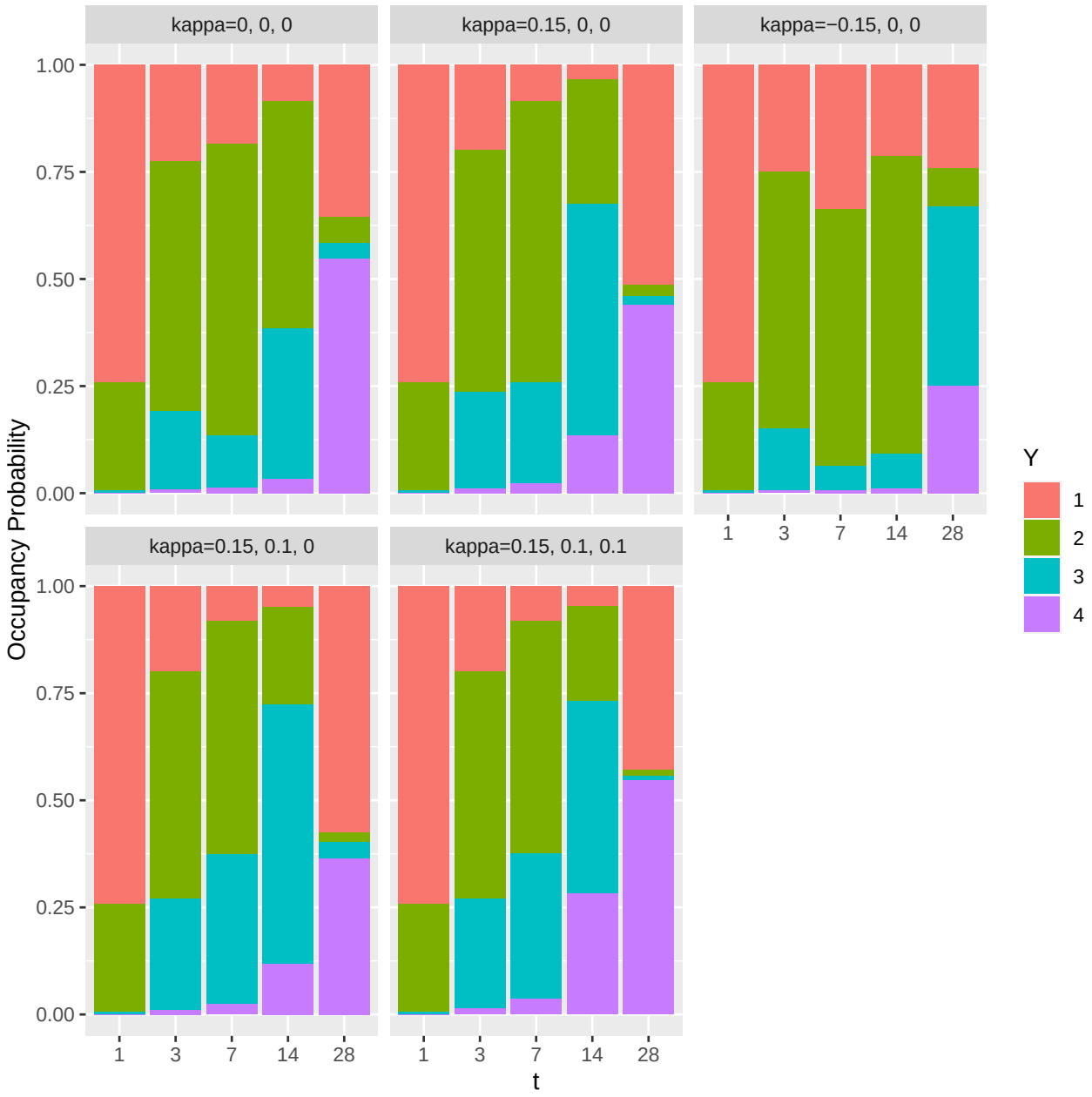

Figure 4: State occupancy probabilities varying κ, with $\tau = -4, 2$, $\gamma = 1, -1$. P(Y(14)=1 | Y(7)=1) = 0.05, 0.01, 0.27, 0.01, 0.01.

```
cap <- 'varying $\\\\tau$, with $\\\\gamma=1, -1$, $\\\\kappa=-0.1, 0, 0$'
taus <- list(c(-2, 2), c(-2, 0), c(0, 2), c(-4, 4))
vary1(tau=taus, cap=cap)
```

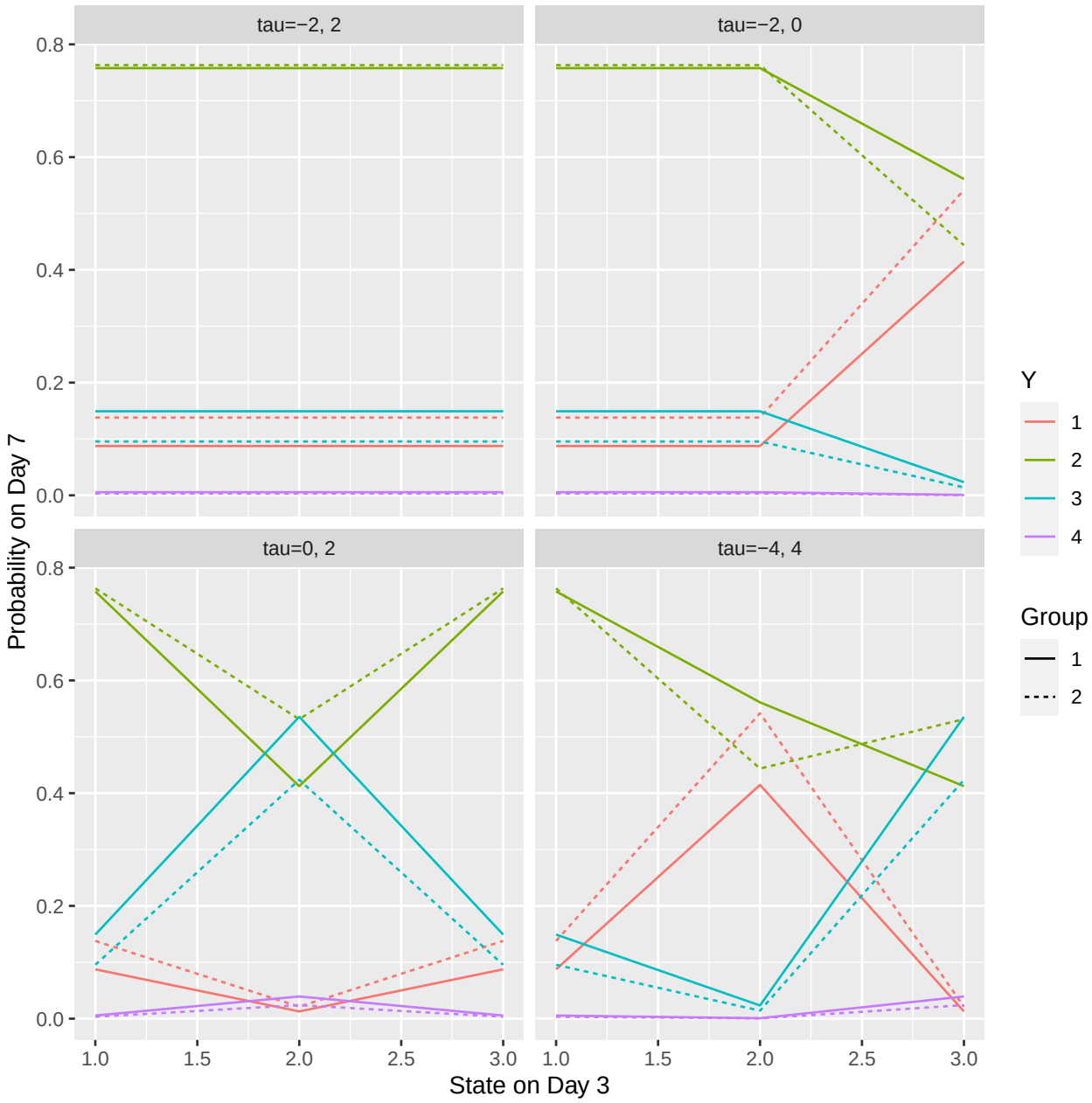

Figure 5: State transition probabilities varying τ, with $\gamma = 1, -1$, $\kappa = -0.1, 0, 0$. P(Y(14)=1 | Y(7)=1) = 0.16, 0.16, 0.16, 0.16.

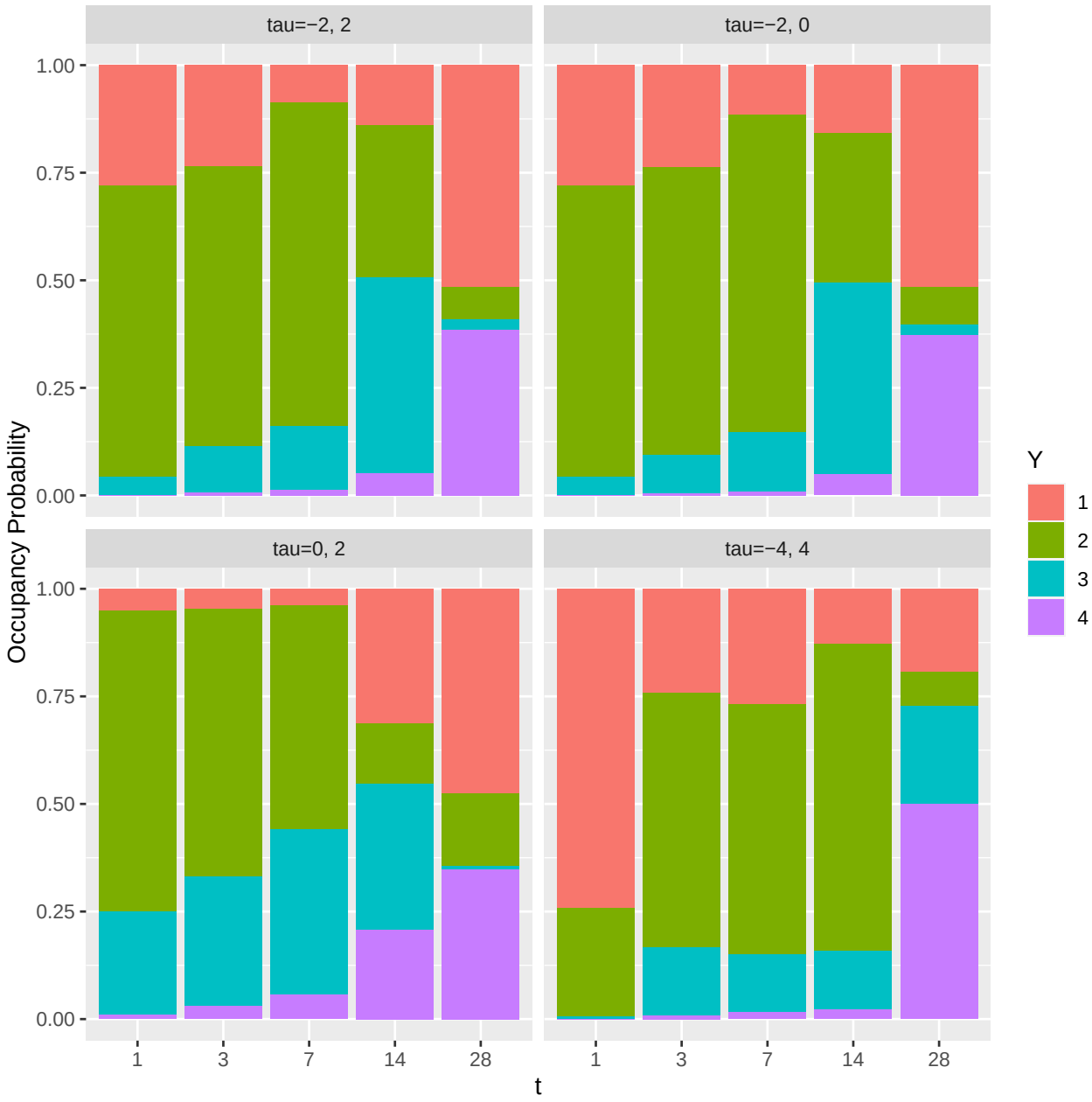

Figure 6: State occupancy probabilities varying τ, with $\gamma = 1, -1$, $\kappa = -0.1, 0, 0$. P(Y(14)=1 | Y(7)=1) = 0.16, 0.16, 0.16, 0.16.

Keep in mind that as defined in the `h` function above, the parameter values used for the parameters not being varied in a given graph must be taken into account in interpreting the graphs. The effects of one type of parameter are most easily seen in the transition probability graphs.

Specification of Other Parameters and Use of `Hmisc` Functions

The `Hmisc` functions we will use require a user-specified function `g` like the one above that computes the transition model linear predictor other than the proportional odds model intercept terms. `g` computes the linear predictor for one observation, or for a series of initial states (Y values at baseline = time 0). The arguments to `g` are `yprev` the previous value of Y (the initial state if `t=1`), the current time `t`, the gap between

the previous measurement time and `t`, covariate setting `X`, and `parameter` which specifies the true treatment effect. `g` returns a matrix with number of rows equal to the length of `yprev` and number of columns equal to one if the model is PO, or equal to the number of intercepts if it is partial PO. Other parameters such as the dependence structural parameters are passed as a vector `extra`. When non-proportional odds is modeled, as in the example above, `g` returns more than one column corresponding to all the intercept increments coming from the partial proportional odds model.

The transition model was defined as `g` above. Let's use the previous intercept values α as starting estimates in the context of solving for the entire set of parameters to meet our criteria. The `Hmisc intMarkovOrd` function uses the standard R optimization function `nlm` to compute the intercepts and other parameters satisfying given occupancy probability targets, once the user specifies initial guesses for all parameters. Note that the intercept values must be in decreasing order. `nlm` uses an efficient trial-and-error process to compute the parameter values that provide the best compromise solution. The criterion being minimized is the sum of absolute differences between all target probabilities and the actual achieved probabilities. With measurements made on days 1, 3, 7, 14, 28, our second set of target values are 28d state occupancy probabilities of 0.7, 0.14, 0.1, and 0.05 for, respectively, Y=1, 2, 3, 4. These probabilities pertain to patients who start with Y=2 at baseline in treatment group 1. Recall that he weighting function for time gaps is taken to be a negative exponential. The decay constant is estimated during the optimization. We apply a constraint during the optimization, namely that the decay constant is positive. Define a function `ftarget` to specify the $y = 1$ transition probability constraint.

```
# State occupancy probability targets
target <- rbind(  # row names define the times for which constraints apply
    '1'  = c(0.05, 0.70, 0.24, 0.01),
    '28' = c(0.70, 0.18, 0.07, 0.05) )

# Transition probability target
ftarget <- function(intercepts, extra) {
    # Constrain P(Y(14) = 1 | Y(7) = 1, X=1) = 0.9
    # Pr(Y = 1) = 1 - Pr(Y > 1) which uses first intercept
    p1 <- 1. - plogis(intercepts[1] + g(1, 14, 7, 1, extra=extra)[1])
    abs(p1 - 0.9)
}
# Try different starting values
z <- findstart(extra=c(tau=c(-3, 2), gamma=c(0,0), kappa=c(kappa1, 0, 0)),
               intercepts=alpha, times=times, ftarget=ftarget)
```

```
Minimum sum of absolute errors: 6.887887e-06


Extra achieving this minimum:

       tau1        tau2      gamma1      gamma2      kappa1      kappa2
-3.37907967  1.25157638  0.45651067 -0.99082651 -0.67992700 -0.01287161
     kappa3
 0.13336507

Iterations: 67
Sum of absolute errors: 6.887887e-06
Intercepts: 3.589 -0.454 -3.95

Extra parameters:

   tau1    tau2  gamma1  gamma2  kappa1  kappa2  kappa3
-0.6447  0.0064  0.8093 -1.0412 -0.4451  0.0787  0.1445
```

```
Log odds ratios at t=1 from occupancy probabilities: 0 0 0

Log odds ratios at t=28 from occupancy probabilities: -0.308 -0.383 -0.243
```

```
# Graph and optionally print (if pdf output)
sop12(y=1:4, times=times, initial=2, absorb=4,
      intercepts=z$intercepts, g=g, extra=z$extra)
```

```
Occupancy probabilities for group 1:

       1     2     3     4
1  0.050 0.700 0.240 0.010
3  0.098 0.728 0.158 0.017
7  0.243 0.618 0.116 0.023
14 0.477 0.412 0.081 0.030
28 0.700 0.180 0.070 0.050

Occupancy probabilities for group 2:

       1     2     3     4
1  0.050 0.700 0.240 0.010
3  0.101 0.729 0.153 0.017
7  0.256 0.616 0.106 0.022
14 0.511 0.396 0.065 0.028
28 0.760 0.154 0.045 0.040
```

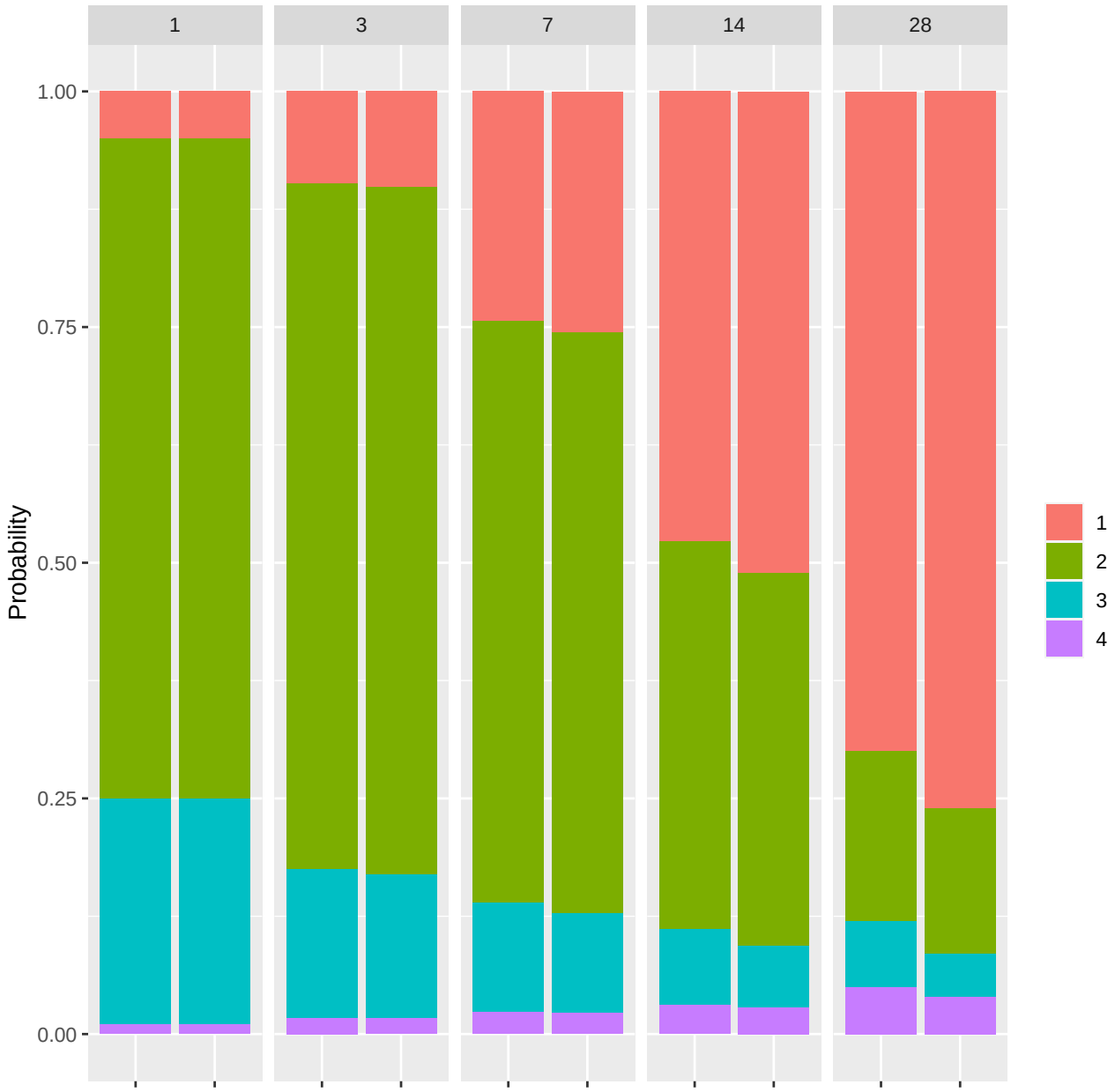

Figure 7: State occupancy probabilities with initial state 2 and absorbing state 4. Each pair of bars represents treatment 1 (left) and treatment 2 (right).

```
# Save the optimal values of the extra vector as default values for the extra argument
# so we don't need to specify them in later steps
formals(g)$extra <- z$extra

# Save intercepts
ints <- z$intercepts
# Save g object and intercepts
saveRDS(list(ints=ints, g=g), 'g.rds')
```

Let's check our $Y(7)$ to $Y(14)$ transition probability to see to what extent it is satisfied.

```
ftarget(ints, z$extra)
```

```
[1] 3.4757e-06
```

```
xb <- ints[1] + g(1, 14, 7, 1)[1]
1. - plogis(xb)
```

```
[1] 0.9000035
```

```
# Reproduce the result using soprobMarkovOrd directly
s <- soprobMarkovOrd(1:4, times, initial=2, absorb=4,
                     intercepts=ints, g=g, X=1)  # extra already stored in function
plotsop(s)
```

```
Occupancy probabilities for group 1:

       1     2     3     4
1  0.050 0.700 0.240 0.010
3  0.098 0.728 0.158 0.017
7  0.243 0.618 0.116 0.023
14 0.477 0.412 0.081 0.030
28 0.700 0.180 0.070 0.050
```

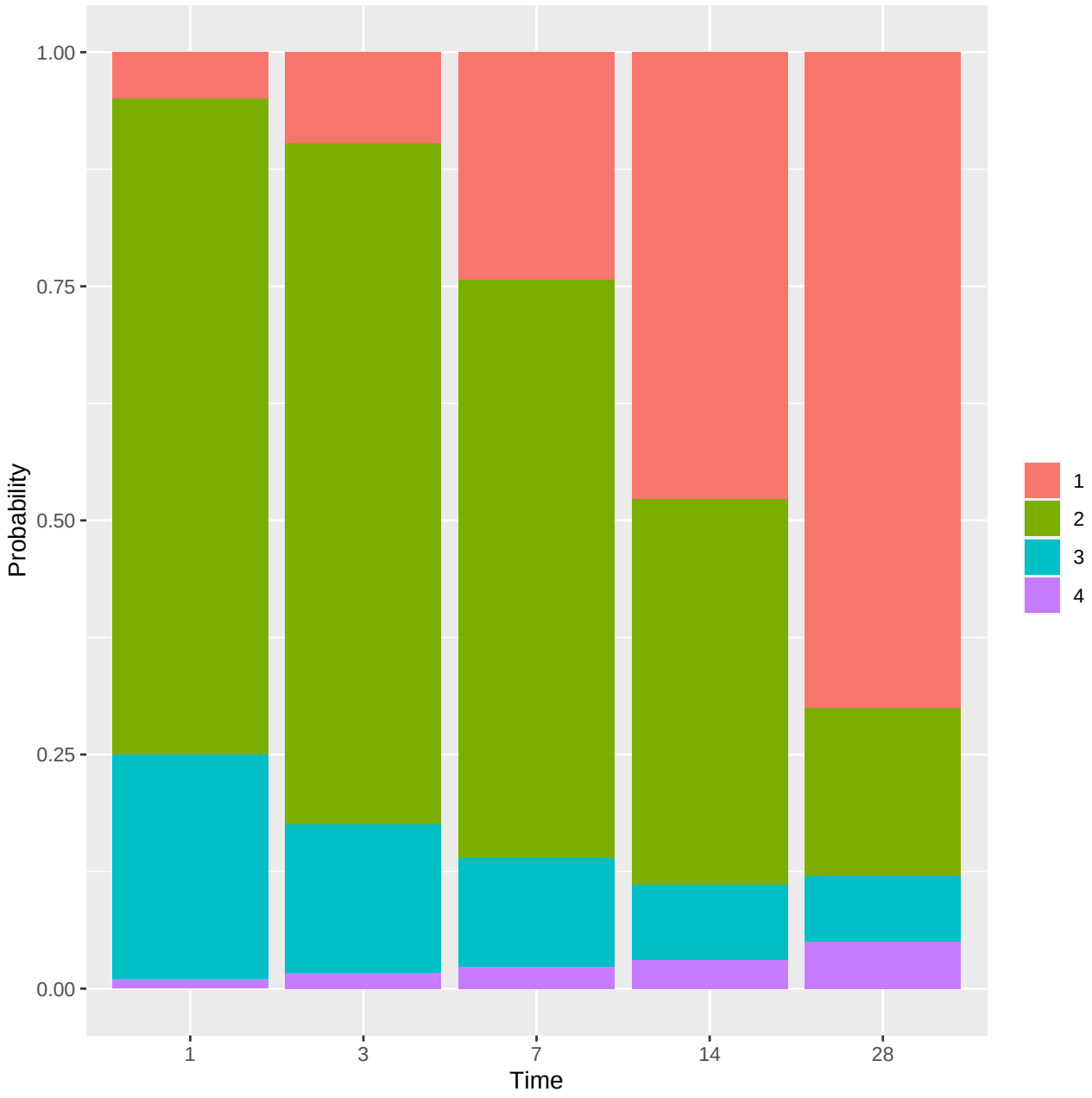

Figure 8: State occupancy probabilities with initial state 2 and absorbing state 4, for group 1

Since our model contains departures from proportional odds, check that the implied exceedance probabilities are in descending order as $y \uparrow$. Only time interacts with the intercept increments, so we only need to vary time.

```
chkints(times=times, intercepts=ints, extra=formals(g)$extra)
```

We also estimate simulation model parameters when there is no absorbing state, just for comparison.

```
zna <- intMarkovOrd(1:4, times, initial=2,
                    intercepts=c(4, 0, -3), g=g,
                    target=target, t=c(1, 28), ftarget=ftarget,
                    extra=c(-1, 3, .5, 2, -.5, 0, -.5))
```

```
Iterations: 71
Sum of absolute errors: 0.01833023
Intercepts: 4.57 0.566 -5.369

Extra parameters:

[1] -1.7075  2.4829  0.8743  1.6818 -0.5205 -0.2827 -0.1441

Log odds ratios at t=1 from occupancy probabilities: 0 0 0

Log odds ratios at t=28 from occupancy probabilities: -0.348 -0.227 -0.468
```

```
sop12(y=1:4, times=times, initial=2, intercepts=zna$intercepts,
      g=g, extra=z$extra)
```

```
Occupancy probabilities for group 1:

       1     2     3     4
1  0.019 0.500 0.478 0.002
3  0.035 0.581 0.382 0.002
7  0.244 0.543 0.212 0.001
14 0.456 0.372 0.171 0.002
28 0.689 0.149 0.157 0.004

Occupancy probabilities for group 2:

       1     2     3     4
1  0.019 0.500 0.478 0.002
3  0.036 0.588 0.373 0.002
7  0.254 0.546 0.199 0.001
14 0.474 0.380 0.145 0.001
28 0.713 0.166 0.118 0.003
```

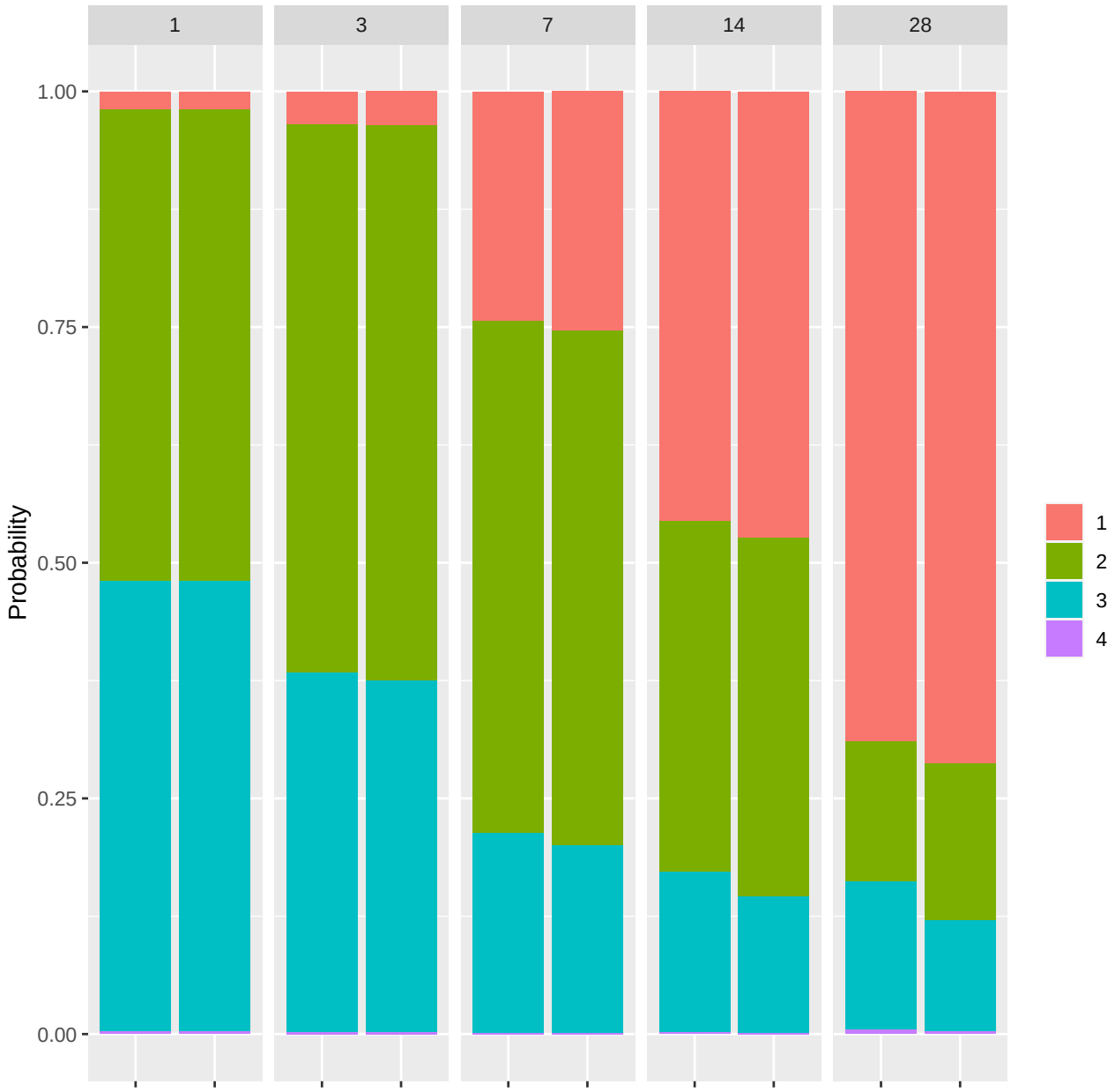

Figure 9: State occupancy probabilities for initial state 2 without an absorbing state

Make sure the sum of absolute errors is small, otherwise the optimization algorithm may have been fooled and you'll need to try different starting values for `intercepts` and/or `extra`.

Before running the clinical trial simulation, simulate a large sample size for one trial to check that the simulation is working correctly. Unlike what follows later, we will carry the absorbing state forward so that we can compute state occupancy proportions easily. A graphic shows the within-patient correlation matrix of ordinal outcomes across time.

```
r <- testsamp(intercepts=ints, times=times)
```

```
State occupancy proportions from 10000 samples

      1     2     3     4
```

```
1  0.048 0.708 0.233 0.011
3  0.099 0.725 0.158 0.019
7  0.246 0.614 0.114 0.026
14 0.480 0.407 0.078 0.034
28 0.699 0.175 0.072 0.053

Correlation matrix

   1     3     7     14    28
1   1.00  0.19  0.08  0.10  0.10
3         1.00 -0.03  0.08  0.12
7               1.00  0.35  0.31
14                    1.00  0.57
28                          1.00
```

```
w <- ggcorr(r)
ggp(w[[1]])
```

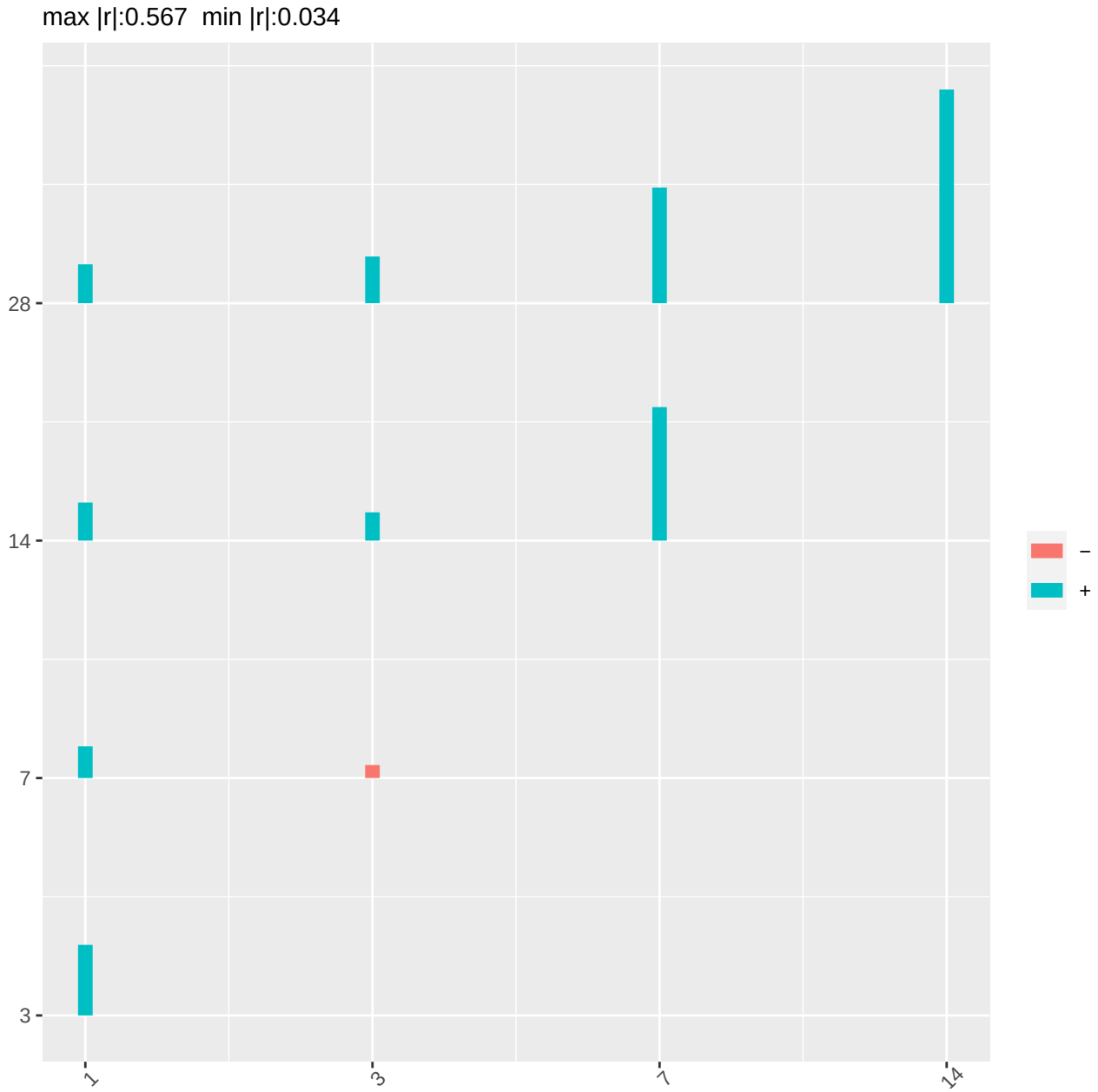

Figure 10: Correlations between numeric states at four measurement times within 10,000 simulated patients with initial state 2 and absorbing state 4

```
ggp(w[[2]])
```

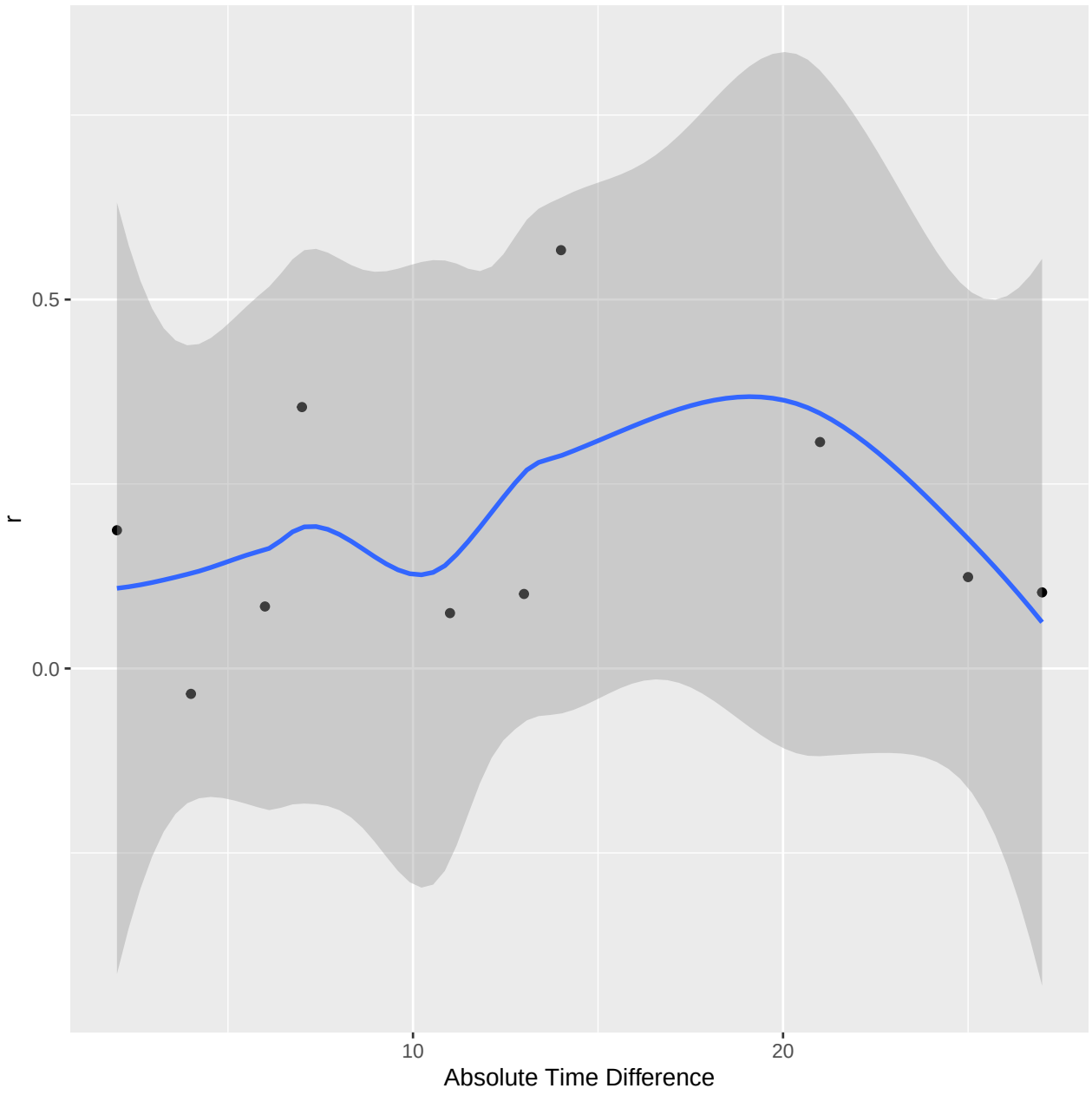

Figure 11: Correlation vs. time gap, computed from 10,000 simulated patients

The correlation pattern from the Markov simulated data is a typical serial correlation pattern with waning correlation as the time gap expands.

Let's also simulate a single 10000-patient trial to compare the usual proportional odds logistic model parameter variances with those from the cluster sandwich robust estimator.

```
n <- 10000
set.seed(8)
s1 <- simMarkovOrd(n=n / 2, y=1:4, times, initial=2, X=c(group=1),
                                    absorb=4, intercepts=ints, g=g)
s2 <- simMarkovOrd(n=n / 2, y=1:4, times, initial=2, X=c(group=2),
                                    absorb=4, intercepts=ints, g=g)
```

```
s2$id <- s2$id + n
s <- rbind(s1, s2)
s$yprev <- as.factor(s$yprev)
s$group <- as.factor(s$group)
dd <- datadist(s); options(datadist='dd')
f        <- lrm(y ~ yprev * pmax(gap - 2, 0) + time * group,
                                data=s, x=TRUE, y=TRUE)
frobust  <- robcov(f, s$id)
# Define needed group contrast at 28 times since time x group interaction present
g1 <- list(group=1, time=28); g2 <- list(group=2, time=28)
contrast(f,       g2, g1)
```

```
  yprev gap time   Contrast       S.E.      Lower     Upper     Z Pr(>|z|)
1     2   4   28 -0.6687118 0.05812905 -0.7826427 -0.554781 -11.5        0

Confidence intervals are 0.95 individual intervals
```

```
contrast(frobust, g2, g1)
```

```
  yprev gap time   Contrast       S.E.      Lower      Upper     Z Pr(>|z|)
1     2   4   28 -0.6687118 0.07370666 -0.8131742 -0.5242494 -9.07        0

Confidence intervals are 0.95 individual intervals
```

The estimated standard error of the 28d group effect is larger with the cluster sandwich estimate.

Simulation Models vs. Analysis Models

The statistical model used on real data is not privy to all the assumptions made by the simulation model. So typically the analysis model needs to have more parameters to account for what we don't know. Here are two examples:

- In our simulation model we don't have a main effect for treatment as we assume a zero treatment effect at the first follow-up time $t = 1$. The analysis model needs the treatment main effect to be added to such a treatment $\times$ time interaction effect.
- Our simulation model assumes that the impact of time gap is absent when the previous state is $y = 1$. But the fitted model allows for a main effect of $\max(\text{gap} - 2, 0)$, this main effect applying to the reference cell of previous $y = 1$.

Let's compare the parameter values used in the simulation with the estimated values from a partial proportional odds model fit, using the 10,000 patient simulated dataset derived just above. Take into account that the simulation model has a one-day offset for t.

```
g   # show simulation model
```

```
function (yprev, t, gap, X, parameter = -0.5, extra = c(tau1 = -0.644663132822171,
tau2 = 0.00638422564455977, gamma1 = 0.809250758250676, gamma2 = -1.04121247162486,
kappa1 = -0.445105768919569, kappa2 = 0.0786688148013411, kappa3 = 0.144460118545511
))
{
    tau <- extra[1:2]
    gamma <- extra[3:4]
    kappa <- extra[5:7]
    lp <- matrix(0, nrow = length(yprev), ncol = 3, dimnames = list(as.character(yprev),
        c("2", "3", "4")))
    gap <- pmax(gap - 2, 0)
    for (yp in yprev) lp[as.character(yp), ] <- tau[1] * (yp ==
```

```
        2) + tau[2] * (yp == 3) + gamma[1] * gap * (yp == 2) +
        gamma[2] * gap * (yp == 3) + (t - 1) * (kappa[1] + c(0,
        kappa[2], kappa[3])) + parameter * (X == 2) * (t - 1)/27
    lp
}
<bytecode: 0x55f081fb9120>
```

```
ints
```

```
[1]  3.5891118 -0.4539481 -3.9504574
```

```
s$tim <- s$time - 1
formula  <- y ~ yprev * pmax(gap - 2, 0) + tim * group
f <- VGAM::vgam(formula, VGAM::cumulative(parallel = FALSE ~ tim, reverse=TRUE), data=s)
k <- coef(f)
k
```

```
          (Intercept):1           (Intercept):2           (Intercept):3
             3.57524478              -0.46683800              -3.78003606
                 yprev2                   yprev3         pmax(gap - 2, 0)
            -0.65415374               0.05374520               0.03045999
                  tim:1                    tim:2                    tim:3
            -0.45156917              -0.37276133              -0.31539805
                 group2 yprev2:pmax(gap - 2, 0) yprev3:pmax(gap - 2, 0)
             0.00986727               0.79562209              -1.05111880
             tim:group2
            -0.01755417
```

```
# vgam parameterizes partial proportional odds terms as total effects and not as offsets
# from the main proportional odds term.  Rephrase to compare with our notation.
kn <- c('time:1', 'time:2', 'time:3')
kn <- c('tim:1', 'tim:2', 'tim:3')
kappa <- k[kn]
k[kn] <- c(kappa[1], kappa[2] - kappa[1], kappa[3] - kappa[1])
# From simulation model:
p <- formals(g)$extra
model <- c(ints, p[c('tau1', 'tau2')], 0,
                    p[c('kappa1','kappa2','kappa3')], 0,
                    p[c('gamma1','gamma2')], -0.5/27)
compp <- round(cbind(Estimated=k, True=model), 3)
compp
```

```
                       Estimated   True
(Intercept):1              3.575  3.589
(Intercept):2             -0.467 -0.454
(Intercept):3             -3.780 -3.950
yprev2                    -0.654 -0.645
yprev3                     0.054  0.006
pmax(gap - 2, 0)           0.030  0.000
tim:1                     -0.452 -0.445
tim:2                      0.079  0.079
tim:3                      0.136  0.144
group2                     0.010  0.000
yprev2:pmax(gap - 2, 0)    0.796  0.809
yprev3:pmax(gap - 2, 0)   -1.051 -1.041
tim:group2                -0.018 -0.019
```

Simulation

Using the above model we simulate, for each treatment effect, 1000 clinical trials each with 600 observations. For each trial the treatments are randomly assigned with probability 1/2 each. The high-level `estSeqMarkovOrd` function calls the `simMarkovOrd` function to simulate each trial. We use a distribution of initial states 1, 2, 3 and sample from that distribution to get each patient's baseline state. Because the contrast of interest needs to take into account a treatment × time interaction, the `groupContrast` argument must be specified below.

A second set of simulations is also run and stored for later, just for OR = 0.6 and using a PO model that has two unnecessary parameters: a square term for the time effect and a square term for the time × treatment interaction. This second simulation will be used to see how much power is lost by allowing for more complexity involving treatment.

A linear gap decay function is used. It would have perhaps been more reasonable to use an exponential decay such as $\exp(-\lambda\delta)$ but we cannot fit models that are nonlinear in the unknown parameters using standard regression software.

Note: `dosim` calls `estSeqMarkovOrd` which fits data using the `rms` package `lrm` function when the model is a proportional odds model. Since there is non-proportional odds, the `VGAM` package's `vgam` function is used to fit the partial proportional odds model. When doing Bayesian analysis, the `rmsb` package `blrm` function will fit a constrained partial PO model. When a PO model was fitted but non-PO was present, one simulation with OR=1.0 (not shown here) resulted in $\alpha > 0.1$. On an 8-core machine we use 7 cores in parallel to speed up simulations by almost a factor of 7.

Partial Proportional Odds Model Simulation

```
initial   <- c('1'=0.02, '2'=0.75, '3'=0.23)   # probabilities of being in baseline states
formula   <- y ~ yprev * pmax(gap - 2, 0) + time * group
ppo       <- ~ time    # partial PO model with PO exception for time
ors       <- c(0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.25)
contrvgam <- function(fit) {   # for customized vgam contrasts if non-PO in effect
    beta <- coef(fit)
    v    <- vcov(fit)
    # Treatment effect at 28 days (linear time x treatment interaction)
    me <- 'group2'; ia <- 'time:group2'
  # names of main effect and interaction parameters
    effect <- beta[me] + 28 * beta[ia]
    if(ia %nin% rownames(v)) saveRDS(fit, 'problemfit.rds')
    vr     <- v[me, me] + 28. * 28. * v[ia, ia] + 2. * 28. * v[me, ia]
    list(Contrast=effect, SE=sqrt(vr))
}
sims  <- dosim(times=times, g=g, initial=initial,
               intercepts=ints,
               ors=ors, formula=formula,
               ppo=ppo,  groupContrast=contrvgam,
               timecriterion=function(y) y == 1, seed=4,
                           file='simppo.rds')
```

```
Median of 1000 logistic regression coefficients for each OR

     (Intercept):1 (Intercept):2 (Intercept):3  yprev2  yprev3 pmax(gap - 2, 0)
0.4         4.0486       -0.0868       -3.6493 -0.6392 -0.0005          -0.0045
0.5         4.0401       -0.0987       -3.6682 -0.6419  0.0020          -0.0059
0.6         4.0630       -0.0739       -3.6547 -0.6634 -0.0101          -0.0151
0.7         4.0336       -0.1071       -3.6944 -0.6409  0.0012          -0.0035
```

```
0.8         4.0524       -0.0734       -3.6631 -0.6535  0.0088           0.0036
0.9         4.0365       -0.0833       -3.6563 -0.6543  0.0015          -0.0210
1           4.0538       -0.0687       -3.6454 -0.6609 -0.0163          -0.0079
1.25        4.0286       -0.0889       -3.6673 -0.6487 -0.0087          -0.0021
      time:1  time:2  time:3  group2 yprev2:pmax(gap - 2, 0)
0.4  -0.4446 -0.3658 -0.3019  0.0286                  0.8090
0.5  -0.4457 -0.3655 -0.3002  0.0236                  0.8157
0.6  -0.4443 -0.3654 -0.3013  0.0241                  0.8204
0.7  -0.4456 -0.3663 -0.3013  0.0083                  0.8117
0.8  -0.4470 -0.3685 -0.3038  0.0129                  0.8094
0.9  -0.4425 -0.3647 -0.2991  0.0112                  0.8194
1    -0.4466 -0.3685 -0.3020 -0.0014                  0.8177
1.25 -0.4436 -0.3669 -0.2978 -0.0112                  0.8138
     yprev3:pmax(gap - 2, 0) time:group2
0.4                  -1.0392     -0.0339
0.5                  -1.0412     -0.0264
0.6                  -1.0462     -0.0192
0.7                  -1.0403     -0.0131
0.8                  -1.0469     -0.0079
0.9                  -1.0376     -0.0042
1                    -1.0302      0.0001
1.25                 -1.0355      0.0082

Standard deviation of simulated group effects and square root of median of estimated variances

     OR    SD SDest
1: 0.40 0.227 0.219
2: 0.50 0.211 0.215
3: 0.60 0.209 0.212
4: 0.70 0.217 0.211
5: 0.80 0.218 0.209
6: 0.90 0.211 0.208
7: 1.00 0.206 0.208
8: 1.25 0.208 0.208
```

```
# Don't run this until figure out how to write a contrugam function for it
# formula2 <- y ~ yprev * pmax(gap - 2, 0) + pol(time, 2) * group
# est2 <- dosim(file='sim2.rds', g=g, initial=initial, intercepts=ints,
#                           ors=ors, formula=formula2, ppo=ppo,
#                         groupContrast=contrugam, timecriterion=function(y) y == 1,
#                         seed=5)
```

Before doing the main analyses on transition model parameters, look at the computed event times and estimate the type I assertion probability α and power of the "time to Y=1" variable analyzed with a Cox model, and look at the the magnitude of non-proportional hazards. Also compute the average log hazard ratio to see how the reciprocal of its anti-log relates to the odds ratio (reciprocal because the Cox model is fit on time until a *good* outcome). Then compute the power of the Markov model treatment comparison, and compare the χ^2 statistic from the transition model to that from the Cox model. For plotting, we take the square root of the χ^2 statistics to get a more symmetric distribution over the simulations. There is one panel per odds ratio.

```
examSim(sims, desc='Partial PO model with absorbing state 4')
```

```
Comparison of OR and HR

     OR   HR
```

```
1: 0.40 0.81
2: 0.50 0.86
3: 0.60 0.89
4: 0.70 0.93
5: 0.80 0.96
6: 0.90 0.98
7: 1.00 1.00
8: 1.25 1.05
```

Power of Cox test for time until Y=1

```
     OR power
1: 0.40  0.63
2: 0.50  0.40
3: 0.60  0.22
4: 0.70  0.11
5: 0.80  0.06
6: 0.90  0.06
7: 1.00  0.05
8: 1.25  0.07
```

Power of Markov proportional odds model test

```
     OR power
1: 0.40 0.982
2: 0.50 0.903
3: 0.60 0.698
4: 0.70 0.401
5: 0.80 0.206
6: 0.90 0.086
7: 1.00 0.049
8: 1.25 0.183
```

Spearman rho correlation between Markov and Cox model chi-square: 0.51

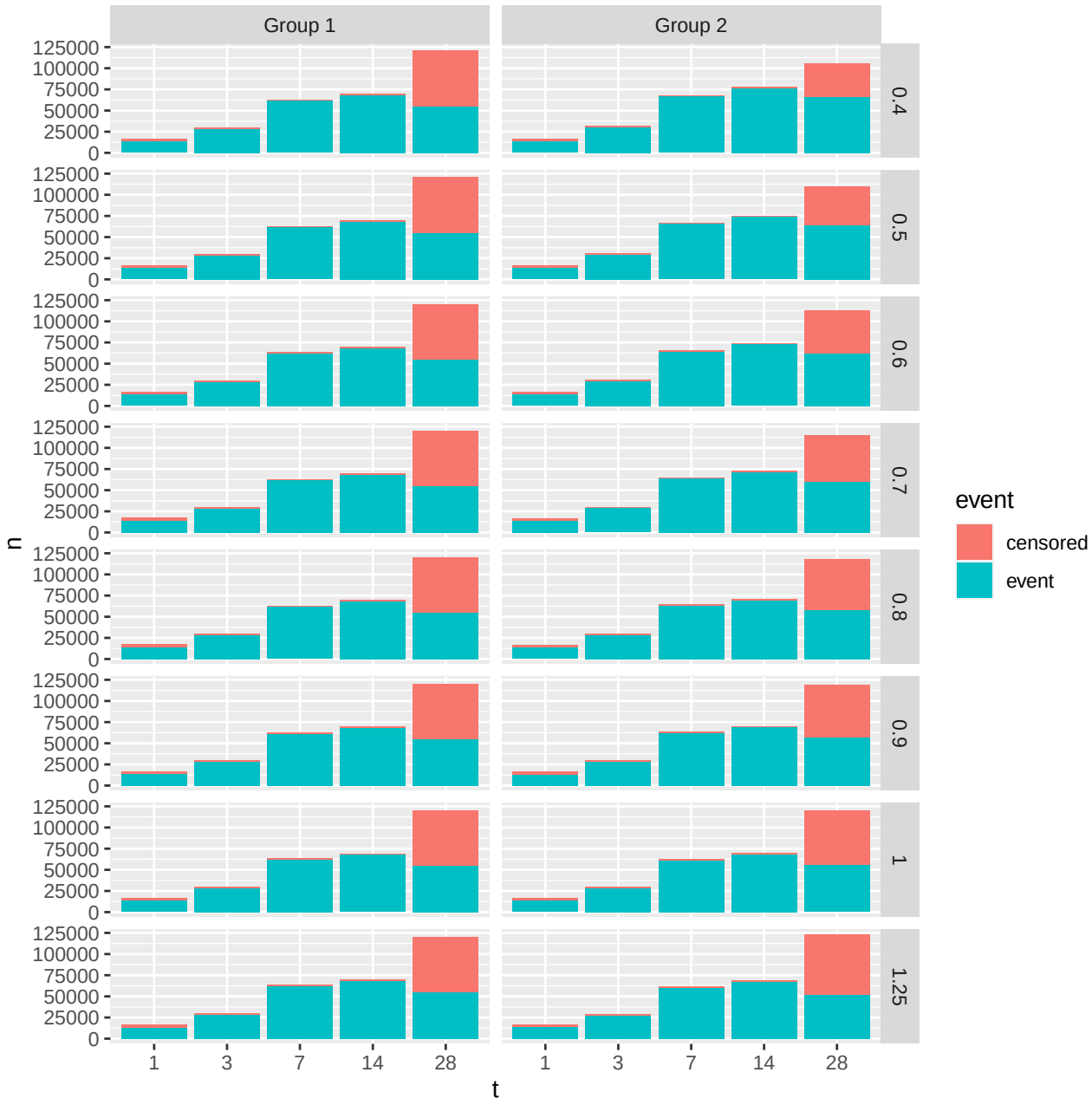

Figure 12: Time to Y=1 by group and OR, over simulations. Partial PO model with absorbing state 4.

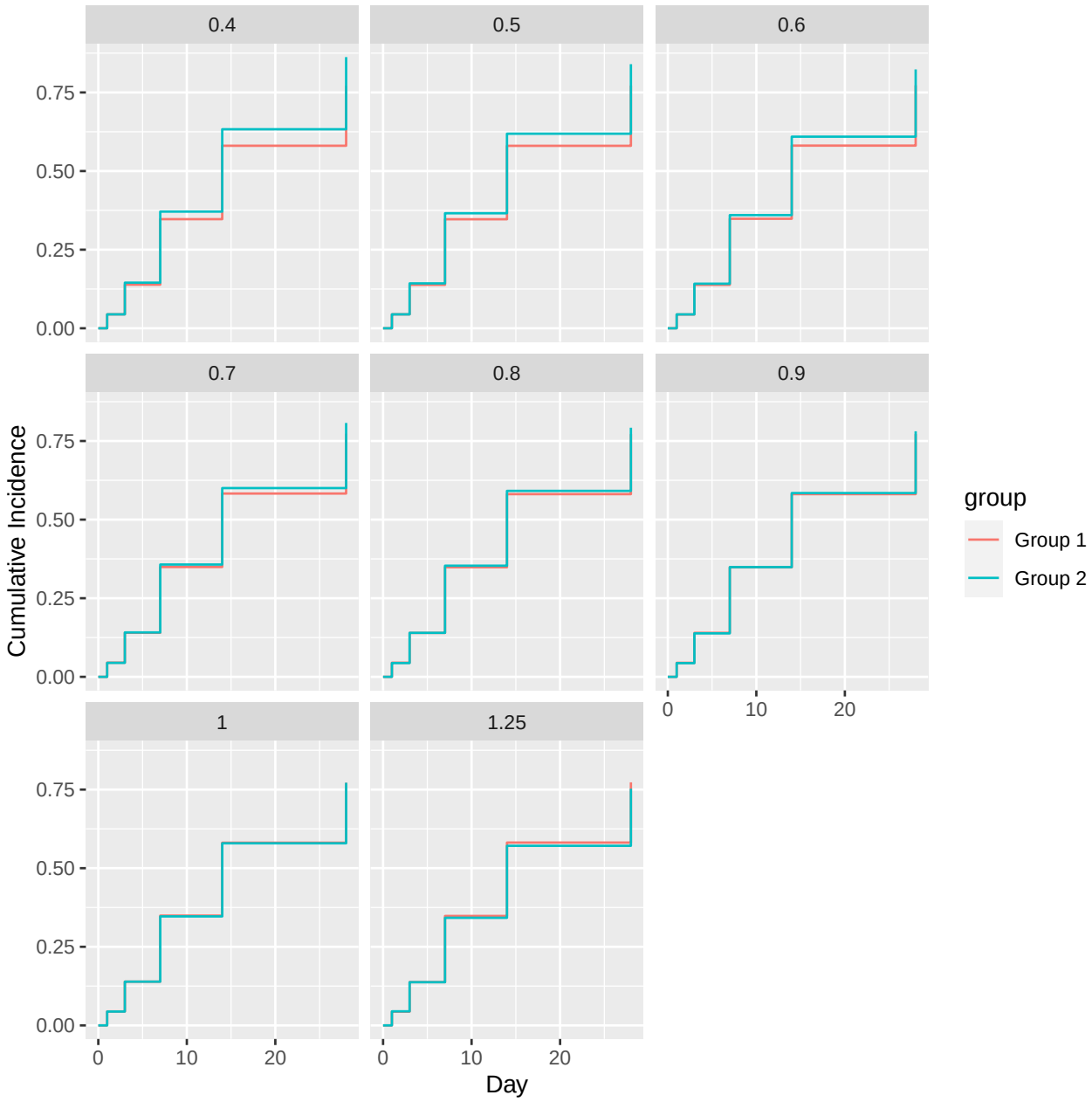

Figure 13: Cumulative incidence of Y=1. Partial PO model with absorbing state 4.

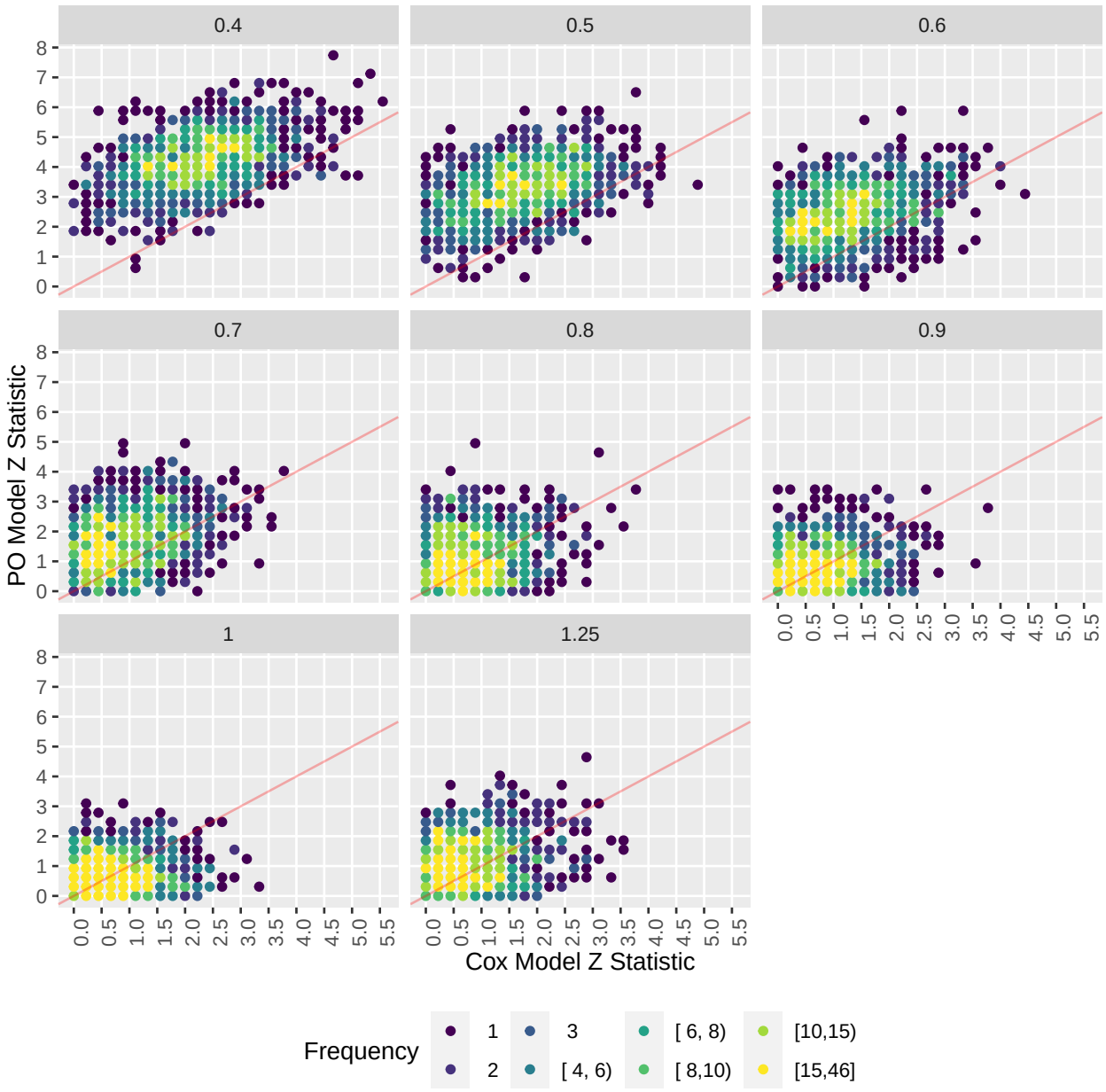

Figure 14: Scatter plot of Cox and PO model Z statistics. Partial PO model with absorbing state 4.

The hazard ratio, after taking the reciprocal so that a large HR means a worse outcome, is not equatable to the odds ratio except at the null value of 1.0. There is little evidence of non-proportional hazards.

The power of the longitudinal transition model is superior to that of the time to $Y = 1$ comparison. The χ^2 statistics from the Markov ordinal longitudinal model are seen to dominate those from the Cox model for time until $Y = 1$ as judged by the scatterplot.

Proportional Odds Model Simulation

The data were simulated using a realistic partial PO model to relax how the mix of events can change over time. What happens when we improperly fit a model that excludes departures from PO with respect to time? We repeat the above simulations to find out. We use the cluster sandwich robust covariance estimator to make up for some of the lack of fit.

```
# Define a contrast that works with rms::lrm so that we get treatment
# effect at day 28
contr <- list(list(group="1", time=28), list(group="2", time=28))
po  <- dosim(times=times, g=g, initial=initial,
             intercepts=ints,
             ors=ors, formula=formula,
             groupContrast=contr, cscov=TRUE,
             seed=4, file='simpo.rds')
```

```
Median of 1000 logistic regression coefficients for each OR

       y>=2    y>=3    y>=4 yprev=2 yprev=3     gap    time group=2
0.4  3.4204 -0.0572 -3.0696 -0.6435 -0.1109 -0.2547 -0.3004  0.0872
0.5  3.4033 -0.0580 -3.0460 -0.6412 -0.1148 -0.2600 -0.2995  0.0766
0.6  3.4127 -0.0312 -3.0058 -0.6546 -0.1254 -0.2616 -0.3019  0.0652
0.7  3.3854 -0.0530 -3.0213 -0.6444 -0.1099 -0.2464 -0.3043  0.0401
0.8  3.4011 -0.0176 -2.9686 -0.6533 -0.1115 -0.2439 -0.3090  0.0312
0.9  3.4087 -0.0126 -2.9456 -0.6553 -0.1173 -0.2558 -0.3067  0.0210
1    3.4195  0.0145 -2.9025 -0.6695 -0.1363 -0.2399 -0.3137 -0.0021
1.25 3.4125  0.0229 -2.8602 -0.6629 -0.1315 -0.2306 -0.3186 -0.0344
     yprev=2 * gap yprev=3 * gap time * group=2
0.4         0.8247       -0.9160        -0.0469
0.5         0.8301       -0.9099        -0.0380
0.6         0.8349       -0.9118        -0.0285
0.7         0.8282       -0.9073        -0.0204
0.8         0.8273       -0.9061        -0.0122
0.9         0.8375       -0.8964        -0.0061
1           0.8385       -0.8891         0.0000
1.25        0.8405       -0.8900         0.0136

Standard deviation of simulated group effects and square root of median of estimated variances

     OR    SD SDest
1: 0.40 0.301 0.292
2: 0.50 0.290 0.296
3: 0.60 0.294 0.299
4: 0.70 0.314 0.304
5: 0.80 0.321 0.307
6: 0.90 0.315 0.311
7: 1.00 0.312 0.316
8: 1.25 0.322 0.323
```

```
examSim(po, FALSE, FALSE, FALSE, FALSE, desc='PO model with lack of fit')
```

```
Power of Markov proportional odds model test

     OR power
1: 0.40 0.987
2: 0.50 0.909
3: 0.60 0.707
4: 0.70 0.406
5: 0.80 0.198
6: 0.90 0.087
7: 1.00 0.053
8: 1.25 0.182
```

Without using a robust covariance estimate, type I α assertion probability was 0.14 (not shown). The robust covariance estimate fixed the α problem, and the group comparison assuming PO had the same power as without assuming PO. The problems with the after-the-fit variance corrections are (1) we can't be certain that the magnitude of the group effect is correct, and more obviously (2) the approach does not transport to our Bayesian model.

Bayesian Power

Assume that the distribution of the maximum likelihood estimates is approximately Gaussian, and use a Gaussian prior distribution for the parameters of interest. Then we can use the maximum likelihood estimates already simulated to get approximate Gaussian Bayesian posterior distributions, and quickly compute such things as Bayesian power, e.g., the probability that the posterior probability of a beneficial effect exceeds 0.95 at the end of the study.

Define the Bayesian assertions and priors to be used for them

- Assertion 1: log(OR) < 0 under prior with prior mean 0 and sigma: P(OR>2)=0.025
- Assertion 2: log(OR) < 0 under flat prior
- Assertion 3: log(OR) > 0 under flat prior (sigma=100)
- Assertion 4: log(OR) > 0 under optimistic prior with mean log(0.85), sigma=0.5

```
asserts <-
  list(list('Efficacy',                  '<', 0, cutprior=log(2), tailprob=0.025),
       list('Efficacy flat',             '<', 0, mu=0,         sigma=100),
       list('Inefficacy/harm flat',      '>', 0, mu=0,         sigma=100),
       list('Inefficacy/harm optimistic', '>', 0, mu=log(0.85), sigma=0.5))

s <- gbayesSeqSim(sims, asserts=asserts)

head(s)
```

```
   sim  parameter look        est       vest     loghr  lrchisq  OR        p1
1:   1 -0.9162907  600 -0.9070252 0.04727730 0.2148584 5.327293 0.4 0.9998100
2:   2 -0.9162907  600 -1.0472731 0.04972046 0.2130254 5.365581 0.4 0.9999645
3:   3 -0.9162907  600 -0.8037527 0.05080288 0.1662093 3.392699 0.4 0.9986815
4:   4 -0.9162907  600 -1.0554004 0.04949816 0.2429140 6.903868 0.4 0.9999703
5:   5 -0.9162907  600 -0.7772264 0.04892012 0.1842283 4.064508 0.4 0.9985556
6:   6 -0.9162907  600 -0.9842713 0.04592889 0.2483342 7.126337 0.4 0.9999571
        mean1       sd1        p2      mean2       sd2           p3      mean3
1: -0.6582160 0.1852255 0.9999849 -0.9070209 0.2174329 1.513021e-05 -0.9070209
2: -0.7493692 0.1886190 0.9999987 -1.0472679 0.2229803 1.322098e-06 -1.0472679
3: -0.5715802 0.1900735 0.9998187 -0.8037487 0.2253944 1.812592e-04 -0.8037487
4: -0.7561463 0.1883167 0.9999990 -1.0553952 0.2224813 1.048997e-06 -1.0553952
5: -0.5586973 0.1875246 0.9997793 -0.7772226 0.2211784 2.206998e-04 -0.7772226
6: -0.7199049 0.1832834 0.9999978 -0.9842668 0.2143098 2.187429e-06 -0.9842668
         sd3           p4      mean4       sd4
1: 0.2174329 3.825630e-05 -0.7886231 0.1993955
2: 0.2229803 4.892701e-06 -0.9005017 0.2036476
3: 0.2253944 3.565318e-04 -0.6954542 0.2054817
4: 0.2224813 3.980939e-06 -0.9078336 0.2032673
5: 0.2211784 4.112276e-04 -0.6766257 0.2022722
6: 0.2143098 6.826450e-06 -0.8567333 0.1969787
```

```
attr(s, 'asserts')
```

```
                      label  cutprior tailprob         mu      sigma assertion
```

```
1                   Efficacy 0.6931472    0.025  0.0000000   0.353653       < 0
2              Efficacy flat        NA       NA  0.0000000 100.000000       < 0
3       Inefficacy/harm flat        NA       NA  0.0000000 100.000000       > 0
4 Inefficacy/harm optimistic        NA       NA -0.1625189   0.500000       > 0
```

```
alabels <- attr(s, 'alabels')  # named vector to map p1 p2 p3 p4 to labels
```

First let's examine the effect of the priors by making two pairwise comparisons: differences in posterior probabilities of efficacy under skeptical vs. flat prior, and differences in posterior probabilities of inefficacy under flat and optimistic priors.

```
w <- data.table(s)
u <- w[, .(p12max=max(abs(p1 - p2)), p12mean=mean(abs(p1 - p2)),
           p34max=max(abs(p3 - p4)), p34mean=mean(abs(p3 - p4))), by=.(look)]
z  <- melt(u, measure.vars=c('p12max', 'p12mean', 'p34max', 'p34mean'),
           variable.name='which', value.name='diff')
k <- c(p12max='Efficacy max',   p12mean='Efficacy mean',
       p34max='Inefficacy max', p34mean='Inefficacy mean')
z[, w := k[which]]
z
```

```
   look   which       diff               w
1:  600  p12max 0.03999198    Efficacy max
2:  600 p12mean 0.01699236   Efficacy mean
3:  600  p34max 0.06436124  Inefficacy max
4:  600 p34mean 0.01713457 Inefficacy mean
```

Compute the Bayesian power—the probability of hitting assertion-specific targets at the planned study end.

```
# Reshape results into taller and thinner data table so can plot over multiple assertions

ps <- names(alabels)
m  <- melt(w,
           measure.vars=list(ps, paste0('mean', 1:4), paste0('sd', 1:4)),
           variable.name='assert', value.name=c('p', 'mean', 'sd'))
m[, assert := alabels[assert]]
head(m)
```

```
   sim  parameter look        est       vest     loghr  lrchisq  OR   assert
1:   1 -0.9162907  600 -0.9070252 0.04727730 0.2148584 5.327293 0.4 Efficacy
2:   2 -0.9162907  600 -1.0472731 0.04972046 0.2130254 5.365581 0.4 Efficacy
3:   3 -0.9162907  600 -0.8037527 0.05080288 0.1662093 3.392699 0.4 Efficacy
4:   4 -0.9162907  600 -1.0554004 0.04949816 0.2429140 6.903868 0.4 Efficacy
5:   5 -0.9162907  600 -0.7772264 0.04892012 0.1842283 4.064508 0.4 Efficacy
6:   6 -0.9162907  600 -0.9842713 0.04592889 0.2483342 7.126337 0.4 Efficacy
           p       mean        sd
1: 0.9998100 -0.6582160 0.1852255
2: 0.9999645 -0.7493692 0.1886190
3: 0.9986815 -0.5715802 0.1900735
4: 0.9999703 -0.7561463 0.1883167
5: 0.9985556 -0.5586973 0.1875246
6: 0.9999571 -0.7199049 0.1832834
```

```
# Define targets
ptarget <- c(Efficacy                   = 0.95,
             'Efficacy flat'            = 0.95,
             'Inefficacy/harm flat'     = 0.9,
```

```
             'Inefficacy/harm optimistic' = 0.9)

m[, ptarget := ptarget[assert]]   # spreads targets to all rows
# hit = 0/1 indicator if hitting target at the single fixed sample size
u <- m[, .(hit = mean(p > ptarget)), by=.(OR, assert)]
u
```

```
      OR                     assert   hit
 1: 0.40                   Efficacy 0.987
 2: 0.50                   Efficacy 0.911
 3: 0.60                   Efficacy 0.715
 4: 0.70                   Efficacy 0.416
 5: 0.80                   Efficacy 0.209
 6: 0.90                   Efficacy 0.085
 7: 1.00                   Efficacy 0.022
 8: 1.25                   Efficacy 0.000
 9: 0.40              Efficacy flat 0.995
10: 0.50              Efficacy flat 0.948
11: 0.60              Efficacy flat 0.798
12: 0.70              Efficacy flat 0.531
13: 0.80              Efficacy flat 0.297
14: 0.90              Efficacy flat 0.130
15: 1.00              Efficacy flat 0.048
16: 1.25              Efficacy flat 0.002
17: 0.40       Inefficacy/harm flat 0.000
18: 0.50       Inefficacy/harm flat 0.000
19: 0.60       Inefficacy/harm flat 0.000
20: 0.70       Inefficacy/harm flat 0.002
21: 0.80       Inefficacy/harm flat 0.010
22: 0.90       Inefficacy/harm flat 0.038
23: 1.00       Inefficacy/harm flat 0.096
24: 1.25       Inefficacy/harm flat 0.416
25: 0.40 Inefficacy/harm optimistic 0.000
26: 0.50 Inefficacy/harm optimistic 0.000
27: 0.60 Inefficacy/harm optimistic 0.000
28: 0.70 Inefficacy/harm optimistic 0.002
29: 0.80 Inefficacy/harm optimistic 0.007
30: 0.90 Inefficacy/harm optimistic 0.027
31: 1.00 Inefficacy/harm optimistic 0.070
32: 1.25 Inefficacy/harm optimistic 0.334
      OR                     assert   hit
```

```
u$txt <- with(u, paste0('OR:', OR, '<br>', assert, '<br>Hit probability:', hit))
ggp(ggplot(u, aes(x=OR, y=hit, color=assert, label=txt)) +
  geom_line() +
  xlab('OR') + ylab('Proportion Hitting Posterior Probability Target') +
  guides(color=guide_legend(title='Assertion')))
```

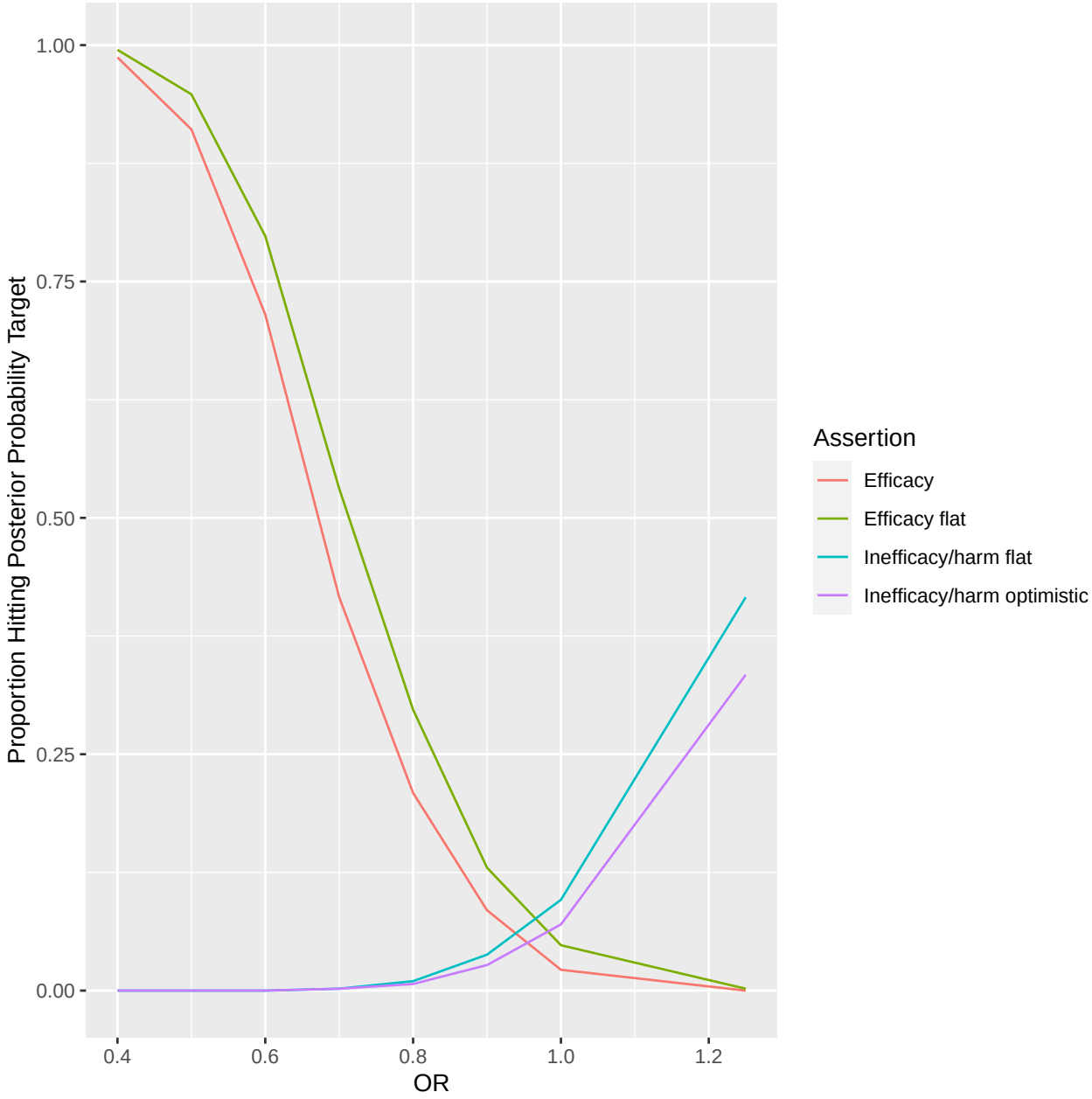

Figure 15: Probability of hitting posterior probability targets for various assertions, based on the posterior distribution of the odds ratio. Posterior probability targets could easily be constructed for state occupancy probabilities.

Frequentist Power of Single Day Ordinal Outcomes

Let's consider only the detection of OR=0.6 for an unadjusted proportional odds two-group comparison of ordinal outcomes measured at a single day. When simulating the data using the same model as above, we carry absorbing states forward here. So once Y=4 occurs on a given day, Y=4 is considered to be in effect at all later days.

```
seed <- 3
nsim <- 1000
```

```
gg   <- g
formals(gg)$parameter <- log(0.6)

prevhash <- NULL
file <- 'simch.rds'
if(file.exists(file)) {
    ch       <- readRDS(file)
    prevhash <- attr(ch, 'hash')
}

# See if previous simulation had identical inputs and Hmisc code
hashobj  <- list(deparse(simMarkovOrd), nsim, seed, times, ints, deparse(gg))
hash     <- digest::digest(hashobj)
if(! length(prevhash) || prevhash != hash) {
  # Run the simulations since something changed or file didn't exist
    set.seed(seed)
    ch <- matrix(NA, nrow=nsim, ncol=length(times))
    colnames(ch) <- as.character(times)
    for(isim in 1 : nsim) {
        s1 <- simMarkovOrd(n=300, 1:4, times, initial=2, X=c(group=1),
                                              absorb=4, intercepts=ints, g=gg, carry=TRUE)
        s2 <- simMarkovOrd(n=300, 1:4, times, initial=2, X=c(group=2),
                                              absorb=4, intercepts=ints, g=gg, carry=TRUE)
        s <- rbind(s1, s2)
        for(tim in times) {
            f <- lrm(y ~ group, data=s, subset=time == tim)
            ch[isim, as.character(tim)] <- f$stats['Model L.R.']
        }
    }
    attr(ch, 'hash') <- hash
    saveRDS(ch, file, compress='xz')
    }

pr('Frequentist power of individual day comparisons',
   apply(ch, 2, function(x) mean(x > 3.8415)) )
```

```
Frequentist power of individual day comparisons

    1     3     7    14    28
0.040 0.066 0.074 0.149 0.439
```

The power for testing differences on day 1 has to only be α because the true treatment effect is zero on that day. The power increases as time marches on. But even on day 28 the power is significantly below the power of the ordinal longitudinal model that uses all days.

Power With More Observations Per Patient

Let's repeat the main result but instead of sampling on selected days between 1 and 28, sample every day. For this setup, since the time gap is a constant 1.0 we no longer need `gap` in the model. The number of ORs simulated is reduced.

```
times <- 1 : 28
g <- function(yprev, t, gap, X, parameter=-0.5, extra) {
    tau   <- extra[1:2]
```

```
    kappa <- extra[3:5]
    lp <- matrix(0., nrow=length(yprev), ncol=3,
                 dimnames=list(as.character(yprev), c('2','3','4')))
    # 3 columns = no. distinct y less 1 = length of intercepts
    # lp[yp, ] is a 3-vector because the kappa components are split out into a 3-vector
    for(yp in yprev)
      lp[as.character(yp), ] <- tau[1] * (yp == 1) + tau[2] * (yp == 3) +
        (t - 1) * (kappa[1] + c(0., kappa[2], kappa[3])) +
            parameter * (X == 2) * (t - 1) / 27
  lp
}
alpha <- qlogis(c(0.95, 0.25, 0.01))

# Try different starting values
z <- findstart(intercepts=alpha,
                              extra=c(tau=c(-2, 2), kappa=c(0, 0, 0)),
                              times=times, seed=1)
```

```
Minimum sum of absolute errors: 0.0003213536


Extra achieving this minimum:

      tau1       tau2     kappa1     kappa2     kappa3
-1.0476586  2.4635850 -0.2865462 -0.1370526 -0.7035769

Iterations: 32
Sum of absolute errors: 0.0003213536
Intercepts: 2.943 -1.098 -4.599

Extra parameters:

   tau1    tau2  kappa1  kappa2  kappa3
-1.0726  2.7165 -0.1307  0.0733 -0.6901

Log odds ratios at t=1 from occupancy probabilities: 0 0 0

Log odds ratios at t=28 from occupancy probabilities: -0.586 -0.463 -0.037
```

```
sop12(y=1:4, times=times, initial=2, absorb=4, intercepts=z$intercepts,
      g=g, extra=z$extra)
```

```
Occupancy probabilities for group 1:

       1     2     3     4
1  0.050 0.700 0.240 0.010
2  0.048 0.571 0.353 0.028
3  0.046 0.502 0.413 0.039
4  0.047 0.466 0.442 0.045
5  0.050 0.449 0.453 0.048
6  0.056 0.444 0.451 0.049
7  0.064 0.446 0.441 0.050
8  0.073 0.451 0.426 0.050
9  0.085 0.457 0.407 0.050
```

```
10 0.100 0.463 0.387 0.050
11 0.117 0.468 0.365 0.050
12 0.136 0.471 0.343 0.050
13 0.159 0.471 0.321 0.050
14 0.184 0.468 0.299 0.050
15 0.212 0.461 0.277 0.050
16 0.243 0.452 0.255 0.050
17 0.277 0.439 0.234 0.050
18 0.314 0.423 0.214 0.050
19 0.352 0.404 0.194 0.050
20 0.392 0.382 0.176 0.050
21 0.434 0.358 0.158 0.050
22 0.476 0.332 0.142 0.050
23 0.517 0.306 0.127 0.050
24 0.558 0.279 0.113 0.050
25 0.597 0.253 0.101 0.050
26 0.634 0.227 0.089 0.050
27 0.668 0.203 0.079 0.050
28 0.700 0.180 0.070 0.050

Occupancy probabilities for group 2:

       1     2     3     4
1  0.050 0.700 0.240 0.010
2  0.049 0.573 0.350 0.028
3  0.048 0.508 0.406 0.039
4  0.050 0.476 0.430 0.044
5  0.055 0.464 0.434 0.046
6  0.063 0.464 0.426 0.047
7  0.074 0.469 0.409 0.048
8  0.088 0.477 0.387 0.048
9  0.105 0.485 0.362 0.048
10 0.125 0.491 0.336 0.048
11 0.149 0.494 0.308 0.048
12 0.177 0.493 0.281 0.048
13 0.209 0.488 0.254 0.048
14 0.245 0.478 0.228 0.048
15 0.285 0.463 0.204 0.048
16 0.328 0.443 0.180 0.048
17 0.374 0.419 0.159 0.048
18 0.421 0.392 0.139 0.048
19 0.470 0.361 0.121 0.048
20 0.518 0.330 0.104 0.048
21 0.565 0.297 0.090 0.048
22 0.610 0.265 0.077 0.048
23 0.652 0.234 0.066 0.048
24 0.690 0.205 0.056 0.048
25 0.725 0.178 0.048 0.048
26 0.756 0.154 0.041 0.048
27 0.784 0.133 0.036 0.048
28 0.807 0.114 0.031 0.048
```

```
# Save the optimal values of the extra vector as default values for the extra argument
# so we don't need to specify them in later steps
```

```
formals(g)$extra <- z$extra
# Save intercepts
ints <- z$intercepts
```

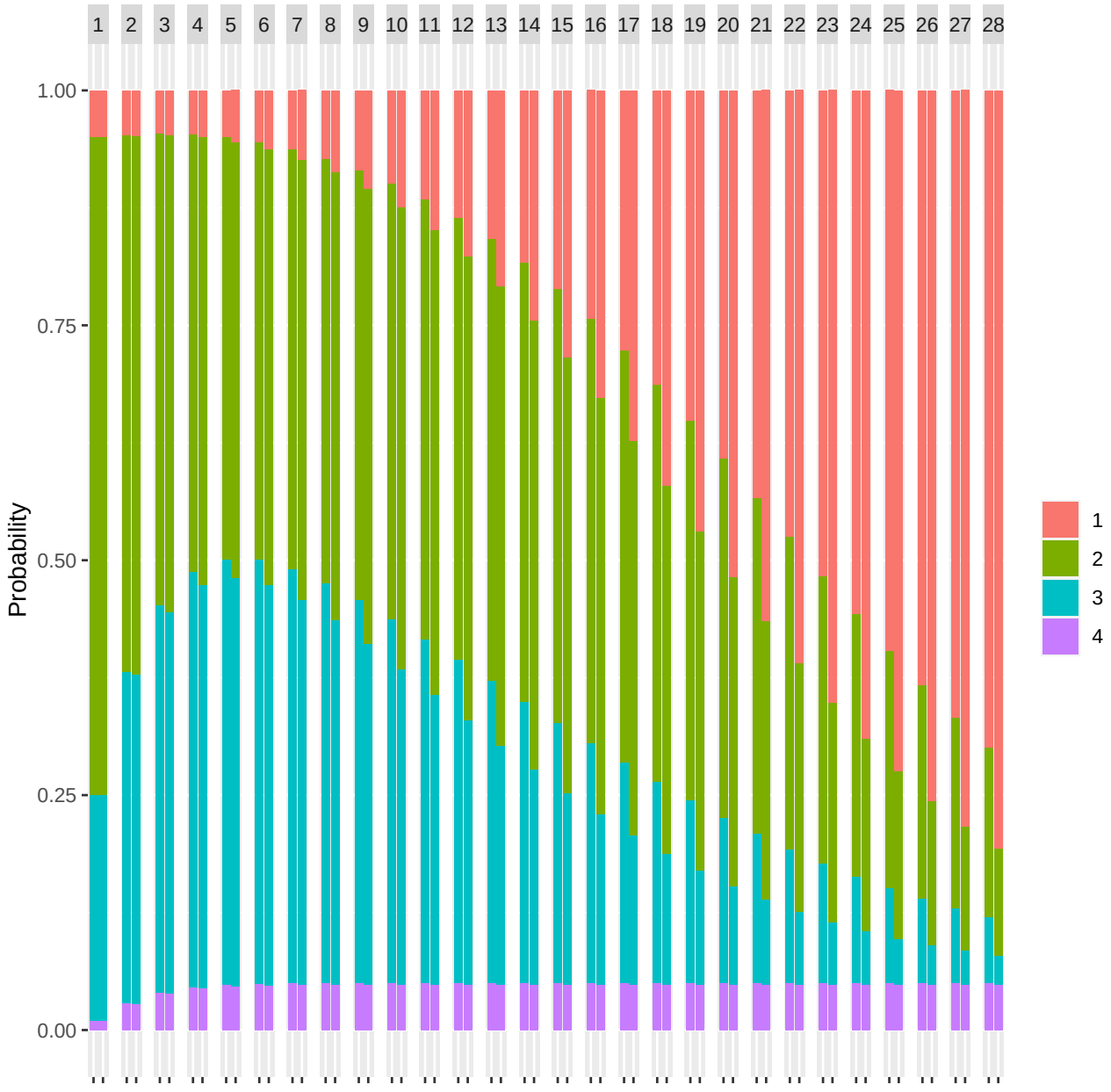

Figure 16: State occupancy probabilities for 28 consecutive days, with absorbing state 4. Pairs of bars indicate treatment 1 (left bar) and treatment 2 (right bar).

Reproduce the result using soprobMarkovOrd directly.

```
s <- soprobMarkovOrd(1:4, times, initial=2, absorb=4,
                     intercepts=ints, g=g, X=1)
plotsop(s)
```

```
Occupancy probabilities for group 1:
```

```
       1     2     3     4
1  0.050 0.700 0.240 0.010
2  0.048 0.571 0.353 0.028
3  0.046 0.502 0.413 0.039
4  0.047 0.466 0.442 0.045
5  0.050 0.449 0.453 0.048
6  0.056 0.444 0.451 0.049
7  0.064 0.446 0.441 0.050
8  0.073 0.451 0.426 0.050
9  0.085 0.457 0.407 0.050
10 0.100 0.463 0.387 0.050
11 0.117 0.468 0.365 0.050
12 0.136 0.471 0.343 0.050
13 0.159 0.471 0.321 0.050
14 0.184 0.468 0.299 0.050
15 0.212 0.461 0.277 0.050
16 0.243 0.452 0.255 0.050
17 0.277 0.439 0.234 0.050
18 0.314 0.423 0.214 0.050
19 0.352 0.404 0.194 0.050
20 0.392 0.382 0.176 0.050
21 0.434 0.358 0.158 0.050
22 0.476 0.332 0.142 0.050
23 0.517 0.306 0.127 0.050
24 0.558 0.279 0.113 0.050
25 0.597 0.253 0.101 0.050
26 0.634 0.227 0.089 0.050
27 0.668 0.203 0.079 0.050
28 0.700 0.180 0.070 0.050
```

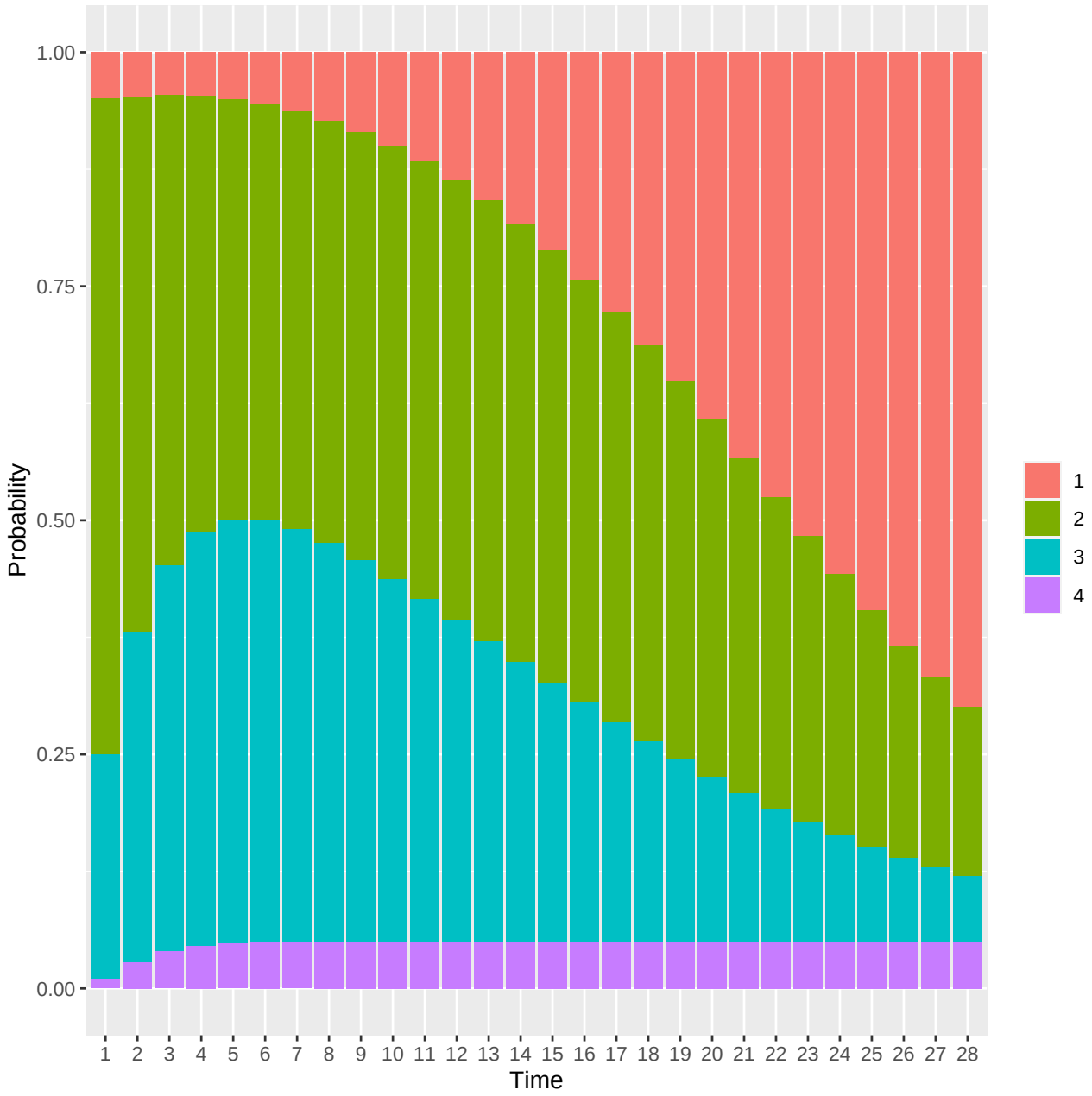

Figure 17: State occupancy probabilities with initial state 2 and absorbing state 4 for treatment group 1

```
r <- testsamp(n=10000, intercepts=ints, times=times)
```

```
State occupancy proportions from 10000 samples

      1     2     3     4
1 0.047 0.698 0.246 0.009
2 0.046 0.566 0.361 0.027
3 0.045 0.502 0.411 0.041
4 0.045 0.474 0.435 0.045
5 0.051 0.455 0.446 0.048
6 0.054 0.450 0.447 0.049
```

```
7  0.062 0.448 0.440 0.049
8  0.074 0.454 0.423 0.050
9  0.079 0.462 0.408 0.050
10 0.105 0.460 0.385 0.050
11 0.121 0.462 0.367 0.050
12 0.143 0.466 0.341 0.050
13 0.162 0.467 0.321 0.050
14 0.179 0.482 0.289 0.050
15 0.208 0.474 0.268 0.050
16 0.244 0.453 0.254 0.050
17 0.279 0.438 0.233 0.050
18 0.311 0.427 0.213 0.050
19 0.346 0.405 0.200 0.050
20 0.393 0.381 0.176 0.050
21 0.430 0.358 0.162 0.050
22 0.475 0.331 0.144 0.050
23 0.522 0.302 0.127 0.050
24 0.553 0.284 0.113 0.050
25 0.593 0.257 0.100 0.050
26 0.636 0.223 0.091 0.050
27 0.668 0.203 0.079 0.050
28 0.697 0.185 0.068 0.050
```

```
w <- ggcorr(r)
ggp(w[[1]])
```

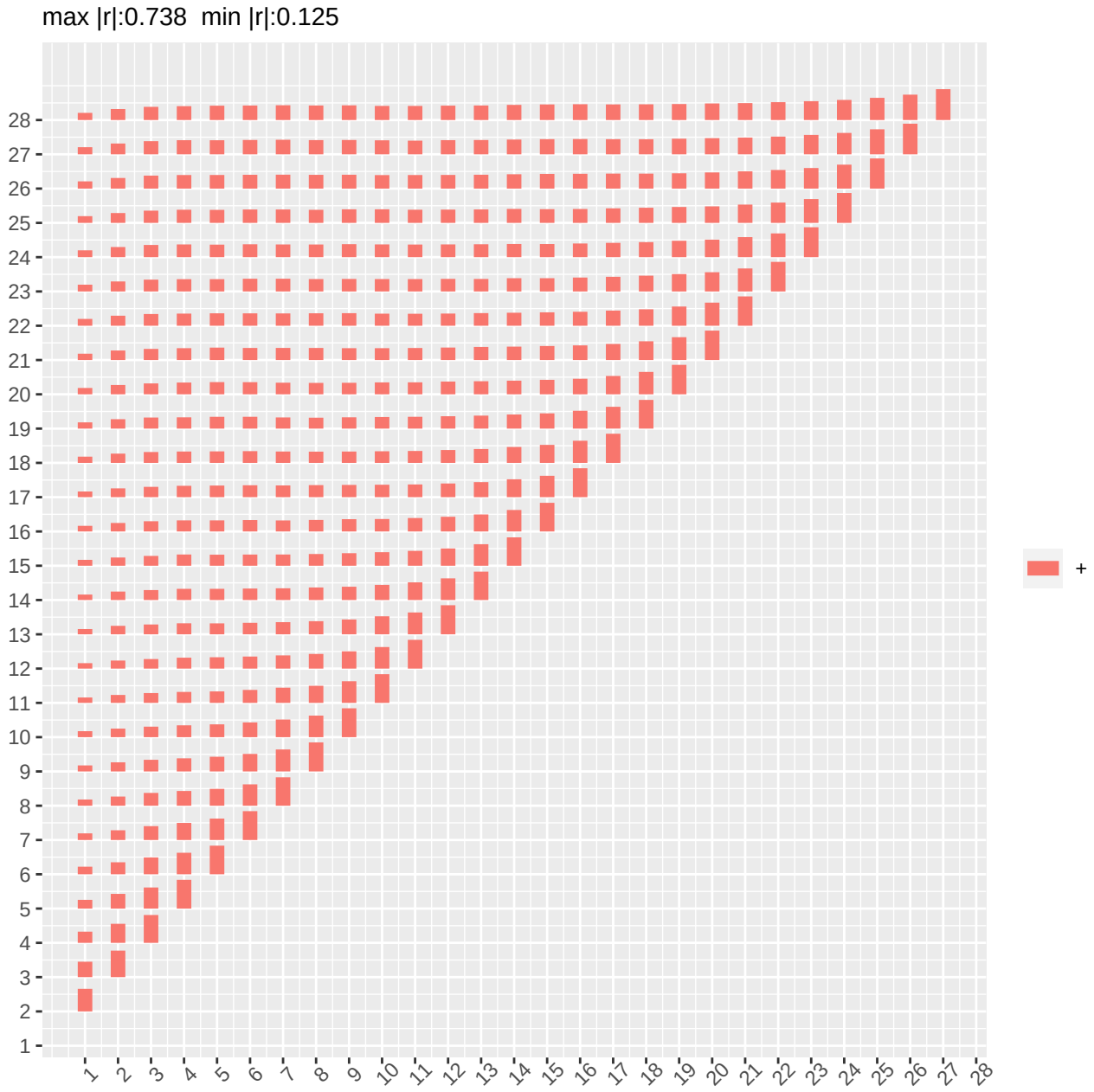

Figure 18: Correlations between times within 10,000 simulated patients with initial state 2 and absorbing state 4

```
ggp(w[[2]])
```

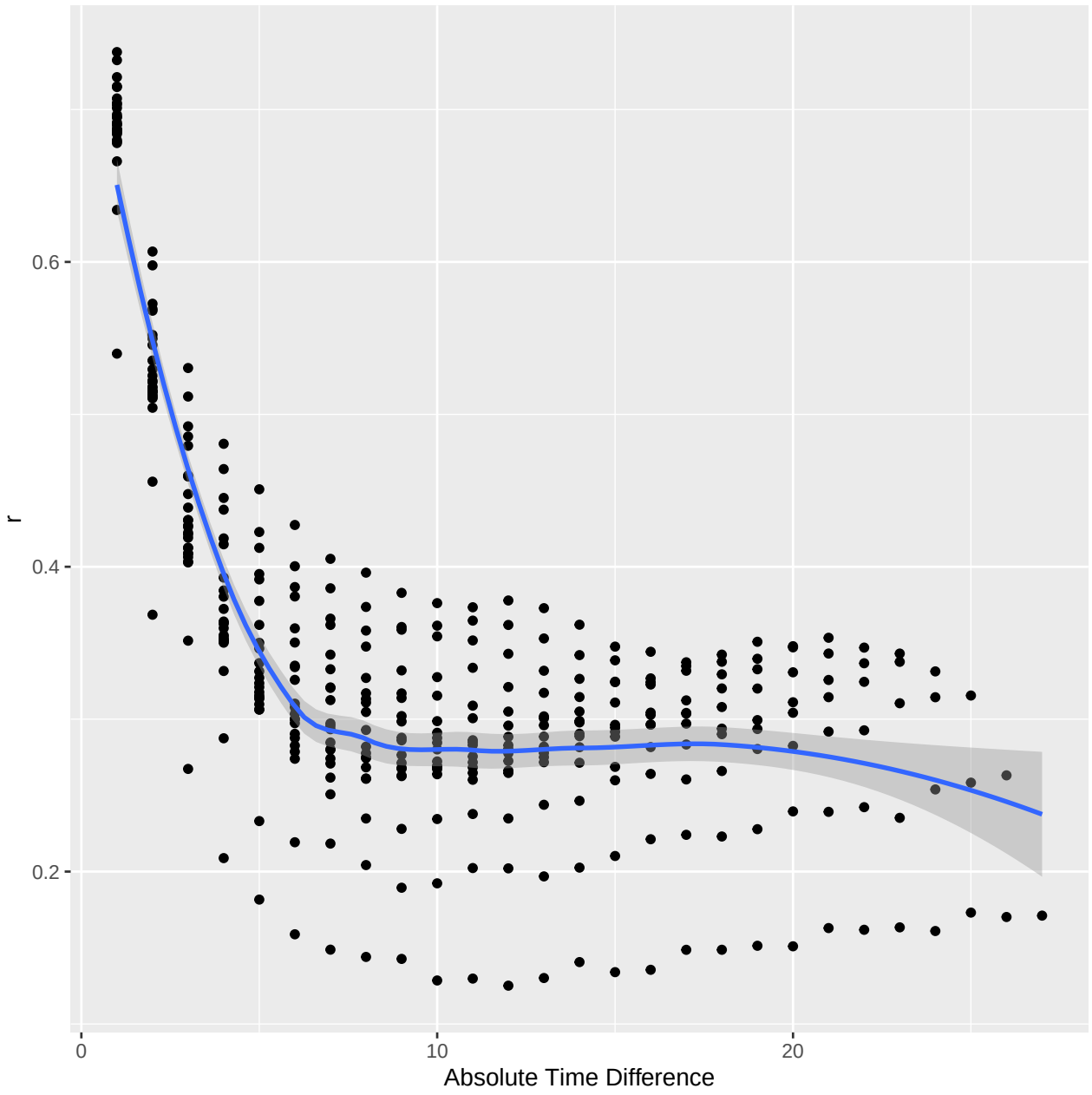

Figure 19: Correlations vs. time gap for 10,000 simulated patients

```
# Repeat without absorbing state
aextra <- c(tau=c(-4, 3), gamma=c(0,0), kappa=c(-0.2, 0.05, 0.15))   # initial guess
x <- findstart(intercepts=c(3, -1, -4), extra=aextra, ftarget=ftarget,
               absorb=NULL, times=times, seed=3)
```

```
Minimum sum of absolute errors: 1.681137e-05


Extra achieving this minimum:

      tau1        tau2      gamma1      gamma2      kappa1      kappa2
```

```
-3.08934089  3.67723230 -0.57315012 -0.01057289  0.07248851  0.89218279
     kappa3
-0.82651192

Iterations: 63
Sum of absolute errors: 1.681137e-05
Intercepts: 2.945 -1.099 -4.595

Extra parameters:

   tau1    tau2  gamma1  gamma2  kappa1  kappa2  kappa3
-4.3222  3.6988 -0.0630  0.0502  0.1031  0.8922 -0.8265

Log odds ratios at t=1 from occupancy probabilities: 0 0 0

Log odds ratios at t=28 from occupancy probabilities: -0.829 -0.886 -0.972
```

```
sop12(y=1:4, times=times, initial=2, intercepts=x$intercepts, g=g,
      extra=x$extra)
```

```
Occupancy probabilities for group 1:

       1     2     3     4
1  0.050 0.700 0.240 0.010
2  0.078 0.522 0.320 0.079
3  0.099 0.456 0.341 0.105
4  0.116 0.432 0.339 0.114
5  0.132 0.422 0.330 0.116
6  0.148 0.417 0.318 0.116
7  0.165 0.414 0.305 0.115
8  0.183 0.411 0.293 0.114
9  0.201 0.406 0.280 0.113
10 0.221 0.401 0.267 0.111
11 0.241 0.395 0.254 0.109
12 0.263 0.388 0.242 0.107
13 0.286 0.381 0.229 0.104
14 0.310 0.372 0.217 0.102
15 0.334 0.362 0.205 0.099
16 0.360 0.351 0.193 0.096
17 0.387 0.340 0.181 0.093
18 0.415 0.327 0.169 0.089
19 0.443 0.314 0.157 0.086
20 0.471 0.300 0.146 0.082
21 0.501 0.286 0.135 0.078
22 0.530 0.271 0.125 0.074
23 0.559 0.256 0.114 0.070
24 0.588 0.241 0.104 0.066
25 0.617 0.225 0.095 0.062
26 0.646 0.210 0.086 0.058
27 0.673 0.195 0.078 0.054
28 0.700 0.180 0.070 0.050

Occupancy probabilities for group 2:

       1     2     3     4
```

```
1  0.050 0.700 0.240 0.010
2  0.079 0.524 0.319 0.078
3  0.101 0.460 0.338 0.101
4  0.120 0.437 0.335 0.108
5  0.139 0.428 0.324 0.109
6  0.159 0.423 0.310 0.107
7  0.181 0.419 0.296 0.105
8  0.203 0.415 0.281 0.101
9  0.228 0.409 0.265 0.098
10 0.254 0.402 0.250 0.094
11 0.282 0.393 0.235 0.090
12 0.312 0.383 0.219 0.086
13 0.343 0.371 0.204 0.082
14 0.376 0.357 0.189 0.078
15 0.410 0.342 0.174 0.073
16 0.446 0.326 0.160 0.069
17 0.482 0.308 0.146 0.064
18 0.519 0.290 0.132 0.059
19 0.556 0.270 0.119 0.055
20 0.593 0.250 0.106 0.050
21 0.630 0.230 0.094 0.046
22 0.665 0.210 0.083 0.041
23 0.700 0.190 0.073 0.037
24 0.732 0.171 0.063 0.033
25 0.763 0.153 0.055 0.029
26 0.792 0.135 0.047 0.026
27 0.818 0.119 0.040 0.023
28 0.842 0.104 0.034 0.020
```

```
zna <- intMarkovOrd(1:4, times, initial=2,
                    intercepts=x$intercepts, g=g,
                    target=target, t=c(1, 28), ftarget=ftarget,
                    extra=x$extra)
```

```
Iterations: 1
Sum of absolute errors: 1.681137e-05
Intercepts: 2.945 -1.099 -4.595

Extra parameters:

   tau1    tau2  gamma1  gamma2  kappa1  kappa2  kappa3
-4.3222  3.6988 -0.0630  0.0502  0.1031  0.8922 -0.8265

Log odds ratios at t=1 from occupancy probabilities: 0 0 0

Log odds ratios at t=28 from occupancy probabilities: -0.829 -0.886 -0.972
```

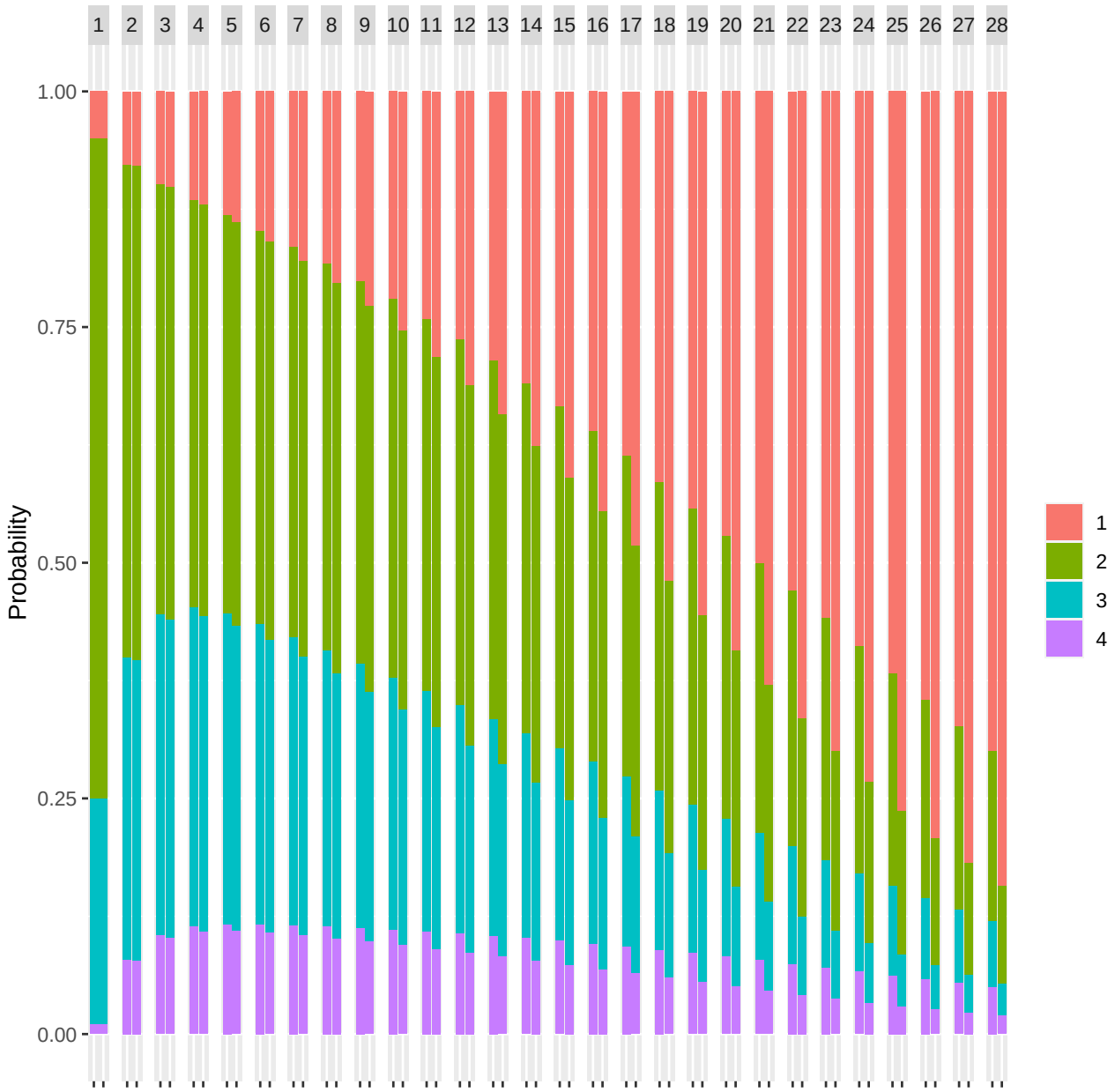

Figure 20: State occupancy probabilities with no absorbing state

Now simulate 28d of data per patient.

```
initial <- c('1'=0.02, '2'=0.75, '3'=0.23)   # probabilities of being in baseline states
ors     <- c(0.5, 0.6, 0.7, 0.8, 0.9, 1.0)
formula <- y ~ yprev + time * group

est <- dosim(times=times, g=g, initial=initial, intercepts=ints,
                         ors=ors, formula=formula, ppo=ppo,
                         groupContrast=contrvgam, timecriterion=function(y) y == 1,
                         seed=6, file='sim28.rds')
```

```
Median of 1000 logistic regression coefficients for each OR
```

```
    (Intercept):1 (Intercept):2 (Intercept):3 yprev2 yprev3  time:1  time:2
0.5        1.9955       -2.1056       -4.8253 1.0713 3.7876 -0.1307 -0.0578
0.6        2.0107       -2.1078       -4.8346 1.0672 3.7834 -0.1312 -0.0573
0.7        2.0017       -2.1118       -4.8479 1.0723 3.7917 -0.1307 -0.0576
0.8        2.0054       -2.1143       -4.8308 1.0713 3.7877 -0.1310 -0.0575
0.9        1.9998       -2.1143       -4.8610 1.0705 3.7867 -0.1306 -0.0575
1          2.0032       -2.1108       -4.8323 1.0697 3.7898 -0.1309 -0.0576
     time:3 group2 time:group2
0.5 -0.8259 0.0207     -0.0256
0.6 -0.8237 0.0206     -0.0189
0.7 -0.8167 0.0125     -0.0133
0.8 -0.8251 0.0095     -0.0081
0.9 -0.8206 0.0074     -0.0038
1   -0.8252 0.0000      0.0002

Standard deviation of simulated group effects and square root of median of estimated variances

    OR    SD SDest
1: 0.5 0.075 0.074
2: 0.6 0.074 0.072
3: 0.7 0.071 0.071
4: 0.8 0.069 0.070
5: 0.9 0.068 0.069
6: 1.0 0.071 0.068
```

As before we also check the performance of a Cox two-sample test for time to $Y = 1$ as well as that of the ordinal longitudinal model. We also show Kaplan-Meier cumulative incidence estimates for being discharged to home, censoring on death (hence competing risks are not really being dealt with optimally). The cumulative incidences are computed after pooling all 1000 studies' data.

```
examSim(est, desc='Partial PO model with 28 days of measurements')
```

```
Comparison of OR and HR

    OR   HR
1: 0.5 0.73
2: 0.6 0.79
3: 0.7 0.85
4: 0.8 0.90
5: 0.9 0.95
6: 1.0 1.00

Power of Cox test for time until Y=1

    OR power
1: 0.5  0.95
2: 0.6  0.77
3: 0.7  0.50
4: 0.8  0.22
5: 0.9  0.08
6: 1.0  0.05

Power of Markov proportional odds model test

    OR power
1: 0.5 1.000
```

```
2: 0.6 1.000
3: 0.7 0.999
4: 0.8 0.888
5: 0.9 0.326
6: 1.0 0.055
```

```
Spearman rho correlation between Markov and Cox model chi-square: 0.71
```

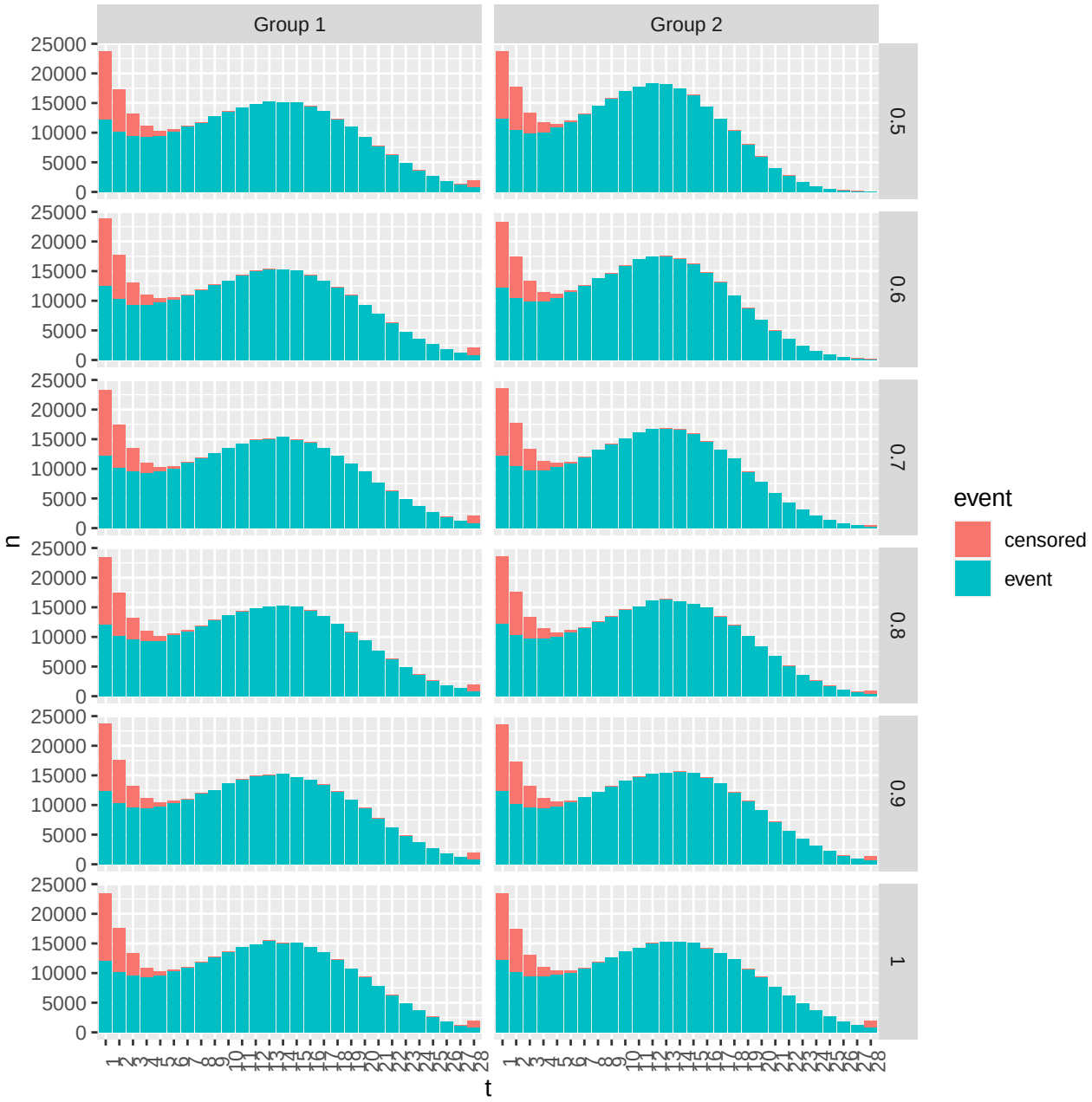

Figure 21: Time to Y=1 by group and OR, over simulations. Partial PO model with 28 days of measurements.

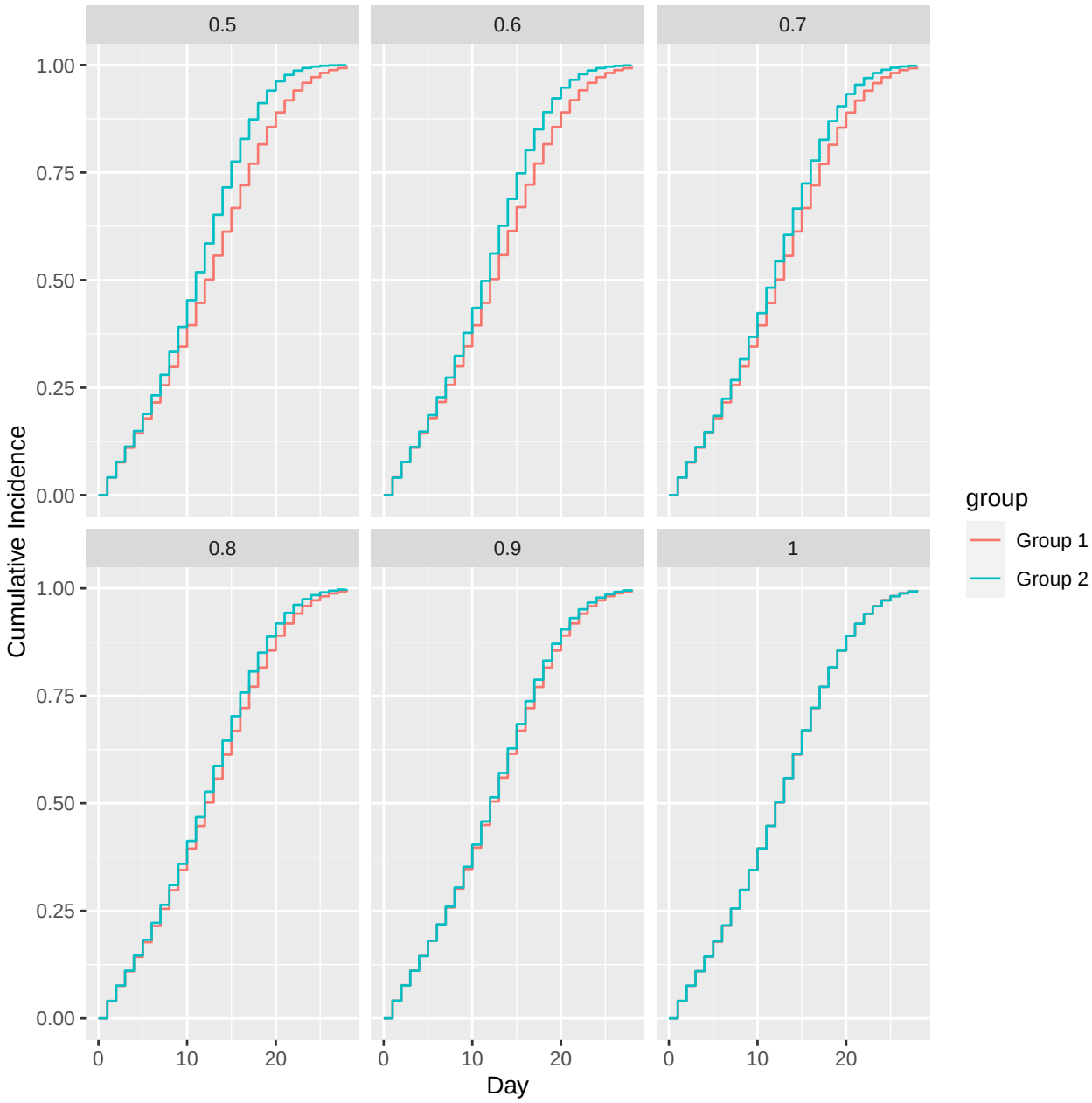

Figure 22: Cumulative incidence of Y=1. Partial PO model with 28 days of measurements.

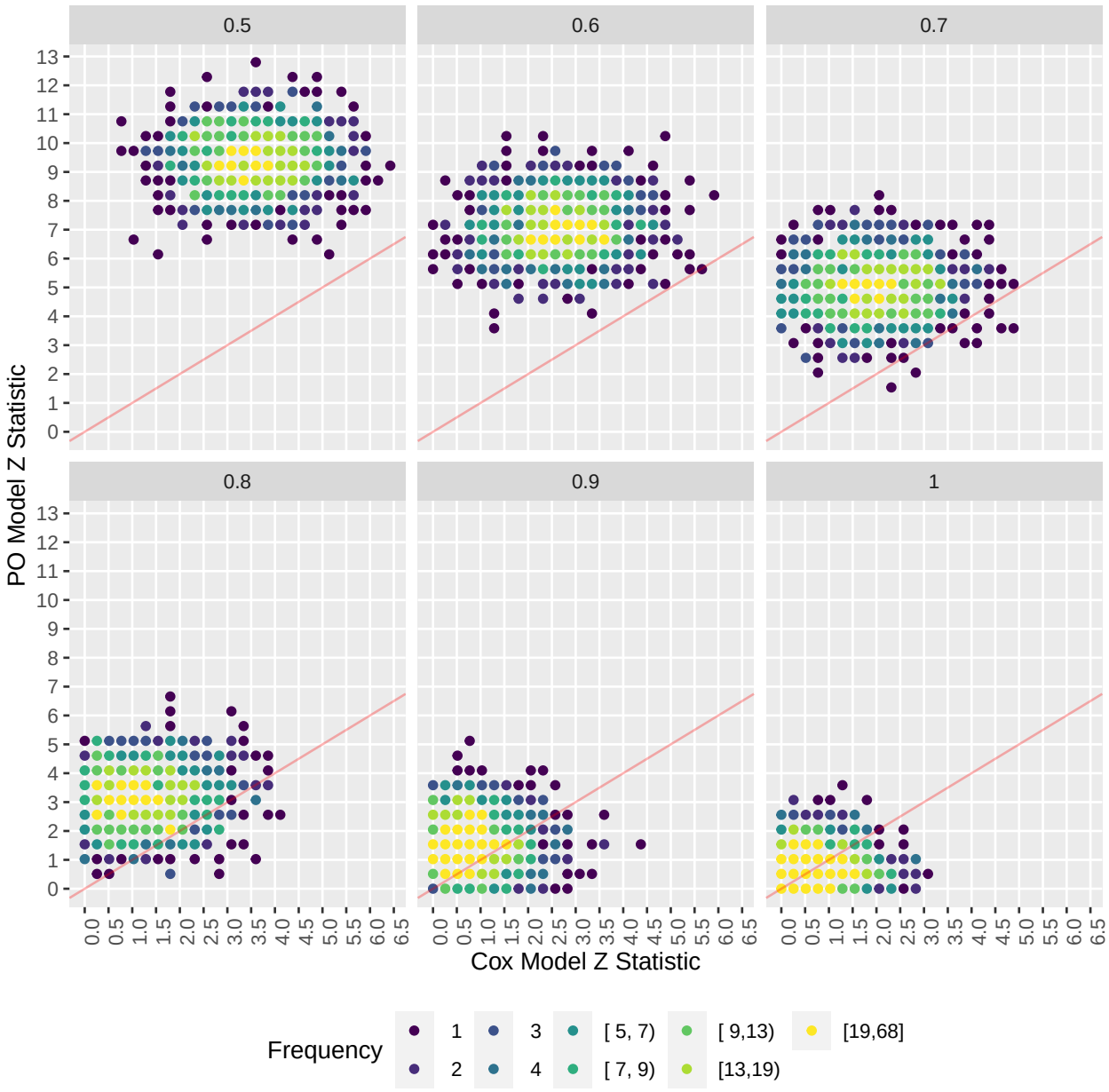

Figure 23: Scatter plot of Cox and PO model Z statistics. Partial PO model with 28 days of measurements.

Operating Characteristics Under Response-Dependent Sampling

In COVID-19 therapeutic trials with longitudinal outcomes, many of the trials randomize patients once they arrive to hospital. While in hospital, the ordinal scale may be assessed frequently, e.g., daily. Once the patient is discharged home, even if this is not an absorbing state (patient can return to hospital if condition worsens again), it may be difficult to do daily outcome assessments. Starting with the previous simulation of 28 days of follow-up for each patient (less for patients reaching absorbing state `Y=4`), let's not sample patients as frequently once they reach `Y=1`. Specifically, the planned assessment times starting at day t that the patient first reached `Y=1` will be `t+3`, `t+6`, `t+9`, ... $\min(28, T_a)$ where T_a is the time until $Y = 4$ if Y was ever 4 in the 28 days of follow-up. We construct an `rdsample` function that accomplishes this. The function returns `NULL` if `Y=1` was not reached before day 28 or if no times were dropped after reaching `Y=1`.

```
rdsamp <- function(times, y) {
    if(! any(y == 1)) return(NULL)
    # Find index of first observation reaching y=1
    j <- min(which( y == 1 ))
    if(j == 28) return(NULL)
    planned.times <- c(if(j > 1) times[1 : (j - 1)],
                                        seq(times[j], max(times), by=3))
    if(length(planned.times) == length(times)) return(NULL)
    planned.times
    }

# Test the function
list(rdsamp(1, 4),
     rdsamp(1:3, 2:4),
     rdsamp(1:28, rep(3, 28)),
     rdsamp(1:28, c(rep(3, 27), 1)),
     rdsamp(1:28, c(2, 3, 1, rep(2, 25))) )
```

```
[[1]]
NULL

[[2]]
NULL

[[3]]
NULL

[[4]]
NULL

[[5]]
 [1]  1  2  3  6  9 12 15 18 21 24 27
```

Simulate 1000 clinical trials with reduced frequency of sampling post reaching `Y=1`.

```
initial  <- c('1'=0.02, '2'=0.75, '3'=0.23)   # probabilities of being in baseline states
ors      <- c(0.7, 0.8, 1.0)
formula   <- y ~ yprev * pmax(gap - 2, 0) + time * group
est <- dosim(times=times, g=g, initial=initial, intercepts=ints,
                        ors=ors, formula=formula, ppo=ppo,
                        groupContrast=contrvgam, timecriterion=function(y) y == 1,
                        rdsample=rdsamp, seed=2, file='simrds.rds')
```

```
Median of 1000 logistic regression coefficients for each OR

    (Intercept):1 (Intercept):2 (Intercept):3 yprev2 yprev3 pmax(gap - 2, 0)
0.7        2.1629       -2.0780       -4.8988 1.1061 3.7482           1.1521
0.8        2.1702       -2.0871       -4.8988 1.1042 3.7559           1.1881
1          2.1579       -2.1032       -4.9302 1.1129 3.7774           1.2186
     time:1  time:2  time:3 group2 yprev2:pmax(gap - 2, 0)
0.7 -0.1586 -0.0550 -0.8116 0.0346                 -0.9596
0.8 -0.1590 -0.0553 -0.8196 0.0218                 -0.9556
1   -0.1595 -0.0560 -0.8198 0.0015                 -0.9684
    yprev3:pmax(gap - 2, 0) time:group2
0.7                 -2.6200     -0.0173
```

```
0.8                 -2.6455     -0.0108
1                   -2.6340     -0.0003

Standard deviation of simulated group effects and square root of median of estimated variances

    OR    SD SDest
1: 0.7 0.100 0.103
2: 0.8 0.102 0.101
3: 1.0 0.100 0.099
```

As before we also check the performance of a Cox two-sample test for time to $Y = 1$ as well as that of the ordinal longitudinal model.

Let's see how much power is lost by the decreased post `Y=1` sampling frequency. Here we look at only three odds ratios. The data model must once again incorporate gap discounting by interacting `gap` with previous states.

```
examSim(est, timedist=FALSE, hist=FALSE, scatter=FALSE,
        desc='Partial PO model with response-dependent measurement times')
```

```
Comparison of OR and HR

    OR   HR
1: 0.7 0.85
2: 0.8 0.90
3: 1.0 1.00

Power of Cox test for time until Y=1

    OR power
1: 0.7  0.49
2: 0.8  0.24
3: 1.0  0.05

Power of Markov proportional odds model test

    OR power
1: 0.7 0.996
2: 0.8 0.797
3: 1.0 0.054

Spearman rho correlation between Markov and Cox model chi-square: 0.5
```

There is some power loss with OR=0.8. The loss would be expected to be smaller had $Y = 1$ been more of an absorbing state, i.e, the probability of staying in that state been greater than the 0.9 used in our simulation model.

Creating a Simulation Model From a Previous Study

VIOLET 2 was a randomized clinical trial of seriously ill adults in ICUs to study the effectiveness of vitamin D vs. placebo. Vitamin D was not effective. VIOLET 2 is a rich dataset having daily ordinal outcomes assessed for 28 days. Due to an apparent anomaly in ventilator use on day 28 we restrict attention to days 1-27. The original NEJM paper focused on 1078 patients with confirmed baseline vitamin D deficiency. The analyses presented here use 1352 of the original 1360 randomized patients. We fit a partial proportional odds Markov transition model, similar to the one used above, to VIOLET 2. We start with a side analysis that examines the relative efficiency of using only some of the 27 days of observation.

The VIOLET 2 data are not yet publicly available.

```
require(VGAM)
```

```
dcf <- readRDS('dcf.rds')
setDT(dcf, key=c('id', 'day'))   # includes records with death carried forward
dcf <- dcf[! is.na(bstatus) & day < 28,
           .(  id, day, bstatus, status, treatment, ddeath)]
# Since baseline status did not distinguish ARDS from on ventilator,
# we must pool follow-up status likewise
s <- dcf[, bstatus]
levels(s) <- list('Vent/ARDS'='ARDS', 'Vent/ARDS'='On Vent',
                  'In Hospital/Facility'='In Hospital')
dcf[, table(bstatus, s)]
```

```
             s
bstatus       Vent/ARDS In Hospital/Facility
  ARDS             2727                    0
  On Vent         10071                    0
  In Hospital         0                23679
```

```
dcf[, bstatus := s]
d   <- dcf

# Get day 14 status carrying death forward
status14 <- d[day == 14, status, by=.(treatment, id, bstatus)]

# Before truncating records at death, get the observed frequency distribution
relf <- function(y) as.list(table(y) / length(y))
w1 <- d[, relf(status), by=day]
setnames(w1, 'day', 'time')
w1[, time := as.integer(time)]
w1[, type := 'VIOLET 2 Results']

# Compute correlation matrix
w <- dcast(d[, .(id, day, y=as.numeric(status))], id ~ day, value.var='y')
rviolet.obs <- cor(as.matrix(w[, -1]), use='pairwise.complete.obs')

# Don't carry death forward when fitting Markov models:
d <- d[day <= ddeath]
setkey(d, id, day)

# Get distribution of ventilator/ARDS-free days observed in VIOLET 2
# Pool treatment groups
vfd <- function(time, y)   # time is not used
  as.integer(ifelse(any(y == 'Dead'), -1, sum(y != 'Vent/ARDS')))
vvfd <- d[, .(vf = vfd(day, status)), by=id]
vvfd <- relfreq(vvfd[, vf])
```

Estimating Efficiency From the Study

The model is simplified to assume a constant treatment effect over time. We vary the number of days of observation used, and compute the variance of the estimated treatment effect for each model. Days are sampled in such a way as to approximate equal gap times between measurements so that gaps can be ignored. Efficiency is the minimum variance (variance using measurements on all 27 days) divided by the variance using a given number of almost equally-spaced days.

The non-PO part of the model has 6 parameters: 2 time components × 3 states corresponding to PO model intercepts. Two time components were required to accommodate the very low number of patients who were sent home on the first day. The linear spline used here essentially grants an exception at $t = 1$.

```
V <- numeric(27)
a <- 'treatmentVitamin D'

# Make sampled days as evenly distributed as possible

for(ndays in 1 : 27) {
  if(ndays == 1) {
    # First analysis uses an ordinary unconditional PO model on day 14
        f <- vgam(ordered(status) ~ bstatus + treatment,
              cumulative(reverse=TRUE, parallel=TRUE), data=status14)
    }
  else {
    s  <- round(seq(1, 27, length=ndays))
    ds <- d[day %in% s]
    ds[, pstatus := shift(status), by=id]
    ds[is.na(pstatus), pstatus := bstatus]
        ds[, pstatus := pstatus[drop=TRUE]]  # drop unused previous status Dead
        f <- if(ndays < 5) vgam(ordered(status) ~ pstatus + treatment + day,
              cumulative(reverse=TRUE, parallel=FALSE ~ day),
              data=ds) else
          vgam(ordered(status) ~ pstatus + treatment + lsp(day, 2),
              cumulative(reverse=TRUE, parallel=FALSE ~ lsp(day, 2)),
              data=ds)

  }
  V[ndays] <- vcov(f)[a,a]
}
if(outfmt == 'pdf') pr(obj=round(V, 5))
```

```
 [1] 0.01205 0.01130 0.00587 0.00482 0.00530 0.00464 0.00418 0.00396 0.00378
[10] 0.00364 0.00355 0.00342 0.00325 0.00325 0.00312 0.00309 0.00303 0.00298
[19] 0.00288 0.00285 0.00282 0.00276 0.00272 0.00270 0.00263 0.00261 0.00257
```

```
w <- rbind(data.frame(days=1:27, y=V[27]/V, type='Relative Efficiency'),
           data.frame(days=1:27, y=V[1]/V,
                      type='Effective Sample Size Per Subject'))
txt <- with(w, paste0(type, '<br>', round(y, 2), '<br>Day:', days))
ggp(ggplot(w, aes(x=days, y=y, label=txt)) + geom_line() +
  facet_wrap(~ type, scales='free_y', nrow=2) +
    xlab('Number of Days Sampled') + ylab(''))
```

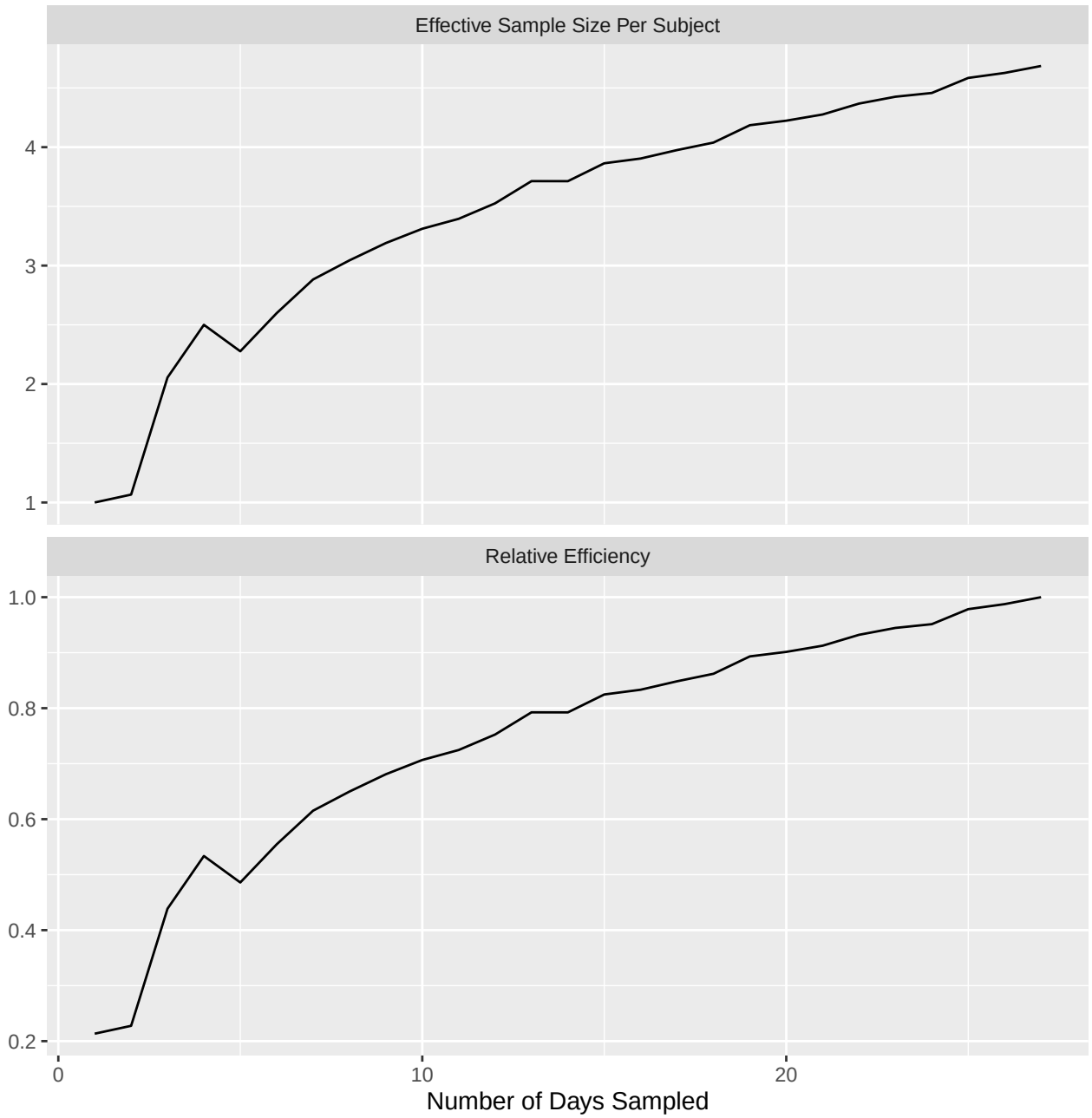

Figure 24: Effective sample size per subject with reference to using only the response at day 14 (top panel); relative efficiency of a treatment comparison with reference to using all 28 days (bottom panel). Both graphs are based on the variance of the log odds ratio for state transitions.

Sampling 3 days is estimated to have the effect of doubling the number of patients compared to sampling only day 14. Sampling all 27 days has the effect of increasing the sample size almost 5-fold.

Evidence for non-PO can be seen by computing Wald z statistics for the last six terms in the model.

```
round(coef(f), 4)
```

```
(Intercept):1 (Intercept):2
       3.7629       -2.5750
```

```
(Intercept):3 pstatusIn Hospital/Facility
          -10.7269                     6.3265
   pstatusHome          treatmentVitamin D
           15.7827                    -0.0017
lsp(day, 2)day:1           lsp(day, 2)day:2
           -0.4177                     0.4544
lsp(day, 2)day:3           lsp(day, 2)day':1
            1.2862                     0.4135
lsp(day, 2)day':2           lsp(day, 2)day':3
           -0.4971                    -1.3584
```

```
coef(f)[7:12] / sqrt(diag(vcov(f)[7:12, 7:12]))
```

```
 lsp(day, 2)day:1  lsp(day, 2)day:2  lsp(day, 2)day:3 lsp(day, 2)day':1
        -1.500879          3.090260          7.370848          1.467740
lsp(day, 2)day':2 lsp(day, 2)day':3
        -3.330681         -7.729806
```

Simulation Model From the Study

The last model fitted was for all 27 days. The `vgam` function's non-PO terms are parameterized to pertain to each category after the first (Dead). We write a `g` function for the state transitions, and save intercepts separately.

```
k     <- coef(f)
ints  <- k[1 : 3]
extra <- k[- (1 : 3)][-3]
g <- function(yprev, t, gap, X, parameter=0, extra) {
    tau   <- extra[1:2]
    kappa <- extra[3:8]
    lp <- matrix(0., nrow=length(yprev), ncol=3,
                 dimnames=list(as.character(yprev),
                 c('Vent/ARDS', 'In Hospital/Facility', 'Home')))
    tp <- pmax(t - 2, 0)   # second time term, from linear spline knot at t=2
    # 3 columns = no. distinct y less 1 = length of intercepts
    # lp[yp, ] is a 3-vector because the kappa components are split out into a 3-vector
    for(yp in yprev)
      lp[as.character(yp), ] <- tau[1] * (yp == 'In Hospital/Facility') +
                                tau[2] * (yp == 'Home') +
                                t * kappa[1:3] + tp * kappa[4:6] +
                                   parameter * (X == 2)
  lp
}
formals(g)$extra <- extra
```

Simulate n patients on placebo and compare the state occupancy frequencies with those observed in the trial, pooling treatments. n is set to 10× the number of patients in the trial sample we are using to make comparisons easy after we divide frequencies by 10. Just for comparison purposes we carry death forward. For the initial state we sample from the observed proportions.

```
n <- length(unique(dcf$id)) * 10
n
```

```
[1] 13510
```

```
initialProp <- d[day==1, relfreq(bstatus)]
set.seed(13)
```

```
initialSamp <- sample(names(initialProp), n, TRUE, prob=initialProp)

# Compute state occupancy probabilities for group 1
# We need to compute a weighted average over the result for each baseline state
#
sop <- list()
for(initial in names(initialProp))
  sop[[initial]] <- soprobMarkovOrd(levels(d$status), 1:27, initial=initial, absorb='Dead',
                                                          intercepts=ints, g=g, X=1)
avgProp <- initialProp['Vent/ARDS']            * sop[['Vent/ARDS']] +
             initialProp['In Hospital/Facility'] * sop[['In Hospital/Facility']]
w2 <- data.table(avgProp)
w2[, time := as.numeric(rownames(w2))]
w2[, type := 'Model Fitted to VIOLET 2']

plotsop(avgProp)
```

```
Occupancy probabilities for group 1:

    Dead Vent/ARDS In Hospital/Facility  Home
1  0.012     0.311                0.650 0.028
2  0.028     0.252                0.603 0.118
3  0.041     0.206                0.557 0.196
4  0.051     0.171                0.514 0.264
5  0.060     0.143                0.474 0.323
6  0.068     0.121                0.438 0.374
7  0.074     0.104                0.405 0.417
8  0.079     0.090                0.375 0.455
9  0.084     0.079                0.349 0.488
10 0.088     0.070                0.325 0.517
11 0.092     0.063                0.304 0.541
12 0.095     0.057                0.285 0.563
13 0.098     0.052                0.268 0.582
14 0.101     0.048                0.253 0.598
15 0.104     0.045                0.239 0.612
16 0.106     0.042                0.227 0.624
17 0.109     0.040                0.216 0.635
18 0.111     0.038                0.206 0.645
19 0.113     0.037                0.197 0.653
20 0.115     0.036                0.189 0.660
21 0.117     0.035                0.182 0.666
22 0.119     0.035                0.175 0.671
23 0.121     0.034                0.169 0.676
24 0.123     0.034                0.164 0.680
25 0.125     0.034                0.159 0.683
26 0.127     0.034                0.154 0.685
27 0.128     0.034                0.150 0.687
```

```
s <- simMarkovOrd(n=n, y=levels(d$status), times=1:27,
                  initial=initialSamp,
                  X=c(group=1), absorb='Dead',
                  intercepts=ints, g=g, carry=TRUE)

# Relative frequencies just for day 27
```

```
with(subset(s, time == 27), relfreq(y))
```

```
x
                Dead            Vent/ARDS In Hospital/Facility
           0.12812731           0.03604737           0.14752036
                Home
           0.68830496
```

```
with(subset(dcf, day == 27), relfreq(status))
```

```
x
                Dead            Vent/ARDS In Hospital/Facility
           0.14433753           0.04219097           0.15099926
                Home
           0.66173205
```

```
s <- setDT(s, key=c('id', 'time'))
w3 <- s[, relf(y), by=time]
w3[, type := 'Data Simulated from Fitted Model']

w <- dcast(s[, .(id, time, y=as.numeric(y))], id ~ time, value.var='y')
rviolet.sim <- cor(as.matrix(w[, -1]))
```

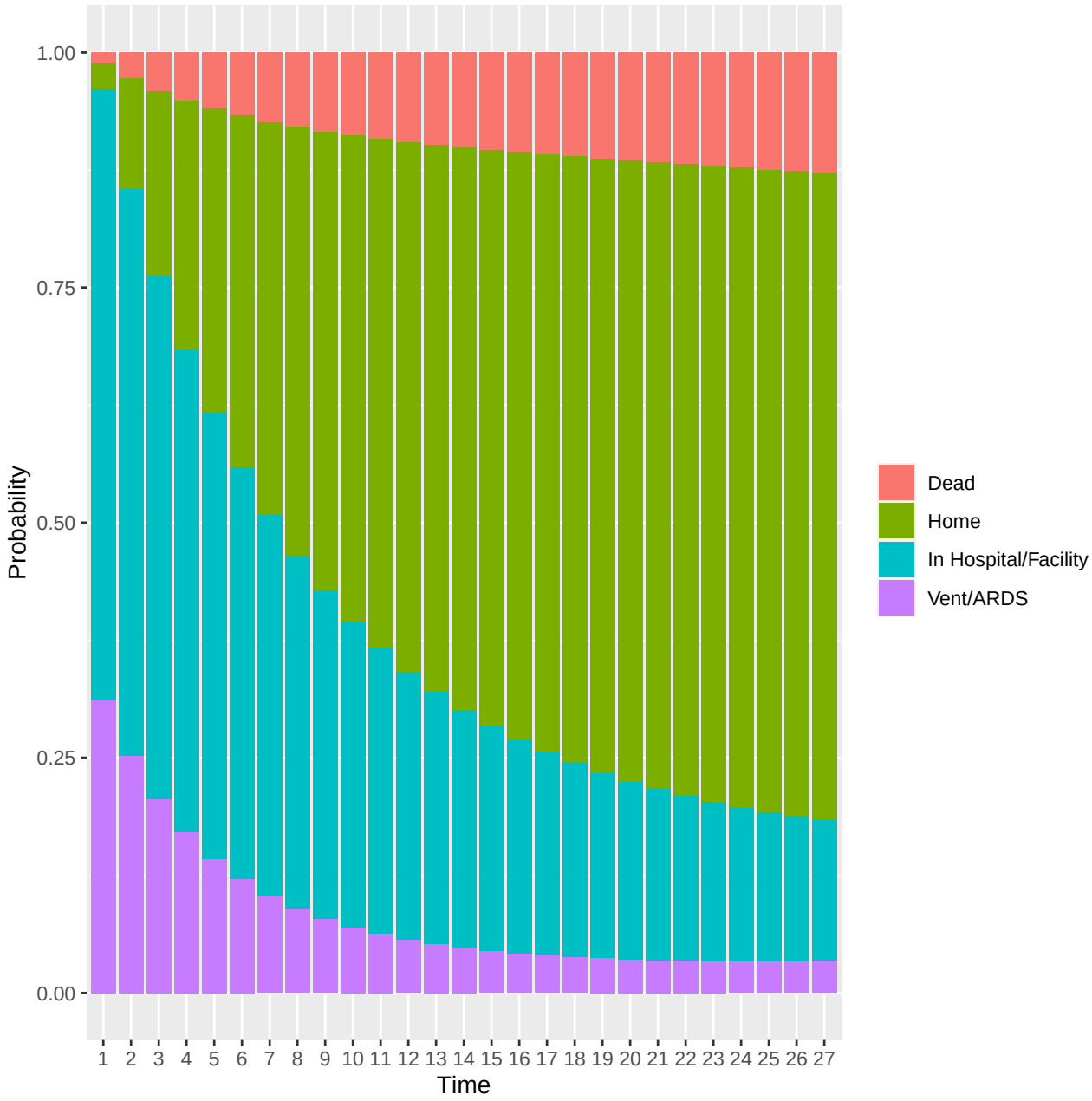

Figure 25: State occupancy probabilities averaged over the observed VIOLET 2 distribution of initial states, for treatment 1

Compare Serial Correlation Patterns from the Study and Simulated Data

Let's look at a variogram-like display of the observed correlations between times within subject to see how correlations vary with changing gap in measurement times. We compare this to the correlations seen in the simulated data.

```
dv <- cbind(ggcorr(rviolet.obs, 'data'), type='Observed')
ds <- cbind(ggcorr(rviolet.sim, 'data'), type='Simulated')
db <- rbind(dv, ds)
ggp(ggplot(db, aes(x=delta, y=r, label=txt)) +
    geom_point() + geom_smooth(method=loess) +
```

```
    facet_wrap(~ type) +
    xlab('Absolute Time Difference, Days') + ylab('r'))
```

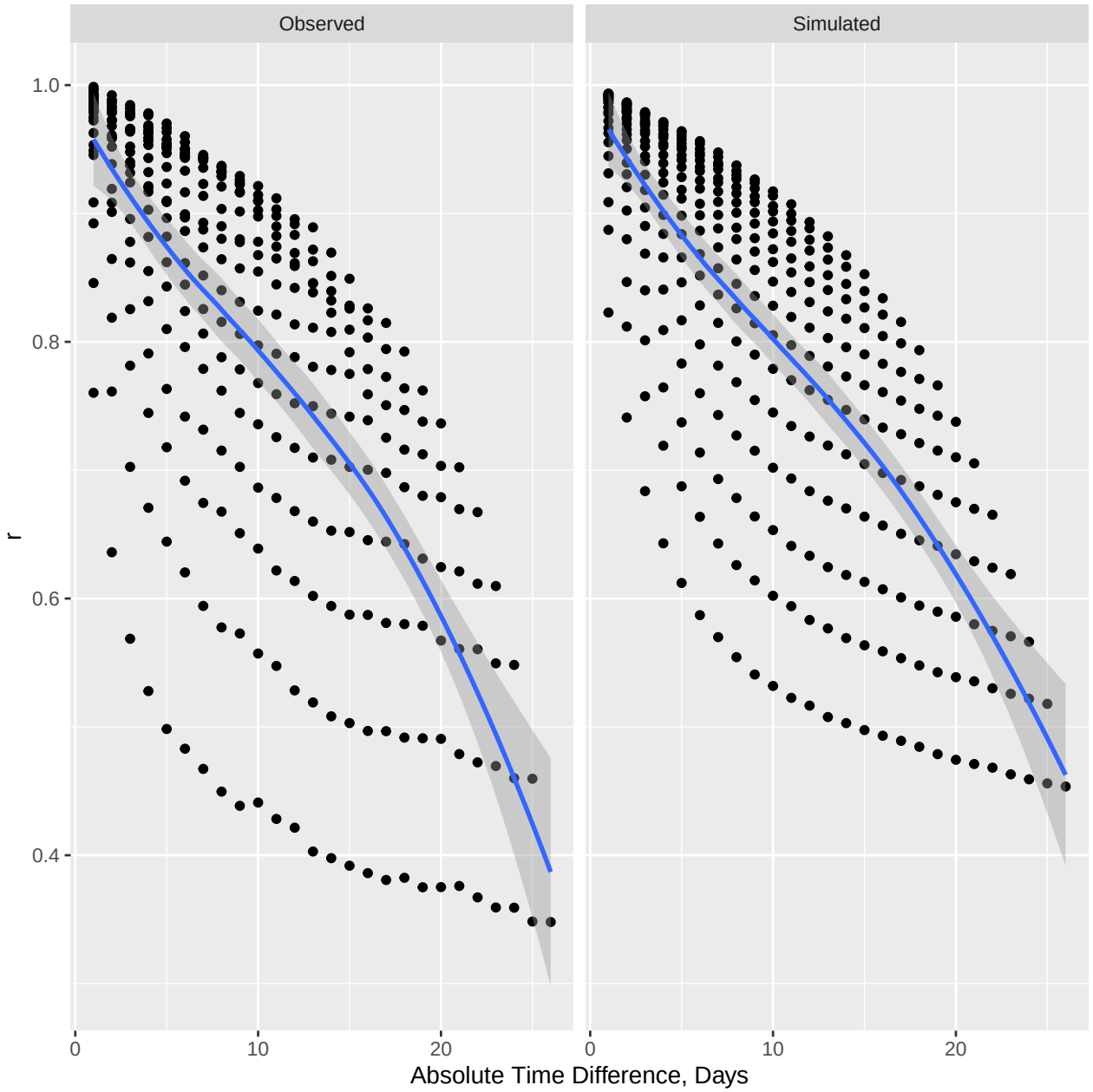

Figure 26: Serial correlation pattern within patient over time observed in VIOLET 2 along with correlations computed from a large random sample from the fitted partial proportional odds ordinal state transition model. In this variogram, the curves going horizontally and downwards represent individual days, and the vertical spread represents a non-isotonic serial correlation pattern. In other words, the correlation between two times within patient depends not only on the (unsigned) time gap but also on absolute time.

The model's correlation pattern is in good agreement with that observed in the study data, except for day 1.

Compare State Probabilities from Study, Model Parameters, and Data Simulated from the Model

```
w <- rbind(w1, w2, w3)
r <- melt(w, measure.vars=levels(dcf$status),
          variable.name='y', value.name='p')
txt <- with(r, paste0(type, '<br>',
               y, '<br>t=', time, '<br>p=', round(p, 3)))
ggp(ggplot(r, aes(x=time,
           y=p, fill=y, label=txt)) + geom_col() +
                    facet_wrap(~ type, ncol=1) +
           guides(fill = guide_legend(title='')) + xlab('Day') +
           ylab('Probability'))
```

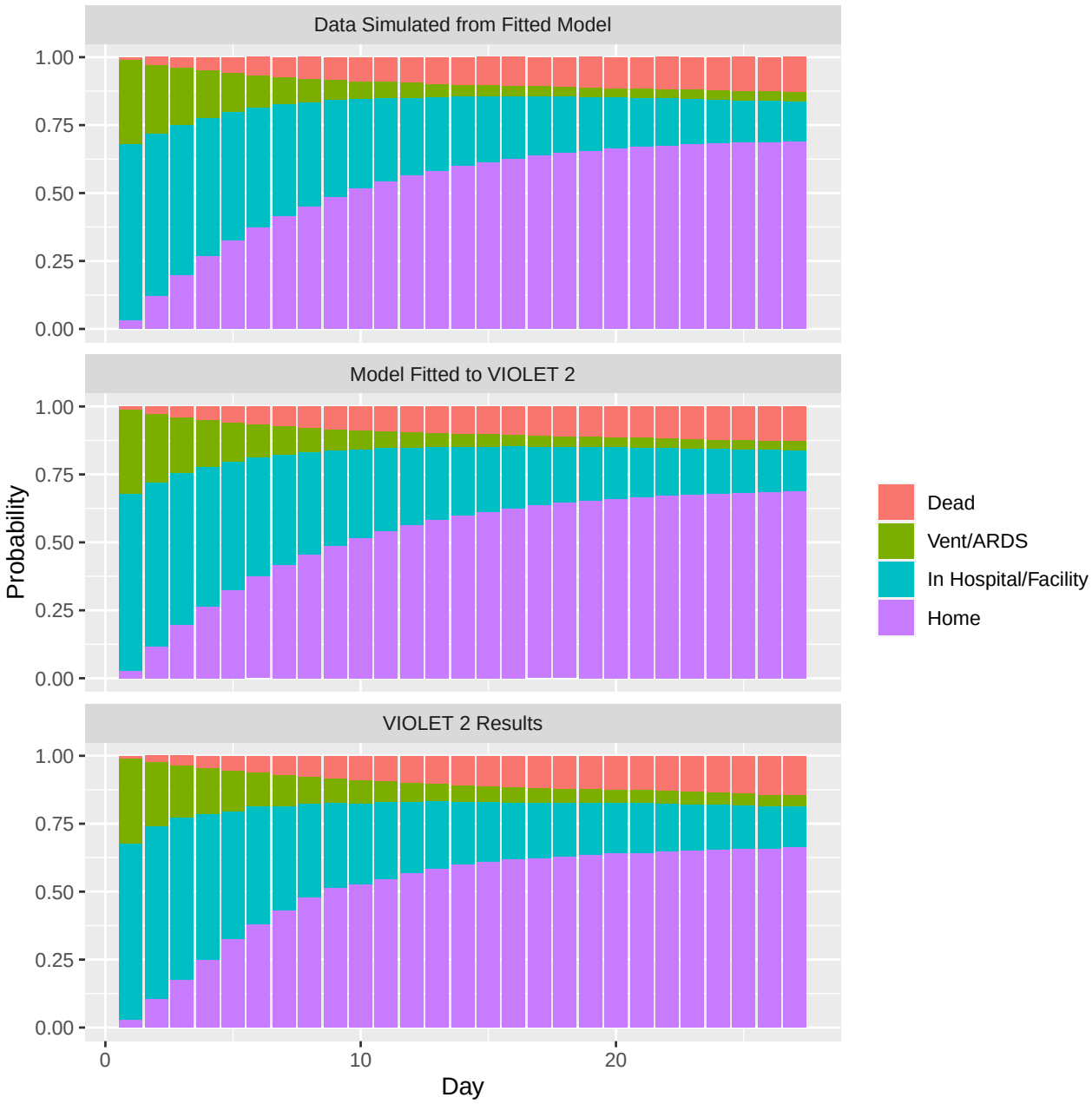

Figure 27: Observed VIOLET 2 state occupancy probabilities, probabilities computed exactly from the fitted model parameters, and proportions from simulating new patients from the fitted model

```
gt <- if(outfmt == 'pdf')
 'Simulated data, fitted model, and VIOLET 2 results (left to right)' else ''
ggp(ggplot(r, aes(x=type, y=p, color=y, label=txt)) + geom_point() +
  facet_wrap(~ time) +
  guides(color = guide_legend(title='')) +
  theme(axis.text.x = element_blank()) +  # element_text(angle=90)) +
  xlab('') + ylab('Probability') + ggtitle(gt))
```

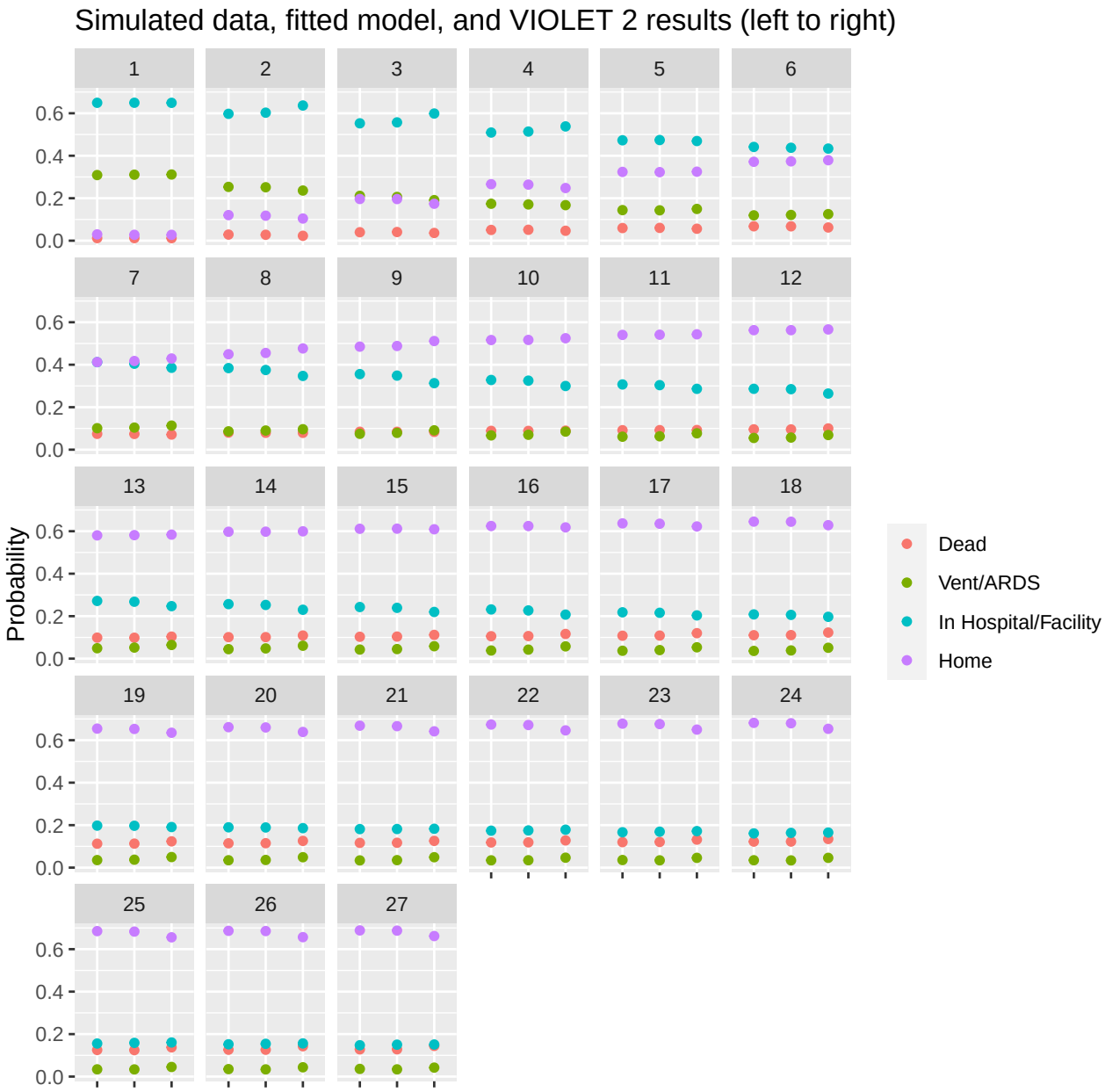

Figure 28: Same quantities as in the previous graph but separated by time so that agreement across the three estimates may be more readily seen

Compare Distribution of Ventilator/ARDS-free Days from Study and Data Simulated from the Model

Here we use data simulated under OR=1.

```
dv <- data.frame(y=as.numeric(names(vvfd)), unname(vvfd),
                 type='VIOLET 2')

# Summarize simulated data
simvf <- s[, .(y = vfd(time, y)), by=id]
r <- relfreq(simvf$y)
```

```
d2 <- data.frame(y=as.numeric(names(r)), unname(r), type='Data Simulated from Model')
dv <- rbind(dv, d2)
dv$txt <- with(dv, paste0(type, '<br>', round(Freq, 3)))

ggp(ggplot(dv, aes(x=y, y=Freq, label=txt)) + geom_col() +
  facet_wrap(~ type, nrow=2) +
  guides(color = guide_legend(title='')) +
  xlab('Ventilator/ARDS-free Days') +
    ylab('Proportion'))
```

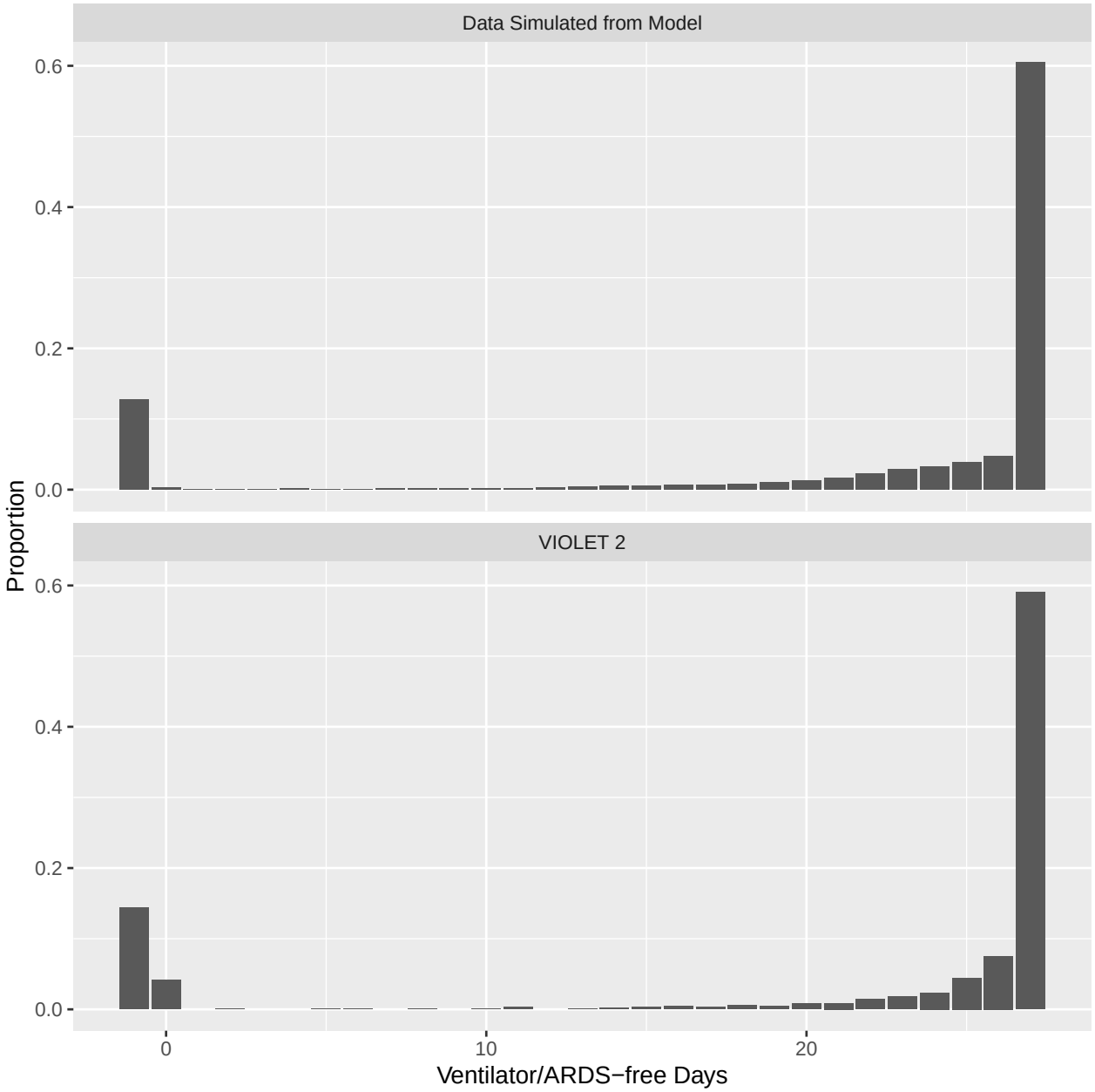

Figure 29: Distribution of observed VIOLET 2 ventilator/ARDS-free days (-1 for death) compared with the same summary measure computed on data simulated from the model

Simulate Performance of Statistical Methods on Studies Like VIOLET 2

Next simulate 1000 clinical trials using this VIOLET 2-based simulation model, with treatment odds ratios of 1.0 and 1.3 (it's greater than 1 because larger Y are better in this setting) for ordinal state transition probabilities. The main purposes of this simulation is to compare the frequentist power of the Markov PO model with that of a "time to home" Cox model test, and with a Wilcoxon test of ventilator/ARDS-free days counting death as -1 day. The clinical trials, unlike VIOLET 2, have 600 patients.

```
formula <- ordered(y) ~ yprev + group + lsp(time, 2)
ppo     <- ~ lsp(time, 2)

est <- dosim(y=levels(d$status), absorb='Dead',
             times=1:27, g=g,
             initial=c(Home=0, initialProp), intercepts=ints,
                         ors=c(1, 1.3), formula=formula, ppo=ppo,
                         timecriterion=function(y) y == 'Home',
                         sstat=vfd,
                         seed=13, file='simv.rds')
```

```
Median of 1000 logistic regression coefficients for each OR

    (Intercept):1 (Intercept):2 (Intercept):3 yprevIn Hospital/Facility
1          3.8133       -2.5726       -10.766                    6.3345
1.3        3.8098       -2.5684       -10.754                    6.3263
    yprevHome group2 lsp(time, 2)time:1 lsp(time, 2)time:2 lsp(time, 2)time:3
1     15.8012 0.0014            -0.4533             0.4510             1.3007
1.3   15.8172 0.2623            -0.4497             0.4569             1.3030
    lsp(time, 2)time':1 lsp(time, 2)time':2 lsp(time, 2)time':3
1                0.4537             -0.4951             -1.3762
1.3              0.4457             -0.4986             -1.3741

Standard deviation of simulated group effects and square root of median of estimated variances

    OR    SD SDest
1: 1.0 0.075 0.077
2: 1.3 0.075 0.078
```

```
examSim(est, hist=FALSE, timeevent='home', desc='Simulation of VIOLET 2')
```

```
Comparison of OR and HR

    OR   HR
1: 1.0 1.00
2: 1.3 0.77

Power of Cox test for time until home

    OR power
1: 1.0  0.04
2: 1.3  0.80

Power of Markov proportional odds model test

    OR power
1: 1.0 0.053
2: 1.3 0.937
```

```
Spearman rho correlation between Markov and Cox model chi-square: 0.89

Power of Wilcoxon test on vent/ards-free days

    OR power
1: 1.0 0.041
2: 1.3 0.304
```

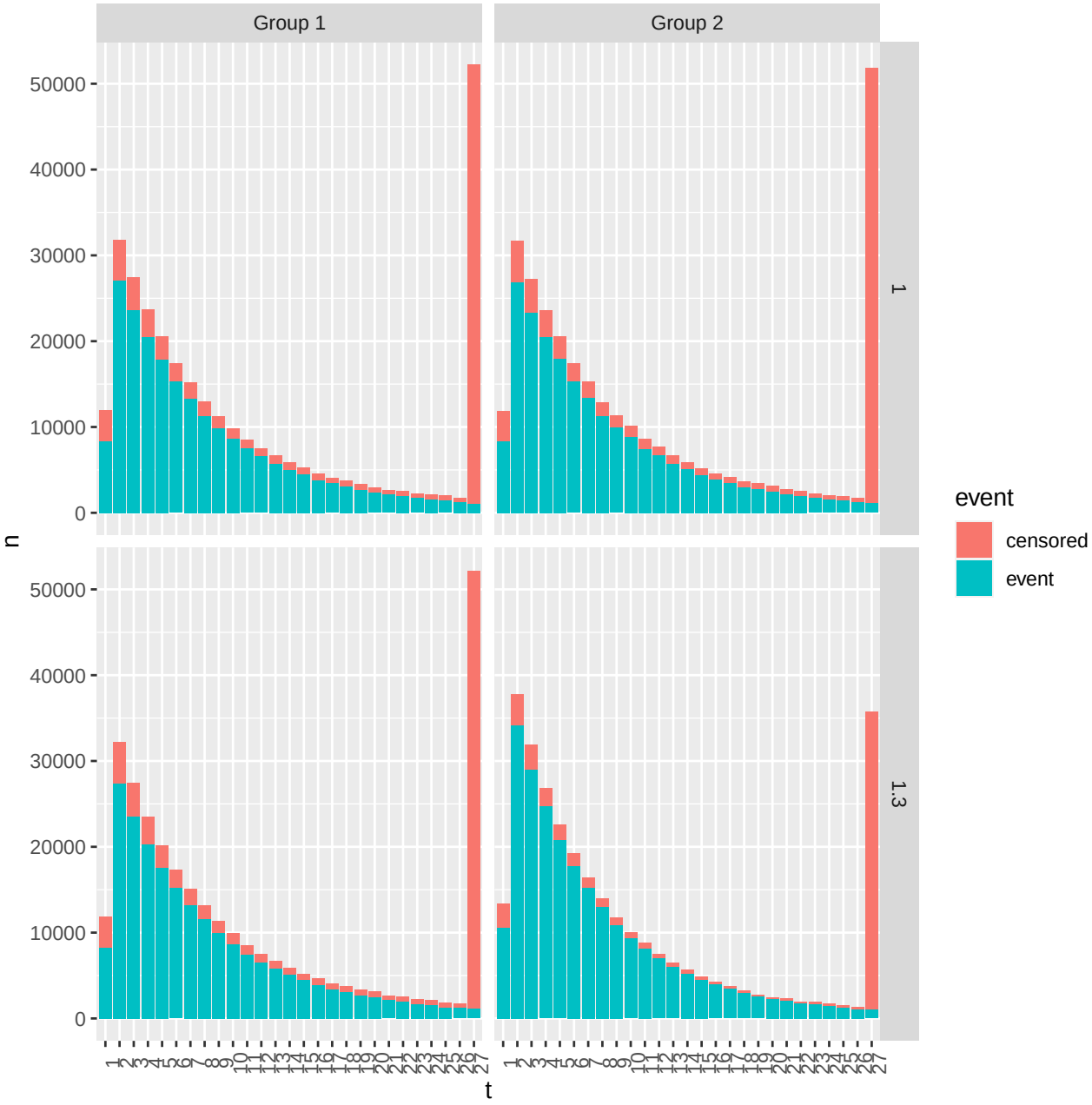

Figure 30: Time to home by group and OR, over simulations. Simulation of VIOLET 2.

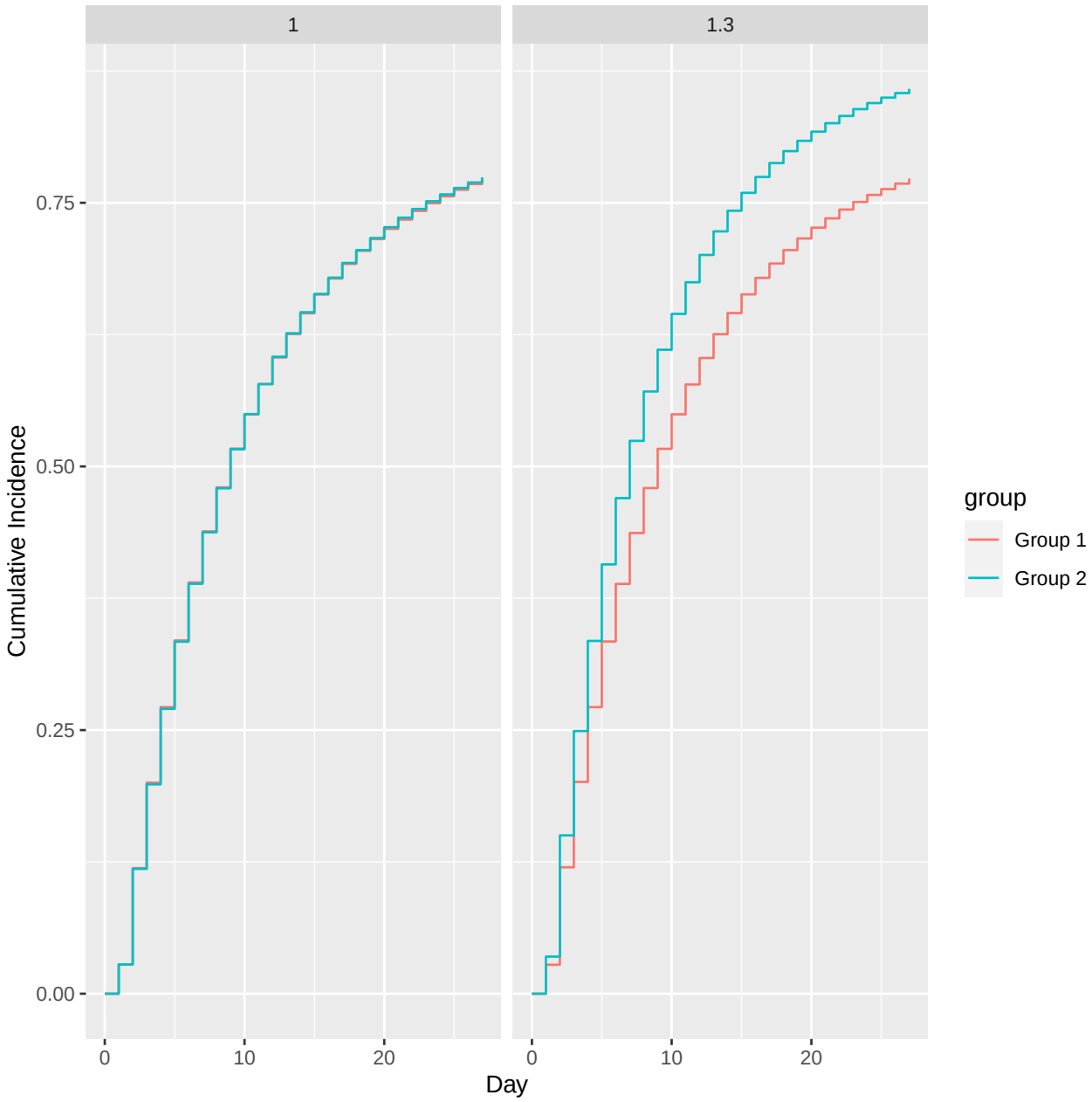

Figure 31: Cumulative incidence of home. Simulation of VIOLET 2.

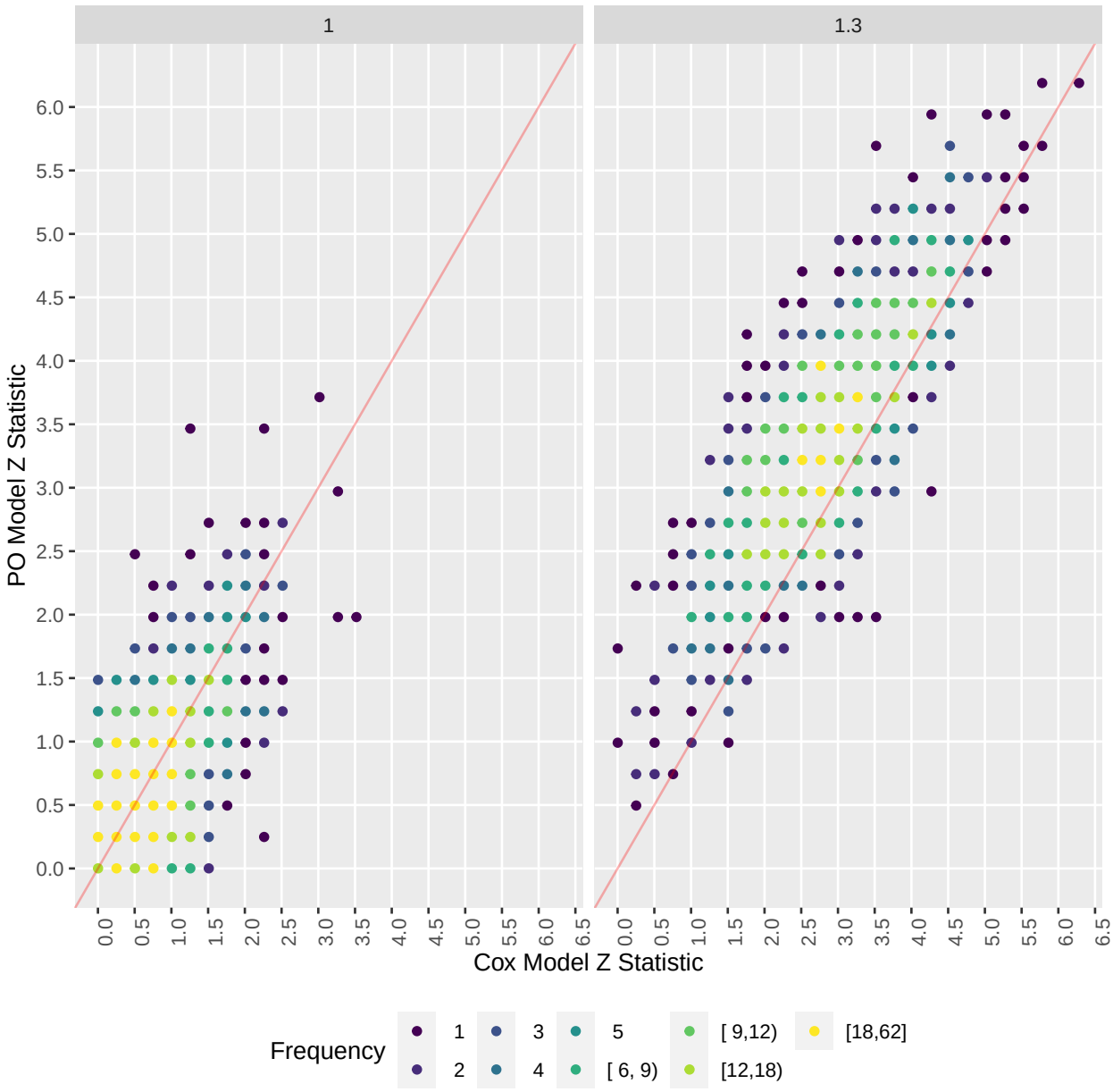

Figure 32: Scatter plot of Cox and PO model Z statistics. Simulation of VIOLET 2.

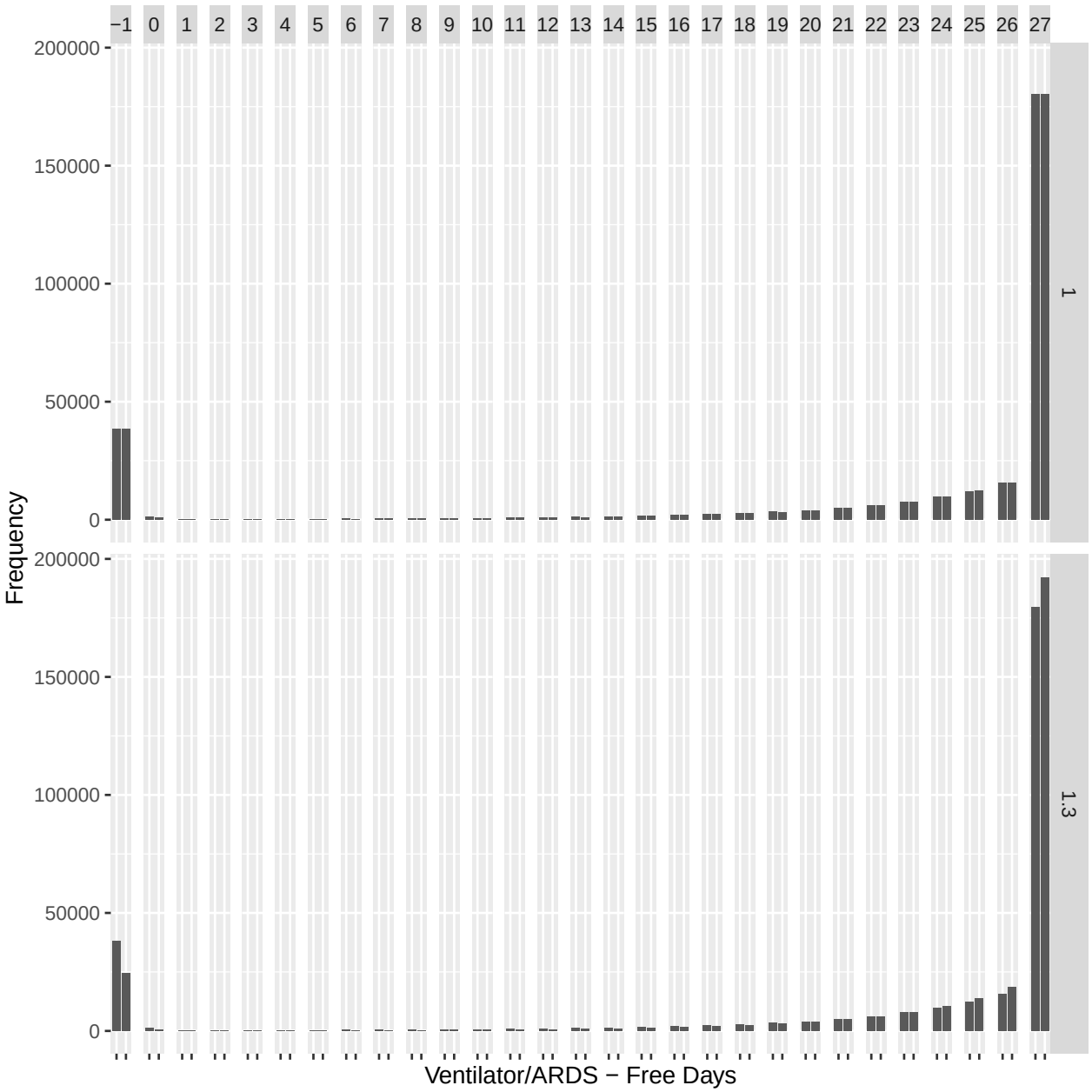

Figure 33: Distribution of ventilator/ARDS-free days by group and OR. Left bar of each pair is group 1, right bar is group 2. -1 represents death.. Simulation of VIOLET 2.

More Information

- Full R markdown script, `Hmisc` and service functions, and `pdf` version of this report: hbiostat.org/R/Hmisc/markov
- COVID-19 statistical resources: hbiostat.org/proj/covid19 including detailed analyses of the VIOLET 2 and ORCHID studies plus an examination of the handling of irregular measurement times in a Markov model
- Bayesian design and analysis resources: hbiostat.org/bayes
- Markov modeling references: hbiostat.org/bib/markov.html
- Longitudinal ordinal analysis references: hbiostat.org/bib/ordSerial.html

- General references on longitudinal data analysis: hbiostat.org/bib/serial.html
- Introduction to the proportional odds model: hbiostat.org/bbr/md chapter 7
- More about the proportional odds model: hbiostat.org/rms

Computing Environment

To cite R in publication use:

R Core Team (2021). R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.