Supplement: A baseline method for nowcasting count data

2025-04-16

# Supplemental methods

## Mathematical model

This model was initially developed as a reference model for the COVID-19 hospitalisations nowcast challenge in Germany in 2021 and 2022 Wolffram et al. (2023). It uses preliminary case count and their delays in the form of reporting triangles, and uses empirical delay distributions to estimate yet-to-be-observed cases. Probabilistic nowcasts are generated using a negative binomial model with means from the point nowcast and dispersion parameters estimated from past nowcast errors. Below, we describe the mathematical details of each component of the model, starting with a definition of the notation used throughout.

### Notation

We denote as the number of cases occurring on time which appear in the dataset with a delay of . For example, a delay means that a case occurring on day arrived in the dataset on day , or on what is considered to be the first possible report date in practice. We only consider cases reporting within a maximum delay . The number of cases reporting for time with a delay of at most can be written as:

Such that is the “final” number of reported cases on time . Conversely, for

is the number of cases still missing after delay. We refer to to describe a random variable, for the corresponding observation, and for an estimated/imputed value. The matrix of available at a given time is referred to as a reporting matrix. In the case where all have yet to be observed (e.g.  is the current time), this reporting matrix is referred to as the reporting triangle, with all values in the bottom right corner of the triangle being missing, except for the first entry at .

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

### Delay distribution estimation

#### Estimate of the delay distribution from a reporting matrix

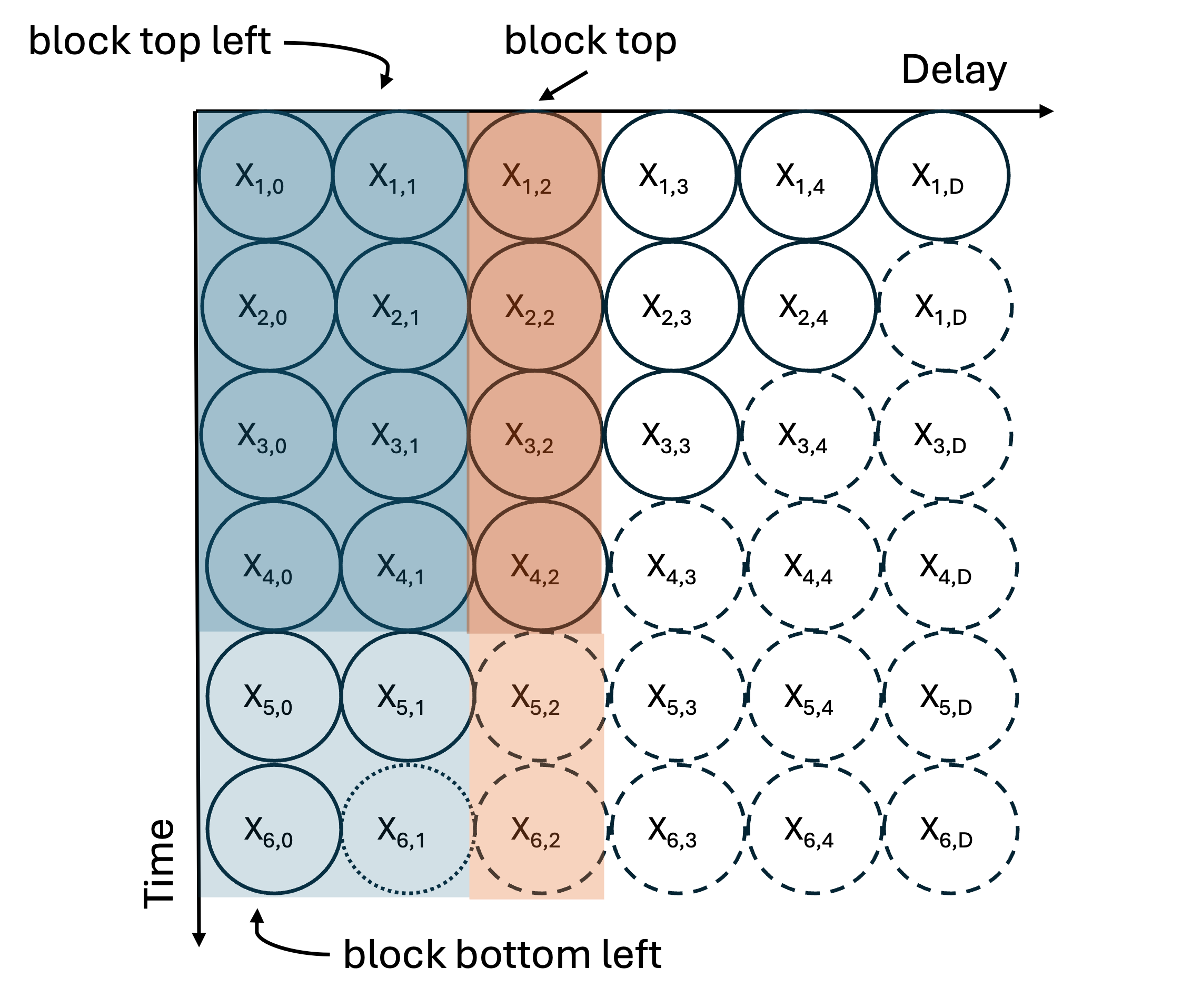
We can use a reporting matrix to compute an empirical estimate of the delay distribution, . The empirical delay distribution, can be computed directly from the reporting matrix using the latest reference times via:

Where the numerator is the sum of all the observations across reference times for a particular delay of interest, and the denominator is the sum across all reference times and indexed delays in the reporting matrix .

#### Estimate of the delay distribution from a reporting triangle

In the case where we have missing values in the bottom right (i.e. we have a reporting triangle), we need to use the multiplicative model to generate a point nowcast matrix containing a mixture of observed and imputed values. Then we can compute the delay distribution as described above for the reporting matrix case.

The multiplicative model works by iteratively “filling in” the reporting triangle starting from the bottom left, and moving column by column from left to right until the bottom right of the triangle is filled in.



Visual description of the iterative “completing” of the reporting triangle, moving from left to right and bottom to top. In this cases, we are imputing $x\_{t=6, d = 2}$ and $x\_{t=5, d= 2}$ assuming that the ratio between $x\_{t=1:4, d = 2}$ (block top), and $x\_{t=1:4, d=0:1}$ (block top left) holds for for $x\_{t=5:6, d = 2}$ (block bottom) and $x\_{t=5:6, d = 0:1}$ (block bottom left). In this example, $\hat{x}\_{t=6, d = 1}$ has already been imputed using the same approach, and we treat it as known going forward. This process is repeated across the reporting triangle to estimate all values outlined in the dashed lines.

The method requires at least one observation, at delay for the most recent reference time, located at the bottom left of the reporting triangle in Figure @ref(fig1) above. The method assumes that the values at each delay for the recent times, , will consist of the same proportion of the values previously reported for earlier times . To fill in the missing values for each column , we sum over the rectangle of completely observed reference dates for all columns (block top left) and sum over the column of completely observed reference dates for all of the entries in column (block left). The ratio of these two sums is assumed to be the same in the missing entries in column , so we use the entries observed up to for each incomplete reference date (block bottom left), and scale by this ratio to get the missing entries in column . This process is repeated for each delay from up to the maximum delay . At each iteration an additional reference time entry is computed as the delay increases.

The delay distribution is then estimated from the filled in reporting matrix, using the same algorithm as described above for the case of the complete reporting matrix.

### Point nowcast generation

To “fill in” the reporting triangle from the delay distribution, we need to estimate the expected total number of eventual observed cases , for each reference time . Let be the sum over all delays that have already been observed (up until ), such that and be the cumulative sum of the delay distribution, up until such that . By assuming that and , it can be shown (See [Zero-handling](#zero-handling-approximation) below) that the expected value of , the total number of reported cases on reference time , can be written as:

Then we can compute directly using the th element of

Where the number of reports at timepoint with delay is the product of the the expected total reports, and the proportion expected at that particular delay , .

### Uncertainty estimation

To estimate the uncertainty in the nowcasts, we use past nowcast errors. In this analysis, we assume a negative binomial observation model.

#### Generation of retrospective reporting triangles

We describe a method which generates retrospective reporting triangles to replicate what would have been available as of time , where for to generate retrospective reporting triangles.

To generate the set of reporting triangles, we work backwards from most recent to oldest, removing the last rows of the existing reporting triangle, to generate truncated reporting triangles. We then replace the bottom right of the triangle with NAs, assuming these would not have been observed as of , to generare retrospective reporting triangles.

The method uses each retrospective reporting triangle to re-estimate a delay distribution using the preceding reference times of the retrospective reporting triangle before , and recomputes a retrospective point nowcast matrix, for realizations of the retrospective reporting triangle (so different values).

Thus in order to estimate uncertainty using reference times, the total training volume must meet or exceed so that the oldest retrospective nowcast dataset at can use reference times for its point nowcast.

#### Generation of retrospective point nowcast matrices

From the reporting triangles, we apply the method described above to estimate a delay distribution from a reporting triangle and generate a point nowcast for each reporting triangle, to generate point nowcasts.

#### Fit predicted point nowcast vectors and obesrved counts to a negative binomial observation model at each forecast horizon

We quantify the uncertainty at the target quantity level, which is assumed, by default, to be the final count at each reference time, summed across reporting delays.

At each retrospective nowcast time , we compute the predicted and corresponding observed nowcast at each forecast horizon by summing across the reporting delays for all delays that have been observed as of time , as indicated below by the indicator function. We define the predicted nowcast for forecast horizon at retrospective nowcast time

and likewise the observed nowcast for forecast horizon at retrospective nowcast time

This generates pairs of predicted nowcasts and observed nowcasts for each forecast horizon . We assume that the observed nowcasts follow a negative binomial observation model with a mean of

for all .

We add a small number (0.1) to the mean to avoid an ill-defined negative binomial. This generates a vector of negative binomial dispersion parameters indexed starting at forecast horizon .

### Probabilistic nowcast generation

Using the dispersion parameters for each forecast horizon, for , we can generate probabilistic nowcast matrices by drawing samples from the negative binomial:

We can sample for any number of draws, and then use the draws to compute any desired quantiles to summarize the outputs.

### Zero-handling approximation

In order to produce a nowcast for a specific reference date, we use the number of already observed cases, and the delay PMF, to estimate the expected final count of cases . For notational simplicity, we will refer to as , to indicate the random variable for the total number of final cases at a particular reference time , and as to indicate the random variable for the number of cases at a particular reference time and delay .

We will denote the CDF of the delay distribution at a particular delay , simply by .

However, this simple calculation is undefined when the number of already observed cases is 0.

The following derivation addresses this practical issue based on assumptions about the underlying distributions for which the observed cases at each reference time and reporting delay are drawn from.

Our problem boils down to the following. If we assume that:

with known and an improper discrete uniform prior on :

We are then interested in:

This can be written out as:

Applying Bayes Theorem we have:

Because this simplifies to:

And substituting the probability mass function of the binomial distribution we get:

Plugging into we get the following (omitting terms for , which are 0):

Which is equivalent to:

Both the numerator and the denominator are known convergent series, with solutions available in standard libraries like Mathematica. We then get:

This formula allows us to have a defined even when .

### Description of KIT simple nowcast implementation issue and revised nowcasts

When verifying the KIT simple nowcast implementation, we noticed that the code used to generate the KIT simple nowcasts in real-time contained an additional indexed element in the delay PMF for all counts observed beyond the maximum delay thus far. In the generation of retrospective nowcasts, these values were being handled as having been observed as 0s, when in fact they had yet to be observed in all of the retrospective nowcasts. This resulted in a comparison of a 0 to a later observed value – which has the impact of inflating the dispersion estimates, and may explain why the KIT simple nowcast method in Wolffram et al. (Wolffram et al. 2023) overcovers relative to the other methods. In the latest implementation in the RESPINOW Hub (KIT Metrics Lab, Johannes Bracher, and Melanie Schienle 2025), the authors have removed the density beyond the maximum delay column from their implementation, and thus it no longer contains this issue.

In order to validate our baselinenowcast method against a revised version with the bug fixed, we regenerated the nowcasts using the pre-processed reporting triangle in the German Nowcast Hub repository (KIT Metrics Lab et al. 2025) and generated retrospective nowcasts, referring to these in the main text and supplement as the KIT simple nowcast revised.

## Additional figures

### German nowcast Hub validation using revised KIT simple nowcast

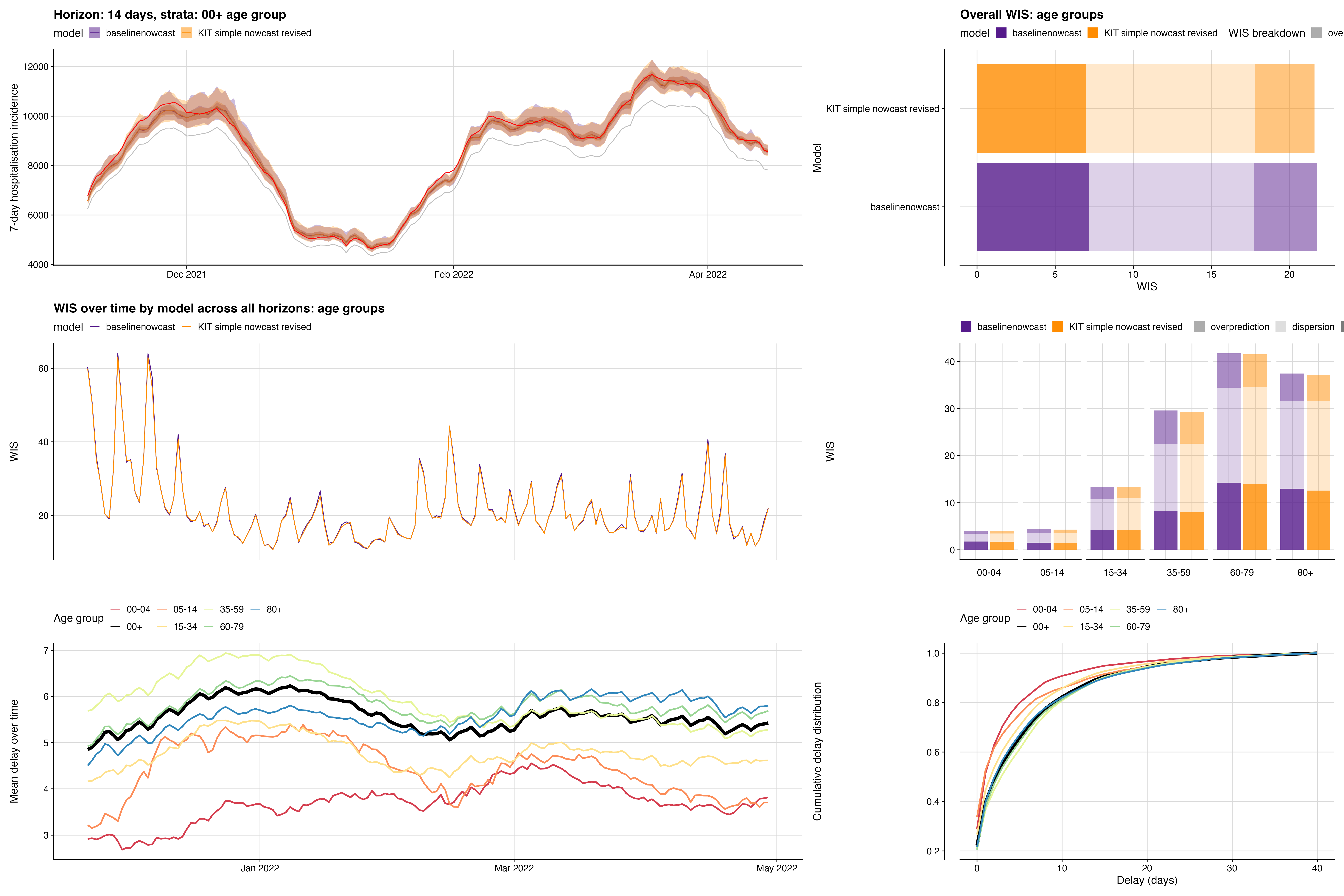


Fig.S1 Validation of baseline nowcasting model using German COVID-19 data compared to revised KIT simple nowcast. A. Weekly nowcasts from our baseline model and the KIT simple nowcast revised model against eventually observed national values, coloured by model, with data available at nowcast date (dotted lines). Nowcasts are shown daily with a 14-day horizon. B. Overall performance comparison between models, with decomposed WIS (dispersion, overprediction, underprediction) displayed as stacked bar charts for direct comparison. Stratified by national and age group nowcasts. C. Performance over time, displaying daily mean WIS scores summarised across age strata and 28 horizon days, coloured by model. D. Performance by age group, showing WIS scores across different age groups on the x axis with grouped bars for both models and stacked components displaying the decomposition of scores. E. Mean reporting delay over time, visualised as multiple coloured lines representing each age group, with the national average shown as a thicker black line. F. Empirical coverage at 50% and 90% prediction intervals for each model.

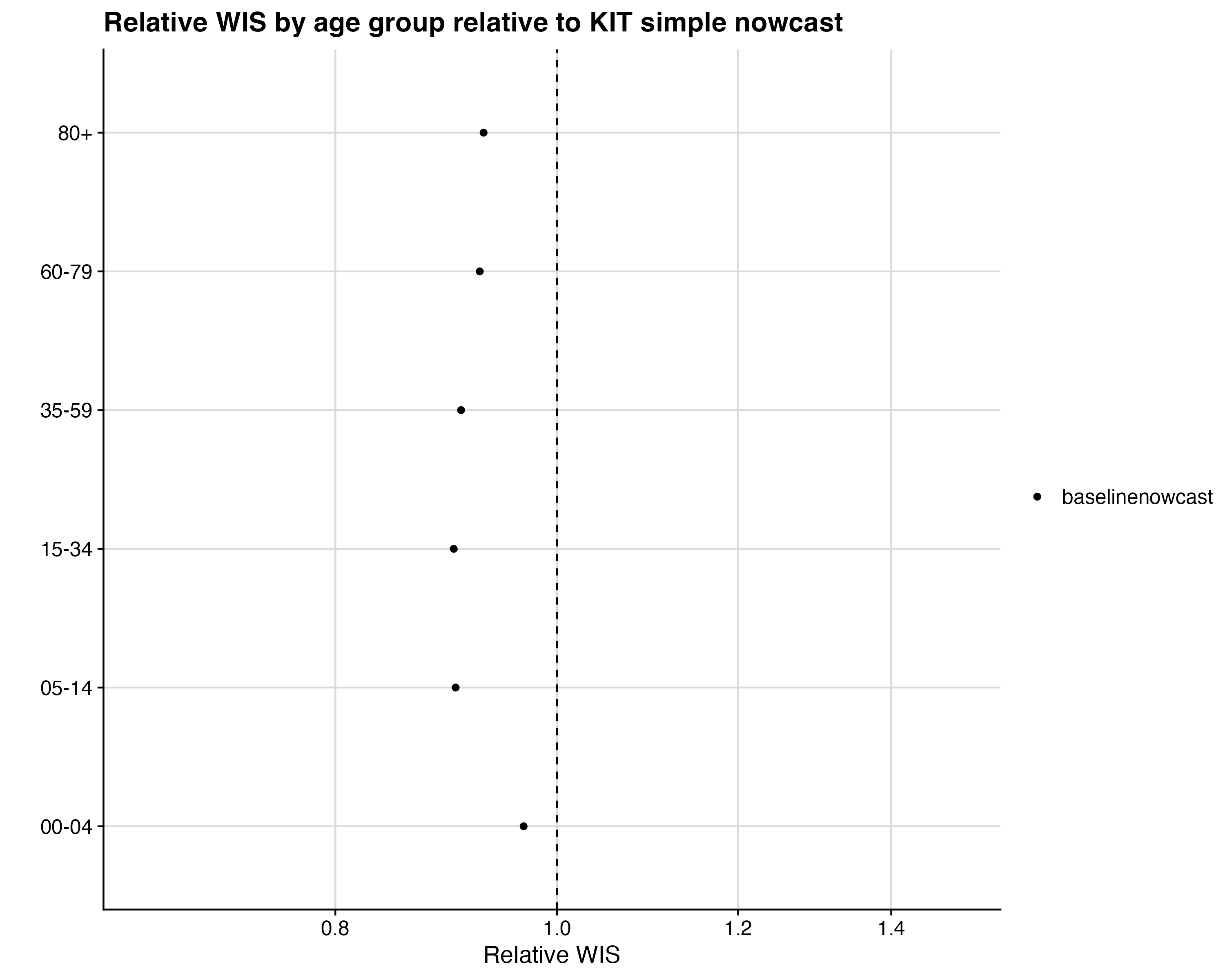


Fig.S2 Relative WIS by age group.

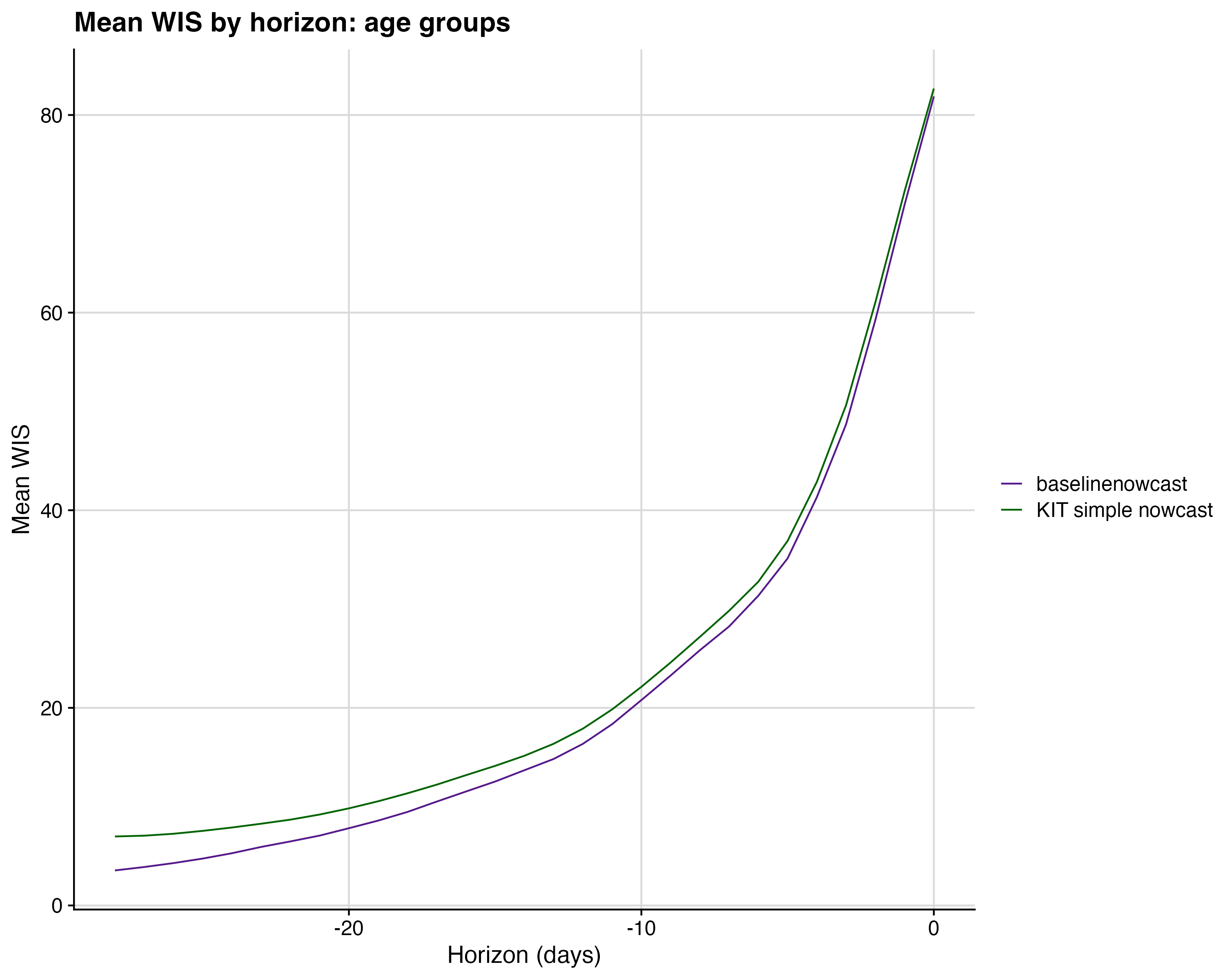


Fig.S3 Mean WIS by nowcast horizon for each model—same panel relative WIS by nowcast horizon.

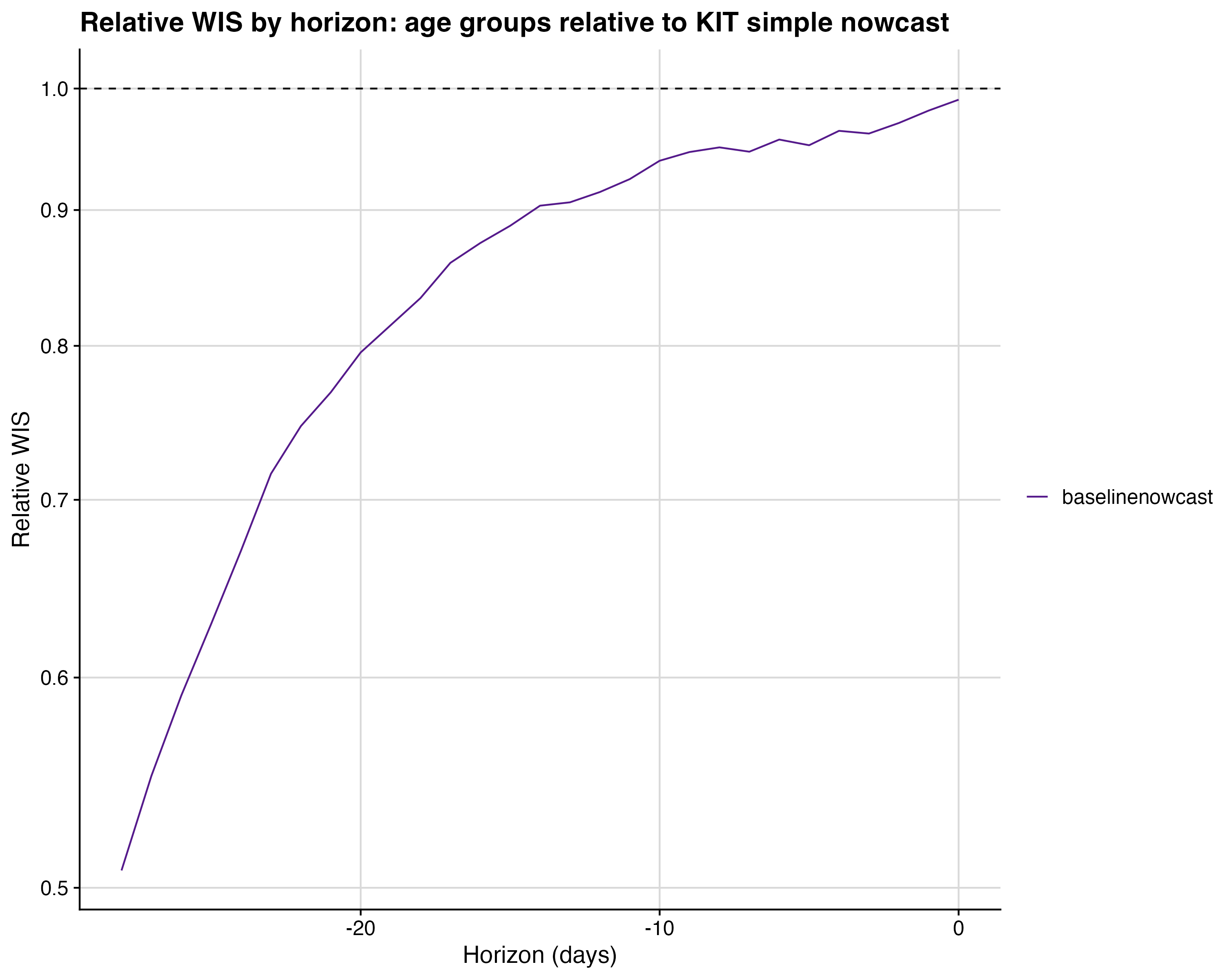


Fig.S4. Relative WIS by nowcast horizon

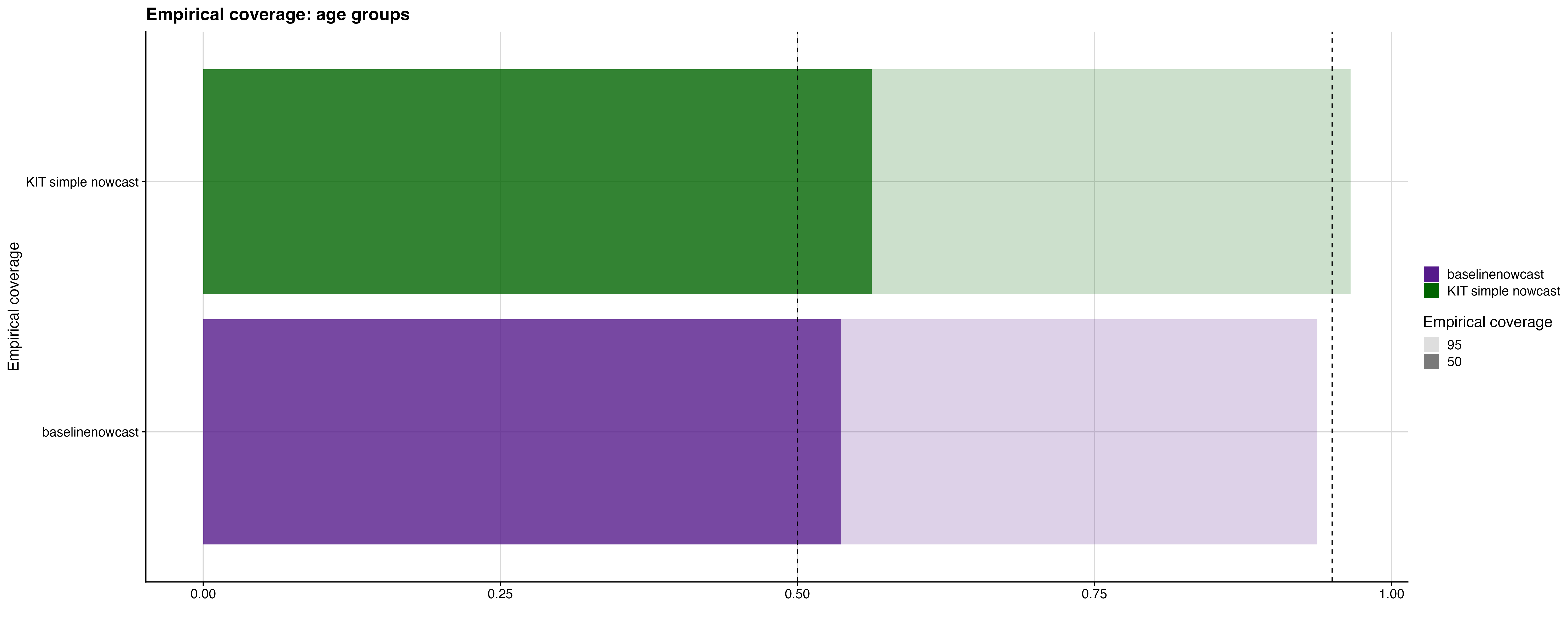


Fig.S5 Empirical coverage at 50% and 90% prediction intervals

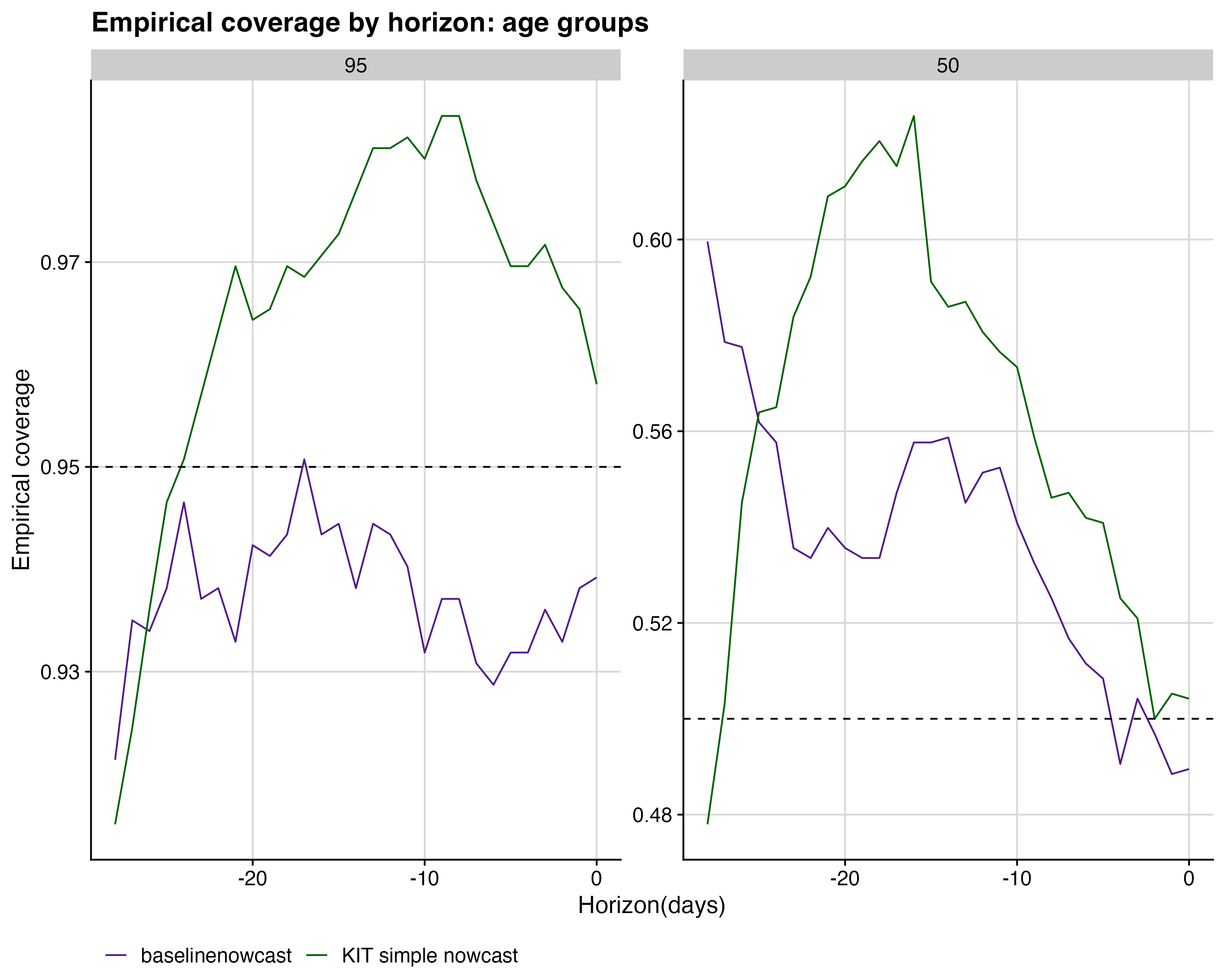


Fig.S6 Empirical coverage at 50% and 90% prediction intervals by horizon for each model.

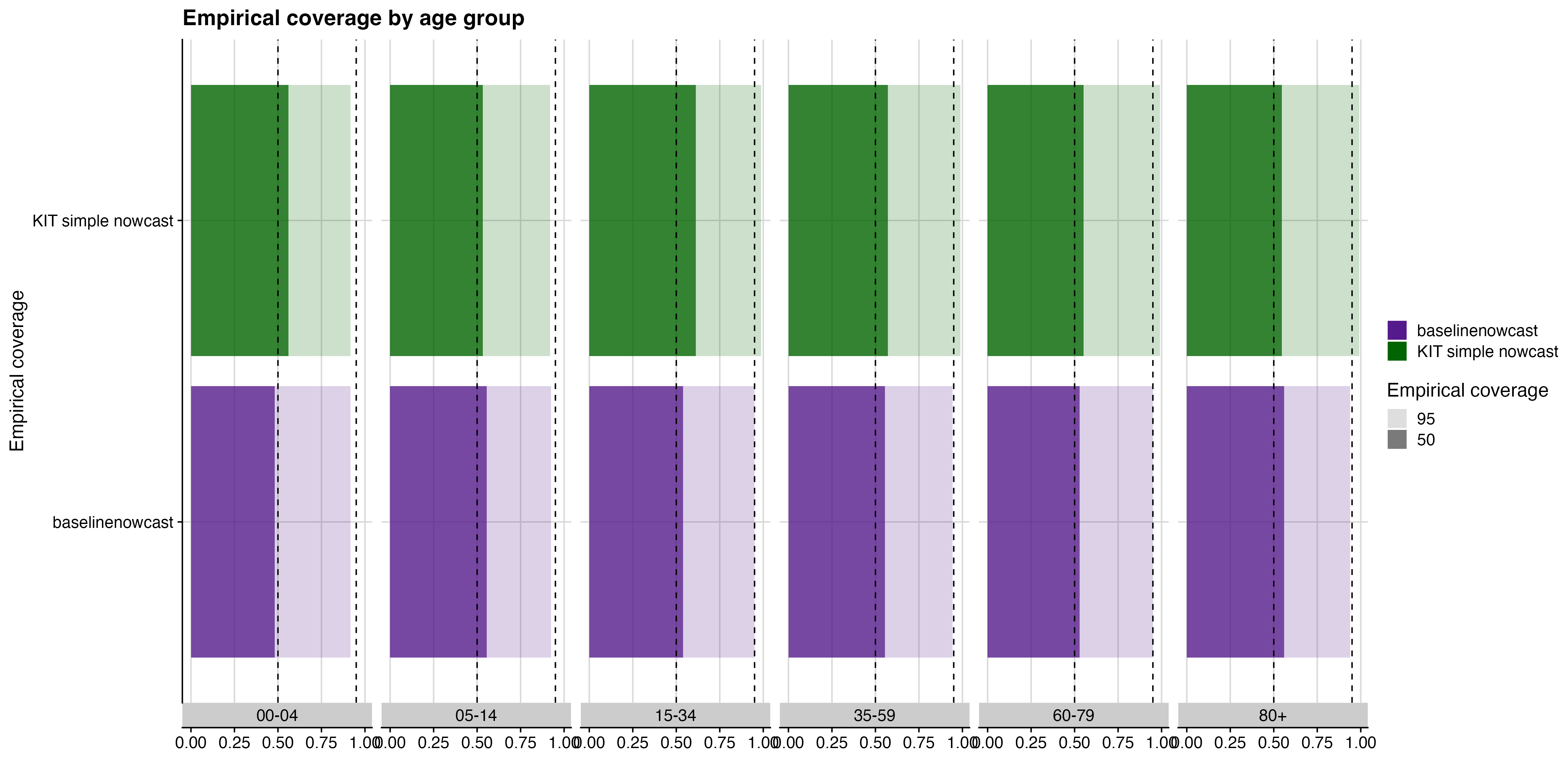


Fig S7. Empirical coverage at 50% and 90% prediction intervals by age group for each model configuration.

### Model permutation study

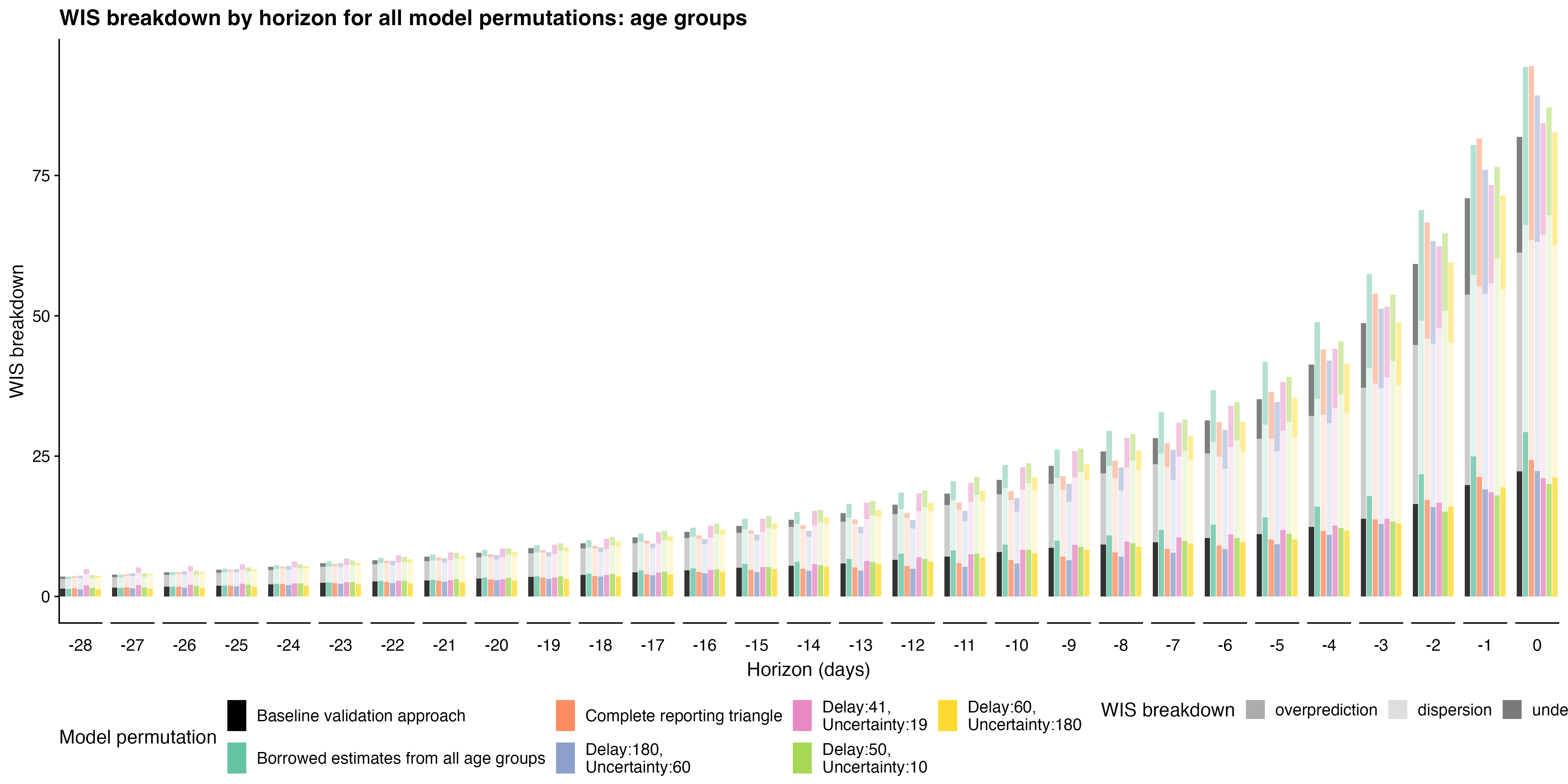


Fig S8. Absolute WIS by nowcast horizon for each model configuration. Colour by model configuration, decomposed by dispersion, overprediction, and underprediction.

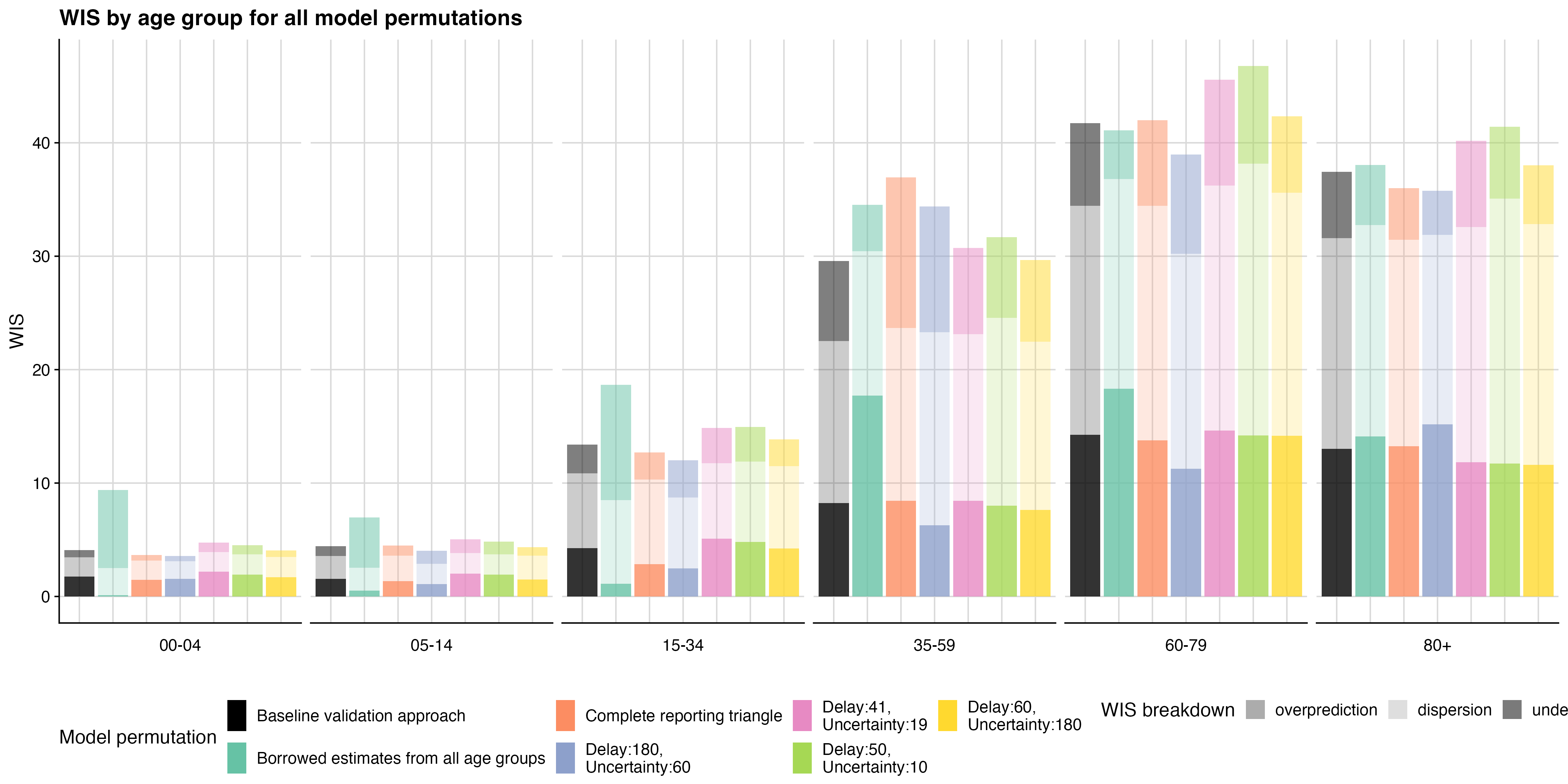


Fig S9. Absolute WIS by age group for each model configuration. Colour by model configuration, decomposed by dispersion, overprediction, and underprediction.

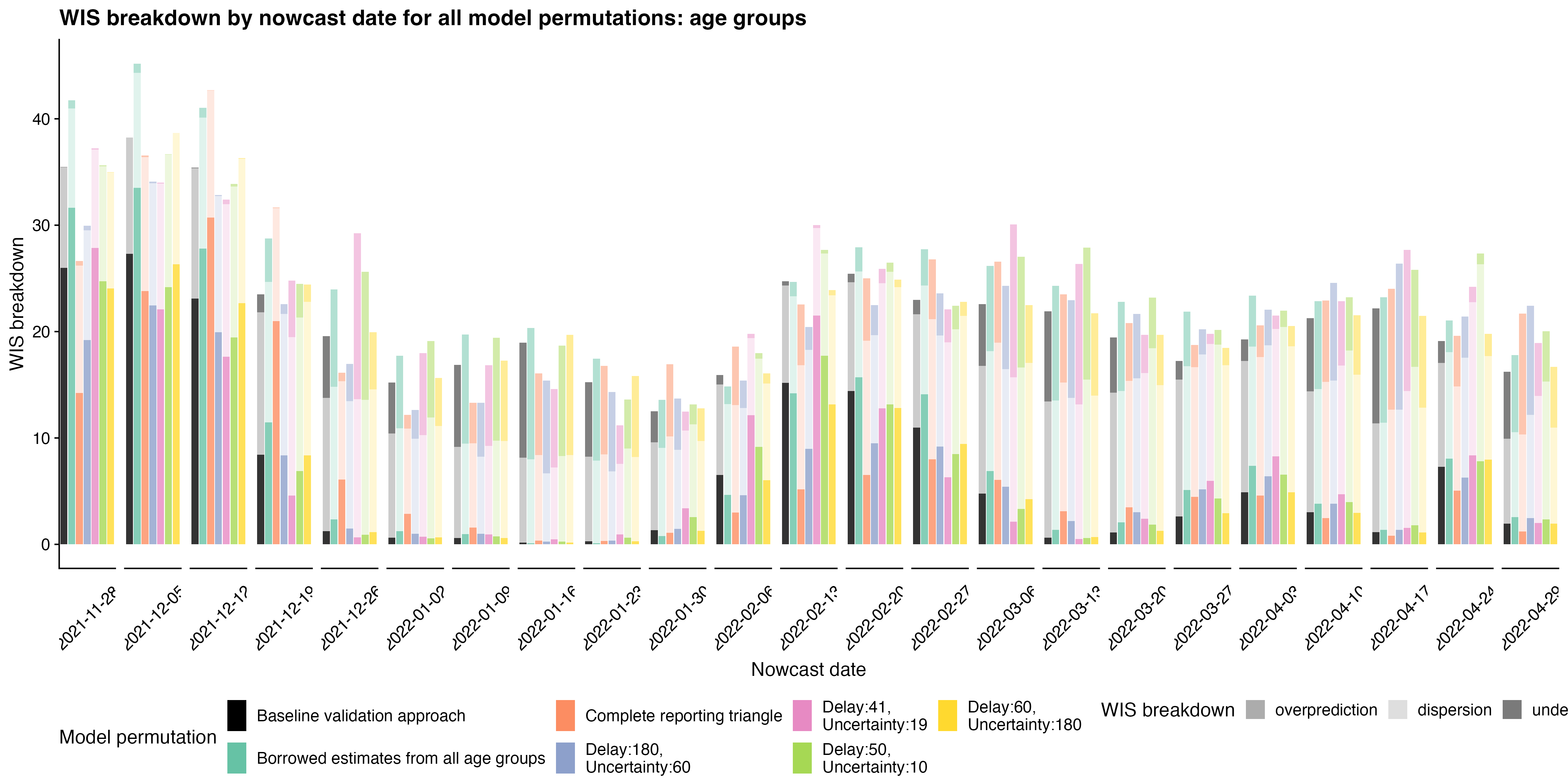


Fig S10. Absolute WIS over time (by week) for each model configuration. Colour by model configuration, decomposed by dispersion, overprediction, and underprediction.

### Case study: UKHSA norovirus surveillance

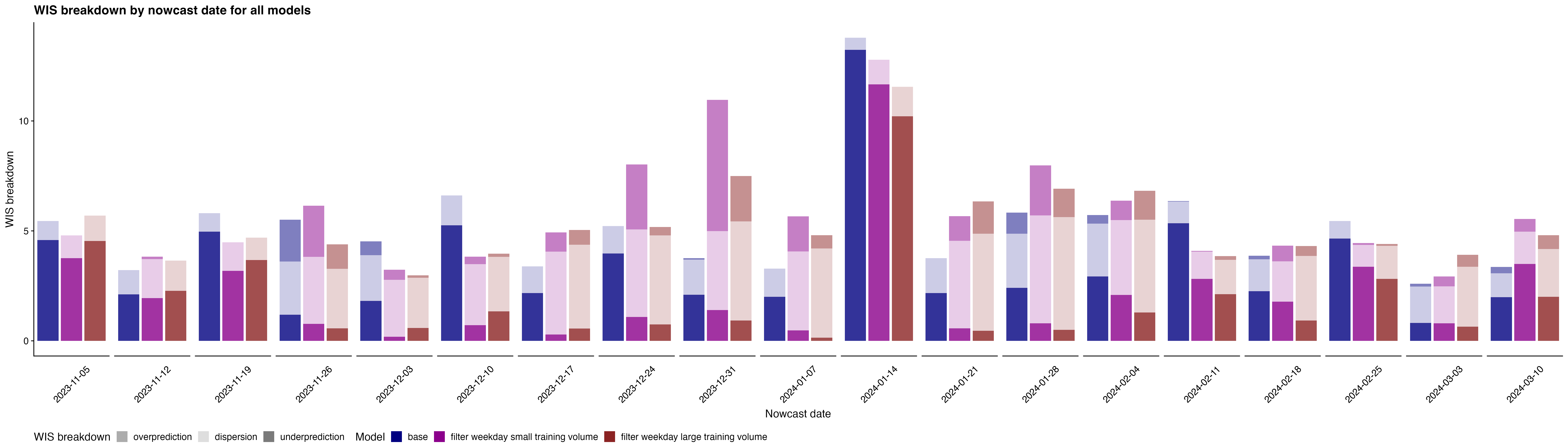


Fig S11. Absolute WIS over time (by week) for each model.

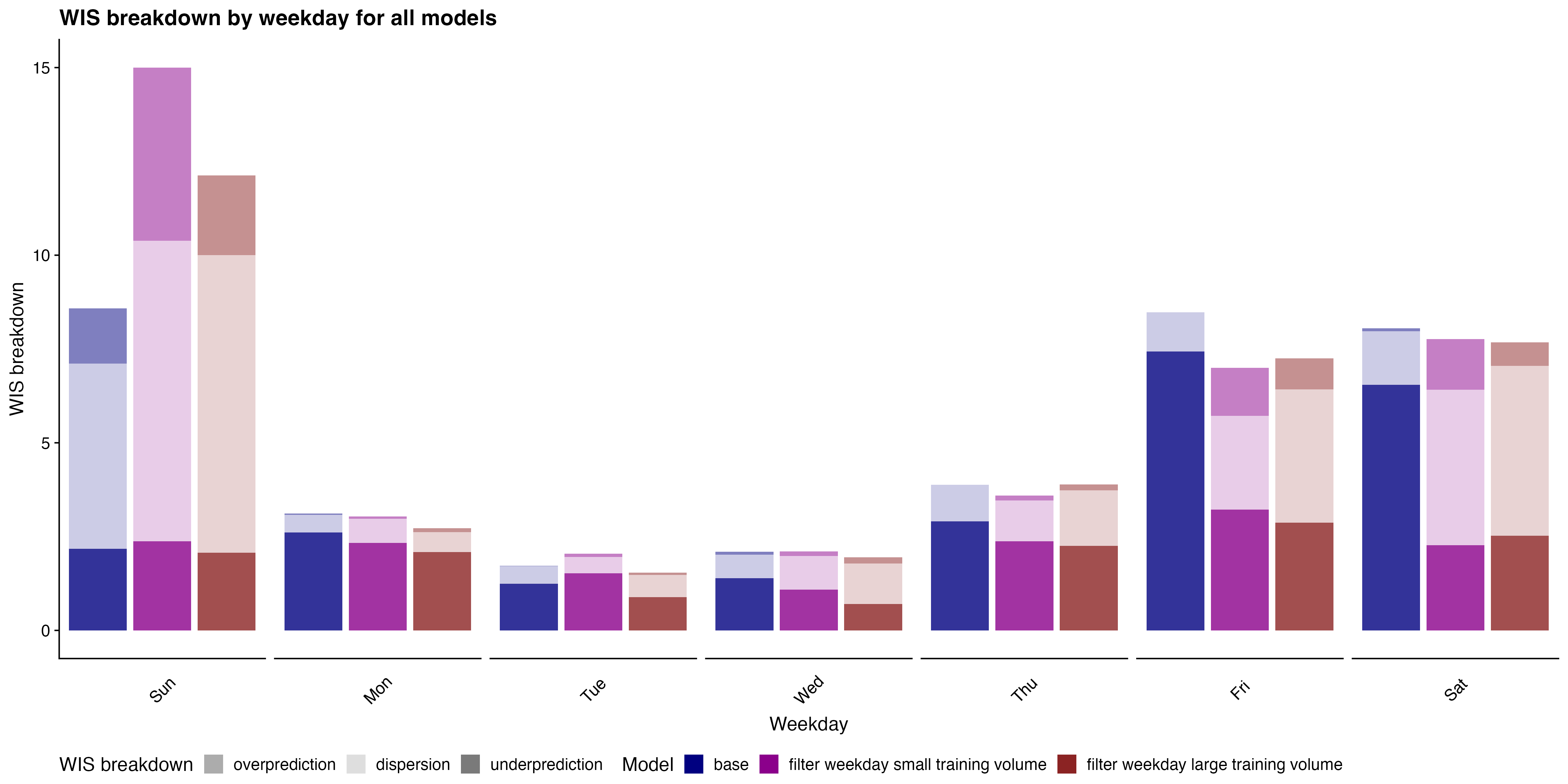


Fig S12. Absolute WIS by day of week for each model.

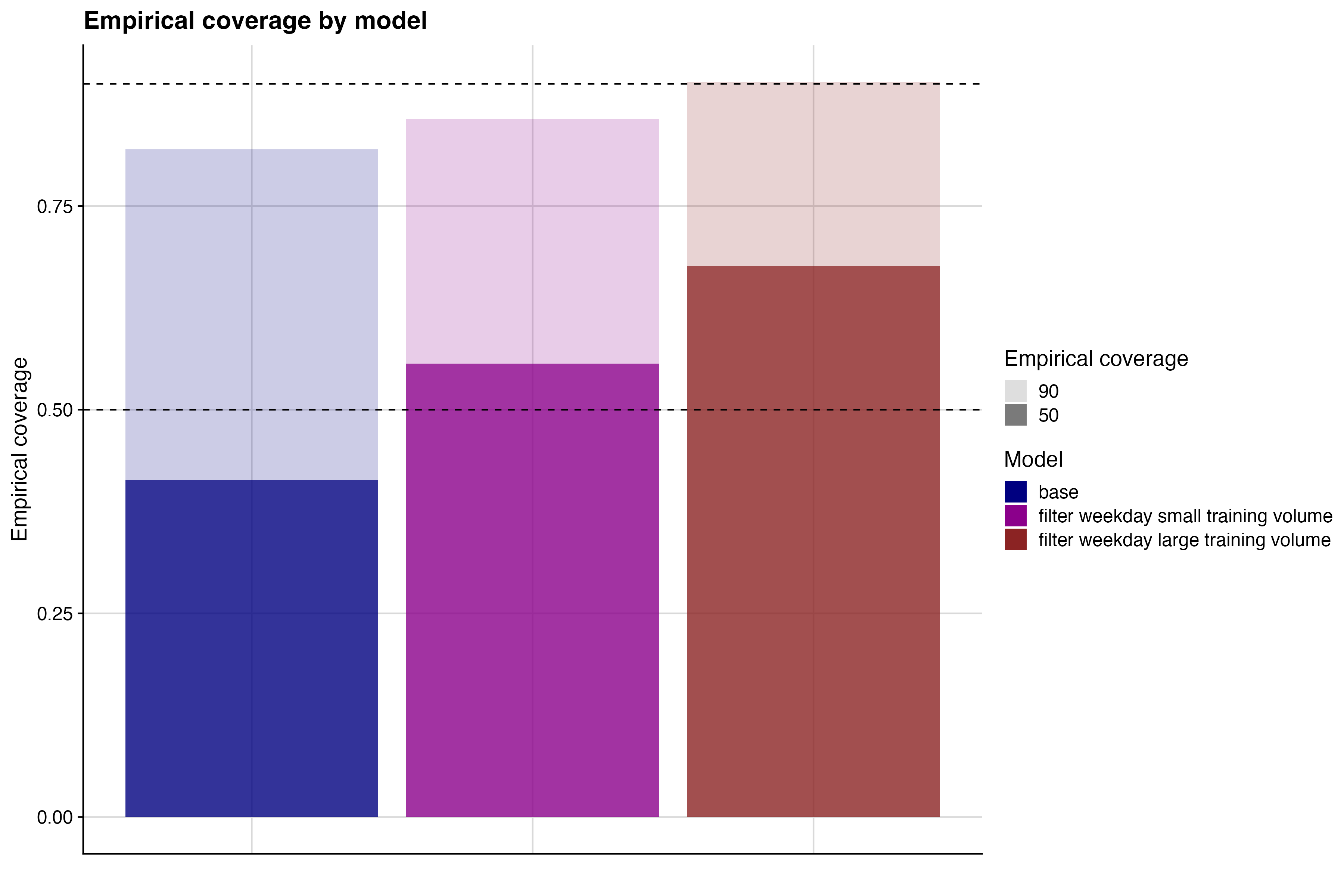


Fig S13. Empirical coverage at 50% and 90% prediction intervals for each model.

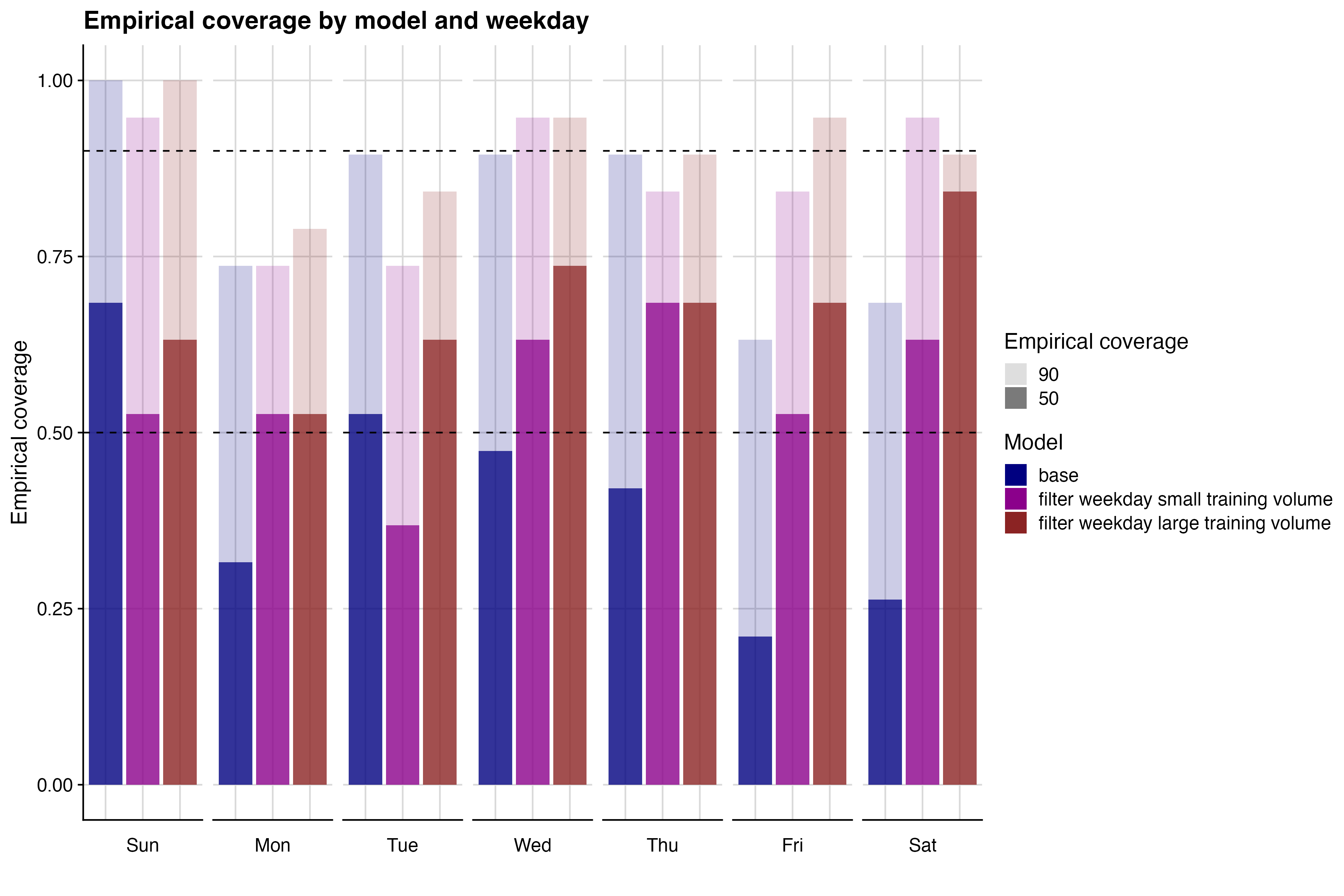


Fig S14. Empirical coverage at 50% and 90% prediction intervals by day of week for each model.

## References

KIT Metrics Lab, Johannes Bracher, Daniel Wolffram, and Melanie Schienle. 2025. “Hospitalization Nowcast Hub: Collecting Nowcasts of the 7-Day Hospitalization Incidence in Germany.” Karlsruhe Institute of Technology; GitHub. <https://github.com/KITmetricslab/hospitalization-nowcast-hub>.

KIT Metrics Lab, Johannes Bracher, and Melanie Schienle. 2025. “RESPINOW-Hub: Nowcasting and Short-term Forecasting Platform for Respiratory Diseases.” Karlsruhe Institute of Technology; GitHub. <https://github.com/KITmetricslab/RESPINOW-Hub>.

Wolffram, Daniel, Sam Abbott, Matthias an der Heiden, Sebastian Funk, Felix Günther, Davide Hailer, Stefan Heyder, et al. 2023. “Collaborative Nowcasting of COVID-19 Hospitalization Incidences in Germany.” *PLOS Computational Biology* 19 (8): 1–25. <https://doi.org/10.1371/journal.pcbi.1011394>.