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 hospitalisation 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:

A special case of this is the “final” number of reported cases at time , :

Conversely, for delays less than the maximum delay , we defined by

representing 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:

{#eq:pi\_d}

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 .

#### Multiplicative model

In the case where we have missing values in the bottom right (i.e. we have a reporting triangle), the estimator can only be evaluated as written in @ref(eq:pi\_d) after discarding all data from the last time points. In order to use the partial information contained in the most recent time points, we use a different representation of the delay distribution via terms of the form:

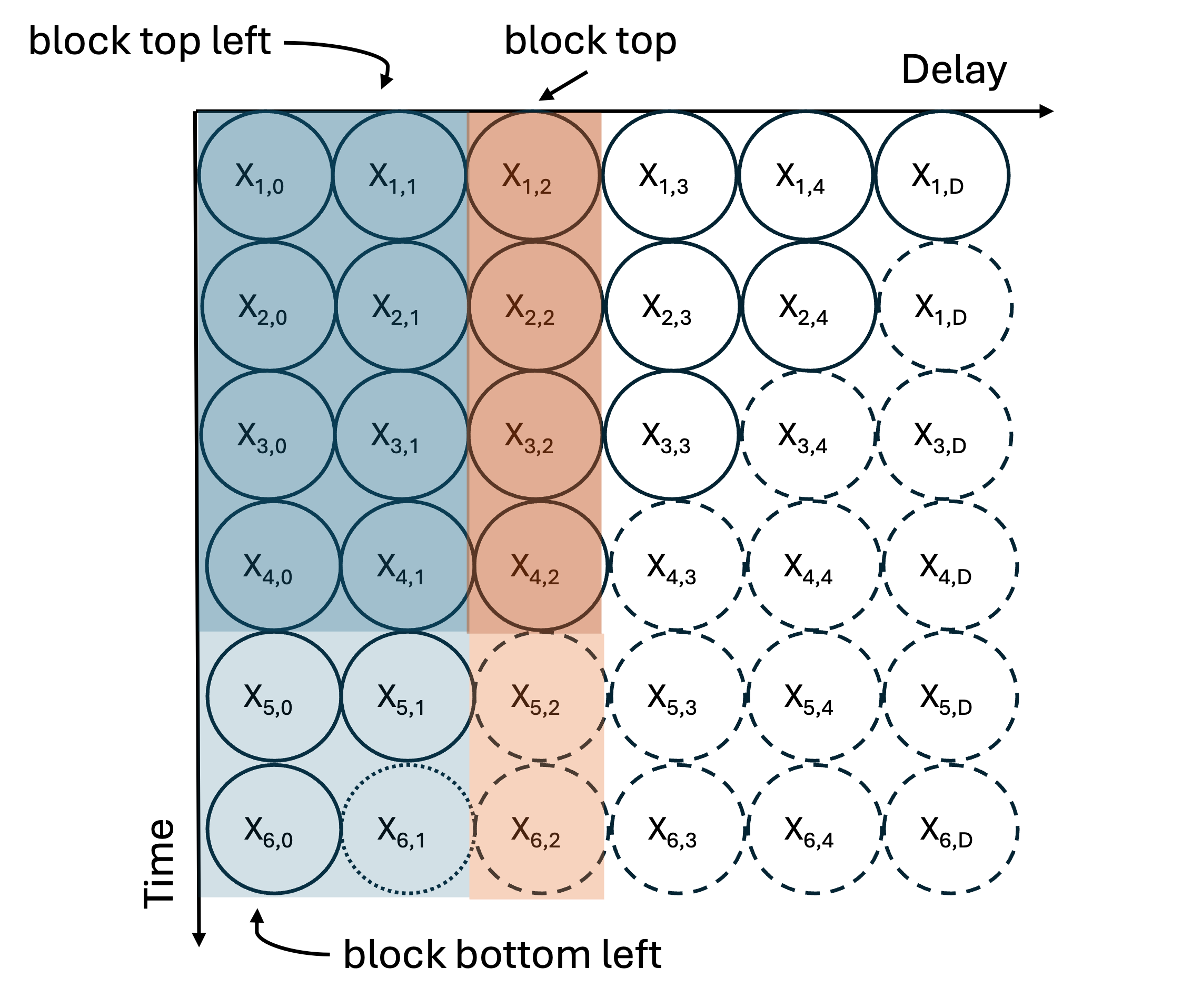
{#eq:theta\_d}

Where

{#eq:pi\_less\_than\_d}

These can be estimated via the equation above @ref(eq:theta\_d) and translated to via the recursion

Subject to the constrain that .



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 intuition behind the multiplicative method is the assumption 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 estimated from the filled in reporting matrix per @ref(eq:pi\_d). It is worth noting that the resulting imputed values in the bottom right of the reporting triangle do not have an effect on the delay distribution, as they are chosen such that they correspond almost precisely to the distribution seen in the combination of the complete and partial rows. The slight imprecision is due to the additional component incorporated to handle 0s in observations of . See [Zero-handling](#zero-handling-approximation) for further details.

### Point nowcast generation

To obtain a point nowcast matrix from a reporting triangle and any arbitrary delay PMF, , 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 . 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 the missing elements of the point nowcast matrix directly using the th element of

Where the expected count at timepoint with delay is the product of the the expected total count, and the proportion expected at that particular delay , . We will use the individual elements of the point nowcast matrix, for uncertainty quantification.

### Uncertainty quantification

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 times 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 current reporting triangle, to generate truncated reporting triangles. We then subsequently remove values which would not have been availbale as of each , to generate retrospective reporting triangles.

#### Generation of retrospective point nowcast matrices

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

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.

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

We quantify the uncertainty in the component of the target that corresponds to the counts that are still to be added. The default in the package assumes that the target is the final count at each reference time, summed across reporting delays, however, for the COVID-19 case study, the target is defined as the 7-day rolling sum of the counts at reference time, and therefore the uncertainty is quantified in the component of this quantity that has yet to be observed as of each reference time.

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, for the retrospective nowcast and the truncated observed component . We define the component of the retrospective nowcast that is predicted as of , but observed as of as:

and likewise the corresponding component that has been observed as of for each retrospective nowcast time as:

This generates pairs of predicted nowcast vectors and corresponding later observed nowcast vectors for each forecast horizon . We assume that the later observed nowcasts vectors 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 for the sum over the still missing counts:

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 . We will assume a simple binomial reporting model to derive an approximation for the expectation .

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

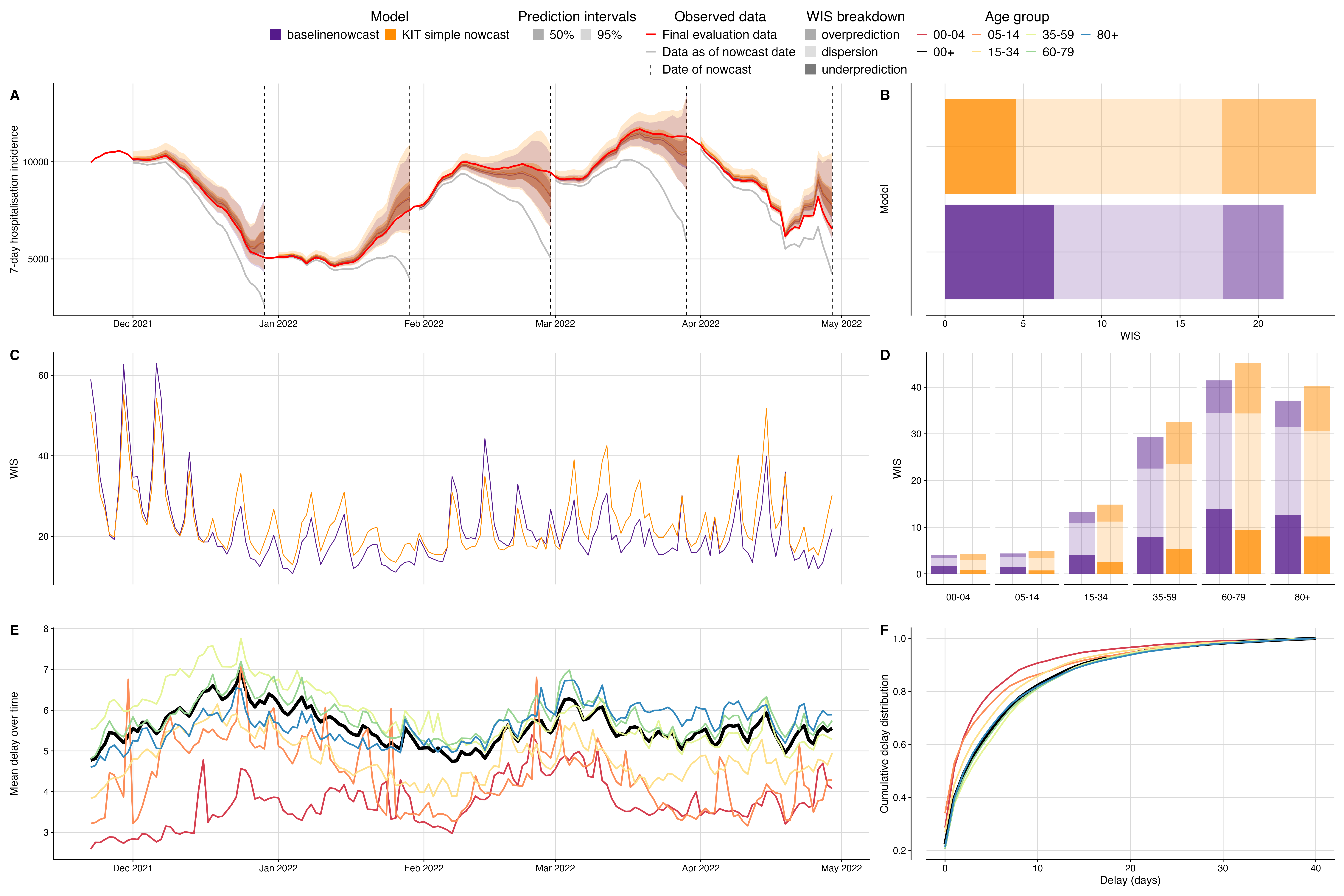


Fig. S1 Validation of baseline nowcasting model using German COVID-19 data compared to real-time KIT simple nowcast. A. Illustration of the nowcasting task from 5 nowcast dates throughout the study period. Nowcasts from our baseline model (purple) and the KIT simple nowcast model plotted against eventually observed values aggregated across all age groups, coloured by model. Observed data are shown as solid lines with gray indicating data available at the nowcast date and red indicating the final evaluation data. Dashed vertical lines indicate the date of the nowcasts. Shading indicates prediction intervals of each nowcast. Nowcasts are shown from horizons 0 to -28 days. B. Overall performance comparison between models aggregated across performance in individual age strata, with decomposed WIS (dispersion, overprediction, underprediction) displayed as stacked bar charts with different shading for direct comparison. 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 shaded 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. Mean reporting delays were calculated using the last 40 reference dates, corresponding to the maximum delay, for each nowcast date and age group. F: Delay distribution presented as cumulative distribution functions (CDFs), with overlaid curves for each age group and the national average highlighted as a thicker black line.

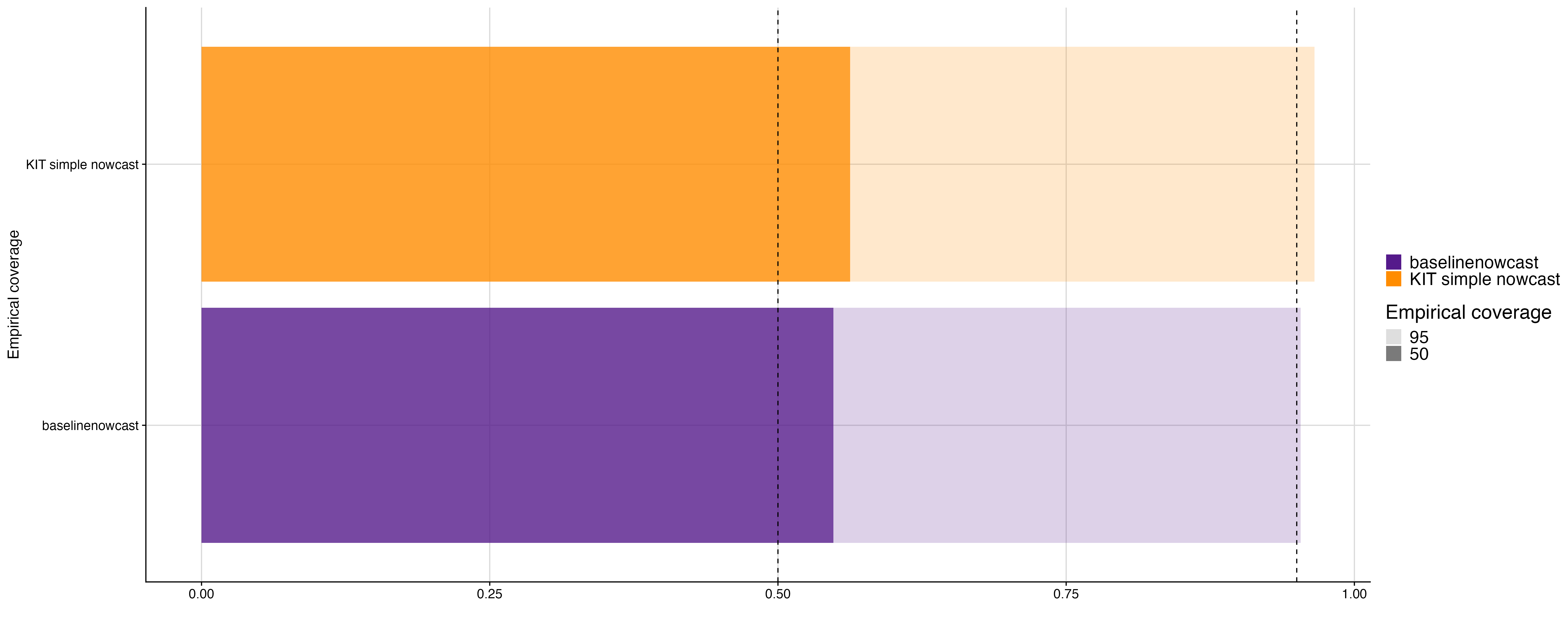


Fig. S2 Empirical coverage of baselinenowcast method and real-time KIT simple nowcast at 50% and 95% prediction intervals. Shading indicates prediction intervals, colors indicate model.

### German Nowcast Hub validation additional figures

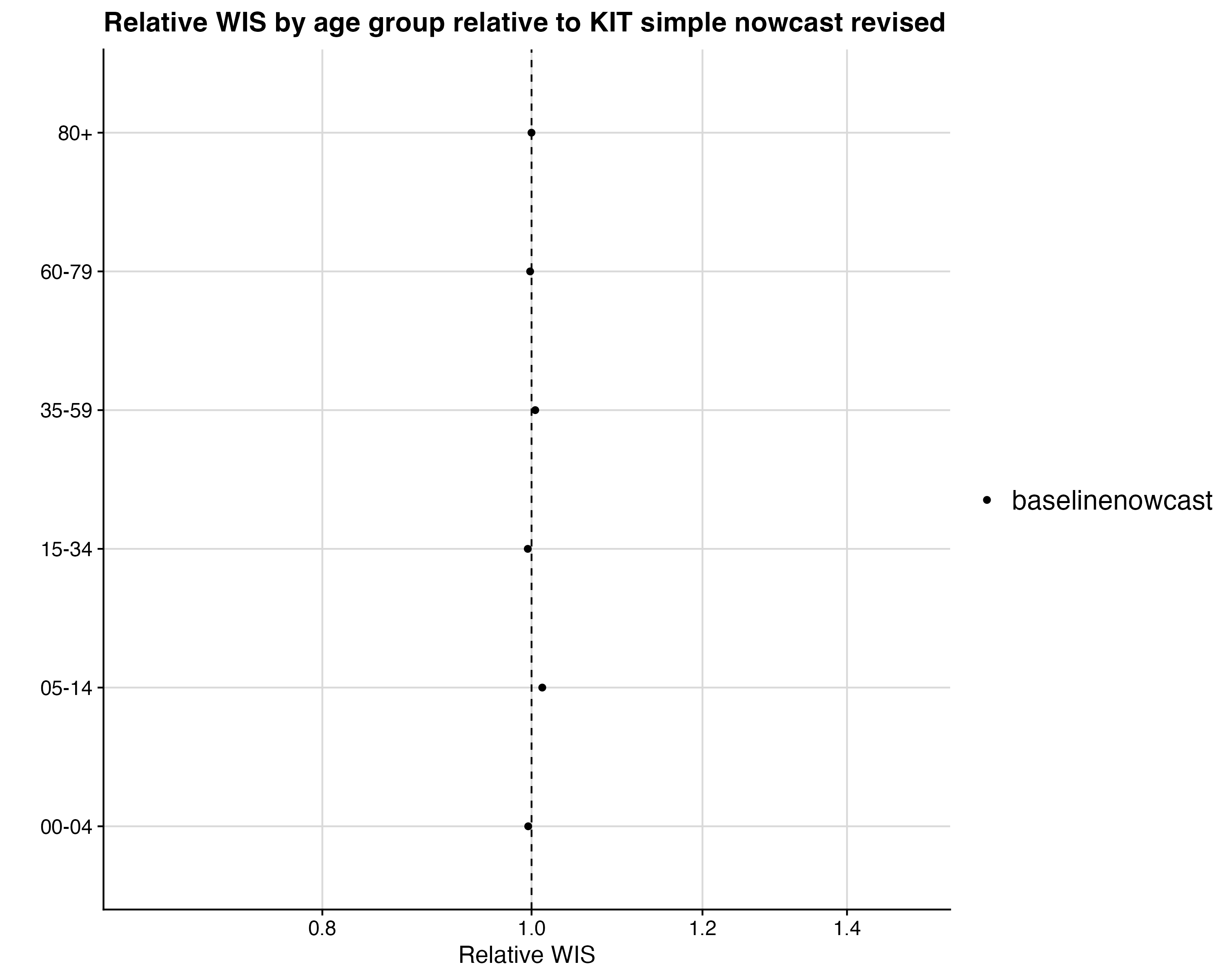


Fig. S3 Relative WIS by age group of baselinenowcast compared to KIT simple nowcast revised. Vertical line at 1.0 indicates parity with KIT simple nowcast revised.

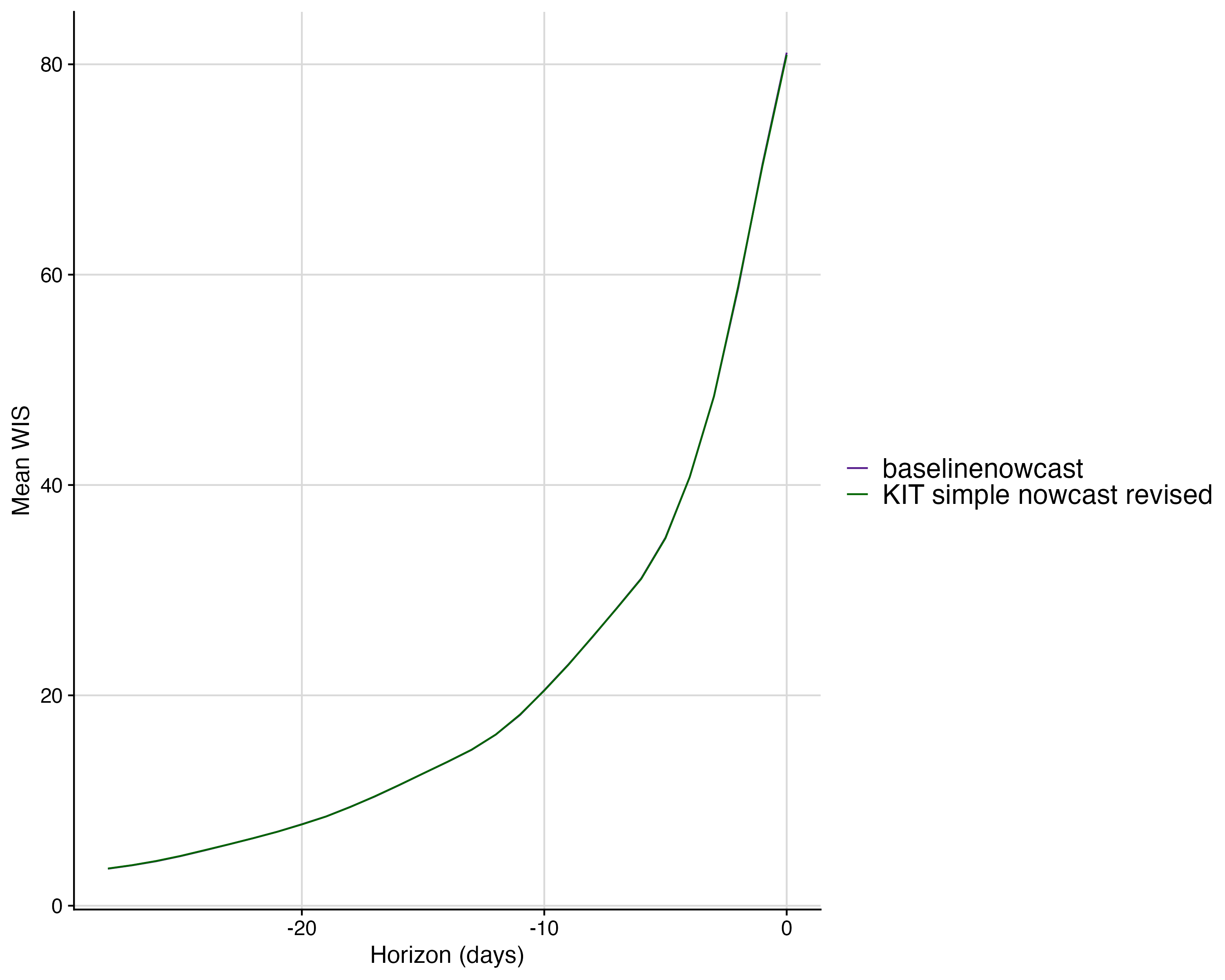


Fig. S4 Mean WIS by nowcast horizon for each model. Colors indicate models.

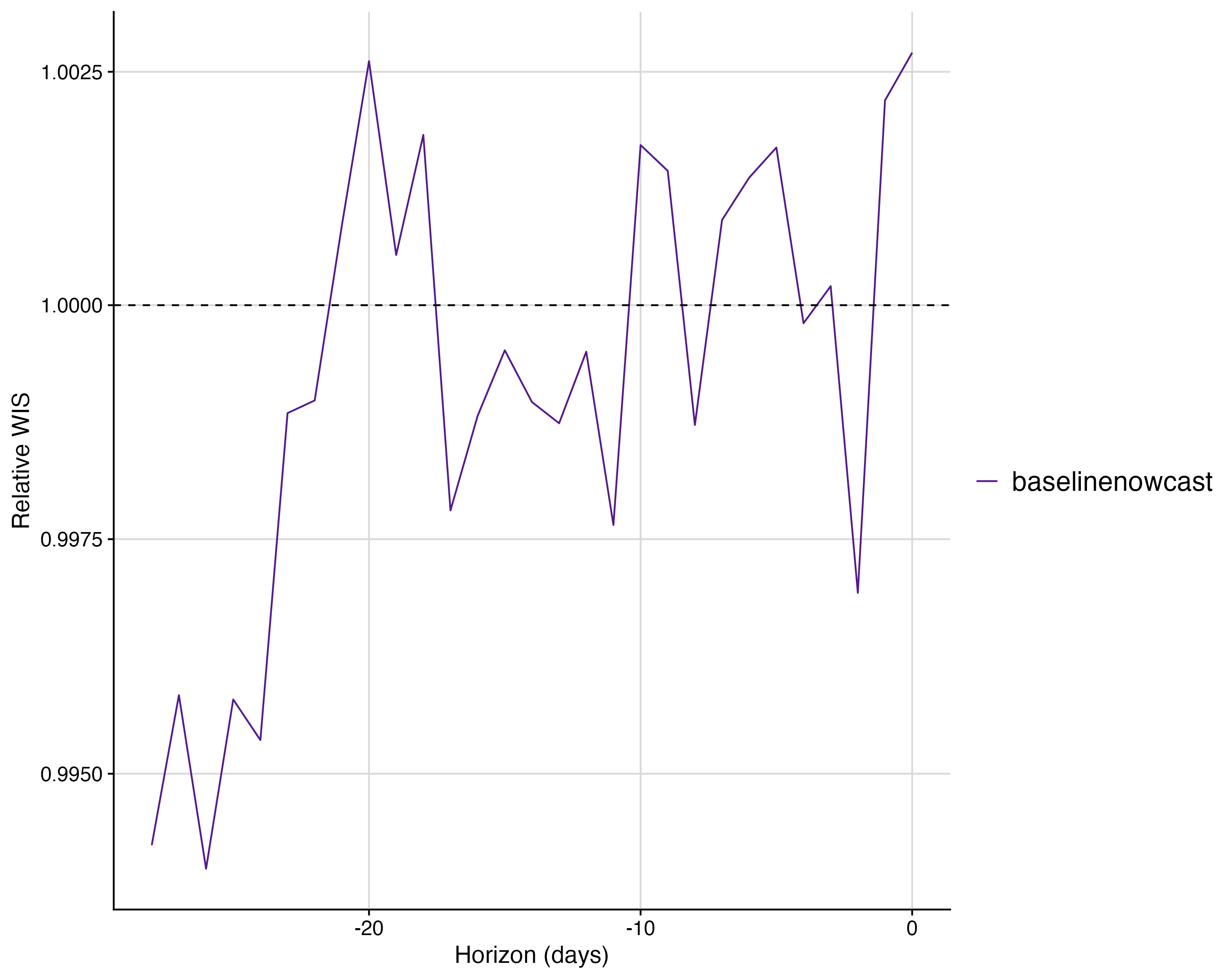


Fig. S5 Relative WIS of baselinenowcast horizon compared to KIT simple nowcast revised.

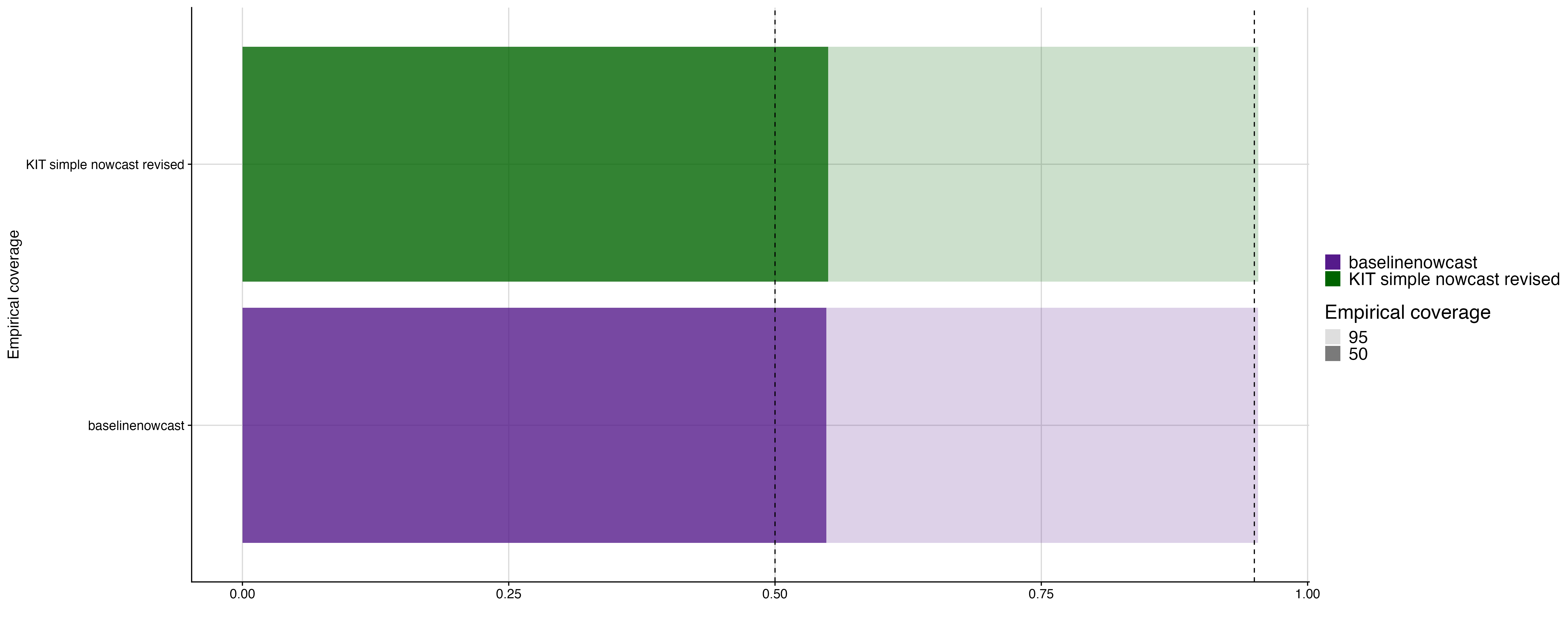


Fig. S6 Empirical coverage at 50% and 95% prediction intervals. Shading indicates prediction intervals, color indicates models, vertical lines indicate the target prediction intervals.

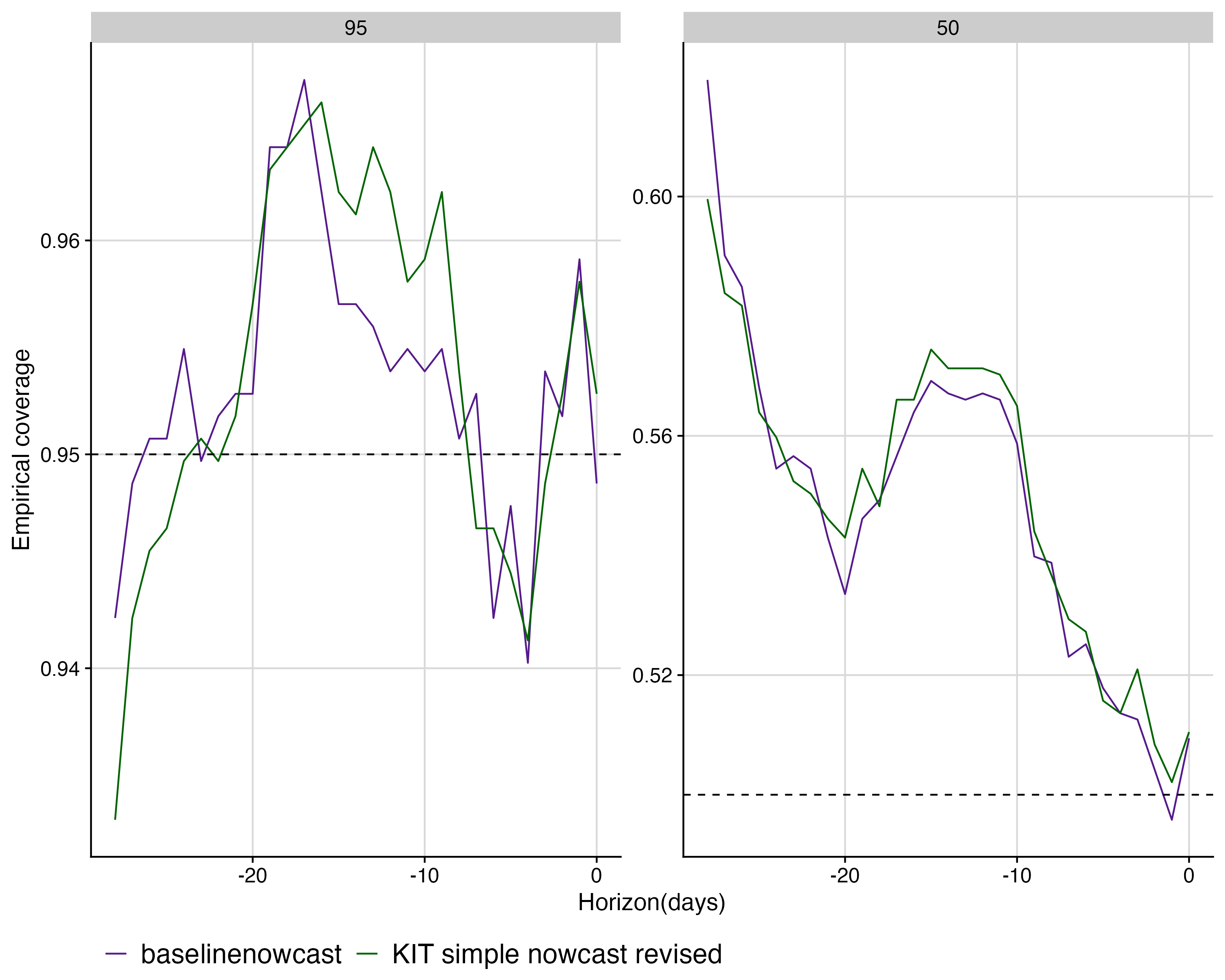


Fig.S7 Empirical coverage at 50% and 95% prediction intervals by horizon for each model. Colors indicate models, horizontal lines indicate the target prediction intervals in each subplot.

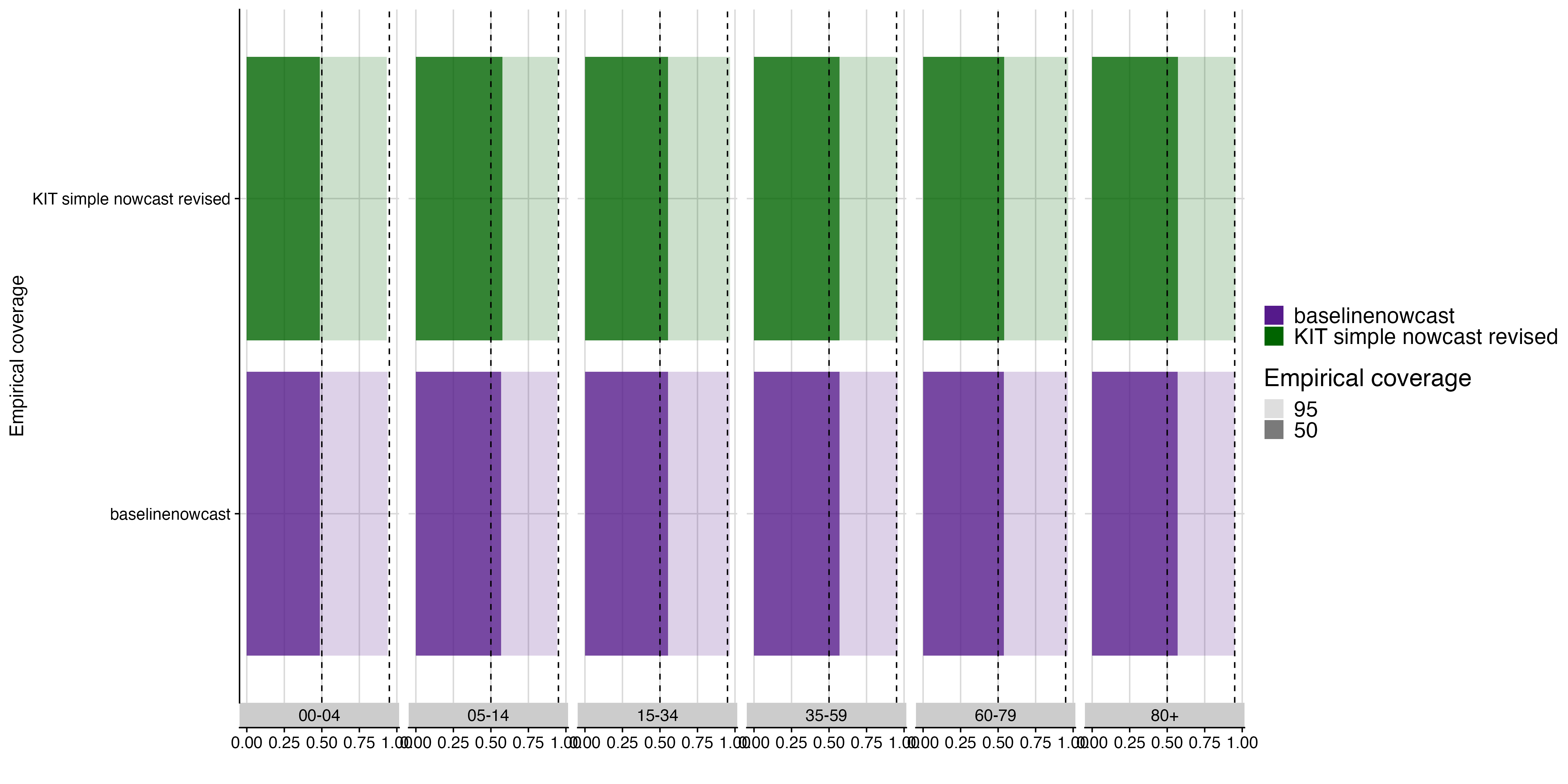


Fig S8. Empirical coverage at 50% and 95% prediction intervals by age group for each model. Shading indicates prediction intervals, color indicates models, vertical lines indicate the target prediction intervals.

### Model permutation study

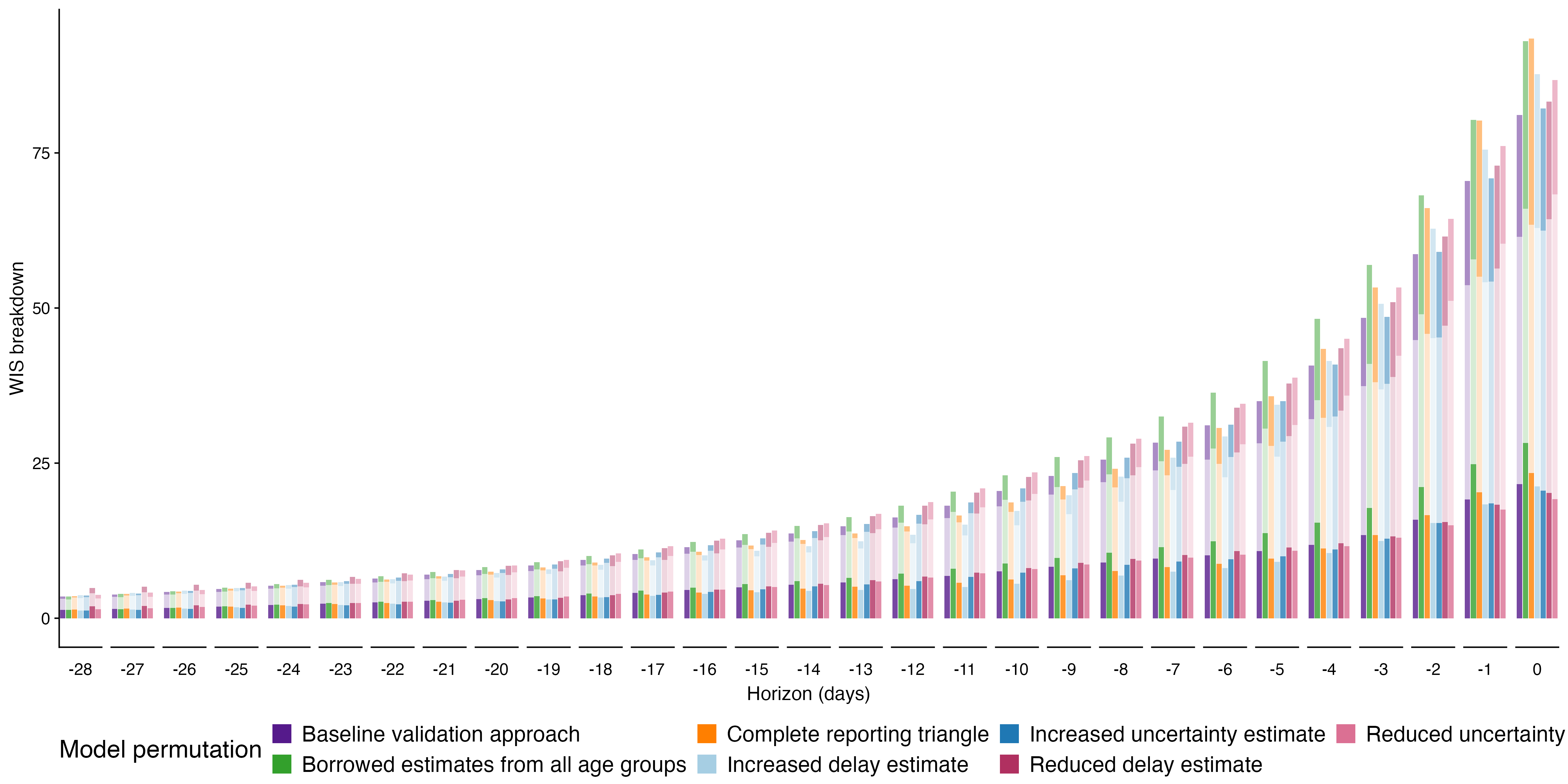


Fig S9. Absolute WIS by nowcast horizon for each model configuration. Colour indicates model configuration, shading indicates WIS decomposed by dispersion, overprediction, and underprediction.

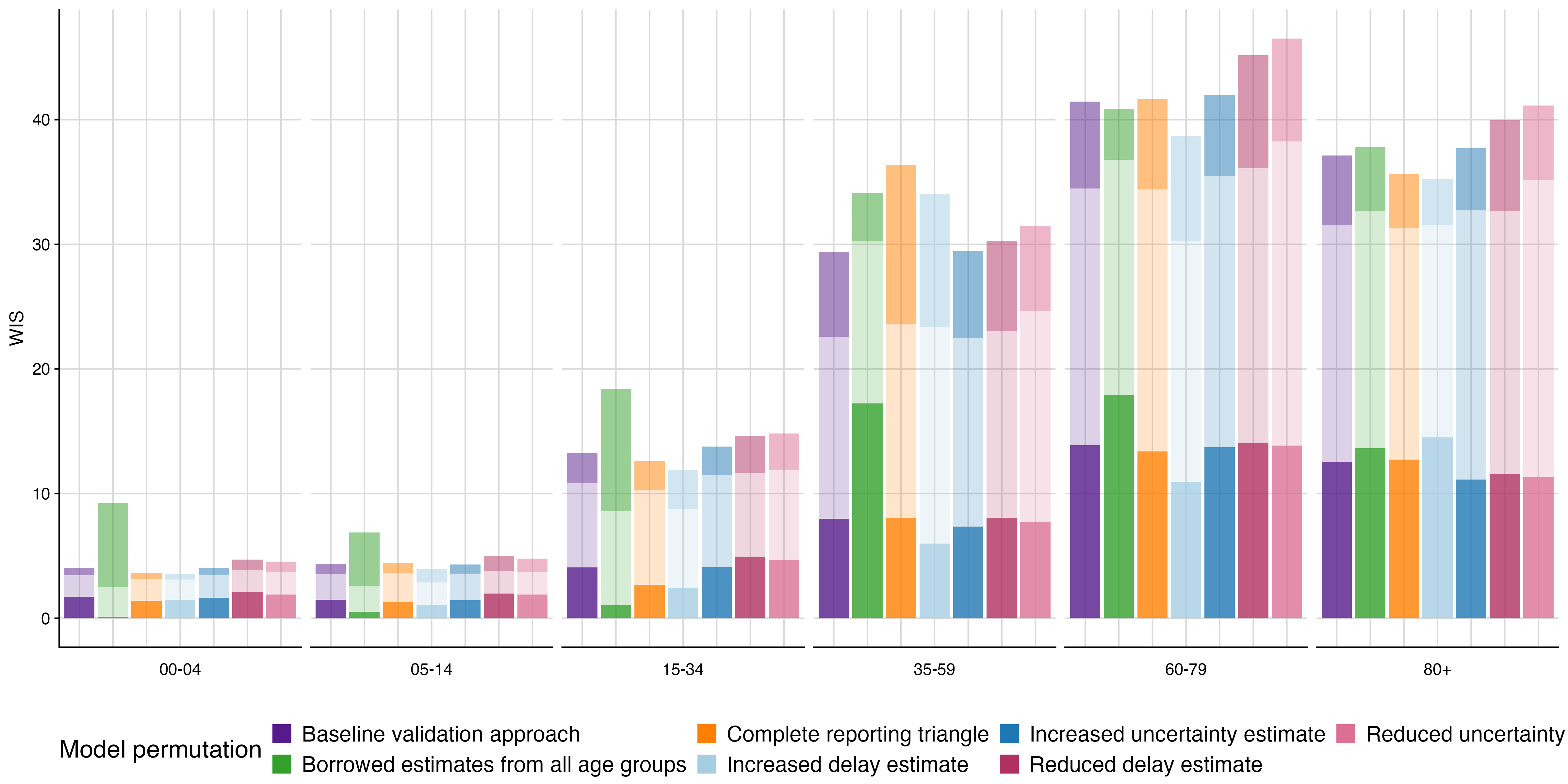


Fig S10. Absolute WIS by age group for each model configuration. Colour indicates model configuration, shading indicates WIS decomposed by dispersion, overprediction, and underprediction.

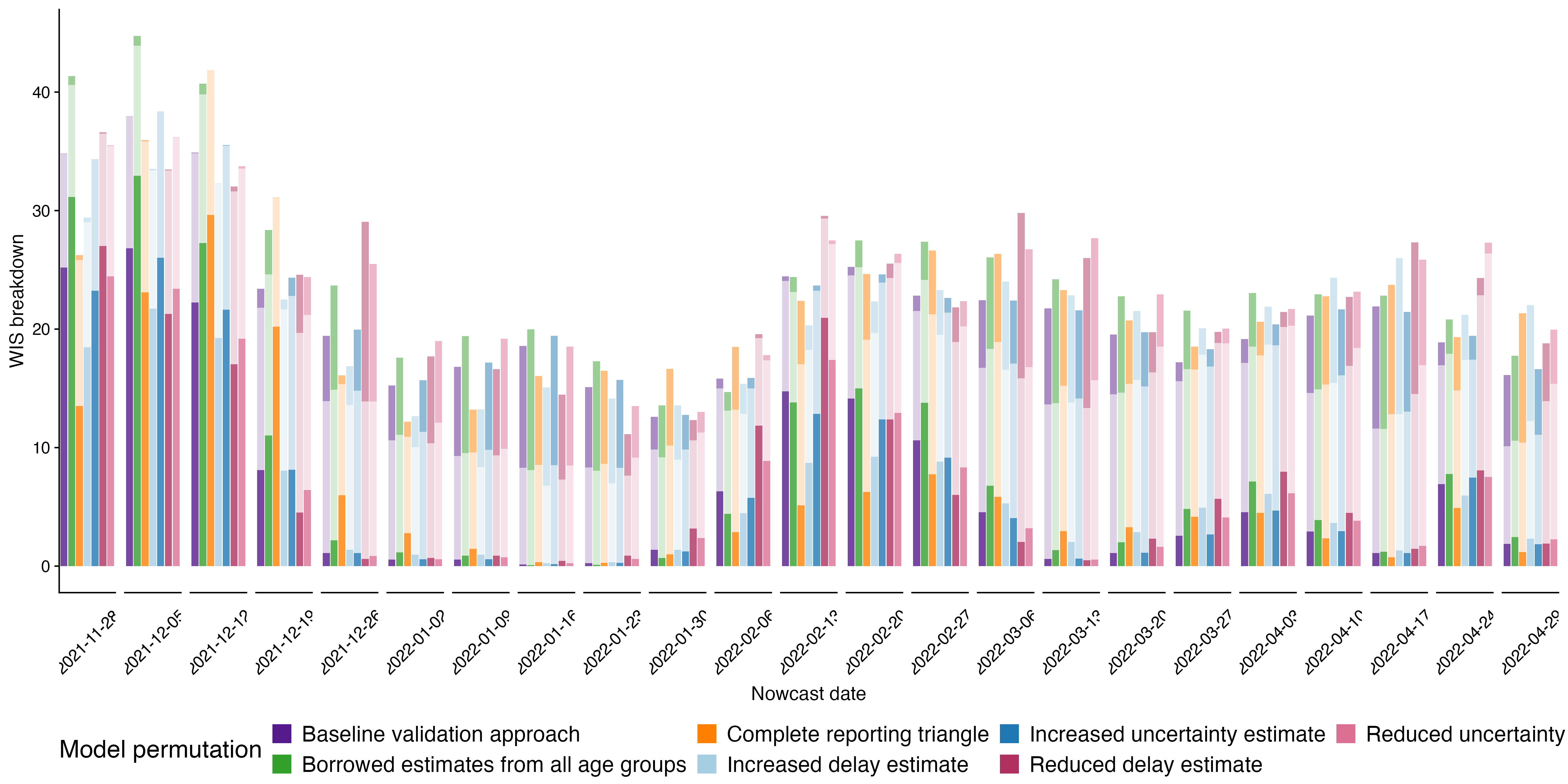


Fig S11. Absolute WIS over time (by week) for each model configuration. Colour indicates model configuration, shading indicates WIS decomposed by dispersion, overprediction, and underprediction.



Fig S12. Relative WIS over time across all age groups for each model configuration. Colours indicate model configuration. Horizontal dashed line indicates parity with the default configuration.

### UKHSA norovirus surveillance model comparison

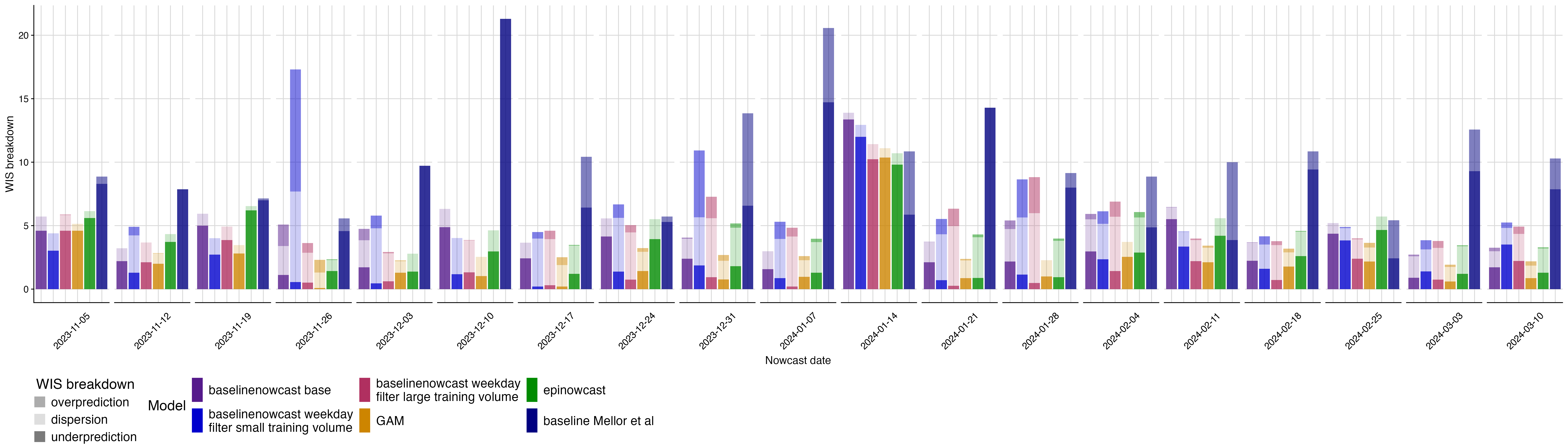


Fig S13. Absolute WIS over time (by week) for each model. Color indicates model, shading indicates WIS broken down by dispersion, overprediction, and underprediction.

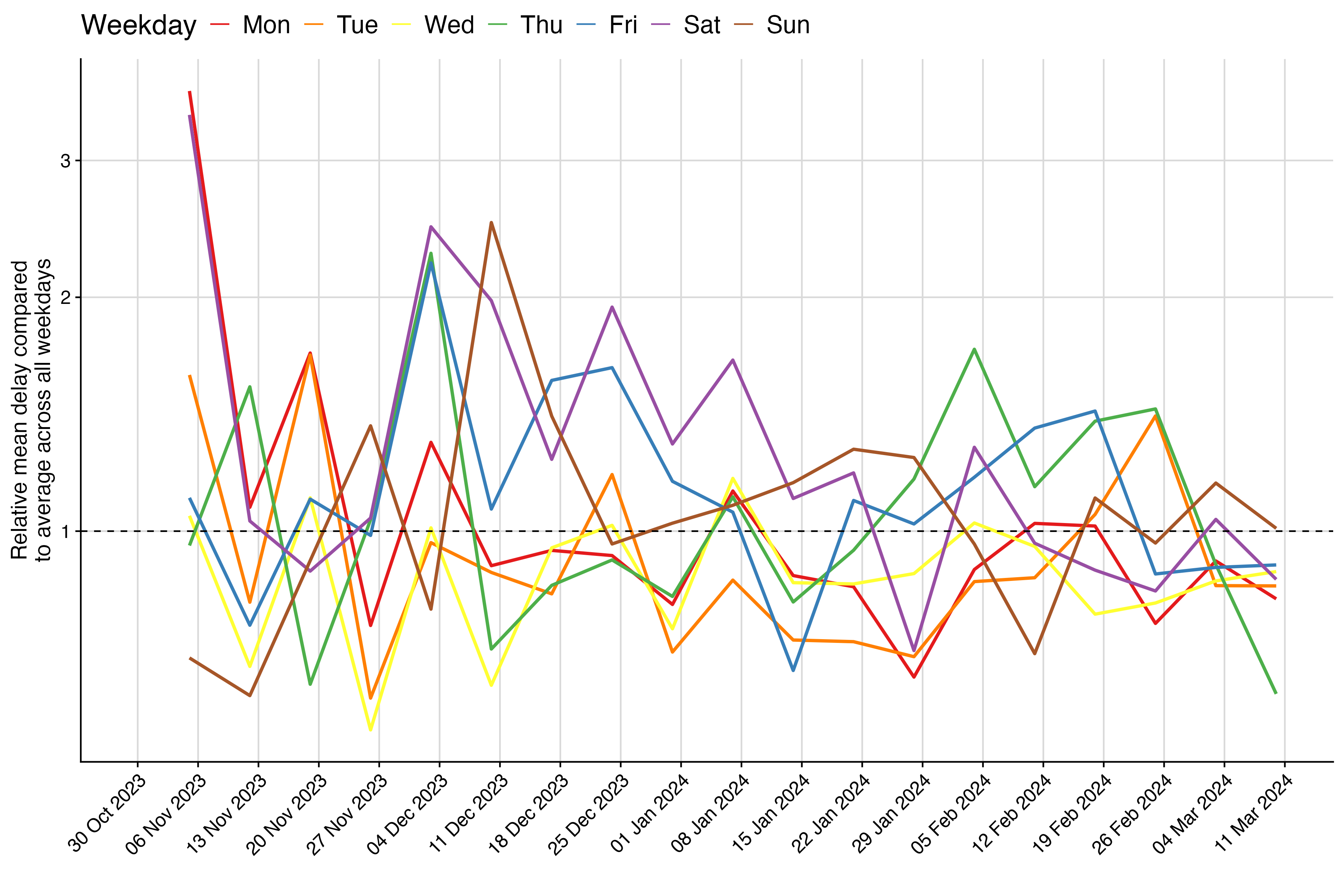


Fig S14. Relative mean delay over time by weekday. Colors indicate weekday. Horizontal dashed line at 1 indicates parity with the delay across all weekdays.

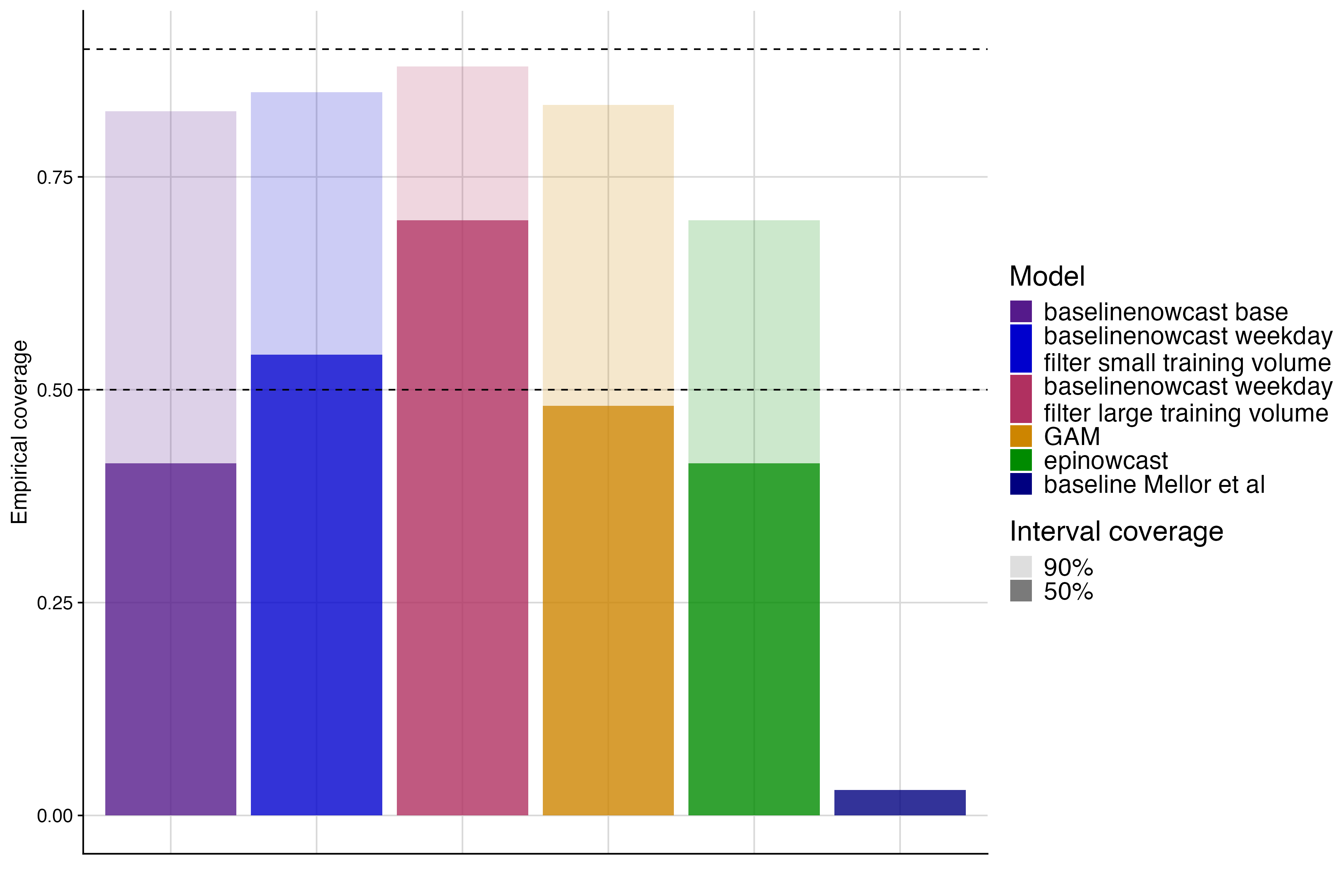


Fig S15. Empirical coverage at 50% and 90% prediction intervals for each model. Color indicates model, shading indicates prediction interval coverage, horizontal dashed lines indicate target prediction intervals.

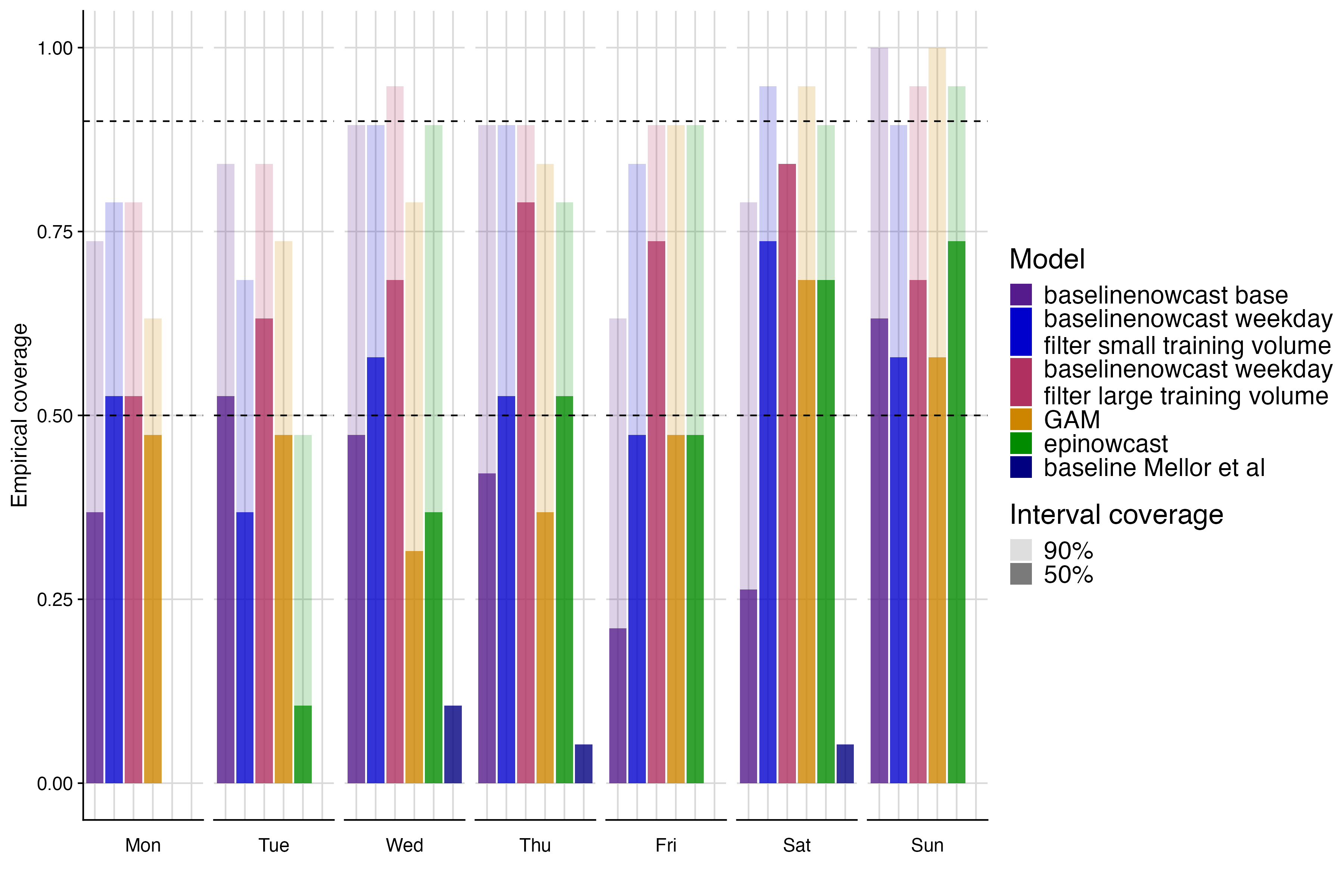


Fig S16. Empirical coverage at 50% and 90% prediction intervals by day of week for each model. Colors indicate model, shading indicates prediction interval coverage, horizontal dashed lines indicate target prediction intervals.

## 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>.