Ensemble Fractions Skill Score with harp

Andrew Singleton [MET Norway]

harp

A set of R packages

A set of R packages

- harpCore :: classes, methods, general functionalities

A set of R packages

- harpCore :: classes, methods, general functionalities
- harpIO :: read and write data

A set of R packages

- **harpCore** :: classes, methods, general functionalities
- **harpIO** :: read and write data
- **harpPoint** :: point verification

A set of R packages

- **harpCore** :: classes, methods, general functionalities
- **harpIO** :: read and write data
- **harpPoint** :: point verification
- **harpSpatial** :: spatial verification

A set of R packages

- **harpCore** :: classes, methods, general functionalities
- **harpIO** :: read and write data
- **harpPoint** :: point verification
- **harpSpatial** :: spatial verification
- **harpVis** :: visualisation

A set of R packages

- harpCore :: classes, methods, general functionalities
- harpIO :: read and write data
- harpPoint :: point verification
- harpSpatial :: spatial verification
- harpVis :: visualisation

A set of R packages

- harpCore :: classes, methods, general functionalities
- harpIO :: read and write data
- ~~harpPoint :: point verification~~
- ~~harpSpatial :: spatial verification~~
- harpVis :: visualisation

A set of R packages

- harpCore :: classes, methods, general functionalities
- harpIO :: read and write data
- harpVerif :: verification
- harpVis :: visualisation

Fractions Skill Score

Fractions Skill Score

Fractions Skill Score

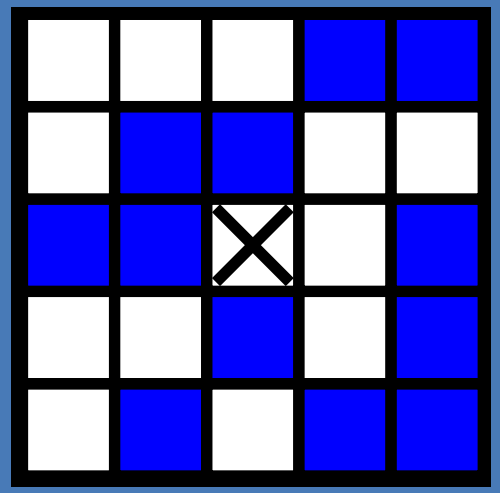
Compares forecast fraction with observed fraction for a neighbourhood

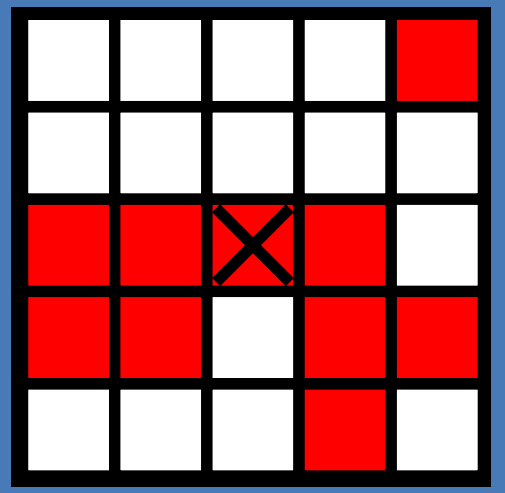
Fractions Skill Score

Compares forecast fraction with observed fraction for a neighbourhood

Fractions Skill Score

Compares forecast fraction with observed fraction for a neighbourhood





Fractions Skill Score

For a single forecast with neighbourhood size n

$$FSS_{(n)} = 1 - \frac{FBS_{(n)}}{FBS_{(n)_{ref}}}$$

Fractions Skill Score

Fractions Brier Score:

$$FBS_{(n)} = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left(F_{(n)_{i,j}} - O_{(n)_{i,j}} \right)^2$$

Reference for Fractions Brier Score:

$$FBS_{(n)_{ref}} = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} F_{(n)_{i,j}}^2 + \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} O_{(n)_{i,j}}^2$$

FSS aggregated over forecasts

$$\left\langle FSS_{(n)} \right\rangle = 1 - \frac{\sum_{k=1}^{N_k} FBS_{(n)_k}}{\sum_{k=1}^{N_k} FBS_{(n)_{ref_k}}}$$

Ensemble Fractions Skill Score

Uses ensemble neighbourhood forecast probability...

$$FBS_{(n)} = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left(\frac{1}{M} \sum_{m=1}^{M} F_{(n)_{i,j,m}} - O_{(n)_{i,j}} \right)^2$$

$$FBS_{(n)_{ref}} = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left(\frac{1}{M} \sum_{m=1}^{M} F_{(n)_{i,j,m}} \right)^2 + \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} O^2_{(n)_{i,j}}$$

Error Fractions Skill Score *eFSS*

Spatial error of the ensemble

$$eFBS_{(n)} = \sum_{m=1}^{M} \sum_{i=1}^{N_x N_y} \left(F_{(n)_{i,m}} - O_{(n)_i} \right)^2$$

$$eFBS_{(n)_{ref}} = \sum_{m=1}^{M} \left(\sum_{i=1}^{N_x N_y} F_{(n)_{i,m}}^2 + \sum_{i=1}^{N_x N_y} O_{(n)_i}^2 \right)$$

Dispersion Fractions Skill Score *dFSS*

Spatial dispersion of the ensemble

$$dFBS_{(n)} = \sum_{m_a=1}^{M-1} \sum_{m_b=m_a+1}^{M} \sum_{i=1}^{N_x N_y} \left(F_{(n)_{i,m_a}} - F_{(n)_{i,m_b}} \right)^2$$

$$dFBS_{(n)_{ref}} = \sum_{m_a=1}^{M-1} \sum_{m_b=m_a+1}^{M} \left(\sum_{i=1}^{N_x N_y} F^2_{(n)_{i,m_a}} + \sum_{i=1}^{N_x N_y} F^2_{(n)_{i,m_b}} \right)$$

dFSS comparisons

2 Members

1 Comparison

dFSS comparisons

3 Members

3 Comparisons

dFSS comparisons

4 Members

6 Comparisons

dFSS comparisons

5 Members

10 Comparisons

dFSS comparisons

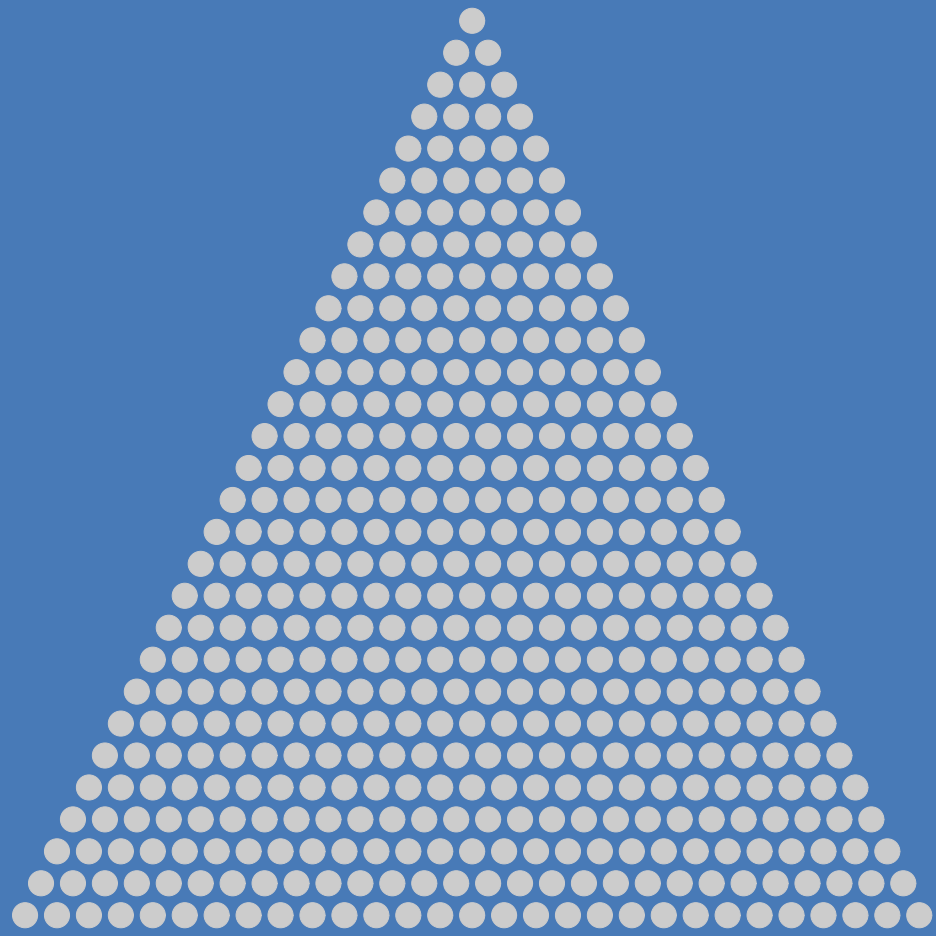
6 Members

15 Comparisons

dFSS comparisons

30 Members

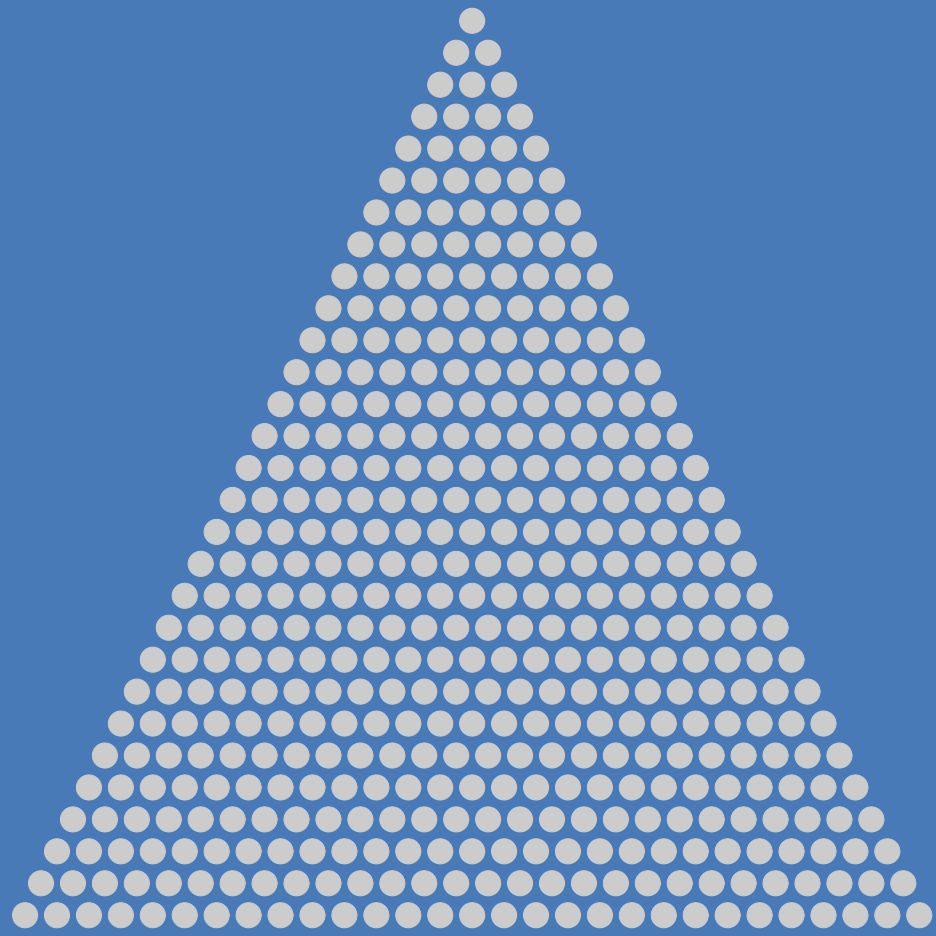
435 Comparisons



dFSS comparisons

M Members

$(M-1)^{th}$ Triangle number Comparisons



Ensemble FSS in harp

Follows normal harp workflow

Follows normal harp workflow

```
library(harp)

# Read forecast and decumulate
fcst <- read_forecast(...)
fcst <- decum(fcst, 1)
```

Follows normal harp workflow

```
library(harp)

# Read forecast and decumulate
fcst <- read_forecast(...)
fcst <- decum(fcst, 1)

# Read analysis and join
obs <- read_analysis(
  unique_valid_dttm(fcst),
  ...,
  transformation      = "regrid",
  transformation_opts = regrid_opts(get_domain(fcst))
)
fcst <- join_to_forecast(fcst, obs)
```

Follows normal harp workflow

```r
library(harp)

# Read forecast and decumulate
fcst <- read_forecast(...)
fcst <- decum(fcst, 1)

# Read analysis and join
obs <- read_analysis(
  unique_valid_dttm(fcst),
  ...,
  transformation      = "regrid",
  transformation_opts = regrid_opts(get_domain(fcst))
)
fcst <- join_to_forecast(fcst, obs)

# Compute the FSS
verif <- nbhd_verify(
  fcst,
  anl,
  threshold = c("q0.99", "0.5"),
  radius    = c(0, 5)
)
```

Speed up with parallelization

Speed up with parallelization

```
1 library(parallel)
2 # Initialize cluster
3 cl <- makeCluster(min(nrow(fcst), detectCores()))
```

Speed up with parallelization

```
1 library(parallel)
2 # Initialize cluster
3 cl <- makeCluster(min(nrow(fcst), detectCores()))
4
5 # Simple function to run on each processor
6 verif_func <- function(x, th, nb) {
7   library(harpSpatial)
8   nbhd_verify(x, anl, th, nb)
9 }
```

Speed up with parallelization

```
library(parallel)
# Initialize cluster
cl <- makeCluster(min(nrow(fcst), detectCores()))

# Simple function to run on each processor
verif_func <- function(x, th, nb) {
  library(harpSpatial)
  nbhd_verify(x, anl, th, nb)
}

# Apply the function on each row of the data frame
# Each processor takes one row at a time
verif <- parLapply(
  cl,
  split(fcst, as.numeric(rownames(fcst))),
  verif_func,
  c("q0.99", "0.5"),
  c(0, 5)
)
```

Speed up with parallelization

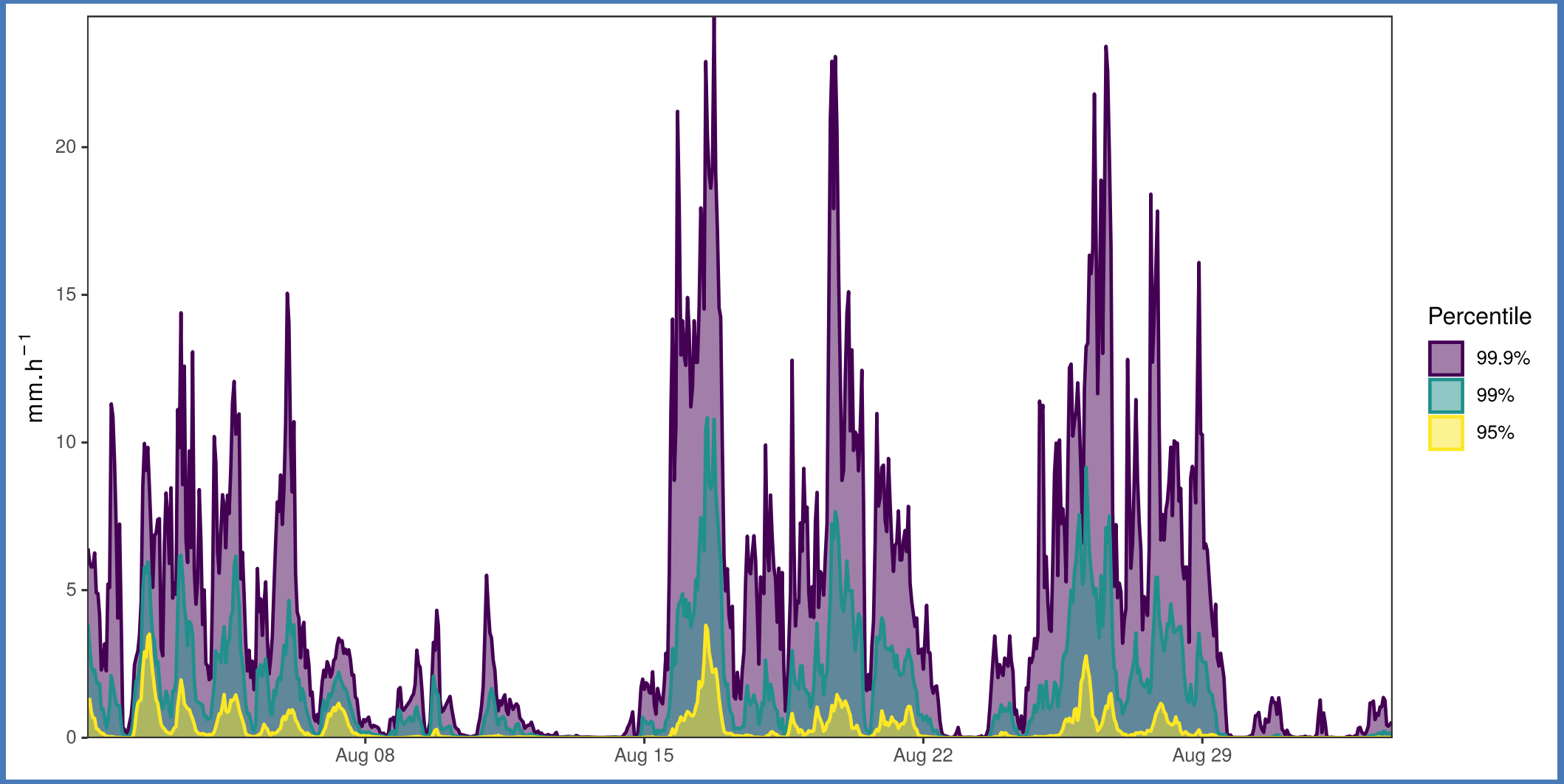
```
library(parallel)
# Initialize cluster
cl <- makeCluster(min(nrow(fcst), detectCores()))

# Simple function to run on each processor
verif_func <- function(x, th, nb) {
  library(harpSpatial)
  nbhd_verify(x, anl, th, nb)
}

# Apply the function on each row of the data frame
# Each processor takes one row at a time
verif <- parLapply(
  cl,
  split(fcst, as.numeric(rownames(fcst))),
  verif_func,
  c("q0.99", "0.5"),
  c(0, 5)
)

# Stop the cluster
stopCluster(cl)
```

Example

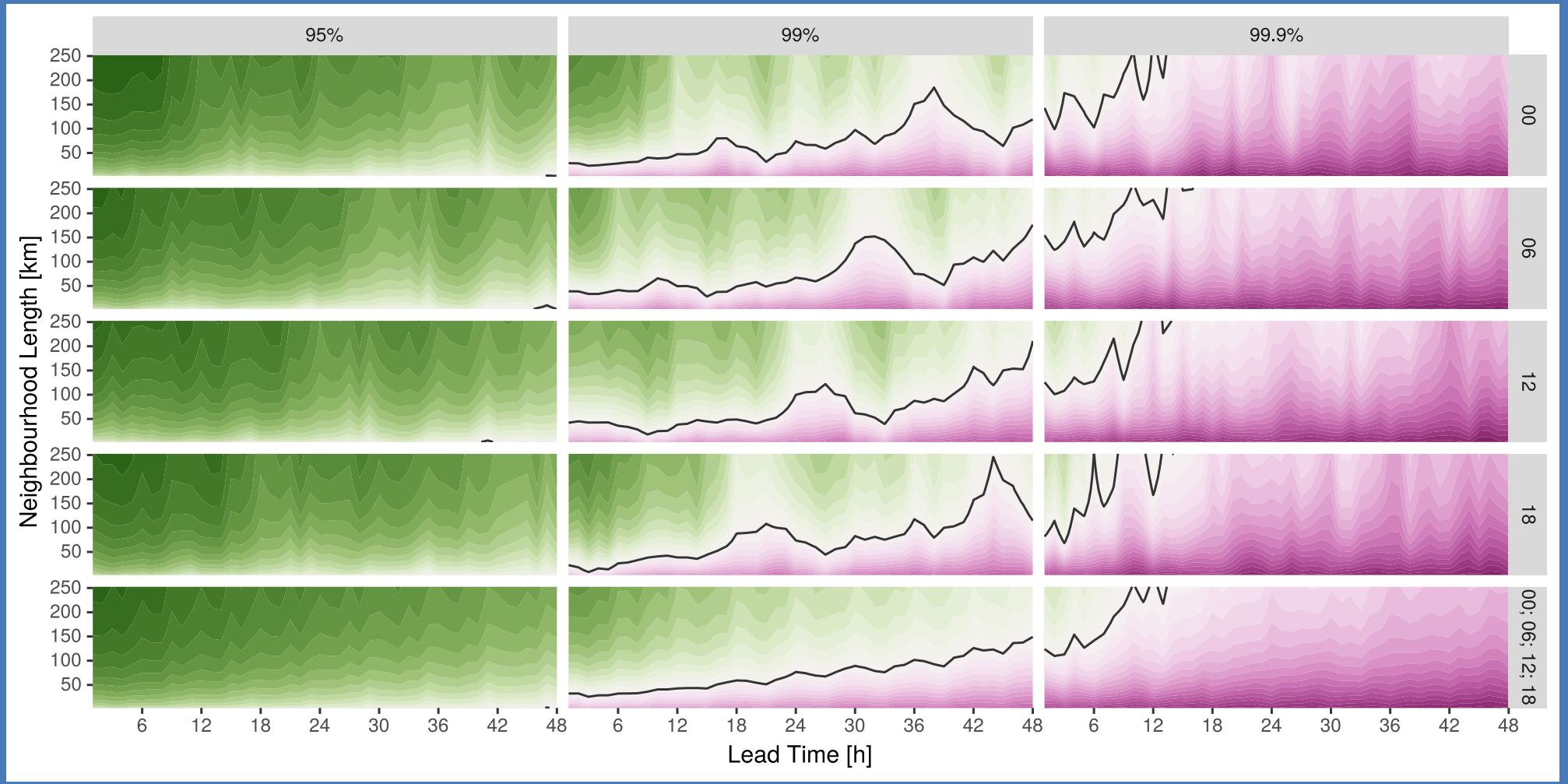
MEPS ensemble

- 1 - 31 August 2022
- 30 member ensemble
- Initialization times at 00; 06; 12; 18 UTC
- 300 x 300 grid points
- 750 km x 750 km
- 1 hour precipitation
- Neighbourhoods up to 250 km

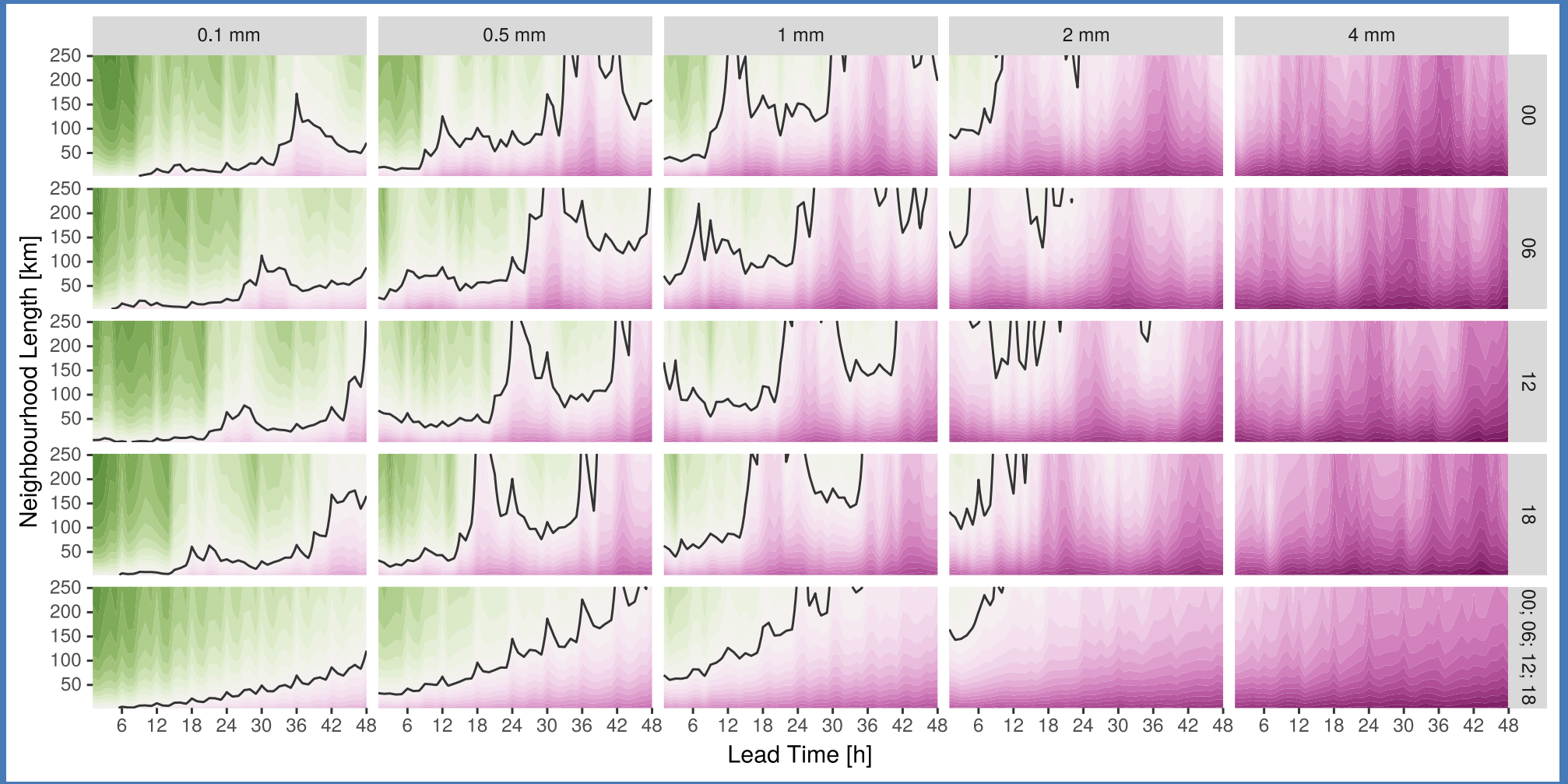
Precipitation thresholds



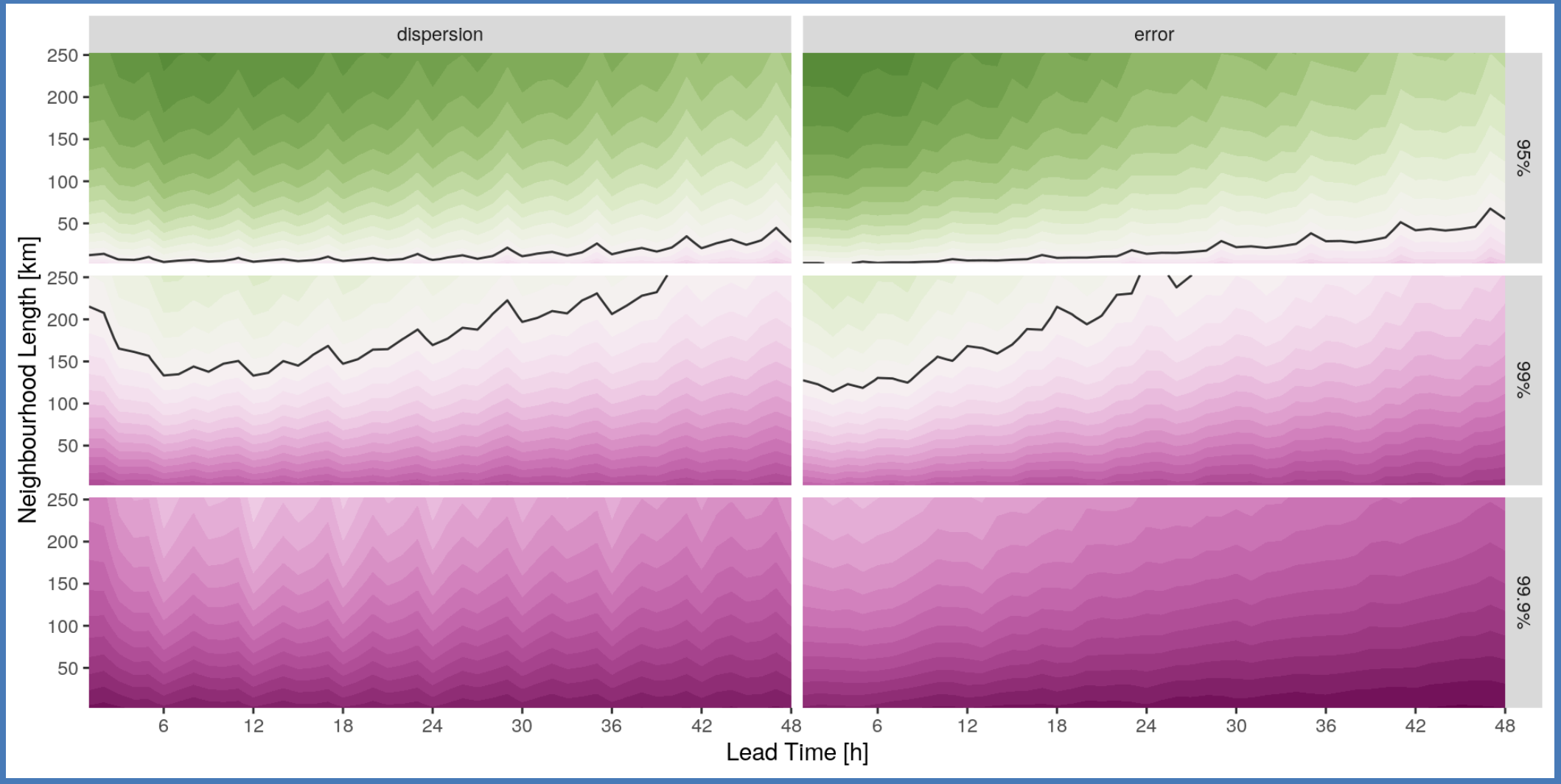
Ensemble FSS - percentiles



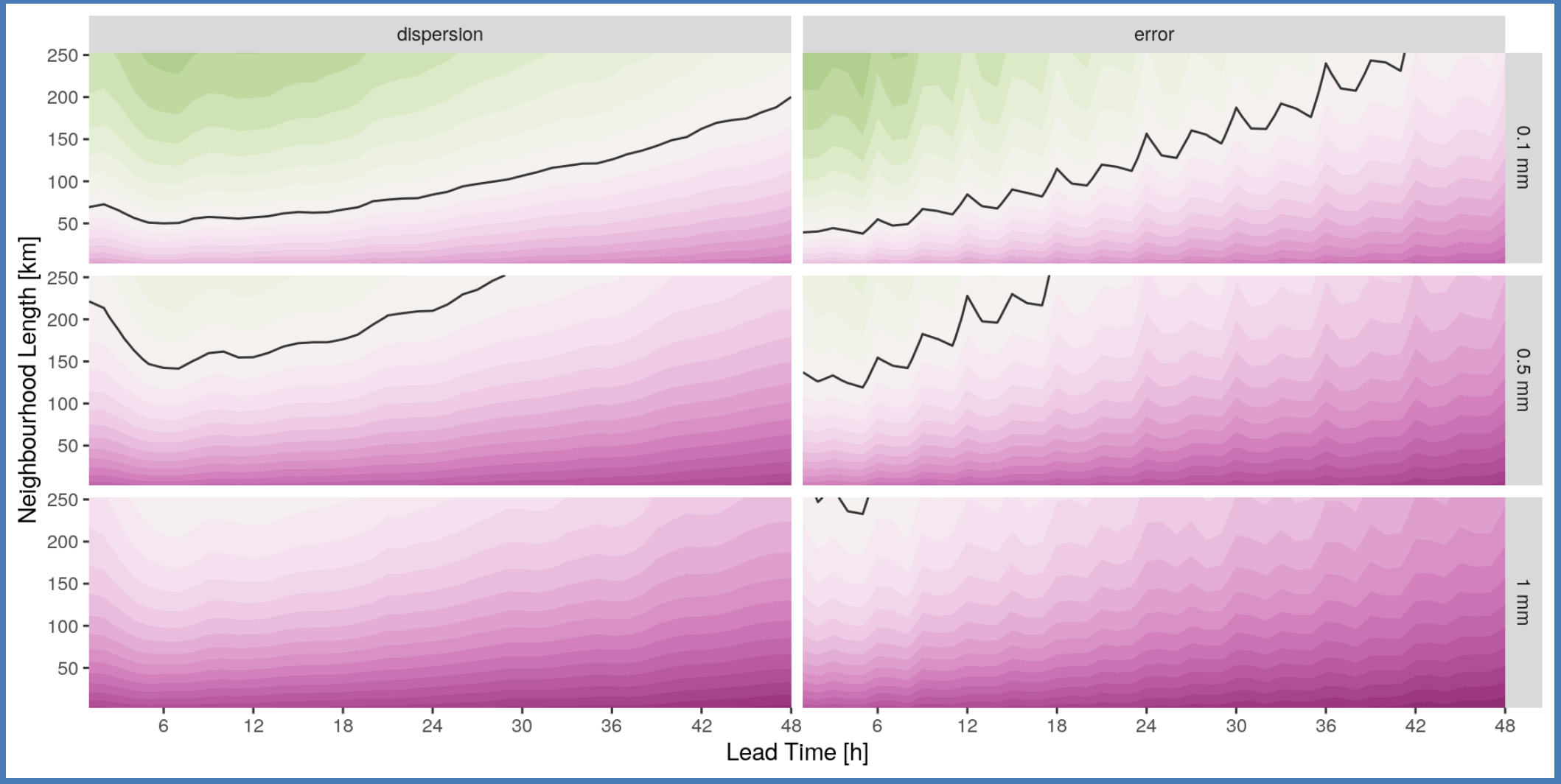
Ensemble FSS - thresholds



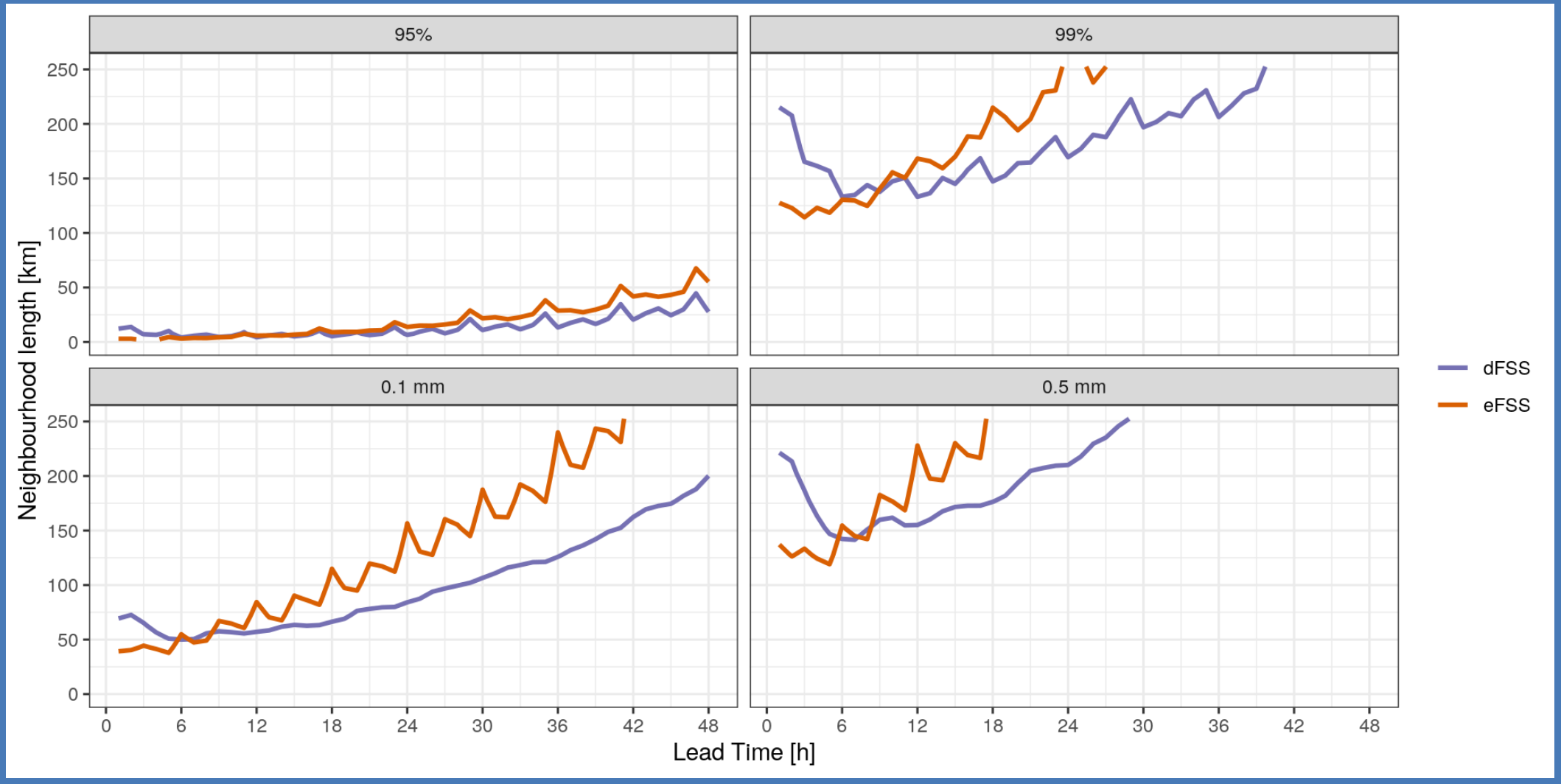
dFSS :: eFSS - percentiles



dFSS :: eFSS - thresholds



Spatial spread :: skill



harp News

New harp version ~Easter

- New version of harp built on harpCore
  - includes comprehensive unit tests
  - `nbhd_verify()` included in harpSpatial
- More improvements later in 2023
  - merging of harpPoint and harpSpatial
  - dispatch of verification function(s) dependent on data
  - more options in shiny app(s)
  - “operational ready” functions