Current Status of
CML Data Assimilation
and
Targeted Covariance Inflation

RealPEP

Deutscher Wetterdienst
Wetter und Klima aus einer Hand

DWD

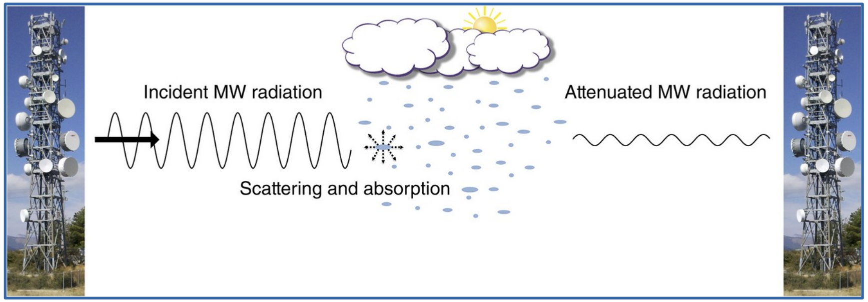
CML Basics

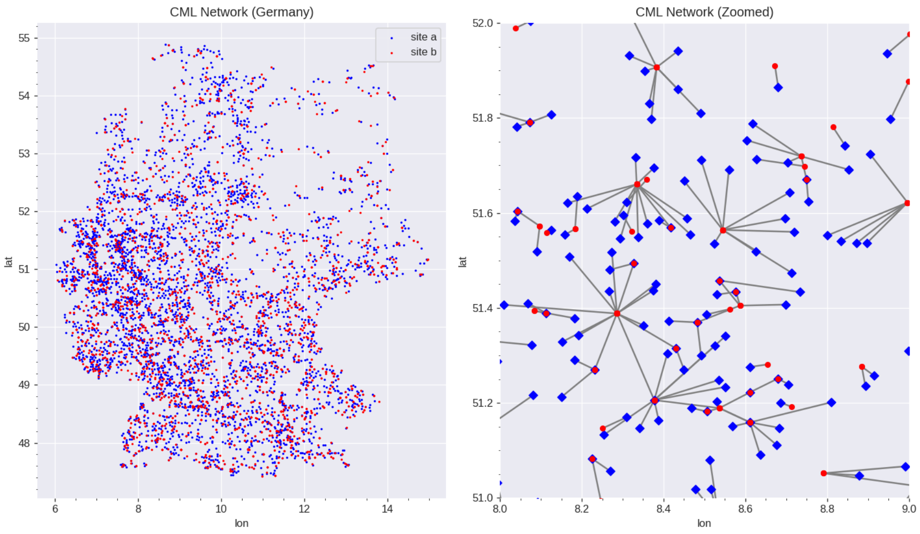
CML Basics: Introduction

- **Commercial microwave link (CML)** data successfully employed for the estimation of rain rates (QPE) (→C. Chwala, P1)
- overall objective here (→P3): **data assimilation** of CML data in **numerical weather prediction models** for **improving QPF**
  - able to contribute to bridging the gap between QPN and NWP?
  - (How much) does it improve QPF?
  - How does it compare to Radar data assimilation?
- in the following: discussion of **technical details** of CML data assimilation and presentation of **first results**

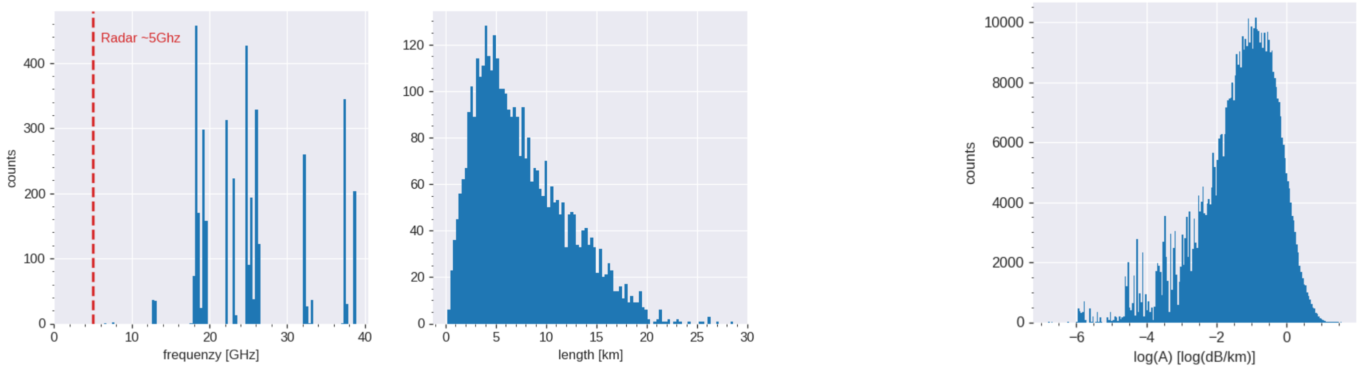
CML Basics: Overview I

- CMLs are used to interconnect cell phone towers
- each CML consists of **sender** and **receiver**
- transmitted radiation gets **attenuated** by (e.g.) raindrops
- ~4000 CMLs in current dataset for June 2019
- temporal resolution 1min



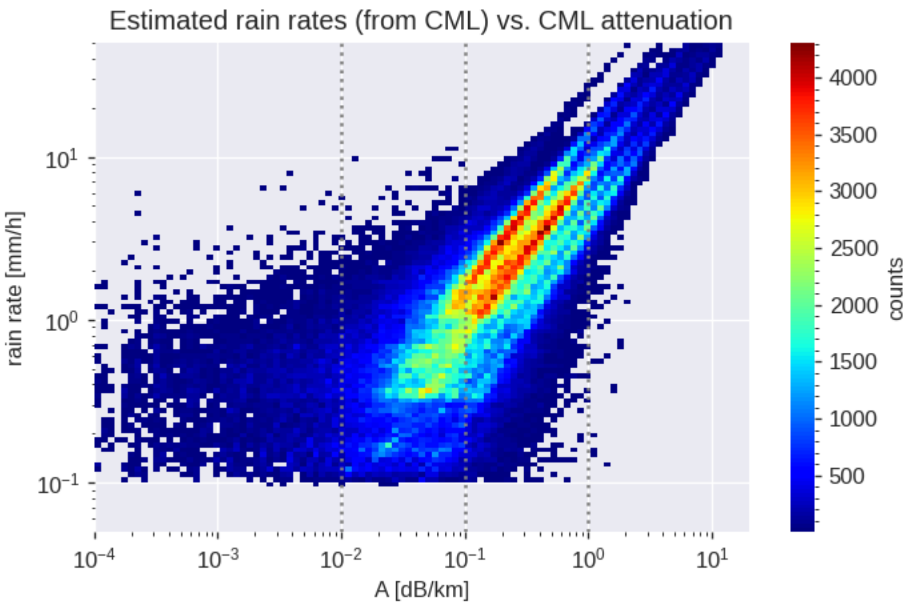


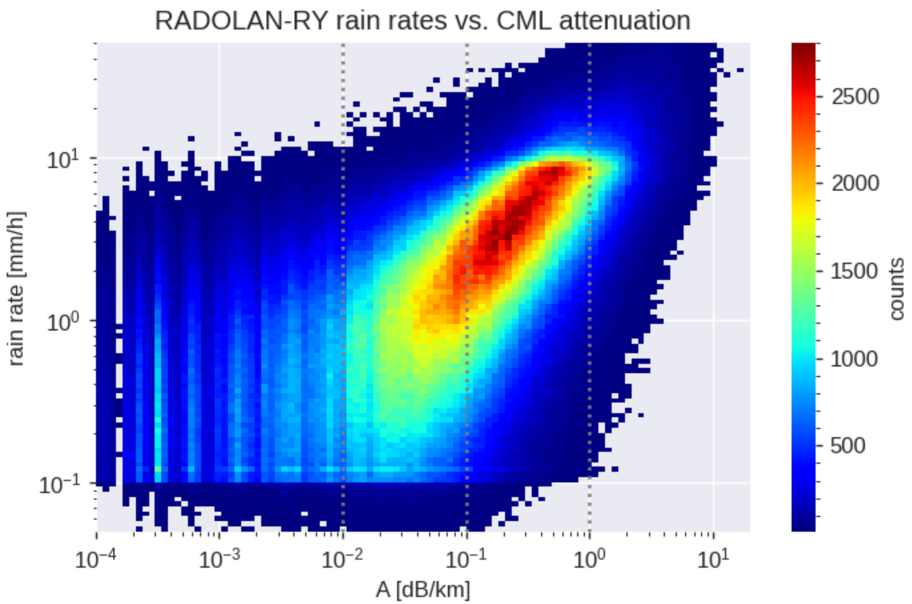
CML Basics: Overview II



- **CML frequency** significantly above DWD **Radar frequency** → different physics involved!
- use **path-integrated specific attenuation** for assimilation
  - referred to as A from now on
  - A [dB/km] = attenuation [dB] / distance [km]
  - direct **relationship of A with rain rate** (→ power law)
- most attenuations **very** small

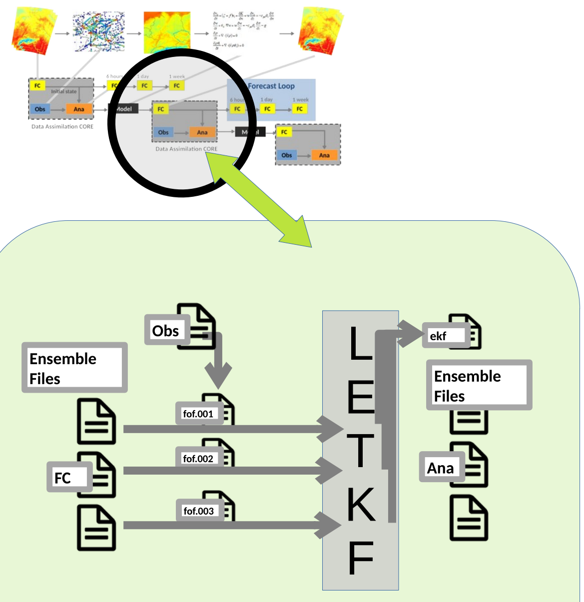
CML Basics: Rain Rate vs. A





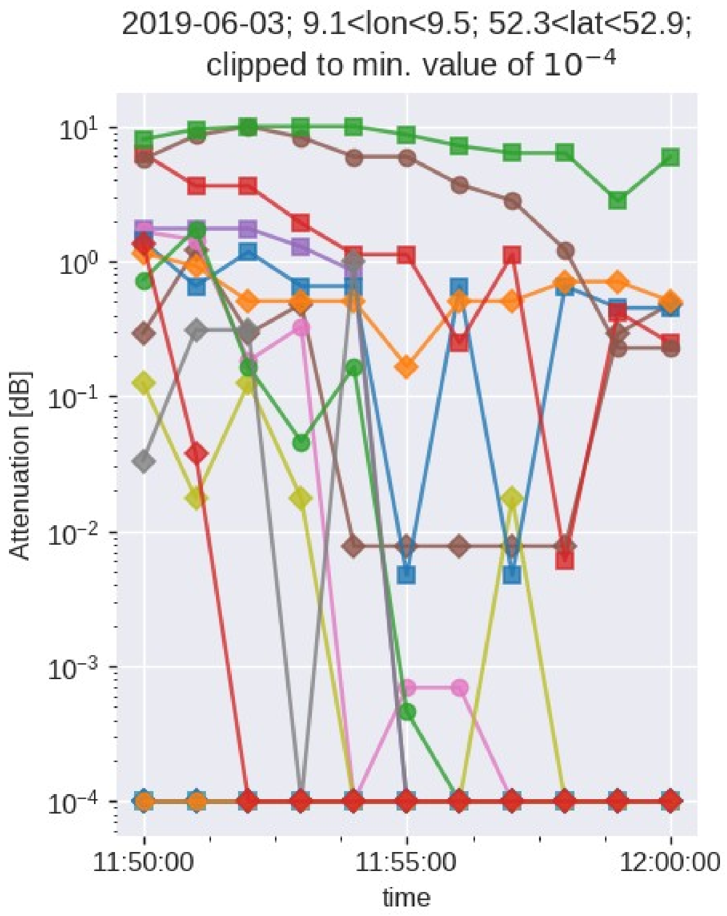
- “linear” relationship (on double logarithmic scale)
  → hint at underlying power law
- (very) noisy data for A < ~10^{-2} dB/km

- for assimilating data **feedback/fof files** have to be generated
- each (ens.) fof file contains all data relevant to LETKF assimilation process (specific date)
- particularly, for each observation there has to be a **simulated model equivalent**
- **built automated system** for the construction of **CML feedback files**
  - includes all necessary data processing steps
  - implemented (mostly) in Python
  - integrated into new BACY experiment

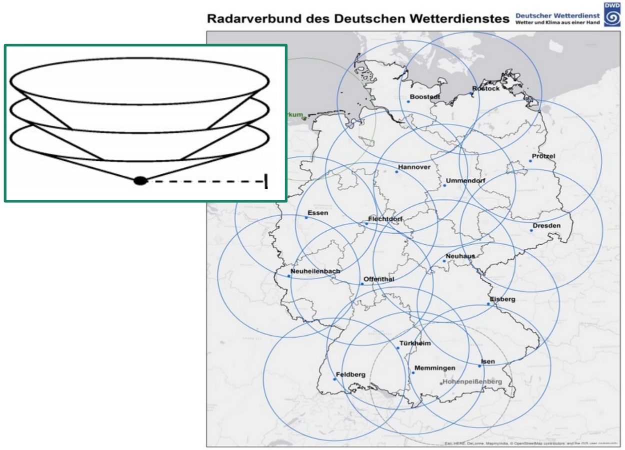


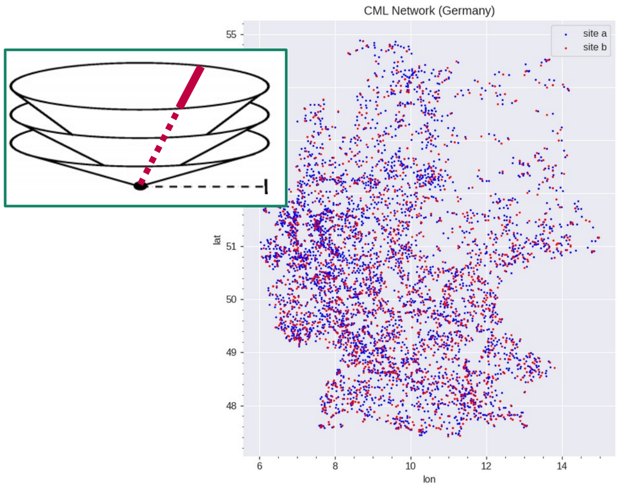
- fof.*: sim. + obs. quantities of ens. members
- LETKF produces increments depending on innovations + Kalman gain

- **temporal superobbing/smoothing**:
  - for an assimilation at t_0 calculate the mean of all observations falling within a 10 min time window [t_0 – 10 min, t_0] for each CML
  - **smooths** out **erratic fluctuations** of attenuations
- **outlook**: also perform spatial thinning and/or superobbing



- using the **Radar forward operator EMVORADO** in offline mode for **simulating attenuations**, i.e., calculating relevant model equivalents
- important **differences** Radar vs. CML:
  - **Radar**: 17 stations, many azimuths, few elevations, frequency ~5 GHz
  - **CML**: ~4000 “stations”/sender, individual azimuth/elevation (only one per station) and frequency within 10 – 40 GHz



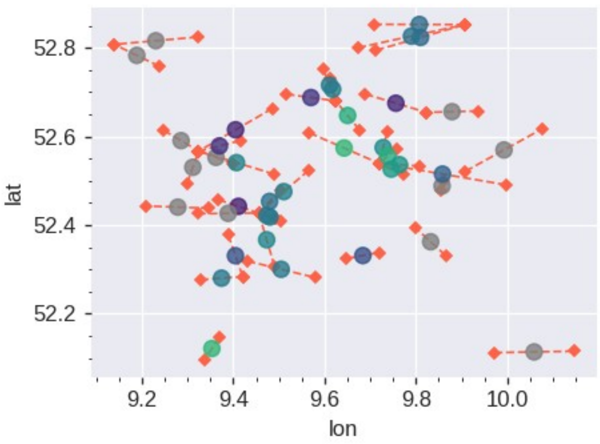


CML Basics: Simulation with EMVORADO (II)

- two **main inputs** for EMVORADO (many other config. options):
  - ICON **model fields** (regular grid) for **hydrom.** qr, qg, qv, ...
  - **auto-generated namelist** with information **for each** CML
    - CML sender is interpreted as Radar station
    - lat/lon/level of “station”, azimuth/elev. of ray, frequency, ...
- extract **path-integrated one-way attenuation** from output
- perform EMVORADO run for **each ensemble member**
- current **limitations**:
  - single EMVORADO run not able to simulate all (~4000) CMLs
  - simulation does not include water vapor attenuation

CML Basics: Feedback File Construction

- **collect** processed **observed** and **simulated** data for specific **assim. date**
- use **halfway** lat/lon/level of each CML in feedback files
- CML data currently assimilated as SYNOP observation (*obstype*) and using an experimental *codetype* and *varno*
- write all data into feedback (netcdf) file

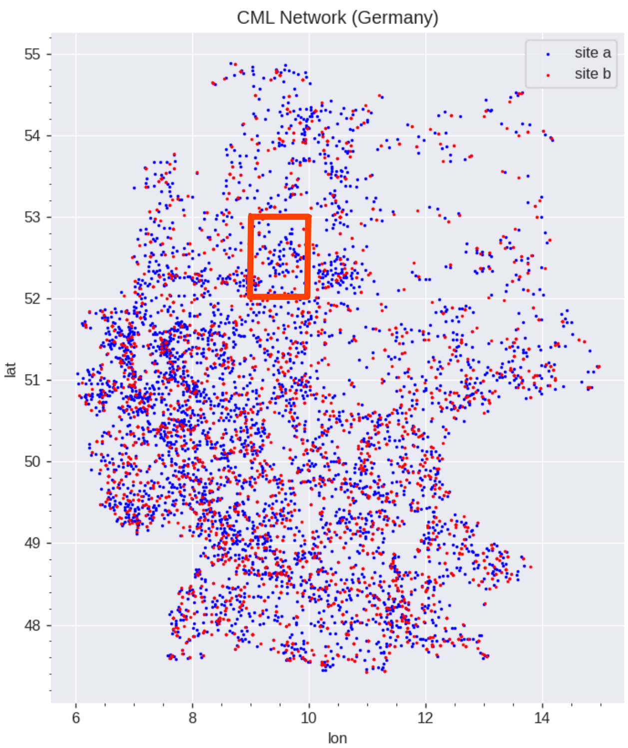


obstype	(d_hdr)	float32	1.0 1.0 1.0 1.0 ... 1.0 1.0 1.0 1.0
codetype	(d_hdr)	float32	991.0 991.0 991.0 ... 991.0 991.0
statid	(d_hdr)	|S10	b'1085' b'1208' ... b'1716' b'1727'
lat	(d_hdr)	float32	52.82 52.78 52.44 ... 52.51 52.37
lon	(d_hdr)	float32	9.23 9.189 9.278 ... 9.858 9.473
time	(d_hdr)	float32	60.0 60.0 60.0 ... 60.0 60.0 60.0
varno	(d_body)	float32	991.0 991.0 991.0 ... 991.0 991.0
obs	(d_body)	float32	0.0 0.0 0.0 ... 0.09817 0.0002519
level	(d_body)	float32	60.52 61.24 103.1 ... 99.79 110.8
state	(d_body)	float32	1.0 1.0 1.0 1.0 ... 1.0 1.0 1.0 1.0
e_o	(d_body)	float32	0.0 0.0 0.0 ... 0.01963 5.038e-05
veri_data	(d_veri, d_body)	float32	0.0 0.0 ... 0.01968 0.07365

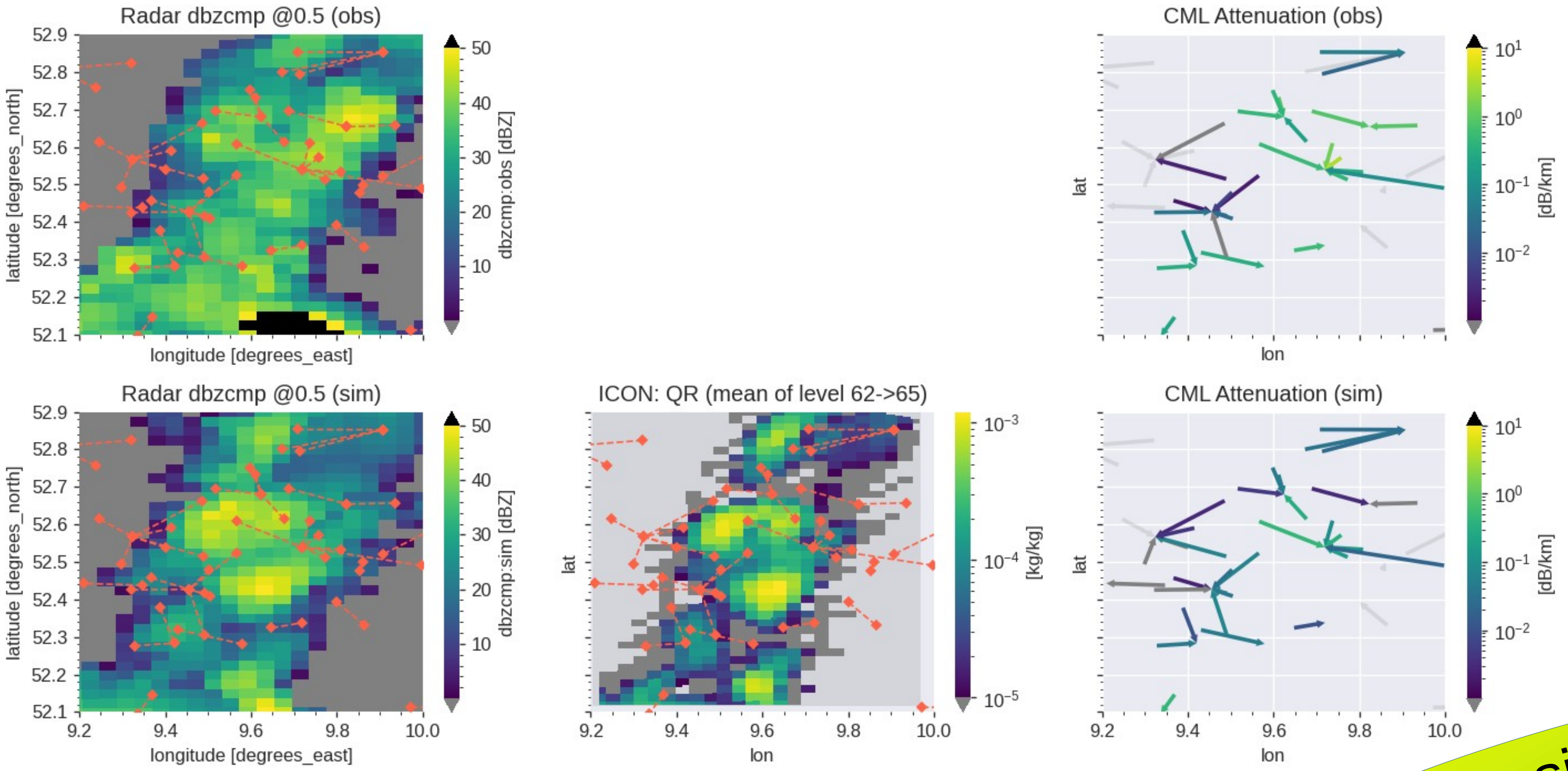
CML Case Study I

CML Case Study: Overview

- perform assimilation on 2019-06-03 at 12:00:00
- only use CMLs within region
  9.2° < lon < 10° and 52.1° < lat < 52.9°
  - evades EMVORADO limitation
  - 40 CMLs within this region
- **only CML** data is set to **active** here!

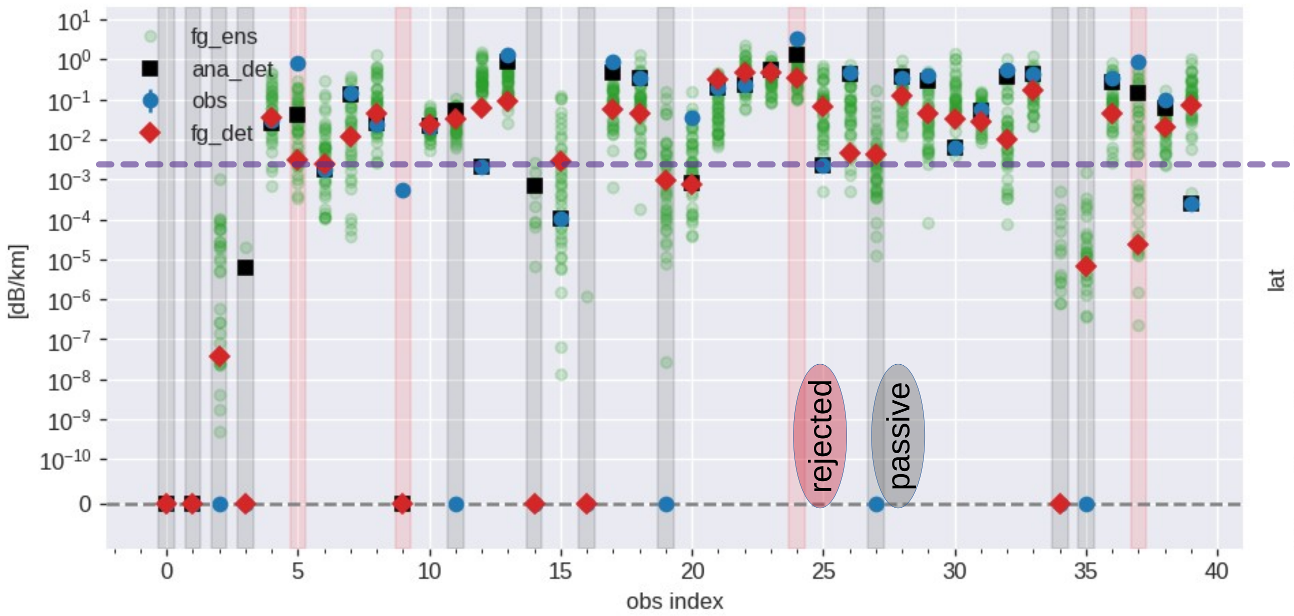


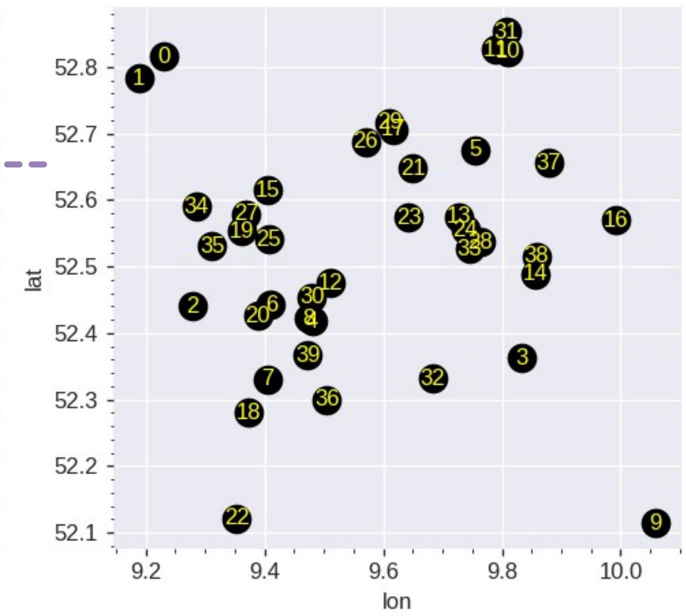
CML Case Study: Plausibility Check of Data



consistency/ plausibility check : ✔

CML Case Study: Assimilation Result





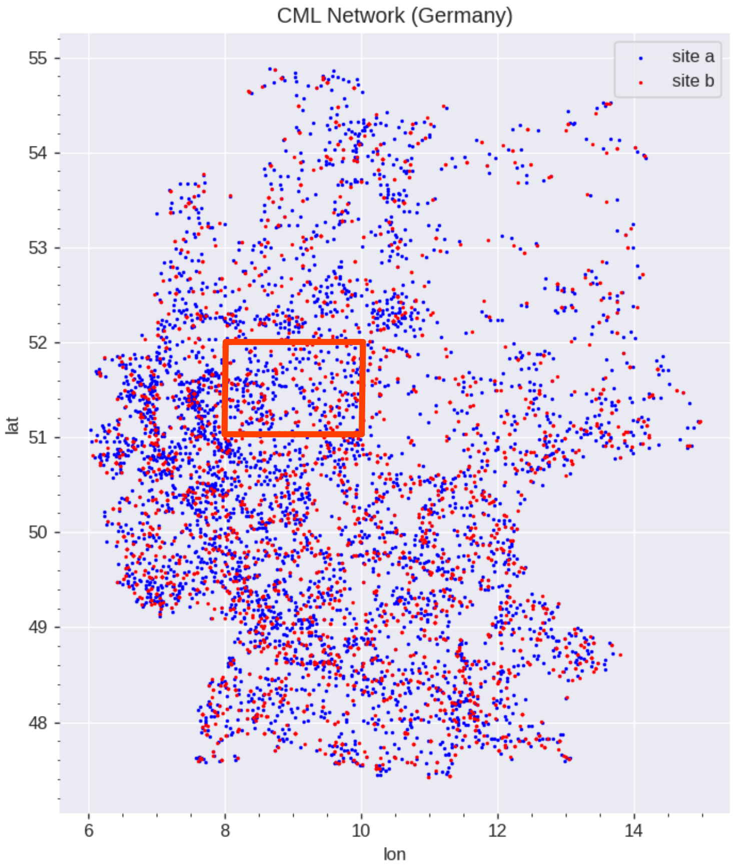
- representation of corresponding “ekf” file (LETKF output)
- shaded background
  → special assimilation state

result: system works (technically)

CML: Case Study II

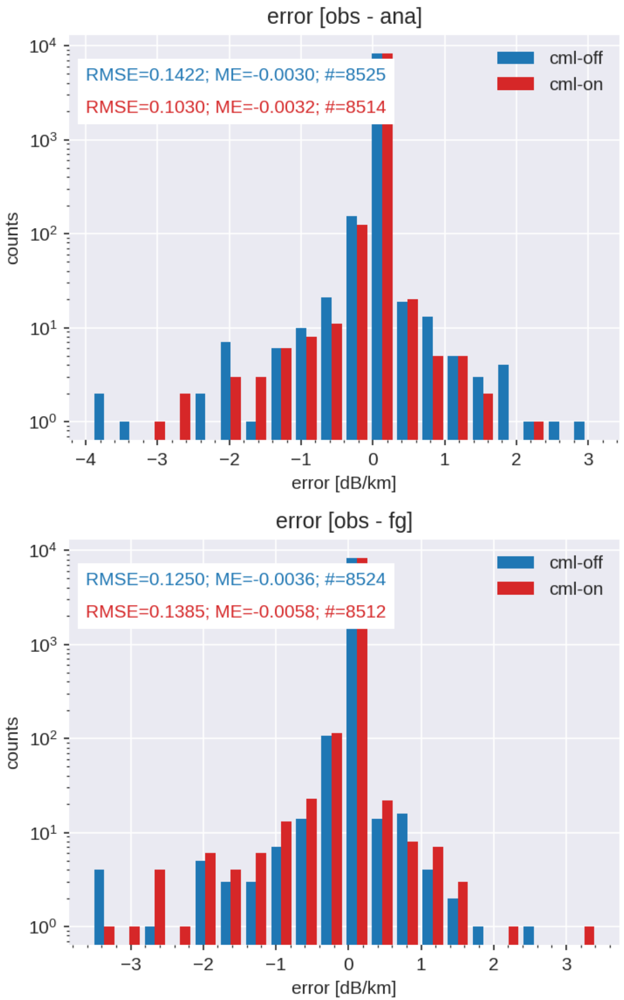
CML Case Study: Overview

- perform **BACY cycle** for **2 days** (on 2019-06-03/04)
- only use **CMLs within region** 8° < lon < 10° and 51° < lat < 52°
- 185 CMLs within this region
- **CML and CONV** data set to active
- vertical localization off

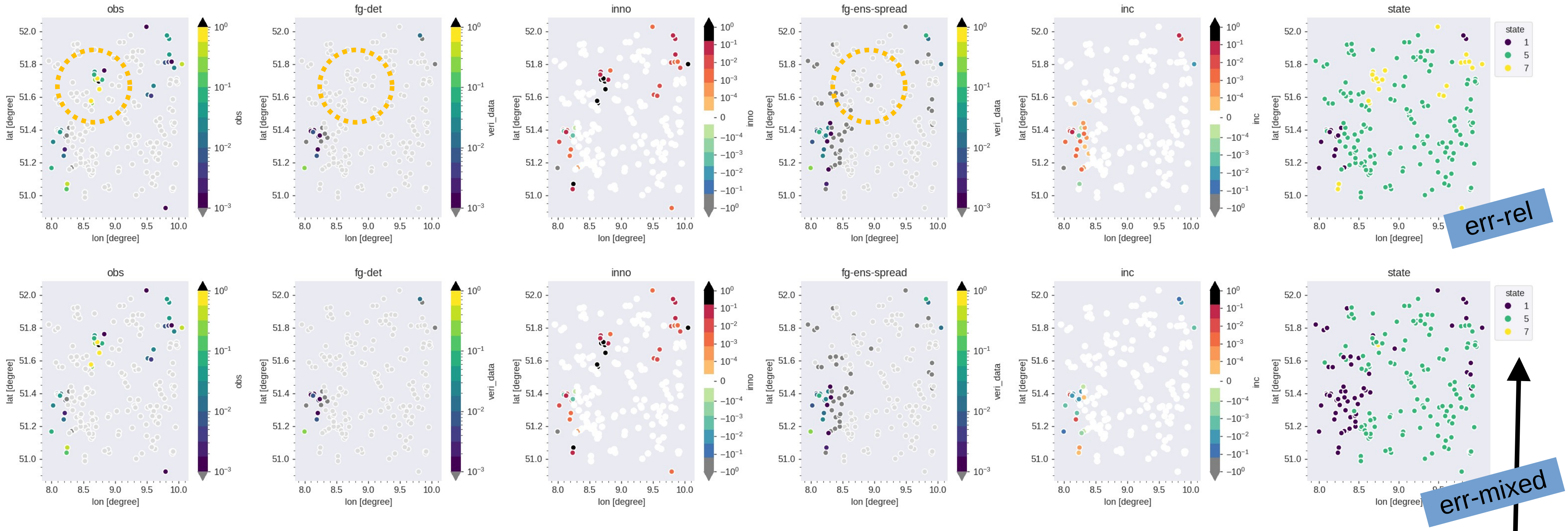


CML Case Study: Obs. Err. Stat.

- humidity and temperature stat. (AIREP/TEMP) looked “okay”
- BUT: CML data itself is pulled in wrong direction. Possible causes:
  - localization (→correlations)
  - observation error (here: 20%)
- next steps: look at effects of CML data assimilation more closely
  - performing single “core-more runs”: single assimilation followed by an ICON model run
  - study LETKF output, increments, and model dynamics (under parameter changes)



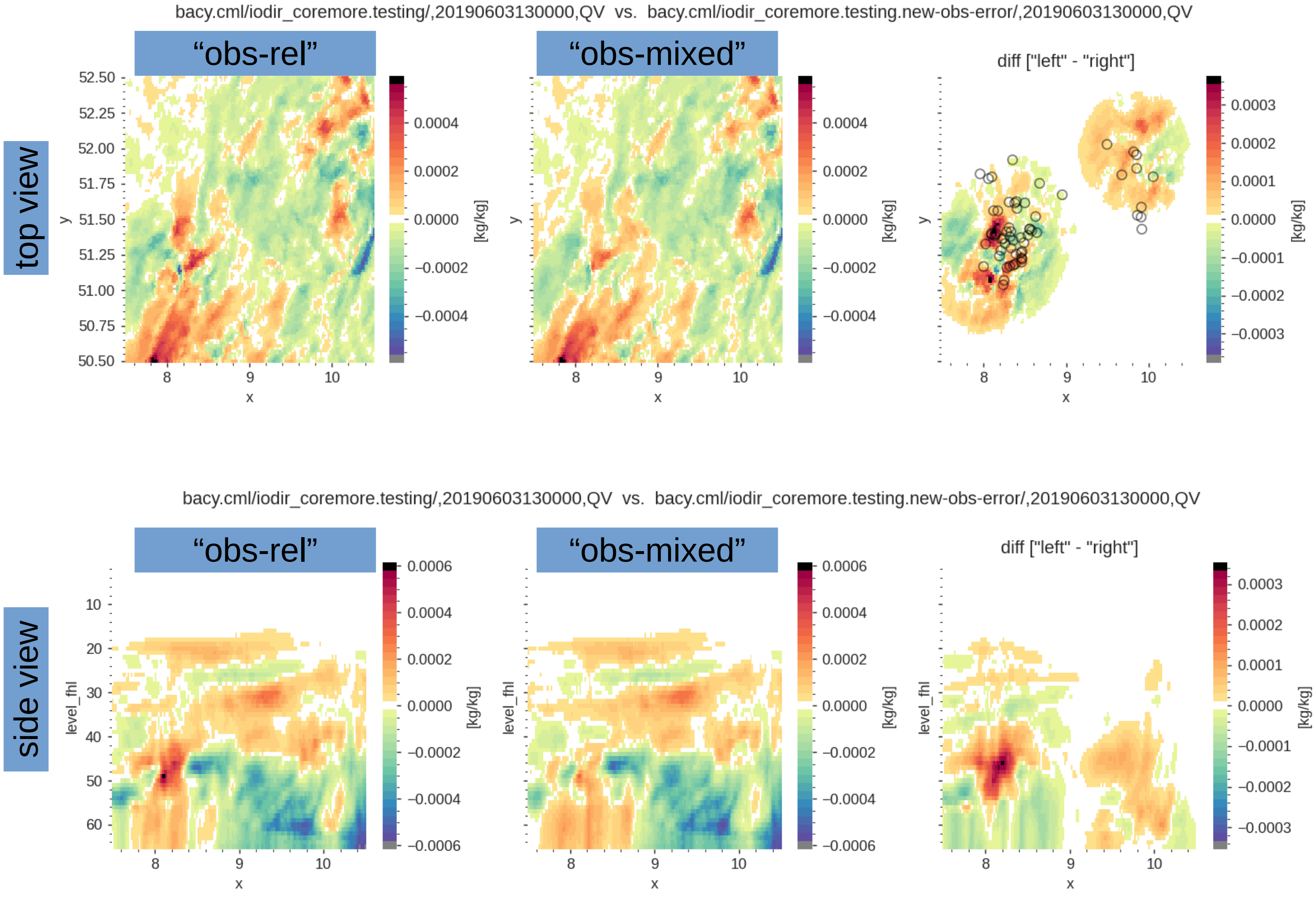
CML Case Study: Assimilation Results



- two options for observation error
  - "err-rel": relative 20% (not really realistic!)
  - "err-mixed": absolute 0.1 dB/km + relative 20% (more realistic?)
- interesting: large region with missing spread (→tci?)

state
1: active
5: passive
7: rejected

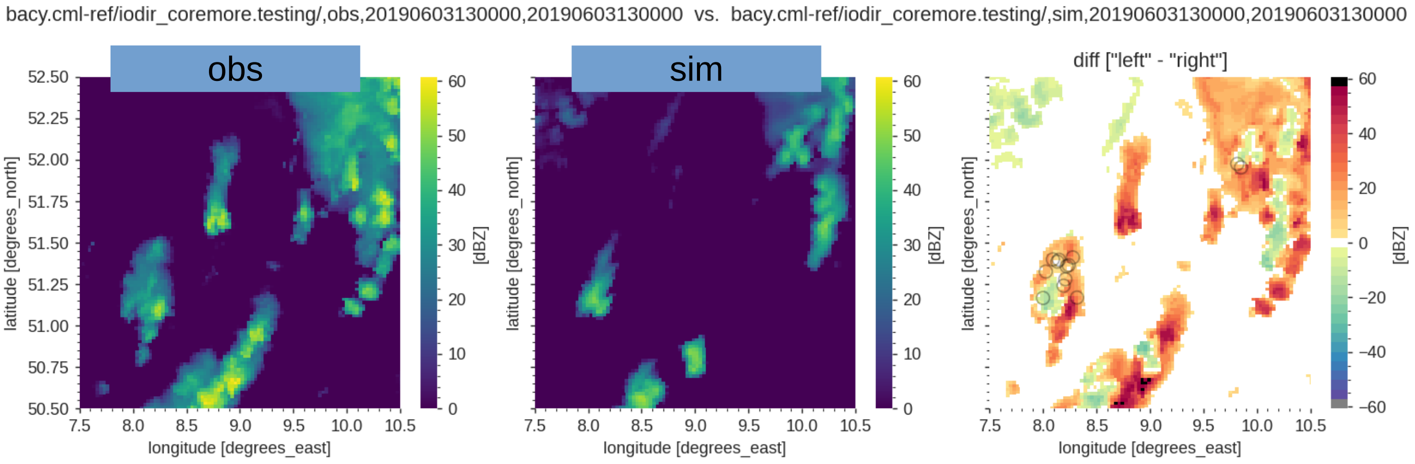
CML Case Study: ICON Increments



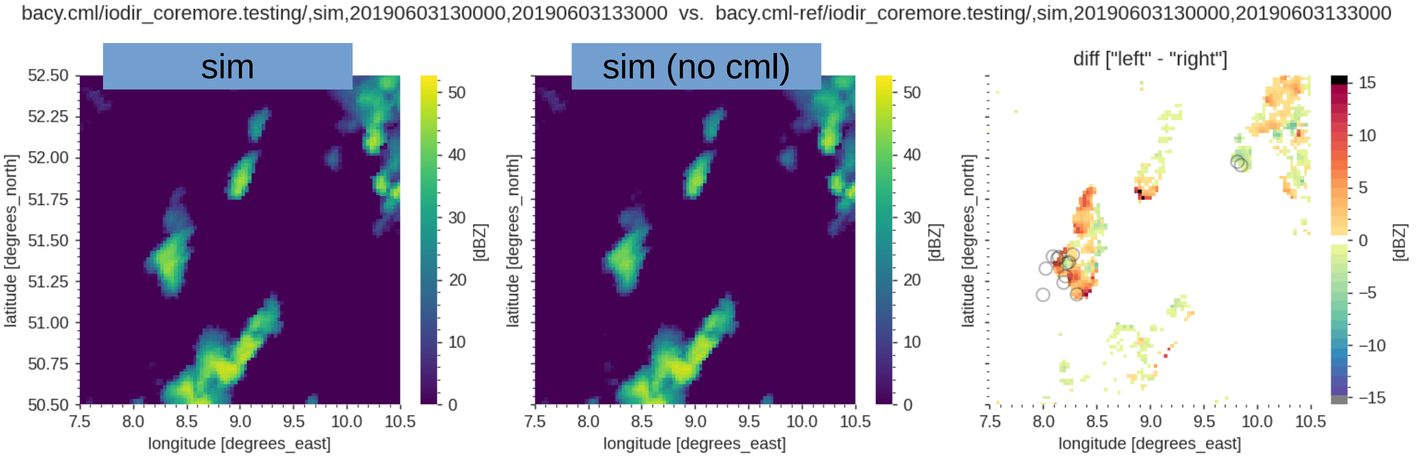
- model field increments for QV from LETKF
- reduced 3D fields to 2D fields via mean along dim. height/y
- clear difference between choices for obs. err.

CML Case Study: Model Response (Dbzcmp)

leadtime 0min



leadtime 30min

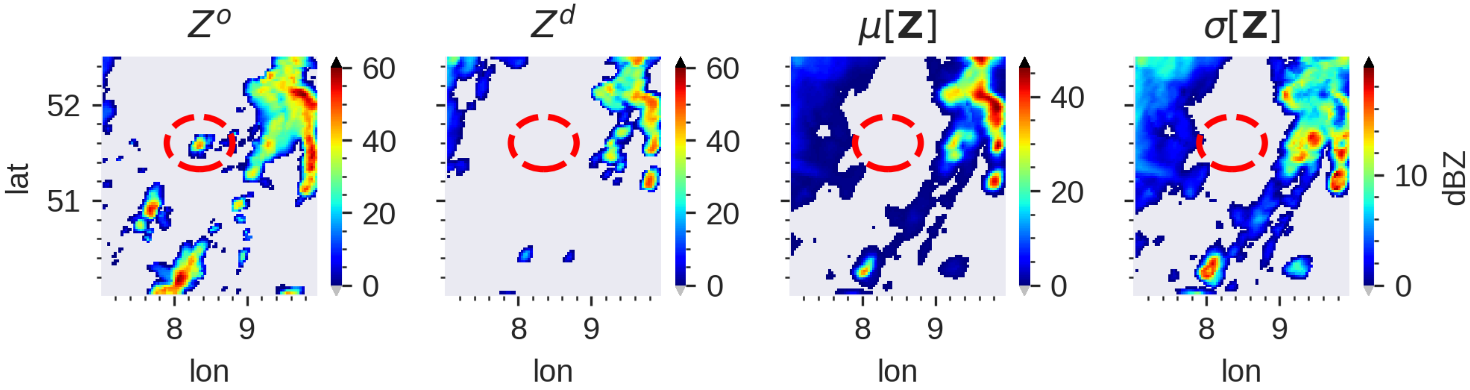


- discrepancies between obs./sim REFL at assimilation time
- clear impact of CML data assim. after 30 minutes

CML: Summary & Outlook

- first version for assimilating CML data (integrated into BACY)
- first assimilation results seem plausible
- performed first BACY cycles comparing “CONV” vs “CONV+CML”
- next steps:
  - further study the detailed effects of CML data assimilation (as already begun via single “core-more” exp.)
  - single-obs. experiments (great for studying correlations)
  - study impact of parameters like obs. error, localization, ...
  - general quality control, spatial thinning/superobbing, bias correction

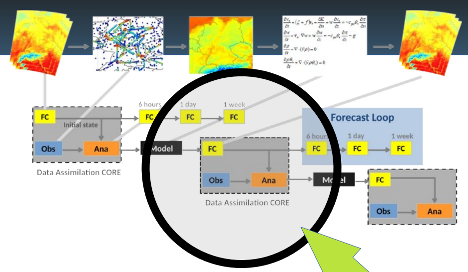
TCI Basics

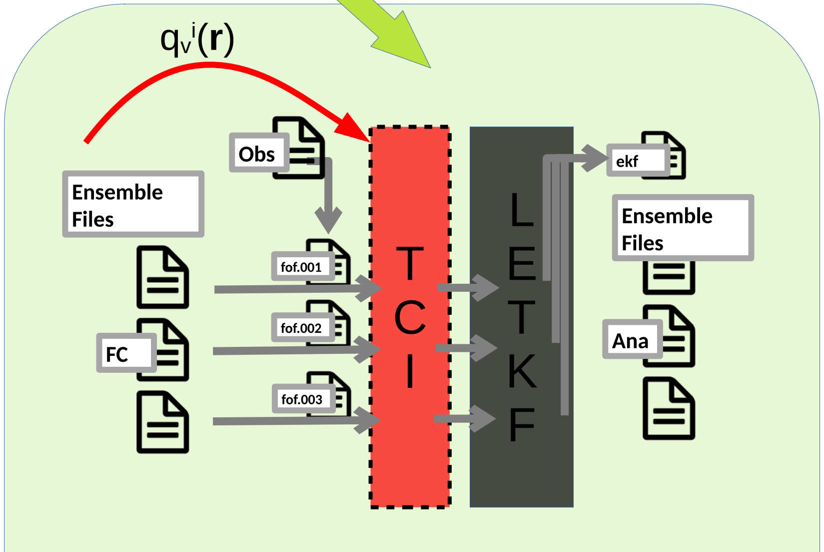


- even for **large discrepancies** between obs./sim. REFL LETKF might give **small increments** due to **small ensemble spread**
- targeted covariance inflation (TCI) approach:
  - **check conditions** (missing spread, large enough obs., ...)
  - apply suitable model: each ensemble member gets individual **“virtual” simulated REFL** leading to an increased spread

TCI Basics: Technical Steps

- implemented via **pre-processing of feedback (fof) files** before entering the LETKF
- apply TCI algorithm and **alter simulated Z** in feedback files
- each member processed separately
- use altered feedback files as input for LETKF



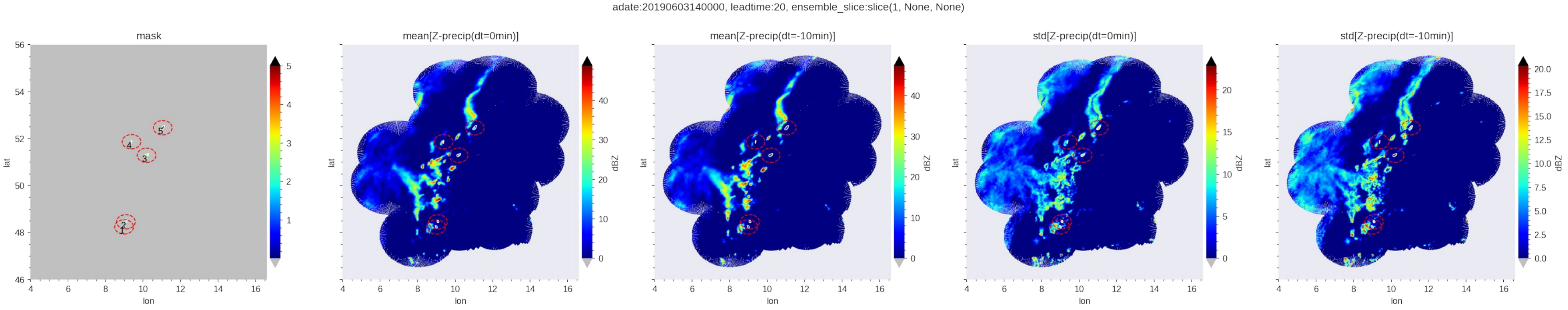


- fof.*: sim. + obs. quantities of ens. members → enter LETKF
- LETKF produces increments depending on innovations + Kalman gain

TCI Basics: Linear Model(s)

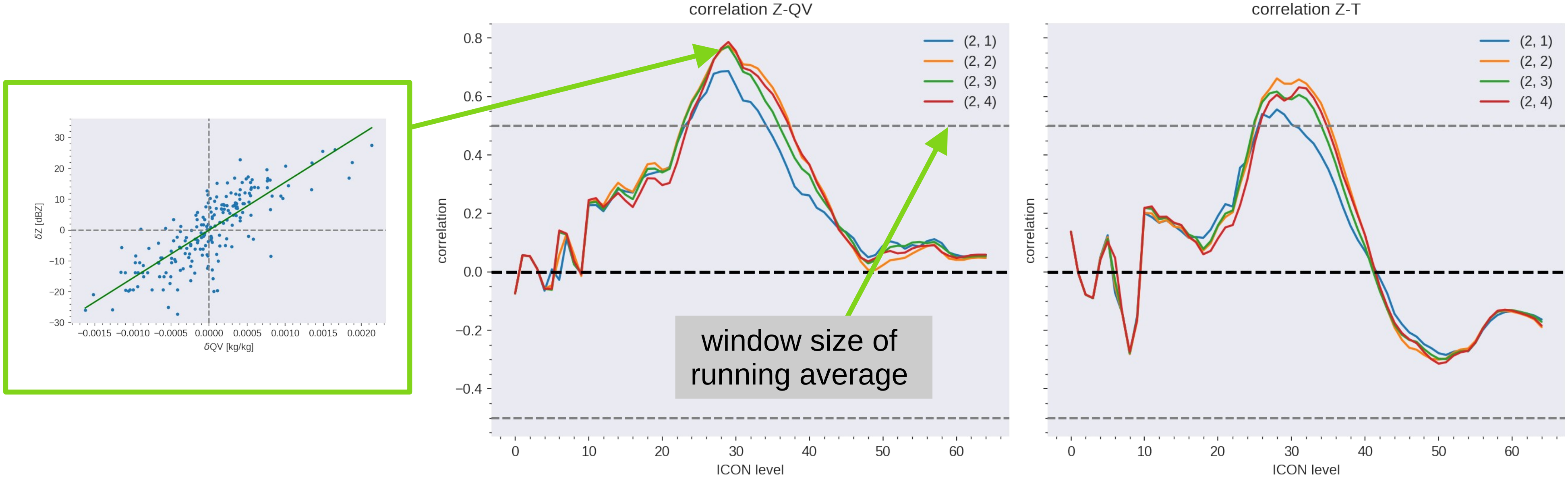
- **current model(s)**: based on simple linear regression
  - $M_{h,h'}$: $\delta Z_i(x,y,h,t) = \alpha * \delta qv_i(x,y,h',t)$
  - δZ_i, δqv_i: ensemble perturbations for Z, qv of i-th member
  - h, h’: categorical/discrete heights
- **overall idea**:
  - spread of qv “imprinted” onto spread of Z
  - assim. “favors” members with more humidity
  - additional increments for humidity qv are produced
  - model (hopefully) generates qr/qs/qg → EMVORADO sim. REFL

TCI Basics: Training Data Generation



- idea: training data should be **representative for convective events**
- built simple algorithm for the **detection of new cells**
  - employs **time series** of (binned) Radar data
  - gives **area and maximum position** of REFL (x_0,y_0) of newly emerged cell at time t_0
- single instance (for training of model $M_{h,h'}$): $\delta Z_i(x_0,y_0,h,t_0)$, $\delta qv_i(x_0,y_0,h',t_0)$

TCI Basics: Model Selection



- fixed height of REFLs of h="3000m-4000m"
- h' determined through **maximum of correlation** → h'=30
- resulting model:
  - $\delta Z_i(h=\text{"3000-4000m"}) = 10^4\,\text{dBZ} * \delta qv_i(h'=30)$

TCI Basics: ML-Based TCI (Outlook)

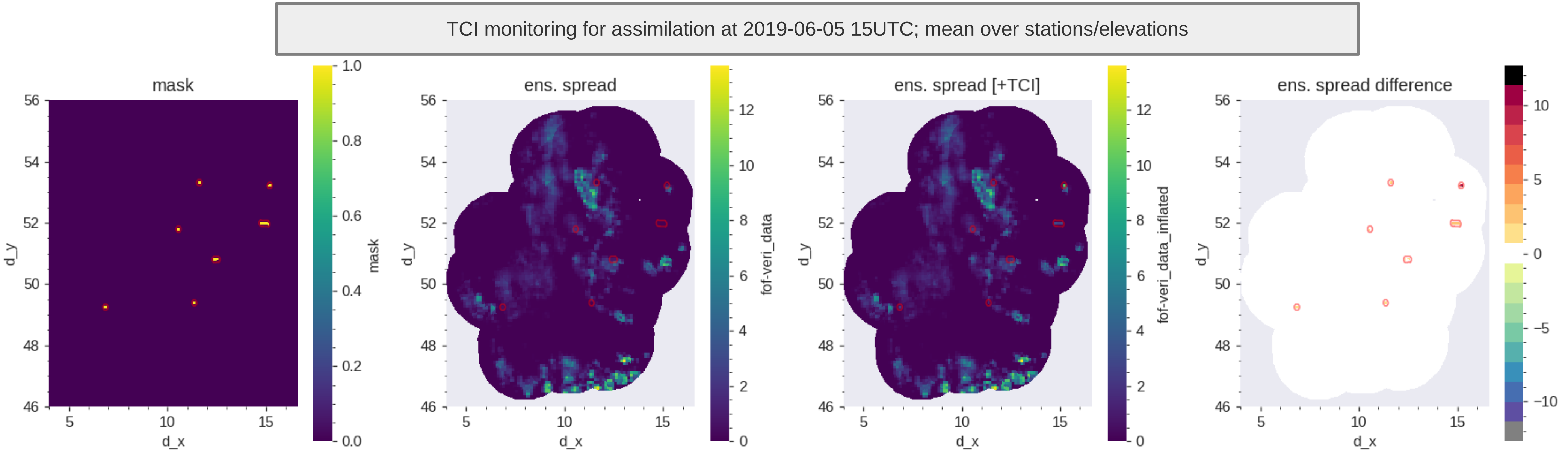
- currently working on "new" TCI based on **machine learning**
  - goal: ultra-short **prediction of newly emerging REFL** and its magnitude ("rough" estimate)
  - learn ICON model dynamics for convection
  - not living within ensemble pert. space!
- **predictors**: qv, T at several heights (+spatial mean/std)
- **target: temporal derivative of REFL** ΔZ (initially vertically integrated qr must be zero $\rightarrow$ no rain!)
- employed ML algorithms: KNN, Decision Tree
- much **more flexible approach** ($\rightarrow$ apply to CML data?)

TCI Case Study

TCI Case Study: Setup

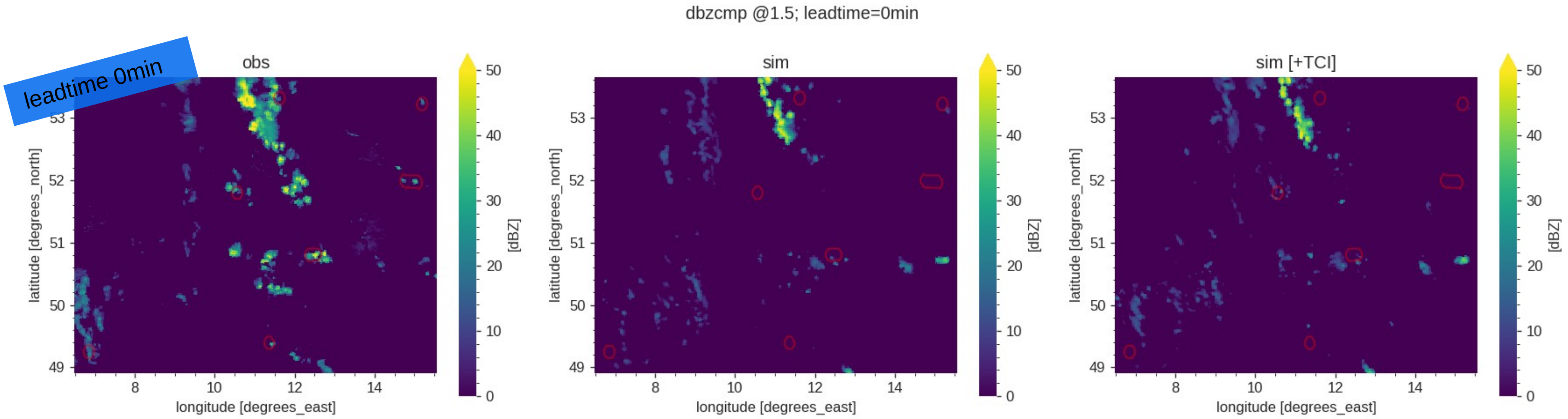
- set up two bacy experiment with/without application of TCI
- period: 2019-06-03 → 2019-06-10
- TCI is applied hourly at every assimilation step
  - TCI based on **simple linear model** (as shown previously)
  - TCI **applied to ALL radar data** over complete model domain
- Initiate main forecast runs every 6h (max. leadtime 6h)

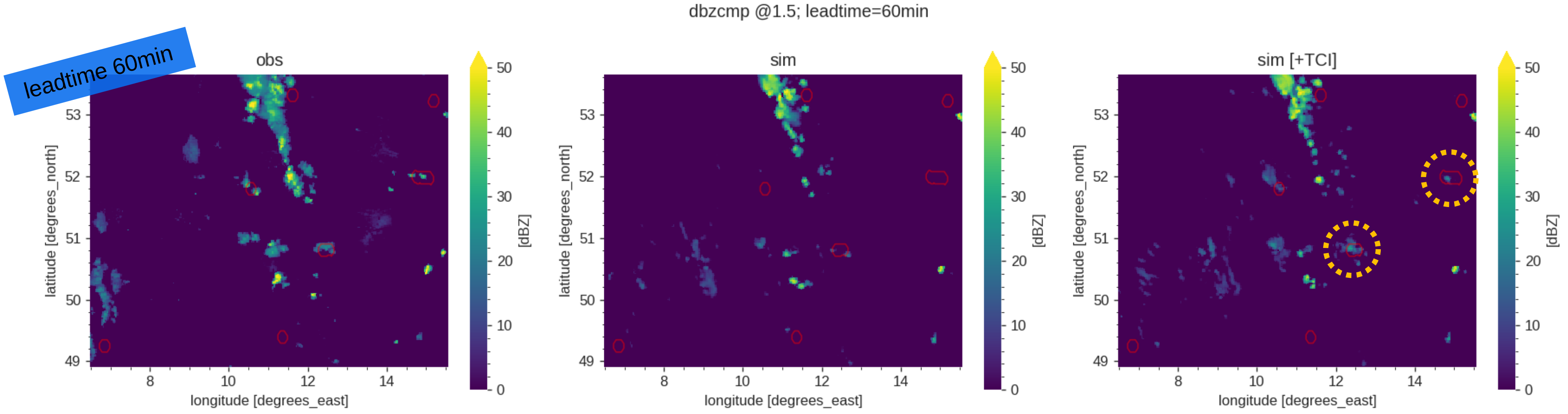
TCI Case Study: Monitoring



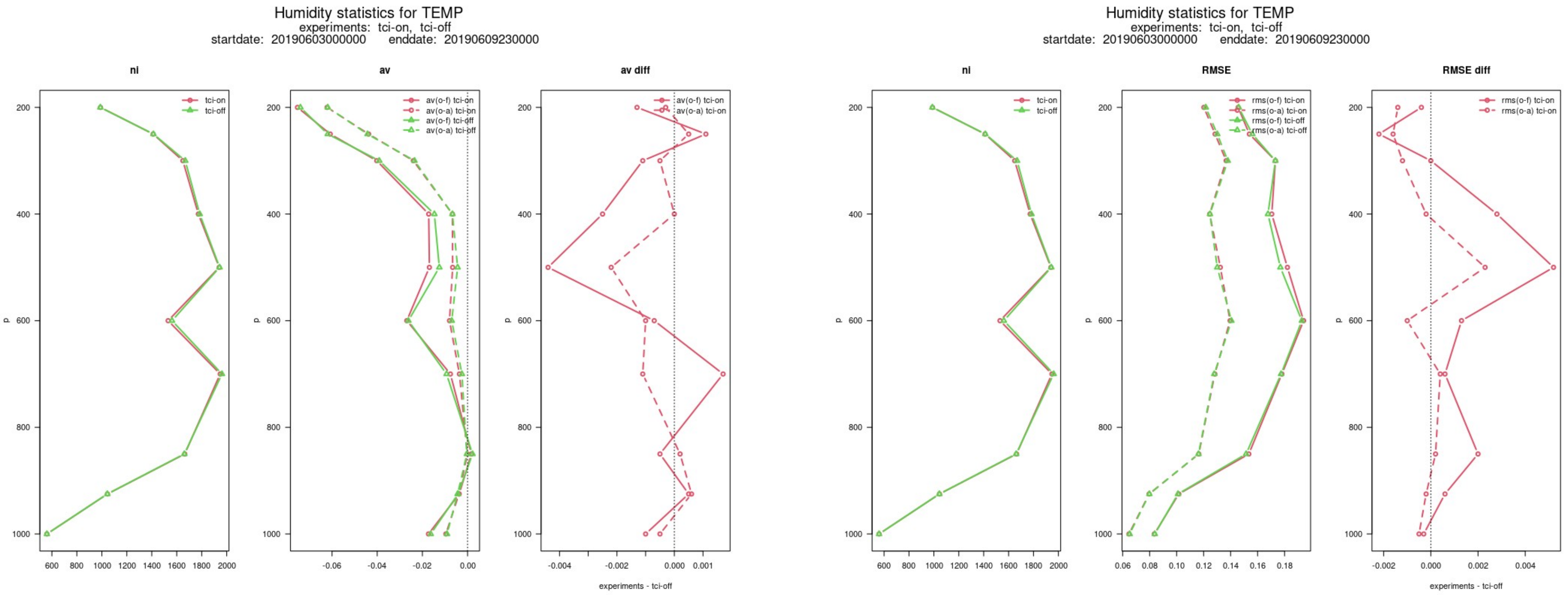
- “mask” shows if TCI got applied (also indicated with red contours)
- **main conditions** for specific obs.: vanishing ensemble spread/mean/det (+running average), sizeable observed REFL (Z>20), REFL height between 3000m and 4000m (all elevations)

TCI Case Study: Evolution of REFL (dbzcmp)



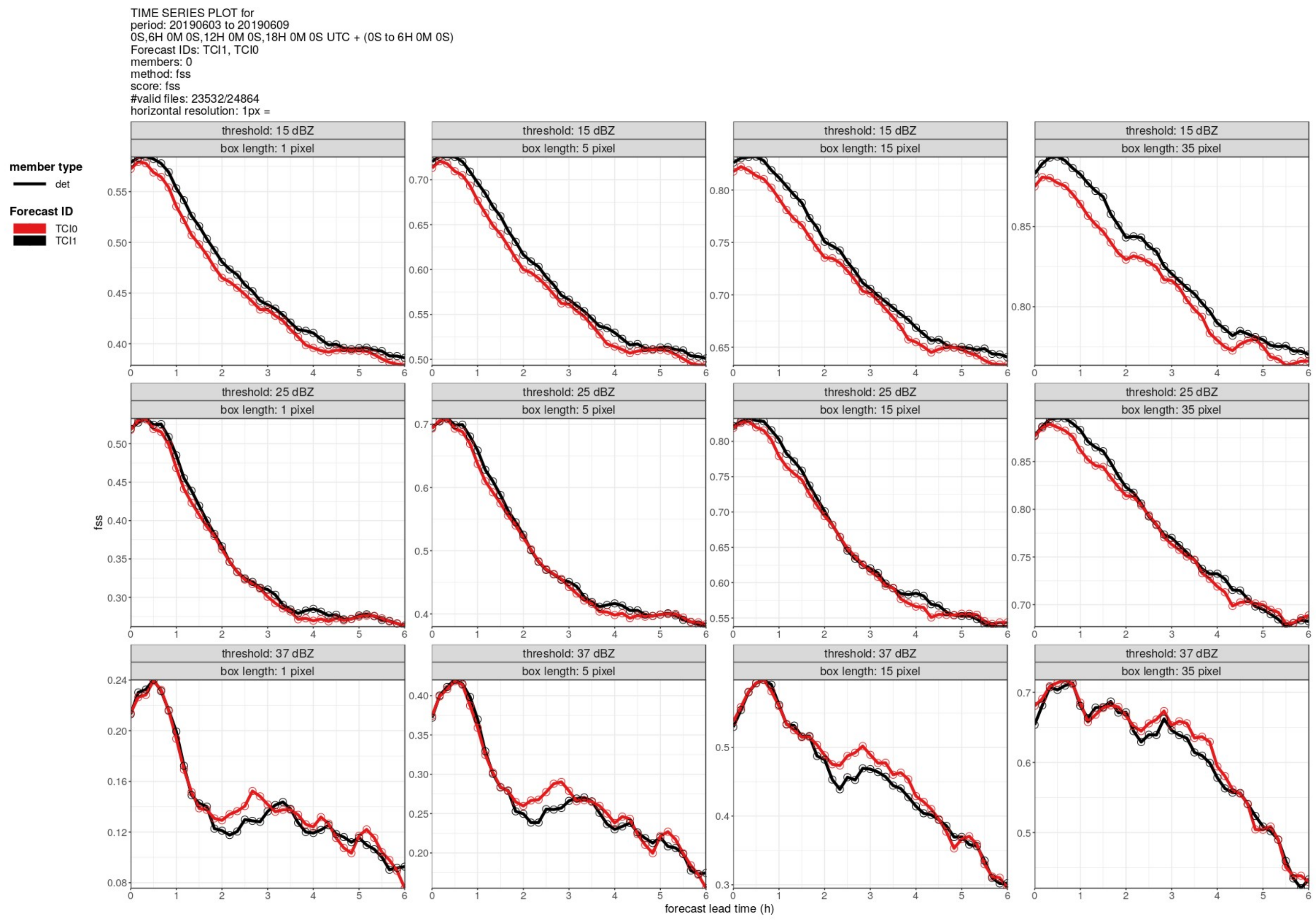


TCI Case Study: Obs. Err. Stat.



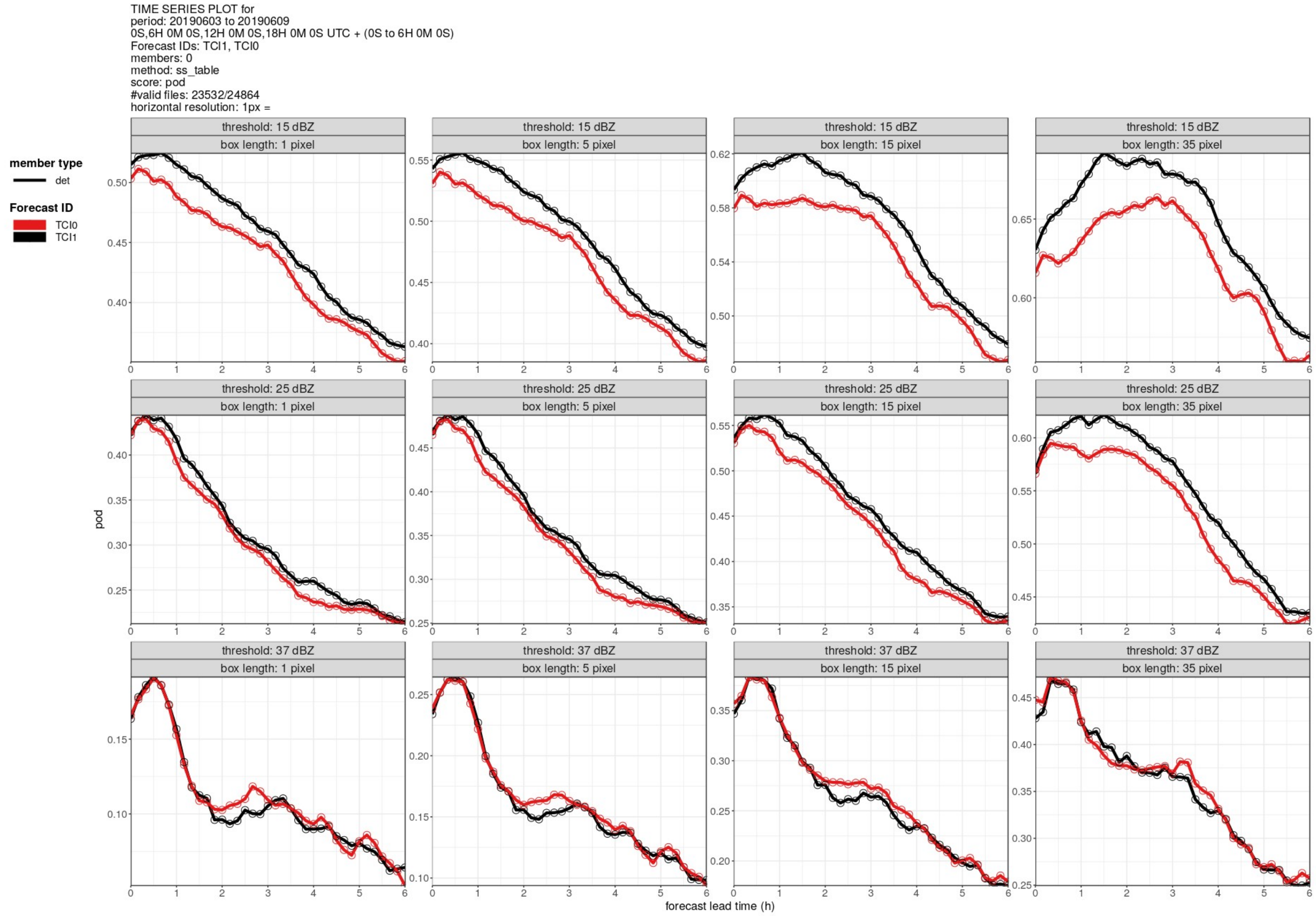
- **reduced negative impact on humidity** stat. (for AIREP/TEMP) w.r.t. previous TCI implementations
- T/RH/WIND/REFL stat. for AIREP/RADAR unobtrusive

TCI Case Study: FSS Verification



- Fractional Skill Scores (FSS) for dbzcmp from main forecast runs
- clear **positive impact** even after longer leadtimes!

TCI Case Study: POD Verification



- Probability of Detection (POD) for dbzcmp from main forecast runs
- clear **positive impact** even after longer leadtimes!

TCI: Summary

- overall, TCI results are promising
  - production of **“new” REFL cells** (consistent with observations)
  - positive impact on **fractional skill score** (w.r.t. dbzcmp)
  - **obs. err. stat.** results are unobtrusive (i.e. not too negative)
- next:
  - further studies necessary (verification of **longer time periods**)
  - continue work on **ML-based TCI**

Thank you for your attention!