

FILE ARCHIVE KIT (FARK)^{MET.NO, 2018}

INTRODUCTION

The process of generating **verification results** by co-locating **observation** and **model data**, typically requires 200 pieces of information. The FARK system provides a web-interface <http://fark.met.no> for the user to specify this information and organize the regular FARK production of verification result.

Use FARK if you need regular production of **verification results** and don't want to spend your time writing scripts.

FARK PRODUCTION

FARK first generates **indexed lists** of the **NetCDF** model files and **BUFR** observation files.

File indexes are usually sorted by the epoch-time.

A basic description of the file formats is available in the *appendix*.

Next, model fields are interpolated to relevant observation locations and time. This co-located data is written to a **table file** according to the specifications in a **plotting script**.









Finally, the **plotting script** produces the **verification plots**.

Verification results are found under:
`/lustre/storeA/project/nwp/fark`

WEB INTERFACE

All information necessary to generate verification results can be put into the FARK web interface. The web interface contains buttons designed to make it easier for the user to provide this information.

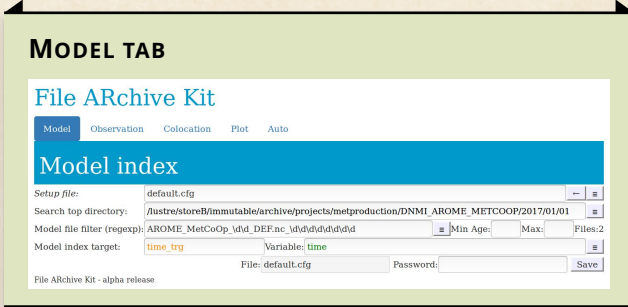
BUTTONS

Button	Description
	show alternatives
	move information
	delete table entry
	add table entry
	test process
	start process
	stop process
	save setup to server

Press the blue "tab title" to reload data from server.

The web interface is further divided into the following tabs: Model, Observation, Colocation, Plot and Auto.

MODEL TAB



In this tab you specify which NetCDF model files to index and which variable to use when sorting the **model index**. The index sorting variable is given a **model index target** name.

A **target** name is a 'short name' used to represent the model variable or observation parameter.

OBSERVATION TAB

Here you specify which BUFR observation files to index and the **expression** used to sort the **observation index**. The index expression is assigned a **target** name. The user must also define the **observation targets** that are used in the index expression. The observation targets point to specific positions in the BUFR sequence.

COLOCATION TAB

This is where you specified how the model and observation data should be matched. The colocation tab contains several tables.

MODEL TARGETS TABLE

Model target	Model variable/(dimension)	Minimum	Maximum
time_trg	time	midnight(-3)	midnight(0)
time_ana	forecast_reference_time	midnight(-3)	midnight(-1)-1
lon	longitude		
lat	latitude		
pres	air_pressure_at_sea_level		
geopot	surface_geopotential		
surpot	surface_potential		
t2m	air_temperature_2m		
rh	relative_humidity_2m		
wind_x	x_wind_10m		
wind_y	y_wind_10m		
precip_acc	precipitation_amount_acc		
precip_acc_m24	precipitation_amount_acc[time_trg:-86]		
hybrid	(hybrid2)		

The **model targets** table lists model variables and their target names. The (saved) **model index** target name is listed first.

Variables in red are not in the scanned model file.

You may also use a **dimension** in your **model**

target. In this case enter the dimension name surrounded by brackets instead of a variable name, for instance (ensemble_member).

MODEL VARIABLE OFFSET

precip_acc	precipitation_amount_acc
precip_acc_m24	precipitation_amount_acc[time_trg:-86]

If you want a model variable, say 24 hours earlier than the observation time, you can use an **offset**. The square brackets added after the variable name should contain the name of the model target that should be offset and the offset amount (seperated by colon). In this example model target precip_acc_m24 contains the accumulated precipitation 24 hours before the target precip_acc. *Matching rules*) should not be defined for offset model targets .

OBSERVATION TARGETS TABLE

Observation target	Position	Description	Minimum	Maximum
yy	5	4001 YEAR		
mm	6	4002 MONTH		
dd	7	4003 DAY		
hh	8	4004 HOUR		
mi	9	4005 MINUTE		
obs_time	sec1970(yy,mm,dd,hh,mi)			
obs_wmo	1	1001 WMO BLOCK N		
obs_id	2	1002 WMO STATION		
obs_sta	3	1015 STATION OR SI		
obs_lat	10	5001 LATITUDE (HIGI		
obs_lon	11	6001 LONGITUDE (HI		
obs_hgt	13	7031 HEIGHT OF BAR		
obs_pres	15	10051 PRESSURE REL		
obs_t2m	22	12101 TEMPERATURE/		
obs_d2m	23	12103 DEW-POINT TEI		
obs_rh	24	13003 RELATIVE HUM		
obs_wdir	W	11001 WIND DIRECTIO		
obs_wspd	W+1	11002 WIND SPEED		
obs_file	oid	Observation file		
obs_bufr	bid	BUFR message i		
obs_loc	lid	Location id		
obs_hybrid		hybrid0-duplicat	1	hybrid0

In the **observation targets** table the user can specify observation targets in the BUFR sequence. The targets already defined in the (saved) **observation index** are listed first. Additional observation targets needed to match the observation with the model fields are added here.

POSITION VARIABLES

In the example above obs_wdir uses a **position variable**, W instead of a number. The reason for this is that the wind speed happens to

appear after a *delayed replicator* in the BUFR sequence. The FARK system will search the BUFR sequence for the specified descriptor, 11001, and assign the corresponding position to the **position variable**, W. The **position variable** can be used in the position expressions of later observation targets, as we see in the example with obs_wspd with the position expression W+1.

Use **position variables** when you process radiosonde TEMP BUFR messages.

If only the descriptor is specified, the system will search the BUFR sequence for the next entry with the given descriptor.

INTERNAL VARIABLES

Internal variables are indicated as position variables in the position field, without any descriptor.

Position	Description
mid	model file index position
oid	observation file index position
bid	BUFR message number
sid	observation number in message
lid	location number in message

A location is identified using **oid**, **bid** and **lid**.

DUPLICATE LOCATION

If you want to compare the same observation location to several model fields, for instance different ensemble members, you need to duplicate the observation location. In this case you specify the min and max values and not the position nor descriptor (the max value may be a model dimension).

Duplicate locations if you need to process multiple **model ensemble members**.

The observation target obs_hybrid in the example above is an example of location duplication. The target obs_hybrid takes the value of the duplication index, i.e. 1,2,3,...,hybrid0.

MATCH RULES TABLE

The **match rules** table specifies how the model targets should match the observation targets.

MATCH RULES TABLE

Model target	Observation target expression
time_trg	obs_time
time_ana	
lon	obs_lon+obs_dlon
lat	obs_lat+obs_dlat
pres	obs_pres*0.01
geopot	
wind_x	
wind_y	
temp	
rh	

Model targets with blank observation target expressions are not used for matching. If insufficient matching rules are specified so that FARK can not determine how to interpolate a dimension used by a model target, FARK will average over that dimension (if it is small).

A location is discarded if a match rule has a target that is undefined.

DEFAULT TABLE

The **default** table is only visible if the observation index file has been set to <none>. The default table specifies how the model targets should match default values.

DEFAULT TABLE

time_trg	time_ana	lon	lat	z2m	rh	information
midnight(0)		10	60			
						+

FILTERS

Co-location takes a lot of computer resources, and it is therefore a good strategy to filter out unwanted data as early as possible in the data processing.

The observation **min** and **max** filters are applied immediately to discard locations when the BUFR messages are read from file.

The **observation filter** expression is applied when all locations in a BUFR message have been read from file, and this filter allows functions that apply to the whole message.

The **observation filter** expression can only be based on observation targets.

For instance `msgclosest` selects the location that produces an expression closest to a list

```
msgclosest (obs_pres*0.01,1000,500)
```

The model **min** and **max** filters are applied immediately when the location has been interpolated.

The **model filter** expression is applied when all relevant model fields have been interpolated to the observation location. This is the most expensive filter in terms of computer resources.

There is a debug option available for testing the built in filter functions. The debug expression can not accept any targets. Note that “blank” returns zero.

A location is rejected if a filter expression returns 0.

PLOT TAB

PLOT TAB

File ARchive Kit

Model Observation Colocation Plot Auto

Plot setup

Setup file: default.cfg

Output table file: /usr/share/ArProject/tpw/fark/default.table

Output prefix: /usr/share/ArProject/tpw/fark/default

Attributes:

Name	Value	Action
Version	1.0	
Id		
Set	set	
Colocation	default.cfg	
Legend	MEPS 2.5km	
Value	0	

Dataset:


Debug expression: name(precinct(10,60))

File: default.cfg Password: Norway Save

File ARchive Kit - alpha release

This is where you provide information requested by the plotting script. The user chooses plotting script in the **Category** field.

Verification results are placed on disk according to the **Output table file** and **Output prefix** paths. Use YY MM DD HH MI as wildcards in the output paths.

There are two tables in the plot tab. The **Attributes** table allows the user to specify attributes that apply for all the data, for instance titles, units and labels. Some attributes can only have fixed values, indicated by an action button , and some attributes are used to add attributes and columns.

A plotting script can compare different colocation datasets. The **Dataset** table assigns

every dataset an Id, Colocation file and legend, along with the column-expressions requested by the plotting script.

AUTO TAB

AUTO TAB					
Type	Setup file	% resolution	Min/max	Last	Info
model	default.cfg			2018-09-21T05:34:26Z	> ok (10s)
model	stari.cfg			2018-09-08T19:44:23Z	# short
model	starcfg_030d.cfg	daily		2018-09-10T04:05:00Z	> ok (4min)
model	starcfg_030d.cfg	daily		2018-09-10T04:04:14Z	> ok (2m13s)
model	starcfg_030d.cfg	daily		2018-09-10T04:06:26Z	> ok (1m19s)
model	starcfg_030d.cfg	daily		2018-09-10T04:06:26Z	> ok (1m19s)
model	starcfg_030d.cfg	daily		2018-09-10T04:06:26Z	> ok (1m19s)
obs	default.cfg			2018-08-21T05:34:24Z	> ok (21s)
obs	stari.cfg			2018-09-08T19:44:26Z	# no new data (1s)
obs	starcfg_030d.cfg	daily		2018-09-10T04:14:48Z	> ok (3m17s)
obs	starcfg_100d.cfg	daily		2018-09-10T04:18:05Z	> ok (2m18s)
obs	starcfg_030d.cfg	daily		2018-09-10T04:20:24Z	> ok (8s)
obs	starcfg_365d.cfg	weekly		2018-09-08T05:36:40Z	> ok (135m48s)
plot	default.cfg			2018-08-22T08:26:12Z	> ok (5m43s)
plot	starcfg_030d.cfg			2018-06-18T18:01:35Z	> ok (13m26s)
plot	starcfg_030d.cfg			2018-06-18T18:02:00Z	> ok (2h10m12s)
plot	starcfg_030d.cfg			2018-06-30T07:57:36Z	> ok (19h54m10s)
plot	stari.cfg			2018-09-08T19:44:26Z	# no data (0s)
plot	starcfg_030d.cfg			2018-09-08T19:44:26Z	> ok (14m43s)
plot	starcfg_030d.cfg	daily		2018-09-10T04:20:32Z	> ok (59s)
plot	starcfg_030d.cfg	daily		2018-09-10T04:21:32Z	> ok (1m56s)
plot	starcfg_030d.cfg	daily		2018-09-10T04:23:29Z	> ok (10m34s)
plot	starcfg_030d.cfg	daily		2018-09-10T04:34:04Z	# running (2h12m4s)
plot	starcfg_030d.cfg	daily		2018-09-09T14:44:01Z	> ok (3m21s)
plot	starcfg_030d.cfg	daily		2018-09-09T14:47:22Z	> ok (3h2m5s)

This is where you tell the computer to actually do some work. Available types of jobs are:

Type Description

model	maintain model index file,
obs	maintain observation index file,
coloc	debugging for advanced users,
plot	generate verification products.

Each job can be executed manually or according to a schedule.

PERFORMANCE

Co-locating large amounts of data takes a lot of resources, and it is quite easy to set up jobs that are not able to run successfully. Generating **table files** does not usually require too much memory but can take much processing time. However the **plotting script** will typically not be able to process a **table file** with more than 10 million entries without running out of memory (on Ares). This corresponds to 20 days of verification for northern Europe. Although FARK has all the features necessary for a full verification of operational forecast ensemble data using radiosonde data, this is option is probably not realistic from a computer resource perspective.

APPENDIX

NETCDF MODEL FILES

The NetCDF model files each contain a set of dimensions and variables, where each variable may have zero or more dimensions, for instance `latitude(x,y)` where `x` and `y` are dimensions. Note that some variables are accumulated, for instance `precipitation_amount_acc(...,time)`. Rain rate is calculated by differentiating this variable with respect to time.

BUFR OBSERVATION FILES

A BUFR observation file may contain many BUFR messages with different BUFR type and sub-type. Each BUFR message may contain many observations of the same type, for instance SYNOP or TEMP. An observation may further contain many locations, for instance a radiosonde TEMP observation may contain data from many different heights in the atmosphere.

BUFR SEQUENCE

BUFR observations with the same BUFR type and sub-type use the same **BUFR sequence**.

BUFR SEQUENCE EXAMPLE

1	: 1001 WMO BLOCK NUMBER ~ 2
2	: 1002 WMO STATION NUMBER ~ 981
3	: 1015 STATION OR SITE NAME ~ 1020
4	: 2001 TYPE OF STATION ~ 0
5	: 4001 YEAR ~ 2018
6	: 4002 MONTH ~ 1
7	: 4003 DAY ~ 1
8	: 4004 HOUR ~ 0
9	: 4005 MINUTE ~ 0
10	: 5001 LATITUDE (HIGH ACCURACY) ~ 59.77909
11	: 6001 LONGITUDE (HIGH ACCURACY) ~ 21.37479
12	: 7030 HEIGHT OF STATION GROUND ABOVE MEAN SEA LEVEL (SEE NOTE 3) ~ 6
13	: 7031 HEIGHT OF BAROMETER ABOVE MEAN SEA LEVEL (SEE NOTE 4) ~ 8.3
14	: 10004 PRESSURE ~ 99530
15	: 10051 PRESSURE REDUCED TO MEAN SEA LEVEL ~ 99640
16	: 10061 3-HOUR PRESSURE CHANGE ~ -210
17	: 10063 CHARACTERISTIC OF PRESSURE TENDENCY ~ 6
18	: 10062 24-HOUR PRESSURE CHANGE
19	: 7004 PRESSURE
20	: 10009 GEOPOTENTIAL HEIGHT
21	: 7032 HEIGHT OF SENSOR ABOVE LOCAL GROUND (OR DECK OF MARINE PLATFORM) ~ 2
22	: 12101 TEMPERATURE/DRY-BULB TEMPERATURE ~ 274.45
23	: 12103 DEW-POINT TEMPERATURE ~ 274.05
24	: 13003 RELATIVE HUMIDITY ~ 97

The BUFR sequence contains a **position**, **descriptor** and **value** for each parameter in the observation. The **descriptor** is used to identify the observation parameter, for instance pressure is identified by the descriptor 7004.

DELAYED REPLICATOR

DELAYED REPLICATOR

36	: 20012 CLOUD TYPE
37	: 31001 DELAYED DESCRIPTOR REPLICATION FACTOR ~ 1
38	: 8002 VERTICAL SIGNIFICANCE (SURFACE OBSERVATIONS)
39	: 20011 CLOUD AMOUNT
40	: 20012 CLOUD TYPE
41	: 20013 HEIGHT OF BASE OF CLOUD
42	: 31001 DELAYED DESCRIPTOR REPLICATION FACTOR ~ 0
43	: 8002 VERTICAL SIGNIFICANCE (SURFACE OBSERVATIONS)
44	: 20054 TRUE DIRECTION FROM WHICH CLOUDS ARE MOVING
45	: 8002 VERTICAL SIGNIFICANCE (SURFACE OBSERVATIONS)
46	: 20054 TRUE DIRECTION FROM WHICH CLOUDS ARE MOVING

A BUFR sequence may contain a **delayed replicator** (descriptor 31001), which will duplicate a sub-section of the BUFR sequence the specified number of times.


BUFR sequence positions after a **delayed replicator** are not “fixed”.

HOW TO...

CREATE A MODEL INDEX

Change focus to the “Model” tab.


1. MAKE SURE INDEX DOES NOT EXIST

Enter the name of your new index in the `setup file:` field, for instance `test.cfg`, and press  next to the field.




If your options include `<mkfile>` then the index does not exist. If the index already exists, the index setup will be loaded automatically. In this case press `<rmfile>` to delete the existing index. Remember to use the correct password when changing or deleting an existing index.




2. FIND A SUITABLE INDEX TO COPY

Enter the name of the index you wish to copy in the `setup file:` field. You may press  to navigate. The index setup will be loaded automatically when an existing index is specified.


3. CREATE NEW INDEX SETUP

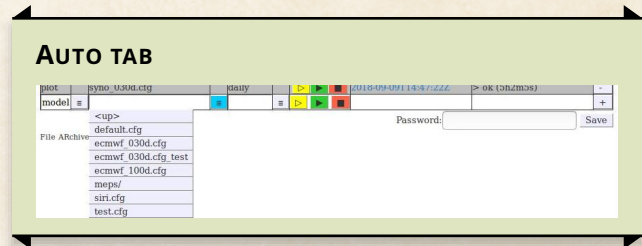
Enter the name of the new and non-existing index you wish to create in the `setup file:` field. Select the password that has to be provided when changing or deleting the new index. Press  to create the index setup on the server.


4. EDIT THE NEW INDEX SETUP


Edit the index setup fields. You may press  next to `Model file filter(regex)` to select which model file to scan. Variables in the scanned file appear as relevant options to other fields (when pressing ) , for instance `Variable::`. When you are satisfied, press  to save your settings.

5. CREATE THE MODEL INDEX

Switch to the `Auto` tab. Select the model type, and press  next to `Setup file`.







Your new model setup `test.cfg` should now be available as an option, select it. Finally press  to the right of your model setup file to create the model index itself.

If you know the password, you can schedule an automatic maintenance of your model index and press .

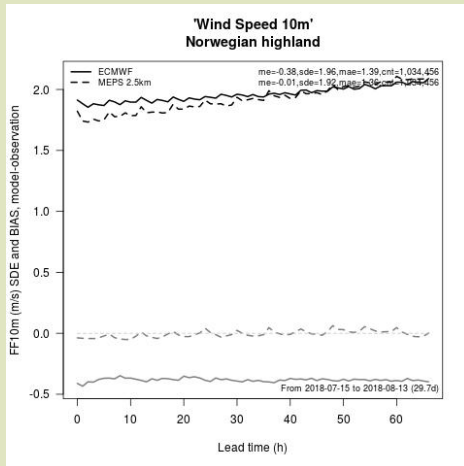
CREATE AN OBSERVATION INDEX

Creating the observation index setup is similar to creating the model index setup detailed above. However, editing the setup file is somewhat different and explained in some detail here.

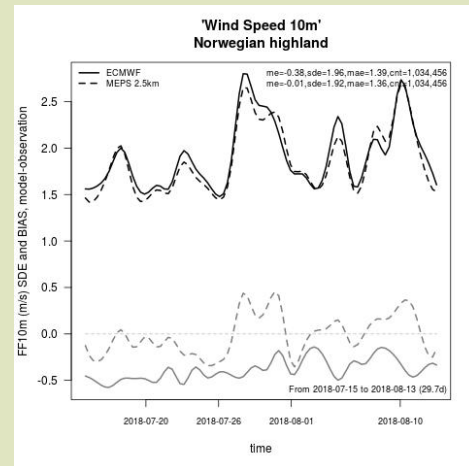
Change focus to the “Observation” tab. After creating your new observation index setup, you may press  next to `Obs file filter(regex)` to select which observation file to scan. The `BUFR type` and `Sub type` options are extracted from the scanned file, along with their associated `BUFR sequences`. Define the targets that you need in your observation index expression by entering values in the bottom row of the `observation targets` table, and by pressing the  to the right. You may remove a row by pressing  to the right of the respective row. The values of the removed row are put in the bottom row. When you are satisfied, press  to save your settings.

EXAMPLES

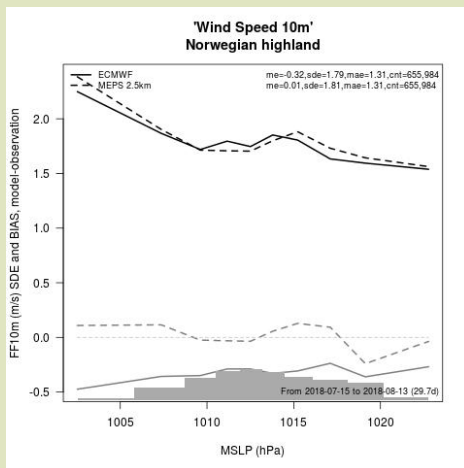
FF10M SCORE



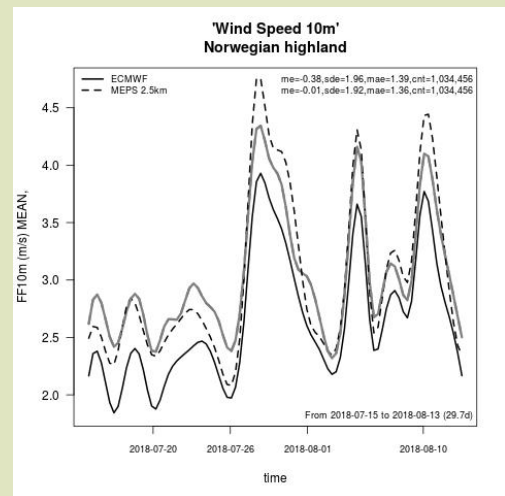
FF10M SDE SERIES



FF10M VS MSLP



FF10M MEAN SERIES



FF10M PROFILE

