

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 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 that will be used in the verification.

File indexes are usually sorted by the epoch-time.

A basic description of the NetCDF and BUFR 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

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

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

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.

When a model file is being indexed, FARK will pre-scan the range of the variables with small array size.

OBSERVATION TAB

OBSERVATION TAB

File ARchive Kit

Model Observation Colocation Plot Auto

Observation index

Setup file: default.cfg

Search top directory: /Astrus/store/B/immutable/archive/projects/metproduction/DNMI_OBS_BUFIR/

Obs file filter (regex): syno_\\d{4}\\d{4}\\d{4}\\d{4}.bufir\$ Min Age: Max: 3 Files?:

BUFR table path: /metfark/bufrtables/

Data filter:

Observation targets:

Observation target	Position	ID	Description	Info
yy	5	I	4001	YEAR
mm	6	I	4002	MONTH
dd	7	I	4003	DAY
hh	8	I	4004	HOUR
mi	9	I	4005	MINUTE
		#		

Observation index target: obs_time Expression: sec1970(yy,mm,dd,hh,mi)

File: default.cfg Password: Save

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.

Files match if their index target range overlap.

COLOCATION TAB

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

MODEL TARGETS TABLE

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.

The **model target** can also be a **dimension**. 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 (separated 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 TARGETS TABLE

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

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`. FARK will repeat the search for position variables until the end of the BUFR message, giving a new location for every match found.

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	z_m	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 plot log in the `Auto` tab contains summaries of how different filters and the quality control removed data.

MODEL TARGET FILTER

Model target	Model variable/(dimension)	Minimum	Maximum
time_trg	time	midnight(-3)	midnight(-1)
time_ana	forecast_reference_time	midnight(-3)-1	midnight(-3)+1

If a model target has **min** and **max** limits that are outside the pre-scanned range, the model file will immediately be rejected. Below is an example of the resulting plot log summary.

PLOT LOG (MODEL FILE FILTER)

```
model_printF Model files: 40
model_printF Model index filter: -3 ( 7.50%)
model_printF 'time_ana' filter: -36 ( 97.30%)
model_printF
```

The observation **min** and **max** filters are applied as the BUFR messages are read from the observation file.

The **observation filter** expression is applied to all locations in a BUFR message at once, and has functions like

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

that selects the locations with the expression, $obs_pres \times 0.01$, closest to the given list, 1000, 500.

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

The model **min** and **max** filters are re-applied immediately after the corresponding model field has been interpolated to the observation location.

The **location 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: /lustre/storeA/project/nwp/fark/default.table

Output prefix: /lustre/storeA/project/nwp/fark/default/

Attributes:

Name	Value
n	1
Version	1.0
Action	+
Id	1
Set	set
Colocation file	default.cfg
Legend	MEPS 2.5km
values1	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 . Special attributes can be used to enumerate other 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 columns requested by the plotting script.

AUTO TAB

AUTO TAB

Type	Setup file	Sc location	Min/max	Last	Info
model	default.cfg			2018-08-21T05:34:262	> ok (10s)
model	test.cfg			2018-09-08T10:44:232	# short
model	scenmf_030d.cfg	daily		2018-09-10T04:05:002	> ok (4m0s)
model	meps/members0_030d.cfg	daily		2018-09-10T04:04:142	> ok (2m13s)
model	meps/members0_100d.cfg	daily		2018-09-10T04:06:282	> ok (1m19s)
model	meps/members0_365d.cfg	weekly		2018-09-08T05:09:022	> ok (3m37s)
obs	default.cfg			2018-08-21T05:34:242	> ok (21s)
obs	scrl.cfg			2018-09-06T10:44:262	# no new data (1s)
obs	starc/syno_030d.cfg	daily		2018-09-10T04:14:462	> ok (3m17s)
obs	starc/syno_100d.cfg	daily		2018-09-10T04:18:052	> ok (2m18s)
obs	starc/temp_030d.cfg	daily		2018-09-10T04:20:242	> ok (0s)
obs	starc/syno_365d.cfg	weekly		2018-09-08T05:36:402	> ok (13m40s)
plot	default.cfg			2018-08-22T08:26:122	> ok (5m43s)
plot	meps_syno_003d.cfg			2018-06-18T18:01:352	> ok (13m26s)
plot	meps_syno_030d.cfg			2018-06-18T18:02:002	> ok (2h10m12s)
plot	meps_syno_365d.cfg			2018-06-30T07:57:382	> ok (19m54m10s)
plot	scrl.cfg			2018-09-06T10:54:252	# no data (0s)
plot	syno_003d.cfg			2018-09-06T11:14:492	> ok (14m43s)
plot	scenmf_syno_002d.cfg	daily		2018-09-10T04:20:322	> ok (59s)
plot	meps_syno_002d.cfg	daily		2018-09-10T04:21:322	> ok (1m56s)
plot	meps/temp_002d.cfg	daily		2018-09-10T04:23:292	> ok (10m34s)
plot	meps/temp_030d.cfg	daily		2018-09-10T04:34:042	# running (2h12m4s)
plot	syno_002d.cfg	daily		2018-09-09T14:44:012	> ok (3m21s)
plot	syno_030d.cfg	daily		2018-09-09T14:47:222	> 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	<i>debugging for advanced users,</i>
plot	generate verification products.

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


The Last column has links to log files from the last **successful** and **un-successful** runs.

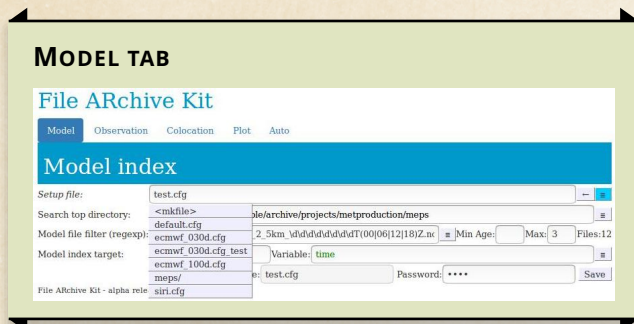
HOW TO...


...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. Finally, press  again and “forget” the index on your client by pressing <fgfile>.




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

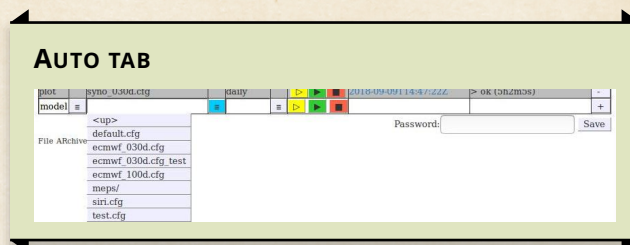
Enter the name of the new and non-existing index you wish to create in the setup file: field. Select the password that has to be provided when changing or deleting the new index. Press  to create the index setup on the server.



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

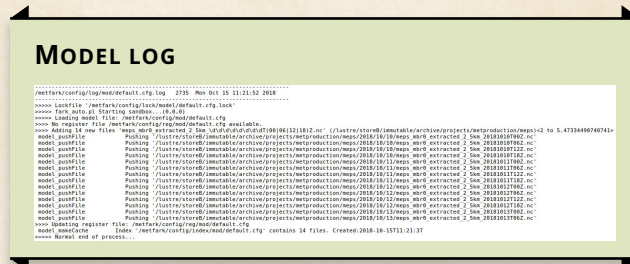
Switch to the Auto tab. The last row in the table allows you to enter new jobs. Select the model type, and press  next to Setup file.



Your new model setup test.cfg should now be available as an option, select it. Enter the password and press  to permanently add your job to the table. Finally press  to the right of your model setup file to create the model index itself.

6. CHECK THE LOG

When the job is finished, check the log. You may view the log by pressing the link in the Last column in the Auto tab table.




The log lists the model-files added to the index together with the size and modification date of the index.

The log of a successful run contains
Normal end of process....

Note that several processes flush output to the log, giving it a somewhat “shuffled” appearance.




...CREATE AN OBSERVATION INDEX


Change focus to the “Observation” tab. Creating the observation index setup is similar to creating the model index setup as indicated above. However, editing the setup file is explained in some detail here.

After creating your new observation index setup, you may press  next to Obs file filter(regex) to select which observation file to scan.

BUFR TYPE AND SUB TYPE

BUFR type: 0		Sub type: 0		Info:	
Observation	0 Surface (8493)	Descriptor	0 : (7847)		
yy	1 Surface (1403)	† 4001	1 : synop land (185)		-
mm	6	† 4002	2 : synop land record 2 (50)		-
dd	7	† 4003	20 : (163)		-
hh	8	† 4004	150 : (39)		-
			151 : (6)		-
			180 : (74)		-
mi	9	† 4005	181 : (38)		-
			255 : (91)		-

The BUFR type and Sub type alternatives 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.

You may now add and run the job in the **Auto** tab by pressing . View the log-file by following the link in the **Last** column. The log lists the observation-files added to the index together with the size and modification date of the index.

[illegible]

...CREATE A COLOCATION SETUP

Change focus to the “Colocation” tab. Creating the colocation setup is similar to creating the model index setup as indicated above.


In the colocation setup you define model- and observation-targets, and how they should match. Make sure you have scanned a model-file (Model tab) and observation-file (Observation tab) so that model variables and observation BUFR sequences can be listed where relevant.


Add the model and observation targets needed to match an observation with the model, for instance `time`, `latitude` and `longitude`. Add targets for the verification parameters, for instance `pressure` and `temperature`. Finally define the matching expressions. When you are satisfied, press **Save** to save your settings.

It is not recommended to run a colocation-job in the `Auto` tab since this is an extremely slow process that generates an enormous amount of output.

...CREATE VERIFICATION PLOTS

Before you can make verification plots, you must have created a model and observation index, and a colocation *setup*.

Next you need to create a plot setup. Change focus to the “Plot” tab. Creating the plot setup is similar to creating the model index setup as indicated above. Remember to change the output-fields if you copied the plot setup. When you are satisfied, press  to save your plot setup.

Next, switch to the `Auto` tab and add your plot-job to the table. To actually create the verification plots, press  next to your plot job and follow the link in the `Last` column to see the log produced by the plot job.

[illegible]

Check if your indexes are up-to-date if your time-dependent filters have removed all data.

The script command (usually listed at the end of the log), contains the root output directory for the verification results, which you specified in the `Plot` tab. The next section shows some examples of verification plots that the “synop” verification script typically would place there.

Here are some examples of verification plots.

'Wind Speed 10m'
Norwegian highland

me=-0.38, sde=1.96, mae=1.39, cnt=1,034,456
me=-0.01, sde=1.92, mae=1.03, cnt=1,034,456

From 2018-07-15 to 2018-08-13 (29.7d)

Wind Speed 10m'

altitude (m)

FF10m (m/s) SDE and BIAS, model-observation

ECMWF (Norwegian highland)
MEPS 2.5km (Norwegian highland)
ECMWF (Norwegian lowland)
MEPS 2.5km (Norwegian lowland)

me=-0.38,sde=1.96,mae=1.39,ct=1.034,456
me=0.01,sde=1.92,mae=1.36,ct=1.034,456
me=-0.36,sde=1.99,mae=1.51,ct=1.041,465
me=-0.17,sde=2.1,mae=1.56,ct=1.041,465

From 2018-07-15 to 2018-09-15 (29.76)

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

30	: 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”.

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.