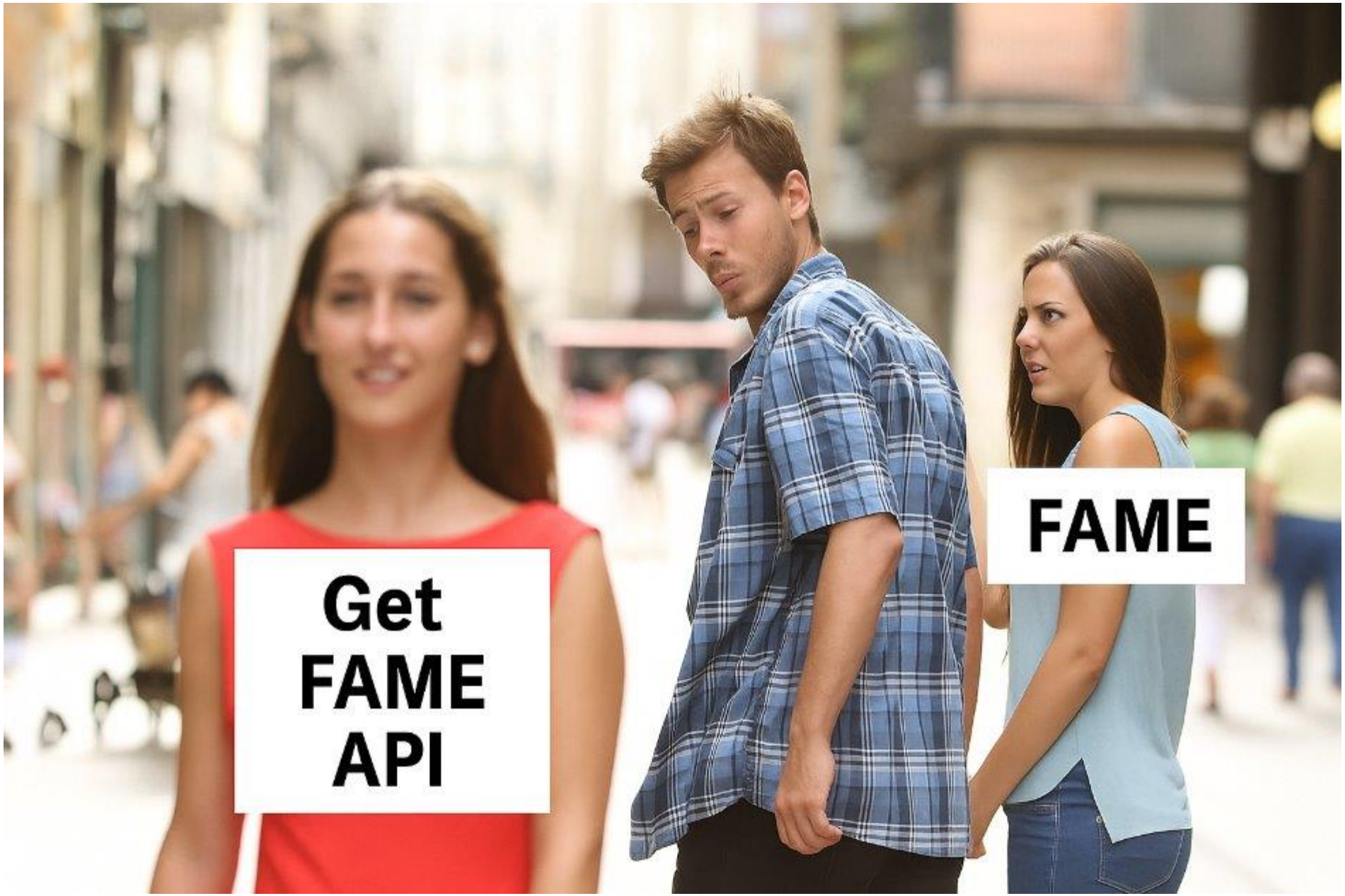

Get
FAME
API
FAME

getfame

-n names

-s series

-e expression

A JSON API 2025 to use from R Python and more..

Erik.Soberg@ssb.no

2025 Supports series & formulas with identical series names in different FAME databases. Evaluate & aggregate from several databases

1. getfame -n gets FAME metadata

sl-fame-1

```
sl-fame-1:/ssb/bruker/refertid/system/myfame/api> getfame -n "$REFERTID/data/kpi_publ; $REFERTID/data/kpi_erik" "total.ipr?;KO1111_?.IPR"
[{"GetFAME_Json_Api": "Erik.Soberg@ssb.no",
"Version": "Hack-20251019",
"Executed": "2025-09-20T09:33:48",
"Famever": "11.53",
"Database": "/ssb/bruker/refertid/data/kpi_publ; /ssb/bruker/refertid/data/kpi_erik",
"Openas": "KPI_PUBL, KPI_ERIK2",
"Result": "$HOME/.GetFAME/getfamenames.json",
"Wildcard": "TOTAL.IPR?;KO1111_?.IPR",
"Found": 10,
"Notfound": 0,
"Missing": "",
"Series":[
{"Name":"KO1111_11111.IPR","Db":"KPI_ERIK2","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Ris_indeks pris","Created":"2017-01-18T
{"Name":"TOTAL.IPR","Db":"KPI_ERIK2","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Totalindeks_indeks pris","Created":"2017-01-18
{"Name":"TOTAL.IPR.A","Db":"KPI_ERIK2","Class":"SERIES","Observed":"AVERAGED","Freq":"ANNUAL","Desc":"Totalindeks_indeks pris_Ersgjsn","Created":"2
{"Name":"TOTAL.IPR.G","Db":"KPI_ERIK2","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Totalindeks_Trend(prog1)","Created":"2025-06-
{"Name":"TOTAL.IPR.S","Db":"KPI_ERIK2","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Totalindeks_Sesongjustert(prog1)","Created":
{"Name":"KO1111_11111.IPR","Db":"KPI_PUBL","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Ris_indeks pris","Created":"2017-01-18T1
{"Name":"TOTAL.IPR","Db":"KPI_PUBL","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Totalindeks_indeks pris","Created":"2017-01-18T
{"Name":"TOTAL.IPR.A","Db":"KPI_PUBL","Class":"SERIES","Observed":"AVERAGED","Freq":"ANNUAL","Desc":"Totalindeks_indeks pris_Ersgjsn","Created":"20
{"Name":"TOTAL.IPR.G","Db":"KPI_PUBL","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Totalindeks_Trend(prog1)","Created":"2025-09-
{"Name":"TOTAL.IPR.S","Db":"KPI_PUBL","Class":"SERIES","Observed":"AVERAGED","Freq":"MONTHLY","Desc":"Totalindeks_Sesongjustert(prog1)","Created":"
"ErrorCount": 0 ,
"Errors": [],
"Elapsed_time_in_seconds":0.014
  } ]
```

getfame –n identical databasenames & identical seriesnames

..

sl-fame-1

```
sl-fame-1:/ssb/bruker/refertid/system/myfame/api> getfame -n "$REFERTID/data/testapi.db; $REFERTID/data/hack25/testapi.db" "?"
[{"GetFAME_Json_Api": "Erik.Soberg@ssb.no",
"Version": "Hack-20251019",
"Executed": "2025-09-20T10:17:48",
"Famever": "11.53",
"Database": "/ssb/bruker/refertid/data/testapi.db; /ssb/bruker/refertid/data/hack25/testapi.db",
"Openas": "TESTAPI, TESTAPI2",
"Result": "$HOME/.GetFAME/getfamenames.json",
"Wildcard": "?",
"Found": 10,
"Notfound": 0,
"Missing": "",
"Series":[
{"Name":"ERIK","Db":"TESTAPI","Class":"SERIES","Observed":"SUMMED","Freq":"MONTHLY","Desc":"secript of erik soeb WOW","Created":"2024-09-09T22:21:26","Upd
{"Name":"EXTRA","Db":"TESTAPI","Class":"SERIES","Observed":"SUMMED","Freq":"ANNUAL","Desc":"extraextras","Created":"2025-05-30T13:12:53","Updated":"2025-0
{"Name":"NEWANN","Db":"TESTAPI","Class":"SERIES","Observed":"SUMMED","Freq":"ANNUAL","Desc":"","Created":"2025-06-01T15:55:33","Updated":"2025-06-01T15:55
{"Name":"TEST.ANN","Db":"TESTAPI","Class":"SERIES","Observed":"SUMMED","Freq":"ANNUAL","Desc":"","Created":"2024-06-16T21:53:09","Updated":"2025-05-30T11:
{"Name":"TEST.MON","Db":"TESTAPI","Class":"SERIES","Observed":"SUMMED","Freq":"MONTHLY","Desc":"","Created":"2024-06-16T21:54:14","Updated":"2024-06-16T22
{"Name":"TEST.MON.F","Db":"TESTAPI","Class":"FORMULA","Observed":"TEST.MON *10","Freq":"NC","Desc":"","Created":"2024-06-16T21:55:16","Updated":"2024-06-1
{"Name":"ERIK","Db":"TESTAPI2","Class":"SERIES","Observed":"SUMMED","Freq":"MONTHLY","Desc":"dEScription of erik","Created":"2024-09-09T22:21:26","Updated
{"Name":"TEST.ANN","Db":"TESTAPI2","Class":"SERIES","Observed":"SUMMED","Freq":"ANNUAL","Desc":"","Created":"2024-06-16T21:53:09","Updated":"2024-06-17T15
{"Name":"TEST.MON","Db":"TESTAPI2","Class":"SERIES","Observed":"SUMMED","Freq":"MONTHLY","Desc":"mytest","Created":"2024-06-16T21:54:14","Updated":"2025-0
{"Name":"TEST.MON.F","Db":"TESTAPI2","Class":"FORMULA","Observed":"TEST.MON *10","Freq":"NC","Desc":"","Created":"2024-06-16T21:55:16","Updated":"2024-06-
"ErrorCount": 0 ,
"Errors": [],
"Elapsed_time_in_seconds":0.009
  } ]
```

getfame -n

Combine with linux commands to find descriptions / series, with correct or incorrect definitions
The command below lists series, given 2 wildcards, but show only the one(s) containing the text «SUM»

2. getfame -s gets observations from different databases.

sl-fame-1

```
sl-fame-1:/ssb/bruker/refertid/system/myfame/api> getfame -s "$REFERTID/data/kpi_publ; $HOME/kpi.db" "KO2.IPR;KO1.IPR" "freq m; date 2025:1 to 2025:03"
[{"GetFAME_Json_Api": "Erik.Soberg@ssb.no",
"Version": "Hack-20251019",
"Executed": "2025-09-20T10:35:13",
"Famever": "11.53",
"Database": "/ssb/bruker/refertid/data/kpi_publ; /ssb/bruker/rsb/kpi.db",
"Openas": "KPI_PUBL, KPI2",
"Result": "$HOME/.GetFAME/getfameseries.json",
"Wildcard": "KO2.IPR;KO1.IPR",
"Found": 4,
"Notfound": 0,
"Missing": "",
"Series": [
{"Name": "KO1.IPR",
"Db": "KPI2",
"Desc": "Matvarer og alkoholfrie drikkevarer_indeks pris",
"Daterange": "2025:01 TO 2025:03",
"Frequency": "MONTHLY",
"Observations":[
{"Date":"2025-01-01", "Value":135.4, "Epo":[1735689600000, 135.4]}
]  }     ,
{"Name": "KO2.IPR",
"Db": "KPI2",
"Desc": "Alkoholholdige drikkevarer og tobakk_indeks pris",
"Daterange": "2025:01 TO 2025:03",
"Frequency": "MONTHLY",
"Observations":[
{"Date":"2025-01-01", "Value":130.7, "Epo":[1735689600000, 130.7]}
]  }     ,
{"Name": "KO1.IPR",
"Db": "KPI_PUBL",
"Desc": "Matvarer og alkoholfrie drikkevarer_indeks pris",
"Daterange": "2025:01 TO 2025:03",
"Frequency": "MONTHLY",
"Observations":[
{"Date":"2025-01-01", "Value":135.4, "Epo":[1735689600000, 135.4]},
{"Date":"2025-02-01", "Value":137.8, "Epo":[1738368000000, 137.8]},
{"Date":"2025-03-01", "Value":136.8, "Epo":[1740787200000, 136.8]}
]  }     ,
{"Name": "KO2.IPR",
"Db": "KPI_PUBL",
"Desc": "Alkoholholdige drikkevarer og tobakk_indeks pris",
"Daterange": "2025:01 TO 2025:03",
"Frequency": "MONTHLY",
"Observations":[
{"Date":"2025-01-01", "Value":130.7, "Epo":[1735689600000, 130.7]},
{"Date":"2025-02-01", "Value":131.5, "Epo":[1738368000000, 131.5]},
{"Date":"2025-03-01", "Value":131.3, "Epo":[1740787200000, 131.3]}
]  }     ],
"ErrorCount": 0 ,
"Errors": [],
"Elapsed_time_in_seconds":0.005
  } ]
```

getfame -s getfameseries samples

$REFERTID/system/myfame/api/getfame -s " /ssb/bruker/refertid/data/kpi_publ.db " "total.ipr"

getfame -s " /ssb/bruker/refertid/data/kpi_publ.db " "total.ipr; K0?IPR " "date 2024 "

getfame -s " /ssb/bruker/refertid/data/kpi_publ.db " "total.ipr" "freq m; date thisday(m)-5 to *"

getfame -s "$REFERTID/data/fornavn.db " "?ERIK;KRISTIN;JIM?" "date 2010 to 2012 "

getfame -s " /ssb/bruker/refertid/data/fornavn.db"   "?JAN?" "date 2000 to 2005 "

getfame -s "pi1.db;cpi2.db;cpi_form.db" "Total.ipr? " " convert on; freq q;date 2025 ; deci 2"

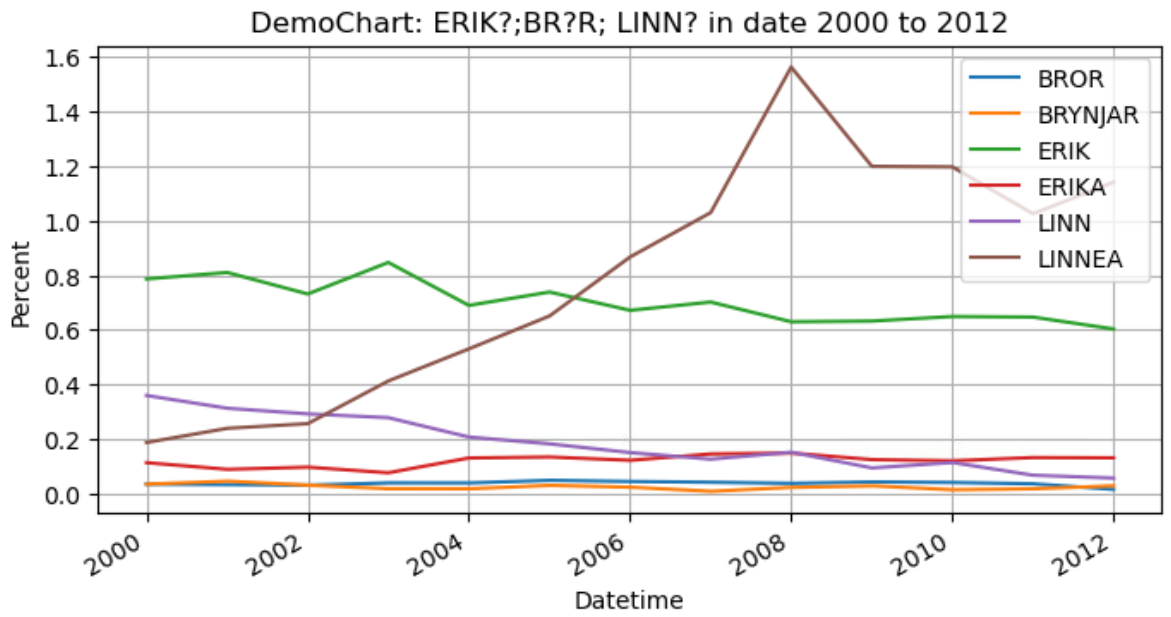
getfame –s

jupyter with python:

Chartest.ipynb | PySample.ipynb | r_2series.ipynb | Rfameexpr.ipynb

Code

```
# Plot each series separately we use epoch in upper leagues
for series_name in df_all['Series'].unique():
    df_series = df_all[df_all['Series'] == series_name]
    ax.plot(df_series['Epoch'], df_series['Value'], label=series_name)


# Use AutoDateLocator and AutoDateFormatter
locator = mdates.AutoDateLocator()
formatter = mdates.ConciseDateFormatter(locator)

ax.xaxis.set_major_locator(locator)
ax.xaxis.set_major_formatter(formatter)

# Auto-format the x-axis dates
fig.autofmt_xdate()

# Adding labels and title
plt.xlabel('Datetime')
plt.ylabel('Percent')
plt.title(f"DemoChart: {famesoek} in {famedato}")
plt.grid(True)
plt.legend()

plt.show()
```



3. getfame -e

getfameexpression using FAME as the engine for calculation

advanced mode

- Data-observations, from FAME database(s) given one or more FAME-**expression:**

```
getfame -e  "$REFERTID/data/fornavn.db "  "mave(JIMMY,2); mave(LINN,3)" "date 2000 to 2010"
getfame -e  "$REFERTID/data/fornavn.db "  "Lsum(ERIK,EIRIK)"  "date 2000 to *"
getfame -e  "$REFERTID/data/fornavn.db "  «SOLVEIG+PETER"  "date 2000 to 2012"
getfame –e "$REFERTID/data/kpi_publ.db;mycpi.db " "convert(total.ipr,annual,constant) "  "date *; deci 1 "
getfame -e  "$REFERTID/data/kpi_publ.db; mycpi.db "  "PCT(mycpi'K09.IPR)"  "date 2025; deci 1 "
getfame -e   "cpi1.db;cpi2.db;cpi_form.db" "cpi1'Total.ipr"  "date 2025 ; deci 2"
```

Be aware to **double quote arguments** when they contain special char like : , (‘ ; or more expressions are used.

Fame names are NOT case sensitive.

getfame –e fameexpression example

sl-fame-1

```
sl-fame-1:/ssb/bruker/refertid/system/myfame/api> getfame -e "$REFERTID/data/kpi_publ" "pct(total.ipr);ytypct(total.ipr)" "freq m; date 2025 to *"
[{"GetFAME_Json_Api": "Erik.Soberg@ssb.no",
"Version": "Hack-20251019",
"Executed": "2025-09-20T17:39:58",
"Famever": "11.53",
"Database": "/ssb/bruker/refertid/data/kpi_publ",
"Openas": "KPI_PUBL",
"Result": "$HOME/.GetFAME/getfameexpr.json",
"Series": [
{"Name": "PCT(TOTAL.IPR)",
"Db": "KPI_PUBL",
"Desc": "pct(total.ipr)",
"Daterange": "2025:01 TO *",
"Frequency": "MONTHLY",
"Observations":[
{"Date":"2025-01-01", "Value":0.2225519, "Epo":[1735689600000, 0.2225519]},
{"Date":"2025-02-01", "Value":1.406366, "Epo":[1738368000000, 1.406366]},
{"Date":"2025-03-01", "Value":-0.6569343, "Epo":[1740787200000, -0.6569343]},
{"Date":"2025-04-01", "Value":0.6612785, "Epo":[1743465600000, 0.6612785]},
{"Date":"2025-05-01", "Value":0.3649635, "Epo":[1746057600000, 0.3649635]},
{"Date":"2025-06-01", "Value":0.2181818, "Epo":[1748736000000, 0.2181818]},
{"Date":"2025-07-01", "Value":0.7982583, "Epo":[1751328000000, 0.7982583]},
{"Date":"2025-08-01", "Value":-0.6479482, "Epo":[1754006400000, -0.6479482]}
]  }
,
{"Name": "YTYPCT(TOTAL.IPR)",
"Db": "KPI_PUBL",
"Desc": "ytypct(total.ipr)",
"Daterange": "2025:01 TO *",
"Frequency": "MONTHLY",
"Observations":[
{"Date":"2025-01-01", "Value":2.348485, "Epo":[1735689600000, 2.348485]},
{"Date":"2025-02-01", "Value":3.552532, "Epo":[1738368000000, 3.552532]},
{"Date":"2025-03-01", "Value":2.639517, "Epo":[1740787200000, 2.639517]},
{"Date":"2025-04-01", "Value":2.468212, "Epo":[1743465600000, 2.468212]},
{"Date":"2025-05-01", "Value":2.996255, "Epo":[1746057600000, 2.996255]},
{"Date":"2025-06-01", "Value":2.989537, "Epo":[1748736000000, 2.989537]},
{"Date":"2025-07-01", "Value":3.271375, "Epo":[1751328000000, 3.271375]},
{"Date":"2025-08-01", "Value":3.525881, "Epo":[1754006400000, 3.525881]}
]  }
],
"ErrorCount": 0 ,
"Errors": [],
"Elapsed_time_in_seconds":0.002
 } ]
```

Using the power of FAME by

getfame –e

with R from Jupyterlab

Chartest.ipynb | PySample.ipynb | r_2series.ipynb | Rfameexpr.ipynb

Code

•[3]:

```r
   # Load required libraries
library(jsonlite)
library(dplyr)
library(ggplot2)
library(scales)
library(lubridate)

famebase <- "$REFERTID/data/kpi_publ.db"
famedato <- "freq m; date 2005 to *"
series_list <- c("pct(convert(total.ipr,ann,con,end))",
                 "pct(convert(total.ipr,ann,linear,ave))",
                 "ytypct(total.ipr)",
                 "mave(pct(K01.IPR),3)" )

# Initialize an empty data frame to store all data
df_all <- data.frame()

# Process each series; can loop in R or apply more all expressions in the command separated by semicolon
for (famesoek in series_list) {
    # Construct the command for the current series
    command <- paste("ssh sl-fame-1.ssb.no '",
                     "$REFERTID/system/myfame/api/getfame -e \"", famebase,
                     "\" \"", famesoek, "\" \"", famedato, "\"'", sep="")

    # Execute the command and capture the output
    output <- system(command, intern = TRUE, ignore.stderr = FALSE)

    # Get the HOME environment variable
    home_dir <- Sys.getenv("HOME")

    # Construct the full path using the home directory
    json_file_path <- file.path(home_dir, ".GetFAME/getfameexpr.json")

    # Read the JSON file
    json_data <- fromJSON(json_file_path)

    # Check if any series was returned
    if (length(json_data$Series) == 0) {
        cat("No series found for:", famesoek, "\n")
        next
    }

    # Get the specific series data
```

getfame -e

R

different expressions & frequences & functions

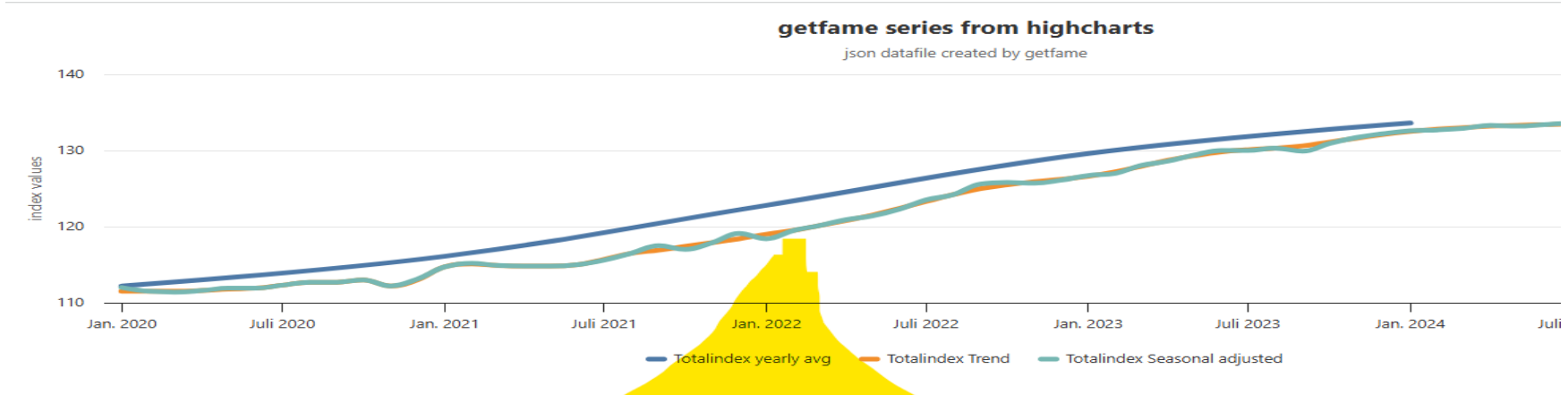
Chartest.ipynb | PySample.ipynb | r_2series.ipynb | Rfameexpr.ipynb

Code | git

```
# Ensure the final data frame handles NA values correctly
if (nrow(df_all) > 0) {
    # Plot the data using ggplot2
    options(repr.plot.width = 16, repr.plot.height = 6)

    ggplot(df_all, aes(x = as_datetime(Epoch_ms / 1000), y = Value, color = Series)) +
      geom_line() +
      scale_x_datetime(labels = date_format("%b%y"), date_breaks = "1 year") +
      labs(title = paste("Chart: ", "CPI % Changes"),
           x = "datetime month/year", y = "% Changes of CPI") +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
      guides(color = guide_legend(title = "")) +
      scale_y_continuous(labels = scales::comma)

} else {
    cat("\nNo data to plot.\n")
}
```

Interactive & zoomable chart with datetime as xaxis

getfame –s & -e create JSON prepared for highcharts




```
"Name": "KPI_PUBL'TOTAL.IPR.A",
Desc": "Totalindex yearly avg",
Daterange": "2020 TO *",
Frequency": "ANNUAL",
Observations":[
"Date":"2020-01-01", "Value":112.2, "Epo":[1577836800000, 112.2]},
"Date":"2021-01-01", "Value":116.1, "Epo":[1609459200000, 116.1]},
"Date":"2022-01-01", "Value":122.8, "Epo":[1640995200000, 122.8]},
"Date":"2023-01-01", "Value":129.6, "Epo":[1672531200000, 129.6]},
"Date":"2024-01-01", "Value":133.6, "Epo":[1704067200000, 133.6]}
  }    ,
"Name": "KPI_PUBL'TOTAL.IPR.G",
Desc": "Totalindex Trend",
Daterange": "2020 TO *",
Frequency": "MONTHLY",
Observations":[
"Date":"2020-01-01", "Value":111.5, "Epo":[1577836800000, 111.5]},
"Date":"2020-02-01", "Value":111.5, "Epo":[1580515200000, 111.5]},
"Date":"2020-03-01", "Value":111.5, "Epo":[1583020800000, 111.5]},
"Date":"2020-04-01", "Value":111.6, "Epo":[1585699200000, 111.6]},
"Date":"2020-05-01", "Value":111.8, "Epo":[1588291200000, 111.8]},
```

Help info - when no arguments passed

getfame –n

getfame –s

getfame -e

For complete samples with R and Python, see Github

getfame Summary

- The **getfame –e** option use the full power of FAME and can evaluate formulas, functions, conversions among various series, formulas,frequiencies and databases

- To get more series with **getfame –e** «simply» loop by expressions and add to same charts or dataset. (R sample)

- **getfame –n** is powerful when combining ***grep*** **|** ***more*** **|*****head*** to search for series/formulas names or metadata

- Use the **Epo** touplet, more robust, no date formatting, and smarter

- Silent mode: **getfame –nq** **getfame –sq** **getfame -eq**