Developing an Application to Compute Fourier Transformations of Hydrographs for Analysis

Jacob T. Lougee

IESA, University of North Georgia

GISC 4500K

Dr. Huidae Cho

May 3, 2021

Abstract

This project sought to develop a simple-to-use application capable of pulling hydrologic data from the United States Geological Survey (USGS) data servers, Fourier transforming it, and displaying the results for the end user to draw conclusions with. The program itself was built to be easy to use, and easy to modify.

Introduction

Fourier transformations are not a new approach to analyzing hydrologic data. In fact, many research articles that were encountered while writing this essay seemed to almost overlook Fourier transformations in favor of more "dynamic" or "adaptable" formulas for analyzing discharge data, such as the wavelet transformation which seems to be the most popular method in this sort of, "family" of mathematical approaches to hydrologic analyses. However, almost from the start, the idea for this project has been, not to invent a fresh new method for analyzing hydrologic data, but to make it very easy to use the existing fast Fourier transformation, and apply it to whatever site a user of the finished application might have in mind, for whatever purpose they have in mind. Additionally, the hope is that others may be able to very easily build on this work and turn it into a, still convenient and intuitive, but fairly powerful and flexible little software tool for this and other forms of analysis of hydrologic data.

Literature

While researching the topic, "hydrologic feature extraction using the fourier transformation," some cursory research uncovered that the open geospatial consortium stated, "Hydrologic features are the unit of water information required to convey identity of real-world water-objects through the data processing chain from observation to water information," or the, "abstraction of real world phenomena." In all honesty, neither of those definitions got the author of this paper any closer to understanding what a good end goal for this project should be, and may actually refer to hydrologic "features" such as lakes, rivers, and the rest.

But, with that in mind, research began by looking into Fourier transformations themselves with the assumption that "features" was to refer to any hydrologic events that would remain in a hydrograph after some form of "de-noising" by performing a Fourier transformation, applying some function to the transformed data and then performing an inverse Fourier transformation, which, in python, is as simple as. "numpy.fft.ifft()". In the paper titled, "*Spectral analysis of base flow separation with digital filters*," one civil engineer, M. E. Spongberg, did essentially that, and suggested optimal filtering procedures based on filter theory. For their analyses, that author consistently examined graphs of the number of cycles per day. This simple fact, that a hydrologist might want to investigate flow rate data, and its frequency, at such a high sample rate, informed the first major decision for this application, which was to "get" and transform 15 minute flow rate data. It would be useful; however, not to limit the functionality of the final application to a single use case, such as base flow separation, so more investigation was warranted.

In Gudmundsson (1970, p.342) the authors state that a, "review paper by Kisiel (1969) deals with the application of spectral analysis to hydrology. It also contains an introduction to the estimation of spectra." This may be something worth further investigation for anyone researching this topic in the future. Anderson(2007), and Gudmundsson (1970 p.345) also analyzed hydrologic data in a small time scale, making use of Fourier transformations. Gudmundsson compared the Fourier transformations of six different months and also utilized a low frequency filter (p. 344) These may be capabilities worth adding to the application, that is frequency filtering and comparing Fourier transformations across different datasets, be it location, or equal intervals of time, but that would certainly make it easier for a user to "break" as well.

Although comparing two hydrographs, or a hyetographs to a hydrograph, was eventually identified as an unattainable goal within the time constraints for this project, a not-insignificant amount of time was committed to researching it, but not entirely in vain. One take-away from the hydrologists in the article, "*Series distance – an intuitive metric to quantify hydrograph similarity in terms of occurrence, amplitude and timing of hydrological event*s," comes, surprisingly, from the introduction. They posed a hypothetical conversation with a hydrologist who was responsible for forecasting surface flow rates, which demonstrated that, "be it for parameter estimation during model calibration, model validation, classification of hydrological systems or identification of scales at which to separate explicit and implicit representations of structures and processes: metrics, measures and objective functions (including subjective visual inspection) are applied in all disciplines of Hydrology." This helped to inform two decisions for the application. The first was that a user should be able to select a timescale, and the second was that, although not ideal, it might be somewhat acceptable for the application not to return a specific value after retrieving the data, but to instead, simply display the information for subjective visual inspection.

It was later determined that the most practical definition of features, for this project, should be "seasonality," or essentially, predictable changes that occur over specific regular intervals in time series data. But what interval determines a season? The initial determination was that in order to be useful in as many cases as possible the interval should not be fixed, but the professor later clarified that a change with a period of one year should be considered seasonal. As an aside, it may be worth noting here that because a water year is 12 months, it may not matter either mathematically or to a hydrologist, if that is a water year or a calendar year. It has been difficult, however to determine if there is an actual calculable metric for seasonality I turned to papers outside of hydrology that use the Fourier transformations such as Sun et al(2020), and found that although they calculated metrics such as seasonal maxima and minima, "seasonality" is essentially a descriptor for the graphical trends in the data, as it's shown on a graph of day-month but not year so that at every x-value there are as many y-values as the number of years being analyzed. This suggests that "seasonality" is something which is typically qualified via graphs, rather than being quantified but it is possible that this is a misinterpretation.

Beyond ordinary Fourier transforms many other "related" transformations and analyses are common in the current literature including; Gabor transformations which could be used to generate spectrograms showing strength of frequency vs time, STFT(short time or windowed Fourier transformations) such as the model in Anderson et al. (2007), and Wavelet Transformations as is seen in Pandey et al. (2017), and an many other papers that were seen in the process of finding this literature but not used, specifically Mexican hat wavelet is of interest to the field of hydrology and seasonality detection, especially when the seasonal fluctuations are not quite periodic, or just for examining a variety of time scales Smith et al. (1998).

Additionally, although they aren't peer reviewed articles, this video series "https://www.youtube.com/playlist?list=PLMrJAkhIeNNT_Xh3Oy0Y4LTj0Oxo8GqsC" would be quite helpful for anyone attempting to fully understand Fourier transforms, and some other uses for them, in order to better imagine some operations that would be useful for hydrologic applications as I was, and this slideshow that I found online might be inspirational "http://www.l3s.de/~anand/tir14/lectures/ws14-tir-foundations-2.pdf". This textbook chapter was also somewhat helpful "http://search.ebscohost.com.proxygsu-nga1.galileo.usg.edu/login.aspx?direct=true&AuthType=ip,shib&db=nlebk&AN=92180&site=eds-live&scope=site&custid=ns235470&ebv=EB&ppid=pp_443" but may require a closer read than I was able to give it under the time limit.

Project statement

Initially, this project set out to design a tool which a hydrologist could feasibly use for any number of analyses which required a Fourier transformation of flow rate data. As the tool was developed, an attainable objective became clear.

Objectives

The primary objective is to develop an application capable of fetching flow rate data from the USGS servers and displaying it, the Fourier transformation of it, as well as all the component parts of the transform. The application should be relatively easy to simply run, use, and not necessarily require the user to interact with the command line.

Materials and methods

This program is written in python and requires the user to install python and the; numpy, matplotlib, os, urllib, json, PySimpleGUI, and pandas, libraries. The data comes from the United States Geological Survey (USGS) rest services. The initial problem for creating this application was how to obtain hydrologic data to analyze. Fortunately, the rest services, daily values API for the USGS is well documented at this url, "https://waterservices.usgs.gov/rest/DV-Service.html#Service". Because we had practiced reading JSON objects as a python object in class with, "json.loads()," it was decided to build a "get" request that asks for the data in that

format. The documentation states that, “The current version of WaterML will be rendered in a JSON structure as a set of name/value pairs. JSON is excellent for Web 2.0 applications. However, use JSON with caution as name/value pairs will change automatically when the default version of WaterML is upgraded.” The documentation for WaterML was put to some use to figure out how to get to the list of values, and it was somewhat helpful but largely the location of information in the object was deduced by “printing.” I found that the data itself was inside a list called value, in the first list within “values”, which was in the first list in “timeseries”, which was in the list called “value”. So the path to any value “i” would be ["value"]["timeSeries"][0]["values"][0]["value"][i]. I also found that the date and time was formatted like, “YYYY-MM-DDT00:00:00.000-00:00” and decided to only used the first 19 characters of that like, “YYYY-MM-DDT00:00:00” to make the output hydrograph more legible.

The decision to use pySimpleGUI was made for two reasons. The first was that I had already used it, and so to some extent, was familiar with how it worked, and second a goal of the project was for the end user not to necessarily be required to interact with the command line. The decision to build the request to return all values within a period from now was made simply to keep the interface and the request to the server simple and “idiot-proof” as they say. It is noteworthy for anyone modifying this in the future that PySimpleGUI has a built in “CalendarButton” feature for selecting a date on a calendar and passing that date as a user input. So, it would be very easy to replace “&period=P{}D” in the url request with, “&startDT= yyy-mm-dd&endDT=yyy-mm-dd” That would make the application more flexible, although ever-so-slightly easier for the end-user to “break” by picking invalid ranges, or struggling to use the calendar button correctly, which is something that, having designed several reasonably simple user interfaces in survey123, the author of this paper has learned to expect.

Now that the data was in python as lists it was possible to graph and Fourier transform it. Some adjustments had to be made to ensure that the graphs would consistently be legible, for example, the length between y ticks on graphs of only a few uneventful days of data was, “out of range” and causing errors so it was necessary to use either the “ideal” value, or 1, whichever was largest. plt.figure() was used before the design for each plot so that plt.show() would produce all the graphs simultaneously, rather than one at a time.The first graph, figure 1, is the hydrograph which simply displays the original data as is. The second, figure 2, is the frequency magnitude spectrum which was meant to be interpreted like a periodogram. Last, figure 3, is a graph of the component sin and cos frequencies that make up the original hydrograph, and borrows from, “https://github.com/HuidaeCho/digip/blob/master/digip.py”

A few small changes were made to that borrowed piece of, “digip” script though. First, the manual methodology for calculation of the Fourier transform was removed and replaced with the fast Fourier transform function in numpy. There were labels in the legend which, as noted in the python file itself, were removed, because for most hydrographs there were too many “u” values for this legend to fit the table, but as commented on those lines, the structure is useful for labeling or displaying specific values. For example, to plot and label a specific F[u] value, you could eliminate the "for loop" line, dedent these three lines that the comment is in, and replace, "u" in the formula that calculates, "yy" with the value you wish to plot. For the monthly average statistics data, which will be discussed more, soon, some experimentation was done to see if

perhaps multiples of 12 or the number of months or years would be particularly useful frequencies to plot or display for the user, but the results were seemingly inconclusive. That may warrant closer investigation in the future though. Lastly, Dr.Cho's script initially printed F(values) to the command line, but there were so many for most hydrographs, that the IDE being used to write the application couldn't display all of them, so it was slightly altered to open a text file, write all the values to that, and open it automatically.

After building the structure for fetching fifteen minute data, it was decided that the application should instead utilize monthly statistics data in order to analyze several years' worth of data for seasonality. Rather than eliminate the work that had already been done, this was added as a second button option in the user interface. Like the rest services, daily values API before, the United States Geological Survey (USGS) statistics service API is also well documented but is found instead, at this url, "https://waterservices.usgs.gov/rest/Statistics-Service.html". Unfortunately, for the URL argument used to specify the output format of the data returned, the documentation states, "Only tab-delimited (RDB) is available at this time. XML, JSON and Excel will be available in the future. rdb is a self-describing tab-delimited format used widely by the USGS." So reading the data would require a different approach. Essentially just by, "brute force," trial and error, a methodology for reading the data was developed. First the response is opened with readlines(), then decoded with utf-8. Next, the lines in the description, which are commented out and therefore denoted by, "#" are passed, and the lines which are rows in the table are appended to a list. The first line in that list is the column names, but there was consistently an extra one which needed to be removed (this is noted in the python file). The data is loaded into a pandas data frame because that makes it easy to read table data. The relevant columns are read as lists and the data is ready to be graphed, transformed and displayed, much like before.

One aim was to plot a spectrogram using a Gabor Transformation, because from the research, it seems that it's a good compromise between the time and frequency domains, which this application plots in figure one, and figure two, respectively. One example online made it appear that it could be as simple as using plt.specgram(flow_rate, NFFT={}, Fs={}, noverlap={}) Unfortunately, I could not find an appropriate "sampling rate" (Fs), window time scale (NFFT), and overlap, that produced anything near as useful and legible as the examples I was seeing demonstrated with sound recordings, but for any future student researching this topic, it's hoped that this url still works, and can help,
"https://ccrma.stanford.edu/~jos/mdft/Sampling_Theory.html"

User guide

In order to use this application, make sure that you have python and the numpy, matplotlib, os, urllib, json, PySimpleGUI, and pandas, libraries installed. Run the program and click on, 'Click "here" to find the site you would like to analyze. (Please be patient)', in the window that appears. This should open the United States Geological Survey (USGS) national water information system mapper in your default web browser. It is important that you pick a site which has the data you are trying to transform. To verify this, click on the site in the USGS national water information system mapper, and a popup will appear. Click on the blue hyperlink

in that popup that says “access data.” That will open a page which displays all the available data for that site.

If you are trying to use the 'Transform 15 Minute Data' button, you must verify three things. One, that the data type, “Daily Data: Discharge” (not, “Daily Statistics: Discharge”) is in that table, two that the end date is the current date, and three that the begin date is not outside of the day range that you want to specify. If all three conditions are met, the 'Transform 15 Minute Data' button will function, otherwise you must pick a different site or day range accordingly. (See figure-4)

If you are trying to use the, ‘Transform Monthly Average Data’ button, you must verify that, “Monthly Statistics: Discharge” is in that table. If this condition is met the, ‘Transform Monthly Average Data’ button will function, otherwise you must pick a different site. (See figure-5)

When either transformation is run, a hydrograph (Figure 1), frequency magnitude spectrum(Figure 2), graph of the wave components of the hydrograph (Figure 3), and text report of all F(u) components in the hydrograph (Figure 4) should appear. If you enter invalid or no values a popup error message explaining the issue should appear.

Results

The application was manually tested over 50 times on at least 30 valid sites and 20 invalid sites, and consistently performed as expected, allowing a skilled user to visually interpret the data. The highest date range tested on the monthly statistics data was 21 years, or 250 data points, and the highest date range tested on the 15 minute data was 365 days or 35,040 data points. Shown in figures one through four are the example outputs from one of the test runs on 15 minute data. Matplotlib was used to display the data so that the user is able to use the built in navigation bar to explore it, and the values for F(u) in the data range are saved to a text document which is automatically opened for viewing.

Conclusions

For non-programmers, the application should be useful for fetching and transforming the data, visually identifying key hydrologic features in the frequency spectrum, and selecting the appropriate values for various hydrologic analyses. For programmers, the script should provide an excellent framework for even further analysis. Future research may aim to explore other, related transformations and methods of analysis which have been discussed such as autocorrelation, Gabor transformations to generate spectrograms showing strength of frequency vs time, STFT(short time or windowed Fourier transformations), and or Wavelet Transformations (specifically Mexican Hat).

Acknowledgments

Thanks to Dr.Cho for the suggestion to research this concept, for teaching me python, and for your advice while I was working on this project.

Appendix

Figure 1

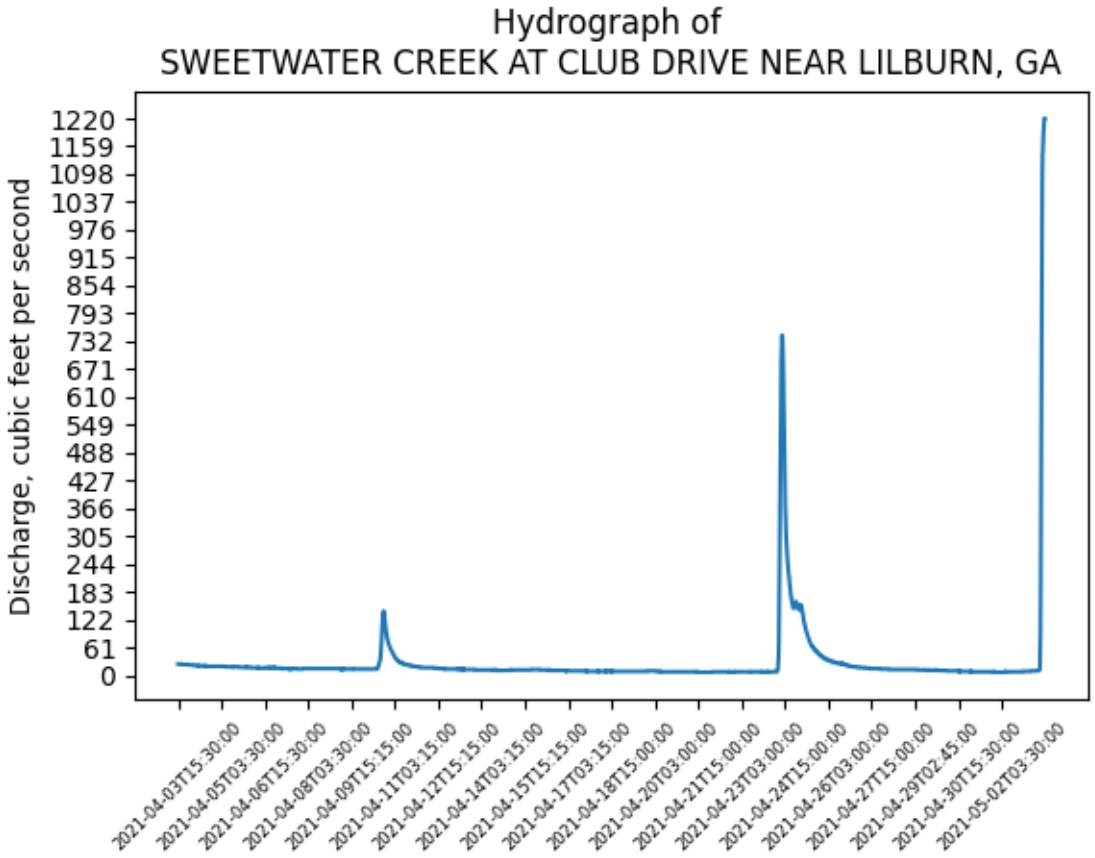


Figure 2

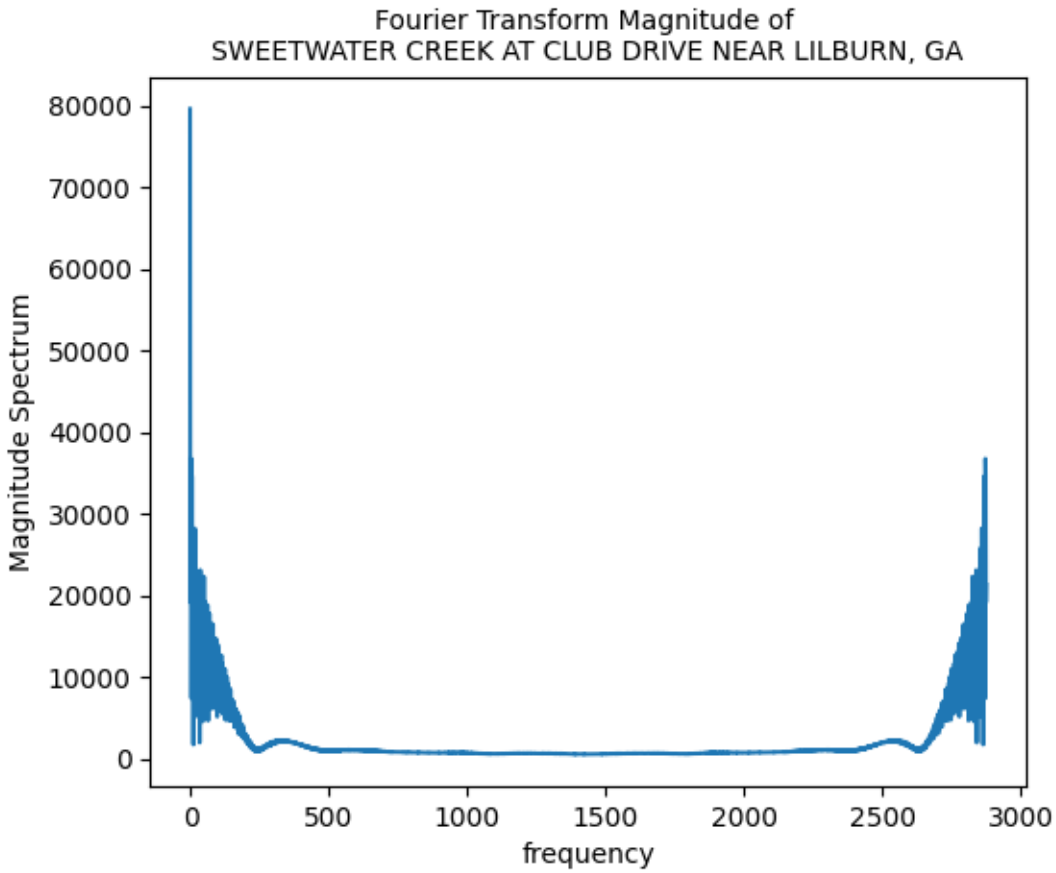

Fourier Transform Magnitude of
SWEETWATER CREEK AT CLUB DRIVE NEAR LILBURN, GA
Magnitude Spectrum
80000
70000
60000
50000
40000
30000
20000
10000
0
0
500
1000
1500
2000
2500
3000
frequency

Figure 3

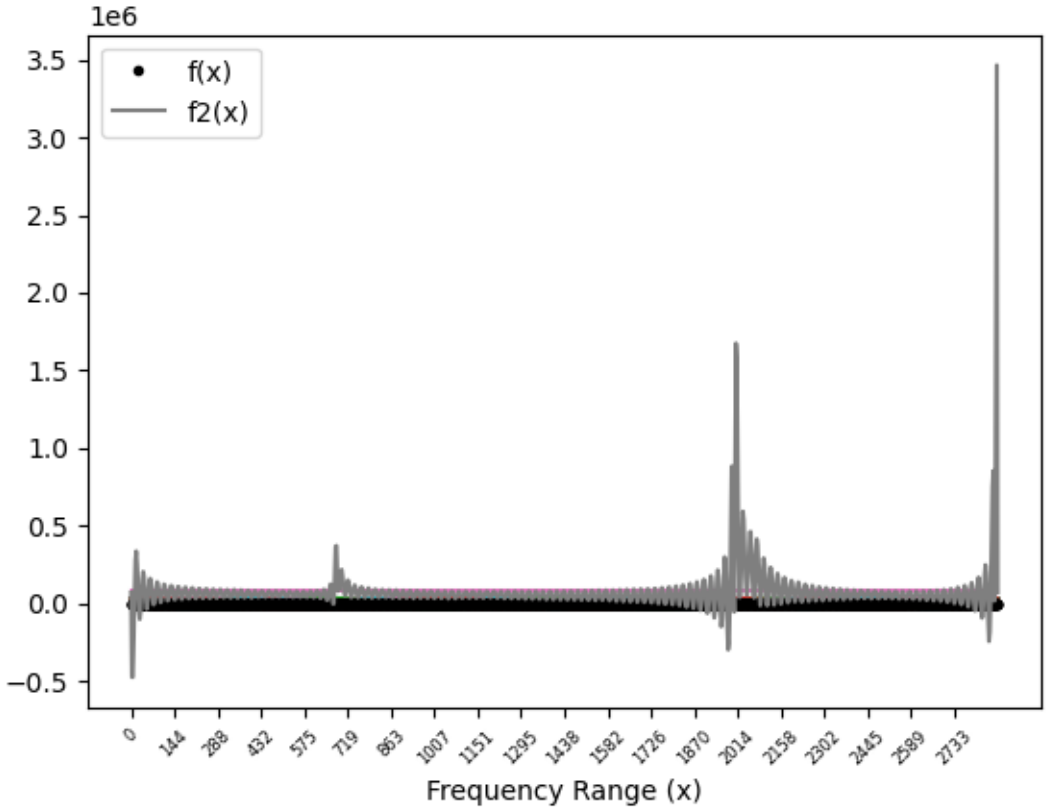

1e6
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
f(x)
f2(x)
0
144
288
432
575
719
863
1007
1151
1295
1438
1582
1726
1870
2014
2158
2302
2445
2589
2733
Frequency Range (x)

Figure 4

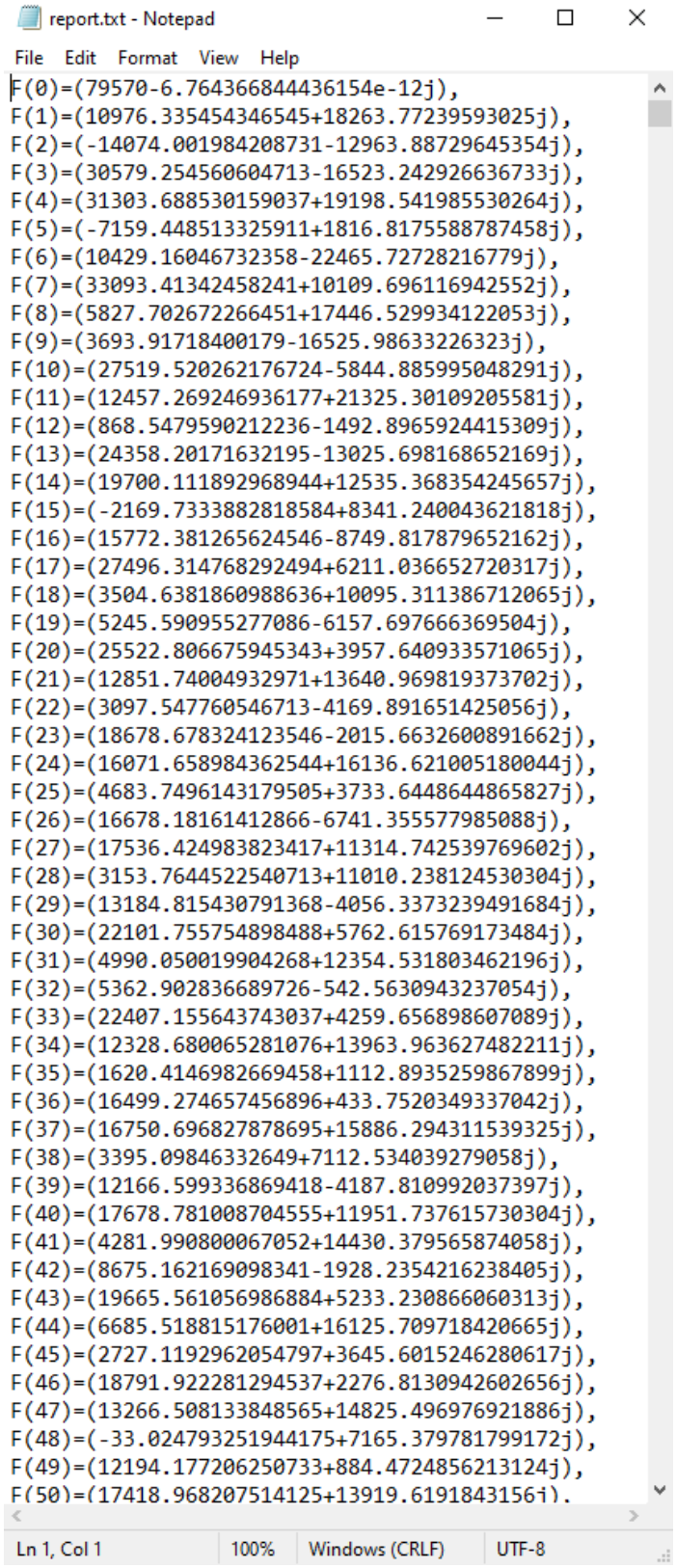
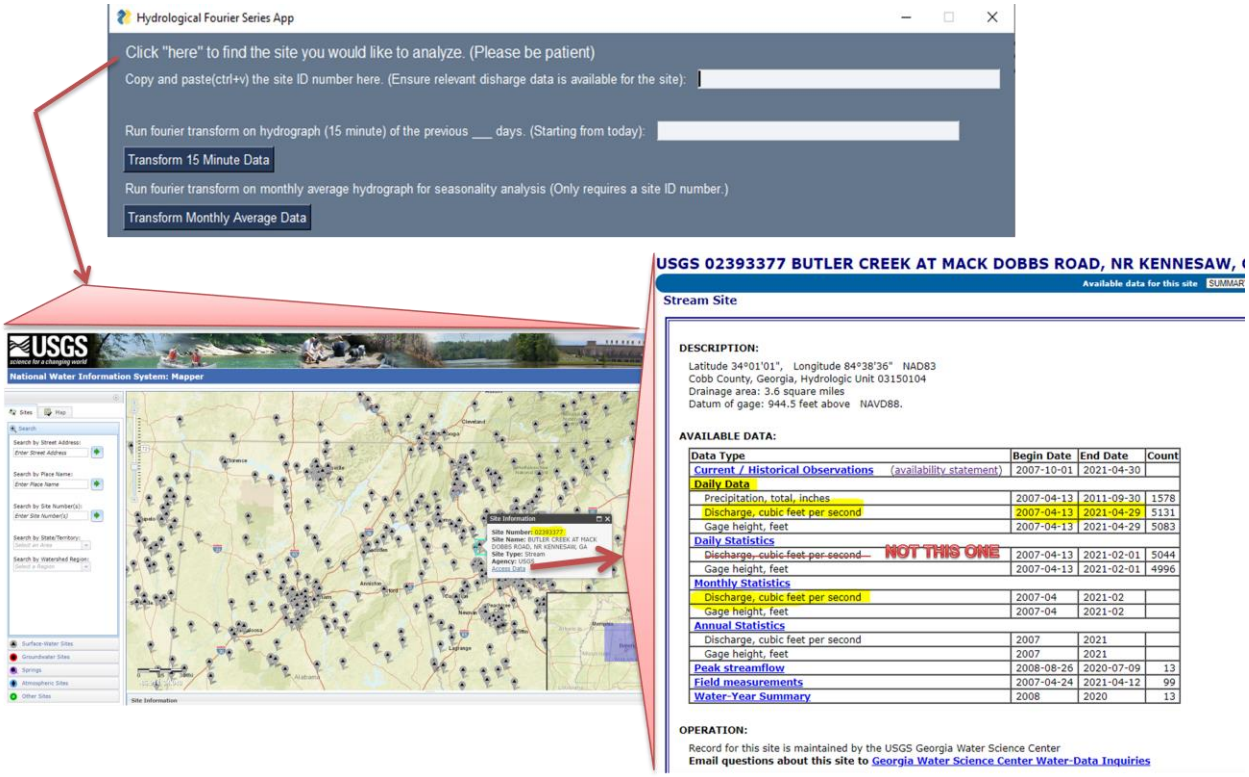

report.txt - Notepad
File Edit Format View Help
F(0)=(79570-6.764366844436154e-12j),
F(1)=(10976.335454346545+18263.77239593025j),
F(2)=(-14074.001984208731-12963.88729645354j),
F(3)=(30579.254560604713-16523.242926636733j),
F(4)=(31303.688530159037+19198.541985530264j),
F(5)=(-7159.448513325911+1816.8175588787458j),
F(6)=(10429.16046732358-22465.72728216779j),
F(7)=(33093.41342458241+10109.696116942552j),
F(8)=(5827.702672266451+17446.5299341220053j),
F(9)=(3693.91718400179-16525.98633226323j),
F(10)=(27519.520262176724-5844.885995048291j),
F(11)=(12457.269246936177+21325.30109205581j),
F(12)=(868.5479590212236-1492.89659244153O9j),
F(13)=(24358.20171632195-13025.698168652169j),
F(14)=(19700.111892968944+12535.368354245657j),
F(15)=(-2169.7333882818584+8341.240043621818j),
F(16)=(15772.381265624546-8749.817879652162j),
F(17)=(27496.314768292494+6211.036652720317j),
F(18)=(3504.6381860988636+10095.311386712065j),
F(19)=(5245.590955277086-6157.697666369504j),
F(20)=(25522.806675945343+3957.6409335710651),
F(21)=(12851.74004932971+13640.969819373702j),
F(22)=(3097.547760546713-4169.891651425056j),
F(23)=(18678.678324123546-2015.6632600891662j),
F(24)=(16071.658984362544+16136.621005180044j),
F(25)=(4683.7496143179505+3733.6448644865827j),
F(26)=(16678.18161412866-6741.355577985088j),
F(27)=(17536.424983823417+11314.742539769602j),
F(28)=(3153.7644522540713+11010.238124530304j),
F(29)=(13184.815430791368-4056.3373239491684j),
F(30)=(22101.755754898488+5762.615769173484j),
F(31)=(4990.050019904268+12354.531803462196j),
F(32)=(5362.902836689726-542.56309432370541),
F(33)=(22407.155643743037+4259.656898607089j),
F(34)=(12328.680065281076+13963.963627482211j),
F(35)=(1620.4146982669458+1112.8935259867899j),
F(36)=(16499.274657456896+433.75203493370421),
F(37)=(16750.696827878695+15886.294311539325j),
F(38)=(3395.09846332649+7112.534039279058j),
F(39)=(12166.599336869418-4187.810992037397j),
F(40)=(17678.781008704555+11951.737615730304j),
F(41)=(4281.990800067052+14430.379565874058j),
F(42)=(8675.162169098341-1928.2354216238405j),
F(43)=(19665.561056986884+5233.230866060313j),
F(44)=(6685.518815176001+16125.709718420665j),
F(45)=(2727.1192962054797+3645.6015246280617j),
F(46)=(18791.922281294537+2276.8130942602656j),
F(47)=(13266.508133848565+14825.496976921886j),
F(48)=(-33.024793251944175+7165.379781799172j),
F(49)=(12194.177206250733+884.4724856213124j),
F(50)=(17418.968207514125+13919.61918431561j),
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8

Figure 5


Hydrological Fourier Series App
Click "here" to find the site you would like to analyze. (Please be patient)
Copy and paste(ctrl+v) the site ID number here. (Ensure relevant disharge data is available for the site):
Run fourier transform on hydrograph (15 minute) of the previous ___ days. (Starting from today):
Transform 15 Minute Data
Run fourier transform on monthly average hydrograph for seasonality analysis (Only requires a site ID number.)
Transform Monthly Average Data
USGS
National Water Information System: Mapper
USGS 02393377 BUTLER CREEK AT MACK DOBBS ROAD, NR KENNESAW,
Stream Site
DESCRIPTION:
Latitude 34°01'01", Longitude 84°38'36" NAD83
Cobb County, Georgia, Hydrologic Unit 03150104
Drainage area: 3.6 square miles
Datum of gage: 944.5 feet above NAVD88.
AVAILABLE DATA:
Data Type | Begin Date | End Date | Count
Current / Historical Observations (availability statement) | 2007-10-01 | 2021-04-30 |
Daily Data
Precipitation, total, inches | 2007-04-13 | 2011-09-30 | 1578
Discharge, cubic feet per second | 2007-04-13 | 2021-04-29 | 5131
Gage height, feet | 2007-04-13 | 2021-04-29 | 5083
Daily Statistics
Discharge, cubic feet per second NOT THIS ONE | 2007-04-13 | 2021-02-01 | 5044
Gage height, feet | 2007-04-13 | 2021-02-01 | 4996
Monthly Statistics
Discharge, cubic feet per second | 2007-04 | 2021-02 |
Gage height, feet | 2007-04 | 2021-02 |
Annual Statistics
Discharge, cubic feet per second | 2007 | 2021 |
Gage height, feet | 2007 | 2021 |
Peak streamflow | 2008-08-26 | 2020-07-09 | 13
Field measurements | 2007-04-24 | 2021-04-12 | 99
Water-Year Summary | 2008 | 2020 | 13
OPERATION:
Record for this site is maintained by the USGS Georgia Water Science Center
Email questions about this site to Georgia Water Science Center Water-Data Inquiries

References

Anderson, P. L., Tesfaye, Y. G., & Meerschaert, M. M. (2007). Fourier-PARMA Models and Their Application to River Flows. *Journal of Hydrologic Engineering, 12(5), 462–472.* https://doi-org.proxygsu-nga1.galileo.usg.edu/10.1061/(ASCE)1084-0699(2007)12:5(462)

G. Gudmundsson (1970) Short term variations of a glacier-fed river, Tellus, 22:3, 341-353, DOI: 10.3402/tellusa.v22i3.10227

Manga, M. (1996). Hydrology of Spring-Dominated Streams in the Oregon Cascades. *Water Resources Research, 32(8), 2435–2439.*

Næ tinger, B., & Gautier, Y. (1998). Use of the Fourier-Laplace transform and of diagrammatical methods to interpret pumping tests in heterogeneous reservoirs. *Advances in Water Resources, 21(7), 581–590.* https://doi-org.proxygsu-nga1.galileo.usg.edu/10.1016/S0309-1708(97)00014-6

Pandey, B. K., Tiwari, H., & Khare, D. (2017). Trend analysis using discrete wavelet transform (DWT) for long-term precipitation (1851–2006) over India. *Hydrological Sciences Journal/Journal Des Sciences Hydrologiques, 62(13), 2187–2208.* https://doi-org.proxygsu-nga1.galileo.usg.edu/10.1080/02626667.2017.1371849

Smith, L. C., Turcotte, D. L., & Isacks, B. L.(1998). Stream Flow characterization and feature detection using a discrete wavelet transform, *Hydrological Processes, 12, 233-249.* https://doi.org/10.1002/(SICI)1099-1085(199802)12:2<233::AID-HYP573>3.0.CO;2-3

Spongberg, M. E. (2000). Spectral analysis of base flow separation with digital filters. *Water Resources Research,36(3), 745-752.* https://doi.org/10.1029/1999WR900303

Sun, Y., Liu, C., Zhang, L., Palm, M., Notholt, J., Yin, H., Vigouroux, C., Lutsch, E., Wang, W., Shan, C., Blumenstock, T., Nagahama, T., Morino, I., Mahieu, E., Strong, K., Langerock, B., De Mazière, M., Hu, Q., Zhang, H., & Petri, C. (2020). Fourier transform infrared time series of tropospheric HCN in eastern China: seasonality, interannual variability, and source attribution. *Atmospheric Chemistry & Physics, 20(9), 5437–5456.* https://doi-org.proxygsu-nga1.galileo.usg.edu/10.5194/acp-20-5437-2020

Taylor, P. (2014) OGC WaterML 2.0: Part 1- Timeseries Implementation Standard Corrigendum Encoding Approved for Public Release. *Open Geospatial Consortium, 10-126r4, (2014)* http://www.opengis.net/doc/IS/waterml/2.0.1

U. Ehret, & E. Zehe,(2011). Series distance – an intuitive metric to quantify hydrograph similarity in terms of occurrence, amplitude, and timing of hydrological events. *Hydrol. Earth Syst. Sci., 15, 877–896, (2011).* www.hydrol-earth-syst-sci.net/15/877/2011/doi:10.5194/hess-15-877-2011

Zhang, X., & Bao, W. (2013). Hydrodynamic Simulation in Tidal Rivers Using Fourier Series. *Journal of Hydrologic Engineering, 18(11), 1408–1415.* https://doi-org.proxygsu-nga1.galileo.usg.edu/10.1061/(ASCE)HE.1943-5584.0000526