Option_pricing

June 18, 2024

[1]:
```
import os
import stat

cd = os.getcwd()
print(cd)
if not cd.endswith("working"):
    if not os.path.exists("working"):
        os.makedirs("working")
    os.chdir("working")
cd = os.getcwd()
print(cd)
os.chmod(cd, stat.S_IRWXU)
```

```
C:\Users\Willie\Desktop\Option_pricing
C:\Users\Willie\Desktop\Option_pricing\working
```

[2]:
```
# Uncomment to install necessary packages.
'''
!pip install --upgrade anyio
!pip install --upgrade argon2-cffi
!pip install --upgrade argon2-cffi-bindings
!pip install --upgrade arrow
!pip install --upgrade asttokens
!pip install --upgrade async-lru
!pip install --upgrade attrs
!pip install --upgrade Babel
!pip install --upgrade beautifulsoup4
!pip install --upgrade bleach
!pip install --upgrade blurhash
!pip install --upgrade boost
!pip install --upgrade certifi
!pip install --upgrade cffi
!pip install --upgrade charset-normalizer
!pip install --upgrade colorama
!pip install --upgrade comm
!pip install --upgrade Cython
!pip install --upgrade debugpy
!pip install --upgrade decorator
```

```
!pip install --upgrade defusedxml
!pip install --upgrade exceptiongroup
!pip install --upgrade executing
!pip install --upgrade fastjsonschema
!pip install --upgrade fqdn
!pip install --upgrade greenlet
!pip install --upgrade h11
!pip install --upgrade httpcore
!pip install --upgrade httpx
!pip install --upgrade idna
!pip install --upgrade ipykernel
!pip install --upgrade ipython
!pip install --upgrade ipywidgets
!pip install --upgrade isoduration
!pip install --upgrade jedi
!pip install --upgrade Jinja2
!pip install --upgrade json5
!pip install --upgrade jsonpointer
!pip install --upgrade jsonschema
!pip install --upgrade jsonschema-specifications
!pip install --upgrade jupyter
!pip install --upgrade jupyter_client
!pip install --upgrade jupyter-console
!pip install --upgrade jupyter_core
!pip install --upgrade jupyter-events
!pip install --upgrade jupyter-lsp
!pip install --upgrade jupyter_server
!pip install --upgrade jupyter_server_terminals
!pip install --upgrade jupyterlab
!pip install --upgrade jupyterlab_pygments
!pip install --upgrade jupyterlab_server
!pip install --upgrade jupyterlab_widgets
!pip install --upgrade MarkupSafe
!pip install --upgrade Mastodon.py
!pip install --upgrade matplotlib
!pip install --upgrade mistune
!pip install --upgrade nbclient
!pip install --upgrade nbconvert
!pip install --upgrade nbformat
!pip install --upgrade nest-asyncio
!pip install --upgrade notebook
!pip install --upgrade notebook_shim
!pip install --upgrade numpy
!pip install --upgrade outcome
!pip install --upgrade overrides
!pip install --upgrade packaging
!pip install --upgrade pandas
```

```
!pip install --upgrade pandocfilters
!pip install --upgrade parso
!pip install --upgrade pip
!pip install --upgrade platformdirs
!pip install --upgrade plotly
!pip install --upgrade prometheus_client
!pip install --upgrade prompt-toolkit
!pip install --upgrade psutil
!pip install --upgrade pure-eval
!pip install --upgrade pybind11
!pip install --upgrade pycparser
!pip install --upgrade Pygments
!pip install --upgrade PySocks
!pip install --upgrade python-dateutil
!pip install --upgrade python-json-logger
!pip install --upgrade python-magic-bin
!pip install --upgrade pytz
!pip install --upgrade pywin32
!pip install --upgrade pywinpty
!pip install --upgrade PyYAML
!pip install --upgrade pyzmq
!pip install --upgrade qtconsole
!pip install --upgrade QtPy
!pip install --upgrade referencing
!pip install --upgrade requests
!pip install --upgrade rfc3339-validator
!pip install --upgrade rfc3986-validator
!pip install --upgrade rpds-py
!pip install --upgrade selenium
!pip install --upgrade Send2Trash
!pip install --upgrade setuptools
!pip install --upgrade six
!pip install --upgrade sniffio
!pip install --upgrade sortedcontainers
!pip install --upgrade soupsieve
!pip install --upgrade SQLAlchemy
!pip install --upgrade stack-data
!pip install --upgrade statsmodels
!pip install --upgrade tenacity
!pip install --upgrade terminado
!pip install --upgrade tinycss2
!pip install --upgrade tomli
!pip install --upgrade tornado
!pip install --upgrade traitlets
!pip install --upgrade trio
!pip install --upgrade trio-websocket
!pip install --upgrade types-python-dateutil
```

```
!pip install --upgrade typing_extensions
!pip install --upgrade tzdata
!pip install --upgrade uri-template
!pip install --upgrade urllib3
!pip install --upgrade wcwidth
!pip install --upgrade webcolors
!pip install --upgrade webencodings
!pip install --upgrade websocket-client
!pip install --upgrade wheel
!pip install --upgrade widgetsnbextension
!pip install --upgrade wsproto
'''
```

[2]:
```
'\n!pip install --upgrade anyio\n!pip install --upgrade argon2-cffi\n!pip
install --upgrade argon2-cffi-bindings\n!pip install --upgrade arrow\n!pip
install --upgrade asttokens\n!pip install --upgrade async-lru\n!pip install
--upgrade attrs\n!pip install --upgrade Babel\n!pip install --upgrade
beautifulsoup4\n!pip install --upgrade bleach\n!pip install --upgrade
blurhash\n!pip install --upgrade boost\n!pip install --upgrade certifi\n!pip
install --upgrade cffi\n!pip install --upgrade charset-normalizer\n!pip install
--upgrade colorama\n!pip install --upgrade comm\n!pip install --upgrade
Cython\n!pip install --upgrade debugpy\n!pip install --upgrade decorator\n!pip
install --upgrade defusedxml\n!pip install --upgrade exceptiongroup\n!pip
install --upgrade executing\n!pip install --upgrade fastjsonschema\n!pip install
--upgrade fqdn\n!pip install --upgrade greenlet\n!pip install --upgrade
h11\n!pip install --upgrade httpcore\n!pip install --upgrade httpx\n!pip install
--upgrade idna\n!pip install --upgrade ipykernel\n!pip install --upgrade
ipython\n!pip install --upgrade ipywidgets\n!pip install --upgrade
isoduration\n!pip install --upgrade jedi\n!pip install --upgrade Jinja2\n!pip
install --upgrade json5\n!pip install --upgrade jsonpointer\n!pip install
--upgrade jsonschema\n!pip install --upgrade jsonschema-specifications\n!pip
install --upgrade jupyter\n!pip install --upgrade jupyter_client\n!pip install
--upgrade jupyter-console\n!pip install --upgrade jupyter_core\n!pip install
--upgrade jupyter-events\n!pip install --upgrade jupyter-lsp\n!pip install
--upgrade jupyter_server\n!pip install --upgrade jupyter_server_terminals\n!pip
install --upgrade jupyterlab\n!pip install --upgrade jupyterlab_pygments\n!pip
install --upgrade jupyterlab_server\n!pip install --upgrade
jupyterlab_widgets\n!pip install --upgrade MarkupSafe\n!pip install --upgrade
Mastodon.py\n!pip install --upgrade matplotlib\n!pip install --upgrade
mistune\n!pip install --upgrade nbclient\n!pip install --upgrade nbconvert\n!pip
install --upgrade nbformat\n!pip install --upgrade nest-asyncio\n!pip install
--upgrade notebook\n!pip install --upgrade notebook_shim\n!pip install --upgrade
numpy\n!pip install --upgrade outcome\n!pip install --upgrade overrides\n!pip
install --upgrade packaging\n!pip install --upgrade pandas\n!pip install
--upgrade pandocfilters\n!pip install --upgrade parso\n!pip install --upgrade
pip\n!pip install --upgrade platformdirs\n!pip install --upgrade plotly\n!pip
install --upgrade prometheus_client\n!pip install --upgrade prompt-toolkit\n!pip
```

```
install --upgrade psutil\n!pip install --upgrade pure-eval\n!pip install
--upgrade pybind11\n!pip install --upgrade pycparser\n!pip install --upgrade
Pygments\n!pip install --upgrade PySocks\n!pip install --upgrade python-
dateutil\n!pip install --upgrade python-json-logger\n!pip install --upgrade
python-magic-bin\n!pip install --upgrade pytz\n!pip install --upgrade
pywin32\n!pip install --upgrade pywinpty\n!pip install --upgrade PyYAML\n!pip
install --upgrade pyzmq\n!pip install --upgrade qtconsole\n!pip install
--upgrade QtPy\n!pip install --upgrade referencing\n!pip install --upgrade
requests\n!pip install --upgrade rfc3339-validator\n!pip install --upgrade
rfc3986-validator\n!pip install --upgrade rpds-py\n!pip install --upgrade
selenium\n!pip install --upgrade Send2Trash\n!pip install --upgrade
setuptools\n!pip install --upgrade six\n!pip install --upgrade sniffio\n!pip
install --upgrade sortedcontainers\n!pip install --upgrade soupsieve\n!pip
install --upgrade SQLAlchemy\n!pip install --upgrade stack-data\n!pip install
--upgrade statsmodels\n!pip install --upgrade tenacity\n!pip install --upgrade
terminado\n!pip install --upgrade tinycss2\n!pip install --upgrade tomli\n!pip
install --upgrade tornado\n!pip install --upgrade traitlets\n!pip install
--upgrade trio\n!pip install --upgrade trio-websocket\n!pip install --upgrade
types-python-dateutil\n!pip install --upgrade typing_extensions\n!pip install
--upgrade tzdata\n!pip install --upgrade uri-template\n!pip install --upgrade
urllib3\n!pip install --upgrade wcwidth\n!pip install --upgrade webcolors\n!pip
install --upgrade webencodings\n!pip install --upgrade websocket-client\n!pip
install --upgrade wheel\n!pip install --upgrade widgetsnbextension\n!pip install
--upgrade wsproto\n'
```

[3]:
```
%load_ext cython
```

[4]:
```
%%cython
# cython: language_level=3str

from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.keys import Keys
import time
import os
import zipfile
from io import StringIO

cdef str cur_directory = os.getcwd()
cdef str dd_dir = os.path.join(cur_directory, "downloads")
cdef str od_dir = os.path.join(cur_directory, "optData")
cdef str fd_dir = os.path.join(cur_directory, "futData")
cdef str cpd_dir = os.path.join(cur_directory, "completeData")
```

```
cdef str opt_dir = os.path.join(dd_dir, "option")
cdef str fut_dir = os.path.join(dd_dir, "future")
cdef str cpi_dir = os.path.join(dd_dir, "cpi")
cdef str ccpi_dir = os.path.join(dd_dir, "ccpi")
cdef str twii_dir = os.path.join(dd_dir, "twii")
cdef str ixic_dir = os.path.join(dd_dir, "ixic")
cdef str trs_dir = os.path.join(dd_dir, "treasury")
cdef str tu_dir = os.path.join(dd_dir, "twdusd")
cdef str fer_dir = os.path.join(dd_dir, "forward")
os.makedirs(od_dir, exist_ok=True)
os.makedirs(fd_dir, exist_ok=True)
os.makedirs(cpd_dir, exist_ok=True)
os.makedirs(opt_dir, exist_ok=True)
os.makedirs(fut_dir, exist_ok=True)
os.makedirs(cpi_dir, exist_ok=True)
os.makedirs(ccpi_dir, exist_ok=True)
os.makedirs(twii_dir, exist_ok=True)
os.makedirs(ixic_dir, exist_ok=True)
os.makedirs(trs_dir, exist_ok=True)
os.makedirs(tu_dir, exist_ok=True)
os.makedirs(fer_dir, exist_ok=True)

def optDownload(int start_year, int end_year):
    chrome_options = Options()
    chrome_options.add_experimental_option("prefs", {
        "download.default_directory": opt_dir,
        "download.prompt_for_download": False,
        "download.directory_upgrade": True,
        "safebrowsing.enabled": True
    })
    driver = webdriver.Chrome(options=chrome_options)

    files = os.listdir(opt_dir)
    ls = [i for i in range(start_year, end_year + 1) if str(i) + "_opt.zip" not␣
 ↪in files and not any(s.startswith(str(i) + "_opt") and s.endswith(".csv")␣
 ↪for s in files)]

    if len(ls):
        driver.get("https://www.taifex.com.tw/cht/3/optDailyMarketView")
        select_year = Select(driver.find_element(By.ID, "his_year"))
        for i in ls:
            select_year.select_by_value(str(i))
            download_button = driver.find_element(By.ID, "button9")
            download_button.click()

            seconds = 0
            dl_wait = True
```

```
            while dl_wait:
                time.sleep(1)
                dl_wait = False
                for fname in os.listdir(opt_dir):
                    if fname.endswith('.crdownload') or fname.endswith('.part'):
                        dl_wait = True
                seconds += 1

        driver.quit()

def futDownload(int start_year, int end_year):
    chrome_options = Options()
    chrome_options.add_experimental_option("prefs", {
        "download.default_directory": fut_dir,
        "download.prompt_for_download": False,
        "download.directory_upgrade": True,
        "safebrowsing.enabled": True
    })
    driver = webdriver.Chrome(options=chrome_options)

    files = os.listdir(fut_dir)
    ls = [i for i in range(start_year, end_year + 1) if str(i) + "_fut.zip" not␣
 ↪in files and not any(s.startswith(str(i) + "_fut") and s.endswith(".csv")␣
 ↪for s in files)]

    if len(ls):
        driver.get("https://www.taifex.com.tw/cht/3/futDailyMarketView")
        select_year = Select(driver.find_element(By.ID, "his_year"))
        for i in ls:
            select_year.select_by_value(str(i))
            download_button = driver.find_element(By.ID, "button9")
            download_button.click()

            seconds = 0
            dl_wait = True
            while dl_wait and seconds < 60:
                time.sleep(1)
                dl_wait = False
                for fname in os.listdir(fut_dir):
                    if fname.endswith('.crdownload') or fname.endswith('.part'):
                        dl_wait = True
                seconds += 1

    driver.quit()

def cpiDownload(dirc=cpi_dir):
    files = os.listdir(dirc)
```

```
    if "pr0102a1m.xml" not in files:
        cpi_options = Options()
        cpi_options.add_experimental_option("prefs", {
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        button_xpath = "//button[@ariadisabled='false' and @class='el-button␣
 ↪el-button--primary is-plain']"

        driver = webdriver.Chrome(options=cpi_options)
        driver.get("https://data.gov.tw/dataset/6305")
        time.sleep(5)
        button = driver.find_element(By.XPATH, button_xpath)
        button.click()
        time.sleep(5)
        seconds = 0
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()

    return os.listdir(dirc)

def ccpiDownload(dirc=ccpi_dir):
    files = os.listdir(dirc)
    if "pr0103a1m.xml" not in files:
        cpi_options = Options()
        cpi_options.add_experimental_option("prefs", {
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        button_xpath = "//button[@ariadisabled='false' and @class='el-button␣
 ↪el-button--primary is-plain']"

        driver = webdriver.Chrome(options=cpi_options)
        driver.get("https://data.gov.tw/dataset/8237")
        time.sleep(5)
```

```
        button = driver.find_element(By.XPATH, button_xpath)
        button.click()

        seconds = 0
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()

    return os.listdir(dirc)

def tuDownload(dirc=tu_dir):
    files = os.listdir(dirc)
    if "EG51D01.csv" not in files:
        cpi_options = Options()
        cpi_options.add_experimental_option("prefs", {
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        button_xpath = "//button[@ariadisabled='false' and @class='el-button␣
 ↪el-button--primary is-plain']"

        driver = webdriver.Chrome(options=cpi_options)
        driver.get("https://data.gov.tw/dataset/10818")
        time.sleep(5)
        button = driver.find_element(By.XPATH, button_xpath)
        button.click()

        seconds = 0
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()
```

```
    return os.listdir(dirc)

def ferDownload(dirc=fer_dir):
    files = os.listdir(dirc)
    if "EG55D01.csv" not in files:
        cpi_options = Options()
        cpi_options.add_experimental_option("prefs", {
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        button_xpath = "//button[@ariadisabled='false' and @class='el-button␣
 ↪el-button--primary is-plain']"

        driver = webdriver.Chrome(options=cpi_options)
        driver.get("https://data.gov.tw/dataset/10817")
        time.sleep(5)
        button = driver.find_element(By.XPATH, button_xpath)
        button.click()

        seconds = 0
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()

    return os.listdir(dirc)

def twiiDownload(dirc=twii_dir):
    if "^TWII.csv" not in os.listdir(dirc):
        chrome_options = Options()
        chrome_options.add_experimental_option("prefs", {
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        driver = webdriver.Chrome(options=chrome_options)
```

```
        url = "https://hk.finance.yahoo.com/quote/%5ETWII/history/"
        driver.get(url)

        wait = WebDriverWait(driver, 1)
        wait.until(EC.presence_of_element_located((By.XPATH, "//
↪section[@data-test='qsp-historical']//table")))

        start_date = "2015-01-01"
        end_date = "2022-12-31"

        date_picker = driver.find_element(By.XPATH, "//div[contains(@class,␣
↪'dateRangeBtn')]")
        date_picker.click()

        start_date_input = driver.find_element(By.XPATH, "//
↪input[@name='startDate']")
        start_date_input.clear()
        start_date_input.send_keys("2015")
        start_date_input.send_keys(Keys.ARROW_RIGHT)
        start_date_input.send_keys("01")
        start_date_input.send_keys(Keys.ARROW_RIGHT)
        start_date_input.send_keys("01")

        end_date_input = driver.find_element(By.XPATH, "//
↪input[@name='endDate']")
        end_date_input.clear()
        end_date_input.send_keys("2022")
        end_date_input.send_keys(Keys.ARROW_RIGHT)
        end_date_input.send_keys("12")
        end_date_input.send_keys(Keys.ARROW_RIGHT)
        end_date_input.send_keys("31")

        apply_button = driver.find_element(By.XPATH, "//button[contains(@class,␣
↪'Bgc($linkColor)') and contains(@class, 'Bdrs(3px)') and contains(@class,␣
↪'Px(20px)') and contains(@class, 'Miw(100px)') and contains(@class,␣
↪'Whs(nw)') and contains(@class, 'Fz(s)') and contains(@class, 'Fw(500)') and␣
↪contains(@class, 'C(white)') and contains(@class, 'Py(9px)')]")
        apply_button.click()

        download_button = driver.find_element(By.XPATH, "//a[@download]")
        download_button.click()

        seconds = 0
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
```

```
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()

    return os.listdir(dirc)

def ixicDownload(dirc=ixic_dir):
    if "^IXIC.csv" not in os.listdir(dirc):
        chrome_options = Options()
        chrome_options.add_experimental_option("prefs", {
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        driver = webdriver.Chrome(options=chrome_options)

        url = "https://hk.finance.yahoo.com/quote/%5EIXIC/history"
        driver.get(url)

        wait = WebDriverWait(driver, 1)
        wait.until(EC.presence_of_element_located((By.XPATH, "//
 ↪section[@data-test='qsp-historical']//table")))

        start_date = "2015-01-01"
        end_date = "2022-12-31"

        date_picker = driver.find_element(By.XPATH, "//div[contains(@class,␣
 ↪'dateRangeBtn')]")
        date_picker.click()

        start_date_input = driver.find_element(By.XPATH, "//
 ↪input[@name='startDate']")
        start_date_input.clear()
        start_date_input.send_keys("2015")
        start_date_input.send_keys(Keys.ARROW_RIGHT)
        start_date_input.send_keys("01")
        start_date_input.send_keys(Keys.ARROW_RIGHT)
        start_date_input.send_keys("01")

        end_date_input = driver.find_element(By.XPATH, "//
 ↪input[@name='endDate']")
        end_date_input.clear()
        end_date_input.send_keys("2022")
```

```
        end_date_input.send_keys(Keys.ARROW_RIGHT)
        end_date_input.send_keys("12")
        end_date_input.send_keys(Keys.ARROW_RIGHT)
        end_date_input.send_keys("31")

        apply_button = driver.find_element(By.XPATH, "//button[contains(@class,␣
 ↪'Bgc($linkColor)') and contains(@class, 'Bdrs(3px)') and contains(@class,␣
 ↪'Px(20px)') and contains(@class, 'Miw(100px)') and contains(@class,␣
 ↪'Whs(nw)') and contains(@class, 'Fz(s)') and contains(@class, 'Fw(500)') and␣
 ↪contains(@class, 'C(white)') and contains(@class, 'Py(9px)')]")
        apply_button.click()

        download_button = driver.find_element(By.XPATH, "//a[@download]")
        download_button.click()

        seconds = 0
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()

    return os.listdir(dirc)

def trsDownload(dirc=trs_dir):
    if "yield-curve-rates-1990-2023.csv" not in os.listdir(dirc):
        chrome_options = Options()
        chrome_options.add_experimental_option("prefs", {
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        driver = webdriver.Chrome(options=chrome_options)

        url = "https://home.treasury.gov/system/files/276/
 ↪yield-curve-rates-1990-2023.csv"
        driver.get(url)

        time.sleep(5)

        seconds = 0
```

```
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()

    return os.listdir(dirc)

def optUnzip(int start_year, int end_year, bint delete=False):
    files = os.listdir(opt_dir)
    for i in range(start_year, end_year + 1):
        zip_file_path = os.path.join(opt_dir, str(i) + "_opt.zip")
        if not any(s.startswith(str(i) + "_opt") and s.endswith(".csv") for s␣
 ↪in files):
            if os.path.exists(zip_file_path):
                with zipfile.ZipFile(zip_file_path, 'r') as zip_ref:
                    zip_ref.extractall(opt_dir)
        if delete and os.path.exists(zip_file_path):
            os.remove(zip_file_path)

    return os.listdir(opt_dir)

def futUnzip(int start_year, int end_year, bint delete=False):
    files = os.listdir(fut_dir)
    for i in range(start_year, end_year + 1):
        zip_file_path = os.path.join(fut_dir, str(i) + "_fut.zip")
        if not any(s.startswith(str(i) + "_fut") and s.endswith(".csv") for s␣
 ↪in files):
            if os.path.exists(zip_file_path):
                with zipfile.ZipFile(zip_file_path, 'r') as zip_ref:
                    zip_ref.extractall(fut_dir)
        if delete and os.path.exists(zip_file_path):
            os.remove(zip_file_path)

    return os.listdir(fut_dir)
```

[5]:
```
optDownload(2015, 2022)
optFiles = optUnzip(2015, 2022, True)
print(optFiles)
```

```
['2015_1_opt.csv', '2015_1_opt_utf8.csv', '2015_1_opt_utf8_popt1.csv',
'2015_1_opt_utf8_popt1_popt2.csv', '2015_2_opt.csv', '2015_2_opt_utf8.csv',
'2015_2_opt_utf8_popt1.csv', '2015_2_opt_utf8_popt1_popt2.csv',
```

'2015_3_opt.csv', '2015_3_opt_utf8.csv', '2015_3_opt_utf8_popt1.csv',
'2015_3_opt_utf8_popt1_popt2.csv', '2015_4_opt.csv', '2015_4_opt_utf8.csv',
'2015_4_opt_utf8_popt1.csv', '2015_4_opt_utf8_popt1_popt2.csv',
'2016_opt_1.csv', '2016_opt_10.csv', '2016_opt_10_utf8.csv',
'2016_opt_10_utf8_popt1.csv', '2016_opt_10_utf8_popt1_popt2.csv',
'2016_opt_11.csv', '2016_opt_11_utf8.csv', '2016_opt_11_utf8_popt1.csv',
'2016_opt_11_utf8_popt1_popt2.csv', '2016_opt_12.csv', '2016_opt_12_utf8.csv',
'2016_opt_12_utf8_popt1.csv', '2016_opt_12_utf8_popt1_popt2.csv',
'2016_opt_1_utf8.csv', '2016_opt_1_utf8_popt1.csv',
'2016_opt_1_utf8_popt1_popt2.csv', '2016_opt_2.csv', '2016_opt_2_utf8.csv',
'2016_opt_2_utf8_popt1.csv', '2016_opt_2_utf8_popt1_popt2.csv',
'2016_opt_3.csv', '2016_opt_3_utf8.csv', '2016_opt_3_utf8_popt1.csv',
'2016_opt_3_utf8_popt1_popt2.csv', '2016_opt_4.csv', '2016_opt_4_utf8.csv',
'2016_opt_4_utf8_popt1.csv', '2016_opt_4_utf8_popt1_popt2.csv',
'2016_opt_5.csv', '2016_opt_5_utf8.csv', '2016_opt_5_utf8_popt1.csv',
'2016_opt_5_utf8_popt1_popt2.csv', '2016_opt_6.csv', '2016_opt_6_utf8.csv',
'2016_opt_6_utf8_popt1.csv', '2016_opt_6_utf8_popt1_popt2.csv',
'2016_opt_7.csv', '2016_opt_7_utf8.csv', '2016_opt_7_utf8_popt1.csv',
'2016_opt_7_utf8_popt1_popt2.csv', '2016_opt_8.csv', '2016_opt_8_utf8.csv',
'2016_opt_8_utf8_popt1.csv', '2016_opt_8_utf8_popt1_popt2.csv',
'2016_opt_9.csv', '2016_opt_9_utf8.csv', '2016_opt_9_utf8_popt1.csv',
'2016_opt_9_utf8_popt1_popt2.csv', '2017_opt_1.csv', '2017_opt_10_1.csv',
'2017_opt_10_1_utf8.csv', '2017_opt_10_1_utf8_popt1.csv',
'2017_opt_10_1_utf8_popt1_popt2.csv', '2017_opt_10_2.csv',
'2017_opt_10_2_utf8.csv', '2017_opt_10_2_utf8_popt1.csv',
'2017_opt_10_2_utf8_popt1_popt2.csv', '2017_opt_11_1.csv',
'2017_opt_11_1_utf8.csv', '2017_opt_11_1_utf8_popt1.csv',
'2017_opt_11_1_utf8_popt1_popt2.csv', '2017_opt_11_2.csv',
'2017_opt_11_2_utf8.csv', '2017_opt_11_2_utf8_popt1.csv',
'2017_opt_11_2_utf8_popt1_popt2.csv', '2017_opt_12_1.csv',
'2017_opt_12_1_utf8.csv', '2017_opt_12_1_utf8_popt1.csv',
'2017_opt_12_1_utf8_popt1_popt2.csv', '2017_opt_12_2.csv',
'2017_opt_12_2_utf8.csv', '2017_opt_12_2_utf8_popt1.csv',
'2017_opt_12_2_utf8_popt1_popt2.csv', '2017_opt_1_utf8.csv',
'2017_opt_1_utf8_popt1.csv', '2017_opt_1_utf8_popt1_popt2.csv',
'2017_opt_2.csv', '2017_opt_2_utf8.csv', '2017_opt_2_utf8_popt1.csv',
'2017_opt_2_utf8_popt1_popt2.csv', '2017_opt_3.csv', '2017_opt_3_utf8.csv',
'2017_opt_3_utf8_popt1.csv', '2017_opt_3_utf8_popt1_popt2.csv',
'2017_opt_4.csv', '2017_opt_4_utf8.csv', '2017_opt_4_utf8_popt1.csv',
'2017_opt_4_utf8_popt1_popt2.csv', '2017_opt_5.csv', '2017_opt_5_utf8.csv',
'2017_opt_5_utf8_popt1.csv', '2017_opt_5_utf8_popt1_popt2.csv',
'2017_opt_6_1.csv', '2017_opt_6_1_utf8.csv', '2017_opt_6_1_utf8_popt1.csv',
'2017_opt_6_1_utf8_popt1_popt2.csv', '2017_opt_6_2.csv',
'2017_opt_6_2_utf8.csv', '2017_opt_6_2_utf8_popt1.csv',
'2017_opt_6_2_utf8_popt1_popt2.csv', '2017_opt_7_1.csv',
'2017_opt_7_1_utf8.csv', '2017_opt_7_1_utf8_popt1.csv',
'2017_opt_7_1_utf8_popt1_popt2.csv', '2017_opt_7_2.csv',
'2017_opt_7_2_utf8.csv', '2017_opt_7_2_utf8_popt1.csv',

```
'2017_opt_7_2_utf8_popt1_popt2.csv', '2017_opt_8_1.csv',
'2017_opt_8_1_utf8.csv', '2017_opt_8_1_utf8_popt1.csv',
'2017_opt_8_1_utf8_popt1_popt2.csv', '2017_opt_8_2.csv',
'2017_opt_8_2_utf8.csv', '2017_opt_8_2_utf8_popt1.csv',
'2017_opt_8_2_utf8_popt1_popt2.csv', '2017_opt_9_1.csv',
'2017_opt_9_1_utf8.csv', '2017_opt_9_1_utf8_popt1.csv',
'2017_opt_9_1_utf8_popt1_popt2.csv', '2017_opt_9_2.csv',
'2017_opt_9_2_utf8.csv', '2017_opt_9_2_utf8_popt1.csv',
'2017_opt_9_2_utf8_popt1_popt2.csv', '2018_opt_01.csv', '2018_opt_01_utf8.csv',
'2018_opt_01_utf8_popt1.csv', '2018_opt_01_utf8_popt1_popt2.csv',
'2018_opt_02.csv', '2018_opt_02_utf8.csv', '2018_opt_02_utf8_popt1.csv',
'2018_opt_02_utf8_popt1_popt2.csv', '2018_opt_03.csv', '2018_opt_03_utf8.csv',
'2018_opt_03_utf8_popt1.csv', '2018_opt_03_utf8_popt1_popt2.csv',
'2018_opt_04.csv', '2018_opt_04_utf8.csv', '2018_opt_04_utf8_popt1.csv',
'2018_opt_04_utf8_popt1_popt2.csv', '2018_opt_05.csv', '2018_opt_05_utf8.csv',
'2018_opt_05_utf8_popt1.csv', '2018_opt_05_utf8_popt1_popt2.csv',
'2018_opt_06.csv', '2018_opt_06_utf8.csv', '2018_opt_06_utf8_popt1.csv',
'2018_opt_06_utf8_popt1_popt2.csv', '2018_opt_07.csv', '2018_opt_07_utf8.csv',
'2018_opt_07_utf8_popt1.csv', '2018_opt_07_utf8_popt1_popt2.csv',
'2018_opt_08.csv', '2018_opt_08_utf8.csv', '2018_opt_08_utf8_popt1.csv',
'2018_opt_08_utf8_popt1_popt2.csv', '2018_opt_09.csv', '2018_opt_09_utf8.csv',
'2018_opt_09_utf8_popt1.csv', '2018_opt_09_utf8_popt1_popt2.csv',
'2018_opt_10.csv', '2018_opt_10_utf8.csv', '2018_opt_10_utf8_popt1.csv',
'2018_opt_10_utf8_popt1_popt2.csv', '2018_opt_11.csv', '2018_opt_11_utf8.csv',
'2018_opt_11_utf8_popt1.csv', '2018_opt_11_utf8_popt1_popt2.csv',
'2018_opt_12.csv', '2018_opt_12_utf8.csv', '2018_opt_12_utf8_popt1.csv',
'2018_opt_12_utf8_popt1_popt2.csv', '2019_opt_01.csv', '2019_opt_01_utf8.csv',
'2019_opt_01_utf8_popt1.csv', '2019_opt_01_utf8_popt1_popt2.csv',
'2019_opt_02.csv', '2019_opt_02_utf8.csv', '2019_opt_02_utf8_popt1.csv',
'2019_opt_02_utf8_popt1_popt2.csv', '2019_opt_03.csv', '2019_opt_03_utf8.csv',
'2019_opt_03_utf8_popt1.csv', '2019_opt_03_utf8_popt1_popt2.csv',
'2019_opt_04.csv', '2019_opt_04_utf8.csv', '2019_opt_04_utf8_popt1.csv',
'2019_opt_04_utf8_popt1_popt2.csv', '2019_opt_05.csv', '2019_opt_05_utf8.csv',
'2019_opt_05_utf8_popt1.csv', '2019_opt_05_utf8_popt1_popt2.csv',
'2019_opt_06.csv', '2019_opt_06_utf8.csv', '2019_opt_06_utf8_popt1.csv',
'2019_opt_06_utf8_popt1_popt2.csv', '2019_opt_07.csv', '2019_opt_07_utf8.csv',
'2019_opt_07_utf8_popt1.csv', '2019_opt_07_utf8_popt1_popt2.csv',
'2019_opt_08.csv', '2019_opt_08_utf8.csv', '2019_opt_08_utf8_popt1.csv',
'2019_opt_08_utf8_popt1_popt2.csv', '2019_opt_09.csv', '2019_opt_09_utf8.csv',
'2019_opt_09_utf8_popt1.csv', '2019_opt_09_utf8_popt1_popt2.csv',
'2019_opt_10.csv', '2019_opt_10_utf8.csv', '2019_opt_10_utf8_popt1.csv',
'2019_opt_10_utf8_popt1_popt2.csv', '2019_opt_11.csv', '2019_opt_11_utf8.csv',
'2019_opt_11_utf8_popt1.csv', '2019_opt_11_utf8_popt1_popt2.csv',
'2019_opt_12.csv', '2019_opt_12_utf8.csv', '2019_opt_12_utf8_popt1.csv',
'2019_opt_12_utf8_popt1_popt2.csv', '2020_opt_01.csv', '2020_opt_01_utf8.csv',
'2020_opt_01_utf8_popt1.csv', '2020_opt_01_utf8_popt1_popt2.csv',
'2020_opt_02.csv', '2020_opt_02_utf8.csv', '2020_opt_02_utf8_popt1.csv',
'2020_opt_02_utf8_popt1_popt2.csv', '2020_opt_03.csv', '2020_opt_03_utf8.csv',
```

'2020_opt_03_utf8_popt1.csv', '2020_opt_03_utf8_popt1_popt2.csv',
'2020_opt_04.csv', '2020_opt_04_utf8.csv', '2020_opt_04_utf8_popt1.csv',
'2020_opt_04_utf8_popt1_popt2.csv', '2020_opt_05.csv', '2020_opt_05_utf8.csv',
'2020_opt_05_utf8_popt1.csv', '2020_opt_05_utf8_popt1_popt2.csv',
'2020_opt_06.csv', '2020_opt_06_utf8.csv', '2020_opt_06_utf8_popt1.csv',
'2020_opt_06_utf8_popt1_popt2.csv', '2020_opt_07.csv', '2020_opt_07_utf8.csv',
'2020_opt_07_utf8_popt1.csv', '2020_opt_07_utf8_popt1_popt2.csv',
'2020_opt_08.csv', '2020_opt_08_utf8.csv', '2020_opt_08_utf8_popt1.csv',
'2020_opt_08_utf8_popt1_popt2.csv', '2020_opt_09.csv', '2020_opt_09_utf8.csv',
'2020_opt_09_utf8_popt1.csv', '2020_opt_09_utf8_popt1_popt2.csv',
'2020_opt_10.csv', '2020_opt_10_utf8.csv', '2020_opt_10_utf8_popt1.csv',
'2020_opt_10_utf8_popt1_popt2.csv', '2020_opt_11.csv', '2020_opt_11_utf8.csv',
'2020_opt_11_utf8_popt1.csv', '2020_opt_11_utf8_popt1_popt2.csv',
'2020_opt_12.csv', '2020_opt_12_utf8.csv', '2020_opt_12_utf8_popt1.csv',
'2020_opt_12_utf8_popt1_popt2.csv', '2021_opt_01.csv', '2021_opt_01_utf8.csv',
'2021_opt_01_utf8_popt1.csv', '2021_opt_01_utf8_popt1_popt2.csv',
'2021_opt_02.csv', '2021_opt_02_utf8.csv', '2021_opt_02_utf8_popt1.csv',
'2021_opt_02_utf8_popt1_popt2.csv', '2021_opt_03.csv', '2021_opt_03_utf8.csv',
'2021_opt_03_utf8_popt1.csv', '2021_opt_03_utf8_popt1_popt2.csv',
'2021_opt_04.csv', '2021_opt_04_utf8.csv', '2021_opt_04_utf8_popt1.csv',
'2021_opt_04_utf8_popt1_popt2.csv', '2021_opt_05.csv', '2021_opt_05_utf8.csv',
'2021_opt_05_utf8_popt1.csv', '2021_opt_05_utf8_popt1_popt2.csv',
'2021_opt_06.csv', '2021_opt_06_utf8.csv', '2021_opt_06_utf8_popt1.csv',
'2021_opt_06_utf8_popt1_popt2.csv', '2021_opt_07.csv', '2021_opt_07_utf8.csv',
'2021_opt_07_utf8_popt1.csv', '2021_opt_07_utf8_popt1_popt2.csv',
'2021_opt_08.csv', '2021_opt_08_utf8.csv', '2021_opt_08_utf8_popt1.csv',
'2021_opt_08_utf8_popt1_popt2.csv', '2021_opt_09.csv', '2021_opt_09_utf8.csv',
'2021_opt_09_utf8_popt1.csv', '2021_opt_09_utf8_popt1_popt2.csv',
'2021_opt_10.csv', '2021_opt_10_utf8.csv', '2021_opt_10_utf8_popt1.csv',
'2021_opt_10_utf8_popt1_popt2.csv', '2021_opt_11.csv', '2021_opt_11_utf8.csv',
'2021_opt_11_utf8_popt1.csv', '2021_opt_11_utf8_popt1_popt2.csv',
'2021_opt_12.csv', '2021_opt_12_utf8.csv', '2021_opt_12_utf8_popt1.csv',
'2021_opt_12_utf8_popt1_popt2.csv', '2022_opt_01.csv', '2022_opt_01_utf8.csv',
'2022_opt_01_utf8_popt1.csv', '2022_opt_01_utf8_popt1_popt2.csv',
'2022_opt_02.csv', '2022_opt_02_utf8.csv', '2022_opt_02_utf8_popt1.csv',
'2022_opt_02_utf8_popt1_popt2.csv', '2022_opt_03.csv', '2022_opt_03_utf8.csv',
'2022_opt_03_utf8_popt1.csv', '2022_opt_03_utf8_popt1_popt2.csv',
'2022_opt_04.csv', '2022_opt_04_utf8.csv', '2022_opt_04_utf8_popt1.csv',
'2022_opt_04_utf8_popt1_popt2.csv', '2022_opt_05.csv', '2022_opt_05_utf8.csv',
'2022_opt_05_utf8_popt1.csv', '2022_opt_05_utf8_popt1_popt2.csv',
'2022_opt_06.csv', '2022_opt_06_utf8.csv', '2022_opt_06_utf8_popt1.csv',
'2022_opt_06_utf8_popt1_popt2.csv', '2022_opt_07.csv', '2022_opt_07_utf8.csv',
'2022_opt_07_utf8_popt1.csv', '2022_opt_07_utf8_popt1_popt2.csv',
'2022_opt_08.csv', '2022_opt_08_utf8.csv', '2022_opt_08_utf8_popt1.csv',
'2022_opt_08_utf8_popt1_popt2.csv', '2022_opt_09.csv', '2022_opt_09_utf8.csv',
'2022_opt_09_utf8_popt1.csv', '2022_opt_09_utf8_popt1_popt2.csv',
'2022_opt_10.csv', '2022_opt_10_utf8.csv', '2022_opt_10_utf8_popt1.csv',
'2022_opt_10_utf8_popt1_popt2.csv', '2022_opt_11.csv', '2022_opt_11_utf8.csv',

```
'2022_opt_11_utf8_popt1.csv', '2022_opt_11_utf8_popt1_popt2.csv',
'2022_opt_12.csv', '2022_opt_12_utf8.csv', '2022_opt_12_utf8_popt1.csv',
'2022_opt_12_utf8_popt1_popt2.csv']
```

[6]:
```
futDownload(2015, 2022)
futFiles = futUnzip(2015, 2022, True)
print(futFiles)
```

```
['2015_fut.csv', '2015_fut_utf8.csv', '2015_fut_utf8_pfut1.csv', '2016_fut.csv',
'2016_fut_utf8.csv', '2016_fut_utf8_pfut1.csv', '2017_fut.csv',
'2017_fut_utf8.csv', '2017_fut_utf8_pfut1.csv', '2018_fut.csv',
'2018_fut_utf8.csv', '2018_fut_utf8_pfut1.csv', '2019_fut.csv',
'2019_fut_utf8.csv', '2019_fut_utf8_pfut1.csv', '2020_fut.csv',
'2020_fut_utf8.csv', '2020_fut_utf8_pfut1.csv', '2021_fut.csv',
'2021_fut_utf8.csv', '2021_fut_utf8_pfut1.csv', '2022_fut.csv',
'2022_fut_utf8.csv', '2022_fut_utf8_pfut1.csv']
```

[7]:
```
%%writefile date.hpp
#ifndef DATE_H
#define DATE_H

#include <iostream>
#include <sstream>
#include <string>
#include <ctime>
#include <iomanip>
using namespace std;

time_t date1(const string& dateStr) {
    tm date = {};
    string dateS = dateStr;
    size_t pos = dateS.find("/");
    date.tm_year = stoi(dateS.substr(0, pos)) - 1900;
    dateS = dateS.substr(pos + 1);
    pos = dateS.find("/");
    date.tm_mon = stoi(dateS.substr(0, pos)) - 1;
    date.tm_mday = stoi(dateS.substr(pos + 1));
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date2(const string& dateStr) {
    tm date = {};
    string dateS = dateStr;
    size_t pos = dateS.find("-");
    date.tm_year = stoi(dateS.substr(0, pos)) - 1900;
```

```
    dateS = dateS.substr(pos + 1);
    pos = dateS.find("-");
    date.tm_mon = stoi(dateS.substr(0, pos)) - 1;
    date.tm_mday = stoi(dateS.substr(pos + 1));
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date3(const string dateStr) {
    tm date = {};
    int year, month;
    char discard;
    stringstream ss(dateStr);
    ss >> year >> discard >> month;
    date.tm_year = year - 1900;
    date.tm_mon = month - 1;
    date.tm_mday = 1;
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date4(const string& dateStr) {
    tm date = {};
    string dateS = dateStr;
    size_t pos = dateS.find("/");
    date.tm_mon = stoi(dateS.substr(0, pos)) - 1;
    dateS = dateS.substr(pos + 1);
    pos = dateS.find("/");
    date.tm_mday = stoi(dateS.substr(0, pos));
    date.tm_year = stoi(dateS.substr(pos + 1)) - 1900;
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date5(const string dateStr) {
    tm date = {};
    stringstream ss(dateStr);
    ss >> get_time(&date, "%Y%m%d");
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
```

```
    return mktime(&date);
}

time_t expirationDiff(const string expStr, time_t date) {
    string expirationStr = expStr;
    if (expirationStr.substr(6, 1) != "W") {
        expirationStr = expirationStr.substr(0, 6) + "W3";
    }
    int year = stoi(expirationStr.substr(0, 4));
    int month = stoi(expirationStr.substr(4, 2));
    int week = stoi(expirationStr.substr(7, 1));
    tm expirationDate = {};
    expirationDate.tm_year = year - 1900;
    expirationDate.tm_mon = month - 1;
    expirationDate.tm_mday = 1;
    mktime(&expirationDate);
    int firstWednesdayOffset = (3 - expirationDate.tm_wday + 7) % 7;
    expirationDate.tm_mday += firstWednesdayOffset + (week - 1) * 7;
    time_t expirationTime = mktime(&expirationDate);
    time_t diff = difftime(expirationTime, date) + 48600;
    if (diff <= 0) {
        int a = stoi(expirationStr.substr(7, 1));
        if (a == 5) return diff;
        expirationStr = expirationStr.substr(0, 7) + to_string(a + 1);
        return expirationDiff(expirationStr, date);
    }
    return diff;
}

void printTm(const tm &date) {
    cout << put_time(&date, "%Y-%m-%d %H:%M:%S") << endl;
}

int test() {
    string dateStr = "2024/05/28";
    string expirationStr = "202405W5";
    cout << expirationDiff(expirationStr, date1(dateStr)) << " seconds␣
 ↪difference" << endl;
    time_t date1Time1 = date1("2022/10/01") + 28800;
    tm date1Result1 = *gmtime(&date1Time1);
    time_t date1Time2 = date1("2022/10/1") + 28800;
    tm date1Result2 = *gmtime(&date1Time2);
    time_t date2Time1 = date2("2022-10-01") + 28800;
    tm date2Result1 = *gmtime(&date2Time1);
    time_t date2Time2 = date2("2022-10-1") + 28800;
    tm date2Result2 = *gmtime(&date2Time2);
    time_t date3Time = date3("2022M10") + 28800;
```

```
    tm date3Result = *gmtime(&date3Time);
    time_t date4Time = date4("10/1/2022") + 28800;
    tm date4Result = *gmtime(&date4Time);
    time_t date5Time = date5("20221001") + 28800;
    tm date5Result = *gmtime(&date5Time);
    printTm(date1Result1);
    printTm(date1Result2);
    printTm(date2Result1);
    printTm(date2Result2);
    printTm(date3Result);
    printTm(date4Result);
    printTm(date5Result);
    return 0;
}

#endif
```

```
Overwriting date.hpp
```

[8]:
```
%%writefile date.cpp
#include <iostream>
#include <sstream>
#include <string>
#include <ctime>
#include <iomanip>
#include <pybind11/pybind11.h>
using namespace std;

time_t date1(const string& dateStr) {
    tm date = {};
    string dateS = dateStr;
    size_t pos = dateS.find("/");
    date.tm_year = stoi(dateS.substr(0, pos)) - 1900;
    dateS = dateS.substr(pos + 1);
    pos = dateS.find("/");
    date.tm_mon = stoi(dateS.substr(0, pos)) - 1;
    date.tm_mday = stoi(dateS.substr(pos + 1));
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date2(const string& dateStr) {
    tm date = {};
    string dateS = dateStr;
    size_t pos = dateS.find("-");
```

```
    date.tm_year = stoi(dateS.substr(0, pos)) - 1900;
    dateS = dateS.substr(pos + 1);
    pos = dateS.find("-");
    date.tm_mon = stoi(dateS.substr(0, pos)) - 1;
    date.tm_mday = stoi(dateS.substr(pos + 1));
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date3(const string dateStr) {
    tm date = {};
    int year, month;
    char discard;
    stringstream ss(dateStr);
    ss >> year >> discard >> month;
    date.tm_year = year - 1900;
    date.tm_mon = month - 1;
    date.tm_mday = 1;
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date4(const string& dateStr) {
    tm date = {};
    string dateS = dateStr;
    size_t pos = dateS.find("/");
    date.tm_mon = stoi(dateS.substr(0, pos)) - 1;
    dateS = dateS.substr(pos + 1);
    pos = dateS.find("/");
    date.tm_mday = stoi(dateS.substr(0, pos));
    date.tm_year = stoi(dateS.substr(pos + 1)) - 1900;
    date.tm_hour = 0;
    date.tm_min = 0;
    date.tm_sec = 0;
    return mktime(&date);
}

time_t date5(const string dateStr) {
    tm date = {};
    stringstream ss(dateStr);
    ss >> get_time(&date, "%Y%m%d");
    date.tm_hour = 0;
    date.tm_min = 0;
```

```
    date.tm_sec = 0;
    return mktime(&date);
}

time_t expirationDiff(const string expStr, time_t date) {
    string expirationStr = expStr;
    if (expirationStr.substr(6, 1) != "W") {
        expirationStr = expirationStr.substr(0, 6) + "W3";
    }
    int year = stoi(expirationStr.substr(0, 4));
    int month = stoi(expirationStr.substr(4, 2));
    int week = stoi(expirationStr.substr(7, 1));
    tm expirationDate = {};
    expirationDate.tm_year = year - 1900;
    expirationDate.tm_mon = month - 1;
    expirationDate.tm_mday = 1;
    mktime(&expirationDate);
    int firstWednesdayOffset = (3 - expirationDate.tm_wday + 7) % 7;
    expirationDate.tm_mday += firstWednesdayOffset + (week - 1) * 7;
    time_t expirationTime = mktime(&expirationDate);
    time_t diff = difftime(expirationTime, date) + 48600;
    if (diff <= 0) {
        int a = stoi(expirationStr.substr(7, 1));
        if (a == 5) return diff;
        expirationStr = expirationStr.substr(0, 7) + to_string(a + 1);
        return expirationDiff(expirationStr, date);
    }
    return diff;
}

void printTm(const tm &date) {
    cout << put_time(&date, "%Y-%m-%d %H:%M:%S") << endl;
}

int test() {
    string dateStr = "2024/05/28";
    string expirationStr = "202405W5";
    cout << expirationDiff(expirationStr, date1(dateStr)) << " seconds␣
 ↪difference" << endl;
    time_t date1Time1 = date1("2022/10/01") + 28800;
    tm date1Result1 = *gmtime(&date1Time1);
    time_t date1Time2 = date1("2022/10/1") + 28800;
    tm date1Result2 = *gmtime(&date1Time2);
    time_t date2Time1 = date2("2022-10-01") + 28800;
    tm date2Result1 = *gmtime(&date2Time1);
    time_t date2Time2 = date2("2022-10-1") + 28800;
    tm date2Result2 = *gmtime(&date2Time2);
```

```
    time_t date3Time = date3("2022M10") + 28800;
    tm date3Result = *gmtime(&date3Time);
    time_t date4Time = date4("10/1/2022") + 28800;
    tm date4Result = *gmtime(&date4Time);
    time_t date5Time = date5("20221001") + 28800;
    tm date5Result = *gmtime(&date5Time);
    printTm(date1Result1);
    printTm(date1Result2);
    printTm(date2Result1);
    printTm(date2Result2);
    printTm(date3Result);
    printTm(date4Result);
    printTm(date5Result);
    return 0;
}

PYBIND11_MODULE(date, m) {
    m.def("date1", &date1);
    m.def("date2", &date2);
    m.def("date3", &date3);
    m.def("date4", &date4);
    m.def("date5", &date5);
    m.def("test", &test);
    m.def("expirationDiff", &expirationDiff);
}
```

```
Overwriting date.cpp
```

[9]:
```
%%writefile date_setup.py
from setuptools import setup, Extension
import pybind11

include_dirs = [pybind11.get_include()]

date_module = Extension(
    'date',
    sources=['date.cpp'],
    include_dirs=include_dirs,
    language='c++'
)

setup(
    name='date',
    ext_modules=[date_module],
    zip_safe=False,
)
```

```
Overwriting date_setup.py
```

[10]:
```
!python date_setup.py build
!python date_setup.py install
```

```
running build
running build_ext
building 'date' extension
"C:\Program Files\Microsoft Visual
Studio\2022\Community\VC\Tools\MSVC\14.39.33519\bin\HostX86\x64\cl.exe" /c
/nologo /O2 /W3 /GL /DNDEBUG /MD -IC:\Pycharm\Python310\Python\lib\site-
packages\pybind11\include -IC:\Pycharm\Python310\Python\include "-IC:\Program
Files\Python310\include" "-IC:\Program Files\Python310\Include" "-IC:\Program
Files\Microsoft Visual Studio\2022\Community\VC\Tools\MSVC\14.39.33519\include"
"-IC:\Program Files\Microsoft Visual
Studio\2022\Community\VC\Tools\MSVC\14.39.33519\ATLMFC\include" "-IC:\Program
Files\Microsoft Visual Studio\2022\Community\VC\Auxiliary\VS\include"
"-IC:\Program Files (x86)\Windows Kits\10\include\10.0.22621.0\ucrt"
"-IC:\Program Files (x86)\Windows Kits\10\\include\10.0.22621.0\\um"
"-IC:\Program Files (x86)\Windows Kits\10\\include\10.0.22621.0\\shared"
"-IC:\Program Files (x86)\Windows Kits\10\\include\10.0.22621.0\\winrt"
"-IC:\Program Files (x86)\Windows Kits\10\\include\10.0.22621.0\\cppwinrt" /EHsc
/Tpdate.cpp /Fobuild\temp.win-amd64-cpython-310\Release\date.obj
date.cpp
date.cpp(95): warning C4244: '   ':   'double'    'time_t'
date.cpp(114): warning C4996: 'gmtime': This function or variable may be unsafe.
Consider using gmtime_s instead. To disable deprecation, use
_CRT_SECURE_NO_WARNINGS. See online help for details.
date.cpp(116): warning C4996: 'gmtime': This function or variable may be unsafe.
Consider using gmtime_s instead. To disable deprecation, use
_CRT_SECURE_NO_WARNINGS. See online help for details.
date.cpp(118): warning C4996: 'gmtime': This function or variable may be unsafe.
Consider using gmtime_s instead. To disable deprecation, use
_CRT_SECURE_NO_WARNINGS. See online help for details.
date.cpp(120): warning C4996: 'gmtime': This function or variable may be unsafe.
Consider using gmtime_s instead. To disable deprecation, use
_CRT_SECURE_NO_WARNINGS. See online help for details.
date.cpp(122): warning C4996: 'gmtime': This function or variable may be unsafe.
Consider using gmtime_s instead. To disable deprecation, use
_CRT_SECURE_NO_WARNINGS. See online help for details.
date.cpp(124): warning C4996: 'gmtime': This function or variable may be unsafe.
Consider using gmtime_s instead. To disable deprecation, use
_CRT_SECURE_NO_WARNINGS. See online help for details.
date.cpp(126): warning C4996: 'gmtime': This function or variable may be unsafe.
Consider using gmtime_s instead. To disable deprecation, use
_CRT_SECURE_NO_WARNINGS. See online help for details.
"C:\Program Files\Microsoft Visual
Studio\2022\Community\VC\Tools\MSVC\14.39.33519\bin\HostX86\x64\link.exe"
/nologo /INCREMENTAL:NO /LTCG /DLL /MANIFEST:EMBED,ID=2 /MANIFESTUAC:NO
/LIBPATH:C:\Pycharm\Python310\Python\libs "/LIBPATH:C:\Program
```

```
Files\Python310\libs" "/LIBPATH:C:\Program Files\Python310"
/LIBPATH:C:\Pycharm\Python310\Python\PCbuild\amd64 "/LIBPATH:C:\Program
Files\Microsoft Visual
Studio\2022\Community\VC\Tools\MSVC\14.39.33519\ATLMFC\lib\x64"
"/LIBPATH:C:\Program Files\Microsoft Visual
Studio\2022\Community\VC\Tools\MSVC\14.39.33519\lib\x64" "/LIBPATH:C:\Program
Files (x86)\Windows Kits\10\lib\10.0.22621.0\ucrt\x64" "/LIBPATH:C:\Program
Files (x86)\Windows Kits\10\\lib\10.0.22621.0\\um\x64" /EXPORT:PyInit_date
build\temp.win-amd64-cpython-310\Release\date.obj /OUT:build\lib.win-
amd64-cpython-310\date.cp310-win_amd64.pyd /IMPLIB:build\temp.win-
amd64-cpython-310\Release\date.cp310-win_amd64.lib
       build\temp.win-amd64-cpython-310\Release\date.cp310-win_amd64.lib
build\temp.win-amd64-cpython-310\Release\date.cp310-win_amd64.exp


running install
running bdist_egg
running egg_info
writing date.egg-info\PKG-INFO
writing dependency_links to date.egg-info\dependency_links.txt
writing top-level names to date.egg-info\top_level.txt
reading manifest file 'date.egg-info\SOURCES.txt'
writing manifest file 'date.egg-info\SOURCES.txt'
installing library code to build\bdist.win-amd64\egg
running install_lib
running build_ext
creating build\bdist.win-amd64\egg
copying build\lib.win-amd64-cpython-310\date.cp310-win_amd64.pyd ->
build\bdist.win-amd64\egg
creating stub loader for date.cp310-win_amd64.pyd
byte-compiling build\bdist.win-amd64\egg\date.py to date.cpython-310.pyc
creating build\bdist.win-amd64\egg\EGG-INFO
copying date.egg-info\PKG-INFO -> build\bdist.win-amd64\egg\EGG-INFO
copying date.egg-info\SOURCES.txt -> build\bdist.win-amd64\egg\EGG-INFO
copying date.egg-info\dependency_links.txt -> build\bdist.win-amd64\egg\EGG-INFO
copying date.egg-info\not-zip-safe -> build\bdist.win-amd64\egg\EGG-INFO
copying date.egg-info\top_level.txt -> build\bdist.win-amd64\egg\EGG-INFO
writing build\bdist.win-amd64\egg\EGG-INFO\native_libs.txt
creating 'dist\date-0.0.0-py3.10-win-amd64.egg' and adding 'build\bdist.win-
amd64\egg' to it
removing 'build\bdist.win-amd64\egg' (and everything under it)
Processing date-0.0.0-py3.10-win-amd64.egg
removing 'c:\pycharm\python310\python\lib\site-packages\date-0.0.0-py3.10-win-
amd64.egg' (and everything under it)
creating c:\pycharm\python310\python\lib\site-packages\date-0.0.0-py3.10-win-
amd64.egg
Extracting date-0.0.0-py3.10-win-amd64.egg to
c:\pycharm\python310\python\lib\site-packages
```

```
Adding date 0.0.0 to easy-install.pth file

Installed c:\pycharm\python310\python\lib\site-packages\date-0.0.0-py3.10-win-
amd64.egg
Processing dependencies for date==0.0.0
Finished processing dependencies for date==0.0.0

C:\Pycharm\Python310\Python\lib\site-packages\setuptools\_distutils\cmd.py:66:
SetuptoolsDeprecationWarning: setup.py install is deprecated.
!!

********************************************************************************
        Please avoid running ``setup.py`` directly.
        Instead, use pypa/build, pypa/installer or other
        standards-based tools.

        See https://blog.ganssle.io/articles/2021/10/setup-py-deprecated.html
for details.
********************************************************************************

!!
  self.initialize_options()
C:\Pycharm\Python310\Python\lib\site-packages\setuptools\_distutils\cmd.py:66:
EasyInstallDeprecationWarning: easy_install command is deprecated.
!!

********************************************************************************
        Please avoid running ``setup.py`` and ``easy_install``.
        Instead, use pypa/build, pypa/installer or other
        standards-based tools.

        See https://github.com/pypa/setuptools/issues/917 for details.
********************************************************************************

!!
  self.initialize_options()
```

[11]:
```python
import codecs
import csv

def convert(inp, out):
    with codecs.open(inp, 'r', encoding='big5') as infile:
        with codecs.open(out, 'w', encoding='utf-8') as outfile:
            outfile.write(infile.read())
            print("ok", end = " ")

def convertS(inp, out):
    with codecs.open(inp, 'r', encoding='utf-8-sig') as infile:
```

```
        with codecs.open(out, 'w', encoding='utf-8') as outfile:
            outfile.write(infile.read())
            print("ok", end = " ")

optFiles = [item for item in optFiles if "_utf8" not in item]
popt_dir = "downloads/option/"
newOptFiles = []
for idx, file_name in enumerate(optFiles):
    convert(popt_dir + file_name, popt_dir + file_name[:-4] + "_utf8" +␣
 ↪file_name[-4:])
    newOptFiles.append(file_name[:-4] + "_utf8" + file_name[-4:])
print()
print(newOptFiles)
```

```
ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok
ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok
ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok ok
ok ok ok ok ok ok ok ok ok ok ok ok ok ok
['2015_1_opt_utf8.csv', '2015_2_opt_utf8.csv', '2015_3_opt_utf8.csv',
'2015_4_opt_utf8.csv', '2016_opt_1_utf8.csv', '2016_opt_10_utf8.csv',
'2016_opt_11_utf8.csv', '2016_opt_12_utf8.csv', '2016_opt_2_utf8.csv',
'2016_opt_3_utf8.csv', '2016_opt_4_utf8.csv', '2016_opt_5_utf8.csv',
'2016_opt_6_utf8.csv', '2016_opt_7_utf8.csv', '2016_opt_8_utf8.csv',
'2016_opt_9_utf8.csv', '2017_opt_1_utf8.csv', '2017_opt_10_1_utf8.csv',
'2017_opt_10_2_utf8.csv', '2017_opt_11_1_utf8.csv', '2017_opt_11_2_utf8.csv',
'2017_opt_12_1_utf8.csv', '2017_opt_12_2_utf8.csv', '2017_opt_2_utf8.csv',
'2017_opt_3_utf8.csv', '2017_opt_4_utf8.csv', '2017_opt_5_utf8.csv',
'2017_opt_6_1_utf8.csv', '2017_opt_6_2_utf8.csv', '2017_opt_7_1_utf8.csv',
'2017_opt_7_2_utf8.csv', '2017_opt_8_1_utf8.csv', '2017_opt_8_2_utf8.csv',
'2017_opt_9_1_utf8.csv', '2017_opt_9_2_utf8.csv', '2018_opt_01_utf8.csv',
'2018_opt_02_utf8.csv', '2018_opt_03_utf8.csv', '2018_opt_04_utf8.csv',
'2018_opt_05_utf8.csv', '2018_opt_06_utf8.csv', '2018_opt_07_utf8.csv',
'2018_opt_08_utf8.csv', '2018_opt_09_utf8.csv', '2018_opt_10_utf8.csv',
'2018_opt_11_utf8.csv', '2018_opt_12_utf8.csv', '2019_opt_01_utf8.csv',
'2019_opt_02_utf8.csv', '2019_opt_03_utf8.csv', '2019_opt_04_utf8.csv',
'2019_opt_05_utf8.csv', '2019_opt_06_utf8.csv', '2019_opt_07_utf8.csv',
'2019_opt_08_utf8.csv', '2019_opt_09_utf8.csv', '2019_opt_10_utf8.csv',
'2019_opt_11_utf8.csv', '2019_opt_12_utf8.csv', '2020_opt_01_utf8.csv',
'2020_opt_02_utf8.csv', '2020_opt_03_utf8.csv', '2020_opt_04_utf8.csv',
'2020_opt_05_utf8.csv', '2020_opt_06_utf8.csv', '2020_opt_07_utf8.csv',
'2020_opt_08_utf8.csv', '2020_opt_09_utf8.csv', '2020_opt_10_utf8.csv',
'2020_opt_11_utf8.csv', '2020_opt_12_utf8.csv', '2021_opt_01_utf8.csv',
'2021_opt_02_utf8.csv', '2021_opt_03_utf8.csv', '2021_opt_04_utf8.csv',
'2021_opt_05_utf8.csv', '2021_opt_06_utf8.csv', '2021_opt_07_utf8.csv',
'2021_opt_08_utf8.csv', '2021_opt_09_utf8.csv', '2021_opt_10_utf8.csv',
'2021_opt_11_utf8.csv', '2021_opt_12_utf8.csv', '2022_opt_01_utf8.csv',
'2022_opt_02_utf8.csv', '2022_opt_03_utf8.csv', '2022_opt_04_utf8.csv',
'2022_opt_05_utf8.csv', '2022_opt_06_utf8.csv', '2022_opt_07_utf8.csv',
```

```
'2022_opt_08_utf8.csv', '2022_opt_09_utf8.csv', '2022_opt_10_utf8.csv',
'2022_opt_11_utf8.csv', '2022_opt_12_utf8.csv']
```

[12]:
```
with open("opt_files.csv", "w") as opt:
    for i in newOptFiles:
        opt.write(i + "\n")
```

[13]:
```
%%writefile popt.cpp
#include <iostream>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>
#include <fstream>
#include "date.hpp"

using namespace std;

string popt1_15(const string& csvContent)
{
    istringstream inputFile(csvContent);
    ostringstream outputFile;
    string line;
    bool c = true;

    while (getline(inputFile, line))
    {
        istringstream iss(line);
        vector<string> sl;
        string token;
        while (getline(iss, token, ','))
        {
            sl.push_back(token);
        }

        if (sl[1] == "TXO")
        {
            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl[1].erase(remove(sl[1].begin(), sl[1].end(), ' '), sl[1].end());

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
```

```
            }
            oss << sl[sl.size() - 1];
            outputFile << oss.str() << "\n";
        }
        else if (c == true)
        {
            c = false;

            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl.erase(sl.end() - 1);

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1];
            outputFile << oss.str() << "\n";
        }
    }
        string o = outputFile.str();
    o.erase(o.size() - 1);
    return o;
}

string popt1_16(const string& csvContent)
{
    istringstream inputFile(csvContent);
    ostringstream outputFile;
    string line;
    bool c = true;

    while (getline(inputFile, line))
    {
        istringstream iss(line);
        vector<string> sl;
        string token;
        while (getline(iss, token, ','))
        {
            sl.push_back(token);
        }

        if (sl[1] == "TXO")
        {
```

```
            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl[1].erase(remove(sl[1].begin(), sl[1].end(), ' '), sl[1].end());

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
        else if (c)
        {
            c = false;

            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl.erase(sl.end() - 1);

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
    }
        string o = outputFile.str();
    o.erase(o.size() - 1);
    return o;
}

string popt1_17(const string& csvContent)
{
    istringstream inputFile(csvContent);
    ostringstream outputFile;
    string line;
    bool c = true;

    while (getline(inputFile, line))
    {
```

```
        istringstream iss(line);
        vector<string> sl;
        string token;
        while (getline(iss, token, ','))
        {
            sl.push_back(token);
        }

        if (sl[1] == "TXO")
        {
            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl[1].erase(remove(sl[1].begin(), sl[1].end(), ' '), sl[1].end());
            sl.erase(sl.end() - 2);

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
        else if (c)
        {
            c = false;

            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl.erase(sl.end() - 2);

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
    }
        string o = outputFile.str();
    o.erase(o.size() - 1);
    return o;
```

```
}

string popt1_18t21(const string& csvContent)
{
    istringstream inputFile(csvContent);
    ostringstream outputFile;
    string line;
    bool c = true;

    while (getline(inputFile, line))
    {
        istringstream iss(line);
        vector<string> sl;
        string token;
        while (getline(iss, token, ','))
        {
            sl.push_back(token);
        }

        if (sl[1] == "TXO")
        {
            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl[1].erase(remove(sl[1].begin(), sl[1].end(), ' '), sl[1].end());
            sl.erase(sl.end() - 2);

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
        else if (c)
        {
            c = false;

            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7);
            sl.erase(sl.begin() + 8, sl.begin() + 13);
            sl.erase(sl.end() - 2);

            ostringstream oss;
```

```
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
    }
        string o = outputFile.str();
    o.erase(o.size() - 1);
    return o;
}

string popt1_22(const string& csvContent)
{
    istringstream inputFile(csvContent);
    ostringstream outputFile;
    string line;
    bool c = true;

    while (getline(inputFile, line))
    {
        istringstream iss(line);
        vector<string> sl;
        while (getline(iss, line, ','))
        {
            sl.push_back(line);
        }

        if (sl[1] == "TXO")
        {
                sl.erase(sl.begin() + 1);
                    sl.erase(sl.begin() + 5, sl.begin() + 7);
                    sl.erase(sl.begin() + 7, sl.begin() + 14);
                    sl.erase(sl.begin() + 8, sl.end());

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
        else if (c)
        {
            c = false;
```

```
            sl.erase(sl.begin() + 1);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 7, sl.begin() + 14);
            sl.erase(sl.begin() + 8, sl.end());

            ostringstream oss;
            for (size_t k = 0; k < sl.size() - 1; ++k)
            {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
            outputFile << oss.str();
        }
    }
        string o = outputFile.str();
    o.erase(o.size() - 1);
    return o;
}

vector<string> popt1(const string &dirc, const vector<string> &pfiles) {
    vector<string> files = pfiles;
    for (int index = 0; index < files.size(); ++index) {
        string &i = files[index];
        ifstream file(dirc + i);
        string content;

        if (file.is_open()) {
            string line;
            while (getline(file, line)) {
                content += line + "\n";
            }
            file.close();
            content.pop_back();
        } else {
            cerr << "Unable to open file: " << i << endl;
            vector<string> e = {"ife"};
            return e;
        }

        int y = stoi(i.substr(2, 2));
        i.insert(i.size() - 4, "_popt1");
        ofstream output_file(dirc + i);
        if (output_file.is_open()) {
            if (y == 15) {
                output_file << popt1_15(content);
            } else if (y == 16) {
```

```
                output_file << popt1_16(content);
            } else if (y == 17) {
                output_file << popt1_17(content);
            } else if (y >= 18 && y <= 21) {
                output_file << popt1_18t21(content);
            } else if (y == 22) {
                output_file << popt1_22(content);
            } else {
                cerr << "File name out of scope: " << i << endl;
                i = i.substr(1);
                files.push_back("ine");
                return files;
            }
            output_file.close();
        } else {
            cerr << "Unable to create output file: " << i << endl;
            i = i.substr(1);
            files.push_back("ofe");
            return files;
        }
    }
    return files;
}

string popt2_15t16(const string &csvContent)
{
        istringstream inputFile(csvContent);
        ostringstream outputFile;
        string line;
        bool c = true;

        while (getline(inputFile, line))
        {
                istringstream iss(line);
                vector<string> sl;
                string token;
                while (getline(iss, token, ','))
                {
                        sl.push_back(token);
                }

                if (c == true)
                {
                        c = false;
                        ostringstream oss;
                        for (size_t k = 0; k < sl.size() - 1; ++k)
                        {
```

```
                                oss << sl[k] << ",";
                        }
                        oss << sl[sl.size() - 1];
                        outputFile << oss.str() << "\n";
                }
                else
                {
                        time_t date = date1(sl[0]);
                        sl[0] = to_string(date);
                        time_t diff = expirationDiff(sl[1], date);
                        sl[1] = to_string(diff);
                        sl[3] = (sl[3] == " ") ? "1" : "0";
                        ostringstream oss;
                        for (size_t k = 0; k < sl.size() - 1; ++k)
                        {
                                oss << sl[k] << ",";
                        }
                        oss << sl[sl.size() - 1];
                        outputFile << oss.str() << "\n";
                }
        }
        string o = outputFile.str();
    if (!o.empty()) o.erase(o.size() - 1);
    return o;
}

string popt2_17t22(const string &csvContent)
{
        istringstream inputFile(csvContent);
        ostringstream outputFile;
        string line;
        bool c = true;

        while (getline(inputFile, line))
        {
                istringstream iss(line);
                vector<string> sl;
                string token;
                while (getline(iss, token, ','))
                {
                        sl.push_back(token);
                }

                if (c == true)
                {
                        c = false;
                        sl.erase(sl.end() - 1);
```

```
            for (auto i : sl) {
                i.erase(remove(i.begin(), i.end(), ' '), i.end());
                i.erase(remove(i.begin(), i.end(), '-'), i.end());
            }
                        ostringstream oss;
                        for (size_t k = 0; k < sl.size() - 1; ++k)
                        {
                                oss << sl[k] << ",";
                        }
                        oss << sl[sl.size() - 1];
                        outputFile << oss.str() << "\n";
                }
                else if (sl.back().find(" ") != string::npos)
                {
                        time_t date = date1(sl[0]);
                        sl[0] = to_string(date);
                        time_t diff = expirationDiff(sl[1], date);
                        sl[1] = to_string(diff);
                        sl[3] = (sl[3] == " ") ? "1" : "0";
                        ostringstream oss;
            sl.erase(sl.end() - 1);
                        for (size_t k = 0; k < sl.size() - 1; ++k)
                        {
                                if (sl[k] == "-" || sl[k] == "")
                                {
                                        sl[k] = "0";
                                }
                                oss << sl[k] << ",";
                        }
                        oss << sl[sl.size() - 1];
                        outputFile << oss.str() << "\n";
                }
        }
        string o = outputFile.str();
    if (!o.empty()) o.erase(o.size() - 1);
    return o;
}

vector<string> popt2(const string &dirc, const vector<string> &pfiles) {
    vector<string> files = pfiles;
    for (int index = 0; index < files.size(); ++index) {
        string &i = files[index];
        ifstream file(dirc + i);
        string content;

        if (file.is_open()) {
            string line;
```

```
            while (getline(file, line)) {
                content += line + "\n";
            }
            file.close();
            content.pop_back();
        } else {
            cerr << "Unable to open file: " << i << endl;
            vector<string> e = {"ife"};
            return e;
        }

        int y = stoi(i.substr(2, 2));
        i.insert(i.size() - 4, "_popt2");
        ofstream output_file(dirc + i);
        if (output_file.is_open()) {
            if (y >= 15 && y <= 16) {
                output_file << popt2_15t16(content);
            } else if (y >= 17 && y <= 22) {
                output_file << popt2_17t22(content);
            } else {
                cerr << "File name out of scope: " << i << endl;
                i = i.substr(1);
                files.push_back("ine");
                return files;
            }
            output_file.close();
        } else {
            cerr << "Unable to create output file: " << i << endl;
            i = i.substr(1);
            files.push_back("ofe");
            return files;
        }
    }
    return files;
}

int main()
{
    ifstream opt("opt_files.csv");
        vector<string> files;
    string line;
    while (getline(opt, line)) {
        files.push_back(line);
    }
    opt.close();
        string dirc = "downloads/option/";
    files = popt1(dirc, files);
```

```
    for (auto i : files){
        cout << i << endl;
    }
    files = popt2(dirc, files);
    for (auto i : files){
        cout << i << endl;
    }
    return 0;
}
```

```
Overwriting popt.cpp
```

[14]:
```
!g++ popt.cpp -o popt.exe
```

[15]:
```
!popt.exe
```

```
2015_1_opt_utf8_popt1.csv
2015_2_opt_utf8_popt1.csv
2015_3_opt_utf8_popt1.csv
2015_4_opt_utf8_popt1.csv
2016_opt_1_utf8_popt1.csv
2016_opt_10_utf8_popt1.csv
2016_opt_11_utf8_popt1.csv
2016_opt_12_utf8_popt1.csv
2016_opt_2_utf8_popt1.csv
2016_opt_3_utf8_popt1.csv
2016_opt_4_utf8_popt1.csv
2016_opt_5_utf8_popt1.csv
2016_opt_6_utf8_popt1.csv
2016_opt_7_utf8_popt1.csv
2016_opt_8_utf8_popt1.csv
2016_opt_9_utf8_popt1.csv
2017_opt_1_utf8_popt1.csv
2017_opt_10_1_utf8_popt1.csv
2017_opt_10_2_utf8_popt1.csv
2017_opt_11_1_utf8_popt1.csv
2017_opt_11_2_utf8_popt1.csv
2017_opt_12_1_utf8_popt1.csv
2017_opt_12_2_utf8_popt1.csv
2017_opt_2_utf8_popt1.csv
2017_opt_3_utf8_popt1.csv
2017_opt_4_utf8_popt1.csv
2017_opt_5_utf8_popt1.csv
2017_opt_6_1_utf8_popt1.csv
2017_opt_6_2_utf8_popt1.csv
2017_opt_7_1_utf8_popt1.csv
2017_opt_7_2_utf8_popt1.csv
2017_opt_8_1_utf8_popt1.csv
```

2017_opt_8_2_utf8_popt1.csv
2017_opt_9_1_utf8_popt1.csv
2017_opt_9_2_utf8_popt1.csv
2018_opt_01_utf8_popt1.csv
2018_opt_02_utf8_popt1.csv
2018_opt_03_utf8_popt1.csv
2018_opt_04_utf8_popt1.csv
2018_opt_05_utf8_popt1.csv
2018_opt_06_utf8_popt1.csv
2018_opt_07_utf8_popt1.csv
2018_opt_08_utf8_popt1.csv
2018_opt_09_utf8_popt1.csv
2018_opt_10_utf8_popt1.csv
2018_opt_11_utf8_popt1.csv
2018_opt_12_utf8_popt1.csv
2019_opt_01_utf8_popt1.csv
2019_opt_02_utf8_popt1.csv
2019_opt_03_utf8_popt1.csv
2019_opt_04_utf8_popt1.csv
2019_opt_05_utf8_popt1.csv
2019_opt_06_utf8_popt1.csv
2019_opt_07_utf8_popt1.csv
2019_opt_08_utf8_popt1.csv
2019_opt_09_utf8_popt1.csv
2019_opt_10_utf8_popt1.csv
2019_opt_11_utf8_popt1.csv
2019_opt_12_utf8_popt1.csv
2020_opt_01_utf8_popt1.csv
2020_opt_02_utf8_popt1.csv
2020_opt_03_utf8_popt1.csv
2020_opt_04_utf8_popt1.csv
2020_opt_05_utf8_popt1.csv
2020_opt_06_utf8_popt1.csv
2020_opt_07_utf8_popt1.csv
2020_opt_08_utf8_popt1.csv
2020_opt_09_utf8_popt1.csv
2020_opt_10_utf8_popt1.csv
2020_opt_11_utf8_popt1.csv
2020_opt_12_utf8_popt1.csv
2021_opt_01_utf8_popt1.csv
2021_opt_02_utf8_popt1.csv
2021_opt_03_utf8_popt1.csv
2021_opt_04_utf8_popt1.csv
2021_opt_05_utf8_popt1.csv
2021_opt_06_utf8_popt1.csv
2021_opt_07_utf8_popt1.csv
2021_opt_08_utf8_popt1.csv
2021_opt_09_utf8_popt1.csv

2021_opt_10_utf8_popt1.csv
2021_opt_11_utf8_popt1.csv
2021_opt_12_utf8_popt1.csv
2022_opt_01_utf8_popt1.csv
2022_opt_02_utf8_popt1.csv
2022_opt_03_utf8_popt1.csv
2022_opt_04_utf8_popt1.csv
2022_opt_05_utf8_popt1.csv
2022_opt_06_utf8_popt1.csv
2022_opt_07_utf8_popt1.csv
2022_opt_08_utf8_popt1.csv
2022_opt_09_utf8_popt1.csv
2022_opt_10_utf8_popt1.csv
2022_opt_11_utf8_popt1.csv
2022_opt_12_utf8_popt1.csv
2015_1_opt_utf8_popt1_popt2.csv
2015_2_opt_utf8_popt1_popt2.csv
2015_3_opt_utf8_popt1_popt2.csv
2015_4_opt_utf8_popt1_popt2.csv
2016_opt_1_utf8_popt1_popt2.csv
2016_opt_10_utf8_popt1_popt2.csv
2016_opt_11_utf8_popt1_popt2.csv
2016_opt_12_utf8_popt1_popt2.csv
2016_opt_2_utf8_popt1_popt2.csv
2016_opt_3_utf8_popt1_popt2.csv
2016_opt_4_utf8_popt1_popt2.csv
2016_opt_5_utf8_popt1_popt2.csv
2016_opt_6_utf8_popt1_popt2.csv
2016_opt_7_utf8_popt1_popt2.csv
2016_opt_8_utf8_popt1_popt2.csv
2016_opt_9_utf8_popt1_popt2.csv
2017_opt_1_utf8_popt1_popt2.csv
2017_opt_10_1_utf8_popt1_popt2.csv
2017_opt_10_2_utf8_popt1_popt2.csv
2017_opt_11_1_utf8_popt1_popt2.csv
2017_opt_11_2_utf8_popt1_popt2.csv
2017_opt_12_1_utf8_popt1_popt2.csv
2017_opt_12_2_utf8_popt1_popt2.csv
2017_opt_2_utf8_popt1_popt2.csv
2017_opt_3_utf8_popt1_popt2.csv
2017_opt_4_utf8_popt1_popt2.csv
2017_opt_5_utf8_popt1_popt2.csv
2017_opt_6_1_utf8_popt1_popt2.csv
2017_opt_6_2_utf8_popt1_popt2.csv
2017_opt_7_1_utf8_popt1_popt2.csv
2017_opt_7_2_utf8_popt1_popt2.csv
2017_opt_8_1_utf8_popt1_popt2.csv
2017_opt_8_2_utf8_popt1_popt2.csv

2017_opt_9_1_utf8_popt1_popt2.csv
2017_opt_9_2_utf8_popt1_popt2.csv
2018_opt_01_utf8_popt1_popt2.csv
2018_opt_02_utf8_popt1_popt2.csv
2018_opt_03_utf8_popt1_popt2.csv
2018_opt_04_utf8_popt1_popt2.csv
2018_opt_05_utf8_popt1_popt2.csv
2018_opt_06_utf8_popt1_popt2.csv
2018_opt_07_utf8_popt1_popt2.csv
2018_opt_08_utf8_popt1_popt2.csv
2018_opt_09_utf8_popt1_popt2.csv
2018_opt_10_utf8_popt1_popt2.csv
2018_opt_11_utf8_popt1_popt2.csv
2018_opt_12_utf8_popt1_popt2.csv
2019_opt_01_utf8_popt1_popt2.csv
2019_opt_02_utf8_popt1_popt2.csv
2019_opt_03_utf8_popt1_popt2.csv
2019_opt_04_utf8_popt1_popt2.csv
2019_opt_05_utf8_popt1_popt2.csv
2019_opt_06_utf8_popt1_popt2.csv
2019_opt_07_utf8_popt1_popt2.csv
2019_opt_08_utf8_popt1_popt2.csv
2019_opt_09_utf8_popt1_popt2.csv
2019_opt_10_utf8_popt1_popt2.csv
2019_opt_11_utf8_popt1_popt2.csv
2019_opt_12_utf8_popt1_popt2.csv
2020_opt_01_utf8_popt1_popt2.csv
2020_opt_02_utf8_popt1_popt2.csv
2020_opt_03_utf8_popt1_popt2.csv
2020_opt_04_utf8_popt1_popt2.csv
2020_opt_05_utf8_popt1_popt2.csv
2020_opt_06_utf8_popt1_popt2.csv
2020_opt_07_utf8_popt1_popt2.csv
2020_opt_08_utf8_popt1_popt2.csv
2020_opt_09_utf8_popt1_popt2.csv
2020_opt_10_utf8_popt1_popt2.csv
2020_opt_11_utf8_popt1_popt2.csv
2020_opt_12_utf8_popt1_popt2.csv
2021_opt_01_utf8_popt1_popt2.csv
2021_opt_02_utf8_popt1_popt2.csv
2021_opt_03_utf8_popt1_popt2.csv
2021_opt_04_utf8_popt1_popt2.csv
2021_opt_05_utf8_popt1_popt2.csv
2021_opt_06_utf8_popt1_popt2.csv
2021_opt_07_utf8_popt1_popt2.csv
2021_opt_08_utf8_popt1_popt2.csv
2021_opt_09_utf8_popt1_popt2.csv
2021_opt_10_utf8_popt1_popt2.csv

```
2021_opt_11_utf8_popt1_popt2.csv
2021_opt_12_utf8_popt1_popt2.csv
2022_opt_01_utf8_popt1_popt2.csv
2022_opt_02_utf8_popt1_popt2.csv
2022_opt_03_utf8_popt1_popt2.csv
2022_opt_04_utf8_popt1_popt2.csv
2022_opt_05_utf8_popt1_popt2.csv
2022_opt_06_utf8_popt1_popt2.csv
2022_opt_07_utf8_popt1_popt2.csv
2022_opt_08_utf8_popt1_popt2.csv
2022_opt_09_utf8_popt1_popt2.csv
2022_opt_10_utf8_popt1_popt2.csv
2022_opt_11_utf8_popt1_popt2.csv
2022_opt_12_utf8_popt1_popt2.csv
```

[16]:
```
futFiles = [item for item in futFiles if "_utf8" not in item]
pfut_dir = "downloads/future/"
newFutFiles = []
temp = []
for idx, file_name in enumerate(futFiles):
    convert(pfut_dir + file_name, pfut_dir + file_name[:-4] + "_utf8" +␣
 ↪file_name[-4:])
    newFutFiles.append(file_name[:-4] + "_utf8_fut1" + file_name[-4:])
    temp.append(file_name[:-4] + "_utf8" + file_name[-4:])
print()
print(temp)
del temp
```

```
ok ok ok ok ok ok ok ok
['2015_fut_utf8.csv', '2016_fut_utf8.csv', '2017_fut_utf8.csv',
'2018_fut_utf8.csv', '2019_fut_utf8.csv', '2020_fut_utf8.csv',
'2021_fut_utf8.csv', '2022_fut_utf8.csv']
```

[17]:
```
%%writefile pfut.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>
#include "date.hpp"

using namespace std;

int pfut1_15t16() {
    string in1 = "201";
    string in2 = "_fut_utf8";
    int y = 5;
```

```
    string sy;
    string line;
    while (true) {
        ostringstream oss;
        sy = to_string(y);
        ifstream inputFile("downloads/future/" + in1 + sy + in2 + ".csv");
        if (!inputFile.is_open()) {
            cerr << "Unable to open file for reading.\n";
            break;
        }

        ofstream outFile("downloads/future/" + in1 + sy + in2 + "_pfut1.csv");
        if (!outFile.is_open()) {
            cerr << "Unable to open file for writing.\n";
            break;
        }

        bool c = true;

        while (getline(inputFile, line)) {
            istringstream iss(line);
            vector<string> sl;
            while (getline(iss, line, ',')) {
                sl.push_back(line);
            }

            if ((sl[1] == "TX") && (sl[2].size() != 13)) {
                sl.erase(sl.begin() + 4, sl.begin() + 6);
                sl.erase(sl.begin() + 5, sl.begin() + 7);
                sl.erase(sl.begin() + 6);
                sl.erase(sl.begin() + 7, sl.begin() + 11);
                sl[2].erase(remove(sl[2].begin(), sl[2].end(), ' '), sl[2].
↪end());
                sl.erase(sl.begin() + 1);

                time_t a = date1(sl[0]);
                sl[0] = to_string(a);
                sl[1] = to_string(expirationDiff(sl[1], a));

                for (size_t k = 0; k < sl.size() - 1; ++k) {
                    oss << sl[k] << ",";
                }
                oss << sl[sl.size() - 1] << "\n";
            } else if (c == true) {
                c = false;

                sl.erase(sl.begin() + 4, sl.begin() + 6);
```

```
                sl.erase(sl.begin() + 5, sl.begin() + 7);
                sl.erase(sl.begin() + 6);
                sl.erase(sl.begin() + 7, sl.begin() + 12);
                sl.erase(sl.begin() + 1);

                for (size_t k = 0; k < sl.size() - 1; ++k) {
                    oss << sl[k] << ",";
                }
                oss << sl[sl.size() - 1] << "\n";
            }
        }
        outFile << oss.str();
        inputFile.close();
        outFile.close();
        cerr << "ok\n";

        if (y == 6) {
            break;
        }
        ++y;
    }
    return 0;
}

int pfut1_17() {
    string line;
    ostringstream oss;
    ifstream inputFile("downloads/future/2017_fut_utf8.csv");
    if (!inputFile.is_open()) {
        cerr << "Unable to open file for reading.\n";
    }

    ofstream outFile("downloads/future/2017_fut_utf8_pfut1.csv");
    if (!outFile.is_open()) {
        cerr << "Unable to open file for writing.\n";
    }

    bool c = true;

    while (getline(inputFile, line)) {
        istringstream iss(line);
        vector<string> sl;
        while (getline(iss, line, ',')) {
            sl.push_back(line);
        }
```

```
        if ((sl[1] == "TX") && (sl[2].size() != 13) && (sl[sl.size() - 1].
    ↪find(" ") != string::npos)) {
            sl.erase(sl.begin() + 4, sl.begin() + 6);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 6);
            sl.erase(sl.begin() + 7, sl.begin() + 13);
            sl[2].erase(remove(sl[2].begin(), sl[2].end(), ' '), sl[2].end());
            sl.erase(sl.begin() + 1);
            time_t a = date1(sl[0]);
            sl[0] = to_string(a);
            sl[1] = to_string(expirationDiff(sl[1], a));

            for (size_t k = 0; k < sl.size() - 1; ++k) {
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
        } else if (c == true) {
            c = false;

            sl.erase(sl.begin() + 4, sl.begin() + 6);
            sl.erase(sl.begin() + 5, sl.begin() + 7);
            sl.erase(sl.begin() + 6);
            sl.erase(sl.begin() + 7, sl.begin() + 12);
            sl.erase(sl.begin() + 8);
            sl.erase(sl.begin() + 1);
            sl.erase(sl.end() - 1);

            for (size_t k = 0; k < sl.size() - 1; ++k) {
                if (sl[k] == "-") { sl[k] = "0"; }
                oss << sl[k] << ",";
            }
            oss << sl[sl.size() - 1] << "\n";
        }
    }
    outFile << oss.str();
    inputFile.close();
    outFile.close();
    cerr << "ok\n";
    return 0;
}

int pfut1_18t22() {
    string in1 = "20";
    string in2 = "_fut_utf8";
    int y = 18;
    string sy;
    string line;
```

```
    while (true) {
        ostringstream oss;
        sy = to_string(y);
        ifstream inputFile("downloads/future/" + in1 + sy + in2 + ".csv");
        if (!inputFile.is_open()) {
            cerr << "Unable to open file for reading.\n";
            break;
        }

        ofstream outFile("downloads/future/" + in1 + sy + in2 + "_pfut1.csv");
        if (!outFile.is_open()) {
            cerr << "Unable to open file for writing.\n";
            break;
        }

        bool c = true;

        while (getline(inputFile, line)) {
            if ((line.find(" ") == string::npos) && not c) continue;

            istringstream iss(line);
            vector<string> sl;
            while (getline(iss, line, ',')) {
                sl.push_back(line);
            }

            if (sl[1] == "TX") {
                sl[2].erase(remove(sl[2].begin(), sl[2].end(), ' '), sl[2].
↪end());
                if (sl[2].size() == 13) continue;
                sl.erase(sl.begin() + 4, sl.begin() + 6);
                sl.erase(sl.begin() + 5, sl.begin() + 7);
                sl.erase(sl.begin() + 6);
                sl.erase(sl.begin() + 7, sl.end());
                sl.erase(sl.begin() + 1);
                time_t a = date1(sl[0]);
                sl[0] = to_string(a);
                sl[1] = to_string(expirationDiff(sl[1], a));

                for (size_t k = 0; k < sl.size() - 1; ++k) {
                    if ((sl[k] == " ") || (sl[k] == "") || (sl[k] == "-")) {
                        sl[k] = "0";
                    }
                    oss << sl[k] << ",";
                }
                oss << sl[sl.size() - 1] << "\n";
            } else if (c == true) {
```

```
                c = false;

                sl.erase(sl.begin() + 4, sl.begin() + 6);
                sl.erase(sl.begin() + 5, sl.begin() + 7);
                sl.erase(sl.begin() + 6);
                sl.erase(sl.begin() + 7, sl.end());
                sl.erase(sl.begin() + 1);

                for (size_t k = 0; k < sl.size() - 1; ++k) {
                    if (sl[k] == "-") { sl[k] = "0"; }
                    oss << sl[k] << ",";
                }
                oss << sl[sl.size() - 1] << "\n";
            }
        }
        outFile << oss.str();
        inputFile.close();
        outFile.close();
        cerr << "ok\n";

        if (y == 22) {
            break;
        }
        ++y;
    }
    return 0;
}

int main() {
    int a = pfut1_15t16();
    cout << a << endl;
    int b = pfut1_17();
    cout << b << endl;
    int c = pfut1_18t22();
    cout << c << endl;
    return 0;
}
```

```
Overwriting pfut.cpp
```

[18]:
```
!g++ pfut.cpp -o pfut.exe
```

[19]:
```
!pfut.exe
```

```
0
0
0

ok
```

```
ok
ok
ok
ok
ok
ok
ok
```

[20]:
```
print(os.listdir(pfut_dir))
```

```
['2015_fut.csv', '2015_fut_utf8.csv', '2015_fut_utf8_pfut1.csv', '2016_fut.csv',
'2016_fut_utf8.csv', '2016_fut_utf8_pfut1.csv', '2017_fut.csv',
'2017_fut_utf8.csv', '2017_fut_utf8_pfut1.csv', '2018_fut.csv',
'2018_fut_utf8.csv', '2018_fut_utf8_pfut1.csv', '2019_fut.csv',
'2019_fut_utf8.csv', '2019_fut_utf8_pfut1.csv', '2020_fut.csv',
'2020_fut_utf8.csv', '2020_fut_utf8_pfut1.csv', '2021_fut.csv',
'2021_fut_utf8.csv', '2021_fut_utf8_pfut1.csv', '2022_fut.csv',
'2022_fut_utf8.csv', '2022_fut_utf8_pfut1.csv']
```

[21]:
```
cpi = cpiDownload()
print(cpi)
```

```
['cpi1.csv', 'cpi10.csv', 'cpi10_c.csv', 'cpi11.csv', 'cpi11_c.csv',
'cpi1_c.csv', 'cpi2.csv', 'cpi2_c.csv', 'cpi3.csv', 'cpi3_c.csv', 'cpi4.csv',
'cpi4_c.csv', 'cpi5.csv', 'cpi5_c.csv', 'cpi6.csv', 'cpi6_c.csv', 'cpi7.csv',
'cpi7_c.csv', 'cpi8.csv', 'cpi8_c.csv', 'cpi9.csv', 'cpi9_c.csv',
'pr0102a1m.xml']
```

[22]:
```
ccpi = ccpiDownload()
print(ccpi)
```

```
['ccpi1.csv', 'ccpi1_c.csv', 'ccpi2.csv', 'ccpi2_c.csv', 'ccpi3.csv',
'ccpi3_c.csv', 'ccpi4.csv', 'ccpi4_c.csv', 'ccpi5.csv', 'ccpi5_c.csv',
'ccpi6.csv', 'ccpi6_c.csv', 'ccpi7.csv', 'ccpi7_c.csv', 'pr0103a1m.xml']
```

[23]:
```
import xml.etree.ElementTree as ET
import csv

def parse_xml_to_csv(input_xml, output_csv, name):
    tree = ET.parse(input_xml)
    root = tree.getroot()

    with open(output_csv, mode='w', newline='', encoding='utf-8') as file:
        writer = csv.writer(file)
        writer.writerow(["TIME_PERIOD", "Item_VALUE"])

        for obs in root.findall('Obs'):
            item = obs.find('Item').text
            time_period = obs.find('TIME_PERIOD').text
```

```
            data_type = obs.find('TYPE').text
            item_value = obs.find('Item_VALUE').text

            year = int(time_period[:4])
            if ((item == name) and
                (2014 < year < 2023) and
                (data_type == "  (%)")):
                writer.writerow([time_period, item_value])
    return output_csv + "," + name + "  (%)\n"

raw = ""

input_xml = 'downloads/cpi/pr0102a1m.xml'
output_csv = 'downloads/cpi/cpi1.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (  )(    110 =100)")
output_csv = 'downloads/cpi/cpi2.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (   )(    110 =100)")
output_csv = 'downloads/cpi/cpi3.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "   (    110 =100)")
output_csv = 'downloads/cpi/cpi4.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "    (    )(    110 =100)")
output_csv = 'downloads/cpi/cpi5.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "     (  )(    110 =100)")
output_csv = 'downloads/cpi/cpi6.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "     (    110 =100)")
output_csv = 'downloads/cpi/cpi7.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "    (    110 =100)")
output_csv = 'downloads/cpi/cpi8.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (    110 =100)")
output_csv = 'downloads/cpi/cpi9.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (    110 =100)")
output_csv = 'downloads/cpi/cpi10.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "     (    110 =100)")
output_csv = 'downloads/cpi/cpi11.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "    (    110 =100)")
print(os.listdir("downloads/cpi"))

input_xml = 'downloads/ccpi/pr0103a1m.xml'
output_csv = 'downloads/ccpi/ccpi1.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (  )(    110 =100)")
output_csv = 'downloads/ccpi/ccpi2.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (    )(    110 =100)")
output_csv = 'downloads/ccpi/ccpi3.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (    )    (    110 =100)")
output_csv = 'downloads/ccpi/ccpi4.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (    )(    110 =100)")
output_csv = 'downloads/ccpi/ccpi5.csv'
```

```
raw += parse_xml_to_csv(input_xml, output_csv, "  (     )(    110 =100)")
output_csv = 'downloads/ccpi/ccpi6.csv'
raw += parse_xml_to_csv(input_xml, output_csv, "  (    )(    110 =100)")
output_csv = 'downloads/ccpi/ccpi7.csv'
raw += parse_xml_to_csv(input_xml, output_csv, " (     )(    110 =100)")
print(os.listdir("downloads/ccpi"))

with open('cpiFiles.csv', 'w', encoding = "utf8") as omf:
    omf.write(raw)
```

```
['cpi1.csv', 'cpi10.csv', 'cpi10_c.csv', 'cpi11.csv', 'cpi11_c.csv',
'cpi1_c.csv', 'cpi2.csv', 'cpi2_c.csv', 'cpi3.csv', 'cpi3_c.csv', 'cpi4.csv',
'cpi4_c.csv', 'cpi5.csv', 'cpi5_c.csv', 'cpi6.csv', 'cpi6_c.csv', 'cpi7.csv',
'cpi7_c.csv', 'cpi8.csv', 'cpi8_c.csv', 'cpi9.csv', 'cpi9_c.csv',
'pr0102a1m.xml']
['ccpi1.csv', 'ccpi1_c.csv', 'ccpi2.csv', 'ccpi2_c.csv', 'ccpi3.csv',
'ccpi3_c.csv', 'ccpi4.csv', 'ccpi4_c.csv', 'ccpi5.csv', 'ccpi5_c.csv',
'ccpi6.csv', 'ccpi6_c.csv', 'ccpi7.csv', 'ccpi7_c.csv', 'pr0103a1m.xml']
```

[24]:
```
%%writefile parseCpiCsv.cpp
#include <iostream>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>
#include <fstream>
#include "date.hpp"

using namespace std;

int parse(const string fn, const string ofn) {
    ifstream cpi(fn);
    if (!cpi.is_open()) {
        cerr << "Error: Could not open the file:" << fn << endl;
        return 1;
    }

    string line, sline;
    vector<string> sv;
    ostringstream oss;
    bool firstLine = true;

    while (getline(cpi, line)) {
        sv.clear();
        stringstream ss(line);

        while (getline(ss, sline, ',')) {
```

```
            sv.push_back(sline);
        }

        if (!sv.empty() && !firstLine) {
            sv[0] = to_string(date3(sv[0]));
        }

        if (firstLine) {
            firstLine = false;
        }

        oss << sv[0];
        for (size_t i = 1; i < sv.size(); ++i) {
            oss << "," << sv[i];
        }
        oss << "\n";
    }

    cpi.close();

    ofstream cpi_c(ofn);
    if (!cpi_c.is_open()) {
        cerr << "Error: Could not open the file:" << ofn << endl;
        return 1;
    }

    cpi_c << oss.str();
    cpi_c.close();

    return 0;
}

int main() {
    ifstream c("cpiFiles.csv");
    ofstream d("1mfiles.csv");
    string line, fn, mod;
    int a;
    while (getline(c, line)) {
        size_t pos = line.find(",");
        fn = line.substr(0, pos);
        mod = fn.substr(0, fn.size() - 4) + "_c" + fn.substr(fn.size() - 4);
        a = parse(fn, mod);
        cout << a << endl;
        d << mod << line.substr(pos) << "\n";
    }
    c.close();
    d.close();
```

```
    return 0;
}
```

```
Overwriting parseCpiCsv.cpp
```

[25]:
```
!g++ parseCpiCsv.cpp -o parseCpiCsv.exe
```

[26]:
```
!parseCpiCsv.exe
```

```
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
```

[27]:
```
twiiFIles = twiiDownload()
```

[28]:
```
ixicFiles = ixicDownload()
```

[29]:
```
%%writefile idx.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>
#include "date.hpp"

using namespace std;

int idx1(string file)
{
    ifstream inputFile(file);
    if (!inputFile.is_open())
    {
```

```
        cerr << "Unable to open file for reading:" << file << endl;
        return 1;
    }

    size_t pos = file.find_last_of('.');
    string newfile = file.substr(0, pos) + "_idx1" + file.substr(pos);

    ofstream outFile(newfile);
    if (!outFile.is_open())
    {
        cerr << "Unable to open file for writing:" << newfile << endl;
        return 1;
    }

        bool c = 1;
    string line;
    while (getline(inputFile, line))
    {
        istringstream iss(line);
        vector<string> sl;
        while (getline(iss, line, ','))
        {
            sl.push_back(line);
        }

        sl.erase(sl.begin() + 2, sl.begin() + 4);
        sl.erase(sl.begin() + 3);
        if (c == 1) {
                c = 0;
        } else {
                sl[0] = to_string(date2(sl[0]));
        }
        ostringstream oss;
        for (size_t k = 0; k < sl.size()-1; ++k)
        {
            oss << sl[k] << ",";
        }
        oss << sl[sl.size()-1] << "\n";
        outFile << oss.str();
    }

    inputFile.close();
    outFile.close();
    cerr << "ok\n";

    return 0;
}
```

```
int main() {
    int c = idx1("downloads/twii/^TWII.csv");
    cout << c << endl;
    int d = idx1("downloads/ixic/^IXIC.csv");
    cout << d << endl;
    return 0;
}
```

```
Overwriting idx.cpp
```

[30]:
```
!g++ idx.cpp -o idx.exe
```

[31]:
```
!idx.exe
```

```
0
0

ok
ok
```

[32]:
```
trsFiles = trsDownload()
print(trsFiles)
```

```
['yield-curve-rates-1990-2023.csv', 'yield-curve-rates-2015-2022.csv']
```

[33]:
```
%%writefile ptrs.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>
#include <ostream>
#include "date.hpp"

using namespace std;

int main()
{
        ifstream inputFile("downloads/treasury/yield-curve-rates-1990-2023.
 ↪csv");
        if (!inputFile.is_open())
        {
                cerr << "Error: Unable to open input file" << endl;
                return 1;
        }

        vector<string> lines;
```

```
        vector<string> sl;
        string line;
        string temp;
        bool c = true;

        while (getline(inputFile, line))
        {
                istringstream iss(line);
                vector<string>().swap(sl);
                ostringstream oss;
                bool d = 1, k = 0;

                if (c == true)
                {
                        c = false;
                        while (getline(iss, line, ','))
                        {
                                sl.push_back(line);
                        }
                        sl.erase(sl.begin() + 2);
                        sl.erase(sl.begin() + 3);
                }
                else
                {
                        while (getline(iss, line, ','))
                        {
                                if (d == true)
                                {
                                        temp = line;
                                        d = false;
                                        if (temp.substr(1, 1) == "/")
                                        {
                                                temp = "0" + temp;
                                        }

                                        if (temp.substr(4, 1) == "/")
                                        {
                                                temp = temp.substr(0, 3) + "0"␣
↪+ temp.substr(3);
                                        }

                                        temp = "20" + temp.substr(6) + "/" +␣
↪temp.substr(0, 5);
                                        if (stoi(temp.substr(0, 4)) > 2014 &&␣
↪stoi(temp.substr(0, 4)) < 2023)
                                        {
                                                time_t ttemp = date1(temp);
```

```
                                        sl.push_back(to_string(ttemp));
                                }
                                else
                                {
                                        k = 1;
                                        break;
                                }
                        }
                        else
                        {
                                sl.push_back(line);
                        }
                }
                if (k == 0)
                {
                        sl.erase(sl.begin() + 2);
                        sl.erase(sl.begin() + 3);
                }
                else
                {
                        continue;
                }
        }

        for (size_t i = 0; i < sl.size(); ++i)
        {
                oss << sl[i];
                if (i < sl.size() - 1)
                {
                        oss << ",";
                }
        }

        lines.push_back(oss.str());
}

inputFile.close();

lines.erase(remove(lines.begin(), lines.end(), ""), lines.end());
reverse(lines.begin() + 1, lines.end());

ostringstream alloss;

for (const string &sortedLine : lines)
{
        alloss << sortedLine << '\n';
}
```

```
        ofstream outputFile("downloads/treasury/yield-curve-rates-2015-2022.
 ↪csv");
        if (!outputFile.is_open())
        {
                cerr << "Error: Unable to open output file\n";
                return 1;
        }

        outputFile << alloss.str();

        outputFile.close();

        return 0;
}
```

```
Overwriting ptrs.cpp
```

[34]:
```
!g++ ptrs.cpp -o ptrs.exe
```

[35]:
```
!ptrs.exe
```

[36]:
```
tuFiles = tuDownload()
print(tuFiles)
for i in tuFiles:
    convertS("downloads/twdusd/" + i, "downloads/twdusd/" + i[0:-4] + "_utf8" +␣
 ↪i[-4:])
```

```
['EG51D01.csv', 'EG51D01_utf8.csv', 'EG51D01_utf8_utf8.csv',
'EG51D01_utf8_utf8_utf8.csv', 'EG51D01_utf8_utf8_utf8_utf8.csv',
'EG51D01_utf8_utf8_utf8_utf8_utf8.csv', 'twdusd_utf8_tu1.csv',
'twdusd_utf8_tu1_utf8.csv', 'twdusd_utf8_tu1_utf8_utf8.csv',
'twdusd_utf8_tu1_utf8_utf8_utf8.csv', 'twdusd_utf8_tu1_utf8_utf8_utf8_utf8.csv']
ok ok ok ok ok ok ok ok ok ok ok
```

[37]:
```
ferFiles = ferDownload()
print(ferFiles)
for i in ferFiles:
    convertS("downloads/forward/" + i, "downloads/forward/" + i[0:-4] + "_utf8"␣
 ↪+ i[-4:])
```

```
['EG55D01.csv', 'EG55D01_utf8.csv', 'EG55D01_utf8_utf8.csv',
'EG55D01_utf8_utf8_utf8.csv', 'EG55D01_utf8_utf8_utf8_utf8.csv',
'EG55D01_utf8_utf8_utf8_utf8_utf8.csv', 'forward_exchange_rate_utf8_fer1.csv',
'forward_exchange_rate_utf8_fer1_utf8.csv',
'forward_exchange_rate_utf8_fer1_utf8_utf8.csv',
'forward_exchange_rate_utf8_fer1_utf8_utf8_utf8.csv',
'forward_exchange_rate_utf8_fer1_utf8_utf8_utf8_utf8.csv']
ok ok ok ok ok ok ok ok ok ok ok
```

[38]:
```
%%writefile tu.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>
#include "date.hpp"
using namespace std;

int main() {
    ifstream inputFile("downloads/twdusd/EG51D01_utf8.csv");
    if (!inputFile.is_open()) {
        cerr << "Error: Unable to open input file.\n";
        return 1;
    }

    ofstream outputFile("downloads/twdusd/twdusd_utf8_tu1.csv");
    if (!outputFile.is_open())
    {
        cerr << "Error: Unable to open output file.\n";
        return 2;
    }

    vector<string> sl;
    string modifiedStr, line, subline;
    ostringstream oss;
    int temp;
    bool c = true;

    while (getline(inputFile, line)) {
        istringstream iss(line);
        while (getline(iss, subline, ',')) {
            sl.push_back(subline);
        }

        if (c == true) {
            c = false;
            for (string& s : sl) {
                    s.erase(remove(s.begin(), s.end(), '\"'), s.end());
            }
            sl.erase(sl.begin() + 4);
        } else {
            temp = stoi(sl[0].substr(1, 4));
            if (temp < 2015 || temp > 2022) {
                sl.clear();
                continue;
```

```
            } else {
                for (string& s : sl) {
                    s.erase(remove(s.begin(), s.end(), '\"'), s.end());
                }
                sl.erase(sl.end() - 2);
                sl[0] = to_string(date5(sl[0]));
            }
        }

        for (size_t i = 0; i < sl.size() - 1; ++i) {
            oss << sl[i];
            oss << ",";
        }
        oss << sl[sl.size() - 1];
        oss << "\n";

        sl.clear();
    }

    inputFile.close();

    outputFile << oss.str();
    outputFile.close();

    return 0;
}
```

```
Overwriting tu.cpp
```

[39]:
```
!g++ tu.cpp -o tu.exe
```

[40]:
```
!tu.exe
```

[41]:
```
%%writefile pfer.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <vector>
#include <algorithm>
#include "date.hpp"
using namespace std;

int main() {
    ifstream inputFile("downloads/forward/EG55D01_utf8.csv");
    if (!inputFile.is_open()) {
        cerr << "Error: Unable to open input file.\n";
        return 1;
```

```
    }

    ofstream outputFile("downloads/forward/forward_exchange_rate_utf8_fer1.
↪csv");
    if (!outputFile.is_open())
    {
        cerr << "Error: Unable to open output file.\n";
        return 2;
    }

    vector<string> sl;
    string modifiedStr, line, subline;
    ostringstream oss;
    int temp;
    bool c = true;

    while (getline(inputFile, line)) {
        istringstream iss(line);
        while (getline(iss, subline, ',')) {
            sl.push_back(subline);
        }

        if (c == true) {
            c = false;
            for (string& s : sl) {
            s.erase(remove(s.begin(), s.end(), '\"'), s.end());
            }
            sl.erase(sl.end() - 4, sl.end() - 2);
        } else {
            temp = stoi(sl[0].substr(1, 4));
            if (temp < 2015 || temp > 2022) {
                sl.clear();
                continue;
            } else {
                for (string& s : sl) {
                    s.erase(remove(s.begin(), s.end(), '\"'), s.end());
                }
                sl[0] = to_string(date5(sl[0]));
                sl.erase(sl.end() - 4, sl.end() - 2);
            }
        }

        for (size_t i = 0; i < sl.size() - 1; ++i) {
            oss << sl[i];
            oss << ",";
        }
        oss << sl[sl.size() - 1];
```

```
        oss << "\n";

        sl.clear();
    }

    inputFile.close();

    outputFile << oss.str();
    outputFile.close();

    return 0;
}
```

```
Overwriting pfer.cpp
```

[42]:
```
!g++ pfer.cpp -o pfer.exe
```

[43]:
```
!pfer.exe
```

[44]:
```
raw = "twii/^TWII_idx1.csv\nixic/^IXIC_idx1.csv\ntreasury/
 ↪yield-curve-rates-2015-2022.csv\ntwdusd/twdusd_utf8_tu1.csv\nforward/
 ↪forward_exchange_rate_utf8_fer1.csv"
with open('1dfiles.csv', 'w') as odf:
    odf.write(raw)
```

```
raw = "2015_1_opt_utf8_popt1_popt2.csv\n2015_2_opt_utf8_popt1_popt2.
 ↪csv\n2015_3_opt_utf8_popt1_popt2.csv\n2015_4_opt_utf8_popt1_popt2.
 ↪csv\n2016_opt_1_utf8_popt1_popt2.csv\n2016_opt_10_utf8_popt1_popt2.
 ↪csv\n2016_opt_11_utf8_popt1_popt2.csv\n2016_opt_12_utf8_popt1_popt2.
 ↪csv\n2016_opt_2_utf8_popt1_popt2.csv\n2016_opt_3_utf8_popt1_popt2.
 ↪csv\n2016_opt_4_utf8_popt1_popt2.csv\n2016_opt_5_utf8_popt1_popt2.
 ↪csv\n2016_opt_6_utf8_popt1_popt2.csv\n2016_opt_7_utf8_popt1_popt2.
 ↪csv\n2016_opt_8_utf8_popt1_popt2.csv\n2016_opt_9_utf8_popt1_popt2.
 ↪csv\n2017_opt_1_utf8_popt1_popt2.csv\n2017_opt_10_1_utf8_popt1_popt2.
 ↪csv\n2017_opt_10_2_utf8_popt1_popt2.csv\n2017_opt_11_1_utf8_popt1_popt2.
 ↪csv\n2017_opt_11_2_utf8_popt1_popt2.csv\n2017_opt_12_1_utf8_popt1_popt2.
 ↪csv\n2017_opt_12_2_utf8_popt1_popt2.csv\n2017_opt_2_utf8_popt1_popt2.
 ↪csv\n2017_opt_3_utf8_popt1_popt2.csv\n2017_opt_4_utf8_popt1_popt2.
 ↪csv\n2017_opt_5_utf8_popt1_popt2.csv\n2017_opt_6_1_utf8_popt1_popt2.
 ↪csv\n2017_opt_6_2_utf8_popt1_popt2.csv\n2017_opt_7_1_utf8_popt1_popt2.
 ↪csv\n2017_opt_7_2_utf8_popt1_popt2.csv\n2017_opt_8_1_utf8_popt1_popt2.
 ↪csv\n2017_opt_8_2_utf8_popt1_popt2.csv\n2017_opt_9_1_utf8_popt1_popt2.
 ↪csv\n2017_opt_9_2_utf8_popt1_popt2.csv\n2018_opt_01_utf8_popt1_popt2.
 ↪csv\n2018_opt_02_utf8_popt1_popt2.csv\n2018_opt_03_utf8_popt1_popt2.
 ↪csv\n2018_opt_04_utf8_popt1_popt2.csv\n2018_opt_05_utf8_popt1_popt2.
 ↪csv\n2018_opt_06_utf8_popt1_popt2.csv\n2018_opt_07_utf8_popt1_popt2.
 ↪csv\n2018_opt_08_utf8_popt1_popt2.csv\n2018_opt_09_utf8_popt1_popt2.
 ↪csv\n2018_opt_10_utf8_popt1_popt2.csv\n2018_opt_11_utf8_popt1_popt2.
 ↪csv\n2018_opt_12_utf8_popt1_popt2.csv\n2019_opt_01_utf8_popt1_popt2.
 ↪csv\n2019_opt_02_utf8_popt1_popt2.csv\n2019_opt_03_utf8_popt1_popt2.
 ↪csv\n2019_opt_04_utf8_popt1_popt2.csv\n2019_opt_05_utf8_popt1_popt2.
 ↪csv\n2019_opt_06_utf8_popt1_popt2.csv\n2019_opt_07_utf8_popt1_popt2.
 ↪csv\n2019_opt_08_utf8_popt1_popt2.csv\n2019_opt_09_utf8_popt1_popt2.
 ↪csv\n2019_opt_10_utf8_popt1_popt2.csv\n2019_opt_11_utf8_popt1_popt2.
 ↪csv\n2019_opt_12_utf8_popt1_popt2.csv\n2020_opt_01_utf8_popt1_popt2.
 ↪csv\n2020_opt_02_utf8_popt1_popt2.csv\n2020_opt_03_utf8_popt1_popt2.
 ↪csv\n2020_opt_04_utf8_popt1_popt2.csv\n2020_opt_05_utf8_popt1_popt2.
 ↪csv\n2020_opt_06_utf8_popt1_popt2.csv\n2020_opt_07_utf8_popt1_popt2.
 ↪csv\n2020_opt_08_utf8_popt1_popt2.csv\n2020_opt_09_utf8_popt1_popt2.
 ↪csv\n2020_opt_10_utf8_popt1_popt2.csv\n2020_opt_11_utf8_popt1_popt2.
 ↪csv\n2020_opt_12_utf8_popt1_popt2.csv\n2021_opt_01_utf8_popt1_popt2.
 ↪csv\n2021_opt_02_utf8_popt1_popt2.csv\n2021_opt_03_utf8_popt1_popt2.
 ↪csv\n2021_opt_04_utf8_popt1_popt2.csv\n2021_opt_05_utf8_popt1_popt2.
 ↪csv\n2021_opt_06_utf8_popt1_popt2.csv\n2021_opt_07_utf8_popt1_popt2.
 ↪csv\n2021_opt_08_utf8_popt1_popt2.csv\n2021_opt_09_utf8_popt1_popt2.
 ↪csv\n2021_opt_10_utf8_popt1_popt2.csv\n2021_opt_11_utf8_popt1_popt2.
 ↪csv\n2021_opt_12_utf8_popt1_popt2.csv\n2022_opt_01_utf8_popt1_popt2.
 ↪csv\n2022_opt_02_utf8_popt1_popt2.csv\n2022_opt_03_utf8_popt1_popt2.
 ↪csv\n2022_opt_04_utf8_popt1_popt2.csv\n2022_opt_05_utf8_popt1_popt2.
 ↪csv\n2022_opt_06_utf8_popt1_popt2.csv\n2022_opt_07_utf8_popt1_popt2.
 ↪csv\n2022_opt_08_utf8_popt1_popt2.csv\n2022_opt_09_utf8_popt1_popt2.
 ↪csv\n2022_opt_10_utf8_popt1_popt2.csv\n2022_opt_11_utf8_popt1_popt2.
 ↪csv\n2022_opt_12_utf8_popt1_popt2.csv"
```

```
with open('optFiles.csv', 'w') as opt:
    opt.write(raw)

raw = "2015_fut_utf8_pfut1.csv\n2016_fut_utf8_pfut1.csv\n2017_fut_utf8_pfut1.
 ↪csv\n2018_fut_utf8_pfut1.csv\n2019_fut_utf8_pfut1.csv\n2020_fut_utf8_pfut1.
 ↪csv\n2021_fut_utf8_pfut1.csv\n2022_fut_utf8_pfut1.csv"
with open('futFiles.csv', 'w') as fut:
    fut.write(raw)
```

[45]:
```
%%writefile merge_csv.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <map>
#include <string>
#include <set>
using namespace std;

string replaceAll(const string& str, const string& from, const string& to) {
    string result = str;
    size_t start_pos = 0;
    while((start_pos = result.find(from, start_pos)) != string::npos) {
        result.replace(start_pos, from.length(), to);
        start_pos += to.length();
    }
    return result;
}

map<string, vector<string>> read_csv(const string& filename) {
    ifstream file(filename);
    ostringstream oss;
    map<string, vector<string>> data;
    string line, cell;
    bool i = 0;

    getline(file, line);

    while (getline(file, line)) {
        if (line.empty()) continue;
        stringstream lineStream(line);
        vector<string> row;
        string key;
        i = 0;

        while (getline(lineStream, cell, ',')) {
            if (!i) {
```

```
                key = cell;
                i = 1;
            } else {
                try{
                    double db = stod(cell);
                } catch (...) {
                    cell = "0";
                }
                row.push_back(cell);
            }
        }

        data[key] = row;
    }

    file.close();

    return data;
}

string merge_csv(const vector<map<string, vector<string>>>& data_maps) {
    ostringstream file;
    set<string> all_keys;

    for (const auto& data_map : data_maps) {
        for (const auto& entry : data_map) {
            all_keys.insert(entry.first);
        }
    }
    for (const string& key : all_keys) {
        file << key;
        for (const auto& data_map : data_maps) {
            if (data_map.find(key) != data_map.end()) {
                for (const auto& value : data_map.at(key)) {
                    file << "," << value;
                }
            } else {
                for (size_t x = 0; x < (data_map.begin()->second).size(); x++) {
                    file << ",0";
                }
            }
        }
        file << "\n";
    }

    string result = file.str();
    if (!result.empty()) result.pop_back();
```

```
    return result;
}

int main() {
    vector<map<string, vector<string>>> data_maps;
    ostringstream h;
    h <<␣
 ↪" ,    ,    ,    ,    ,    ,    ,   1 ,   3 ,   6 ,   1 ,   2 ,   3 ,   5 ,   7
 ↪

    vector<string> files;
    string line;
    ifstream odf("1dfiles.csv");
    while (getline(odf, line)) {
            files.push_back(line);
    }

    for (size_t i = 0; i < files.size(); ++i) {
        map<string, vector<string>> r = read_csv("downloads/" + files[i]);
        data_maps.push_back(r);
    }

    string merged = merge_csv(data_maps);
    h << merged;

    ofstream op("1dData.csv");
    op << h.str();
    op.close();

    cout << "CSV files merged successfully." << endl;
    return 0;
}
```

```
Overwriting merge_csv.cpp
```

[46]:
```
!g++ merge_csv.cpp -o merge_csv.exe
```

[47]:
```
!merge_csv.exe
```

```
CSV files merged successfully.
```

[48]:
```
%%writefile cpiToDaily.cpp
#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <vector>
```

```cpp
#include <memory>
using namespace std;

int main()
{
        ifstream mf("1mfiles.csv");
        string line, fn, key;
    vector<string> dk, mk1, dk1;
        size_t pos;
    vector<vector<string>> cpis;

        ostringstream oss;
        ifstream od("1dData.csv");
        getline(od, line);
        oss << line;

    while (getline(mf, line)) {
        size_t p = line.find(",");
        fn = line.substr(0, p);
        ifstream cpi(fn);
        oss << line.substr(p);
            getline(cpi, line);
        vector<string> vec;
        getline(cpi, line);
            while (getline(cpi, line))
            {
                    vec.push_back(line);
            }
        cpi.close();
        cpis.push_back(vec);
    }
    oss << "\n";

    for (auto vec : cpis) {
            pos = vec[0].find(",");
            if (pos != string::npos)
            {
                    dk.push_back(vec[0].substr(pos + 1));
            }

            pos = vec[1].find(",");
            if (pos != string::npos)
            {
                    mk1.push_back(vec[1].substr(0, pos));
                    dk1.push_back(vec[1].substr(pos + 1));
            }
    }
```

```
    int i = 2;
        while (getline(od, line))
        {
                pos = line.find(',');
                if (pos != string::npos)
                {
                        key = line.substr(0, pos);
                } else {
            cout << "e";
            return 1;
        }
        oss << line;

        for (size_t j = 0; j < cpis.size(); j++) {
                    if (stoi(key) >= stoi(mk1[j]))
                    {
                            dk[j] = dk1[j];
                            if (i < cpis[j].size())
                            {
                                    pos = cpis[j][i].find(',');
                                    mk1[j] = cpis[j][i].substr(0, pos);
                    dk1[j] = cpis[j][i].substr(pos + 1);
                                    i++;
                            }
                    }
                    oss << "," << dk[j];
        }
        oss << "\n";
        }

        ofstream output("Daily.csv");
        string k = oss.str();
        if (k.empty())
                return 1;
        k.pop_back();
        output << k;
        output.close();

        return 0;
}
```

```
Overwriting cpiToDaily.cpp
```

[49]:
```
!g++ cpiToDaily.cpp -o  cpiToDaily.exe
```

[50]:
```
!cpiToDaily.exe
```

[51]:
```
%%writefile optCount.cpp
#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <vector>
#include <algorithm>

using namespace std;

vector<int> count(const string& fn) {
        ifstream opt(fn);
        int count = 1, mc = 0, count2 = 1, mc2 = 0, count3 = 1, mc3 = 0;
        string d = "", e = "", f = "";
        string line;
        while (getline(opt, line)) {
                size_t pos = line.find(",");
                string date = line.substr(0, pos);
                string left = line.substr(pos + 1);
                size_t pos2 = left.find(",");
                string expr = left.substr(0, pos2);
                left = left.substr(pos2 + 1);
                size_t pos3 = left.find(",");
                left = left.substr(0, pos3);
                if (d == date) {
                        if (e == expr) {
                                if (f == left) {
                                        count3 ++;
                                } else {
                                        f = left;
                                        mc3 = max(mc3, count3);
                                        count3 = 1;
                                        count2 ++;
                                }
                        } else {
                                f = "";
                                mc3 = max(mc3, count3);
                                count3 = 1;
                                e = expr;
                                mc2 = max(mc2, count2);
                                count2 = 1;
                            count ++;
                        }
                } else {
                        f = "";
                        mc3 = max(mc3, count3);
                        count3 = 1;
```

```
                        e = "";
                        mc2 = max(mc2, count2);
                        count2 = 1;
                        d = date;
                        mc = max(mc, count);
                        count = 1;
                }
        }

        opt.close();

        vector<int> vec = {mc, mc2, mc3};

        return vec;
}

int main()
{
        ifstream ff("optFiles.csv");

        string line;
        vector<int> c = {0, 0, 0};
        while (getline(ff, line)) {
                vector<int> d = count("downloads/option/" + line);

                for (size_t i = 0; i < c.size(); i++) {
                        c[i] = max(c[i], d[i]);
                }
        }

        ff.close();

        ofstream op("opt_count.csv");
        for (auto i : c) {
        cout << i << endl;
                op << to_string(i) << "\n";
        }
        op.close();

        return 0;
}
```

```
Overwriting optCount.cpp
```

[52]:
```
!g++ optCount.cpp -o optCount.exe
```

[53]:
```
!optCount.exe
```

```
9
105
2
```

[54]:
```
with open("opt_count.csv", "r", encoding="utf-8") as count:
    fst = int(count.readline())
    snd = int(count.readline())
with open("optHeader.csv", "w", encoding="utf-8") as file:
    file.write("  Unix time ")
    for i in range(1, fst + 1):
        file.write(", " + str(i) + "       ")
        for j in range(1, snd + 1):
            file.write(", " + str(i) + "    " + str(j) + "   , " + str(i) +␣
 ↪"    " + str(j) + "     , " + str(i) + "    " + str(j) + "     , " + str(i) +␣
 ↪"    " + str(j) + "     , " + str(i) + "    " + str(j) + "     , " + str(i) +␣
 ↪"    " + str(j) + "     , " + str(i) + "    " + str(j) + "     ")
```

[55]:
```
%%writefile optUlt.cpp
#include <iostream>
#include <string>
#include <vector>
#include <sstream>
#include <fstream>
#include <iomanip>
#include <memory>
#include "date.hpp"
using namespace std;

int main() {
    ifstream co("opt_count.csv");
    if (!co.is_open()) {
        cerr << "Can't open opt_count.csv" << endl;
        return 1;
    }
    string line;
    getline(co, line);
    int dateMax = stoi(line);
    cout << "Date Max: " << dateMax << endl;
    getline(co, line);
    int exprMax = stoi(line);
    cout << "Expiration Max: " << exprMax << endl;
    co.close();

    ifstream head("optHeader.csv");
    if (!head.is_open()) {
        cerr << "Can't open optHeader.csv";
        return 1;
```

```
    }
    getline(head, line);
    head.close();
    ostringstream hoss;
    hoss << line;
    string hd = hoss.str();

    ifstream optfs("optFiles.csv");
    if (!optfs.is_open()) {
        cerr << "Can't open optFiles.csv";
        return 1;
    }
    vector<vector<double>> allvec;
    while (getline(optfs, line)) {
        ifstream opti("downloads/option/" + line);
        if (!opti.is_open()) {
            cerr << "Can't open file: downloads/option/" << line << endl;
            return 1;
        }
        cout << "downloads/option/" + line << " opened" << endl;
        getline(opti, line);
        while (getline(opti, line)) {
            if (line.empty()) continue;
            istringstream iss(line);
            vector<double> svec;
            string cell;
            while (getline(iss, cell, ',')) {
                svec.push_back(stod(cell));
            }
            allvec.push_back(svec);
        }
            opti.close();
    }
    optfs.close();

    double date = allvec[0][0], expr = allvec[0][1], price = allvec[0][2];
    vector<double> osvec, call, put;
    bool c, p;
    int exprCount = 0, dateCount = 0;
    vector<vector<double>> oallvec;
    osvec.push_back(date);
    osvec.push_back(expr);
    for (auto svec : allvec) {
        if (date == svec[0]) {
            if (expr == svec[1]) {
                if (price == svec[2]) {
                    if (svec[3] == 0) {
```

```
                    put.push_back(svec[4]);
                    put.push_back(svec[5]);
                    put.push_back(svec[6]);
                    p = 1;
                } else {
                    call.push_back(svec[4]);
                    call.push_back(svec[5]);
                    call.push_back(svec[6]);
                    c = 1;
                }
            } else {
                osvec.push_back(price);
                price = svec[2];
                exprCount ++;
                if (c) {
                    osvec.push_back(call[0]);
                    osvec.push_back(call[1]);
                    osvec.push_back(call[2]);
                } else {
                    osvec.push_back(0);
                    osvec.push_back(0);
                    osvec.push_back(0);
                }
                if (p) {
                    osvec.push_back(put[0]);
                    osvec.push_back(put[1]);
                    osvec.push_back(put[2]);
                } else {
                    osvec.push_back(0);
                    osvec.push_back(0);
                    osvec.push_back(0);
                }
                put = {};
                call = {};
                if (svec[3] == 0) {
                    put.push_back(svec[4]);
                    put.push_back(svec[5]);
                    put.push_back(svec[6]);
                    p = 1;
                    c = 0;
                } else {
                    call.push_back(svec[4]);
                    call.push_back(svec[5]);
                    call.push_back(svec[6]);
                    c = 1;
                    p = 0;
                }
```

```
            }
        } else {
            osvec.push_back(price);
            price = svec[2];
            expr = svec[1];
            exprCount ++;
            dateCount ++;
            if (c) {
                osvec.push_back(call[0]);
                osvec.push_back(call[1]);
                osvec.push_back(call[2]);
            } else {
                osvec.push_back(0);
                osvec.push_back(0);
                osvec.push_back(0);
            }
            if (p) {
                osvec.push_back(put[0]);
                osvec.push_back(put[1]);
                osvec.push_back(put[2]);
            } else {
                osvec.push_back(0);
                osvec.push_back(0);
                osvec.push_back(0);
            }
            put = {};
            call = {};
            if (svec[3] == 0) {
                put.push_back(svec[4]);
                put.push_back(svec[5]);
                put.push_back(svec[6]);
                p = 1;
                c = 0;
            } else {
                call.push_back(svec[4]);
                call.push_back(svec[5]);
                call.push_back(svec[6]);
                c = 1;
                p = 0;
            }
            for (int i = 0; i < (exprMax - exprCount); i ++) {
                for (int j = 0; j < 7; j ++) {
                    osvec.push_back(0);
                }
            }
            exprCount = 0;
            osvec.push_back(expr);
```

```
            }
        } else {
            osvec.push_back(price);
            price = svec[2];
            expr = svec[1];
            date = svec[0];
            exprCount ++;
            if (c) {
                osvec.push_back(call[0]);
                osvec.push_back(call[1]);
                osvec.push_back(call[2]);
            } else {
                osvec.push_back(0);
                osvec.push_back(0);
                osvec.push_back(0);
            }
            if (p) {
                osvec.push_back(put[0]);
                osvec.push_back(put[1]);
                osvec.push_back(put[2]);
            } else {
                osvec.push_back(0);
                osvec.push_back(0);
                osvec.push_back(0);
            }
            put = {};
            call = {};
            if (svec[3] == 0) {
                put.push_back(svec[4]);
                put.push_back(svec[5]);
                put.push_back(svec[6]);
                p = 1;
                c = 0;
            } else {
                call.push_back(svec[4]);
                call.push_back(svec[5]);
                call.push_back(svec[6]);
                c = 1;
                p = 0;
            }
            for (int i = 0; i < (exprMax - exprCount); i ++) {
                for (int j = 0; j < 7; j ++) {
                    osvec.push_back(0);
                }
            }
            exprCount = 0;
            for (int h = 0; h < (dateMax - dateCount - 1); h ++) {
```

```
                osvec.push_back(0);
                for (int i = 0; i < exprMax; i ++){
                    for (int j = 0; j < 7; j ++) {
                        osvec.push_back(0);
                    }
                }
            }
            dateCount = 0;
            oallvec.push_back(osvec);
            osvec = {};
            osvec.clear();
            osvec.push_back(date);
            osvec.push_back(expr);
        }
    }
    osvec.push_back(price);
    exprCount ++;
    if (c) {
        osvec.push_back(call[0]);
        osvec.push_back(call[1]);
        osvec.push_back(call[2]);
    } else {
        osvec.push_back(0);
        osvec.push_back(0);
        osvec.push_back(0);
    }
    if (p) {
        osvec.push_back(put[0]);
        osvec.push_back(put[1]);
        osvec.push_back(put[2]);
    } else {
        osvec.push_back(0);
        osvec.push_back(0);
        osvec.push_back(0);
    }
    put = {};
    call = {};
    for (int i = 0; i < (exprMax - exprCount); i ++) {
        for (int j = 0; j < 7; j ++) {
            osvec.push_back(0);
        }
    }
    for (int h = 0; h < (dateMax - dateCount - 1); h ++) {
        osvec.push_back(0);
        for (int i = 0; i < exprMax; i ++){
            for (int j = 0; j < 7; j ++) {
                osvec.push_back(0);
```

```
            }
        }
    }
    oallvec.push_back(osvec);
    cout << "Processed" << endl;

    ostringstream oss;
    int year = 16;
    for (auto i : oallvec) {
        if (i[0] >= date2("20" + to_string(year) + "-01-01")) {
            ofstream output("optData/20" + to_string(year - 1) + "optData.csv");
            if (!output.is_open()) {
                cerr << "Can't open optData/20" + to_string(year - 1) +␣
↪"optData.csv";
                return 1;
            }
            output << hd << oss.str();
            output.close();
            cout << "Outputted to optData/20" + to_string(year - 1) + "optData.
↪csv" << endl;
            oss.str("");
            oss.clear();
            year ++;
        }
        oss << "\n";
        for (auto j : i) {
            oss << fixed << setprecision(6) << j << ",";
        }
        string temp = oss.str();
        oss.str("");
        oss.clear();
        if (!temp.empty()) temp.pop_back();
        oss << temp;
    }
    ofstream output("optData/20" + to_string(year - 1) + "optData.csv");
    if (!output.is_open()) {
        cerr << "Can't open optData/20" + to_string(year - 1) + "optData.csv";
        return 1;
    }
    output << hd << oss.str();
    output.close();
    cout << "Outputted to optData/20" + to_string(year - 1) + "optData.csv" <<␣
↪endl;

    return 0;
}
```

```
Overwriting optUlt.cpp
```

[56]:
```
!g++ optUlt.cpp -o optUlt.exe
```

[57]:
```
!optUlt.exe
```

```
Date Max: 9
Expiration Max: 105
downloads/option/2015_1_opt_utf8_popt1_popt2.csv opened
downloads/option/2015_2_opt_utf8_popt1_popt2.csv opened
downloads/option/2015_3_opt_utf8_popt1_popt2.csv opened
downloads/option/2015_4_opt_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_1_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_10_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_11_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_12_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_2_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_3_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_4_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_5_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_6_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_7_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_8_utf8_popt1_popt2.csv opened
downloads/option/2016_opt_9_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_10_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_10_2_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_11_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_11_2_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_12_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_12_2_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_2_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_3_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_4_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_5_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_6_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_6_2_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_7_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_7_2_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_8_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_8_2_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_9_1_utf8_popt1_popt2.csv opened
downloads/option/2017_opt_9_2_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_01_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_02_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_03_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_04_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_05_utf8_popt1_popt2.csv opened
```

```
downloads/option/2018_opt_06_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_07_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_08_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_09_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_10_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_11_utf8_popt1_popt2.csv opened
downloads/option/2018_opt_12_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_01_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_02_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_03_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_04_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_05_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_06_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_07_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_08_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_09_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_10_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_11_utf8_popt1_popt2.csv opened
downloads/option/2019_opt_12_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_01_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_02_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_03_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_04_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_05_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_06_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_07_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_08_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_09_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_10_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_11_utf8_popt1_popt2.csv opened
downloads/option/2020_opt_12_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_01_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_02_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_03_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_04_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_05_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_06_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_07_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_08_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_09_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_10_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_11_utf8_popt1_popt2.csv opened
downloads/option/2021_opt_12_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_01_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_02_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_03_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_04_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_05_utf8_popt1_popt2.csv opened
```

```
downloads/option/2022_opt_06_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_07_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_08_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_09_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_10_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_11_utf8_popt1_popt2.csv opened
downloads/option/2022_opt_12_utf8_popt1_popt2.csv opened
Processed
Outputted to optData/2015optData.csv
Outputted to optData/2016optData.csv
Outputted to optData/2017optData.csv
Outputted to optData/2018optData.csv
Outputted to optData/2019optData.csv
Outputted to optData/2020optData.csv
Outputted to optData/2021optData.csv
Outputted to optData/2022optData.csv
```

[58]:
```
%%writefile futCount.cpp
#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <vector>
#include <algorithm>

using namespace std;

int count(const string& fn) {
        ifstream fut(fn);
        int count = 1, mc = 0;
        string d = "";
        string line;
        while (getline(fut, line)) {
                size_t pos = line.find(",");
                string date = line.substr(0, pos);
                if (d == date) {
            count ++;
                } else {
                        d = date;
                        mc = max(mc, count);
                        count = 1;
                }
        }

        fut.close();

        return mc;
```

```
}

int main()
{
        ifstream ff("futFiles.csv");

        string line;
        int c = 0;
        while (getline(ff, line)) {
                int d = count("downloads/future/" + line);
        c = max(c, d);
        }

        ff.close();

        ofstream fu("fut_count.csv");

    cout << c << endl;
    fu << to_string(c) << "\n";

    fu.close();

        return 0;
}
```

```
Overwriting futCount.cpp
```

[59]:
```
!g++ futCount.cpp -o futCount.exe
```

[60]:
```
!futCount.exe
```

```
6
```

[61]:
```
with open("fut_count.csv", "r", encoding="utf-8") as count:
    fst = int(count.readline())
with open("futHeader.csv", "w", encoding="utf-8") as file:
    file.write("  Unix time ")
    for i in range(1, fst + 1):
        file.write(", " + str(i) + "      , " + str(i) + "     , " + str(i) + ␣
 ↪"     , " + str(i) + "     , " + str(i) + "      ")
```

[62]:
```
%%writefile futUlt.cpp
#include <iostream>
#include <string>
#include <vector>
#include <sstream>
#include <fstream>
#include <iomanip>
```

```cpp
#include <memory>
#include "date.hpp"
using namespace std;

int main() {
    ifstream co("fut_count.csv");
    if (!co.is_open()) {
        cerr << "Can't open fut_count.csv" << endl;
        return 1;
    }
    string line;
    getline(co, line);
    int dateMax = stoi(line);
    cout << "Date Max: " << dateMax << endl;
    co.close();

    ifstream head("futHeader.csv");
    if (!head.is_open()) {
        cerr << "Can't open futHeader.csv";
        return 1;
    }
    getline(head, line);
    head.close();
    ostringstream hoss;
    hoss << line;
    string hd = hoss.str();

    ifstream futfs("futFiles.csv");
    if (!futfs.is_open()) {
        cerr << "Can't open futFiles.csv";
        return 1;
    }
    vector<vector<double>> allvec;
    while (getline(futfs, line)) {
        ifstream futi("downloads/future/" + line);
        if (!futi.is_open()) {
            cerr << "Can't open file: downloads/future/" << line << endl;
            return 1;
        }
        cout << "downloads/future/" + line << " opened" << endl;
        getline(futi, line);
        while (getline(futi, line)) {
            if (line.empty()) continue;
            istringstream iss(line);
            vector<double> svec;
            string cell;
            while (getline(iss, cell, ',')) {
```

```
            svec.push_back(stod(cell));
        }
        allvec.push_back(svec);
    }
    futi.close();
}
futfs.close();

double date = allvec[0][0];
vector<double> osvec;
int dateCount = 0;
vector<vector<double>> oallvec;
osvec.push_back(date);
for (auto svec : allvec) {
    if (date == svec[0]) {
        dateCount ++;
        for (size_t k = 1; k < svec.size(); k++) {
            osvec.push_back(svec[k]);
        }
    } else {
        date = svec[0];
        for (int i = 0; i < (dateMax - dateCount); i++) {
            for (int j = 0; j < 5; j++) {
                osvec.push_back(0);
            }
        }
        dateCount = 1;
        oallvec.push_back(osvec);
        osvec = {};
        osvec.clear();
        osvec.push_back(date);
        for (size_t k = 1; k < svec.size(); k++) {
            osvec.push_back(svec[k]);
        }
    }
}
for (int i = 0; i < (dateMax - dateCount); i++) {
    for (int j = 0; j < 5; j++) {
        osvec.push_back(0);
    }
}
oallvec.push_back(osvec);
cout << "Processed" << endl;

ostringstream oss;
int year = 16;
for (auto i : oallvec) {
```

```
        if (i[0] >= date2("20" + to_string(year) + "-01-01")) {
            ofstream output("futData/20" + to_string(year - 1) + "futData.csv");
            if (!output.is_open()) {
                cerr << "Can't open futData/20" + to_string(year - 1) +␣
↪"futData.csv";
                return 1;
            }
            output << hd << oss.str();
            output.close();
            cout << "Outputted to futData/20" + to_string(year - 1) + "futData.
↪csv" << endl;
            oss.str("");
            oss.clear();
            year ++;
        }
        oss << "\n";
        for (auto j : i) {
            oss << fixed << setprecision(6) << j << ",";
        }
        string temp = oss.str();
        oss.str("");
        oss.clear();
        if (!temp.empty()) temp.pop_back();
        oss << temp;
    }
    ofstream output("futData/20" + to_string(year - 1) + "futData.csv");
    if (!output.is_open()) {
        cerr << "Can't open futData/20" + to_string(year - 1) + "futData.csv";
        return 1;
    }
    output << hd << oss.str();
    output.close();
    cout << "Outputted to futData/20" + to_string(year - 1) + "futData.csv" <<␣
↪endl;

    return 0;
}
```

```
Overwriting futUlt.cpp
```

[63]:
```
!g++ futUlt.cpp -o futUlt.exe
```

[64]:
```
!futUlt.exe
```

```
Date Max: 6
downloads/future/2015_fut_utf8_pfut1.csv opened
downloads/future/2016_fut_utf8_pfut1.csv opened
downloads/future/2017_fut_utf8_pfut1.csv opened
```

```
downloads/future/2018_fut_utf8_pfut1.csv opened
downloads/future/2019_fut_utf8_pfut1.csv opened
downloads/future/2020_fut_utf8_pfut1.csv opened
downloads/future/2021_fut_utf8_pfut1.csv opened
downloads/future/2022_fut_utf8_pfut1.csv opened
Processed
Outputted to futData/2015futData.csv
Outputted to futData/2016futData.csv
Outputted to futData/2017futData.csv
Outputted to futData/2018futData.csv
Outputted to futData/2019futData.csv
Outputted to futData/2020futData.csv
Outputted to futData/2021futData.csv
Outputted to futData/2022futData.csv
```

[65]:
```python
dataFs = os.listdir(os.getcwd() + "/optdata")
with open("optDataFiles.csv", "w", encoding="utf-8") as opt:
    for i in range(len(dataFs) - 1):
        opt.write(dataFs[i])
        opt.write("\n")
    opt.write(dataFs[-1])
dataFs = os.listdir(os.getcwd() + "/futdata")
with open("futDataFiles.csv", "w", encoding="utf-8") as fut:
    for i in range(len(dataFs) - 1):
        fut.write(dataFs[i])
        fut.write("\n")
    fut.write(dataFs[-1])
```

[66]:
```python
%%writefile completeData.cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <map>
#include <string>
#include <set>
#include <tuple>
#include "date.hpp"
using namespace std;

map<double, vector<double>> mergeMaps(const map<double, vector<double>>& map1,␣
 ↪const map<double, vector<double>>& map2) {
    map<double, vector<double>> result = map1;

    for (auto it = map2.begin(); it != map2.end(); ++it) {
        result[it->first] = it->second;
    }
```

```
    return result;
}

tuple<string, map<double, vector<double>>> read_csv(const string& filename) {
    cout << "Reading " << filename << endl;
    ifstream file(filename);
    if (!file.is_open()) cerr << "Can't open " << filename;
    map<double, vector<double>> data;
    string line, cell;

    getline(file, line);

    size_t cpos = line.find(",");
    string head = line.substr(cpos);

    while (getline(file, line)) {
        if (line.empty()) continue;
        istringstream lineStream(line);
        vector<double> row;
        double key;
        bool i = 0;

        while (getline(lineStream, cell, ',')) {
            if (!i) {
                key = stod(cell);
                i = 1;
            } else {
                try{
                        row.push_back(stod(cell));
                } catch (const exception& e) {
                        cout << cell << endl << e.what() << endl;
                        row.push_back(0);
                }
            }
        }

        data[key] = row;
    }

    file.close();

    auto result = make_tuple(head, data);
    return result;
}
```

```
map<double, string> merge_csv(const vector<map<double, vector<double>>>&␣
 ↪data_maps) {
    ostringstream file;
    set<double> all_keys;
    map<double, string> mm;

    for (const auto& data_map : data_maps) {
        for (const auto& entry : data_map) {
            all_keys.insert(entry.first);
        }
    }
    for (const auto& key : all_keys) {
        for (const auto& data_map : data_maps) {
            if (data_map.find(key) != data_map.end()) {
                for (const auto& value : data_map.at(key)) {
                    file << "," << fixed << setprecision(6) << value;
                }
            } else {
                for (size_t x = 0; x < (data_map.begin()->second).size(); x++) {
                    file << "," << fixed << setprecision(6) << 0;
                }
            }
        }
        mm[key] = file.str();
        file.str("");
        file.clear();
    }

    return mm;
}

int main() {
    vector<map<double, vector<double>>> data_maps(3);
    ostringstream header;
    header << "  Unix time ";
    string temp;

    auto da = read_csv("Daily.csv");
    header << get<0>(da);
    data_maps[0] = get<1>(da);

    vector<string> files;
    string line;
    ifstream fdf("futDataFiles.csv");
    while (getline(fdf, line)) {
            files.push_back(line);
    }
```

```
    fdf.close();
    for (size_t i = 0; i < files.size(); ++i) {
        tuple<string, map<double, vector<double>>> r = read_csv("futData/" +␣
↪files[i]);
        data_maps[1] = mergeMaps(data_maps[1], get<1>(r));
        if (i == 0) temp = get<0>(r);
    }
    header << temp;

    files = {};
    files.clear();
    ifstream odf("optDataFiles.csv");
    while (getline(odf, line)) {
            files.push_back(line);
    }
    odf.close();
    for (size_t i = 0; i < files.size(); ++i) {
        tuple<string, map<double, vector<double>>> r = read_csv("optData/" +␣
↪files[i]);
        data_maps[2] = mergeMaps(data_maps[2], get<1>(r));
        if (i == 0) temp = get<0>(r);
    }
    header << temp;

    const auto& merged = merge_csv(data_maps);

    ostringstream oss;
    int year = 16;
    for (const auto& pair : merged) {
        if (pair.first >= date2("20" + to_string(year) + "-01-01")) {
            ofstream output("completeData/20" + to_string(year - 1) +␣
↪"completeData.csv");
            output << header.str() << "\n" << oss.str();
            output.close();
            cout << "completeData/20" + to_string(year - 1) + "completeData.
↪csv" << endl;
            oss.str("");
            oss.clear();
            year ++;
        }
        oss << fixed << setprecision(6) << pair.first << pair.second << "\n";
    }
    ofstream output("completeData/20" + to_string(year - 1) + "completeData.
↪csv");
    output << header.str() << "\n" << oss.str();
    output.close();
```

```
    cout << "completeData/20" + to_string(year - 1) + "completeData.csv" <<␣
 ↪endl;

    cout << "CSV files merged successfully." << endl;
    return 0;
}
```

```
Overwriting completeData.cpp
```

[67]:
```
!g++ completeData.cpp -o completeData.exe
```

[68]:
```
!completeData.exe
```

```
Reading Daily.csv
Reading futData/2015futData.csv
Reading futData/2016futData.csv
Reading futData/2017futData.csv
Reading futData/2018futData.csv
Reading futData/2019futData.csv
Reading futData/2020futData.csv
Reading futData/2021futData.csv
Reading futData/2022futData.csv
Reading optData/2015optData.csv
Reading optData/2016optData.csv
Reading optData/2017optData.csv
Reading optData/2018optData.csv
Reading optData/2019optData.csv
Reading optData/2020optData.csv
Reading optData/2021optData.csv
Reading optData/2022optData.csv
completeData/2015completeData.csv
completeData/2016completeData.csv
completeData/2017completeData.csv
completeData/2018completeData.csv
completeData/2019completeData.csv
completeData/2020completeData.csv
completeData/2021completeData.csv
completeData/2022completeData.csv
CSV files merged successfully.
```

[69]:
```
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
import time

def fontDownload(dirc=os.getcwd()):
    if "Iansui-Regular.ttf" not in os.listdir(dirc):
        chrome_options = Options()
        chrome_options.add_experimental_option("prefs", {
```

```
            "download.default_directory": dirc,
            "download.prompt_for_download": False,
            "download.directory_upgrade": True,
            "safebrowsing.enabled": True
        })
        driver = webdriver.Chrome(options=chrome_options)

        url = "https://raw.githubusercontent.com/ButTaiwan/iansui/main/fonts/
↪Iansui-Regular.ttf"
        driver.get(url)

        seconds = 0
        dl_wait = True
        while dl_wait:
            time.sleep(1)
            dl_wait = False
            for fname in os.listdir(dirc):
                if fname.endswith('.crdownload') or fname.endswith('.part'):
                    dl_wait = True
            seconds += 1

        driver.quit()
```

[70]:
```
fontDownload()
```

[2]:
```
import pandas as pd
import matplotlib.pyplot as plt

csv_files = []
for i in range(15, 23):
    csv_files.append(f"completeData/20{i}completeData.csv")

list_of_dfs = []

for file in csv_files:
    df = pd.read_csv(file)
    list_of_dfs.append(df)

df = pd.concat(list_of_dfs, ignore_index=True)

df
```

[2]:

	Unix time				\
0	1.420128e+09	0.000000	0.000000	0.0	4760.240234
1	1.420387e+09	9292.309570	9274.110352	2311000.0	4700.339844
2	1.420474e+09	9209.929688	9048.339844	2725800.0	4666.850098
3	1.420560e+09	9051.940430	9080.089844	2384100.0	4626.839844

```
4      1.420646e+09   9154.030273   9238.030273  2657600.0   4689.540039
…               …             …             …          …             …
2089   1.671984e+09  14271.200195  14285.129883  1574600.0      0.000000
2090   1.672070e+09  14310.190430  14328.429688  1821100.0  10462.190430
2091   1.672157e+09  14249.830078  14173.099609  1958900.0  10339.200195
2092   1.672243e+09  14097.509766  14085.019531  1839200.0  10321.459961
2093   1.672330e+09  14183.519531  14137.690430  1748100.0  10368.370117

                              1       3       6   …  \
0       4726.810059  1.435150e+09        0.02        0.02        0.11  …
1       4652.569824  1.794470e+09        0.02        0.03        0.10  …
2       4592.740234  2.167320e+09        0.02        0.03        0.10  …
3       4650.470215  1.957950e+09        0.02        0.03        0.09  …
4       4736.189941  2.105450e+09        0.01        0.03        0.08  …
…               …             …           …           …           …  …
2089       0.000000  0.000000e+00        0.00        0.00        0.00  …
2090   10353.230469  3.827290e+09        3.87        4.46        4.76  …
2091   10213.290039  3.842970e+09        3.86        4.46        4.75  …
2092   10478.089844  4.154100e+09        4.04        4.45        4.73  …
2093   10466.480469  3.959030e+09        4.12        4.42        4.76  …

       9    104        9    104        9    104       \
0                     0.0                   0.0                   0.0
1                     0.0                   0.0                   0.0
2                     0.0                   0.0                   0.0
3                     0.0                   0.0                   0.0
4                     0.0                   0.0                   0.0
…                     …                     …                     …
2089                  0.0                   0.0                   0.0
2090                  0.0                   0.0                   0.0
2091                  0.0                   0.0                   0.0
2092                  0.0                   0.0                   0.0
2093                  0.0                   0.0                   0.0

       9    105     9    105        9    105       \
0                  0.0                   0.0                   0.0
1                  0.0                   0.0                   0.0
2                  0.0                   0.0                   0.0
3                  0.0                   0.0                   0.0
4                  0.0                   0.0                   0.0
…                  …                     …                     …
2089               0.0                   0.0                   0.0
2090               0.0                   0.0                   0.0
2091               0.0                   0.0                   0.0
2092               0.0                   0.0                   0.0
2093               0.0                   0.0                   0.0
```

```
      9    105        9    105        9    105       \
0                    0.0                  0.0                  0.0
1                    0.0                  0.0                  0.0
2                    0.0                  0.0                  0.0
3                    0.0                  0.0                  0.0
4                    0.0                  0.0                  0.0
…                    …                    …                    …
2089                 0.0                  0.0                  0.0
2090                 0.0                  0.0                  0.0
2091                 0.0                  0.0                  0.0
2092                 0.0                  0.0                  0.0
2093                 0.0                  0.0                  0.0

      9    105
0                    0.0
1                    0.0
2                    0.0
3                    0.0
4                    0.0
…                    …
2089                 0.0
2090                 0.0
2091                 0.0
2092                 0.0
2093                 0.0

[2094 rows x 6706 columns]
```

[4]:
```
from matplotlib.font_manager import FontProperties
custom_font = FontProperties(fname='Iansui-Regular.ttf')
plt.rcParams['font.family'] = custom_font.get_name()
plt.rcParams['axes.unicode_minus'] = False
```

[73]:
```
subdf = df[[' (  )(    110 =100)  (%)', ' (  )(    110 =100)  (%)',␣
 ↪' (    110 =100)  (%)', '  (    )(    110 =100)  (%)',␣
 ↪'    (  )(    110 =100)  (%)', '    (    110 =100)  (%)',␣
 ↪'  (    110 =100)  (%)', ' (    110 =100)  (%)', ' (    110 =100)  (%)',␣
 ↪'    (    110 =100)  (%)', '    (    110 =100)  (%)',␣
 ↪' (  )(    110 =100)  (%)', ' (    )(    110 =100)  (%)',␣
 ↪' (    )    (    110 =100)  (%)', ' (    )(    110 =100)  (%)',␣
 ↪' (     )(    110 =100)  (%)', ' (    )(    110 =100)  (%)',␣
 ↪' (     )(    110 =100)  (%)']].replace(0, pd.NA).fillna(method='pad').
 ↪fillna(method='bfill').set_index(df[' Unix time ']).
 ↪drop_duplicates(subset=[' (  )(    110 =100)  (%)'], keep='last')
fig, ax = plt.subplots(figsize=(10, 6))
for column in subdf.columns:
    ax.plot(subdf.index, subdf[column], label=column)
```

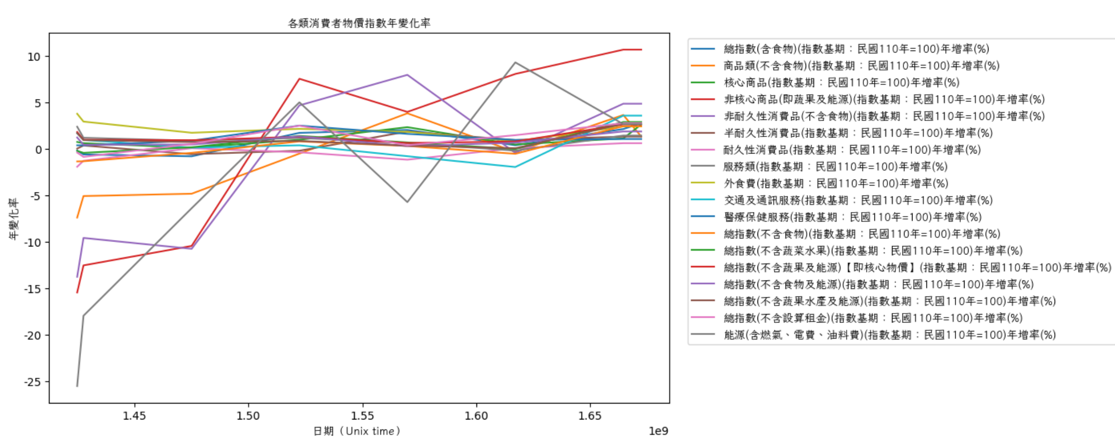
```
ax.set_title('       ', fontproperties=custom_font)
ax.set_xlabel(' Unix time ', fontproperties=custom_font)
ax.set_ylabel('  ', fontproperties=custom_font)
ax.legend(loc='upper left', bbox_to_anchor=(1.02, 1), prop=custom_font)
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_17372\322824476.py:1:
FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a
future version. Use obj.ffill() or obj.bfill() instead.
  subdf = df[['  (  )(    110 =100)  (%)',
'  (  )(    110 =100)  (%)', '  (    110 =100)  (%)',
'   (   )(    110 =100)  (%)', '    (  )(    110 =100)  (%)',
'    (    110 =100)  (%)', '    (    110 =100)  (%)',
'  (    110 =100)  (%)', '  (    110 =100)  (%)',
'    (    110 =100)  (%)', '    (    110 =100)  (%)',
'  (  )(    110 =100)  (%)', '  (   )(    110 =100)  (%)',
'  (   )    (    110 =100)  (%)',
'  (   )(    110 =100)  (%)', '  (     )(    110 =100)  (%)',
'  (  )(    110 =100)  (%)',
' (     )(    110 =100)  (%)']].replace(0,
pd.NA).fillna(method='pad').fillna(method='bfill').set_index(df['  Unix
time ']).drop_duplicates(subset=['  (  )(    110 =100)  (%)'],
keep='last')
```

[73]: `<matplotlib.legend.Legend at 0x20b3dc3f1f0>`

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```



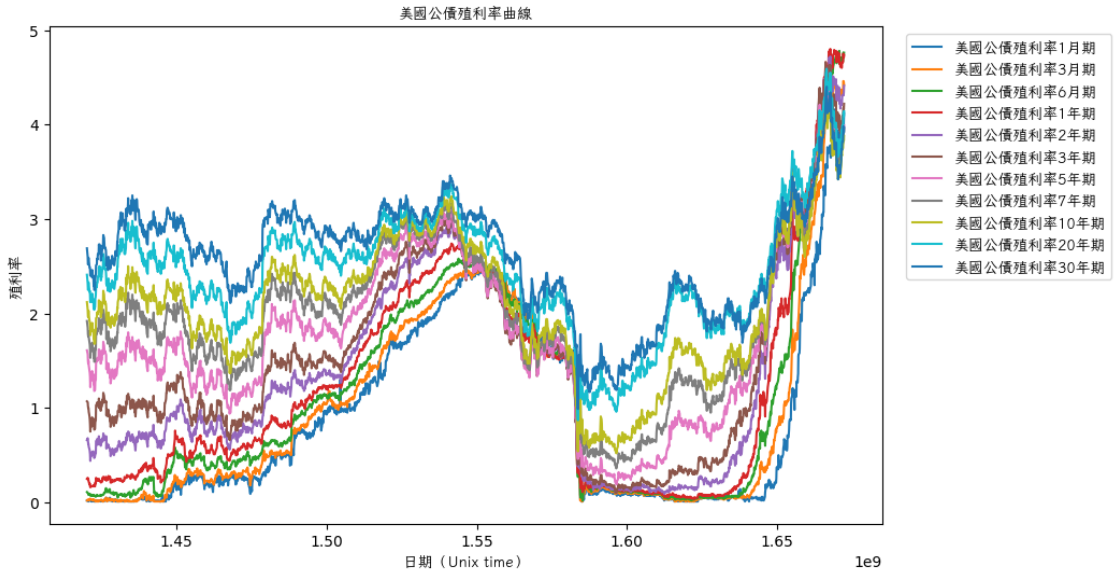
[74]:
```
subdf =␣
 ↪df[['    1 ','    3 ','    6 ','    1 ','    2 ','    3 ','    5 ','    7 ','    10 ','    20
 ↪replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
 ↪set_index(df['  Unix time '])
fig, ax = plt.subplots(figsize=(10, 6))
for column in subdf.columns:
    ax.plot(subdf.index, subdf[column], label=column)
ax.set_title('     ', fontproperties=custom_font)
ax.set_xlabel('  Unix time ', fontproperties=custom_font)
ax.set_ylabel('  ', fontproperties=custom_font)
ax.legend(loc='upper left', bbox_to_anchor=(1.02, 1), prop=custom_font)
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_17372\1434413455.py:1:
FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a
future version. Use obj.ffill() or obj.bfill() instead.
  subdf = df[['    1 ','    3 ','    6 ','    1 ','    2 ','
    3 ','    5 ','    7 ','    10 ','    20 ','    30 ']
].replace(0, pd.NA).fillna(method = 'pad').fillna(method =
'bfill').set_index(df['  Unix time '])
C:\Users\Willie\AppData\Local\Temp\ipykernel_17372\1434413455.py:1:
FutureWarning: Downcasting object dtype arrays on .fillna, .ffill, .bfill is
deprecated and will change in a future version. Call
result.infer_objects(copy=False) instead. To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
  subdf = df[['    1 ','    3 ','    6 ','    1 ','    2 ','
    3 ','    5 ','    7 ','    10 ','    20 ','    30 ']
].replace(0, pd.NA).fillna(method = 'pad').fillna(method =
'bfill').set_index(df['  Unix time '])
```

[74]:
```
<matplotlib.legend.Legend at 0x20b3f1de260>
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```



[75]:
```
subdf =␣
 ↪df[['          ','          ','          ','   10   ','   10   ','   30   ','   30   ','   60
 ↪replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
 ↪set_index(df['  Unix time '])
fig, ax = plt.subplots(figsize=(10, 6))
for column in subdf.columns:
    ax.plot(subdf.index, subdf[column], label=column)
ax.set_title('        ', fontproperties=custom_font)
ax.set_xlabel('  Unix time ', fontproperties=custom_font)
ax.set_ylabel(' ', fontproperties=custom_font)
ax.legend(loc='upper left', bbox_to_anchor=(1.02, 1), prop=custom_font)
```

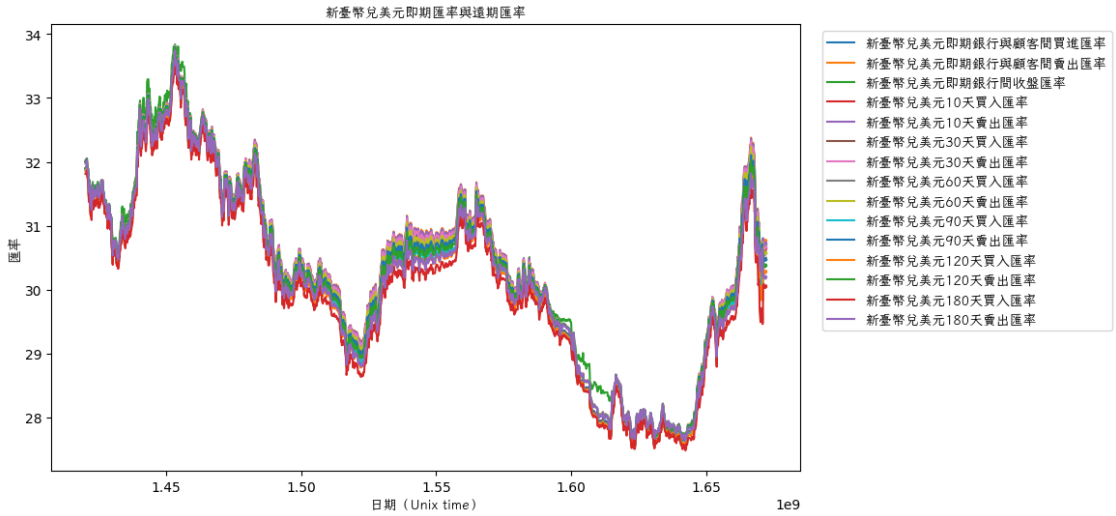
```
C:\Users\Willie\AppData\Local\Temp\ipykernel_17372\3233260870.py:1:
FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a
future version. Use obj.ffill() or obj.bfill() instead.
  subdf = df[['          ','          ','          ','
 10   ','   10   ','   30   ','   30   ','   60   ','
 60   ','   90   ','   90   ','   120   ','   120   ','
  180   ','   180   ']].replace(0, pd.NA).fillna(method =
'pad').fillna(method = 'bfill').set_index(df['  Unix time '])
C:\Users\Willie\AppData\Local\Temp\ipykernel_17372\3233260870.py:1:
FutureWarning: Downcasting object dtype arrays on .fillna, .ffill, .bfill is
deprecated and will change in a future version. Call
result.infer_objects(copy=False) instead. To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
  subdf = df[['          ','          ','          ','
 10   ','   10   ','   30   ','   30   ','   60   ','
 60   ','   90   ','   90   ','   120   ','   120   ','
```

```
 180   ','   180   ']].replace(0, pd.NA).fillna(method =
'pad').fillna(method = 'bfill').set_index(df['  Unix time '])
```

[75]:
```
<matplotlib.legend.Legend at 0x20b41ee8520>
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```



[76]:
```
subdf = df[['     ','     ','     ','     ']].replace(0, pd.NA).fillna(method =␣
↪'pad').fillna(method = 'bfill').set_index(df['  Unix time '])
fig, ax = plt.subplots(figsize=(10, 6))
for column in subdf.columns:
    ax.plot(subdf.index, subdf[column], label=column)
ax.set_title('          ', fontproperties=custom_font)
ax.set_xlabel('  Unix time ', fontproperties=custom_font)
ax.set_ylabel('  ', fontproperties=custom_font)
ax.legend(loc='upper left', bbox_to_anchor=(1.02, 1), prop=custom_font)
```

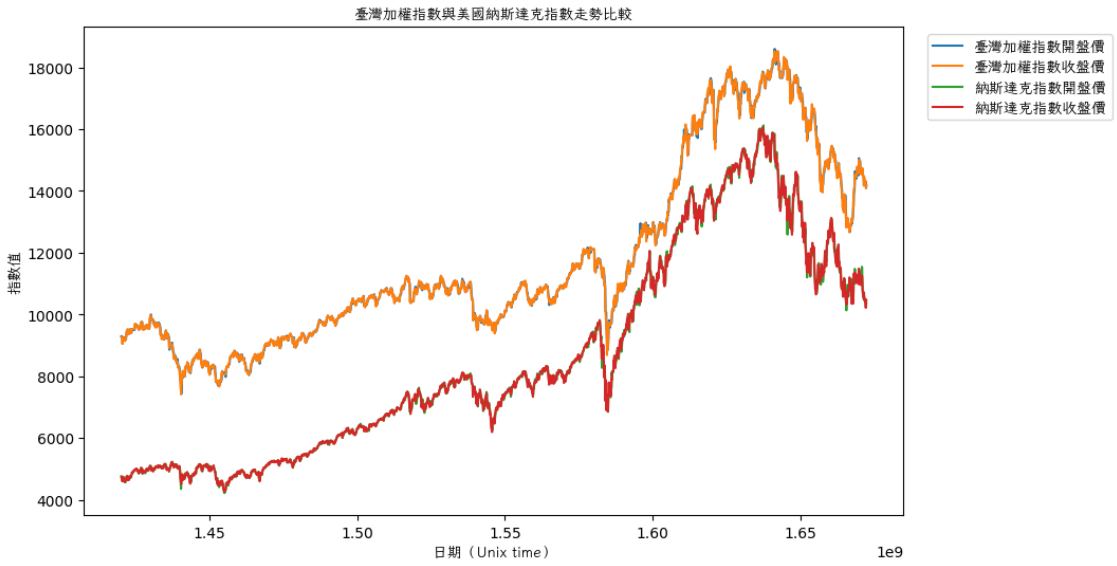
```
C:\Users\Willie\AppData\Local\Temp\ipykernel_17372\760296671.py:1:
FutureWarning: DataFrame.fillna with 'method' is deprecated and will raise in a
future version. Use obj.ffill() or obj.bfill() instead.
  subdf = df[['     ','     ','     ','     ']].replace(0,
pd.NA).fillna(method = 'pad').fillna(method = 'bfill').set_index(df['  Unix
time '])
C:\Users\Willie\AppData\Local\Temp\ipykernel_17372\760296671.py:1:
FutureWarning: Downcasting object dtype arrays on .fillna, .ffill, .bfill is
```

```
deprecated and will change in a future version. Call
result.infer_objects(copy=False) instead. To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
  subdf = df[['     ','     ','     ','     ']].replace(0,
pd.NA).fillna(method = 'pad').fillna(method = 'bfill').set_index(df['  Unix
time '])
```

[76]:
```
<matplotlib.legend.Legend at 0x20b41e18160>
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```



[10]:
```
subdf = df[['     ','     ',' 1     ',' 2     ',' 3     ',' 4     ',' 5     ']].
  ↪replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
  ↪set_index(df['  Unix time '])
fig, ax = plt.subplots(figsize=(10, 6))
for column in subdf.columns:
    ax.plot(subdf.index, subdf[column], label=column)
ax.set_title('       ', fontproperties=custom_font)
ax.set_xlabel('  Unix time ', fontproperties=custom_font)
```

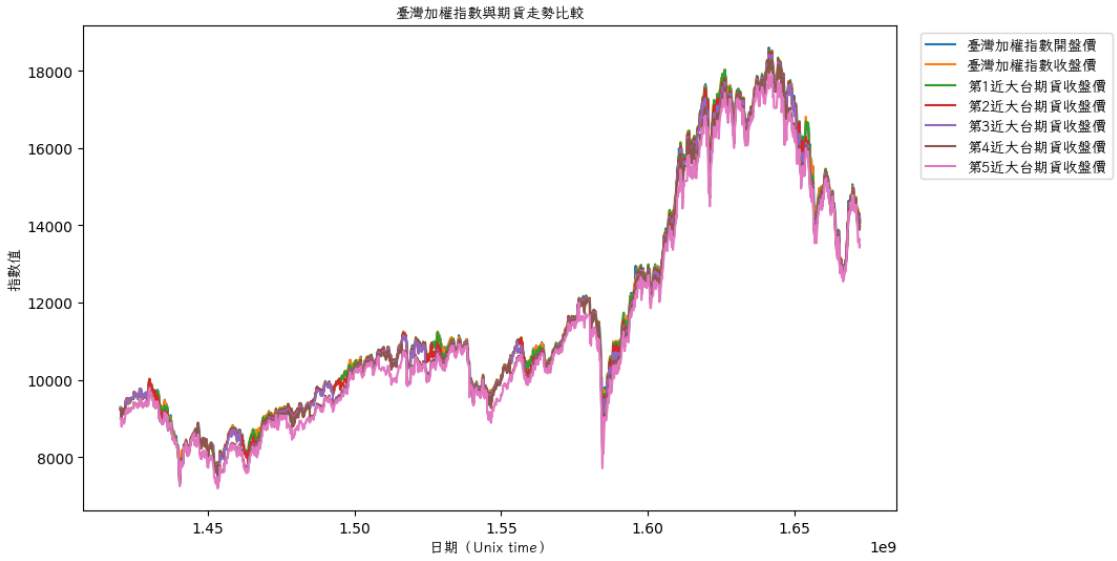
```python
ax.set_ylabel('  ', fontproperties=custom_font)
ax.legend(loc='upper left', bbox_to_anchor=(1.02, 1), prop=custom_font)
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\2193609048.py:1:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\2193609048.py:1:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[10]:
```
<matplotlib.legend.Legend at 0x216f87aa350>
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```



[27]:

```
import plotly.express as px
fig = px.scatter(
    df.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
  ↪set_index(df['  Unix time ']), x='     ', y=' 1     ', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
```

```
)

fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\2387428399.py:3:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\2387428399.py:3:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[28]:
```
fig = px.scatter(
    df.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
 ↪set_index(df['  Unix time ']), x='     ', y=' 2     ', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\4018956947.py:2:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\4018956947.py:2:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[29]:
```
fig = px.scatter(
    df.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
 ↪set_index(df['  Unix time ']), x=' 1     ', y=' 2     ', color='     ',␣
 ↪opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)
```

```
fig.show()
```

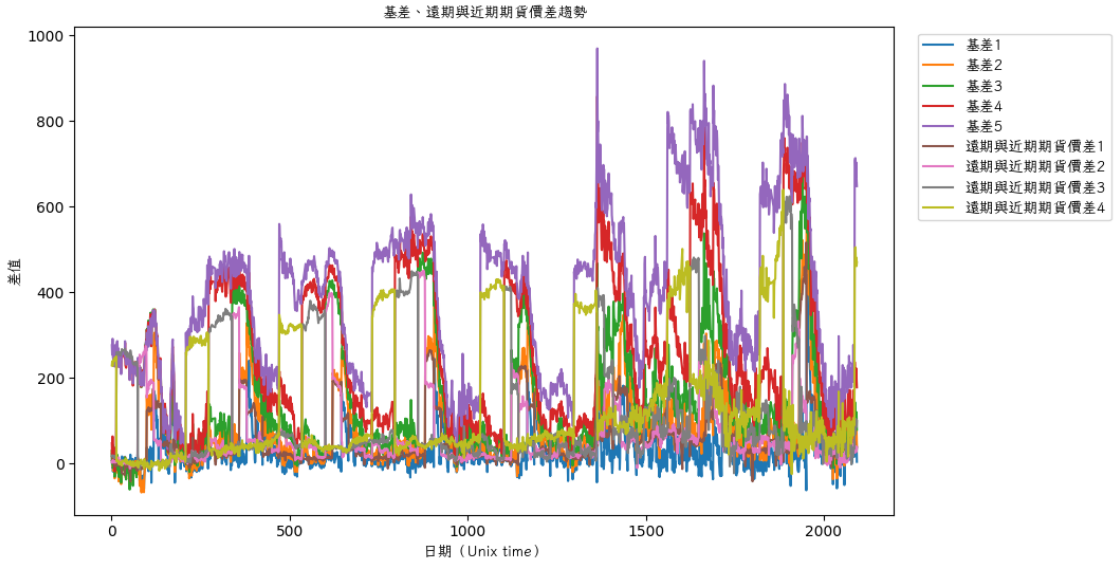
```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\4001763095.py:2:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\4001763095.py:2:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[30]:
```
tdf = df[[' Unix time ', '     ', ' 1    ', ' 2    ', ' 3    ', ' 4    ',␣
 ↪' 5    ']].replace(0, pd.NA).dropna()
idf=pd.DataFrame()
idf[' 1'] = tdf['     '] - tdf[' 1    ']
idf[' 2'] = tdf['     '] - tdf[' 2    ']
idf[' 3'] = tdf['     '] - tdf[' 3    ']
idf[' 4'] = tdf['     '] - tdf[' 4    ']
idf[' 5'] = tdf['     '] - tdf[' 5    ']
idf['     1'] = tdf[' 1    '] - tdf[' 2    ']
idf['     2'] = tdf[' 2    '] - tdf[' 3    ']
idf['     3'] = tdf[' 3    '] - tdf[' 4    ']
idf['     4'] = tdf[' 4    '] - tdf[' 5    ']
idf.set_index(tdf[' Unix time '])
fig, ax = plt.subplots(figsize=(10, 6))
for column in idf.columns:
    ax.plot(idf.index, idf[column], label=column)
ax.set_title('       ', fontproperties=custom_font)
ax.set_xlabel(' Unix time ', fontproperties=custom_font)
ax.set_ylabel(' ', fontproperties=custom_font)
ax.legend(loc='upper left', bbox_to_anchor=(1.02, 1), prop=custom_font)
```

[30]:
```
<matplotlib.legend.Legend at 0x2168fafd1b0>
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```

```
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
findfont: Font family 'Iansui' not found.
```



[31]:
```
idf['     ']=tdf['     ']
fig = px.scatter(
    idf, x='     ', y=' 1', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[32]:
```
fig = px.scatter(
    idf, x='     ', y=' 2', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[33]:
```
fig = px.scatter(
    idf, x='     ', y=' 3', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[34]:
```
fig = px.scatter(
    idf, x='     ', y=' 4', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[35]:
```
fig = px.scatter(
    idf, x='     ', y=' 5', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[36]:
```
fig = px.scatter(
    idf, x='     ', y='     1', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[37]:
```
fig = px.scatter(
    idf, x='     ', y='     2', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[38]:
```
fig = px.scatter(
    idf, x='     ', y='     3', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[39]:
```
fig = px.scatter(
    idf, x='     ', y='     4', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

[40]:
```
fig = px.scatter(
    df.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
 ↪set_index(df['  Unix time ']), x='     ', y='     ', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\3203966008.py:2:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\3203966008.py:2:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[41]:
```
fig = px.scatter_3d(df.replace(0, pd.NA).fillna(method = 'pad').fillna(method =␣
↪'bfill').set_index(df['  Unix time ']), x='     ', y=' 1     ', z=' 2     ',␣
↪color=' 3     ')
fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\53588074.py:1: FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\53588074.py:1: FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[42]:
```
fig = px.scatter(
    df.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
↪set_index(df['  Unix time ']), x='     ', y='     ', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\3353602965.py:2:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\3353602965.py:2:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
```

`pd.set_option('future.no_silent_downcasting', True)`

[43]:
```
fig = px.scatter(
    df.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
 ↪set_index(df[' Unix time ']), x='     ', y=' 1    10     ', opacity=0.65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\272649523.py:2:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\272649523.py:2:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[44]:
```
fig = px.scatter(
    df.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill').
 ↪set_index(df[' Unix time ']), x=' 1    10     ', y=' 1    10     ', opacity=0.
 ↪65,
    trendline='ols', trendline_color_override='darkblue'
)

fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\99133955.py:2: FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\99133955.py:2: FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[45]:
```
fig = px.scatter(df.replace(0, pd.NA).fillna(method = 'pad').fillna(method =␣
 ↪'bfill').set_index(df['  Unix time ']), x="     ", y="     ",␣
 ↪color='          ')
fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\952016150.py:1:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\952016150.py:1:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[46]:
```
ndf=df.replace(0, pd.NA).fillna(method = 'pad').fillna(method =␣
 ↪'bfill')[['          ']]
ndf['     /     ']=df['     ']/df['     ']
ndf.set_index(df.replace(0, pd.NA).fillna(method = 'pad').fillna(method =␣
 ↪'bfill')['  Unix time '])
ndf.replace(0, pd.NA).fillna(method = 'pad').fillna(method = 'bfill')
fig = px.scatter(ndf, x='          ', y='     /     ')
fig.show()
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\437625938.py:1:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\437625938.py:1:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\437625938.py:3:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.
```

```
C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\437625938.py:3:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\437625938.py:4:
FutureWarning:

DataFrame.fillna with 'method' is deprecated and will raise in a future version.
Use obj.ffill() or obj.bfill() instead.

C:\Users\Willie\AppData\Local\Temp\ipykernel_11284\437625938.py:4:
FutureWarning:

Downcasting object dtype arrays on .fillna, .ffill, .bfill is deprecated and
will change in a future version. Call result.infer_objects(copy=False) instead.
To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
```

[47]:
```
%%writefile matrix.hpp
#ifndef MATRIX_H
#define MATRIX_H

#include <iostream>
#include <vector>
#include <cassert>
#include <functional>
#include <cmath>
#include <algorithm>

using namespace std;

class matrix {
public:
    int rows;
    int cols;
    vector<double> vals;

    matrix(int _rows, int _cols) {
        rows = _rows;
        cols = _cols;
        vals.resize(cols * rows, 0);
```

```
    }

    matrix() {
        rows = 0;
        cols = 0;
        vals = {};
    }

    matrix dot(matrix a) {
        assert(cols == a.rows);
        if (cols != a.rows) cout << "dot error" << endl;
        matrix temp(rows, a.cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < a.cols; j++) {
                for (int k = 0; k < a.rows; k++) {
                    double mult = vals[i * cols + k] * a.vals[k * a.cols + j];
                    temp.vals[i * a.cols + j] += mult;
                }
            }
        }
        return temp;
    }

    matrix transpose() {
        matrix temp(cols, rows);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[j * rows + i] = vals[i * cols + j];
            }
        }
        return temp;
    }

    matrix add(matrix a) {
        assert(a.rows == rows && a.cols == cols);
        matrix temp(rows, cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[i * cols + j] = vals[i * cols + j] + a.vals[i * cols␣
↪+ j];
            }
        }
        return temp;
    }

    matrix minus(matrix a) {
        assert(a.rows == rows && a.cols == cols);
```

```
        matrix temp(rows, cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[i * cols + j] = vals[i * cols + j] - a.vals[i * cols␣
↪+ j];
            }
        }
        return temp;
    }

    matrix multiply_scalar(double x) {
        matrix temp(rows, cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[i * cols + j] = vals[i * cols + j] * x;
            }
        }
        return temp;
    }

    matrix element_add(double x) {
        matrix temp(rows, cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[i * cols + j] = vals[i * cols + j] + x;
            }
        }
        return temp;
    }

    matrix multiply_element(matrix a) {
        assert(a.rows == rows && a.cols == cols);
        matrix temp(rows, cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[i * cols + j] = vals[i * cols + j] * a.vals[i * cols␣
↪+ j];
            }
        }
        return temp;
    }

    matrix negative() {
        matrix temp(rows, cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[i * cols + j] = -vals[i * cols + j];
```

```
            }
        }
        return temp;
    }

    matrix apply_func(function<double(const double&)> func) {
        matrix temp(rows, cols);
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                temp.vals[i * cols + j] = func(vals[i * cols + j]);
            }
        }
        return temp;
    }

    double mean() const {
        double sum = 0.0;
        int num = rows * cols;
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                sum += vals[i * cols + j];
            }
        }
        return sum / (double)num;
    }
};

#endif
```

```
Overwriting matrix.hpp
```

[48]:
```
%%writefile RNN.hpp
#ifndef RNN_H
#define RNN_H

#include <iostream>
#include <cmath>
#include <vector>
#include <algorithm>
#include <fstream>
#include <sstream>
#include <math.h>
#include "matrix.hpp"
using namespace std;

double random_double(const double &min, const double &max)
{
```

```
        return ((double)rand() / RAND_MAX) * (max - min) + min;
}

double mse_seq(vector<matrix> output, vector<matrix> target)
{
        int n = output.size();
        double result = 0.0;

        for (int i = 0; i < n; i++)
        {
                matrix err = output[i].minus(target[i]);

                for (int j = 0; j < err.vals.size(); j++)
                        result += pow(err.vals[j], 2);
        }

        result /= n * output[0].vals.size();

        return result;
}

matrix random_matrix(const double &l, const double &r, const int &row, const␣
 ↪int &col)
{
        matrix temp(row, col);
        for (int i = 0; i < row * col; i++)
        {
                temp.vals[i] = random_double(l, r);
        }

        return temp;
}

matrix tanh_function(matrix v)
{
        for (int i = 0; i < v.vals.size(); i++)
                v.vals[i] = tanh(v.vals[i]);

        return v;
}

matrix clip(matrix a, const double &mn, const double &mx)
{
        for (int i = 0; i < a.vals.size(); i++)
                a.vals[i] = clamp(a.vals[i], mn, mx);

        return a;
```

```
}

vector<matrix> mask_seq(vector<matrix> input_seq, int signal_size)
{
        int n = input_seq.size();
        vector<matrix> res = input_seq;

        for (int i = 0; i < n; ++i)
        {
                for (int j = 0; j < signal_size; ++j)
                {
                        res[i].vals[j] = 1.0;
                }
        }

        return res;
}

vector<matrix> convert_input_masks_to_hidden_masks(const vector<matrix>␣
 ↪&input_masks, const int &hidden_size)
{
        int n = input_masks.size();
        vector<matrix> hidden_masks(n);

        for (int i = 0; i < n; ++i)
        {
                int rows = hidden_size;
                int cols = 1;
                matrix mask(rows, cols);

                double mask_value = all_of(input_masks[i].vals.begin(),␣
 ↪input_masks[i].vals.end(),
                                                                        [](double␣
 ↪v) { return v == 1.0; })
                                                                        ? 1.0
                                                                        : 0.0;

                for (int j = 0; j < rows * cols; ++j)
                {
                        mask.vals[j] = mask_value;
                }

                hidden_masks[i] = mask;
        }

        return hidden_masks;
}
```

```
class rnn
{
  public:
        int hidden_size;
        int signal_size;
        int seq_length;
        double learning_rate;
        matrix U;
        matrix V;
        matrix W;
        matrix b;
        matrix c;
        string message;

        rnn(const int &h_s, const int &s_s, const int &s_l, const double &l_r)
        {
                hidden_size = h_s;
                signal_size = s_s;
                seq_length = s_l;
                learning_rate = l_r;

                U = random_matrix(-1 * sqrt(1.0 / double(signal_size)), sqrt(1.
 ↪0 / double(signal_size)), hidden_size, signal_size);
                V = random_matrix(-1 * sqrt(1.0 / double(hidden_size)), sqrt(1.
 ↪0 / double(hidden_size)), signal_size, hidden_size);
                W = random_matrix(-1 * sqrt(1.0 / double(hidden_size)), sqrt(1.
 ↪0 / double(hidden_size)), hidden_size, hidden_size);
                matrix b_(hidden_size, 1);
                matrix c_(signal_size, 1);
                b = b_;
                c = c_;
        }

        vector<vector<matrix>> forward(vector<matrix> inputs, matrix h_0,␣
 ↪vector<matrix> input_masks, vector<matrix> hidden_masks)
        {
                int sz = inputs.size();
                vector<matrix> X(sz), H(sz), O(sz);

                for (int i = 0; i < sz; i++)
                {
                        X[i] = inputs[i];

                        matrix h;

                        if (i)
```

```
                        h = H[i - 1];
                else
                        h = h_0;

                H[i] = tanh_function(U.dot(X[i]).add(W.dot(h)).add(b));
                H[i] = H[i].multiply_element(hidden_masks[i]);

                O[i] = V.dot(H[i]).add(c);
                O[i] = O[i].multiply_element(input_masks[i]);
        }

        return {X, H, O};
    }

    void backward(vector<matrix> X, vector<matrix> H, vector<matrix> O,␣
↪vector<matrix> targets, matrix h_0, vector<matrix> input_masks,␣
↪vector<matrix> hidden_masks)
    {
        matrix dU(hidden_size, signal_size);
        matrix dW(hidden_size, hidden_size);
        matrix dV(signal_size, hidden_size);
        matrix db(hidden_size, 1);
        matrix dc(signal_size, 1);
        matrix dhnext(hidden_size, 1);

        for (int i = seq_length - 1; i >= 0; i--)
        {
                matrix dy = O[i].minus(targets[i]).multiply_scalar(2.0 /
↪ double(signal_size));
                dy = dy.multiply_element(input_masks[i]);

                dV = dV.add(dy.dot(H[i].transpose()));
                dc = dc.add(dy);

                matrix dh = V.transpose().dot(dy).add(dhnext);
                dh = dh.multiply_element(hidden_masks[i]);

                matrix dhrec = H[i].multiply_element(H[i]).negative().
↪element_add(1.0).multiply_element(dh);
                db = db.add(dhrec);

                dU = dU.add(dhrec.dot(X[i].transpose()));
                if (i)
                        dW = dW.add(dhrec.dot(H[i - 1].transpose()));
                else
                        dW = dW.add(dhrec.dot(h_0.transpose()));
```

```
                        dhnext = W.transpose().dot(dhrec);
                }

                dU = clip(dU, -5.0, 5.0);
                dW = clip(dW, -5.0, 5.0);
                dV = clip(dV, -5.0, 5.0);
                db = clip(db, -5.0, 5.0);
                dc = clip(dc, -5.0, 5.0);

                U = U.minus(dU.multiply_scalar(learning_rate));
                W = W.minus(dW.multiply_scalar(learning_rate));
                V = V.minus(dV.multiply_scalar(learning_rate));
                b = b.minus(db.multiply_scalar(learning_rate));
                c = c.minus(dc.multiply_scalar(learning_rate));
        }

        void train(vector<vector<matrix>> input_data, vector<vector<matrix>>␣
↪targets, int epochs)
        {
                matrix h0 = random_matrix(-1 * sqrt(1.0 / double(hidden_size)),␣
↪sqrt(1.0 / double(hidden_size)), hidden_size, 1);
                int n = input_data.size();

                for (int t = 0; t < epochs; ++t)
                {
                        double error = 0.0;

                        for (int i = 0; i < n; ++i)
                        {
                                vector<matrix> inputs = input_data[i];

                                vector<matrix> input_masks = mask_seq(inputs,␣
↪signal_size);

                                vector<matrix> hidden_masks =␣
↪convert_input_masks_to_hidden_masks(input_masks, hidden_size);

                                vector<vector<matrix>> XHO = forward(inputs,␣
↪h0, input_masks, hidden_masks);

                                vector<matrix> target = targets[i];

                                error += mse_seq(XHO[2], target);

                                backward(XHO[0], XHO[1], XHO[2], target, h0,␣
↪input_masks, hidden_masks);
```

```
                        }

                        error /= double(n);

            ostringstream oss;
                        oss << "epoch " << t + 1 << "/" << epochs <<␣
↪"------------------"
                                    << "mse : " << error << endl;
            cout << oss.str();
            message = message + oss.str();
                }
        }

        void save_weights(const string& filename) {
        ofstream file(filename, ios::binary);
        if (!file.is_open()) {
            cerr << "Failed to open file: " << filename << endl;
            return;
        }

        // Write U
        file.write(reinterpret_cast<const char*>(&U.rows), sizeof(int));
        file.write(reinterpret_cast<const char*>(&U.cols), sizeof(int));
        file.write(reinterpret_cast<const char*>(U.vals.data()), U.vals.size()␣
↪* sizeof(double));

        // Write V
        file.write(reinterpret_cast<const char*>(&V.rows), sizeof(int));
        file.write(reinterpret_cast<const char*>(&V.cols), sizeof(int));
        file.write(reinterpret_cast<const char*>(V.vals.data()), V.vals.size()␣
↪* sizeof(double));

        // Write W
        file.write(reinterpret_cast<const char*>(&W.rows), sizeof(int));
        file.write(reinterpret_cast<const char*>(&W.cols), sizeof(int));
        file.write(reinterpret_cast<const char*>(W.vals.data()), W.vals.size()␣
↪* sizeof(double));

        // Write b
        file.write(reinterpret_cast<const char*>(&b.rows), sizeof(int));
        file.write(reinterpret_cast<const char*>(&b.cols), sizeof(int));
        file.write(reinterpret_cast<const char*>(b.vals.data()), b.vals.size()␣
↪* sizeof(double));

        // Write c
        file.write(reinterpret_cast<const char*>(&c.rows), sizeof(int));
        file.write(reinterpret_cast<const char*>(&c.cols), sizeof(int));
```

```
        file.write(reinterpret_cast<const char*>(c.vals.data()), c.vals.size()␣
↪* sizeof(double));

        // Write message
        const char* msg_data = message.c_str();
        uint32_t msg_size = static_cast<uint32_t>(message.size());
        file.write(reinterpret_cast<const char*>(&msg_size), sizeof(uint32_t));
        file.write(msg_data, msg_size);

        file.close();
        cout << "Weights saved to " << filename << endl;
    }

    void load_weights(const string& filename) {
        ifstream file(filename, ios::binary);
        if (!file.is_open()) {
            cerr << "Failed to open file: " << filename << endl;
            return;
        }

        // Read U
        int rows, cols;
        file.read(reinterpret_cast<char*>(&rows), sizeof(int));
        file.read(reinterpret_cast<char*>(&cols), sizeof(int));
        U = matrix(rows, cols);
        file.read(reinterpret_cast<char*>(U.vals.data()), U.vals.size() *␣
↪sizeof(double));

        // Read V
        file.read(reinterpret_cast<char*>(&rows), sizeof(int));
        file.read(reinterpret_cast<char*>(&cols), sizeof(int));
        V = matrix(rows, cols);
        file.read(reinterpret_cast<char*>(V.vals.data()), V.vals.size() *␣
↪sizeof(double));

        // Read W
        file.read(reinterpret_cast<char*>(&rows), sizeof(int));
        file.read(reinterpret_cast<char*>(&cols), sizeof(int));
        W = matrix(rows, cols);
        file.read(reinterpret_cast<char*>(W.vals.data()), W.vals.size() *␣
↪sizeof(double));

        // Read b
        file.read(reinterpret_cast<char*>(&rows), sizeof(int));
        file.read(reinterpret_cast<char*>(&cols), sizeof(int));
        b = matrix(rows, cols);
```

```
        file.read(reinterpret_cast<char*>(b.vals.data()), b.vals.size() *␣
 ↪sizeof(double));

        // Read c
        file.read(reinterpret_cast<char*>(&rows), sizeof(int));
        file.read(reinterpret_cast<char*>(&cols), sizeof(int));
        c = matrix(rows, cols);
        file.read(reinterpret_cast<char*>(c.vals.data()), c.vals.size() *␣
 ↪sizeof(double));

        // Read the custom message
        uint32_t msg_size;
        file.read(reinterpret_cast<char*>(&msg_size), sizeof(uint32_t));
        message.resize(msg_size);
        file.read(&message[0], msg_size);

        file.close();
        cout << "Weights loaded from " << filename << endl;
        cout << "Message: " << endl << message << endl;
    }
};

#endif
```

Overwriting RNN.hpp

[49]:
```
%%writefile rnnmain.hpp
#ifndef RNNMAIN_H
#define RNNMAIN_H

#include "RNN.hpp"
#include "matrix.hpp"
#include <fstream>
#include <sstream>
#include <tuple>
using namespace std;

vector<matrix> neg_vec_matrix(vector<matrix> vm)
{
        for (int i = 0; i < vm.size(); i++)
                vm[i] = vm[i].negative();

        return vm;
}

vector<matrix> normalized_seq(vector<matrix> seq)
{
```

```
        for (int i = 0; i < seq.size(); i++)
        {
                vector<double> temp = seq[i].vals;

                double minVal = *min_element(temp.begin(), temp.end());
                double maxVal = *max_element(temp.begin(), temp.end());

                if (maxVal == minVal)
                        maxVal += 1e-9;

                for (int j = 0; j < temp.size(); j++)
                        temp[j] = (temp[j] - minVal) / (maxVal - minVal);

                seq[i].vals = temp;
        }

        return seq;
}

int trainModel(int starty, int endy, int _seq_length, int _hidden_size, double␣
 ↪_learning_rate, int _epoch, string _filename)
{
        vector<matrix> vm;
        string line, cell;

        vector<string> files;
        for (int i = starty; i < endy; i++)
        {
                files.push_back("completeData/20" + to_string(i) +␣
 ↪"completeData.csv");
        }

        for (size_t i = 0; i < files.size(); i++)
        {
                ifstream data(files[i]);
                if (!data.is_open())
                        return 1;
                getline(data, line);
                while (getline(data, line))
                {
                        istringstream iss(line);
                        vector<double> sv;
                        while (getline(iss, cell, ','))
                        {
                                sv.push_back(stod(cell));
                        }
                        matrix mat(1, 6706);
```

```
                        mat.vals = sv;
                        mat = mat.transpose();
                        vm.push_back(mat);
                }
                data.close();
        }

        vm = normalized_seq(vm);

    vector<vector<matrix>> sequences;
    int seq_length = _seq_length;

    for(int i = 0; i < vm.size() - seq_length; i += seq_length){
        vector<matrix> temp;

        for(int j = i; j < i + seq_length; j++)
            temp.push_back(vm[j]);

        sequences.push_back(temp);
    }

    cout<<sequences.size()<<" sequences"<<endl;

    int hidden_size = _hidden_size;
    int signal_size = 6706;
    double learning_rate = _learning_rate;
    matrix h0 = random_matrix(-1 * sqrt(1.0 / double(hidden_size)), sqrt(1.0 /␣
 ↪double(hidden_size)), hidden_size, 1);

    rnn tra(hidden_size, signal_size, seq_length, learning_rate);

    int epoch = _epoch;
    vector<vector<matrix>> inputs(sequences.begin(), sequences.end() - 1);
    vector<vector<matrix>> outputs(sequences.begin() + 1, sequences.end());
    tra.train(inputs, outputs, epoch);

    tra.save_weights(_filename);

        return 0;
}

tuple<vector<vector<matrix>>, vector<double>> load(int starty, int endy, string␣
 ↪_filename) {
    rnn my_rnn(0,0,0,0);
    my_rnn.load_weights(_filename);
    vector<vector<matrix>> predictions;
    vector<double> msev;
```

```
/*Haven't been tested yet because we don't have time. Uncomment to use.
    vector<matrix> vm;
        string line, cell;

        vector<string> files;
        for (int i = starty; i < endy; i++)
        {
                files.push_back("completeData/20" + to_string(i) +␣
 ↪"completeData.csv");
        }

        for (size_t i = 0; i < files.size(); i++)
        {
                ifstream data(files[i]);
                if (!data.is_open())
                        return make_tuple(predictions, msev);
                getline(data, line);
                while (getline(data, line))
                {
                        istringstream iss(line);
                        vector<double> sv;
                        while (getline(iss, cell, ','))
                        {
                                sv.push_back(stod(cell));
                        }
                        matrix mat(1, 6706);
                        mat.vals = sv;
                        mat = mat.transpose();
                        vm.push_back(mat);
                }
                data.close();
        }

        vm = normalized_seq(vm);

    vector<vector<matrix>> sequences;
    int seq_length = my_rnn.seq_length;

    for(int i = 0; i < vm.size() - seq_length; i += seq_length){
        vector<matrix> temp;

        for(int j = i; j < i + seq_length; j++)
            temp.push_back(vm[j]);

        sequences.push_back(temp);
    }
```

```
    cout<<sequences.size()<<" sequences"<<endl;

    vector<vector<matrix>> input_data(sequences.begin(), sequences.end() - 1);
    vector<vector<matrix>> targets(sequences.begin() + 1, sequences.end());

    matrix h0 = random_matrix(-1 * sqrt(1.0 / double(my_rnn.hidden_size)),␣
 ↪sqrt(1.0 / double(my_rnn.hidden_size)), my_rnn.hidden_size, 1);

    for (int i = 0; i < input_data.size(); ++i) {
        vector<matrix> input_masks = mask_seq(input_data[i], my_rnn.
 ↪signal_size);
        vector<matrix> hidden_masks =␣
 ↪convert_input_masks_to_hidden_masks(input_masks, my_rnn.hidden_size);
        predictions = my_rnn.forward(input_data[i], h0, input_masks,␣
 ↪hidden_masks);

        double mse = mse_seq(predictions[2], targets[i]);
        msev.push_back(mse);
        cout << "MSE for sequence " << i << ": " << mse << endl;
    }
*/
    return make_tuple(predictions, msev);
}

#endif
```

Overwriting rnnmain.hpp

[50]:
```
%%writefile rnnmain1.cpp
#include "rnnmain.hpp"

int main() {
    cout << "Training Model: 15 21 10 100 0.01 5" << endl;
        int a = trainModel(15, 21, 10, 100, 0.01, 5,␣
 ↪"trainedModel_15_21_10_100_0.01_5");
    tuple<vector<vector<matrix>>, vector<double>> b = load(21, 23,␣
 ↪"trainedModel_15_21_10_100_0.01_5");
        return 0;
}
```

Overwriting rnnmain1.cpp

[51]:
```
#!g++ -std=c++17 rnnmain1.cpp -o rnnmain1.exe
# We've run it on Code::Block for efficiency. Uncomment to run.
```

[52]:
```
#!rnnmain1.exe
# We've run it on Code::Block for efficiency. Uncomment to run.
# Result:
```

```
'''
Training Model: 15 21 10 100 0.01 5
157 sequences
epoch 1/5-----------------mse : 0.000292019
epoch 2/5-----------------mse : 0.000277335
epoch 3/5-----------------mse : 0.000270005
epoch 4/5-----------------mse : 0.000264911
epoch 5/5-----------------mse : 0.000260859
Weights saved to trainedModel_15_21_10_100_0.01_5
Weights loaded from trainedModel_15_21_10_100_0.01_5
Message:
epoch 1/5-----------------mse : 0.000292019
epoch 2/5-----------------mse : 0.000277335
epoch 3/5-----------------mse : 0.000270005
epoch 4/5-----------------mse : 0.000264911
epoch 5/5-----------------mse : 0.000260859


Process returned 0 (0x0)   execution time : 803.395 s
Press any key to continue.
'''
```

[52]:
```
'\nTraining Model: 15 21 10 100 0.01 5\n157 sequences\nepoch 1/5
-----------------mse : 0.000292019\nepoch 2/5-----------------mse :
0.000277335\nepoch 3/5-----------------mse : 0.000270005\nepoch 4/5
-----------------mse : 0.000264911\nepoch 5/5-----------------mse :
0.000260859\nWeights saved to trainedModel_15_21_10_100_0.01_5\nWeights loaded
from trainedModel_15_21_10_100_0.01_5\nMessage:\nepoch 1/5-----------------mse :
0.000292019\nepoch 2/5-----------------mse : 0.000277335\nepoch 3/5
-----------------mse : 0.000270005\nepoch 4/5-----------------mse :
0.000264911\nepoch 5/5-----------------mse : 0.000260859\n\n\nProcess returned 0
(0x0)   execution time : 803.395 s\nPress any key to continue.\n'
```

[ ]:
```

```