#allin

Seeq®

conneqt

ENERGY: OIL & GAS

Kyle Miller

Marathon Pipeline

Pressure Cycle Counting is Easy in Data Lab!

Kyle Miller, P. E.

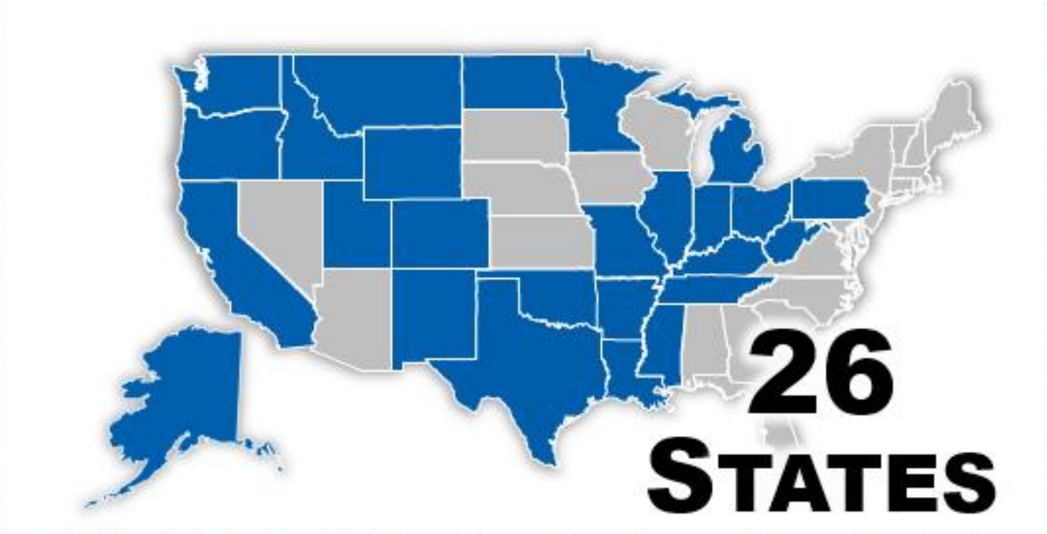
Innovation Manager

Marathon Pipe Line LLC

conneqt

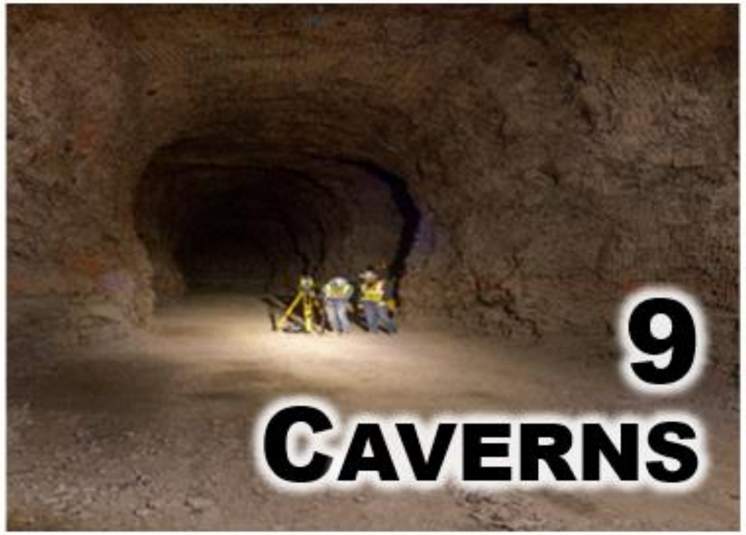
Marathon Pipe Line – By The Numbers

~1,000 Employees



26 States

10,000 mi of Pipeline

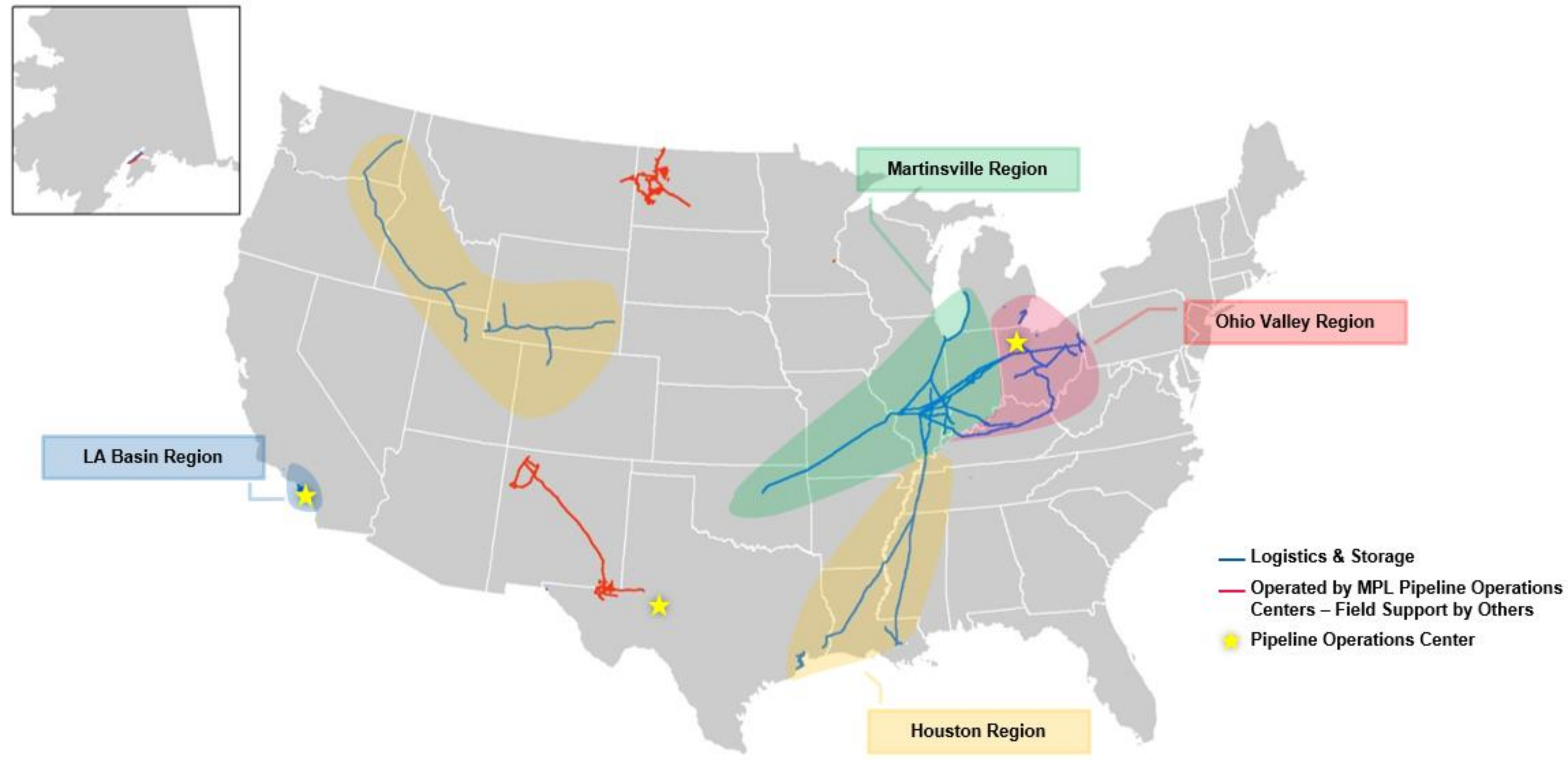


9 Caverns

3 Operations Centers

360 Tanks

Marathon Pipe Line – Operating Map


Martinsville Region
Ohio Valley Region
LA Basin Region
Houston Region
Logistics & Storage
Operated by MPL Pipeline Operations Centers – Field Support by Others
Pipeline Operations Center

The Amazing Soda Can

What is Pressure Cycling?

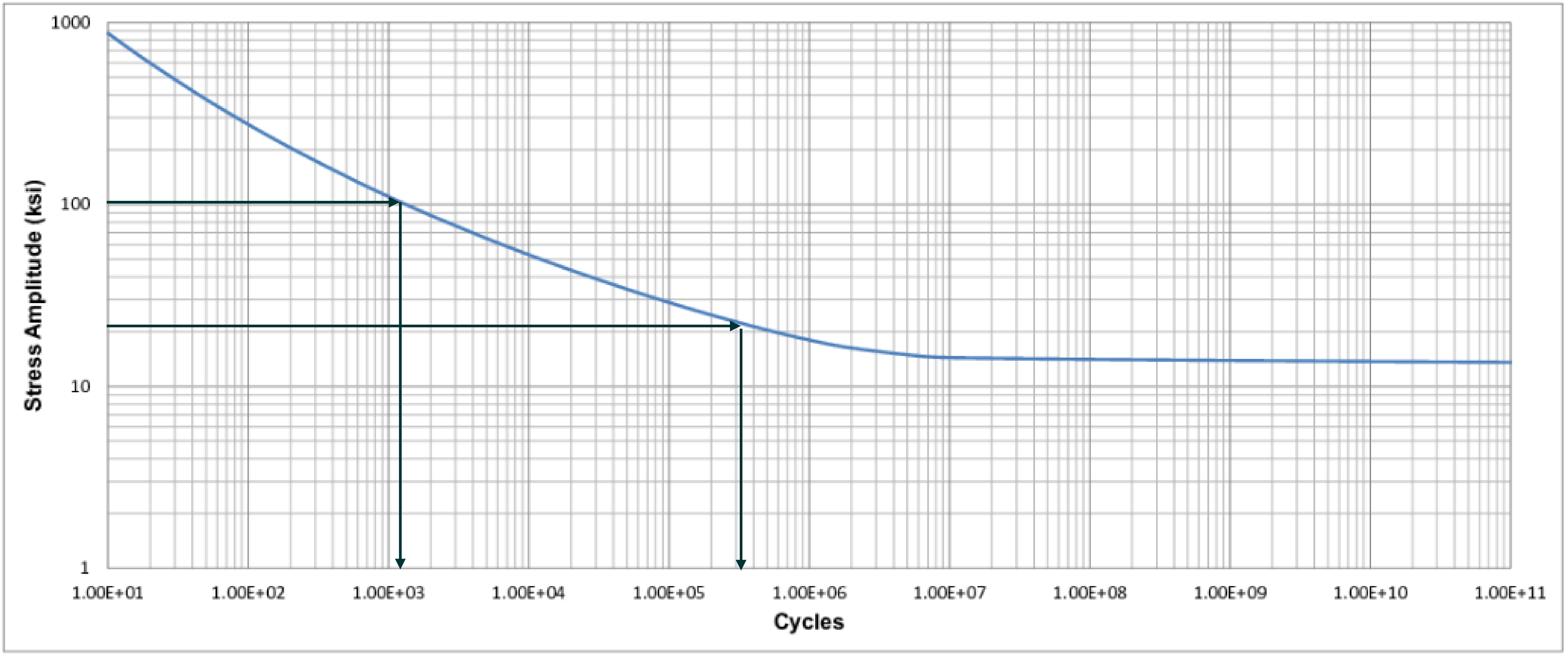
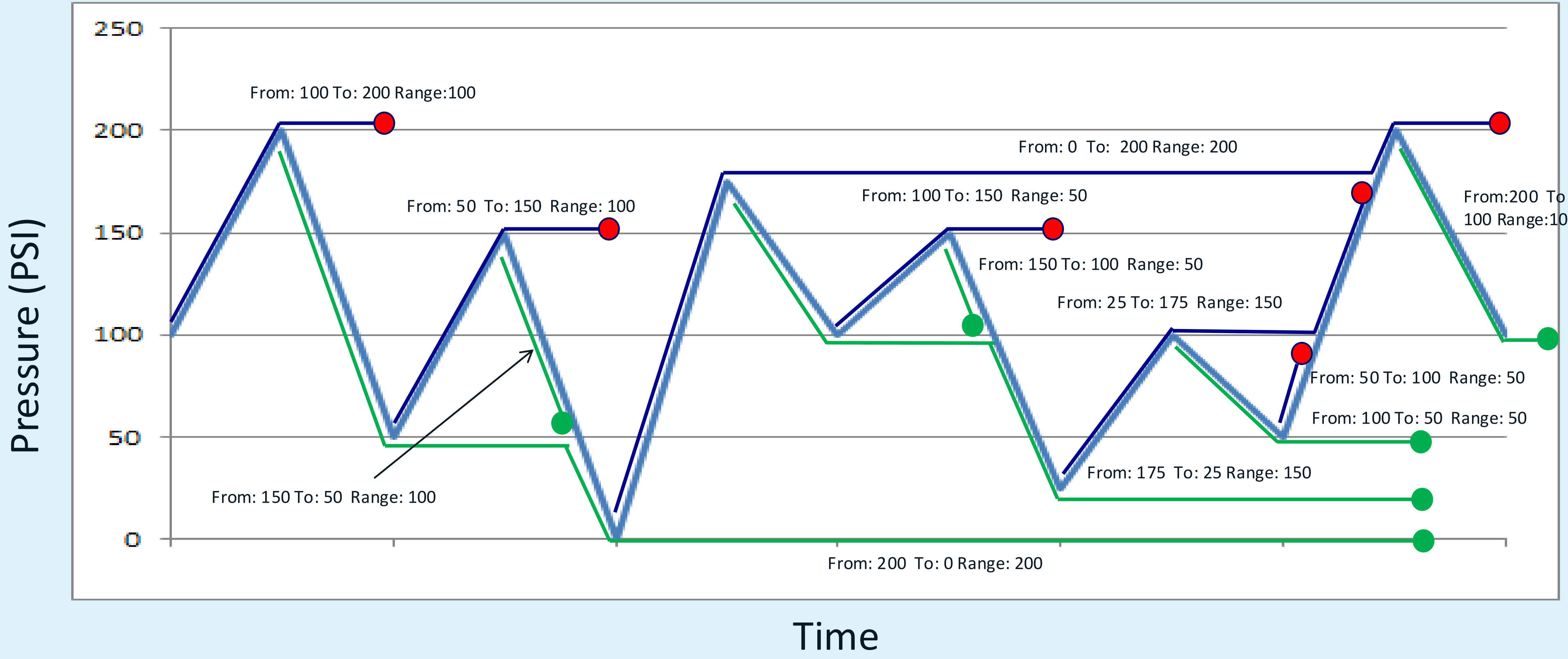
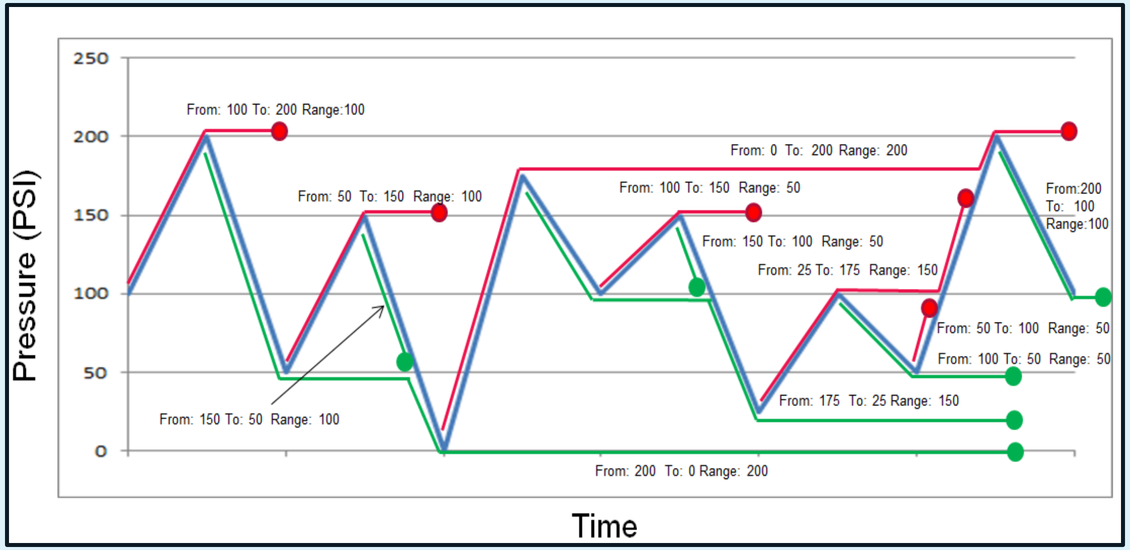


Fig 14B.3 in API 579-1 (Fitness For Service)

Pressure Cycle Rainflow Counting



Pressure Cycle Rainflow Counting



Cycle Event	From (PSI)	To (PSI)	Range (PSI)	Cycles
1	100	200	100	0.5
2	50	150	100	0.5
3	0	200	200	0.5
4	100	150	50	0.5
5	25	175	150	0.5
6	50	100	50	0.5
7	200	0	200	0.5
8	150	50	100	0.5
9	175	25	150	0.5
10	150	100	50	0.5
11	100	50	50	0.5
12	200	100	100	0.5

Cumulative Damage

Unitless ratio of the number of cycles completed in a stress range to the max number of cycles allowed per stress range

Miner's Rule

$$D = \sum \frac{n_i}{N_i}$$

d - *Damage to pipe from a stress cycling range*

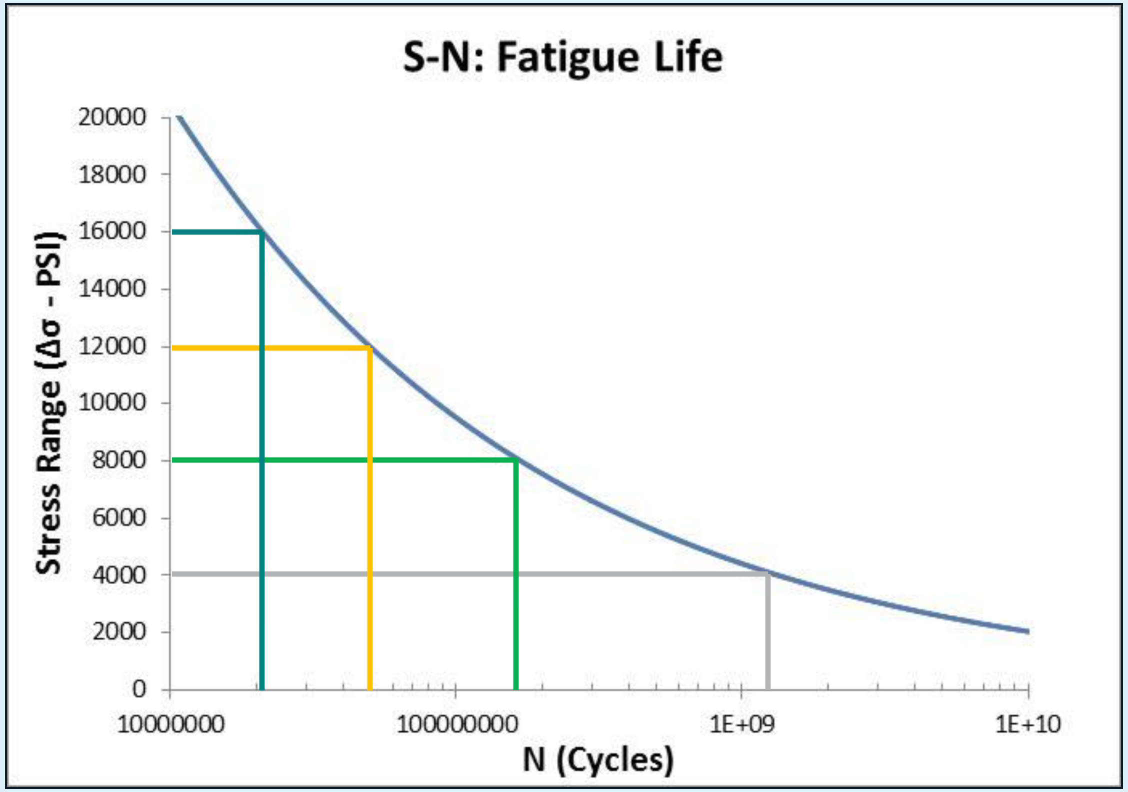
n – *Number of cycles completed per stress range*

N – *Max number of cycles per stress range*

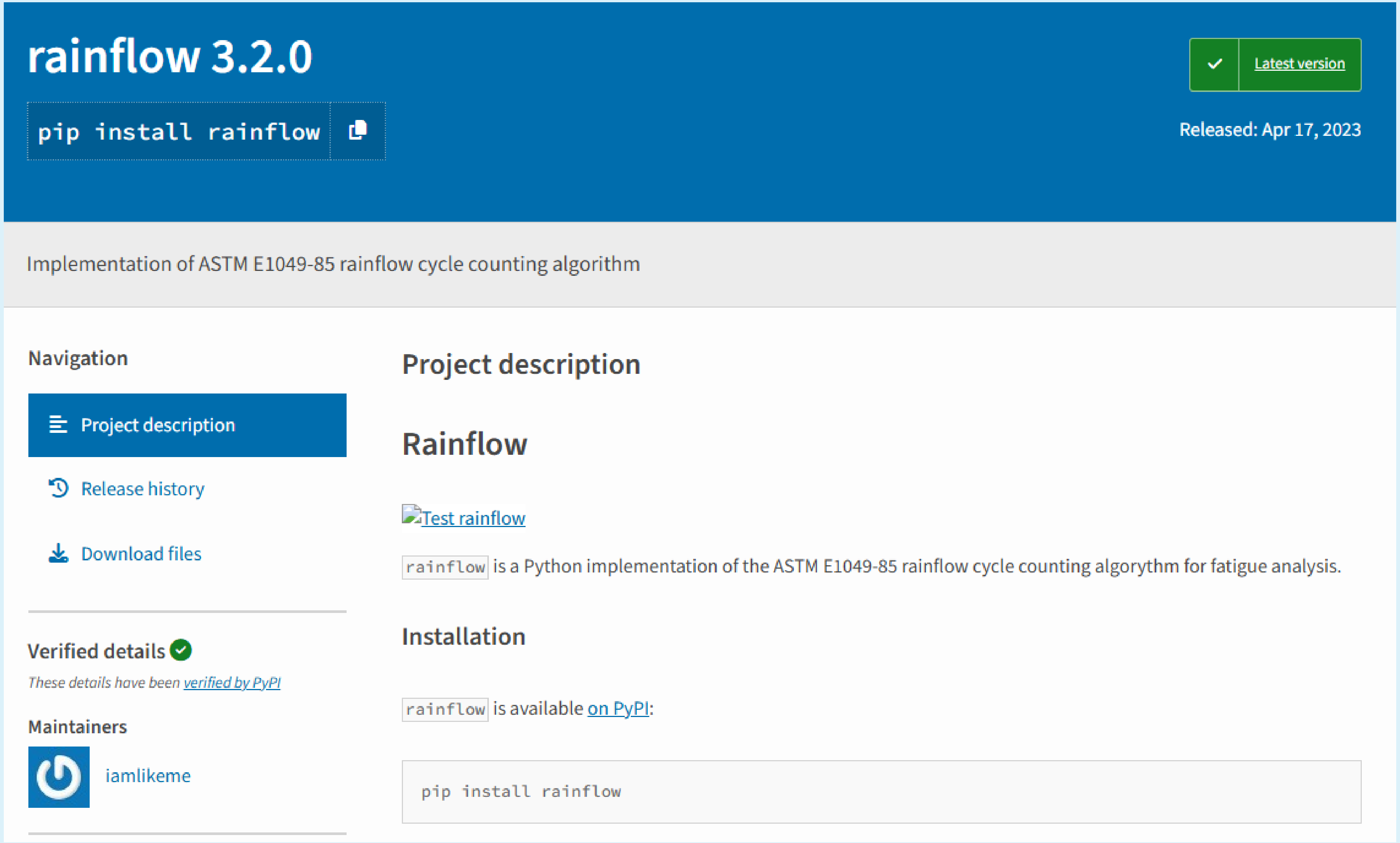
Cumulative Damage

- Example: O.D. – 40.00” / W.T. - 0.250”

Internal Pressure Range (PSI)	Total Cycles (n)	Stress Range (Δσ - PSI)	Max Cycles (N)	Damage (ni/Ni)
50	2	4000	1.34E+9	1.5E+-9
100	2	8000	1.68E+8	1.19E+-8
150	1	12000	4.98E+7	2.01E+-8
200	1	16000	2.10E+7	4.79E+-8



How Long Is It Going to Take to Code This?



rainflow 3.2.0

```
pip install rainflow
```

Released: Apr 17, 2023

Implementation of ASTM E1049-85 rainflow cycle counting algorithm

Navigation

Verified details ✓

These details have been verified by PyPI

Maintainers

iamlikeme

Project description

Rainflow

Test rainflow

`rainflow` is a Python implementation of the ASTM E1049-85 rainflow cycle counting algorythm for fatigue analysis.

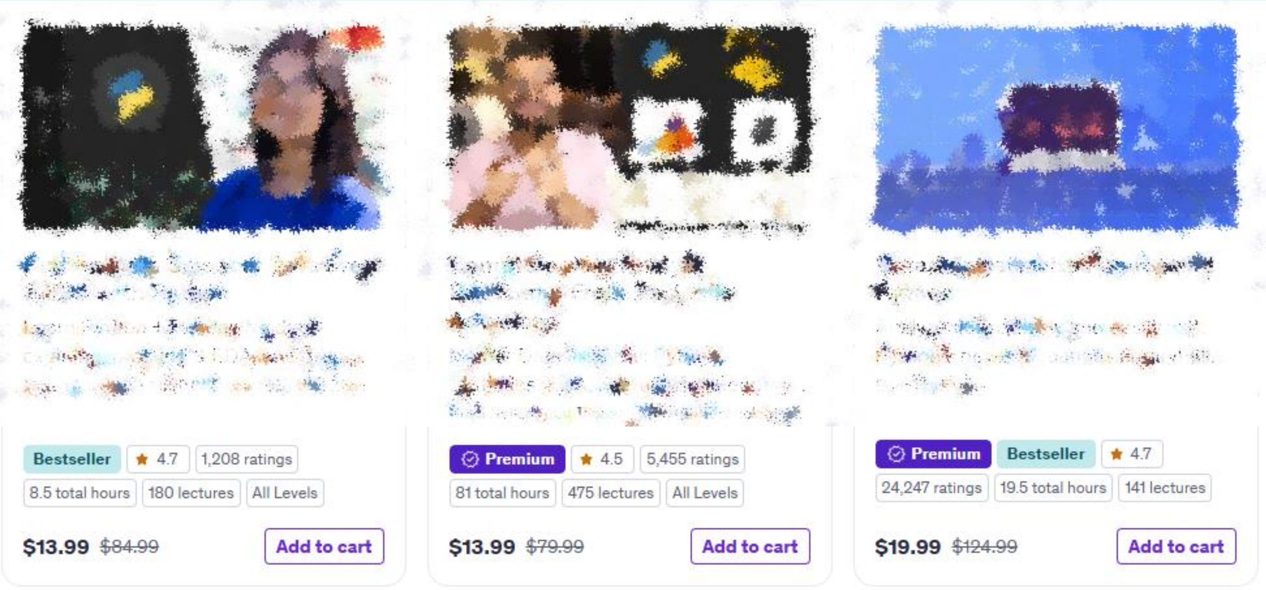
Installation

`rainflow` is available on PyPI:

```
pip install rainflow
```

Python Starter Pack (While We’re Talking About It…)

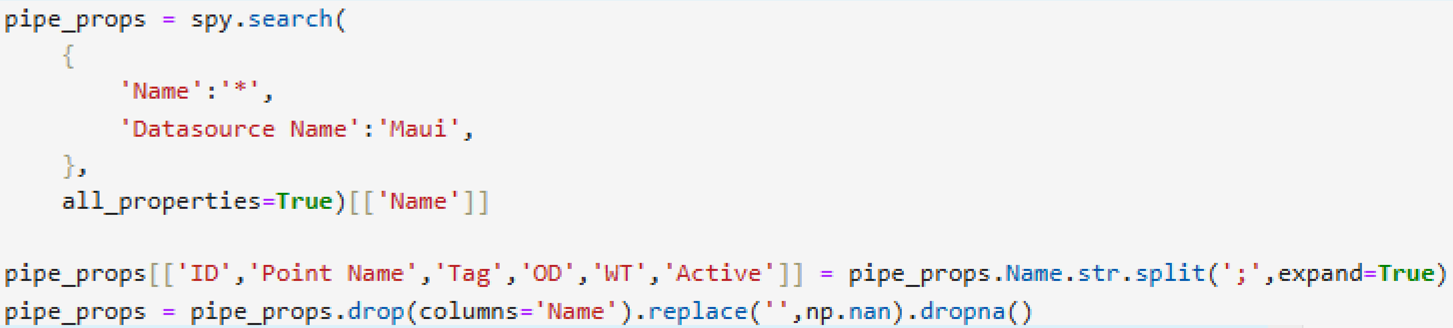
1. Open source – go get it!
2. Or just do everything in Seeq DATA LAB
3. Search ‘Exploratory Data Analysis in Python’
4. Take some cheap courses
5. Believe in yourself


Bestseller ★ 4.7 1,208 ratings
8.5 total hours 180 lectures All Levels
$13.99 ~~$84.99~~ Add to cart

Premium ★ 4.5 5,455 ratings
81 total hours 475 lectures All Levels
$13.99 ~~$79.99~~ Add to cart

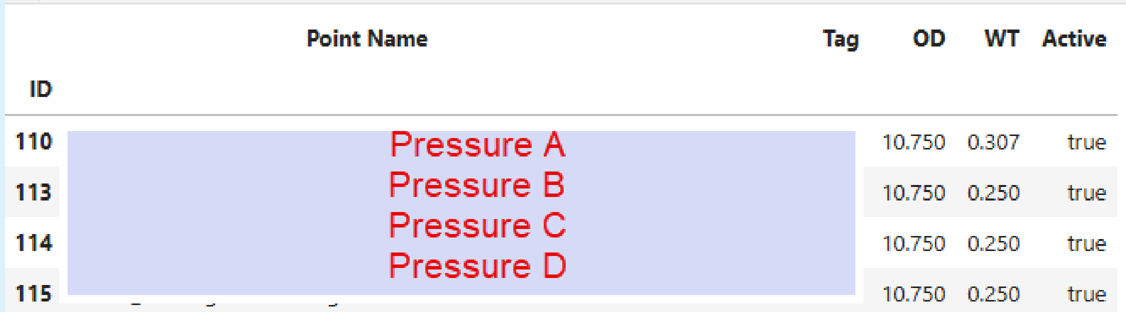
Premium Bestseller ★ 4.7
24,247 ratings 19.5 total hours 141 lectures
$19.99 ~~$124.99~~ Add to cart

Under the Hood

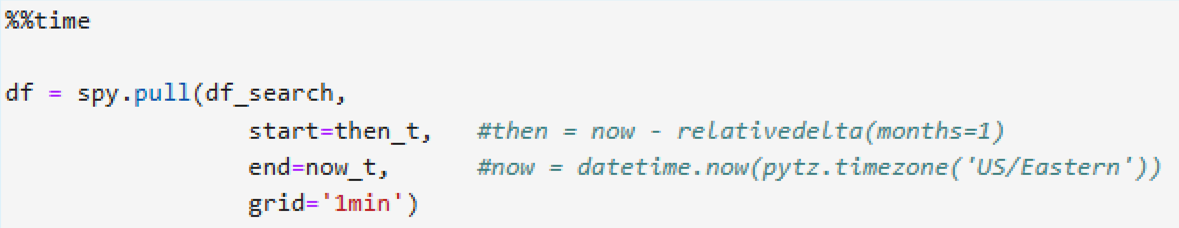


```
pipe_props = spy.search(
    {
        'Name':'*',
        'Datasource Name':'Maui',
    },
    all_properties=True)[['Name']]

pipe_props[['ID','Point Name','Tag','OD','WT','Active']] = pipe_props.Name.str.split(';',expand=True)
pipe_props = pipe_props.drop(columns='Name').replace('',np.nan).dropna()
```

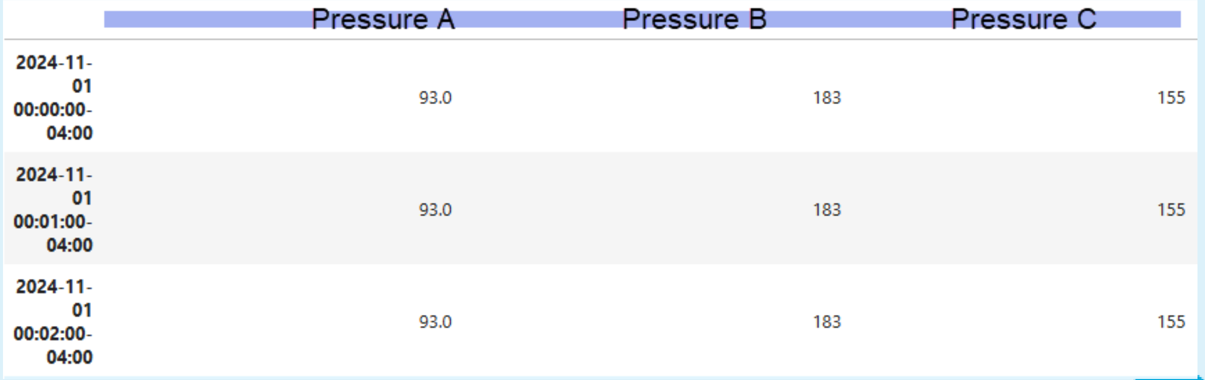


	Point Name	Tag	OD	WT	Active
ID					
110	Pressure A		10.750	0.307	true
113	Pressure B		10.750	0.250	true
114	Pressure C		10.750	0.250	true
115	Pressure D		10.750	0.250	true

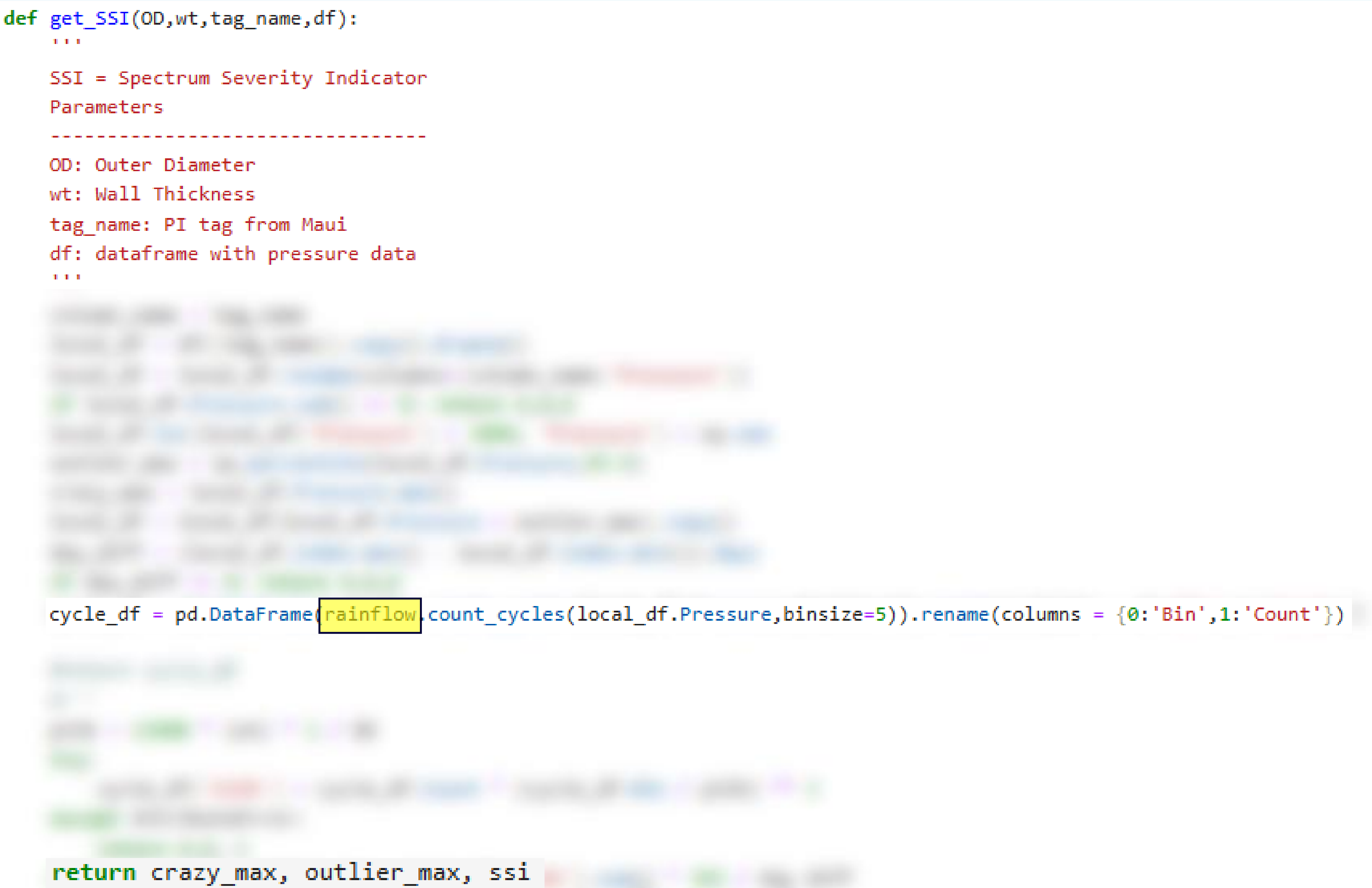


```
%%time

df = spy.pull(df_search,
              start=then_t,    #then = now - relativedelta(months=1)
              end=now_t,       #now = datetime.now(pytz.timezone('US/Eastern'))
              grid='1min')
```

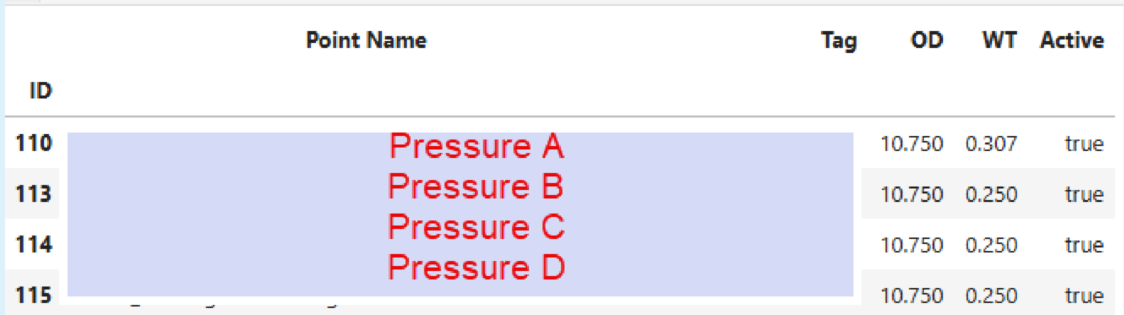


	Pressure A	Pressure B	Pressure C
2024-11-01 00:00:00-04:00	93.0	183	155
2024-11-01 00:01:00-04:00	93.0	183	155
2024-11-01 00:02:00-04:00	93.0	183	155

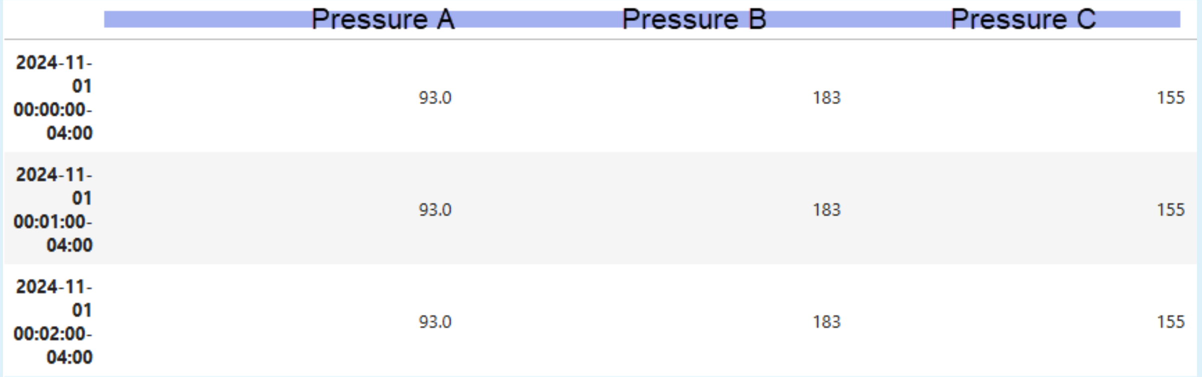


```
def get_SSI(OD,wt,tag_name,df):
    '''
    SSI = Spectrum Severity Indicator
    Parameters
    ---------------------------------
    OD: Outer Diameter
    wt: Wall Thickness
    tag_name: PI tag from Maui
    df: dataframe with pressure data
    '''
    [illegible]
    cycle_df = pd.DataFrame(rainflow.count_cycles(local_df.Pressure,binsize=5)).rename(columns = {0:'Bin',1:'Count'})
    [illegible]
    return crazy_max, outlier_max, ssi
```

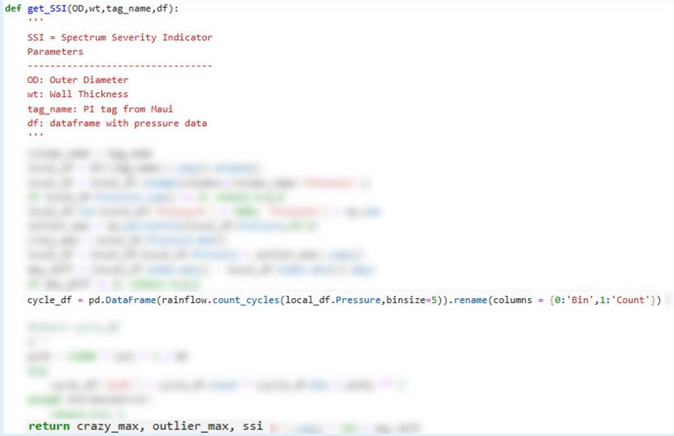
Under the Hood



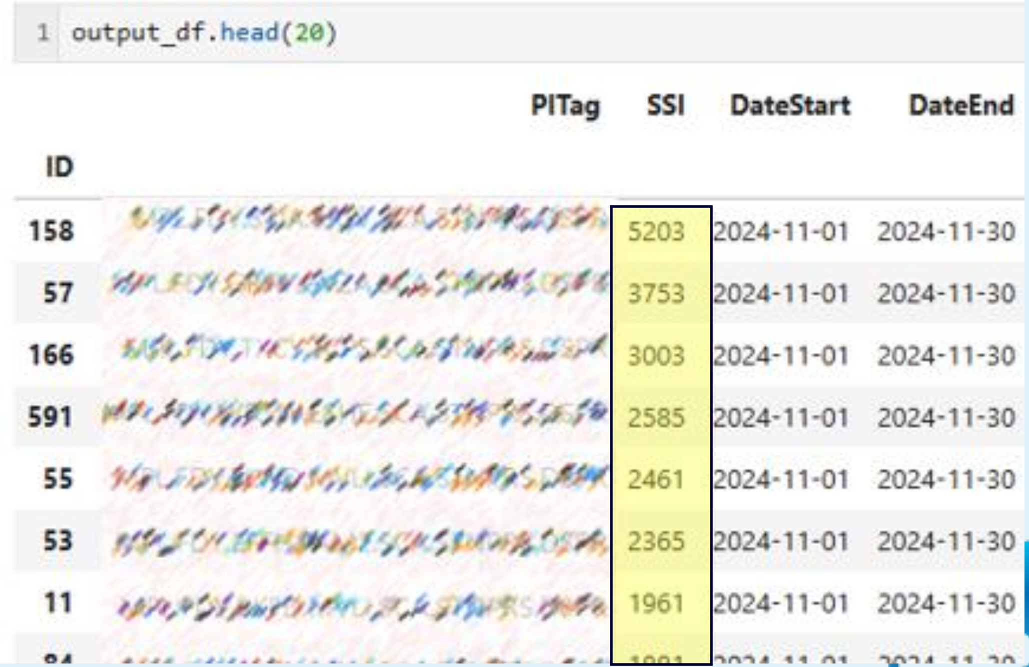
ID	Point Name	Tag	OD	WT	Active
110	Pressure A		10.750	0.307	true
113	Pressure B		10.750	0.250	true
114	Pressure C		10.750	0.250	true
115	Pressure D		10.750	0.250	true



	Pressure A	Pressure B	Pressure C
2024-11-01 00:00:00-04:00	93.0	183	155
2024-11-01 00:01:00-04:00	93.0	183	155
2024-11-01 00:02:00-04:00	93.0	183	155



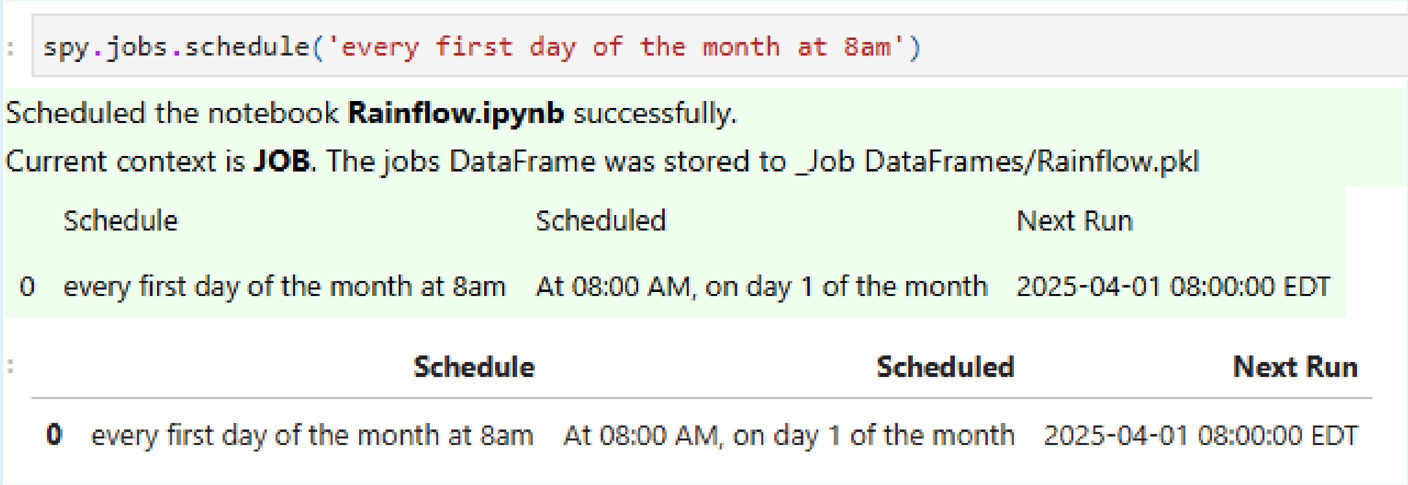
```python
def get_SSI(OD,wt,tag_name,df):
    '''
    SSI = Spectrum Severity Indicator
    Parameters
    --------------------------------
    OD: Outer Diameter
    wt: Wall Thickness
    tag_name: PI tag from Maui
    df: dataframe with pressure data
    '''
    [redacted]
    cycle_df = pd.DataFrame(rainflow.count_cycles(local_df.Pressure,binsize=5)).rename(columns = {0:'Bin',1:'Count'})
    [redacted]
    return crazy_max, outlier_max, ssi [redacted]
```



```python
output_df.head(20)
```

ID	PITag	SSI	DateStart	DateEnd
158	[redacted]	5203	2024-11-01	2024-11-30
57	[redacted]	3753	2024-11-01	2024-11-30
166	[redacted]	3003	2024-11-01	2024-11-30
591	[redacted]	2585	2024-11-01	2024-11-30
55	[redacted]	2461	2024-11-01	2024-11-30
53	[redacted]	2365	2024-11-01	2024-11-30
11	[redacted]	1961	2024-11-01	2024-11-30
84	[redacted]	[illegible]	2024-11-01	2024-11-30

Scheduling Jobs… and Mailing the Results



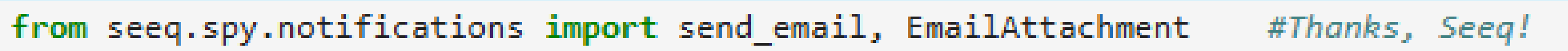
```
spy.jobs.schedule('every first day of the month at 8am')
```

Scheduled the notebook **Rainflow.ipynb** successfully.

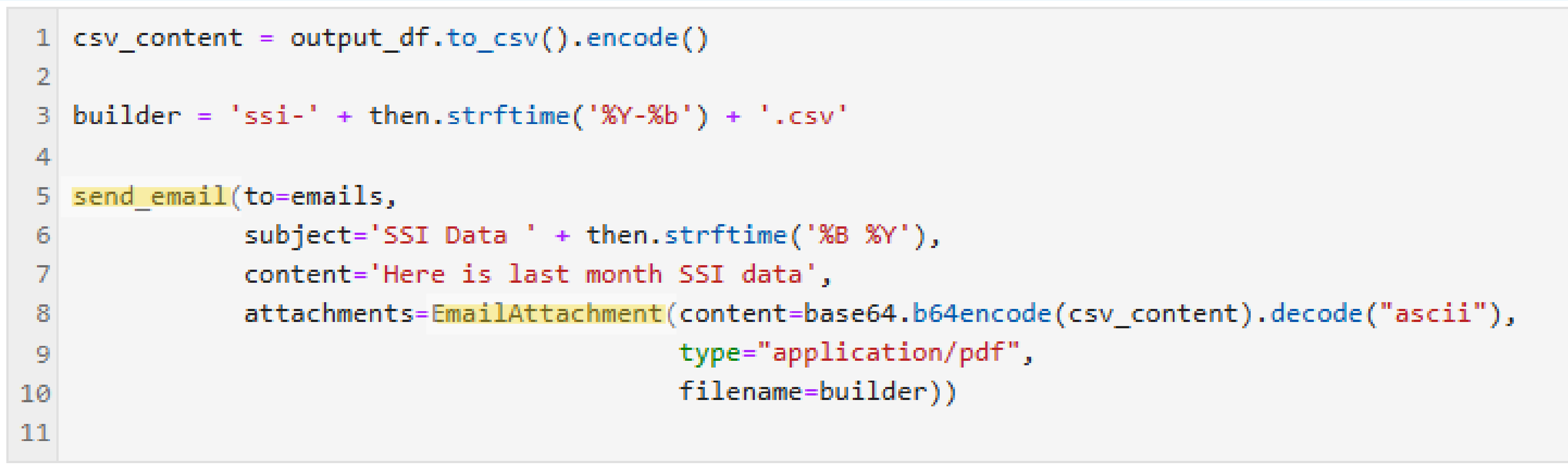
Current context is **JOB**. The jobs DataFrame was stored to _Job DataFrames/Rainflow.pkl

	Schedule	Scheduled	Next Run
0	every first day of the month at 8am	At 08:00 AM, on day 1 of the month	2025-04-01 08:00:00 EDT

	Schedule	Scheduled	Next Run
0	every first day of the month at 8am	At 08:00 AM, on day 1 of the month	2025-04-01 08:00:00 EDT



```
from seeq.spy.notifications import send_email, EmailAttachment    #Thanks, Seeq!
```



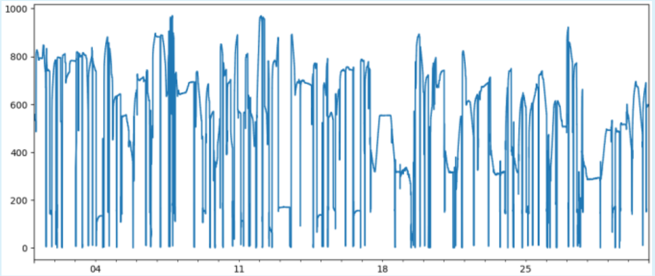
```
csv_content = output_df.to_csv().encode()

builder = 'ssi-' + then.strftime('%Y-%b') + '.csv'

send_email(to=emails,
           subject='SSI Data ' + then.strftime('%B %Y'),
           content='Here is last month SSI data',
           attachments=EmailAttachment(content=base64.b64encode(csv_content).decode("ascii"),
                                       type="application/pdf",
                                       filename=builder))

```

The Bottom Line



CHALLENGE

Calculate pressure cycling results as painlessly as possible to maximize time on analysis

SOLUTION

A one-stop shop in Data Lab!

Seeq DATA LAB

I can help you do magic with
Seeq Data Lab and Python code

RESULTS

- A week of work is done in 60 seconds
- No need for third party software
- No more lost months due to #VALUE errors
- Quick upload of results to server
- Approximately $50k per year

1010
1010

conneqt

Seeq®

conneqt

Questions?

Seeq®

conneqt

Thank You