

Convert Metocean data into a SACS seastate input file

C Kunte

November 8, 2022

Contents

1	Scope	2
1.1	Limitations and workarounds	3
1.2	Requirements	3
A	Scripts	4
A.1	Script to convert formatted csv file into a SACS seastate input file	4
A.2	Script for formatting csv file	6
B	Metocean data file example	7
C	Sample output	8

1 Scope

The stand-alone script ([Appendix A](#)) converts Metocean data ([Appendix B](#)), into a readily usable SACS seastate input file ([Appendix C](#)).

The script is written in python scripting language, and it requires some user inputs. These can be keyed-in by editing the block of data within BEGIN USER INPUTS and END USER INPUTS in the script file `slc.py`, set pre-determined values (or defaults) as appropriate, and provide a comma separated data file (csv) from Metocean at the command line. The SACS seastate input file may be generated in two steps.

Generate a formatted csv file using the command. If the given Metocean data csv file name is, say, `test.csv`, then the formatted csv file name becomes `Ftest.csv`.

```
python3 fdf.py -f <csv file>
```

Then, generate a seastate file using the following command:

```
python3 slc.py -f <formatted csv file> > seastate.inp
```

```
$ python3 fdf.py --help
Format CSV file with Pandas
fdf.py 2022 ckunte

Usage: fdf.py (-f <file>)
      fdf.py --help
      fdf.py --version

Options:
  -h, --help    Show this help
  -f --file     Specify CSV input file to format (required)
```

```
$ python3 slc.py --help
Generate SACS storm load cards from a CSV file
slc.py 2022 ckunte
Tested for python v3.8.10, v3.10.8 w/ pandas >= v1.5.1

Usage: slc.py (-f <file>)
      slc.py --help
      slc.py --version

Options:
  -h, --help    Show this help
  -f --file     Specify CSV input file (required)
```

1.1 Limitations and workarounds

The limitations of the script are (a) only the system memory allocated for the script, and (b) SACS's four character field for load cases. In other words for load cases greater than 9999, the load case numbering will default back to 0000. This can be overcome by splitting the data file to be less than 9999 load cases.

1.2 Requirements

The script requires the following and can be run at command line interface with the following installed and available:

1. python v3 scripting language,
2. pandas data analysis library,¹
3. docopt command-line interface description language.² .

¹Install using the command: `python3 -m pip install pandas` . The numpy is auto-installed through dependency management.

²Install using command: `python3 -m pip install docopt`

A Scripts

A.1 Script to convert formatted csv file into a SACS seastate input file

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  """Generate SACS storm load cards from a CSV file
5  slc.py 2022 ckunte
6
7  Tested for python v3.8.10, v3.10.8 with pandas >= v1.5.1
8
9  Usage: slc.py (-f <file>)
10         slc.py --help
11         slc.py --version
12
13  Options:
14     -h, --help    Show this help
15     -f --file     Specify CSV input file (required)
16
17  """
18  import pandas as pd
19  from docopt import docopt
20
21
22  def main(*args):
23      print("# Reading " + datfile + " file...", end="")
24      df = pd.read_csv("./" + datfile)
25      print("done.")
26      print("FILE B")
27      for i in range(len(df)):
28          # PRINTING WAVE INPUT LINES
29          print(f"LOADCN{i+1:4}")
30          print(f"LOADLB{i+1:4}Envir for pile storm analysis")
31          print(W[0])
32          print(
33              f"{W[0]:4}" # col 1-4, line label
34              + f"{W[1]:4}" # col 5-8, kinematics fac.
35              + f"{W[2]:4}" # col 9-12, wave type
36              + f"{df.iat[i, 0]:>6}" # col 13-18, wave height
37              + f"{F[0]:>6}" # col 19-24, SWL, skip (from LDOPT)
38              + f"{df.iat[i, 1]:>6}" # col 25-30, wave period
39              + f"{F[0]:>8}" # col 31-38, wave length, skip if period is given
40              + f"{df.iat[i, 2]:>6}" # col 39-44, wave angle
41              + f"{F[0]:>6}" # col 45-50, mud line elev., skip (from LDOPT)
42              + f"{W[3]:>0}" # col 51, input mode
43              + f"{W[4]:>7}" # col 52-58, crest position
44              + f"{W[5]:>6}" # col 59-64, step size
45              + f"{F[0]:1}" # col 65-66, steps for dyn. analysis, skip
46              + f"{W[6]:1}" # col 67-68, static steps
47              + f"{W[7]:1}" # col 69-70, critical position
48              + f"{W[8]:1}" # col 71-72, member seg. (max)
49              + f"{W[9]:1}" # col 73-74, member seg. (min)
50              + f"{0:0}".format(F[0]) # col 75, local accel. only, skip
51              + f"{0:0}".format(F[0]) # col 76, print opt, skip
52              + f"{0:<1}".format(F[0]) # col 77-78, order of stream func., skip
53          )
54      # PRINTING CURRENT INPUT LINES
55      print(C[0])
56      print(
57          f"{C[0]:4}" # col 1-4, line label
58          + f"{F[0]:>4}" # col 5-8, min inline curr velocity, skip
59          + f"{eam[9]:>8}" # col 9-16, elev above mud line
60          + f"{df.iat[i, 12]:>8}" # col 17-24, curr velocity
61          + f"{df.iat[i, 2]:>8}" # col 25-32, curr dir
62          + f"{F[0]:>8}" # col 33-40, mudline elev override, skip
63          + f"{F[0]:>8}" # col 41-48, blocking factor, skip
64          + f"{F[0]:>8}" # col 49-56, elev, skip
```

```

65         + f"{C[1]:1}" # col 57-58, elev, generate blocking fac.
66         + f"{F[0]:>0}" # col 59, null
67         + f"{C[2]:1}" # col 60-61, crest stretching opt.
68         + f"{F[0]:>0}" # col 62, null
69         + f"{F[0]:2}" # col 63-65, velocity units opt., skip
70         + f"{F[0]:>0}" # col 66, null
71         + f"{F[0]:2}" # col 67-69, elev percent opt., skip
72         + f"{F[0]:>3}" # col 70, null (for now this is a workaround)
73         + f"{C[3]:>2}" # col 71-73, AWP opt.
74     )
75     # adjust ranges depending upon the current profile
76     for n, m in zip(range(8, -1, -1), range(11, 2, -1)):
77         print(
78             f"{C[0]}" # col 1-4, line label
79             + f"{F[0]:>4}" # col 5-8, min inline curr velocity, skip
80             + f"{eam[n]:>8}" # col 9-16, elev above mud line
81             + f"{df.iat[i, m]:>8}" # col 17-24, curr velocity
82             + f"{df.iat[i, 2]:>8}" # col 25-32, curr dir
83         )
84     pass
85
86
87 if __name__ == "__main__":
88     args = docopt(
89         __doc__, version="Generate SACS storm load cards from a CSV file, v0.1"
90     )
91     datfile = "%s" % (args["<file>"])
92     #
93     # -- BEGIN USER INPUTS --
94     #
95     # WAVE DEFINITION AND POSITION PARAMETERS (SACS SEASTATE MANUAL, PG 170)
96     #
97     W = [
98         "WAVE", # line label
99         0.95, # kinematics factor
100        "STOK", # wave type
101        "D", # input mode (length (L), degree (D), or time (T))
102        -90.0, # crest position -- wave
103        4.00, # step size -- wave
104        " 90", # static steps -- wave
105        "MM", # critical position -- wave
106        "10", # member segmentation (max)
107        " 1", # member segmentation (min)
108    ]
109    # CURRENT PARAMETERS (SACS SEASTATE MANUAL, PG 171)
110    #
111    C = [
112        "CURR", # line label
113        "BC", # option to generate blocking factor
114        "NL", # crest stretching option
115        "AWP", # apparent wave period option
116    ]
117    # ELEVATION ABOVE MUDLINE (FOR CURRENT PROFILE)
118    #
119    eam = [
120        166.18,
121        151.18,
122        141.18,
123        121.18,
124        101.18,
125        81.18,
126        61.18,
127        41.18,
128        21.18,
129        1.18,
130    ]
131    # FILLER FOR EMPTY (OR NULL) COLUMN BLOCKS
132    #
133    F = [" "]

```

```

134 #
135 # CSV DATA FILE FROM METOCEAN TO USE
136 #
137 # Headers in CSV file:
138 # H (m), T(s), ThetaP PlatformNth(Deg), WS (m/s), CS5(m/s), CS20(m/s), CS30
(m/s), CS50(m/s), CS70(m/s), CS90
(m/s), CS110(m/s), CS130(m/s), CS150
(m/s), CS170(m/s)
139
140 # -- END USER INPUTS --
141
142 main(datfile, W, F, C)

```

A.2 Script for formatting csv file

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  """Format CSV file with Pandas
5  fdf.py 2022 ckunte
6
7  Usage: fdf.py (-f <file>)
8          fdf.py --help
9          fdf.py --version
10
11  Options:
12  -h, --help  Show this help
13  -f --file   Specify CSV input file to format (required)
14
15  """
16  import numpy as np
17  import pandas as pd
18  from docopt import docopt
19
20
21  def main(*args):
22      print("# Reading " + datfile + " file...", end="")
23      df = pd.read_csv("./" + datfile)
24      print("done.")
25      # remove wind speed column from data (by index -- this is a workaround:
26      # [should be [3], but somehow [2] works -- possibly a python 3.8.10 bug)
27      df2 = df.drop(df.columns[[2]], axis=1)
28      return df2.to_csv("F" + datfile)
29
30
31  if __name__ == "__main__":
32      args = docopt(
33          __doc__, version="Generate SACS storm load cards from a CSV file, v0.1"
34      )
35      datfile = "%s" % (args["<file>"])
36      main(datfile)
37      print("Formatted file:", "F" + datfile)

```

B Metocean data file example

1	H (m), T(s), ThetaP PlatformNth(Deg), WS (m/s), CS5(m/s), CS20(m/s), CS30(m/s), CS50(m/s), CS70(m/s), CS90(m/s), CS110(m/s),
	CS130(m/s), CS150(m/s), CS170(m/s)
2	4.48, 14.56,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
3	4.81, 14.67,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
4	4.40, 14.21,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
5	4.18, 12.34,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
6	2.83, 8.32,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
7	3.76, 14.89,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
8	6.07, 14.04,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
9	4.13, 15.90,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
10	4.76, 13.01,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
11	3.94, 13.43,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
12	2.26, 8.39,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
13	1.66, 12.44,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
14	6.48, 11.75,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
15	6.07, 12.40,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
16	3.64, 9.72,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
17	4.74, 8.99,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
18	7.18, 12.71,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
19	5.18, 15.47,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
20	4.33, 14.93,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
21	2.18, 14.52,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
22	3.64, 13.86,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
23	4.83, 12.67,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
24	3.14, 11.35,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
25	0.78, 5.32,290.00, 13.84, 0.75, 0.68, 0.60, 0.50, 0.44, 0.41, 0.35, 0.30, 0.25, 0.15,
...	

C Sample output

```

1  # Reading FTS001.000040TS.csv file...done.
2  FILE B
3  LOADCN 1
4  LOADLB 1Envir for pile storm analysis
5  WAVE
6  WAVE0.95STOK 4.48 14.56 290.0 D -90.0 4.0 90MM10 1
7  CURR
8  CURR 1.18 0.15 290.0 BC NL AWP
9  CURR 21.18 0.25 290.0
10 CURR 41.18 0.3 290.0
11 CURR 61.18 0.35 290.0
12 CURR 81.18 0.41 290.0
13 CURR 101.18 0.44 290.0
14 CURR 121.18 0.5 290.0
15 CURR 141.18 0.6 290.0
16 CURR 151.18 0.68 290.0
17 CURR 166.18 0.75 290.0
18 LOADCN 2
19 LOADLB 2Envir for pile storm analysis
20 WAVE
21 WAVE0.95STOK 4.81 14.67 290.0 D -90.0 4.0 90MM10 1
22 CURR
23 CURR 1.18 0.15 290.0 BC NL AWP
24 CURR 21.18 0.25 290.0
25 CURR 41.18 0.3 290.0
26 CURR 61.18 0.35 290.0
27 CURR 81.18 0.41 290.0
28 CURR 101.18 0.44 290.0
29 CURR 121.18 0.5 290.0
30 CURR 141.18 0.6 290.0
31 CURR 151.18 0.68 290.0
32 CURR 166.18 0.75 290.0

...

118937 CURR 166.18 0.83 185.0
118938 LOADCN7930
118939 LOADLB7930Envir for pile storm analysis
118940 WAVE
118941 WAVE0.95STOK 5.14 12.48 185.0 D -90.0 4.0 90MM10 1
118942 CURR
118943 CURR 1.18 0.17 185.0 BC NL AWP
118944 CURR 21.18 0.28 185.0
118945 CURR 41.18 0.33 185.0
118946 CURR 61.18 0.39 185.0
118947 CURR 81.18 0.45 185.0
118948 CURR 101.18 0.48 185.0
118949 CURR 121.18 0.55 185.0
118950 CURR 141.18 0.67 185.0
118951 CURR 151.18 0.75 185.0
118952 CURR 166.18 0.83 185.0
118953 LOADCN7931
118954 LOADLB7931Envir for pile storm analysis
118955 WAVE
118956 WAVE0.95STOK 6.63 11.36 185.0 D -90.0 4.0 90MM10 1
118957 CURR
118958 CURR 1.18 0.17 185.0 BC NL AWP
118959 CURR 21.18 0.28 185.0
118960 CURR 41.18 0.33 185.0
118961 CURR 61.18 0.39 185.0
118962 CURR 81.18 0.45 185.0
118963 CURR 101.18 0.48 185.0
118964 CURR 121.18 0.55 185.0
118965 CURR 141.18 0.67 185.0
118966 CURR 151.18 0.75 185.0
118967 CURR 166.18 0.83 185.0

```