IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE)
e-ISSN: 2278-1684,p-ISSN: 2320-334X, Volume 19, Issue 6 Ser. III (Nov. – Dec. 2022), PP 01-07
www.iosrjournals.org

Implementation of the algorithm of the TUC method for fatigue cycle counting

Manuel López Godínez[1], Samuel Alcántara Montes[2], Marco A. Gutiérrez Villegas[3], Esiquio Martín Gutiérrez Armenta[4]

[1,2](Mechanical Engineering, SEPI ESIME Zacatenco/Insituto Politécnico Nacional, México)
[3,4](Mechanical Engineering, Unidad Azcapotzalco/ Universidad Autónoma Metropolitana, México)

Abstract: *The Rainflow fatigue counting algorithm was formalized by Rychlik through the counting method called Top Level-up cycle (TUC). In this work, the programming of the TUC is proposed, complementing it with the Rainflow Matrix, where the cycles of greater amplitude are summarized, those that cause greater damage to the material.*
Background*: The material life-time assessment of any structural body in service has been the main purposes of the fatigue of materials. The Palmgrem-Miner is a well known model that estimates the material damage given the Stress-Time data. The aforementioned model depends on the failing number of cycles obtained in laboratory tests to build the S-N diagrams used for the cycle counting necessary for this model. The Rainflow cycle counting method is used and it was mathematically formalized by Rychlik, who proposed a new cycle counting definition which we implemented in R language programming.*
Materials and Methods*: We have used random Stress-Time data against laboratory Stress-Time data, because the main purpose of the Rainflow cycle counting is the classification of the stresses amplitudes found along the time series Stress-Time, however the TUC method also is used to match the results between both counting cycle methods. Furthermore, the Rainflow Matrix was implemented to visualize the greatest and lesser amplitudes.*
Key Word*: Rainflow, TUC, Rainflow Matrix.*

Date of Submission: 10-12-2022 Date of Acceptance: 24-12-2022

I. Introduction

The evaluation of the remaining useful life of a structure in service (residual life) has been one of the main purposes of material fatigue. One of the best known models to estimate the accumulated damage caused by the initiation of the crack suffered by the material due to fatigue is the Palmgren-Miner rule in which one of its parameters involves the number of cycles N that the material undergoes at the failure given a certain stress value. The stress cycle is a concept that depends on the cycle counting method used[1]. To find N, it is required to classify the cycles by fatigue through various cycle counting algorithms, the most used is the Rainflow algorithm[2]; however, it had not been given mathematical meaning until Rychlik proposed it and developed an equivalent algorithm called the Top-level Up Cycle (TUC)[3]. Its definition starts from the fact that the Stress-Time history takes it from zero but in this work it is considered different from zero so that it takes any window of the history, likewise the Rainflow Matrix algorithm was adapted to graphically visualize the loads of greater and lesser amplitude.

II. Rychlik's formal definition of the Rainflow algorithm

Originally, the formal Rychlik Rainflow algorithm definition for cycle counting[3,4], givenany Stress-Time history window, is taken from the origin to a certain time T, that is, in the interval $[0, T^+]$, but in this work the window is made variable by modifying the Rychlik algorithm so that the interval becomes $[T^-, T^+]$.

Redefinition of Rychlik's algorithm

Let f be a continuous function of time t, in the interval$t \in [T^-, T^+]$. Let $f_{max}(t)$be a local maximum at $t \in [T^-, T^+]$with the times t^- and t^+ defined in equations (1) and (2).

$$t^- = \begin{cases} sup\{s \in [T^-, t) & \text{if} \quad f(s) > f_{max}(t)\} \\ T^- & \text{if} \quad f(s) \le f_{max}(t) \quad \forall t \in [T^-, t) \\ \nexists & if \quad T^- = t \end{cases} \tag{1}$$

$$t^+ = \begin{cases} inf\{s \in (\mathrm{t}, T^+] \quad \text{if} \quad f(s) \geq f_{max}(t)\} \\ \quad T^+ \qquad\qquad \text{if} \quad f(s) < f_{max}(t) \quad \forall t \in (t, T^+] \\ \quad \nexists \qquad\qquad\quad \text{if} \quad T^+ = t \end{cases} \tag{2}$$

Let m_t^- and m_t^+ be the points in the intervals $[t^-, t)$ and $(\mathrm{t},\ \mathrm{t}^+]$ defined in equation (3).

$$\begin{aligned} m_t^-(s) &= \min\{f(s); t^- < s < t\} \\ m_t^+(s) &= \min\{f(s); t < s < t^+\} \end{aligned} \tag{3}$$

The variable $m_t(s)$ is defined in equation (4).

$$m_t(s) = \begin{cases} \text{máx}\{m_t^-, m_t^+\} \quad \text{If} \quad t^+ < T^+ \text{ó } f(t) = f(T^+) \\ \quad m_t^+ \qquad\qquad\qquad otherwise \end{cases} \tag{4}$$

TUC Method

In Rychlik's TUC method, it formalizes the amplitudes $H^-(t)$and$H^+(t)$ according to equation (5).

$$\begin{aligned} H^-(t) &= f_{max}(t) - \min\{f(s); t^- s < t\} \\ H^+(t) &= f_{max}(t) - \min\{f(s); t < s < t^+\} \end{aligned} \tag{5}$$

From equations (3) and (5) equation (6) is obtained:

$$\begin{aligned} m_t^-(s) &= \min\{f(s); t^- s < t\} \\ m_t^+(s) &= \min\{f(s); t < s < t^+\} \end{aligned} \tag{6}$$

Hence, the amplitudes can be rewritten as follows in equation (7):

$$\begin{aligned} H^-(t) &= f_{max}(t) - m_t^-(s) \\ H^+(t) &= f_{max}(t) - m_t^+(s) \end{aligned} \tag{7}$$

Rules for cycle identification

Rychlik established the following rules to identify the full and half cycles considering the amplitudes defined in the TUC method.

Rule 1. The count of a complete cycle is established, if any of the conditions established in equation (8) are met.

If $H^-(t) \leq H^+(t)\ and\ T^- < t^-$, or, If $H^-(t) > H^+(t)\ and\ t^+ < T^+$ (8)

The amplitude of the cycle is defined in equation (9).

$$H(t) = \min\big(H^-(t), H^+(t)\big) \tag{9}$$

Using expression (4), we can rewrite the amplitude of the cycle from expression (9), as follows in equation (10).

$$H(t) = f_{max}(t) - m_t(s) \tag{10}$$

Rule 2. A half-cycle count is set if $f(t)$ is at the far right or far left of the Stress-Time $[T^-, T^+]$ history window. The left or right amplitudes are: $H^-(t)$ or$H^+(t)$ respectively.

Rule3.The counting of two half cycles with amplitudes is established: $H^-(t)$ and $H^+(t)$, in other cases that do not comply with Rules 1 and 2.

Implementation of the Rychlik method for counting cycles

The Rychlik cycle counting algorithm was programmed in **R** language. Figure 1 shows the flowchart to implement the TUC algorithm.

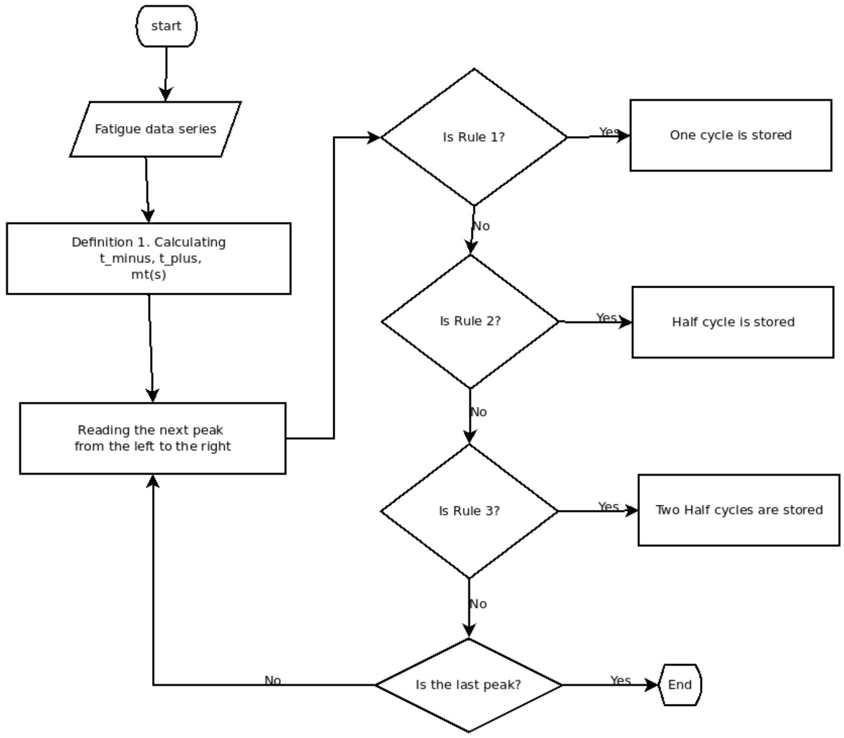


Fig. 1 Flowchart of the TUC algorithm.

In addition, the Rainflow Matrix[5] was built, in order to group those cycles which ranges are tension or compression. With Algorithm 1 the bins are obtained in such a way that they depend on the maximum amplitude of the Stress-Time history and with Algorithm 2 the Rainflow Matrix is built.

Algorithm 1. Algorithm for the creation of bins in the load time series (Stress-Time).

Input:
Loads, Effort-Time history filtered only with valleys and peaks
nBins, Number of bins

Output:
Bins, Bin Range Arrangement

```
maxCarga ← maximum value of the time load series
minCarga ← minimum value of the time load series
Maximum amplitude of the time load series
maxAmp ← maxCargas − minCargas
Bin range size
BinT am ← maxAmp/nBins
mediaBinT am ← BinT am/2
Creation of each i-th bin (iBins)
for i in 0:nBins+1 do
    Create the first bin
    if i == 1 then
        Lower bound of the first bin
        iBin[1] ← minCarga − mediaBinT am
        Upper bound of the first bin
        iBin[2] ← minCarga − mediaBinT am
        Creation of each i-th bin
    else
        Lower bound of the i-th bin
        iBin[1] ← Bins[, 2][i − 1]
        Upper bound of the i-th bin
        iBin[2] ← iBin[1] + BinT am
    Adds the i-th iBin to the Bins array
```

```
        Bins.append(iBin)
```

Algorithm 2. Cycle frequency algorithm for each bin in the Time Load series.

Input:

Bins, Bin Range Arrangement
cycles, Array of counted cycles by Rychlik's redefinition

Output: nBinsMatrix, Rainflow Matrix nBins × nBins

```
Iteration over each i-th Cycle (iCycle) to determine which bin it belongs to
for iCycle in 0:cycles.length do
        Half and full cycles are included in the cycles array.
filter out those that meet Rules 1 or 3
                if cycles[iCycle]['T ipoRegla'] == 1 or
                        cycles[iCycle]['T ipoRegla'] == 3 then
                        if cycles[iCycle]['T ipoRegla'] == 1 then
                                Setting of the Begin and End i-th cycle
                                if cycles[iCycle]['s'] < cycles[iCycle]['t'] then
                                        cycleBegin ← cycles[iCycle]['mt(s)']
                                        cycleEnd ← cycles[iCycle]['fmax(t)']
                                else
                                        cycleBegin ← cycles[iCycle]['fmax(t)']
                                        cycleEnd ← cycles[iCycle]['mt(s)']
                        if cycles[iCycle]['T ipoRegla'] == 3 then
                              The ranges of both media are compared in fmax(t) if they are equal,
                              then it is taken as a cycle for the matrix
                                if cycles[iCycle]['izquierda'] == cycles[iCycle]['derecha']
                                then
                                        cycleBegin ← cycles[iCycle]['mt(s)']
                                        cycleEnd ← cycles[iCycle]['fmax(t)']
        Obtaining the Start and End of the Bin with respect to the Start and Endof the i-the cycle
        for i in 1:nBins do
                if (Bins[i]['CotaInferior'] ≤ cycleBegin) &&
                        (cycleBegin < Bins[i]['CotaSuperior']) then
                        fromBin ← i
                        if (Bins[i]['CotaInferior'] ≤ cycleEnd) &&
                                (cycleEnd < Bins[i]['CotaSuperior']) then
                                toBin ← i
        Count the cycle frequency in the Bin for the Rainflow Array
        nBinsMatrix[fromBin, toBin] ← 1 + nBinsM atrix[fromBin, toBin]
```

Through the following example, the cycles counted between both Rychlik algorithms and the original Rainflow algorithm are compared through the Vibration software[5] programmed in Fortran.

III. Cycle Counting Example

Table no 1 sows the Stress-Time history[5]from cycle counting is performed by using both programs: TUC method and Vibration software[6].

Table no 1: Stress-Time history[5]

Time	Stress	Time	Stress
1	1	11	0.3
2	4.8	12	4.8
3	2.2	13	2.2
4	5.7	14	5.7
5	1	15	2.7
6	3.8	16	5.7

7	0.3	17	0.3
8	5.7	18	4.8
9	0.3	19	1
10	3.8		

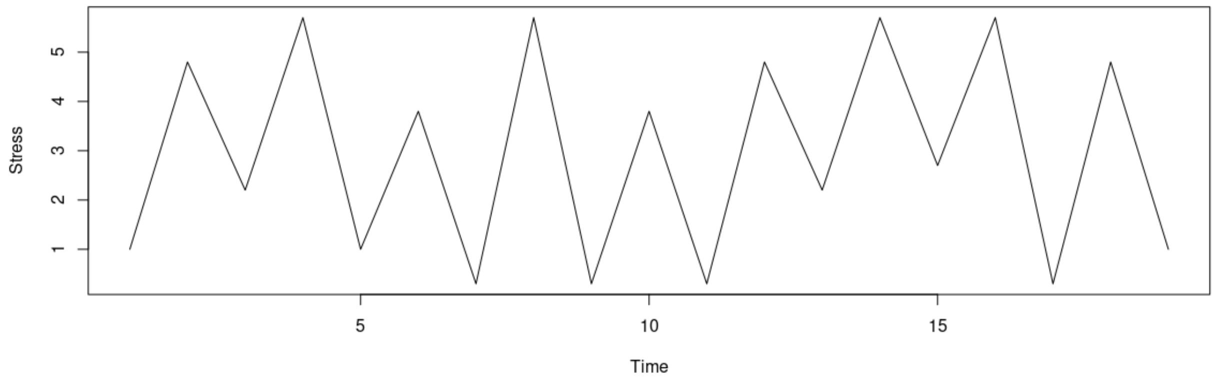


Fig. 2 Stress-Time hostory plot from Table no 1.

	mt_minus(s)	mt_plus(s)	mt(s)	s_left	Rule	t	Y(t)	t_minus	t_plus	Cycles	Range
1	"1"	"2.2"	"2.2"	"3"	"R-1"	"2"	"4.8"	"1"	"3.74285714285714"	"1 - cycle"	"2.6"
2	"1"	"0.3"	"1"	NA	"R-3"	"4"	"5.7"	"1"	"8"	"1/2 cycle"	"4.7"
3	"1"	"0.3"	"0.3"	NA	"R-3"	"4"	"5.7"	"1"	"8"	"1/2 cycle"	"5.4"
4	"1"	"0.3"	"1"	"5"	"R-1"	"6"	"3.8"	"4.40425531914894"	"7.64814814814815"	"1 - cycle"	"2.8"
5	"0.3"	"0.3"	"0.3"	NA	"R-3"	"8"	"5.7"	"1"	"14"	"1/2 cycle"	"5.4"
6	"0.3"	"0.3"	"0.3"	NA	"R-3"	"8"	"5.7"	"1"	"14"	"1/2 cycle"	"5.4"
7	"0.3"	"0.3"	"0.3"	"9"	"R-1"	"10"	"3.8"	"8.35185185185185"	"11.7777777777778"	"1 - cycle"	"3.5"
8	"0.3"	"2.2"	"2.2"	"13"	"R-1"	"12"	"4.8"	"8.16666666666667"	"13.7428571428571"	"1 - cycle"	"2.6"
9	"0.3"	"2.7"	"2.7"	"15"	"R-1"	"14"	"5.7"	"1"	"16"	"1 - cycle"	"3"
10	"0.3"	"0.3"	"0.3"	NA	"R-3"	"16"	"5.7"	"1"	"19"	"1/2 cycle"	"5.4"
11	"0.3"	"0.3"	"0.3"	NA	"R-3"	"16"	"5.7"	"1"	"19"	"1/2 cycle"	"5.4"
12	"0.3"	"1"	"0.3"	NA	"R-3"	"18"	"4.8"	"16.1666666666667"	"19"	"1/2 cycle"	"4.5"
13	"0.3"	"1"	"1"	NA	"R-3"	"18"	"4.8"	"16.1666666666667"	"19"	"1/2 cycle"	"3.8"

Fig. 3 Cycles obtained by the TUC method programmed in R.

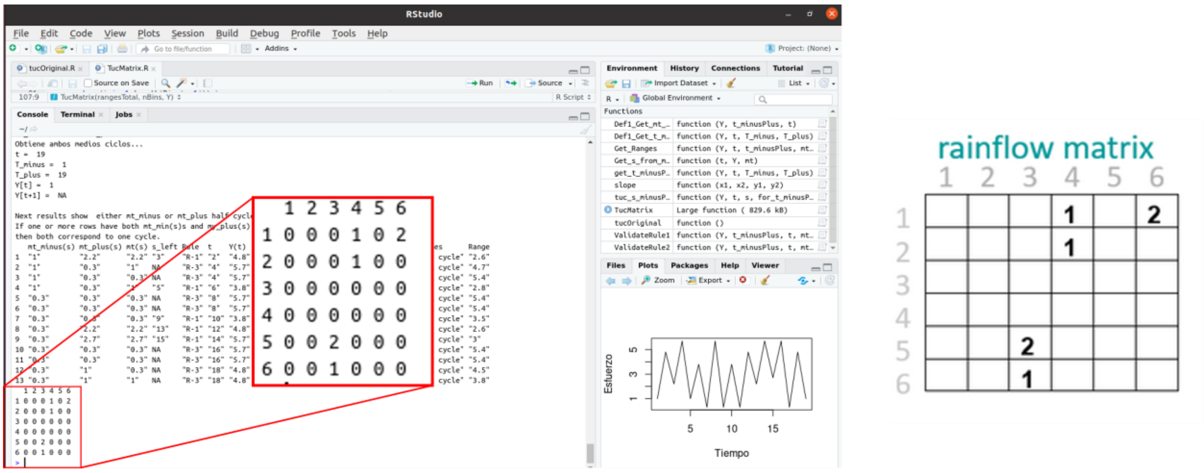
AMPLITUDE = (PEAK-VALLEY)/2

RANGE LIMITS (UNITS)	CYCLE COUNTS	AVERAGE AMP	MAX AMP	MIN MEAN	AVE MEAN	MAX MEAN	MIN VALLEY	MAX PEAK
4.860 to 5.400	2.5	2.700	2.700	3.000	3.000	3.000	0.3000	5.700
4.320 to 4.860	1.0	2.300	2.350	2.550	2.950	3.350	0.3000	5.700
3.780 to 4.320	0.5	1.900	1.900	2.900	2.900	2.900	1.000	4.800
3.240 to 3.780	1.0	1.750	1.750	2.050	2.050	2.050	0.3000	3.800
2.700 to 3.240	2.0	1.450	1.500	2.400	3.300	4.200	1.000	5.700
2.160 to 2.700	2.0	1.300	1.300	3.500	3.500	3.500	2.200	4.800
1.620 to 2.160	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1.080 to 1.620	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.8100 to 1.080	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.5400 to 0.8100	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.2700 to 0.5400	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.1350 to 0.2700	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.000 to 0.1350	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Fig. 4 Cycles obtained from Vibration software[6].

Fig. 3 and Fig. 4 show that 9 cycles are obtained; take into account that the full cycles and half cycles are not grouped by the Vibration software[6] shown in Fig. 4.

With the cycles obtained (Fig. 3), the Rainflow Matrix implemented in R with the proposed variant is obtained, validating it with respect to the matrix obtained with the Siemens software[5] (Fig. 5 (a) and (b)).



(a) (b)

Fig. 5 Rainflow Matrix. (a) Modified algorithm results, (b) Original algorithm of the Siemens Software[5].

Tensile and compressive stress regions

The Rainflow Matrix allows visualizing the tension and compression stresses that the material undergoes during the Stress-Time history. To visualize them, more data is required in the Stress-Time history[7], by generating the Rainflow Matrix and then plotting them as shown in Fig. 6 where the regions of greater and lesser damage are visualized.

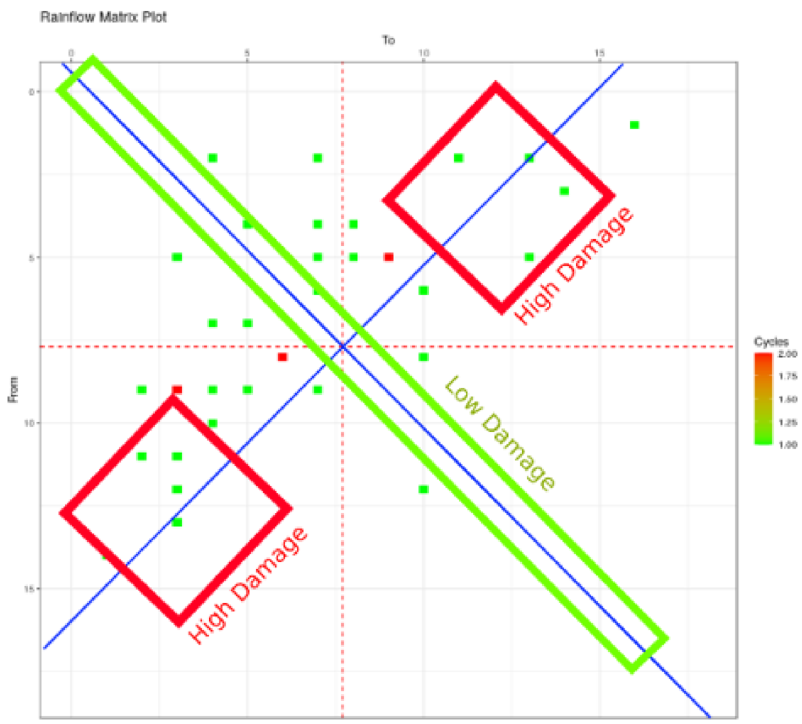


Fig. 6 Regions of greater and lesser amplitudes.

In Fig. 6, shows the graph where cycles of greater amplitude (greatest damage) and the cycles of lesser amplitude indicated by the green rectangle (less damage) are represented with red boxes.

IV. Conclusion

The TUC cycle counting method proposed by Rychlik was slightly modified to take any window in the Stress-Time history and count the cycles in that window. The counted and grouped cycles are used to obtain the Rainflow Matrix whose algorithm was slightly modified to avoid moving the peaks and valleys and leave the history unaltered. If the information is obtained with this type of algorithm, a prediction can be made, to know if the material is going to fail. The SIMCENTER Test lab by Siemens is a commercial software which calculates the Rainflow matrix, results are matched between the aforementioned software and the one adapted in the TUC method by the Rychlik´s redefinition.

References

[1]. Standard practices for cycle counting in fatigue analysis. American Society for Testing and Materials, E 1049-85, 1997
[2]. Murakami, Y. The Rainflow Method in Fatigue. Butterworth Heinemann, 1991
[3]. Rychlik, I. A new definition of the rainflow cycle counting method. Int. J. Fatigue, 9, 119–121, 1987
[4]. Igor Rychlik. Extremes, rainflow cycles and damage functionals in continuous random processes. I Department of Mathematical Statistics, University of Lund. Box 118, S-22100 Lund, 1994
[5]. Siemens Simcenter Testing, Rainflow counting, SIMCENTER Testlab Desktop Neo. Retrieved from https://community.sw.siemens.com/s/article/rainflow-counting
[6]. Vibration data, Consulting and educational services in acoustics, 2010, Shock and vibration analysis. Retrieved from https://vibrationdata.com/
[7]. Azamfar, M. & Moshrefifar, M. Moshrefifar and Azamafar method. A new cycle counting method for evaluating fatigue life, Elsevier, International Journal of Fatigue, 69, pp 2–15, 2014

Manuel López Godínez, et. al. "Implementation of the algorithm of the TUC method for fatigue cycle counting". *IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE)*, 19(6), 2022, pp. 01-07.