Available online at www.sciencedirect.com
ScienceDirect
International Journal of Fatigue 30 (2008) 547–559
International Journal of Fatigue
www.elsevier.com/locate/ijfatigue

Equivalence of four-point and three-point rainflow cycle counting algorithms

C.H. McInnes *, P.A. Meehan

CRC Mining, School of Engineering, The University of Queensland, Building 45, Brisbane QLD 4072, Australia

Received 19 October 2006; received in revised form 28 February 2007; accepted 6 March 2007
Available online 20 March 2007

Abstract

Two forms of the rainflow cycle counting algorithm for fatigue damage prediction are shown to be equivalent, namely the three-point algorithm as presented by Bannantine et al. [Bannantine JA, Comer JJ, Handrock JL. Fundamentals of metal fatigue analysis. Englewood Cliffs, NJ: Prentice-Hall; 1990], and the four-point algorithm as presented by Amzallag et al. [Amzallag C, Gerey JP, Robert JL, Bahuaud J. Standardization of the rainflow counting method for fatigue analysis. Int J Fatigue 1994;16:287–293]. While it can be demonstrated by example that the same result is obtained by the two algorithms, no generalised proof of their equivalence has previously been presented. The proof is built on several identified properties of the four-point algorithm, including that certain modifications to the stress series being analysed do not alter the outcome.

Keywords: Fatigue analysis; Rainflow method; Cycle counting methods; Damage; Load histories

1. Introduction

Metal fatigue is of critical importance in parts subject to dynamic loads because it can result in sudden catastrophic failure at nominal stress levels well below the yield stress. The phenomenon of fatigue was first examined in detail in the 1800s. Thomas Woehler conducted the first fatigue test in 1852 [1] which lead to the discovery of the 'fatigue limit'; the stress range below which fatigue does not occur in steel. The typical shape of an S–N curve, which relates fatigue life (number of cycles to failure) to stress range for a constant amplitude test, was clarified in 1910 by Basquin [2]. The linear damage rule, first suggested by Palmgren in 1924, states that the amount of damage caused by a stress cycle is independent of the order in which the stress cycles occur [2]. This rule is not strictly accurate as tests have shown that the residual compressive stress at the crack tip caused by a tensile overload can temporarily arrest crack growth. However, it is generally considered a reasonable approximation for typical stress profiles encountered in practice.

It was not until the late 1960s that a suitable algorithm was developed for estimating fatigue damage (in terms of partial consumption of the lifetime of a part) from a stress profile that does not consist of steady or quasi-steady oscillations in stress. This 'rainflow' cycle counting algorithm [3] identifies hysteresis loops in stress–strain space. In order to estimate the fatigue damage, the impact of each individual hysteresis loop is assumed to be the same as the impact of a hysteresis loop of the same magnitude during a constant amplitude (alternating stress) test. The fraction of fatigue life consumed by a single hysteresis loop of a given stress magnitude is the inverse of the number of cycles to failure during such a test.

The rainflow algorithm has been adapted since the 1960s for use by computers. Two different forms are currently in popular use. The first algorithm, presented by Bannantine et al. [2] is most similar to the original one and is referred to as the three-point algorithm. The alternative four-point algorithm was popularised by Amzallag et al. [4]. The same

* Corresponding author. Tel.: +61 7 33652969; fax: +61 7 33654799.
E-mail address: c.mcinnes@uq.edu.au (C.H. McInnes).

0142-1123/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ijfatigue.2007.03.006

method for estimating the fatigue damage for a given hysteresis loop is incorporated into both algorithms; it is the method of identifying the hysteresis loops (i.e. the method of counting cycles) that differs.

In this paper, a general proof that the two algorithms are equivalent is provided. The two algorithms are first defined in Section 2. In Section 3, some properties of the four-point algorithm which are used to form the proof are defined. Section 4 contains the proof of equivalence of the two algorithms. It is first shown in Section 4.1 that the outcome of the four-point algorithm is not changed by rearranging the stress series prior to analysis. Sections 4.2 and 4.3 show that after these manipulations, the two sets of criteria for identifying hysteresis loops are equivalent. These two sections assume that there is only one overall maximum stress peak in the original series. By including additional manipulations to the stress series, Sections 4.4 and 4.5 expand on the previous sections to give a generalised proof of the equivalence of the two algorithms without any assumptions about the stress series being analysed.

2. Definition of rainflow counting algorithms

Rainflow cycle counting algorithms are methods for comparing measured stress data of varying amplitude with constant amplitude stress data. They identify closed hysteresis loops in stress–strain space and provide a mechanism for dealing with open hysteresis loops. Rainflow algorithms may be described in terms of the 'pagoda analogy' [2,5,6], however this form of the algorithm will not be presented here.

In the following subsections, the three and four-point algorithms will be described with reference to the arbitrary stress signal of Fig. 1, shown in stress–strain and stress–time spaces. The unmodified stress series is shown in Fig. 1a and b, while the rearrangement of the same series required in the three-point algorithm is shown in Fig. 1c and d. The hysteresis loops are easiest to identify visually in the stress–strain plot of Fig. 1a. The closed hysteresis loops are bound by stress value pairs EF and GH, while the open hysteresis loops are bound by AD and BC. With both three- and four-point algorithms, the stress pairs AD, BC, EF and GH in Fig. 1 are identified as the hysteresis loops. In order to do this, the peaks and valleys (local maxima and minima) must first be extracted from measured stress data to obtain the input for the algorithms. The hysteresis loops are then extracted from the series of peaks and valleys $s[\,]$. The algorithms are defined more rigorously in the subsequent sections to facilitate the generalised proof of equivalence.

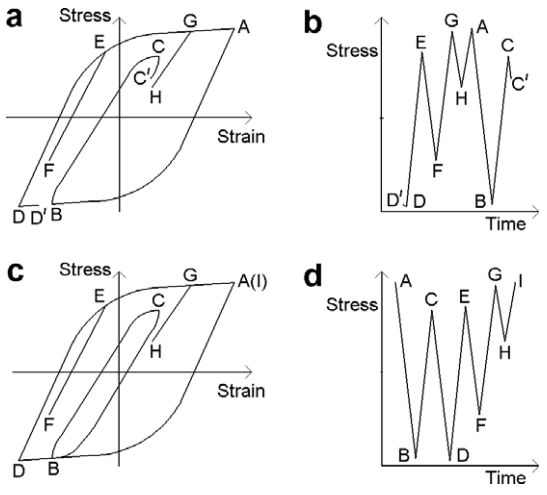

Fig. 1. Arbitrary stress series in stress–strain and stress–time spaces; (a, b) unmodified and (c,d) modified.

2.1. The three-point algorithm

In the three-point algorithm, each sequence of three consecutive points from the stress signal $s[\,]$ is considered in turn. Closed hysteresis loops are identified by comparing relative stress values. At the start of the algorithm, the unmodified stress series must be rearranged to begin and end with the overall maximum (or minimum) stress, as shown in Fig. 1c and d (modified from Bannantine et al. [2]). The algorithm then checks for hysteresis loops.

Each check for a hysteresis loop involves comparing the magnitude of the two stress ranges X_n and X_{n-1} formed by the sequential stress values $s[n-2, n-1, n]$,

$$\begin{aligned} X_n &= |s[n] - s[n-1]|, \\ X_{n-1} &= |s[n-1] - s[n-2]|, \end{aligned} \tag{1}$$

where $s[\,]$ is the stress sequence in the form of a series of peaks and valleys (local maxima and minima) and n is an index to the stress series. A hysteresis loop is identified if $X_n \geqslant X_{n-1}$, in which case the hysteresis loop $s[n-1, n-2]$ is removed from the series. The remaining stress values $s[n-4, n-3, n]$ are then considered in a similar manner. If no hysteresis loop is identified then the next sequence of three points, ending in $s[n+1]$, is considered and so on until the entire series is exhausted.

The following pseudocode details the three-point algorithm, $f_{3p}(\,)$. It shows the hierarchy of the major functions, which are defined in greater detail in Appendix A.

```
rearr( ) {rearrange s[ ] and delete one or two data points if necessary}
f3p( ) { f( ) {
           n = 0
         1 if at end of s[ ] STOP
           n = n + 1, assign next stress value to s[n]
         2 if n < 3 GOTO 1
           crit( ) { calculate X_n and X_{n-1}
                     if X_n < X_{n-1} GOTO 1 }
           store X_{n-1} for later fatigue damage calculation
           remove X_{n-1}(s[n-1, n-2]) from stress
           sequence so that s[n] becomes s[n-2], n = n-2
           GOTO 2
       } }
```

(2)

In (2), *crit*( ) is the criterion for identifying a hysteresis loop.

The step of calculating the estimated fatigue damage based on each stress range X_{n-1} can be performed during each iteration or after completion of the algorithm. An example of the algorithm applied to the modified stress signal of Fig. 1 is shown graphically in Fig. 2a. Frame 1 shows the rearranged stress signal of Fig. 1c. The three points and two stress ranges under consideration at any time are indicated by the dashed lines. The first two stress ranges, highlighted in frame 2, do not satisfy the three-point criterion. Points B, C and D, highlighted in frame 3, do satisfy the criterion and the hysteresis loop is removed in frame 4. Other hysteresis loops are identified in frames 7, 10 and 12. Examples of the three-point algorithm implemented in FORTAN are available in the literature [2,7].

The three-point algorithm is unsuitable for use in a real-time scenario because it requires the entire stress history to be rearranged at the start of the analysis. A more suitable form of the algorithm that is commonly used is presented by Amzallag et al. [4] and is described subsequently.

2.2. The four-point algorithm

The four-point algorithm differs from the three-point algorithm in that there is no re-arrangement of the stress series prior to analysis. This requires an extra point to be considered at each step in order to identify closed hysteresis loops. However, it allows the algorithm to be initiated prior to obtaining all measured stress data, hence allowing realtime analysis. For comparison, Fig. 3 shows a closed hysteresis loop and the main criteria that form part of the three- and four-point algorithms for its detection.

In the four-point algorithm, hysteresis loops are identified in sequences of three stress ranges, where the second of the three stress ranges is smaller than or equal to the first and third (see Fig. 3c). This second stress range is identified as a hysteresis loop, analysed and discarded from the series in a similar manner to the three-point algorithm. The stress ranges considered at each step are

$$\begin{aligned} X_n &= |s[n] - s[n-1]| \\ X_{n-1} &= |s[n-1] - s[n-2]| \\ X_{n-2} &= |s[n-2] - s[n-3]| \end{aligned} \tag{3}$$

The four-point algorithm, $f_{4p}(\,)$, may be described in the form of pseudocode as

$$f_{4p}(\,)\left\{\begin{array}{l} f_{4p1}(\,)\left\{\begin{array}{ll} & n = 0 \\ & pass = 1 \\ 1 & \text{if at end of } s[\,] \text{ and } pass = 1 \text{ GOTO 3} \\ & \text{if at end of } s[\,] \text{ and } pass = 2 \text{ STOP} \\ & n = n + 1, \text{ assign next stress value to } s[n] \\ 2 & \text{if } n < 4 \text{ GOTO 1} \\ & crit(\,)\left\{\begin{array}{l} \text{calculate } X_n, X_{n-1} \text{ and } X_{n-2} \\ \text{if } X_n < X_{n-1} \text{ or if } X_{n-2} < X_{n-1} \text{ GOTO 1} \end{array}\right\} \\ & \text{store } X_{n-1} \text{ for later fatigue damage calculation} \\ & \text{remove } X_{n-1}(s[n-1, n-2]) \text{ from stress} \\ & \text{sequence so that } s[n] \text{ becomes } s[n-2], n = n-2 \\ & \text{GOTO 2} \end{array}\right\} \\ f_{4p2}(\,)\left\{\begin{array}{ll} 3 & n = 0 \\ & pass = 2 \\ & dbl(\,)\left\{\begin{array}{l} \text{repeat } s[\,] \text{ sequentially and delete one or two data} \\ \text{points if necessary to maintain the max–min sequence} \end{array}\right\} \\ & \text{GOTO 1} \end{array}\right\} \end{array}\right\}. \tag{4}$$

This pseudocode is also expressed as a group of functions, detailed in Appendix.

Fig. 2b shows an illustrative example of the four-point algorithm applied to the arbitrary stress series of Fig. 1a. Hysteresis loops are identified in frames 3, 5, 12 and 13. For the three-point algorithm, the same loops were identified in frames 7, 10, 3 and 12, respectively, in Fig. 2a. By frame 8, all closed hysteresis loops in the original stress series have been found (EF and GH). The remaining stress values, called the residue (*res*[ ]), form open hysteresis loops (AD and BC). They are only 'closed' for the three-point algorithm due to the initial rearrangement of the stress series (see first line of pseudocode in Section 2.1). The equivalent action for the four-point algorithm is taken in frame 9 by repeating the stress values sequentially. This adds the stress values J, K, L and M (the repeated values D, A, B and C, respectively) to give a new sequence D′DABCJKLMC′. Points D′ and C′ are removed from the central part of the sequence to maintain the peak–valley sequence, but remain at the start and end of the new sequence. The last two hysteresis loops are then identified, leaving the remaining values in frame 16, which are the same as the residue in frame 8 [4].

Note that Rychlik [6] gives an equivalent definition of a closed hysteresis loop (referred to as a toplevel-up cycle), but treats the residue differently. Rychlik's definition has the advantage of not requiring the smaller closed loops to be removed in order to identify the larger ones.

While the same results are obtained from the three-point and four-point algorithms for the same data, no general proof that the two methods are equivalent, is available in the literature.

3. Properties of four-point algorithm

The following properties of the four-point algorithm form part of the proof of equivalence of the two algorithms that follows. The first property is essentially a redefinition of the four-point criterion and the second is a property of the four-point criterion, related to the outer values of a satisfying sequence. The third property relates to the order in which the criterion is applied, which can be varied without changing the outcome. The next introduces the properties of an increasing–decreasing sequence. These properties are then developed to identify sections of *s*[ ], called 'end-point bounded sequences', that can be analysed in isolation. The following four sections reveal properties of the first and second pass of the algorithm, the residue, and the hysteresis loops identified in the first pass and from the repeated residue. The final property is that the number of hysteresis loops in *rearr*(*s*[ ]) is equal to the number of valleys.

Unless otherwise stated, the term maximum and minimum refer to the overall (global) maximum and minimum stress in the series being analysed.

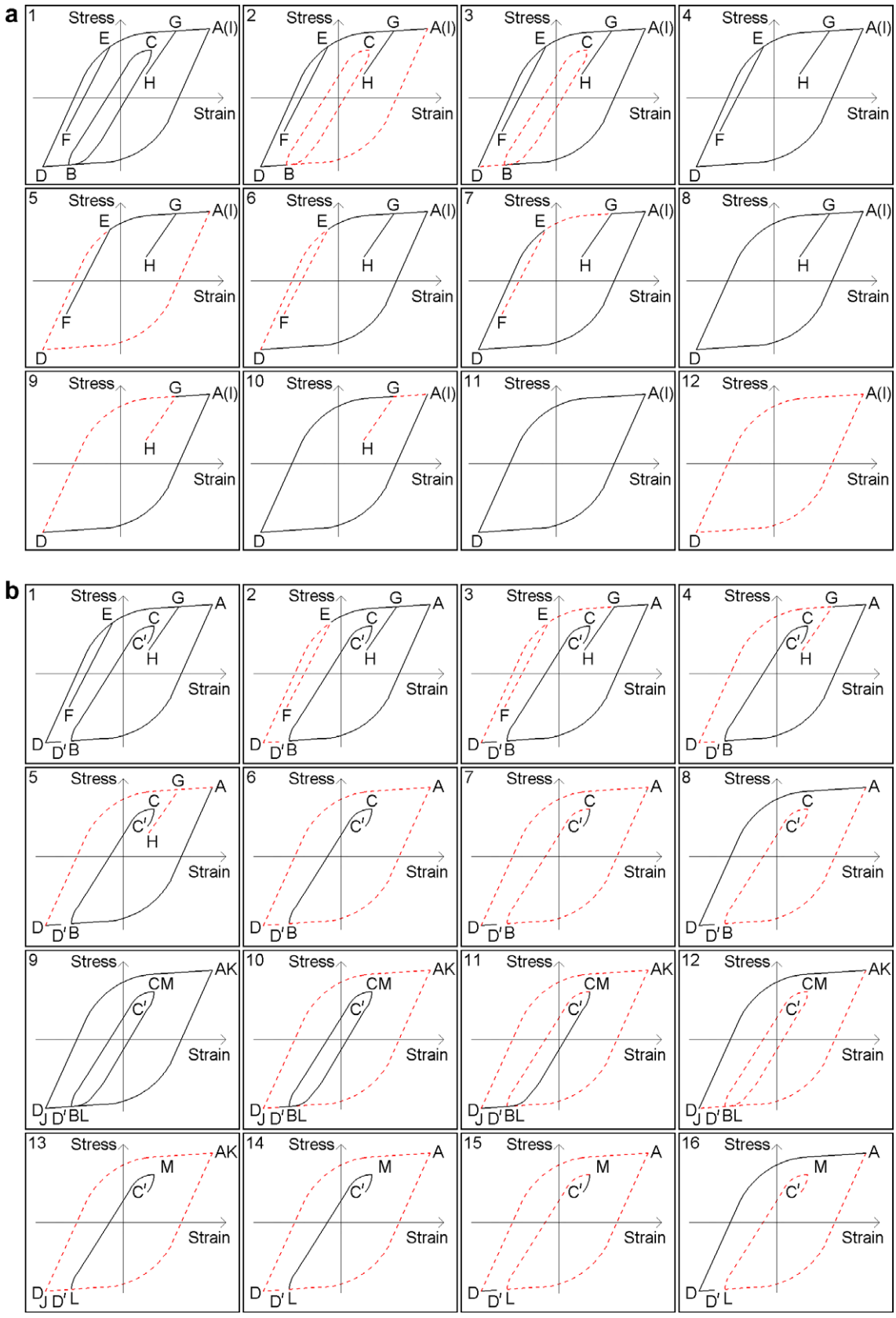

Fig. 2. Sequential steps of the (a) three- and (b) four-point algorithm, in stress–strain space.

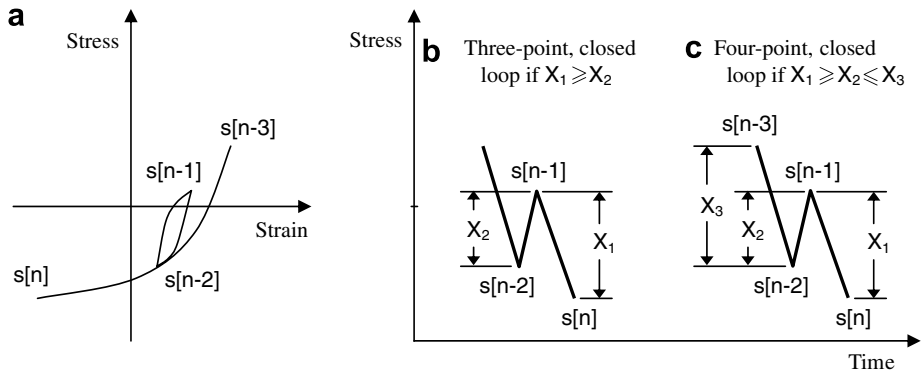

Fig. 3. Closed hysteresis loop in: (a) stress–strain space and (b,c) stress–time space, showing criteria for detection.

3.1. The four-point criterion

Property 3.1. *A sequence of four points* $s[n-3,n-2,n-1,n]$ *satisfies the four-point criterion if,*

$$s[n-3,n] = [\max(s[n-3,n-2,n-1,n]),\ \min(s[n-3,n-2,n-1,n])]$$
$$\text{or } s[n-3,n] = [\min(s[n-3,n-2,n-1,n]),\ \max(s[n-3,n-2,n-1,n])] \quad (5)$$

Proof. This property can be inferred directly from the algorithm definition (see (4)). □

Remark. Such a sequence of four points will be referred to as a 'satisfying sequence'.

3.2. Replacing the outer values does not affect the four-point criterion

In this section, it is shown that replacing the outer values of a satisfying sequence with a higher peak and lower valley does not alter whether the four points satisfy the four-point criterion.

Property 3.2

$$crit(4,n,s[n-3,n-2,n-1,n]) = crit(4,n,[x_1,s[n-2,n-1],x_2]), \quad (6)$$

where $s[n-3,n-2,n-1,n]$ *is a satisfying sequence and where*

$x_1 \leqslant s[n-3]$ *and* $x_2 \geqslant s[n]$,
if $s[n]$ *is a peak* $(s[n] > s[n-1])$, *or*
$x_1 \geqslant s[n-3]$ *and* $x_2 \leqslant s[n]$,
if $s[n]$ *is a valley* $(s[n] < s[n-1])$.

Proof. By definition, the two inner values of a satisfying sequence, $s[n-1,n-2]$, form the closed hysteresis loop. According to Property 3.1, the two outer values must be the overall minimum and maximum from the sequence of four (see also Fig. 3). Thus, increasing the outer peak or decreasing the outer valley (by replacement) will not affect whether the sequence satisfies the four-point criterion. □

Remark. This property allows points to be removed to maintain the peak–valley sequence when joining series of stress values without affecting the identification of any closed hysteresis loops, and also forms the basis for the next property to be considered.

3.3. Uniqueness of closed hysteresis loops with respect to order of application of four-point criterion

The four-point algorithm (4) consists of the four-point criterion *crit*( ), a set of instructions 'nested' around this criterion for applying it to a series in a specific order f_{4p1}( ), and a set of instructions nested around this for dealing with the residue, f_{4p}( ). It will be proven that the order in which the criterion *crit*( ) is applied is not critical to the outcome of the algorithm.

Property 3.3

$$f_{4pa}(s[\,]) = f_{4p1}(s[\,]), \quad (7)$$

where f_{4pa}( ) *is an alternative algorithm to* f_{4p1}( ) *that tests sub-sequences of four points from the series in a random order for closed hysteresis loops until all have been identified, as defined in Appendix.*

This will be proven by considering the possibility of the existence of closed hysteresis loops that are adjacent or overlapping in the strain history series and considering the implications of removing one hysteresis loop on the identification of another. That is, it will be shown that the removal of a hysteresis loop in any given order does not alter the identification of nearby hysteresis loops.

Proof. Consider a satisfying sequence $s[n-3,n-2,n-1,n]$. If the sequence $s[n-1,n,n+1,n+2]$ also satisfies the four-point criterion then the hysteresis loop $s[n,n+1]$ can be removed without altering the identification of the first hysteresis loop or its magnitude. The point $s[n]$ will be replaced with $s[n+2]$ in the first sequence. This does not alter the magnitude of the hysteresis loop, which is formed by $s[n-2,n-1]$. Furthermore, $s[n+2]$ will be equal to or greater than $s[n]$ if it is a local maxima and equal or smaller if it is a local minima, thus not affecting the identification of the first hysteresis loop as per Property

3.2 (6). Likewise, removal of the first hysteresis loop will not affect the magnitude or identification of the second. Similar arguments can be used to show that removal of a hysteresis loop will not affect the identification or magnitude of any other hysteresis loops. Therefore by generalisation of this case, the order in which the criterion is applied is not critical. □

Remarks. Fig. 4 demonstrates this property for the case outlined in the proof. Frames b and c show the paths taken when different sub-sequences of four points are considered first, with frame d showing identical remaining stress values after two closed hysteresis loops have been removed.

3.4. No closed hysteresis loops in increasing–decreasing sequences

In this section, it is shown that there are no closed hysteresis loops (satisfying sequences) in an increasing–decreasing sequence and that there must be closed hysteresis loops in all other sequences. An increasing–decreasing sequence is a sequence in which successive stress ranges $|s[n]-s[n-1]|$ preceding one of the extrema are all higher than the previous stress range, and successive stress ranges after the extrema are all lower than the previous stress range. An example of this is shown in Fig. 5. Note that the extrema can be the first or last point in the sequence, meaning that a sequence that increases monotonically over its length is considered a special type of increasing–decreasing sequence, as is a sequence that decreases monotonically.

Property 3.4. $[f_{4p1}(s[\,])]_o = 0$ *if* $s[\,]$ *is an increasing–decreasing sequence, otherwise* $[f_{4p1}(s[\,])]_o > 0$. *The subscript* 'o' *refers to the stress ranges that are output, not the residue. A sequence* $s[1,\ldots,n^*,\ldots,N]$ *is increasing–decreasing if,*

$$\begin{aligned} &s|[n]-s[n-1]| > |s[n-1]-s[n-2]| \quad \text{for all } n \leqslant n^* \\ &\text{and } |s[n]-s[n+1]| > |s[n+1]-s[n+2]| \quad \text{for all } n \geqslant n^* \end{aligned} \tag{8}$$

where n^* *can be either the maximum or minimum (*B *or* A*, respectively, in Fig. 5). Note that if the residue has two maximums,* n^* *must be the overall minimum which will be located between the two maximums, and vice versa.*

Proof. Firstly it is noted that every sub-sequence of an increasing–decreasing sequence must also satisfy (8). Furthermore, (8) is mutually exclusive of the definition of a satisfying sequence in Property 3.1 (5), (see Fig. 3). Thus, no satisfying sequence can be found in an increasing–decreasing sequence since the first pair of stress ranges is decreasing and the next is increasing. Therefore, there can be no closed hysteresis loops in an increasing–decreasing sequence. This is evident in Fig. 5a. Furthermore, if (8) is not satisfied there must be a sequence of three consecutive stress ranges in the sequence with the central stress range being equal to or smaller than the outer two, i.e. the sequence must contain a 'satisfying sequence' and a closed hysteresis loop. Thus, the converse is also true. □

Remarks. Note that three consecutive equal stress ranges can satisfy the definition of a satisfying sequence but not the definition of an increasing–decreasing sequence. Eq. (8) can be satisfied with two equal consecutive stress ranges and n^* being the point they have in common.

3.5. Independence of 'end-point bounded sequences'

In this section, it will be proven that a sequence of points, in which the first and last points are the maximum and minimum of that sequence, can be analysed separately from the rest of the series without altering the outcome of the algorithm. Fig. 6 shows two examples of four-point analyses of a series containing an end-point bounded sequence. Frame a shows the initial stress series, followed by frames b and c in which two hysteresis loops have been removed in each case. Frame b shows the normal order of hysteresis loop identification while frame c shows the alternative case of removing two hysteresis loops from within the end-point bounded sequence first. Note that in frame c, none of the points outside of the end point bounded sequence are within the satisfying set of four points for removal of the two loops. In frame d, all four closed hysteresis loops have been removed. The identified loops were identical in both cases, as is the remaining sequence of points.

Property 3.5

$$\begin{aligned} f_{4p}(s[1,\ldots,N]) = {} & f_{4p}(s[1,\ldots,n_a,n_b,\ldots,N]) \\ & + f_{4p1}(s[n_a,\ldots,n_b]), \end{aligned} \tag{9}$$

where $s[n_a,\ldots,n_b]$ *is an 'end-point bounded sequence', which is defined as*

$$\begin{aligned} &s[n_a] \geqslant s[n] \geqslant s[n_b] \quad \text{or} \quad s[n_a] \leqslant s[n] \leqslant s[n_b] \\ &\text{for all } n_a \leqslant n \leqslant n_b. \end{aligned} \tag{10}$$

Proof. It can be concluded from Property 3.3 that if it was possible to identify (and remove) all intermediate points (i.e. all points between $s[n_a]$ and $s[n_b]$) as closed hysteresis loops then (9) would hold. That is, combining Property 3.3 (7) and the definition $f_{4pa}(\,)$ of in Appendix,

$$\begin{aligned} f_{4p1}(s[1,\ldots,N]) &= f_{4pa}(s[1,\ldots,N]) \\ &= f_{4pa}(s[1,\ldots,n_a,n_b,\ldots,N]) \\ &\quad + f_{4pa}(s[n_a,\ldots,n_b]) \\ &= f_{4p1}(s[1,\ldots,n_a,n_b,\ldots,N]) \\ &\quad + f_{4p1}(s[n_a,\ldots,n_b]) \end{aligned} \tag{11}$$

if $f_{4pa}(s[n_a,\ldots,n_b])$ and f_{4p1} $(s[n_a,\ldots,n_b])$ remove all the intermediate points, $s[n_a+1,\ldots,n_b-1]$. The absence of

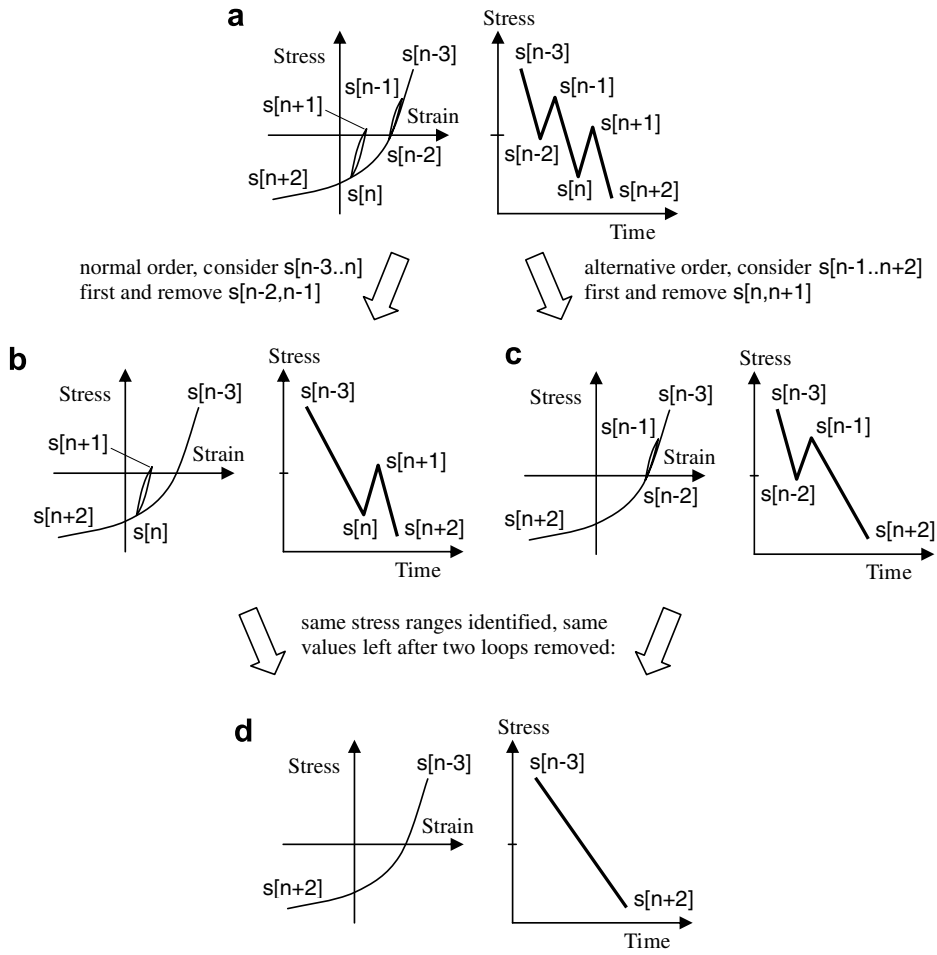

Fig. 4. Demonstration of uniqueness of closed hysteresis loops: (a) initial series, (b and c) alternative intermediate steps, and (d) remaining points.

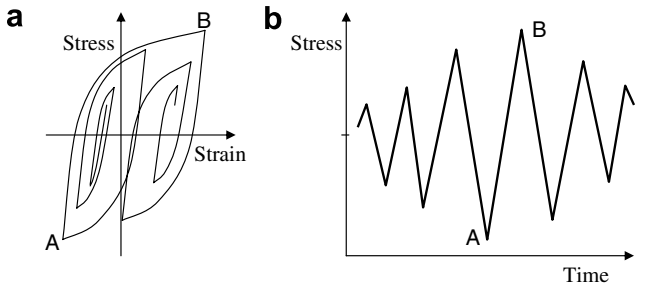

Fig. 5. An increasing–decreasing sequence in (a) stress–strain space and (b) stress–time space.

these intermediate points from the residue makes (11) equivalent to (9).

Furthermore, it is always possible to identify (and remove) all intermediate points if $s[n_a,\ldots,n_b]$ satisfies (10), because the conditions defined in (10) and the conditions for an increasing–decreasing sequence given in Property 3.4 (8) are mutually exclusive for all but the trivial case of a series consisting of only two points. That is, $s[n_a,\ldots,n_b]$ can only satisfy Property 3.4, (8) and (10) if it consists of only two points. If it contained an intermediate point, (8) would require that this point be greater than both $s[n_a]$ and $s[n_b]$, or less than both of them. However, (10) would require that the intermediate point lie between them or is equal to them. Thus, it has been shown using (10), (11) and Property 3.4 that (9) holds for all end-point bounded sequences. □

Remarks. An end-point bounded sequence (10) is similar to a satisfying sequence (5), except that a satisfying sequence must contain four points whereas an end-point bounded sequence can contain any positive even number of points. (The number of points must be even so that the first and last points are a peak and valley.) Thus, a satisfying sequence is a type of end-point bounded sequence. The more general end-point bounded sequence is used later to effectively break the sequence being analysed down into the residue and a number of end-point bounded sequences.

3.6. Relationship between first and second pass of algorithm

Apart from the inclusion of the *dbl*( ) function for treating the residue, the second pass of the four-point algorithm is equivalent to the first pass. That is,

Property 3.6

$$f_{4p2}(res[\,], out[\,]) = out[\,] + [f_{4p1}(dbl(res[\,]))]_o. \quad (12)$$

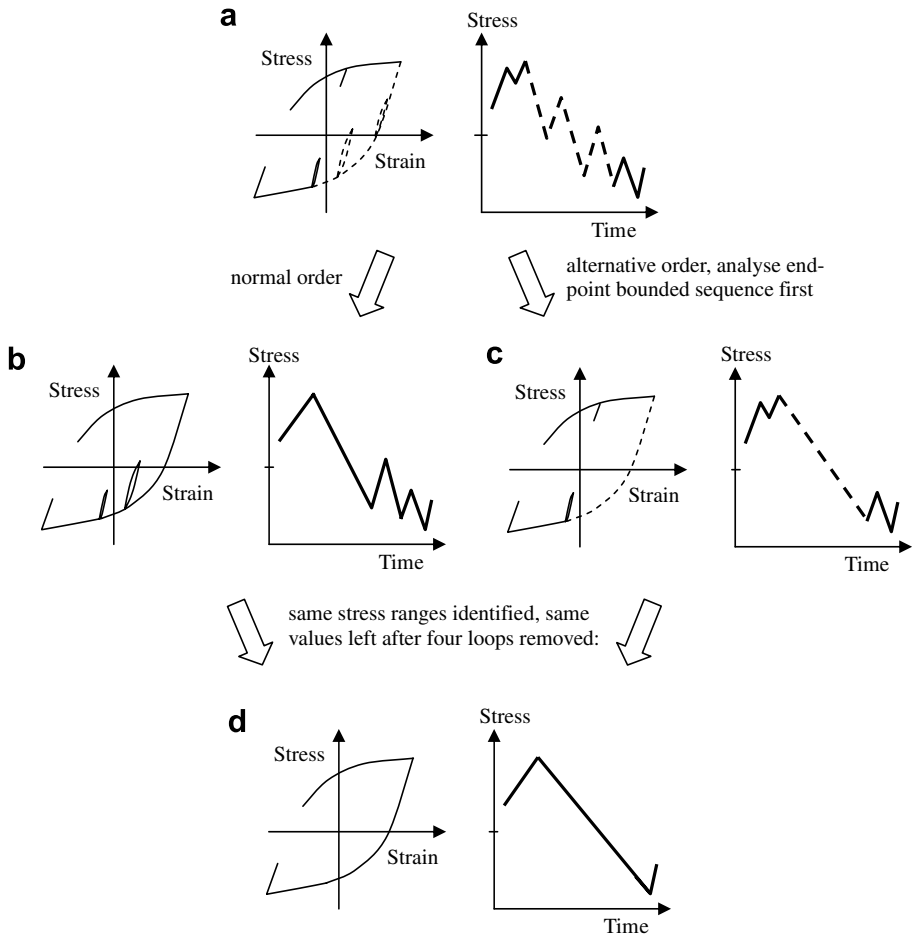

Fig. 6. Demonstration of the independence of an arbitrary end-point bounded sequence (indicated by dashed lines).

where the subscript 'o' *refers to the stress range component of the output. For future use this notation can be dropped to be expressed as*

$$f_{4p2}(res[\,], out[\,]) = out[\,] + f_{4p1}(dbl(res[\,])). \quad (13)$$

Proof. This property can be inferred directly from the definition of $f_{4p2}(\,)$ in Appendix. The two lines of pseudocode are equivalent to (12). □

Remarks. It is noted that based on this property, the four-point algorithm may be redefined as

$$f_{4p}(s[\,]) = f_{4p1}(s[\,]) + f_{4p1}(dbl(res[\,])). \quad (14)$$

3.7. The residue is an increasing–decreasing sequence

Property 3.7. *The residue from the first pass of the four-point algorithm is an increasing–decreasing sequence.*

Proof. This follows directly from Property 3.4 because the residue is by definition a sequence with no remaining closed hysteresis loops. If the residue were not an increasing–decreasing sequence, it would contain closed hysteresis loops as shown in Section 3.4.

3.8. Residue points define bounds of end-point bounded sequences in the original series

According to Property 3.5, every pair of points that remains in the residue are the end points of an end-point bounded sequence in the original stress series, though some may consist of only two points.

Property 3.8

$$f_{4p}(s[\,]) = f_{4p}(res[\,]) + \sum f_{4p1}(\text{end-point bounded sequences}). \quad (15)$$

Proof. Two points alone will always form an end-point bounded sequence, according to Property 3.5. Any pairs of points that were removed from between the remaining pairs in the residue must have been bound by them, or bound by other points that were bound by them, as per the definition of a satisfying sequence (see Property 3.1). Thus, they satisfied the conditions of an end-point bounded sequence (see Property 3.5). Thus, the first pass of the algorithm is equivalent to extracting all of these end-point bounded sequences and analysing them separately as described by (9). □

3.9. Location of hysteresis loops identified in second pass of algorithm

In this section, it will be proven that all hysteresis loops identified in the second pass of the algorithm are located in the central region of the repeated residue between the two largest stress ranges. More specifically,

Property 3.9

$$f_{4p1}(res[1,\ldots,i_{m\_r},i_{m\_r}+1,\ldots,N',p,\ldots,i_{m\_r},i_{m\_r}+1,\ldots,N]) = f_{4p1}(res[i_{m\_r},i_{m\_r}+1,\ldots,N',p,\ldots,i_{m\_r},i_{m\_r}+1]), \quad (16)$$

where i_{m_r} is the index of the first extrema (overall maximum or minimum) in the residue, N is the number of points in the sequence, and

$$N' = N \quad \text{or} \quad N-1 \quad \text{and} \quad p = 1 \text{ or } 2, \quad (17)$$

depending on whether a point is removed to maintain the peak–valley sequence.

Proof. The proof consists of showing that the sub-sequence of *res*[ ] on the right-hand side of (16) contains all of the points necessary to identify the hysteresis loops. According to Property 3.7 the residue is an increasing–decreasing sequence. Thus, the first part of the repeated residue $res[1,\ldots,i_{m\_r}]$ is an increasing sequence and contains no hysteresis loops. The last part $res[i_{m\_r}+1,\ldots,N]$ is a decreasing sequence and thus also contains no closed hysteresis loops. The central part $res[i_{m\_r}+1,\ldots,N',p,\ldots,i_{m\_r}]$ consists of one end-point bounded sequence so all points from the central part, other than $res[i_{m\_r}+1,i_{m\_r}]$ will be identified as closed hysteresis loops and removed, as shown in Property 3.5. This leaves $res[1,\ldots,i_{m\_r},i_{m\_r}+1,i_{m\_r},i_{m\_r}+1,\ldots,N]$, which contains one last satisfying sequence (see Section 3.1) formed by $res[i_{m\_r},i_{m\_r}+1,i_{m\_r},i_{m\_r},+1]$. After this hysteresis loop is removed the remaining points are $res[1,\ldots,i_{m\_r},i_{m\_r}+1,\ldots,N]$, which is the original residue. Therefore, none of the points removed from the sequence in (16) are required for the identification of the hysteresis loops so removing them does not affect the results of the four-point algorithm. □

Remarks. Fig. 7 highlights the central part of the repeated residue identified by (16) from a sample stress history. The dashed stress ranges indicate which points are not present on the right-hand side of Eq. (16).

Note that although it is possible to have three extremas in the residue, the removal of one as per (16) does not negate the equality. There can be either two maximums separated by a minimum, or two minimums separated by a maximum. This will give six extremas in the repeated residue. In the case of two maximums in the original residue, there will be two maximums in the repeated residue with no (overall) minimum separating them. Thus, one of these maximums must be removed as part of a smaller hysteresis loop. This leaves five extremas. Two will be removed with the identification of the largest hysteresis loop, leaving three, which is insufficient for the identification of another hysteresis loop. The removal of one extrema as per (16) leaves four instead of five, which is still sufficient for the identification of the last (the largest) hysteresis loop. The same argument applies for two minimums in the original residue.

3.10. Number of hysteresis loops in a sequence of points

Property 3.10. *The number of hysteresis loops identified by $f_{4p}(rearr(s[\,]))$ is equal to the number of valleys in the sequence rearr(s[ ]).*

Proof. According to Property 3.5 and the definition in Appendix, *rearr*(*s*[ ]) consists of two end-point bounded sequences bound by a maximum on each end and a minimum somewhere in between. Thus, one hysteresis loop will be removed for each valley other than the minimum. The minimum is included in the remaining hysteresis loop which is identified in the second pass of the algorithm. Note that only one hysteresis loop is identified because the residue consists of only three points. □

Remark. Fig. 1c can help to visualise this property.

4. Proof of equivalence of four-point and three-point algorithms

The properties developed in Section 3 will now be used to show that the two algorithms are equivalent. First, it will be shown that the output of the four-point algorithm is not changed if the stress series is first manipulated as it is at the start of the three-point algorithm. It will then be shown that after this manipulation, the two algorithms will always produce the same output. A slightly more complex proof is required for the case where there are multiple peaks in the original stress series equal to the overall maximum stress. For simplicity, the proof is first established with the assumption that there is only one such overall maximum stress value, and subsequently generalised.

Both algorithms use the same function to estimate the fatigue damage from the stress ranges, so it is sufficient to prove that

$$f_{4p}(s[\,]) = f_{3p}(rearr(s[\,])). \quad (18)$$

Note that the estimated fatigue damage, which is based on the linear damage rule (see Section 1), is not affected by the order of the stress ranges output from any rainflow algorithms.

Fig. 8 shows the derivative relationship between the various properties, where the shaded properties form the proof of equivalence to be detailed in the following.

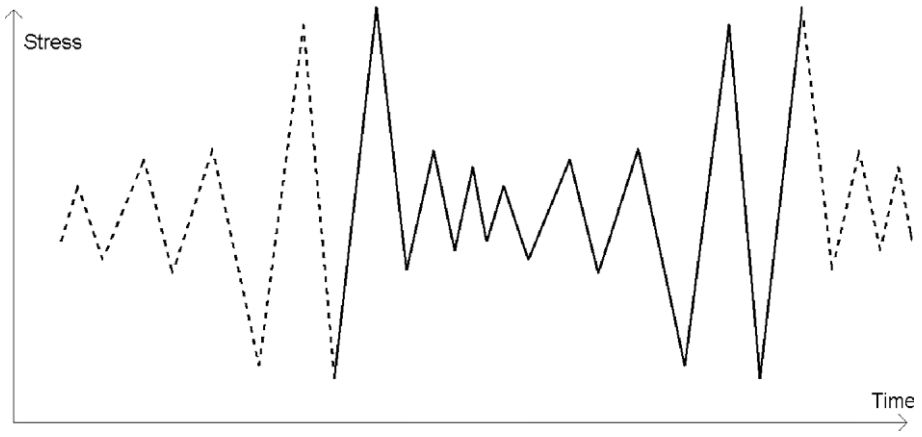

Fig. 7. Removal of 'outer' stress values from the repeated residue, as per Eq. (16).

4.1. Consistency of four-point algorithm after stress series manipulation

It is first proven that the manipulation of the stress series performed at the start of the three-point algorithm can also be performed at the start of the four-point algorithm without altering its outcome,

Property 4.1

$$f_{4p}(s[\,]) = f_{4p}(rearr(s[\,])). \tag{19}$$

Proof. This will be proven using a number of modifications equivalent to *rearr*( ), each of which do not effect the outcome of the algorithm.

Combining Properties 3.6 (14) and 3.9 (16) gives

$$f_{4p}(s[\,]) = f_{4p1}(s[\,]) + f_{4p1}(res[i_{m\_r}, i_{m\_r} + 1, \ldots, N', p, \ldots, i_{m\_r}, i_{m\_r} + 1]). \tag{20}$$

Note also that, from the definition of *rearr*( ) in Appendix

$$res[i_{m\_r}, i_{m\_r} + 1, \ldots, N', p, \ldots, i_{m\_r}, i_{m\_r} + 1] = rearr(res[\,])\&res[i_{min}], \tag{21}$$

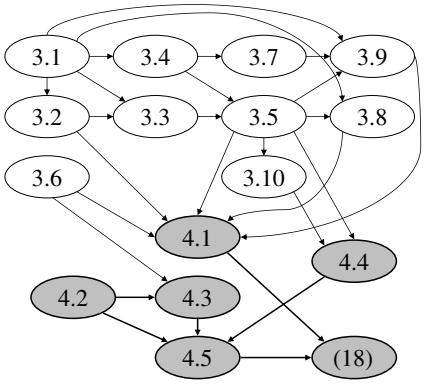

Fig. 8. Flow chart of property dependencies.

where & refers to the addition of the minimum to the start of the sequence if $res[i_{m\_r}]$ is a peak, or to the end of the sequence $res[i_{m\_r}]$ if is a valley. Combining Property 3.8 (15) with (20), (21) and $f_{4p1}(res[\,]) = 0$ gives

$$f_{4p}(s[\,]) = f_{4p1}(rearr(res[\,])\&res[i_{min}]) + \sum f_{4p1}\ \text{(end-point bounded sequences)}. \tag{22}$$

Consider now the corollary application of Property 3.5 (the independence of 'end-point bounded sequences') to (22). In particular, by 'reinserting' the end-point bounded sequences into the modified residue, the summation term may be removed and *res*[ ] replaced with *s*[ ]. This would give

$$f_{4p}(s[\,]) = f_{4p1}(rearr(s[\,])\&s[i_{min}]). \tag{23}$$

Based on Property 3.5, the hysteresis loops contained within the two end-point bounded sequences forming $rearr(s[\,])$ could be extracted by $f_{4p1}(rearr(s[\,]))$, leaving a residue of three points,

$$f_{4p1}(rearr(s[\,])\&s[i_{min}]) = f_{4p1}(rearr(s[\,])) + f_{4p1}(res[s[i_{max}], s[i_{min}], s[_{max}]]\&s[i_{min}]). \tag{24}$$

The second term, $f_{4p1}(res[\,]\&s[i_{min}])$ identifies a single hysteresis loop. The same hysteresis loop would be identified by a second pass analysis of the residue. Thus,

$$f_{4p1}(rearr(s[\,])\&s[i_{min}]) = f_{4p1}(rearr(s[\,])) + f_{4p2}(res[s[i_{max}], s[i_{min}], s[i_{max}]]). \tag{25}$$

Substituting (25) into (23) gives

$$f_{4p}(s[\,]) = f_{4p1}(rearr(s[\,])) + f_{4p2}(res[s[i_{max}], s[i_{min}], s[i_{max}]]) = f_{4p}(rearr(s[\,])) \tag{26}$$

because $[s[i_{max}],s[i_{min}],s[i_{max}]]$ is the residue from $f_{4p1}(rearr(s[\,]))$. Therefore, (19) holds and the result of the four-point algorithm does not change if, at the start, the stress

series is rearranged to begin and end with the maximum, as it is for the three-point algorithm. □

Remarks. The function *rearr*(*res*[ ]) does not separate any consecutive stress values because the maximum at which the series is broken is left at the beginning and end of the new series. Thus, it will be possible to reinsert the end-point bounded sequences between the remaining stress values. While the function may remove one or both end points to maintain the peak–valley sequence, they will be replaced with a higher value if they are a maximum or a lower value if the are a minimum. Thus, this removal will not affect the identification of any hysteresis loops from within the end-point bounded sequences (see Property 3.2). Also, $res[i_{\min}]$ can be replaced with $s[i_{\min}]$ as they are the same value. Thus, (23) holds.

4.2. Equivalence of three-point and four-point criteria after series manipulation

In this section, it is shown that when the two algorithms are applied to *rearr*(*s*[ ]), the two criteria and the logic for applying them are equivalent. That is,

Property 4.2

$$\Big[crit(q,n,s'[\,])\ \text{AND}\ n \geqslant q\Big]_{q=3,n} = \Big[crit(q,n,s'[\,])\ \text{AND}\ n \geqslant q\Big]_{q=4,n}, \tag{27}$$

holds for all values of n that will be encountered by the algorithm in analysing rearr(*s*[ ]), *except when the last three available points are being considered. In this case, the second pass of the four-point algorithm is equivalent to the three-point algorithm. It is assumed that there is only one peak equal to the maximum stress in the original series.*

Proof. Several different cases need to be considered to show that this property holds under all possible circumstances. For conciseness and clarity, the reasons for the equivalence of the criteria are listed in Table 1 according to the number of stress values that have been input (minus those removed after the identification of hysteresis loops). This is the number *n* in the left-hand column. For direct comparison, the two columns on the right show the outcome of the two sets of criteria. If four or more points have been input, the criteria either both return true or both return false, depending on the stress values. For the general case involving five or more points the reasons for equivalence, expressed as a comparison of the stress values, depend on whether *n* is even or odd.

As an example of one case from the table, consider the case where five points have been read in. This case is detailed in the last two rows of the table, which begin with $n = 5+$. The variable *n* is odd, so the second example given in each explanation in the second column applies. If $s[5] < s[3]$ both sets of criteria will return F (false) because $X_n < X_{n-1}$ (see Appendix), as described in the second last row. If $s[5] \geqslant s[3]$ then both sets of criteria return T (true) because $X_n \geqslant X_{n-1}$, as described in the bottom row. The four-point criterion also requires that $X_{n-2} \geqslant X_{n-1}$, however this must also be true otherwise the criterion would have returned true at $n = 4$. The fifth point will only be read in if the criteria fail at $s[4]$. □

Table 1
Equivalence of three and four-point criteria

n	Condition/reason for equivalence	*crit*(*q*,*n*,*s*′[ ]) returns	
		3p	4p
1, 2	$n < q$	F	F
3	$n < 4, s[1] > s[3]$	F	F
3	$s[1] = s[3]$	T	F[a]
4	$s[4] > s[2]$	F	F
4	$s[4] \leqslant s[2]$, four-point criterion satisfied because $s[1] > s[3]$	T	T
5+	$s[n_{\text{even}}] > s[n-2]$ or $s[n_{\text{odd}}] < s[n-2]$	F	F
5+	$s[n_{\text{even}}] \leqslant s[n-2]$ or $s[n_{\text{odd}}] \geqslant s[n-2]$, four-point criterion satisfied because $s[n_{\text{even}}-3] > s[n-1]$ or $s[n_{\text{odd}}-3] < s[n-1]$, otherwise both criteria would have returned *TRUE* at $n-1$ and $s[n-2, n-3]$ would have been removed	T	T

[a] The same hysteresis loop is identified in the second pass of the four-point algorithm over the last three points. This condition only arises for the last three points.

Thus, the two sets of criteria and the logic for applying them are equivalent.

4.3. Equivalence of three-point algorithm and four-point algorithms after series manipulation

In Section 4.2, it has been shown that the criteria *crit*( ) for the three-point and four-point algorithms are equivalent. The equivalence of the algorithms as a whole, $f_{4p}(\,)$ and $f_{3p}(\,)$, when applied to *rearr*(*s*[ ]) will be proven in the following:

Property 4.3

$$f_{4p}(rearr(s[\,])) = f_{3p}(rearr(s[\,])), \tag{28}$$

with the assumption of one maximum in s[ ].

Proof. Property 3.6 (14) shows that the four-point algorithm $f_{4p}(\,)$ can be defined in terms of $f_{4p1}(\,)$. The definitions of $f_{4p1}(\,)$ and $f_{3p}(\,)$ in Appendix show that, apart from rearranging the points at the start of the algorithm, the only difference between these two algorithms is the criteria and logic for applying them. Thus, it is sufficient to show that the criteria and logic for applying them are equivalent and (28) follows directly from Property 4.2 (27). Therefore, the four-point algorithm is equivalent to the three-point algorithm when applied to *rearr*(*s*[ ]). □

Remarks. Properties 4.1 (19) and 4.3 (28) combine to give the proof that is sought (18), but only under the given assumption of a single maximum. This assumption is relaxed in Sections 4.4 and 4.5. In addition to the identification of

the same hysteresis loops (28), the hysteresis loops are also identified in the same order. The order of identification does not alter the estimated fatigue, however manipulating the input so that the hysteresis loops are identified in the same order facilitates the present proof.

The same reasoning can be used to show that the algorithms are equivalent when analysing a peak–valley sequence beginning and ending with the overall minimum.

4.4. Breaking the series at intermediate maximums

So far it has been shown that the two algorithms are equivalent when applied to an 'original' series of points with only one value equal to the overall maximum stress. For completeness, it still needs to be proven that the same result is produced when the algorithms are applied to a series of points with multiple stress values equal to the maximum. When rearranged, such a series will have a maximum on each end and a number of intermediate maximums. It will first be proven that the four-point algorithm output remains the same if the (rearranged) stress series is broken into separate sequences at each intermediate global maximum:

Property 4.4

$$f_{4p}(rearr(s[\,])) = \sum_i f_{4p}(break_i(rearr(s[\,]))). \tag{29}$$

Proof. When the stress series is broken at each intermediate maximum, the maximum appears at the end of one sequence and at the beginning of the next and the sequences are analysed separately. Each broken section consists of two 'end-point bounded sequences' (see Property 3.5) which remain unchanged. Thus, all hysteresis loops with a peak that is not the maximum stress will still be identified in the same way in the first pass of the four-point algorithm over each of the new sequences.

Once these hysteresis loops have been removed, there is one valley remaining in each broken section. One hysteresis loop is identified for each such valley (see Property 3.10). The only remaining peaks for them to be paired with are the maximums. Thus, no hysteresis loops are omitted or introduced because no valleys are omitted or added by *break*( ), and the loops that are identified must be of the same magnitude, so (29) holds. □

4.5. General equivalence after breaking series

When the stress series is broken as described in Section 4.4, the two algorithms again analyse the series and identify hysteresis loops in the same order because the two sets of criteria for identifying hysteresis loops are again equivalent:

Property 4.5

$$\sum_i f_{4p}(break_i(rearr(s[\,]))) = f_{3p}(rearr(s[\,])). \tag{30}$$

Proof. Once *break*( ) is applied, as described in Section 4.4, the argument used in Section 4.2 to show that $crit(3,n,s'[\,])$ and $crit(4,n,s'[\,])$ and the logic for applying them are equivalent holds without assumptions, as does Property 4.3. The only difference is that there can be multiple instances in which the four-point algorithm goes to the second pass at $n = 3$. When the next broken section is analysed by the four-point algorithm, the two algorithms again follow the same path starting with $n = 1$ and $s[1] = \max(s[\,])$.

Remark. By building on the proof presented in Sections 4.2 and 4.3 the necessity to assume a single maximum in the original unmodified series has been removed. Thus, the two algorithms are equivalent for any series of peaks and valleys.

5. Conclusion

The equivalence of the two algorithms has been demonstrated by showing that the input stress series can be modified without affecting the outcome of the four-point algorithm, and that after such modifications the two algorithms identify the same hysteresis loops in the same order. In particular, it has been shown that:

- The outcome of the four-point algorithm is not changed if the stress series is first rearranged as it is for the three-point algorithm.
- The two algorithms produce the same outcome by following similar paths when applied to such a rearranged stress series, assuming there are only two peaks equal to the maximum (there is always a minimum of two – the first and last stress value).
- If there are three or more stress values in the modified series equal to the maximum, the series can be broken at an intermediate maximum without changing the outcome of the four-point algorithm.
- When applied to such broken series, the four-point algorithm again follows a similar path and produces the same result as the three-point algorithm.

Thus, the same amount of estimated fatigue damage is calculated by the three-point and four-point algorithms because they both identify the same hysteresis loops. The only difference between the outcomes of the algorithms is the order in which the hysteresis loops are listed in the output.

Acknowledgement

The authors thank the Cooperative Research Centre for Mining and the Australian Coal Association Research Program (ACARP) for financially supporting this research.

Appendix A. Function definitions

The series of stress values $s[\,]$ is obtained from measured data the same way by both algorithms, by extracting the peaks and valleys. Hence, that part of the algorithm will not be defined. For the following definitions $s[\,]$ is represented as $s[1,\ldots N]$ where $s[1]$ is the first point (peak or valley), $s[N]$ the last point, and N is the number of points in the series. The overall maximum is $s[i_{\max}]$, where $i_{\max}$ is an index between 1 and N. The residue $res[\,]$ is also represented in this manner.

Function/ variable	Definition/description
$s[\,]$	the original unmodified series of peaks and valleys
$s'[\,]$	modified series of peaks and valleys
q	3 or 4, indicates which algorithm to use in $f(\,)$ and $crit(\,)$
n	integer, indicates stress values to be considered by $crit(\,)$
$res[\,]$	residue from first pass of four-point algorithm
$out[\,]$	output of each algorithm – a list of stress ranges
$rearr(s[1,\ldots,N])$	function that rearranges a stress series to begin and end with the maximum (Section 2.1) $s'[\,] = s[i_{\max},\ldots,N',p,\ldots,i_{\max}]$ where $N' = N$ or $N-1$ and $p = 1$ or 2, depending on whether a point is removed to maintain the max–min sequence return $s'[\,]$
$crit(q,n,s'[\,])$	function representing the three- and four-point criteria $X_n = \lvert s'[n] - s'[n-1]\rvert$ $X_{n-1} = \lvert s'[n-1] - s'[n-2]\rvert$ $X_{n-2} = X_{n-1} + 1$ if $q = 4$ $X_{n-2} = \lvert s'[n-2] - s'[n-3]\rvert$ if $X_n \geqslant X_{n-1}$ and $X_{n-2} \geqslant X_{n-1}$ return TRUE if $X_n < X_{n-1}$ or $X_{n-2} < X_{n-1}$ return FALSE
$f(q,s[\,])$	function representing core of three- and four-point algorithms $n = 0$ $j = 0$ for $i = 1,\ldots,$length$(s[\,])$ $n = n + 1$ $s'[n] = s[i]$ while $n \geqslant q$ and $crit(q,n,s'\,[\,])$ $j = j + 1$ $out[j] = X_{n-1}$ $s'[\,] = s'[1,\ldots,n-3,n]$ $n = n - 2$ return $res[\,] = s'[\,], out[\,]$
$f_{3p}(rearr(s[\,]))$	function representing the three-point algorithm, as described in Section 2.1 excluding $rearr(\,)$ and the step of estimating the fatigue damage $[res[\,], out[\,]] = f(3, rearr(s[\,]))$ return $out[\,]$
$f_{4p}(s[\,])$	function representing the four-point algorithm, as described in Section 2.2 excluding the step of estimating the fatigue damage $[res[\,], out[\,]] = f_{4p1}(s[\,])$ $out[\,] = f_{4p2}(res[\,], out[\,])$ return $out[\,]$
$f_{4p1}(s[\,])$	*function representing the first pass of the four-point algorithm* $[res[\,], out[\,]] = f(4, s[\,])$ return $res[\,]$, $out[\,]$
$f_{4p2}(res[\,], out[\,])$	function representing the second pass of the four-point algorithm, including treatment of the residue $[s'[\,], \mathrm{out}_2[\,]] = f_{4p1}(dbl(res[\,]))$ return $(out[\,] + \mathrm{out}_2[\,])$ The addition of stress range vectors $out[\,] = out[\,] + out_2[\,]$ refers to the combination of the two sequences into one longer sequence, not the addition of the elements
$dbl(res[\,])$	function representing the treatment of residue $s'[\,] = res[1,\ldots,N',p,\ldots,N]$ where $N' = N$ or $N-1$ and $p = 1$ or 2, depending on whether a point is removed to maintain the peak–valley sequence return $s'[\,]$
$f_{4p\alpha}(s[\,])$	alternative algorithm to $f_{4p1}(\,)$ (see Property 3.3) $s'[\,] = s[\,]$ $j = 0$ while $crit(4,n,s'[\,]) =$ TRUE for any n s.t. $4 \leqslant n \leqslant length(s'[\,])$ $n =$ random$(4,\ldots,$length$(s'[\,]))$ if $crit(4,n,s'[\,])$ $j = j + 1$ $out[j] = X_{n-1}$ $s'[\,] = s'[1,\ldots,n-3,n,\ldots,$length$(s'[\,])]$ $n = n - 2$ return $res[\,] = s'[\,], out[\,]$
$break(rearr(s[\,]))$	function that breaks the stress series at each intermediate maximum (used in Section 4.4) $s'[\,] = rearr(s[\,])$ $s'_{\max} = s'[1]$ $start = 1$ $n = 1$ while $n <$ length$(s'[\,])$ $n = n + 1$ if $s'[n] = s_{\max}$ return s'[start,…,n] start $= n$

References

[1] Weibull W. Fatigue testing and analysis of results. Pergamon Press; 1961.

[2] Bannantine JA, Comer JJ, Handrock JL. Fundamentals of metal fatigue analysis. Englewood Cliffs (NJ): Prentice-Hall; 1990.

[3] Matsuishi M, Endo T. Fatigue of metals subjected to varying stress. Presented to Japan Society of Mechanical Engineers, Fukuoka, Japan, 1968.

[4] Amzallag C, Gerey JP, Robert JL, Bahuaud J. Standardization of the rainflow counting method for fatigue analysis. Int J Fatigue 1994;16:287–93.

[5] Dowling NE. Fatigue failure predictions for complicated stress–strain histories. J Mater 1972;7(1):71–87.

[6] Rychlik I. A new definition of the rainflow cycle counting method. Int J Fatigue 1987;9(2):119–21.

[7] Downing SD, Socie DF. Simple rainflow counting algorithms. Int J Fatigue 1982;4(1):31–40.