# A fast algorithm to racetrack filter multiaxial histories preserving load shape

*Marco Antonio* Meggiolaro[1,1], *Jaime Tupiassú* Pinho de Castro[1] , and *Hao* Wu[2]

[1]Pontifical Catholic University of Rio de Janeiro, PUC-Rio - Brazil
[2]School of Aerospace Engineering and Applied Mechanics Tongji University - China

**Abstract.** A fast algorithm is presented to implement a multiaxial version of the racetrack amplitude filter, improved upon the authors' previous works. The algorithm only requires a single user-defined scalar filter amplitude to eliminate noise and potentially non-damaging events from arbitrary non-proportional multiaxial stress or strain histories, while preserving all key features of the load path. It can be applied as a pre-processing step to both invariant-based and critical-plane multiaxial damage models, highly decreasing the computational cost in assessments involving oversampled or noisy data, much improving the usefulness of computationally-intensive multiaxial models. Since load path shape is well preserved (within the chosen filter resolution), peaks and valleys from subsequent candidate plane projections are not filtered out. Moreover, path-equivalent stress or strain calculations can still be applied after the filtering process, without precision loss beyond the filter resolution. The algorithm is synchronous and preserves load order, becoming suitable for strain-based multiaxial damage models involving plastic memory, as well as for mixed-mode crack propagation calculations. The filter efficiency is validated from oversampled tension-torsion experimental data in 316L stainless steel tubular specimens, following complex non-proportional histories. Finally, a fully-tested computer implementation of the algorithm is presented.

## 1 Introduction

To properly measure service load signals and reproduce them in fatigue tests, it is necessary to oversample the digitized data at a rate high enough not to distort the signal [1, 2]. This is particularly important in fatigue analyses, where (at least under uniaxial loads) the important features are the peaks and valleys of the load, which are modified if the sampling frequency is too low. These peaks and valleys can be distorted as well if the resulting signal is frequency-filtered, for that matter. That is why noisy or oversampled signals must be amplitude-filtered instead for fatigue applications, using e.g. rainflow techniques. This problem is much magnified in multiaxial fatigue analyses, where such oversampled or noisy data simply can have a too high computational cost if the load history is not properly

[1] Corresponding author: meggi@puc-rio.br

filtered. Therefore, like in the one-dimensional case, it may seem a good idea to first remove from the multiaxial variable amplitude loading (VAL) history all data points that are not peaks or valleys of any of their stress or strain components.

However, this simple filtering practice *cannot* be used in non-proportional (NP) multiaxial analyses, for two reasons: first, the path between two load reversals is needed to evaluate the path-equivalent stress or strain ranges associated with each rainflow count, e.g. using a convex-enclosure method [3] or an integral-averaging procedure such as the Moment Of Inertia (MOI) method [4]. Filtering out too many points in the path almost certainly results in lowering the equivalent stresses or strains; thus some points along the path should not be filtered out, even if they do not constitute a load component reversal.

The second reason not to filter out points that are not peaks and valleys of individual load components is because the reversal points obtained from a multiaxial rainflow algorithm might not occur at the reversal of one of the stress or strain components. For example, the relative von Mises strain, used in the Wang-Brown [5] and Modified Wang Brown (MWB) [6] rainflow counts, may reach a peak value at a point that is neither a maximum nor a minimum of any strain component. Nevertheless, these most important loading points would be removed by a non-reversal filtering algorithm, resulting thus in *non*-conservative fatigue damage and life predictions.

For instance, the “Peaks Procedure” proposed in [1] filters out all events which components are not peaks or valleys, potentially eliminating important load points that could have the highest von Mises stresses or strains of the load history, even though each individual component was not maximized. Moreover, this “Peaks Procedure” stores each and every event that constitutes a peak or valley from any single component, which for noisy measurements could result in no events at all being filtered out.

To avoid these issues, some metric (such as the von Mises stress or strain) must be used to evaluate how much a measured multiaxial VAL path deviates from its course. This is needed to avoid filtering out important counting points for multiaxial rainflow algorithms, or significant paths that can affect the calculation of a path-equivalent stress or strain range, since all stress or strain components contribute altogether for the reversals that can be eliminated. A true multiaxial *amplitude* filter is therefore a most desirable step in practical applications, to eliminate unavoidable measurement noise and redundant oversampled data, as well as small amplitudes that do not cause fatigue damage.

In fact, it could be too costly or even practically impossible to analyze unfiltered multiaxial fatigue data. Despite the high speed of today’s computers, fatigue damage calculations can have an intrinsically high computational cost, e.g. in the analysis of oversampled and/or noisy data for very long multiaxial load histories using a critical plane search [1], or for damage assessments in multiple candidate critical points of a structure.

The racetrack amplitude filter, originally proposed in [7] for uniaxial histories, aims to eliminate from a VAL history small amplitude events that do not induce fatigue damage. The resulting condensed histories can accelerate both experiments and computations, focusing only on the few significant events that cause most or all of the damage.

A multiaxial version of the racetrack filter must perform these tasks even for NP histories. In the multiaxial racetrack filter (MRF) algorithm proposed in [8], the stress or strain history is represented in general in a 6D space, and a suitable scalar filter amplitude $r$ must then be chosen, like in the 1D case. This MRF algorithm was inspired on incremental plasticity calculations, where stress history variations within the yield surface are considered purely elastic without plasticity, while elastoplastic events cause the yield surface to translate in the 5D deviatoric stress space [9]. Analogously, in the MRF from [8], the load history is filtered out while its variations are within a filtering surface (with radius equal to the user-defined $r$), otherwise events beyond it cause the filtering surface to translate in the 6D stress (or strain) space. The MRF highly reduces the number of points in oversampled and/or noisy NP histories without losing information on the load history shape and order, as verified in [10].

Ideally, the MRF amplitude $r$ should be chosen based on the fatigue limit of the studied component. However, the fatigue limit is not only a function of the load amplitude, but also of the mean components. Thus, to incorporate mean or maximum stress effects in the MRF, a variable filter amplitude was adopted in [11], which depends on the current stress level along the stress or strain path. This way, a small amplitude event can be filtered out if associated with a non-damaging low mean or peak stress level, while another event with the very same stress or strain amplitude can be preserved if happening under a more damaging high mean or peak stress level. The variable value of the filter amplitude is calculated in real time along the multiaxial load path. Furthermore, an improvement of the computational speed of the MRF is obtained in [12] with the proposal of a pre-processing step for the load history data, which selects key points between which the filter surface translations are applied (instead of applying the translations to all sampled points).

Despite their associated reductions in the computational cost for fatigue damage assessments, all variations of the MRF algorithm [8, 11, 12] still require the definition and translation of multi-dimensional filter surfaces in computer calculations. To further reduce the computational cost, in this work the MRF algorithm is modified to eliminate the need of such multi-dimensional surfaces and translations. The MRF algorithm proposed here highly reduces calculation times and computer requirements, even allowing its implementation in inexpensive micro-processors for use in field applications for data acquisition.

## 2 Stress and strain spaces

The MRF must be applied to the multiaxial stress or strain history on a user-defined multi-dimensional space. For a general 6D history, the adopted space for the MRF can be e.g. defined by the normal and effective shear components, $\left[\sigma_x \;\; \sigma_y \;\; \sigma_z \;\; \tau_{xy}\sqrt{3} \;\; \tau_{xz}\sqrt{3} \;\; \tau_{yz}\sqrt{3}\right]^T$ or $\left[\varepsilon_x \;\; \varepsilon_y \;\; \varepsilon_z \;\; \gamma_{xy}/\sqrt{3} \;\; \gamma_{xz}/\sqrt{3} \;\; \gamma_{yz}/\sqrt{3}\right]^T$, respectively for stress or strain histories. Alternatively, a given stress history could be represented in a 6D stress space composed of a 5D deviatoric space such as

$$\vec{s}' \equiv \left[\sigma_x - (\sigma_y + \sigma_z)/2 \;\; (\sigma_y - \sigma_z)\sqrt{3}/2 \;\; \tau_{xy}\sqrt{3} \;\; \tau_{xz}\sqrt{3} \;\; \tau_{yz}\sqrt{3}\right]^T \tag{1}$$

with the 6th dimension equal or proportional to the hydrostatic stress $\sigma_h = (\sigma_x + \sigma_y + \sigma_z)/3$. Analogously, a given strain history could be represented in a 6D space combining the hydrostatic strain $\varepsilon_h$ and a 5D deviatoric strain vector such as

$$\vec{e}' \equiv \left[\varepsilon_x - (\varepsilon_y + \varepsilon_z)/2 \;\; (\varepsilon_y - \varepsilon_z)\sqrt{3}/2 \;\; \gamma_{xy}\sqrt{3}/2 \;\; \gamma_{xz}\sqrt{3}/2 \;\; \gamma_{yz}\sqrt{3}/2\right]^T \tag{2}$$

Note that the MRF algorithm can be used not only to synchronously filter out stress or strain histories, either in 6D or lower dimensional stress, strain, deviatoric stress, or deviatoric strain spaces, but also to filter any history of multi-dimensional quantities such as forces, moments, and/or displacements acting at different points of a structure [10]. Therefore, the MRF has several practical applications in engineering besides multiaxial fatigue, e.g. in Finite Element (FE) simulations of structures (such as automobiles or aircrafts) subject to variable amplitude force/moment inputs on several points. Filtering out the negligible events from the input history highly reduces the simulation time, but the FE analyses must not lose the synchronicity of the several load sources. These desirable conditions, met by the proposed MRF, would not be possible with unidimensional amplitude filters individually applied to each load source.

## 3 Multiaxial racetrack filter algorithm

The MRF version proposed in this work is performed in two steps. First, a partitioning of the multiaxial load history data must be performed, based on the process introduced in [12], described as follows.

Initially, the first sample point of the load history, described in one of the spaces listed in the previous section, is tagged with the number 1, while the last sample point becomes point 2. For periodic histories, point 1 can be any one from the cyclic load path, while point 2 is obtained as the one most distant to point 1 in a Euclidean sense, see Fig. 1(a). If more than one point has the same maximum distance from 1, then choose any one of them to become point 2, ignoring the others. Two partitions of the original sample points are then obtained, defined by the paths {1→2} and {2→1}.

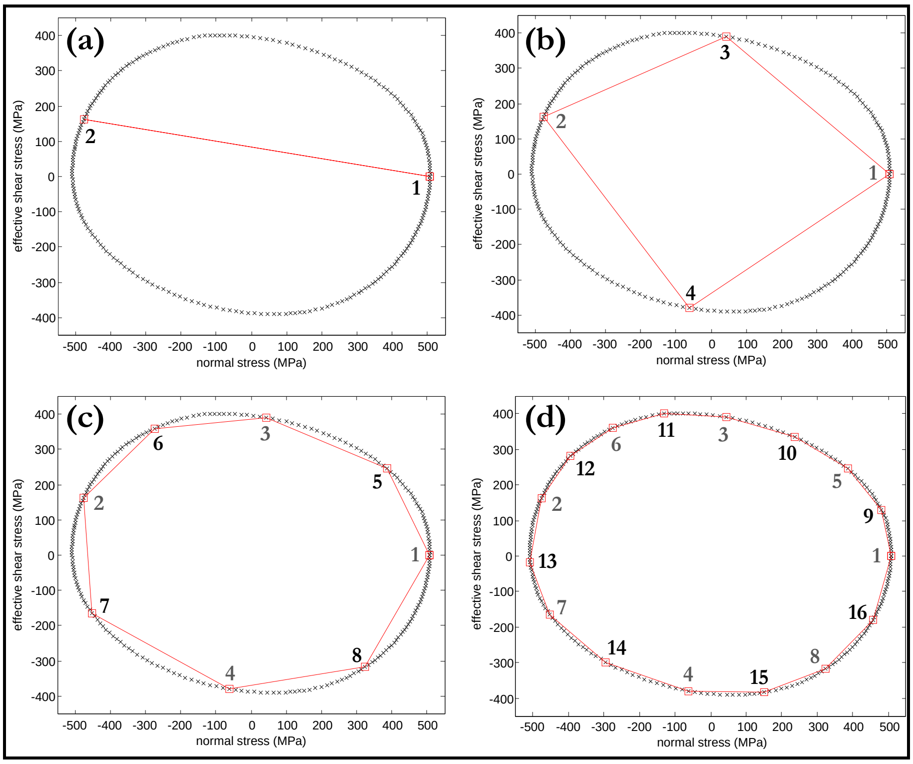

**Fig. 1.** Partitioning operation on tension-torsion sampled data [12].

For the tension-torsion example in the $\sigma_x \times \tau_{xy}\sqrt{3}$ space from this figure, note that this Euclidean distance would become the relative von Mises stress range between points 1 and 2, defined as $[(\Delta\sigma)^2 + (\Delta\tau_{xy}\sqrt{3})^2]^{1/2} = [\Delta\sigma^2 + 3\Delta\tau_{xy}{}^2]^{1/2}$. The use of deviatoric spaces in the load history representation, such as the ones in Eq. (1) and (2), is able to provide a physical meaning of the MRF amplitude, becoming a relative von Mises stress or strain range.

Then, points 3 and 4 are obtained as the ones most distant from the straight line joining points 1 and 2. Note that point 3 lies in the 1→2 path whereas point 4 lies in the 2→1, as shown in Fig. 1(b). If these maximum distances are larger than the chosen filter amplitude *r*, then further partitions {1→3}, {3→2}, {2→4} and {4→1} of the original path are created, see Fig. 1(b).

If the sample point in the {1→3} partition most distant from the straight line joining 1 and 3 is higher than *r*, then two additional partitions {1→5} and {5→3} are created from

{1→3}, see Fig. 1(c). The process continues for the other partitions {3→2}, {2→4} and {4→1}, generating additional partitions as long as the associated maximum distances are higher than *r*. In the presented tension-torsion example, this step marks points 5, 6, 7 and 8 among the sample history points, see Fig. 1(c).

This process continues for all created partitions, ending when all such maximum distances between sample points and straight segments are not higher than the chosen filter amplitude *r*. In this example, if the chosen *r* is 5MPa, then these partitioning steps result in Fig. 1(d), with 16 of the original sample points marked from 1 to 16. These marked points (from the extremes of each partition) are called here *key points*, which are not filtered out in the MRF. Note, however, that the remaining non-marked points are not filtered out yet, since they can contain load reversals along the path between consecutive key points, see Fig. 2. Note as well in Fig. 2 that, even though all sample points from this partition {A→F} are close to the segment AF (within the filter amplitude *r*), there are significant oscillations and reversals parallel to AF.

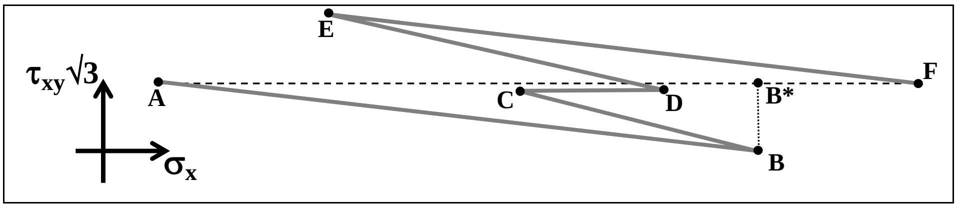

**Fig. 2.** Partition example {A→F} containing sample points A→B→C→D→E→F from a multiaxial load history.

After the entire partitioning process ends, each resulting partition is individually analyzed in the search of load reversals within it. For each analysis, all points from a certain partition must be orthogonally projected onto the associated straight segment joining its extreme key points. This can be seen in the example from Fig. 2, containing sample points A→B→C→D→E→F from the partition {A→F}, where A and F are the key points, and B* is a perpendicular projection of a sample point B onto segment AF. The scalar coordinate *p* of each projection B* along the straight segment starting from the first point of its partition (A in this example) is then computed, to be compared to the distance *q* between the partition extremes (A and F in this example). As a result, each and every point B of the partition becomes associated with a scalar *p* (equal to the signed distance AB* in the Fig. 2 example). If A and F are the vector coordinates of the partition key points in the history space, and B* is the vector coordinate of the projection of a sample point B from the partition onto the straight line AF, then

$$p = (B-A)\cdot(F-A) / |F-A| = (B-A)\cdot n \tag{3}$$

where $n = (F-A) / q$ is the unit vector along AF.

Clearly, the value of *p* for the first point of the partition (A in the example) is 0, while for the last point (F) it is the distance *q* (between A and F). For any other sample point B, the value of *p* can result in any real number: between 0 and *q* if the projection B* is within the segment AF (as in Fig. 2, the most usual case in practice), larger than *q* if it is beyond F, or negative if it is before A.

Then, a uniaxial racetrack procedure is applied to the scalar values *p* of all points from each partition, using the same filter amplitude *r* from the previous step. This uniaxial procedure – the second step of the proposed MRF algorithm – can be easily described through a physical analogy: a small round peg P oscillating inside a slotted plate whose center is the point O, see Fig. 3.

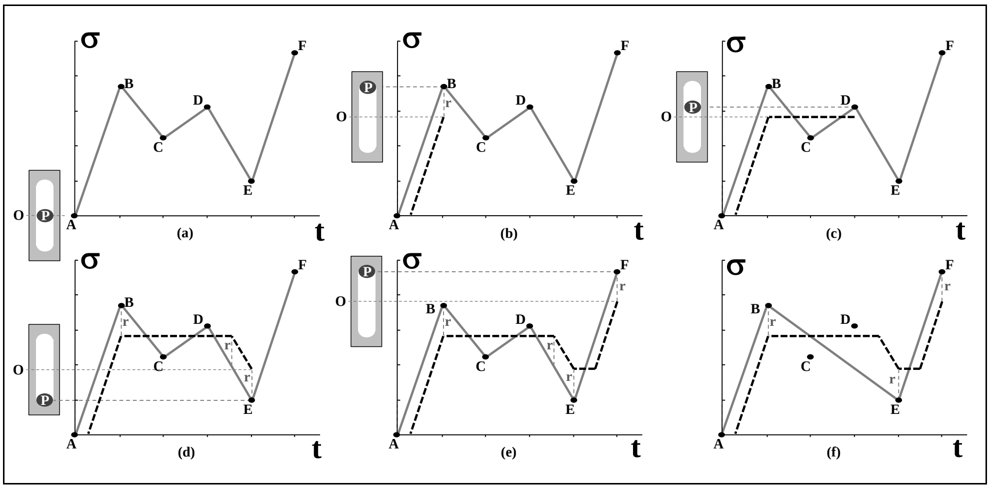

**Fig. 3.** Uniaxial racetrack procedure over the signed distances *p* from a stress history, using a physical analogy with a peg P oscillating inside a slotted plate with center O and slot range *2r*.

The range that the peg can oscillate inside the slot is set as twice the filter amplitude, i.e. *2r*. For Fig. 2, a partition {A→F} with sample points A→B→C→D→E→F, their associated *p* values are represented on the *vertical* axis of the graphs from Fig. 3 (the *horizontal* axis in this figure is just a representation of sample number, or time). As seen in Fig. 3(a), both peg and slot center are initially aligned with point A. During the path AB, the peg moves up until reaching the upper limit of the slotted plate, which then starts to move up, see Fig. 3(b), where the dark dashed line represents the path of point O. As shown in Fig. 3(c), the path BCD does not involve any translation of the slotted plate, an indication that the points C and D will be filtered out. Then, both paths DE and EF involve slotted-plate translations, see Fig. 3(d) and (e).

Besides the initial point A, only the peg locations P associated with the *end* of a translation of the slotted plate are stored in the filtered history. In the Fig. 3 example, Point B is the peg location at the end of the first upward translation of the slotted plate, so it is stored. Similarly, points E and F are stored since they are the peg locations at the end of the next downward and upward slotted plate translations. The filtered history from this partition is then the path ABEF, as shown in Fig. 3(f). This amplitude filter is very efficient in practice since, besides small non-damaging load events, it can remove the unavoidable noise e.g. from strain measurements, which can induce much more apparent reversals than the load itself in the measured signal. It is in fact an almost indispensable tool for practical fatigue analyses.

This process involving projections and uniaxial racetrack filtering is then repeated for all previously obtained load history partitions. The resulting points that are *not* filtered out are the key points from the partitioning step, plus the points associated with the *end* of a translation of the slotted plate in the uniaxial racetrack procedure of each partition. All other points are eliminated from the load history, significantly reducing the memory and processing requirements for oversampled data.

From the definition of the presented algorithm it can be said that, in the partitioning step, no sample point oscillates more than an amplitude *r* perpendicularly to its partition segment. Moreover, after the uniaxial racetrack step, all filtered points are the ones associated with projected oscillations parallel to its segment with amplitudes smaller than *r*. As a result, the highest amplitude that is filtered out in the proposed MRF is lower than $r\sqrt{2}$, calculated from the maximum amplitudes *r* perpendicular and *r* parallel to the partition segment. So, if it is desired to filter out all amplitudes higher than some value $a$, then the filter amplitude should be chosen in the proposed MRF as $r = a / \sqrt{2}$.

A fully functional computer implementation of the proposed MRF is shown in the Appendix, programmed in the Matlab© environment. This code includes both partitioning and uniaxial racetrack steps, where the original load history is represented by an $(N \times m)$ input matrix $P$ that stores $N$ sample points of $m$ dimensions each; the output matrix *P_out* gives the resulting filtered history. This code can be easily translated to other computer languages such as Python, which shares several similarities with the Matlab© syntax.

## 4 Experimental verification

To evaluate the proposed MRF, non-proportional (NP) tension-torsion experiments are performed on annealed tubular 316L stainless steel specimens in a multiaxial testing machine. Thin 2mm walls are used in the specimens to avoid significant stress gradient effects across their thickness. Engineering stresses and strains are calculated from load/torque cell and from axial/torsional extensometer measurements, converted later to true stresses and strains.

Strain-controlled $\varepsilon_x \times \gamma_{xy} / \sqrt{3}$ tension-torsion cycles are applied to the tubular specimens, following a challenging cyclic path originally proposed by Socie [13]. Figure 4 shows the resulting experimentally measured stabilized $\sigma \times \tau\sqrt{3}$ stress path (oversampled, with samples using × markers), as well as the MRF output (square markers) for a chosen filter amplitude $r = 20MPa$. The measured $\sigma \times \tau\sqrt{3}$ stress paths are reduced from *1227* data points ***per cycle*** to only *46*, showing a very good agreement in both ranges and shape. Even for low-cycle assessments this experiment would involve the acquisition of several million data points (*1227* per cycle) if not filtered.

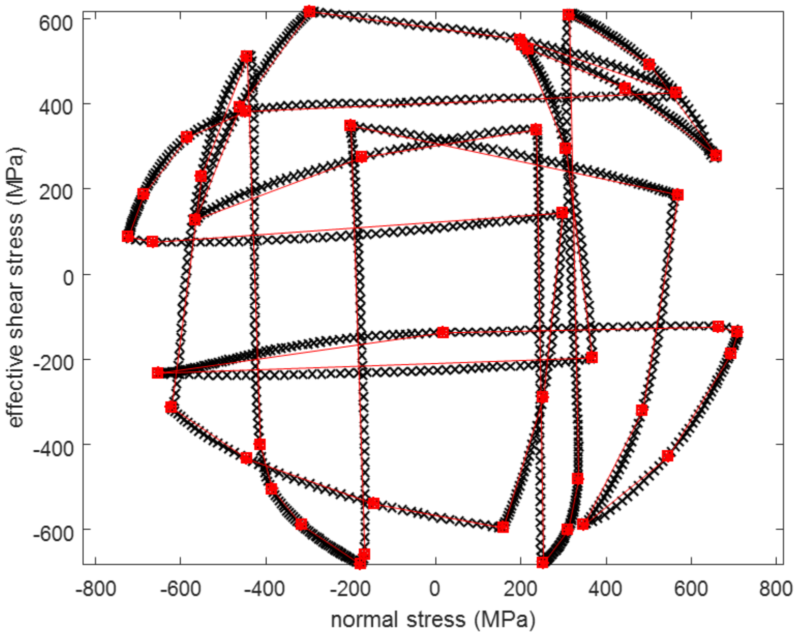

**Fig. 4.** Experimentally measured stress data points (× markers) and MRF output (square markers) for a chosen filter amplitude $r = 20MPa$.

## 5 Conclusions

Amplitude filters are simply indispensable in practical fatigue analyses to manage their computational cost without compromising accuracy when (as usual) the input load history is noisy, oversampled, too long, and/or contains too many non-damaging low-amplitude cycles. In this work, a fast multiaxial racetrack filter (MRF) algorithm was presented, which is an amplitude filter applicable to general non-proportional multiaxial histories. It allows the synchronous filtering of stress and strain histories, preserving load order and path shape, without filtering out load reversal points. The algorithm does not need the definition, calculation and translation of multi-dimensional filter surfaces, required in the previous versions proposed by the authors, highly decreasing computational cost both in the filtering itself and in the subsequent multiaxial damage calculations with reduced sample points.

## References

**1.** J.A. Bannantine. A Variable Amplitude Multiaxial Fatigue Life Prediction Method. FCP Report 151, University of Illinois at Urbana-Champaign (1989).

**2.** T.E. Langlais, J.H. Vogel, T.R. Chase. Multiaxial cycle counting for critical plane methods. Int J Fatigue, **25**:641-647 (2003).

**3.** M.A. Meggiolaro, J.T.P. Castro. An improved multiaxial rainflow algorithm for non-proportional stress or strain histories - part I: enclosing surface methods. Int J Fatigue, **42**: 217-226 (2012).

**4.** H. Wu, M.A. Meggiolaro, J.T.P. Castro. Application of the Moment Of Inertia Method to the Critical-Plane Approach. Fratt Int Strutturale, **38**:99-105 (2016).

**5.** C.H. Wang, M.W. Brown. Life prediction techniques for variable amplitude multiaxial fatigue - part 1: theories. J Eng Mater Technology, **118**:367-370 (1996).

**6.** M.A. Meggiolaro, J.T.P. Castro. An improved multiaxial rainflow algorithm for non-proportional stress or strain histories - part II: the modified Wang-Brown method. Int J Fatigue, **42**:194-206 (2012).

**7.** H.O. Fuchs, D.V. Nelson, M.A. Burke, T.L. Toomay. Shortcuts in Cumulative Damage Analysis. SAE Automobile Engineering Meeting, Paper 730565 (1973).

**8.** M.A. Meggiolaro, J.T.P. Castro, H. Wu. Shortcuts in multiple dimensions: the multiaxial racetrack filter. Fratt Int Strutturale, **33**:368-375 (2015).

**9.** M.A. Meggiolaro, J.T.P. Castro, H. Wu. A general class of non-linear kinematic models to predict mean stress relaxation and multiaxial ratcheting in fatigue problems – Part I: Ilyushin spaces. Int J Fatigue, **82**:158-166 (2016).

**10.** H. Wu, M.A. Meggiolaro, J.T.P. Castro. Validation of the Multiaxial Racetrack Amplitude Filter. Int J Fatigue, **87**:167-179 (2016).

**11.** M.A. Meggiolaro, J.T.P. Castro, H. Wu. Incorporation of Mean/Maximum Stress Effects in the Multiaxial Racetrack Filter. Fratt Int Strutturale, **38**:67-75 (2016).

**12.** M.A. Meggiolaro, J.T.P. Castro, H. Wu. A two-step multiaxial racetrack filter algorithm for non-proportional load histories. Fratt Int Strutturale, **41**:1-7 (2017).

**13.** D.F. Socie, G.B. Marquis. *Multiaxial Fatigue*. SAE (2000).

## Appendix: Matlab© implementation of the proposed MRF

```
r = 20; r2 = r*r; %chosen filter amplitude r and its square value r2

%load history is an (N x m) input matrix P with N sample points of m dimensions
N = size(P,1);

%partitioning step
index = [ ]; %initializes indices of the already selected partitioning key points
new_index = [1 N]; %same indices, but for the next iteration of the partitioning algorithm
while (size(new_index,2) ~= size(index,2)) %new index vector has increased over old one
   index = new_index; new_index = [1]; %updates old index vector and prepares a new one
   for j = 2:size(index,2) %sweeps all current partitions
      d2_max = 0; %initialization of the square of the maximum distance to the segment
      q = P(index(j),:)–P(index(j–1),:); %vector q between key points of the partition
      if (norm(q)>0) n = q/norm(q); end; %calculates normalized vector n between key points
      for i = (index(j–1)+1):(index(j)–1) %sweep all indices inside this partition
         p = P(i,:)–P(index(j–1),:); %vector between sample point i and last key point
         d2 = p*p' – (n*p')^2; %square of distance d from sample point i to segment
         if (d2 > d2_max) d2_max = d2; i_max = i; end; %point i is more distant to segment
      end
      if (d2_max > r2) new_index = [new_index i_max]; end; %adds a partition
      new_index = [new_index index(j)]; %updates next index
   end
end

%uniaxial racetrack step
index = new_index; new_index = [ ]; %updates old index vector and prepares a new one
for j = 2:size(index,2) %sweeps through the current index vector
   q = P(index(j),:)–P(index(j–1),:); %vector q between key points of the partition
   if (norm(q)>0) %checks whether key points do not coincide
      n = q/norm(q); %normalized vector between key points in the translation direction
      nt = 0; %initial translation direction of slotted plate is unknown
      i_last = index(j–1); %last translation point is initialized as the partition's first
      Oc = 0; %initial values of the signed distance p and slotted plate center are zero
      for i = (index(j–1)+1):(index(j)) %indices inside this partition
         p = (P(i,:)–P(index(j–1),:))*n'; %signed distance p to the next projection, see Eq.(3)
         d = (p – Oc); %signed distance between slotted plate center and the next projection
         if (d*d > r2)  %next point causes a slotted plate translation
            if (d*nt<=0) %slotted plate translation direction has reversed
               new_index = [new_index i_last];
               nt = sign(d); %reverse translation direction through its normalized vector
            end
            Oc = Oc + (abs(d)–r)*nt; %translation of the slotted plate center
            i_last = i;
         end
      end
      if (i_last < index(j)) new_index = [new_index index(j)]; end %adds last partition point
   end
end
new_index = [new_index i_last]; %last point of the entire history is included in the output

P_out = P(new_index,:); %obtains the output history P_out from the chosen indices from P
```