Differentiable Fatigue Damage Modeling for Neural-Network-Based Virtual Sensing

Katharina Conrad

Master's thesis
2026:E79

Lund University

Faculty of Science
Centre for Mathematical Sciences
Numerical Analysis

Centrum Scientiarum Mathematicarum

Differentiable Fatigue Damage Modeling for Neural-Network-Based Virtual Sensing

LUND
UNIVERSITY

written by

Katharina Conrad

Field of Studies: Numerical Analysis

Supervision: Dr. Mengwu Guo (Lund University), Ruben Schwiedernoch (Mercedes–Benz) and Dr. David Neuhäuser (Mercedes–Benz)

Abstract

Virtual sensors implemented in vehicles enable the prediction of forces experienced during their operational lifetime and thus allow an extension of durability assessment into the operational phase of customer vehicles.

Due to limited computational capacity of onboard electronic control units, full-scale deep learning models are often too expensive for real-time operation. As a result, compact neural network architectures are employed. However, their limited model capacity prevents them from capturing all relevant characteristics of the underlying load history, leading to significant deviations in fatigue damage estimation based on Rainflow Counting compared to the true damage.

To address this limitation, this thesis proposes a fatigue damage surrogate for Rainflow Counting based on the spectral method by Dirlik. The underlying spectral estimates, in particular the periodogram and the Welch method, are investigated with respect to their influence on the resulting damage, their computational efficiency, and their differentiability. Ensuring differentiability down to the spectral estimation level enables the integration of the damage surrogate as an additional loss term within the deep learning training process.

It is shown that the physics-informed Dirlik formulation produces extremely small gradients, introducing a bottleneck in the training process due to vanishing-gradient-like behaviour. By adapting the loss function, this issue is mitigated through effective gradient rescaling, allowing stable optimization.

The results demonstrate that training with a multitask-like loss improves the prediction of fatigue damage during inference. These findings provide a framework for enhancing virtual sensor predictions with respect to fatigue damage assessment.

Popular Scientific Summary

Fatigue damage is caused by repeated stress that vehicle components experience throughout their operational lifetime. For example, when a car drives over uneven road surfaces such as bumps or potholes, small deformations occur in certain components. Over time, these repeated stresses accumulate and lead to material damage.

To monitor this process and improve vehicle design, virtual sensors are used to estimate the loads acting on a vehicle during real operation. These sensors are based on machine learning models, which are trained to predict the loads in real-time by minimizing the difference between measured (true) load and model predictions. During training, the model iteratively updates its parameters to improve this prediction.

However, in practical applications these models must run directly inside the vehicle, where computational resources are limited. Therefore, only relatively small neural network architectures can be used. Such models often fail to capture extreme events, for example large stress peaks caused by driving over deep potholes. Although these events occur less frequently, they have a significant impact on fatigue damage.

Fatigue damage is commonly computed using so-called counting methods, which identify and evaluate peaks and valleys in the load-time history. When extreme values are incorrectly predicted, these methods produce inaccurate damage estimates. To address this issue, fatigue damage is incorporated directly into the training objective of the model.

Since traditional counting methods are not differentiable and therefore cannot be used within gradient-based optimization, this thesis replaces them with a spectral fatigue damage model proposed by Dirlik. Spectral methods are based on the power spectral density of a signal and, as shown in this work, can be formulated in a differentiable way. By ensuring differentiability of the entire method, including the spectral estimation step, the damage can be integrated directly into the training process.

However, the gradients resulting from the Dirlik formulation are extremely small, which can prevent effective learning. This issue is addressed by adapting the loss function, allowing these gradients to be rescaled and enabling stable training.

The results show that incorporating fatigue damage into training improves the model's ability to predict damage. In particular, the model becomes better at capturing extreme events, reducing the mismatch between predicted and actual damage. A remaining challenge is that different load histories can result in similar damage values, which may lead to variability in training results and requires further investigation.

Acknowledgements

I would like to express my gratitude to my university supervisor, Dr. Mengwu Guo, for his guidance and support throughout the development of this thesis, as well as for his detailed feedback on my drafts. I also thank him for supporting my decision to conduct this thesis in cooperation with Mercedes-Benz in Germany and for agreeing to a largely online supervision, which enabled me to work on an applied topic in the field of operational durability.

In this context, I would also like to thank the director of studies for the programme of Mathematics at Lund University, Anna-Maria Persson, for approving my request to conduct this thesis outside the usual university setting and abroad.

Furthermore, this endeavour would not have been possible without my supervisors at Mercedes-Benz, Ruben Schwiedernoch and Dr. David Neuhäuser, who offered me the opportunity to work as a master's student within their project on virtual sensors in vehicles. Their continuous support, technical guidance, and valuable feedback enabled me to gain the necessary knowledge in fatigue damage assessment and durability analysis in automotive applications.

Lastly, I would like to thank my parents for their continuous support throughout my academic journey, enabling me to pursue my interests across different places and stages, whether in Munich, Sweden, or Germany. I am also grateful to my friends for encouraging me to follow both my academic and personal goals.

I acknowledge the use of AI-based tools for code debugging and minor language refinement. These tools were used as supportive aids only; all scientific content, analyses, and conclusions were developed and verified independently.

Contents

List of Symbols and Abbreviations

Symbols

$a^{(k)}$	pre-activation values at k-th layer
α	weighting parameter
α_2	bandwidth parameter, irregularity factor
α_n	bandwidth parameter of order n
b	weighting parameter for Tovo-Benasciutti damage rate
$b^{(k)}$	bias at k-th layer
b_{app}	approximate weighting parameter for Tovo-Benasciutti damage rate in dependence on irregularity factor α_2
β	weighting parameter
C_1	coefficient of relative contribution of exponential term of rainflow amplitude probability density function
C_2	scaling factor of Rayleigh-type terms of rainflow amplitude probability density function
C_3	scaling factor of Rayleigh-type terms of rainflow amplitude probability density function
D	(fatigue) damage
D_{DK}	Dirlik damage rate
D_{NB}	narrow-band approximation of rainflow damage
D_{RC}	approximate range-mean damage
D_{TB}	Tovo-Benasciutti damage rate
$\bar{D}_{DK}(T)$	accumulated Dirlik damage over a time interval T
$\bar{D}_{TB}(T)$	accumulated Tovo-Benasciutti damage over a time interval T
Δs	stress range
f	(discrete-time) frequency: $f = \omega/2\pi$

$f(x, \theta)$	prediction of Neural Network (NN)
$\Gamma(\cdot)$	gamma function
$h^{(k)}$	k-th hidden layer
J	Jacobian
$\rho(J)$	spectral radius of Jacobian J
K	value of overlap
L_1	loss function one (e.g. MSE, MAE)
L_2	loss function two (e.g. MSE, MAE)
$\mathcal{L}_D$	loss of fatigue damage
$\mathcal{L}_{total}$	combined total loss
$\mathcal{L}_y$	loss of original model (without additional loss)
M	length of data segments of windowed periodograms
m	slope exponent of S-N curve
$\mu(J)$	singular values of Jacobian J
n	order of spectral moment
n_l	number of cycles with stress range l
N	signal/data length in spectral estimation
$N(s)$	expected number of stress cycles
$N(s_{a,l})$	number of cycles to failure corresponding to $s_{a,l}$
N_D	number of cycles at endurance limit s_D
$p(s)$	probability density function
$p_{RFC}^{DK}(s)$	rainflow amplitude probability density function
P	power of temporal window
Q	function of irregularity factor α_2
r	stress ratio
R	function of irregularity factor α_2
$\bar{R}$	residuum
s	stress

s_a	stress amplitude
$s_{a,l}$	stress amplitude corresponding to l
s_D	endurance limit
s_l	minimum stress (lower stress)
s_m	mean stress
s_u	maximum stress (upper stress)
S	number of data segments for windowed periodograms
$S(l)$	frequency of occurrence of counted cycles with span l
$SP(l)$	spanning pairs corresponding to span l
σ^2	variance
σ_X^2	variance of the process X
t	time
Θ	parameter space of NN
θ	weights and biases
$\tilde{y}_j(t)$	windowed zero-mean data segments
$y(t)$	signal/time-history value
$y_j(t)$	windowed data segments
y	ground truth, target data
$\hat{\Phi}_j$	windowed periodograms
$\hat{\Phi}_p$	periodogram definition of PSD
$\hat{\Phi}_W$	averaged windowed periodograms (Welch)
Φ	power spectral density
$\varphi(i\omega)$	discrete Fourier transform with window
ψ	activation function
ν_0	rate of mean upcrossings
ν_p	number of peaks (peak rate)
λ_n	spectral moment of order n
$\mathbb{E}$	expectation

$\mathcal{X}$	input space of NN inputs
x	NN input
x_m	mean frequency (Dirlik)
$X(t)$	process
$X(t_i)$	stress-time history
$v(t)$	temporal window function
$w(t)$	lag window function
ω	radian (angular) frequency, in radians/sampling interval
Z	normalized stress amplitude

Abbreviations

CPU	central processing unit
DFT	discrete Fourier transform
FFT	fast Fourier transform
FFNN	feedforward neural network
LSTM	long short-term memory
MAE	mean absolute error
MAPE	mean absolute percentage error
MSE	mean squared error
MSLE	mean squared logarithmic error
NN	neural network
PSD	power spectral density
RFC	rainflow counting

1. Introduction

1.1. Background

Operational durability and fatigue strength are fundamental aspects in the development of automotive structures. Mechanical components subjected to stochastic vibration loads experience cumulative material degradation, which may ultimately lead to fatigue failure. In the context of finite life design, fatigue analysis provides the theoretical basis for ensuring structural integrity under realistic service conditions [Hai06].

The principles of fatigue strength and durability have been established since the mid–20th century and are widely applied in the design of vehicles, aircraft, railway systems, and wind turbines. Contemporary durability assessment relies on time–domain load measurements and damage accumulation models to estimate component lifetime under expected operating conditions [Hai06].

In industrial practice, durability evaluation is primarily conducted during the development phase through test bench experiments and proving–ground measurements. These procedures ensure compliance with safety and reliability requirements but necessarily represent only a limited set of operating conditions.

1.2. Motivation

Durability investigations performed during the vehicle development process provide essential insights into fatigue behavior under predefined test scenarios. However, these investigations do not capture the full variability of customer usage over the vehicle lifetime. Extending durability assessment into the operational phase of customer vehicles could therefore provide further insights into real-world load spectra and enable a more realistic evaluation of fatigue damage. Gaining information about the usage by customers offers the potential to further optimize structural design, particularly with respect to lightweight construction and durability.

Direct measurement of the relevant load signals in every vehicle is economically and technically infeasible due to sensor costs, data storage limitations, and computational constraints. As a result, the concept of *virtual sensors* has emerged, aiming to estimate unmeasured quantities, such as chassis loads, from a reduced set of available signals using machine learning models. In this way, relevant load information can be inferred during vehicle operation without extensive additional hardware [Pro05].

A central challenge in deploying such virtual sensor models in customer vehicles is the limited computational capacity of onboard software components. Full-scale deep learning models are often too computationally expensive for online operation, motivating the use of compact and efficient neural network architectures.

1.3. Problem Statement

The use of compact neural network architectures for virtual sensing introduces a fundamental trade-off between computational efficiency and prediction accuracy. Due to their limited model capacity, such models often fail to capture all relevant characteristics of the underlying load history. As a consequence, the predicted force–time signals may deviate significantly from the true values.

This limitation becomes particularly critical when fatigue damage is estimated based on the predicted load histories. Fatigue damage estimation is a nonlinear process in fatigue strength and is commonly performed using discrete counting methods such as Rainflow Counting (RFC) in combination with the linear damage accumulation rule according to Miner [Hai06]. These procedures are inherently non-differentiable and highly sensitive to variations in load amplitudes and distributions. As a result, even moderate signal-level errors can lead to substantial discrepancies between fatigue damage computed from measured and predicted signals in post-processing.

In practical applications, this frequently leads to large deviations between ground truth fatigue damage and damage estimated from virtual sensor predictions, even for high time series correlations. Since conventional virtual sensor models are typically trained by minimizing signal-level loss functions, fatigue-related information is not explicitly considered during the optimization process.

To address this challenge, this thesis investigates the integration of fatigue damage information directly into the training of a virtual sensor model. The objective is to derive a surrogate model for fatigue damage estimation that approximates the rainflow-based damage measure while being at least once continuously differentiable. Such a formulation enables the incorporation of fatigue damage into the loss function of a deep learning model.

Since the optimization problem induced by neural network training is commonly solved using gradient-based methods, the differentiability of the fatigue damage approximation is a fundamental requirement. By introducing a differentiable fatigue damage surrogate as a loss function into the training, the discrepancy between true and predicted fatigue damage is intended to be reduced, even when compact neural network architectures are employed.

1.4. Literature Review

This thesis investigates a hybrid modeling approach that combines methods from operational durability assessment with machine learning techniques, together with the mathematical foundations required to link these domains.

The fundamentals of fatigue damage estimation based on RFC and linear damage accumulation according to Miner are well established in the engineering literature. Classical references such as [Hai06] and [Köh+12] provide a comprehensive treatment of rainflow-based counting procedures and fatigue damage evaluation. The formal description of RFC using four-point counting and rainflow matrices, from which span pairs are derived for fatigue damage computation, is presented in the work of Krüger and Petersen [KP85] and is used in this thesis to explain the classical fatigue damage assessment process.

As an alternative to time-domain fatigue evaluation, spectral methods for fatigue damage estimation have been proposed. The spectral method introduced by Dirlik and further discussed and compared by Tovo and Benasciutti is presented in [Dir85] and [BT05]. These works form the basis for frequency-domain fatigue damage estimation and motivate the use of spectral fatigue damage surrogates.

Since spectral methods rely on spectral densities, their estimation through (windowed) periodograms is introduced and discussed based on the work by P. Stoica and R. Moses [SM05].

Furthermore, this thesis builds on established literature in machine learning and deep learning. Standard textbooks such as [GBC16] and [Agg23] introduce the fundamentals of multilayer neural networks and gradient-based training algorithms. Since machine learning models are trained by solving optimization problems, mathematical concepts related to stability and robustness play an important role. In particular, the theory of Lipschitz continuity has been discussed as an indicator of solvability and robustness of neural networks, for example in [KS24].

1.5. Contents of this Thesis

This thesis is structured as follows:

Chapter 2 introduces the theoretical background required for fatigue damage estimation. In Section 2.1 the representation of stress and force–time histories is discussed, followed by a detailed description of the RFC method in Section 2.2. The individual steps required for rainflow-based fatigue damage computation are presented, including four-point counting, stress range pair formation (Section 2.3), and linear damage accumulation according to Miner's rule (Section 2.4).

Chapter 3 introduces the fatigue damage estimation method considered as a surrogate for Rainflow-based damage. The spectral fatigue damage method proposed by Dirlik is presented in Section 3.1.3 and discussed in relation to the spectral method by Tovo–Benasciutti (Section 3.1.4).Since spectral fatigue damage estimation relies on spectral moments (Section 3.1.2) derived from the power spectral density (PSD), different PSD estimation approaches are introduced in this chapter in Section 3.1.1. Two PSD estimation methods are considered: the basic periodogram and the windowed periodogram by Welch. Although these methods are commonly implemented as black-box algorithms in numerical software, their theoretical definitions allow the derivation of analytical gradients with respect to the input signal. The PSD estimation methods and the spectral fatigue damage model are implemented and validated in Section 3.2. Furthermore, the influence of PSD estimation on the fatigue damage computed using the Dirlik method is investigated in Section 3.3, with a particular focus on comparing the periodogram and the Welch method.

Chapter 4 introduces the fundamentals of deep learning required for the formulation of the fatigue-informed loss. In particular, the loss function is defined in Section 4.1, and relevant aspects of neural network architectures and the backpropagation algorithm are discussed in Section 4.2.

Chapter 5 is dedicated to the analysis of the gradients of the spectral fatigue loss. The analytical gradients of the PSD estimators and the Dirlik fatigue damage model are derived and implemented in Section 5.1, and their correctness is verified by comparison

with finite difference approximations in Section 5.2. Furthermore, in Section 5.3 the behavior of the gradients is investigated, with particular emphasis on issues such as vanishing gradients and their impact on the training process. Additionally, different loss function formulations suitable for this type of problem are investigated and compared. The performance of a baseline virtual sensor model and a model trained with the fatigue-informed loss function is evaluated and discussed.

Finally, Chapter 6 presents the results of the numerical experiments, followed by a summary of the main findings and an outlook on potential directions for future work in Chapter 7.

2. Theoretical Background

This chapter introduces the theoretical foundations required for fatigue damage estimation within the framework of operational durability. In particular, the classical rainflow-based approach for fatigue damage assessment is presented, which later serves as a reference for the development of surrogate fatigue damage models.

At the beginning of this chapter, the notation and fundamental concepts of vibratory stress are introduced, with stress–time histories serving as the primary load representation. Based on this formulation, the rainflow-based fatigue damage method is defined step by step. The RFC method is presented as the classical procedure for cycle identification in fatigue analysis. Its formulation is based on the four-point counting method, which enables the extraction of closed load cycles from stress–time histories and their representation in the form of a rainflow matrix. From the rainflow matrix, stress–range pairs (span pairs) are identified and counted. These span pairs form the basis for fatigue damage evaluation, as they characterize both the magnitude and frequency of load cycles. Finally, Miner's rule is introduced as a linear damage accumulation model that combines the identified stress range pairs into a scalar fatigue damage measure. This rainflow-based formulation constitutes the classical reference against which alternative, differentiable fatigue damage estimation methods are assessed in this thesis.

2.1. Vibratory Stress

The Figure 2.1, compared to the illustration depicted in [Hai06], displays a single cycle of vibratory stress and its characteristic quantities, which are described as follows.

The stress associated with a force–time history is denoted by s. The maximum (upper) and minimum (lower) stresses are denoted respectively as s_u and s_l, with the mean stress s_m located between them. The difference between upper and lower stress is reffered to as the stress range Δs, which is symmetrically distributed into the stress amplitude s_a at the upper and lower extrema. These characteristic quantities of a stress cycle are related to each other as depicted in Figure 2.1 and formalized in the following equations.

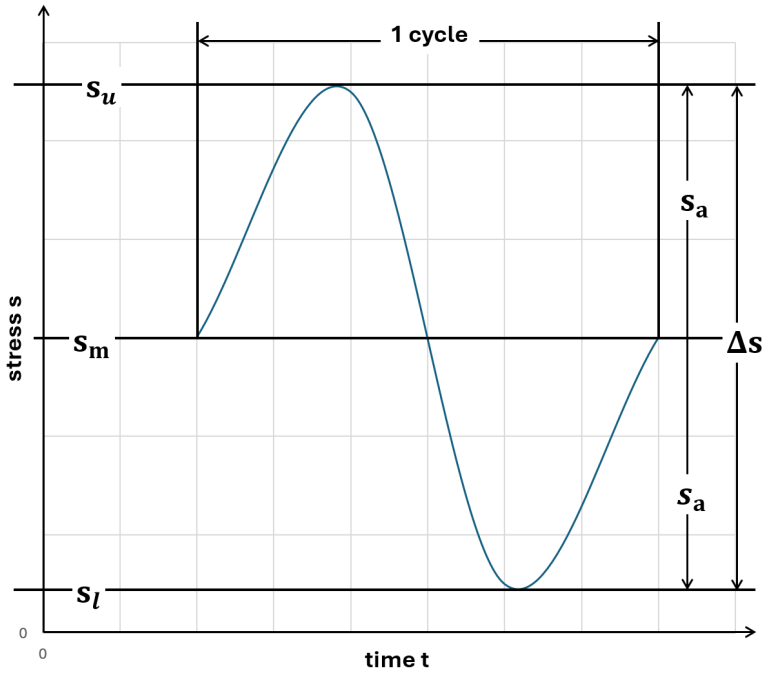


Figure 2.1.: Characteristics of stress cycles

$$s_u = s_m + s_a, \tag{2.1}$$
$$s_l = s_m - s_a, \tag{2.2}$$
$$s_a = (s_u - s_l)/2, \tag{2.3}$$
$$s_m = (s_u + s_l)/2, \tag{2.4}$$
$$r = s_l/s_u, \tag{2.5}$$
$$s_a = s_u \cdot (1 - r)/2, \tag{2.6}$$
$$s_m = s_u \cdot (1 + r)/2, \tag{2.7}$$
$$s_m = s_u \cdot (1 + r)/(1 - r), \tag{2.8}$$
$$\Delta s = (s_u - s_l) = 2 \cdot s_a. \tag{2.9}$$

The relations (2.1) and (2.2) describe the upper and lower stress values as the sum and difference of mean stress and stress amplitude. From these definitions, the stress amplitude follows from half the difference between upper and lower stress (2.3), which corresponds to the stress range Δs (Figure 2.1), while the mean stress is obtained as the arithmetic mean of the upper and lower stress levels (2.4). An alternative description of the stress cycle is provided by the stress ratio r, which relates the lower and upper stress values (2.5). By rearranging the defining relations and substituting the stress ratio, the last equations yield equivalent expressions for the stress amplitude and mean stress (2.6), (2.7), and (2.8).

In addition to these characteristics, a variety of counting algorithms, including RFC, discretize the stress–time history into a finite number of equidistant stress levels, as shown in Figure 2.2. These stress levels are represented by horizontal blue lines in

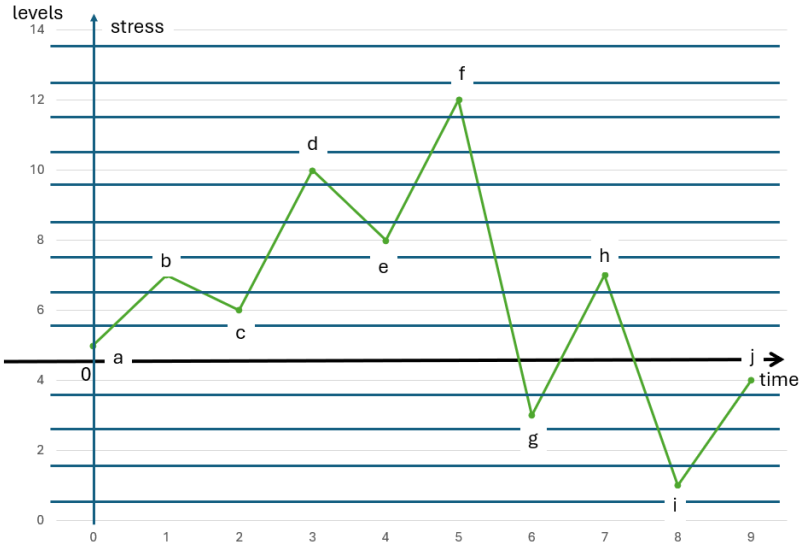


Figure 2.2.: Stress level division for counting

Figure 2.2. In the present example, the stress range is divided into 14 equidistant levels, where level 12 is the highest level at which a peak occurs.

2.2. Rainflow Counting

Rainflow Counting (RFC) is based on the RFC method introduced in 1968 and the range–mean pair counting approach proposed in 1969, both of which lead to identical results. Consequently, the method is commonly referred to as RFC as stated in [Hai06]. The RFC algorithm counts closed stress–strain hysteresis loops within a stress–time history. The area enclosed by a hysteresis loop provides an intuitive physical measure of the energy dissipated in the material, which is directly related to fatigue in the form of plastic deformation and crack initiation and growth. Due to this physically motivated foundation, RFC is considered the only cycle counting method with a direct physical interpretation and is therefore widely used in fatigue analysis [Köh+12].
In practice, the algorithm consists of several preprocessing steps, namely hysteresis filtering, peak–valley filtering, and discretization, followed by the core counting procedure known as the *four-point counting method*. While this method is often only described verbally in the literature, W. Krüger and J. Petersen provide a formal definition of the conditions required for four-point counting in their report *Simulation und Extrapolation von Rainflow-Matrizen* [KP85]. Their formulation introduces a precise notation and general conditions, extending example-based descriptions commonly found in the literature.

In the following, the discretized stress–time history is denoted by $X(t_i) = y_u$, where $X(t_i)$ represents the stress value at the discrete time instant t_i, and y_u denotes this value at the corresponding stress level u. The time points t_i, $i = 1, 2, \ldots, m$, represent a discrete stress–time history, while the stress levels y_u, $u = 1, 2, \ldots, n$, arise from the discretization of the stress range. As a result, the number of stress levels generally differs from the number of time points, and multiple extrema may be associated with

the same stress level.

For the further explanation of the four-point counting method, the stress function X considered here is assumed to have already been filtered and discretized by the previously described preprocessing steps, and thus to represent only local maxima and minima. Accordingly, these extrema satisfy

$$\begin{aligned} & X(t_i) > X(t_{i-1}) \text{ and } X(t_i) > X(t_{i+1}), \\ \text{or} \quad & X(t_i) < X(t_{i-1}) \text{ and } X(t_i) < X(t_{i+1}). \end{aligned}$$

Based on this, the RFC method in the sense of the four-point counting rule is defined as follows: the algorithm proceeds by identifying intermediate cycles represented by pairs of stress levels (y_i, y_j), counting their frequency a_{ij}, and eliminating these cycles from the stress–time history. After all intermediate cycles have been removed, the remaining sequence is stored separately as the residual function $\bar{R}$. Here, the indices i and j denote the indices of the discretized stress levels involved in a candidate cycle, such that (y_i, y_j) represents a cycle between the stress levels y_i and y_j.

The decision rules used to determine whether an intermediate cycle exists are given below. For a candidate cycle formed by the stress levels i and j with $y_i < y_j$, the conditions are evaluated for indices $k \in \{4, \ldots, m\}$:

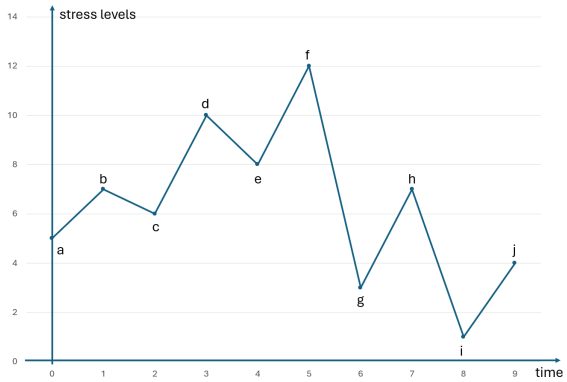
- Case $(y_i, y_j),\ y_i < y_j$:

$$X(t_{k-3}) \geq X(t_{k-1}) = y_j \text{ and } X(t_k) \leq X(t_{k-2}) = y_i$$

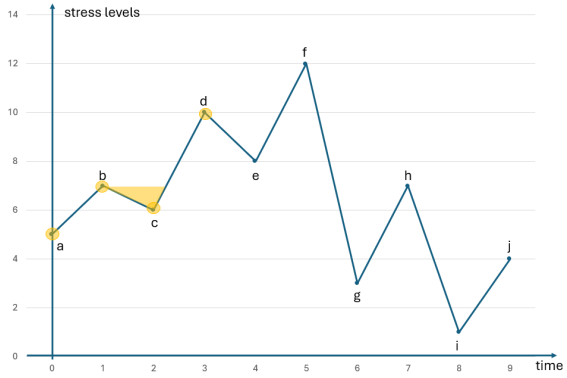
- Case $(y_j, y_i),\ y_i < y_j$:

$$X(t_{k-3}) \leq X(t_{k-1}) = y_i \text{ and } X(t_k) \geq X(t_{k-2}) = y_j.$$

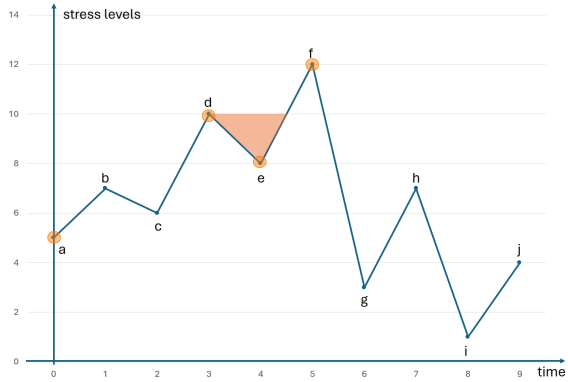
An example illustrating the four-point counting rule and, consequently, the RFC algorithm is shown in Figure 2.3.



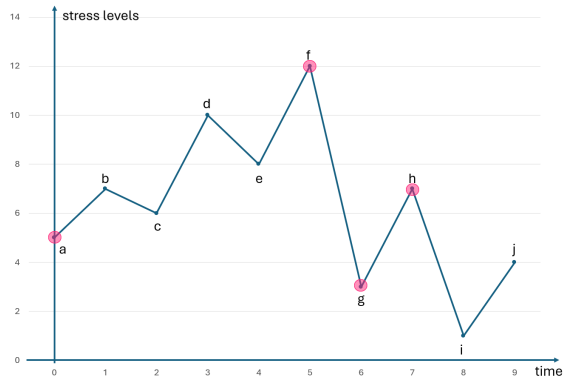
(a) Stress–time history discretized into equidistant stress levels (horizontal lines from 0 to 14).



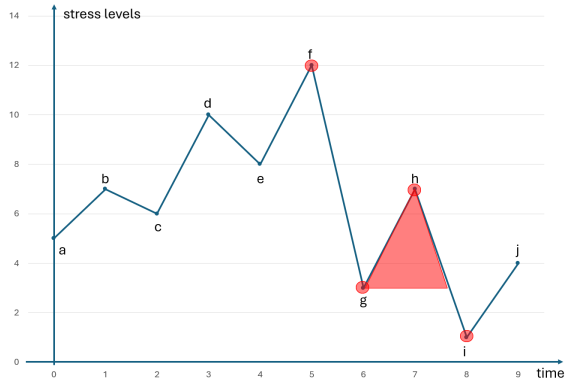
(b) Selection of the first four points a, b, c, and d. The conditions $a < c$, $b < d$, and $c < b$ identify a full cycle b–c.



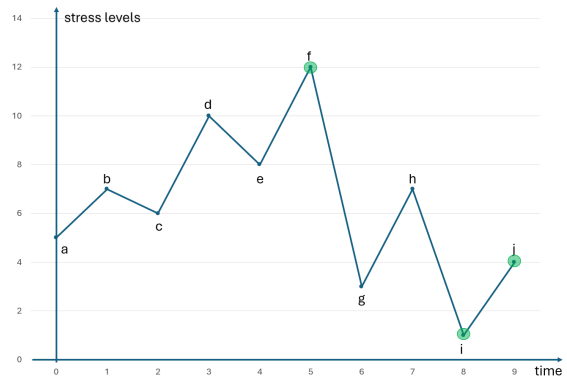
(c) Next iteration using the reduced sequence. The conditions $a < e$, $d < f$, and $e < d$ result in a full cycle d–e.



(d) Selection of the next four points. The conditions $a > g$, $f > h$, and $g < f$ identify a half cycle a–f.



(e) After counting a half cycle, the first point is discarded. The conditions $f > h$, $g > i$, and $h > g$ identify a full cycle g–h.



(f) Evaluation of the remaining points. The remaining sequence results in a half cycle i–j.

Figure 2.3.: Illustrative example of the Rainflow Counting algorithm based on the four-point counting rule.

Equivalently, the identification of intermediate cycles within the RFC method can

be formulated in terms of necessary and sufficient conditions. These conditions are evaluated as part of the four-point counting rule.A necessary condition for the existence of an intermediate cycle is given by:

$$|X(t_{k-2}) - X(t_{k-1})| \leq |X(t_{k-3}) - X(t_k)|, \quad k \in \{4, \ldots, m\}.$$

Sufficient conditions are obtained by the inequalities:

$$|X(t_{k-2}) - X(t_{k-1})| \leq |X(t_{k-3}) - X(t_{k-2})|,$$
$$|X(t_{k-2}) - X(t_{k-1})| \leq |X(t_{k-1}) - X(t_k)|, \quad k \in \{4, \ldots, m\}.$$

If these conditions are satisfied for a given pair of stress levels (y_i, y_j), the corresponding cycle is identified as an intermediate cycle and counted in the rainflow matrix. Specifically, each occurrence of a cycle between the stress levels y_i and y_j increases the corresponding matrix entry a_{ij} by one, after which the cycle is eliminated from the stress sequence [KP85].

The above conditions form the basis for an algorithmic implementation of the four-point counting method, which is summarized in Algorithm 1. The algorithm assigns an empty rainflow matrix A (line 5), an empty list for storing the residuum $\bar{R}$ (line 6), and initializes the working list x containing the stress level values (line 7).

Algorithm 1 Four-Point Counting

```
1:  Function: count cycles
2:  Input: Preprocessed stress-time history X(t_i) = y_{j_i}, i = 1,...,m, j = 1,...,n
3:  m: number of peak and valley points, n: number of levels
4:  Output: Rainflow Matrix A, Residuum R̄
5:  A ← ∅
6:  R̄ ← ∅
7:  x ← y_j                          ▷ working list, i.e. x[j] indicates the list entry at level j
8:  while length(x) ≥ 4 do
9:      if |x[2] − x[3]| ≤ |x[1] − x[4]| then
10:         A(x[2], x[3]) ← A[x[2]][x[3]] + 1
11:         x ← x \ {x[2], x[3]}
12:     else
13:         R̄ ← R̄ ∪ {x[1]}
14:         x ← x \ {x[1]}
15:     end if
16: end while
17: R̄ ← R̄ ∪ x
18: return A, R̄
```

While the list of stress levels contains at least 4 elements (line 8), the sufficient condition for a cycle is evaluated for the first four entries of the list (line 9), denoted by $x[1], x[2], x[3], x[4]$. If this condition is satisfied, the corresponding cycle is counted by incrementing the respective entry of the rainflow matrix by 1 (line 10). Subsequently, the two middle elements are removed from the working list x (line 11). If the condition is not satisfied, no cycle is counted. Instead, the first entry is added to the residuum

list $\bar{R}$ (line 13) and removed from the working list (line 14). After the loop terminates, the remaining elements of the list are added to the residuum. Various approaches exist for handling the residuum, one of which consists of repeatedly applying the four-point counting rule to the residuum until no further cycles can be identified.

2.3. Stress Range Pair Method

The stress range pair method quantifies the frequency of occurrence of stress cycles identified by the RFC method by grouping them according to their discrete stress range. In this setting, the stress range is characterized by the difference of the corresponding stress-level indices $|i - j|$ of a cycle. Based on the discretized stress representation introduced in the previous section, each Rainflow-counted cycle corresponds to an ordered pair (y_i, y_j) of stress levels, where $i, j \in \{1, \ldots, n\}$ denote the corresponding stress-level classes, and y_i, y_j the associated stress values.

Definition 2.1 (Spanning pairs [KP85]) For a fixed integer $l \in \{1, \ldots, n-1\}$, the set of *spanning pairs* of span l is defined as

$$SP(l) \coloneqq \Big\{ (y_i, y_j) \;\Big|\; |i - j| = l,\ i \neq j \Big\}.$$

Consequently, all pairs in $SP(l)$ correspond to rainflow-counted cycles with identical stress range l.

The stress range pair method determines the total number of cycles associated with each discrete stress range l by summing the corresponding entries of the rainflow matrix (a_{ij}):

$$S(l) \coloneqq \sum_{\substack{i,j=1 \\ |i-j|=l}}^{m} a_{ij}, \qquad l = 1, 2, \ldots, n-1.$$

Thus, $S(l)$ denotes the frequency of occurrence of rainflow-counted cycles with span l.

2.4. Miner's Rule

To compute the accumulated fatigue damage from the rainflow-counted cycles obtained via the stress range pair method, the classical linear damage accumulation rule according to Palmgren and Miner is employed. Based on the rainflow matrix representation introduced previously, the stress range pair method yields the number of cycles $S(l)$ associated with each discrete stress range $l = |i - j|$. These cycle counts form the input for Miner's rule.

2.4.1. Stress Amplitude Associated with a Spanning Pair

Let $\Delta\sigma$ denote the stress discretization step used in the construction of the rainflow matrix. For a spanning pair (y_i, y_j) with discrete stress range $l = |i - j|$, the corresponding physical stress amplitude is given by

$$s_{a,l} = \frac{|y_j - y_i|}{2} = \frac{|j - i|}{2}\,\Delta\sigma = \frac{l}{2}\,\Delta\sigma.$$

Thus, each discrete stress range l corresponds to a unique stress amplitude $s_{a,l}$.

2.4.2. Linear Damage Accumulation

According to Miner's rule, the accumulated fatigue damage is defined as

$$D = \sum_{l=1}^{n-1} \frac{n_l}{N(s_{a,l})},$$

where $n_l = S(l)$ denotes the number of cycles with discrete stress range l, and $N(s_{a,l})$ is the number of cycles to failure corresponding to the stress amplitude $s_{a,l}$. The damage computed in this way is normalized such that $D = 0$ represents the start of fatigue life, while $D = 1$ indicates failure.

Following Haibach [Hai06], the fatigue life is described by the Wöhler (S–N) curve

$$N(s_{a,l}) = N_D \left(\frac{s_{a,l}}{s_D} \right)^{-m}, \tag{2.10}$$

where N_D is the number of cycles at the endurance limit s_D, and m is the slope exponent of the S–N curve as displayed in Figure 2.4. This formulation establishes a direct link between the discrete rainflow cycle counts obtained from the stress range pair method and the physically interpreted fatigue damage measure.

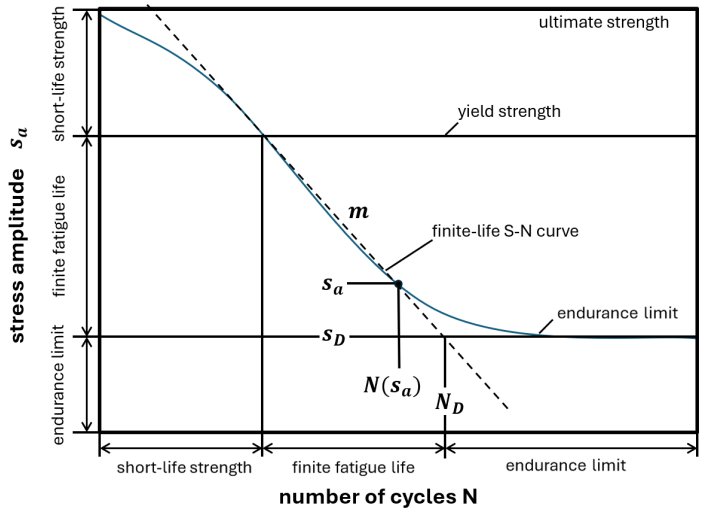


Figure 2.4.: Characteristic values of an S–N curve and delineation of the regions of endurance limit, finite fatigue life, and low-cycle fatigue. (compare [Hai06])

3. Fatigue Damage Methods and Power Spectral Density Estimation

Over the years, many methods have been proposed and validated to replace widely used cycle counting methods such as the RFC method, with the aim of improving their applicability for computer-based fatigue analysis. Among these approaches are spectral methods. One of the earliest and most widely used spectral methods was developed in 1985 by T. Dirlik in his work *Application of Computers in Fatigue Analysis* [Dir85]. This method is one of the most commonly applied spectral approaches and provides fatigue damage estimates that closely resemble those obtained via the RFC method. Another spectral method was proposed more than 20 years later, in 2005, by Tovo and Benasciutti [BT05].

Spectral methods formulate fatigue damage in terms of damage rates derived from the theory of rainflow density functions. The accumulated fatigue damage is obtained by multiplying this damage rate by the duration of the considered stress signal [BT05].As spectral moments form the foundation of these methods and are derived from the one-sided power spectral density (PSD), the estimation of the PSD is a key component of the spectral approach. The theory of spectral analysis offers various methods for PSD estimation depending on the underlying problem and the available data [SM05]. In this work, both the simple periodogram and the windowed periodogram according to Welch, including the option of overlapping windows, are considered. Based on this framework, the spectral damage estimation methods of Dirlik and Tovo–Benasciutti are introduced, and the resulting fatigue damages are subsequently validated by comparison with the RFC method.

Since the choice of the PSD estimation method influences the spectral moments and, consequently, the estimated fatigue damage, the different PSD estimation approaches are assessed with respect to their influence on the damage results in relation to the Rainflow method.

3.1. Spectral Methods

Spectral methods for estimating the fatigue damage of mechanical components are formulated in the frequency domain, where the underlying theory generally assumes random loadings —which commonly occur in engineering applications— to be stationary and Gaussian. This assumption may be restrictive for measured stress signals, as such loadings tend to exhibit non-stationary behavior [BT05]. A variety of spectral methods have been proposed in the literature, among which the approaches by Dirlik [Dir85] and by Tovo–Benasciutti [BT05] are the most prominent, due to their close agreement with fatigue damage estimates obtained from the RFC method.

The PSD is first defined formally, ensuring differentiability with respect to the input signal, which is required as the PSD constitutes the final element of the gradient chain used to derive the gradient of the Dirlik method for backpropagation in neural networks. In addition, further investigations are conducted in connection with the Welch method, comparing the efficiency and accuracy of the apodized and windowed periodogram approaches.

Building on these foundations, both spectral fatigue damage estimation methods and the underlying spectral theory are introduced and defined. The resulting fatigue damages are subsequently validated by comparison with the reference method of RFC.

3.1.1. Power Spectral Density Estimation

The rate of occurrence, also referred to as the peak rate ν_p, is defined based on the spectral moments, as will be shown in Section 3.1.2. Since these spectral moments are derived from the PSD, the definition and estimation of the PSD constitutes a central component of the spectral approach.

A distinction is made between parametric and nonparametric methods of spectral estimation, where the latter directly estimate the PSD from the measured signal without assuming an explicit signal model [SM05]. In this work, the focus is placed on nonparametric approaches, in particular the simple periodogram and the windowed periodogram method according to Welch, including the option of overlapping windows.

As the spectral fatigue damage estimation methods considered here rely on the PSD and must be differentiable in order to be integrated into neural network frameworks, the PSD estimation has to be implemented in a differentiable manner. This requirement arises because backpropagation is based on gradient computations, while commonly used nonparametric PSD estimators, such as the Welch method, are often treated as black-box algorithms. Consequently, numerical differentiation by finite differences would be computationally infeasible for long signals or large data sets.

First, the simple periodogram method is described, which can be implemented using the Fast Fourier Transform, as shown in Section A.1.1. The corresponding implementation is realized in TensorFlow to allow seamless integration into the training of deep learning models based on `TensorFlow.keras`. However, the periodogram exhibits several well-known limitations, which motivates the use of windowed correlogram and windowed periodogram approaches to mitigate these issues. In this context, the Welch method is introduced together with commonly used window functions, including the option of overlapping windows. In addition, the apodization approach is explained and investigated with respect to a potential trade-off between estimation accuracy and computational efficiency, and the reasons why it is ultimately not selected for the present application are discussed.

The theoretical definition of the one-sided power spectral density is described in standard references on spectral analysis, as well as in the work of Benasciutti and Tovo [BT05].

PSD Estimation via Periodogram

In the notation used here, $\hat{\Phi}$ denotes the spectral estimate, also referred to as the PSD, as a function of the angular frequency ω. Since the periodogram method considers

the entire signal without windowing, the quantity N corresponds to the length of the discrete time signal $y(t)$. Here, $y(t)$ denotes a discrete-time process, i.e., the time variable t takes values in a discrete index set.

Based on this notation, periodogram-based PSD estimates, indicated by the subscript p, are defined as

$$\hat{\Phi}_p(\omega) = \frac{1}{N} \left| \sum_{t=1}^{N} y(t)\, \mathrm{e}^{-\mathrm{i}\omega t} \right|^2 . \tag{3.1}$$

In this expression, the term $\sum_{t=1}^{N} y(t)\, \mathrm{e}^{-\mathrm{i}\omega t}$ corresponds to the discrete Fourier transform (DFT) of the signal $y(t)$, which is evaluated numerically using the fast Fourier transform (FFT) [SM05]. The FFT refers to a class of algorithms that compute the DFT efficiently [SM05]. While the DFT is complex-valued, the squared magnitude yields a real-valued quantity. As it consists of compositions of elementary operations on real-valued inputs, the resulting PSD estimate is differentiable with respect to the signal values $y(t)$.

This estimator can be related to the theoretical definition of the power spectral density of a discrete-time signal $y(t)$, given by

$$\Phi(\omega) = \lim_{N\to\infty} \mathbb{E}\left\{ \frac{1}{N} \left| \sum_{t=1}^{N} y(t)\, \mathrm{e}^{-\mathrm{i}\omega t} \right|^2 \right\},$$

as described in [SM05]. Note that the signal $y(t)$ is assumed to be of zero mean, i.e. $\mathbb{E}[y(t)] = 0$. If this requirement is not satisfied, it can be enforced by defining $\tilde{y}(t) = y(t) - \mathbb{E}[y(t)]$, which yields $\mathbb{E}[\tilde{y}(t)] = 0$.

Windowed Periodogram by Welch

The Welch method is a so-called *windowed periodogram spectral estimator*, developed by Welch [Wel67] as an alternative approach based on the windowed correlogram method of Blackman and Tukey [SM05].The latter was originally introduced to address the high statistical variability of the spectral estimator defined by the unwindowed periodogram. A major drawback of the periodogram is its large variance around the true PSD. By applying a window (or weighting function) to the estimated autocorrelation function, the Blackman–Tukey approach aims to smooth the resulting spectral estimate and thereby reduce these large fluctuations [SM05].

In the theory of spectral analysis, the weighting functions used in the Blackman–Tukey approach are interpreted as *lag windows*, whereas the window functions applied in Welch's method are referred to as *temporal windows*. Although the two approaches differ conceptually, they exhibit identical average behavior provided that their respective windows are related by

$$w(k) = \frac{1}{N} \sum_{n=1}^{N} v(n)\, v^*(n-k),$$

where $v(n)$ denotes the temporal window of the Welch method, $v^*(\cdot)$ its complex conjugate, and $w(k)$ the corresponding lag window of the Blackman–Tukey estimator. For example, this condition is satisfied for the rectangular window, which explains the equivalence of the two approaches in this case [SM05].

Although the windowed correlogram approach is preferred from a theoretical perspective, the Welch method is more suitable for practical applications and is therefore

employed in this work. In contrast to the correlogram approach, which additionally requires the estimation of the covariance function, the Welch method entails a lower computational effort [SM05]. For this reason, only the Welch method is formally defined in the following.

For the windowing procedure, the signal y is split into S data segments, each of length M,

$$y_j(t) = y\Big((j-1)K + t\Big),\ t = 1, \ldots, M,\ j = 1, \ldots, S. \tag{3.2}$$

The starting point for the jth data segment is $(j-1)K$. Thus, if $K = M$, the sequences do not overlap. A commonly recommended value for K is an overlap of 50%, which is obtained by choosing $K = M/2$ [SM05].

To ensure that the mean of the signal is approximately zero, which is common practice to avoid biased spectral estimates, the mean correction is performed as follows:

$$\tilde{y}_j(t) = y_j(t) - \mathbb{E}[y_j(t)]. \tag{3.3}$$

The corresponding windowed periodogram for each of the data segments $\tilde{y}_j(t)$ is defined by

$$\hat{\Phi}_j(\omega) = \frac{1}{MP}\left|\sum_{t=1}^{M} v(t)\,\tilde{y}_j(t)\,\mathrm{e}^{-\mathrm{i}\omega t}\right|^2. \tag{3.4}$$

Here, P denotes the power of the temporal window $\{v(t)\}$ and is given by

$$P = \frac{1}{M}\sum_{t=1}^{M} |v(t)|^2. \tag{3.5}$$

The spectral estimate of the PSD based on Welch's method is then obtained by averaging the windowed periodograms,

$$\hat{\Phi}_W(\omega) = \frac{1}{S}\sum_{j=1}^{S} \hat{\Phi}_j(\omega). \tag{3.6}$$

From the definitions of the periodograms as well as the PSD, it follows that $\hat{\Phi}(\omega)$ is a periodic function. Consequently, $\hat{\Phi}(\omega)$ is completely described on the interval

$$\omega \in [-\pi, \pi].$$

The PSD can equivalently be viewed as a function of the frequency $f = \frac{\omega}{2\pi}$ [SM05].

Even if the windowed methods theoretically resolve the problem of high variability in the periodogram, they introduce a possible trade-off between the reduction of leakage and, consequently, a reduction of resolution [SM05]. As the common windows (Table 3.1) are data- and frequency-dependent, the method to resolve this is the apodization approach as presented in [SM05].

For this approach, the temporal windows $v(t)$ have to fulfill the following two constraints [SM05], namely being non-negative and having a normalized sum:

$$v(t) \geq 0 \qquad \text{and} \qquad \sum_{t=1}^{M} v(t) = 1. \tag{3.7}$$

Checking these constraints for some common temporal windows [SM05], it appears that they do not initially fulfill them, as shown in Table 3.1.

In the present work, these commonly used window functions are nevertheless considered in the subsequent analysis, even though they do not strictly satisfy the apodization constraints. This is done to investigate the behavior of the apodization-based formulation in the non-adaptive setting, and to assess its computational implications when applied in a framework comparable to the Welch method.

Table 3.1.: Window functions

Window Name	Defining Equation	Fulfillment window constraints
Rectangular	$v(t) = 1$	$v(t) \geq 0$ ✓ $\sum_{t=1}^{M} v(t) = M \neq 1$ ✗
Bartlett	$v(t) = \frac{M-t}{M}$	$v(t) \geq 0$ for $t \leq M$ ✓ $\sum_{t=1}^{M} v(t) = \frac{M-1}{2} \neq 1$ for $M \neq 3$ ✗
Hannning	$v(t) = 0.5 + 0.5\cos(\frac{\pi t}{M})$	$v(t) \geq 0$ ✓ $\sum_{t=1}^{M} v(t) = \frac{M-1}{2} \neq 1$ ✗
Hamming	$v(t) = 0.54 + 0.46\cos(\frac{\pi t}{M-1})$	$v(t) \geq 0$ ✓ $\sum_{t=1}^{M} v(t) \neq 1$ ✗
Blackman	$v(t) = 0.42 + 0.5\cos(\frac{\pi t}{M-1})$ $+0.08\cos(\frac{\pi t}{M-1})$	$v(t) \geq 0$ ✓ $\sum_{t=1}^{M} v(t) \neq 1$ ✗

In general, there exists a class of temporal windows of the form

$$v(t) = \frac{1}{N}[\alpha - \beta\cos(\frac{2\pi}{N}t)], \qquad t = 1, \ldots, N \tag{3.8}$$

mentioned in *Spectral Analysis of Signals* by Stoica and Moses [SM05], which satisfy these constraints. However, this result only holds under the parameter conditions stated in Proposition 3.1, which are given in [SM05], while the formalization and proof are provided here.

Proposition 3.1. *Temporal windows of the form given by Equation 3.8 satisfy the constraints of windows for the apodization approach (Equation 3.7) if and only if*

$$\alpha = 1 \qquad \textit{and} \qquad |\beta| \leq 1,\ \beta \geq 0. \tag{3.9}$$

Proof.
"⇐":
Let $\alpha = 1$ and $|\beta| \leq 1$, $\beta \geq 0$. Then, Equation 3.8 is non-negative, because the cosine term lies in the interval $[-1, 1]$ and β is assumed to lie in $[0, 1]$. Hence, the product $\beta\cos(\frac{2\pi}{N}t)$ lies in the interval $[-\beta, \beta] \subseteq [-1, 1]$.

It follows that the minimal value of the term $1 - \beta\cos(\frac{2\pi}{N}t)$ is attained at $\cos(\frac{2\pi}{N}t) = 1$, yielding

$$v(t) = \frac{1}{N}(1 - \beta) \geq 0,$$

and therefore $v(t) \geq 0$ for all t. Thus, the first constraint is fulfilled.

Next, the second condition, namely that the sum of the window equals 1, follows by substituting $\alpha = 1$ and using the identity $\sum_{t=1}^{N} \cos\left(\frac{2\pi}{N}t\right) = 0$:

$$\sum_{t=1}^{N} \frac{1}{N}[\alpha - \beta\cos(\frac{2\pi}{N}t)] = \frac{1}{N}\sum_{t=1}^{N}(1 - \beta\cos(\frac{2\pi}{N}t)) = \frac{1}{N}(N - \beta\underbrace{\sum_{t=1}^{N}\cos(\frac{2\pi}{N}t)}_{=0}) = 1.$$

"⇒":
Conversely, assume that the conditions in Equation 3.7 hold for the class of windows of Equation 3.8. Then,

$$v(t) = \frac{1}{N}[\alpha - \beta\underbrace{\cos(\frac{2\pi}{N}t)}_{\in[-1,1]}] \geq 0.$$

From the normalization condition,

$$\sum_{t=1}^{N} v(t) = 1 \Rightarrow \frac{1}{N}(\alpha N - \beta\underbrace{\sum_{t=1}^{N}\cos(\frac{2\pi}{N}t)}_{=0}) = 1 \Rightarrow \alpha = 1.$$

Substituting this into the non-negativity condition yields

$$v(t) = \frac{1}{N}[1 - \beta\cos(\frac{2\pi}{N}t)] \geq 0.$$

Since $\cos(\frac{2\pi}{N}t) \in [-1, 1]$, the minimum of $v(t)$ is attained at $\cos(\cdot) = 1$ and the maximum at $\cos(\cdot) = -1$. Therefore,

$$1 - \beta \geq 0 \quad \text{and} \quad 1 + \beta \geq 0,$$

which implies

$$|\beta| \leq 1.$$

□

It should be noted that, in the apodization framework proposed by Stoica and Moses, the parameter β is determined adaptively as a function of both the frequency and the data by solving an optimization problem. In contrast, in the present work β is fixed through the choice of classical window functions (e.g., Blackman, Hann, Hamming). Consequently, the fully adaptive apodization procedure is not implemented here.

For this class of temporal windows, the windowed periodogram of the Welch method (Equation 3.4) takes the following form. Since the solution of this system is obtained via the DFT, the relation

$$\omega = \frac{2\pi}{M}k \qquad \Leftrightarrow \qquad f = \frac{k}{M}, \quad 0 \leq k \leq M - 1$$

holds for each window of length M.

First, the window $v(t)$ in the DFT term of the Welch method in Equation 3.4 is replaced by the definition of this general window class given in Equation 3.8, i.e. $v(t) = \frac{1}{M}[1 - \beta\cos(\frac{2\pi}{M}t)]$:

$$\varphi(i\omega) = \sum_{t=1}^{M} v(t)\, y_j(t) e^{-i\omega t} = \frac{1}{M}\sum_{t=1}^{M}[1 - \beta\cos(\frac{2\pi}{M}t)] y_j(t) e^{-i\omega t}.$$

In the next step, the cosine is replaced by its exponential representation, and $\omega = \frac{2\pi}{M}k$ is applied:

$$\varphi(i\omega) = \frac{1}{M}\sum_{t=1}^{M}\left[1 - \frac{\beta}{2}\left(e^{i\frac{2\pi}{M}t} + e^{-i\frac{2\pi}{M}t}\right)\right] y_j(t) e^{-i\omega t}$$

$$= \frac{1}{M}\left(\sum_{t=1}^{M} y_j(t) e^{-i\frac{2\pi}{M}tk} - \frac{\beta}{2}\left(\sum_{t=1}^{M} y_j(t) e^{i\frac{2\pi}{M}t} e^{-i\frac{2\pi}{M}tk} + \sum_{t=1}^{M} y_j(t) e^{-i\frac{2\pi}{M}t} e^{-i\frac{2\pi}{M}tk}\right)\right).$$

This expression can then be simplified by introducing the DFT definition

$$Y_j(k) = \sum_{t=1}^{M} y_j(t) e^{-i\frac{2\pi}{M}tk},$$

and replacing all corresponding terms. This leads to a closed-form expression of the apodization approach for the windowed periodogram within this general class of windows:

$$\varphi(i\omega) = \frac{1}{M}\left(Y_j(k) - \frac{\beta}{2}\left(\sum_{t=1}^{M} y_j(t) e^{-(k-1)i\frac{2\pi}{M}t} + \sum_{t=1}^{M} y_j(t) e^{-(k+1)i\frac{2\pi}{M}t}\right)\right)$$

$$= \frac{1}{M}\left(Y_j(k) - \frac{\beta}{2}\Big(Y_j(k-1) + Y_j(k+1)\Big)\right)$$

$$\Rightarrow \hat{\Phi}_j(k) = \frac{1}{MP}\left|\frac{1}{M}\left[Y_j(k) - \frac{\beta}{2}(Y_j(k-1) + Y_j(k+1))\right]\right|^2.$$

Through similar computations, the windowed periodogram of the Blackman window is derived as

$$\hat{\Phi}_j(k) = \frac{1}{MP}\left|0.42Y_j(k) - 0.25(Y_j(k-1) + Y_j(k+1)) + 0.04(Y_j(k-2) + Y_j(k+2))\right|^2.$$

The Blackman window is defined as

$$v(t) = 0.42 - 0.5\cos\left(\frac{2\pi t}{N}\right) + 0.08\cos\left(\frac{4\pi t}{N}\right), \qquad t = 0, 1, \ldots, N-1,$$

as described, for instance, in the `scipy` documentation [Vir+20], and is used in the present derivation.

This representation shows that the application of the temporal window corresponds to a weighted combination of shifted Fourier coefficients, which can be interpreted as a local smoothing in the frequency domain. It is used to analyze the computational structure of the windowed periodogram and to compare it with the standard implementation of the Welch method. In particular, it enables an assessment of the additional computational effort introduced by the apodization-based formulation, independent of any adaptive window selection.

The following Table 3.2 summarizes the windowed periodograms for the different window functions stated before in Table 3.1, expressed in terms of the apodization approach.

Table 3.2.: Apodization approach for different windows

Window function	Definition
Blackman	$v(t) = 0.42 - 0.5\cos\Big(\frac{2\pi t}{M}\Big) + 0.08\cos\Big(\frac{4\pi t}{M}\Big),$ $\hat{\Phi}_j(k) = \frac{1}{MP}\Big\lvert 0.42Y_j(k) - 0.25\Big(Y_j(k-1) + Y_j(k+1)\Big) + 0.04\Big(Y_j(k-2) + Y_j(k+2)\Big)\Big\rvert^2$
Hanning	$v(t) = 0.5 - 0.5\cos\Big(\frac{2\pi t}{M}\Big),$ $\hat{\Phi}_j(k) = \frac{1}{MP}\Big\lvert 0.5Y_j(k) - 0.25\Big(Y_j(k-1) + Y_j(k+1)\Big)\Big\rvert^2$
Hamming	$v(t) = 0.54 - 0.46\cos\Big(\frac{2\pi t}{M}\Big),$ $\hat{\Phi}_j(k) = \frac{1}{MP}\Big\lvert 0.54Y_j(k) - 0.23\Big(Y_j(k-1) + Y_j(k+1)\Big)\Big\rvert^2$
Rectangular	$v(t) = 1,$ $\hat{\Phi}_j(k) = \frac{1}{MP}\lvert Y_j(k)\rvert^2$

In Section 3.2, the spectral estimates of the Welch method and the apodization approach introduced here are validated against the numerical implementation of Welch provided by `scipy` [Vir+20]. In addition, both methods are compared to each other, as the apodization approach is intended to improve the leakage behavior of individual spectral estimates while minimizing the associated loss in frequency resolution, albeit at the cost of increased computational effort [SM05]. Thus, a trade-off between both methods is expected. Since the PSD serves as input for the Dirlik method, it must furthermore be sufficiently accurate while remaining computationally efficient with regard to a potential integration into a neural network. Moreover, the effectiveness of apodization in reducing leakage depends on the selection of suitable windows for

different signal segments, and the identification of such windows constitutes an additional optimization problem. This is therefore assumed to be too costly for a wide variety of signals compared to the potential benefit.

3.1.2. Spectral Theory

After considering the different spectral estimation choices, the components of the spectral fatigue damage method that rely on the estimation of the PSD are defined. In chronological order, starting from the one-sided spectral density, the spectral moments and a number of further variables depending on the spectral moments are introduced.

The spectral moments are defined based on the one-sided spectral density, which considers the stress loading in a structure as a realization of a random process $X(t)$ [BT05]. In the following, the deterministic load history $y(t)$ is interpreted as one realization of this stochastic process. Using the notation of the windowed periodogram by Welch introduced before, the one-sided PSD is given by

$$W_X(\omega) = \begin{cases} 2\ \hat{\Phi}_W(\omega),\ 0 < \omega < \infty \\ \hat{\Phi}_W(0),\ \omega = 0 \end{cases}$$

[BT05]. In the case where the periodogram is chosen instead of the windowed variant, the same definition holds analogously for $\hat{\Phi}_p$.

In general, the motivation for introducing the one-sided spectral density lies in the physical interpretation of real-valued signals, for which negative and positive frequencies contain redundant information. The one-sided PSD therefore represents the contribution of positive frequencies while preserving the total spectral energy [GB20; BT05].

Spectral moments are defined based on the following relation:

$$\lambda_n = \int_0^\infty \omega^n\ W_X(\omega)\, d\omega. \tag{3.10}$$

The spectral moments relevant for the considered fatigue damage estimation are therefore given by the variances of the process $X(t)$ and its derivatives $\dot{X}(t)$ and $\ddot{X}(t)$, denoted by σ_X^2, $\sigma_{\dot{X}}^2$, and $\sigma_{\ddot{X}}^2$, respectively, and defined as

$$\lambda_0 = \sigma_X^2, \qquad \lambda_2 = \sigma_{\dot{X}}^2, \qquad \lambda_4 = \sigma_{\ddot{X}}^2.$$

The higher-order moments λ_2 and λ_4 are thus associated with the variances of the first and second derivatives of the process and characterize its velocity and acceleration behavior, respectively [GB20]. These relations establish the direct connection between spectral properties of the process and its time-domain characteristics.

In the paper [BT05], the rate of mean upcrossings ν_0 and the peak rate ν_p are defined as

$$\nu_0 = \frac{1}{2\pi}\sqrt{\frac{\lambda_2}{\lambda_0}}, \qquad \nu_p = \frac{1}{2\pi}\sqrt{\frac{\lambda_4}{\lambda_2}}. \tag{3.11}$$

These quantities describe fundamental characteristics of the stochastic process in the time domain, as they quantify the expected number of level crossings and local extrema per unit time.

Dirlik, on the other hand, defines the number of peaks for his fatigue spectral method as

$$\nu_p = \sqrt{\frac{\lambda_4}{\lambda_2}}$$

in [Dir85], which is equivalently stated in the later review [DB21] by Dirlik and Benasciutti, but differs by a scaling factor compared to the formulation by Benasciutti and Tovo [BT05]. This difference arises from the distinction between a normalized rate and the total number of peaks considered in the respective formulations.

The bandwidth parameters are defined as

$$\alpha_1 = \frac{\lambda_1}{\sqrt{\lambda_0 \lambda_2}}, \qquad \alpha_2 = \frac{\lambda_2}{\sqrt{\lambda_0 \lambda_4}}. \tag{3.12}$$

These parameters characterize the spectral width of the process and therefore provide a measure of how narrow-band or broad-band the underlying signal is, which is of central importance for the applicability and accuracy of spectral fatigue damage estimation methods.

For both methods of Dirlik and Tovo–Benasciutti, the expected fatigue damage rate is expressed in terms of the respective probability density function of stress amplitudes, as described in the following.

Damage Rate

In general, the expected rainflow damage rate is obtained by integrating the m-th power of the stress amplitude s, weighted by the corresponding probability density function $p(s)$ of the spectral method, and scaling the result by the ratio ν_a/C, where $\nu_a = \nu_p$ denotes the rate of occurrence of counted cycles [BT05]. Formally, this yields

$$D_{RFC} = \frac{\nu_a}{C} \int_0^{+\infty} s^m p(s)\, ds. \tag{3.13}$$

The parameters m and C are derived from the S–N curve [Hai06], which, for $C = N_D s_D^m$, is typically expressed in the equivalent form of Equation 2.10

$$N(s_{a,l}) = C\, s_a^{-m}.$$

Both parameters are determined experimentally for different materials, as described, for instance, in Chapter 2 of [Hai06]. Consequently, the damage D_{RFC} is evaluated under the assumption of linear damage accumulation with respect to the S–N curve, thereby establishing the connection between the statistical description of the load process in the frequency domain and the resulting fatigue damage in the time domain [BT05].

3.1.3. Dirlik's Spectral Method

The Dirlik spectral method is defined by a set of parameters derived from the spectral moments of the process. The standard deviation of the process is given by

$$\sigma_X = \sqrt{\lambda_0}.$$

Based on this, the normalized stress amplitude is defined as

$$Z = \frac{s}{\sigma_X},$$

which is used here in the formulation of the rainflow amplitude density in Equation 3.20 [BT05].

The remaining parameters are obtained as fitting quantities derived from the spectral moments, as described by Benasciutti and Tovo [BT05]. The parameter x_m, introduced by Dirlik [Dir85] as the mean frequency and defined in (3.14), is given by the product of the bandwidth parameters defined in Equation 3.12 leading to

$$x_m = \frac{\lambda_1}{\lambda_0}\sqrt{\frac{\lambda_2}{\lambda_4}}. \tag{3.14}$$

Within the spectral method, Dirlik defines the rainflow amplitude probability density function (Equation 3.20) as a weighted sum of an exponential function, a Rayleigh function with variable Rayleigh parameter, and a standard Rayleigh function [Dir85]. The coefficients C_1, C_2, and C_3 determine the relative contributions of these components, where C_1 corresponds to the exponential term and C_2, C_3 represent scaling factors of the Rayleigh-type terms. Furthermore, the parameters C_1, C_2, C_3, Q, and R are functions of the bandwidth parameter α_2 (also referred to as the irregularity factor) and the mean frequency x_m, and are defined by (3.15)–(3.19)

$$C_1 = \frac{2(x_m - \alpha_2^2)}{1 + \alpha_2^2}, \tag{3.15}$$

$$C_2 = \frac{1 - \alpha_2 - C_1 + C_1^2}{1 - R}, \tag{3.16}$$

$$C_3 = 1 - C_1 - C_2, \tag{3.17}$$

$$Q = \frac{1.25(\alpha_2 - C_3 - C_2 R)}{C_1}, \tag{3.18}$$

$$R = \frac{\alpha_2 - x_m - C_1^2}{1 - \alpha_2 - C_1 + C_1^2}. \tag{3.19}$$

These parameters collectively determine the shape of the rainflow amplitude probability density function, which is described by

$$p_{RFC}^{DK}(s) = \frac{1}{\sigma_X}\left[\frac{C_1}{Q}e^{-\frac{Z}{Q}} + \frac{C_2 Z}{R^2}e^{-\frac{Z^2}{2R^2}} + C_3 Z e^{-\frac{Z^2}{2}}\right], \tag{3.20}$$

as given in [BT05], where $Z = s/\sigma_X$ denotes the normalized stress amplitude, and is later employed in Section 3.3.

By inserting this density into the damage rate expression of Equation 3.13, the Dirlik damage rate is obtained in closed form:

$$D_{DK} = \frac{\nu_p}{C}\sigma_X^m\left[C_1 Q^m \Gamma(1+m) + 2^{\frac{m}{2}}\Gamma\left(1 + \frac{m}{2}\right)(C_2|R|^m + C_3)\right], \tag{3.21}$$

where C and m are parameters of the S–N curve, and the Gamma function $\Gamma(\cdot)$ arises as the closed-form representation of integrals of the form $\int_0^\infty s^m e^{-s}\, ds = \Gamma(1+m)$. As a consequence of this closed-form representation, fatigue damage can be computed directly from spectral quantities without explicit rainflow cycle counting.

3.1.4. Tovo–Benasciutti's Spectral Method

The damage rate defined by Tovo and Benasciutti is the combination of the narrow-band approximation of rainflow damage D_{NB} and the approximate range-mean damage D_{RC}. Those damage rates are defined as

$$D_{NB} = \nu_0 \frac{1}{C} (\sqrt{2}\sigma_X)^m \Gamma\left(1 + \frac{m}{2}\right), \tag{3.22}$$

$$D_{RC} \approx \nu_p \frac{1}{C} (\sqrt{2}\sigma_X \alpha_2)^m \Gamma\left(1 + \frac{m}{2}\right) = D_{NB} \alpha_2^{m-1}. \tag{3.23}$$

Thus, the approximate damage by Tovo–Benasciutti is derived through

$$D_{TB} = b\, D_{NB} + (1-b)\, D_{RC}, \tag{3.24}$$

where b depends on the spectral density of the process and can therefore be approximated accordingly [BT05].

Through the validation of the methods described in Section 3.2, both the spectral estimations as well as the spectral fatigue damage methods, it becomes evident that Tovo–Benasciutti generates nearly identical results to Dirlik for an approximation of b that depends on the parameters introduced in Section 3.1.2.

3.2. Validation of Methods

This chapter is divided into two sections. The first section validates the PSD estimation via the windowed periodogram by Welch, while the second section builds on these results by considering the spectral methods of Dirlik and Tovo–Benasciutti, which depend on the PSD estimation.

3.2.1. Validation of the Power Spectral Density Estimation

The validation of the Welch windowed periodogram method introduced in Section 3.1.1 is performed using alternative numerical solutions [JS20]. To this end, the `scipy` library in `Python` provides a `welch` function for PSD estimation within its `scipy.signal` module, offering various window functions and parameter configurations, where the window functions considered and compared in this work are those introduced in Table 3.1.

An important aspect to investigate, as mentioned before, is the difference between implementing the Welch method directly and using the apodization approach for each window, as given in Table 3.2. In the theory of spectral analysis, the *apodization approach* refers to the design of appropriately chosen temporal windows with desirable spectral properties, which are intended in particular to reduce leakage effects in the periodogram while minimizing the loss in frequency resolution, and to achieve this with

only a marginal increase in computational effort [SM05].
Accordingly, the implementation of the Welch method, as presented in Subsection A.1.2, is validated against the reference implementation provided by the `scipy` library. In addition, it is investigated whether the apodization approach fulfills the expected properties, thereby enabling a direct comparison between theoretical expectations and practical implementations.

This comparison is carried out by applying both implementations to a synthetic test signal y_{syn} defined by

$$y_{\text{syn}}(t) = \sin(t) + \cos(t), \quad t \in [0, 2n\pi], \quad n \in \mathbb{N}$$

for increasing values of n, thereby generating signals with a correspondingly increasing number of data points.

Considering the first three plots of Figure 3.1, which compare the different implementations of the windowed periodogram by Welch using a Blackman window with 50% overlap, the following observations can be made.

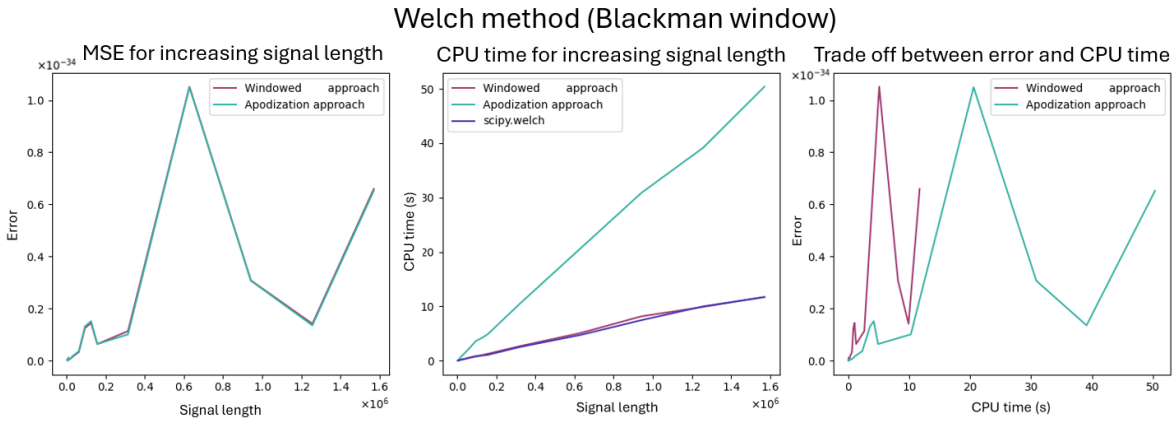


Figure 3.1.: Plots of the Welch method with a Blackman window: error for increasing signal length (left), CPU time for increasing signal length (centre), and error versus corresponding CPU time (right)

For increasing signal lengths up to approximately 1.6 million data points, the corresponding mean squared errors (MSE) with respect to the `scipy.welch` results increase, but remain sufficiently small. This indicates that both implementations provide a stable approximation of the PSD even for large signal sizes. Although the error of the apodization-based formulation is not significantly smaller than that of the direct implementation, this contrasts with the expectations derived from the theory, which suggest improved spectral properties of individual estimates due to reduced leakage.

In contrast, when considering the central processing unit (CPU) time for increasing numbers of data points, the results align with the theoretical expectations, as the runtime for the apodization approach is significantly higher and increases more rapidly than that of the direct implementation. In addition, it is observed that the implementation of the Welch method exhibits approximately the same, or only slightly higher, runtime compared to the Python reference implementation.

This implies that no advantageous trade-off is achieved: both methods yield similar accuracy, while the apodization approach is significantly more computationally expensive. Consequently, the windowed periodogram approach based on Welch is preferred.

This observation is particularly relevant in the context of the full apodization approach, where the window parameter β is determined via an optimization problem for each frequency component. Since even the non-adaptive formulation considered here leads to increased computational effort without improving accuracy, the additional cost associated with solving this optimization problem would further increase the computational complexity. This raises concerns regarding the practical applicability of the apodization approach in computationally demanding settings.

The findings described for the Welch periodogram with a Blackman window are consistent across all considered window types, including Hanning displayed in Figure 3.2, Hamming, and rectangular windows, as shown in Subsection A.1.4.

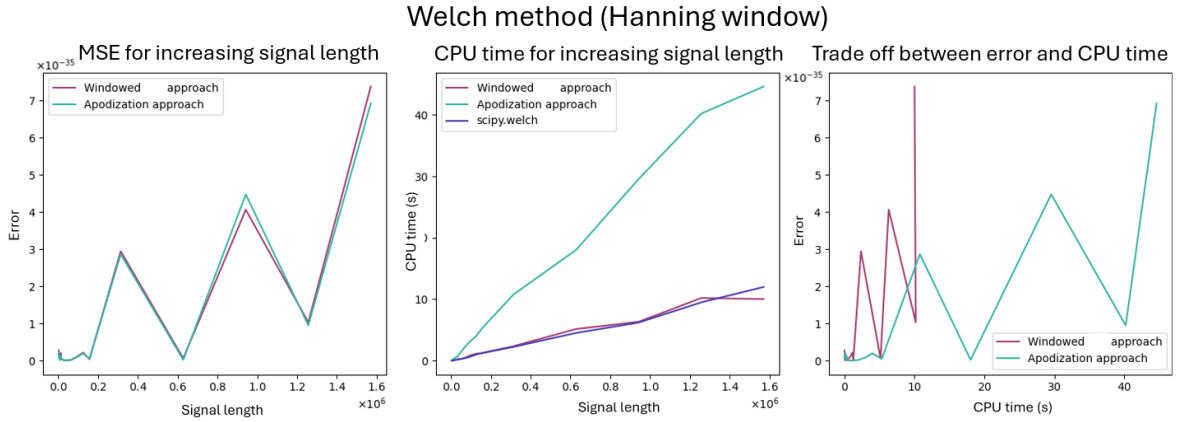


Figure 3.2.: Plots of the Welch method with a Hann window: error for increasing signal length (left), CPU time for increasing signal length (centre), and error versus corresponding CPU time (right)

However, aside from these findings and the decision to proceed with the Welch method as the preferred approach for numerical implementations, the apodization approach was applied here using the same window type on sequences of the signal with the same length as the window size chosen within the Welch method.
Note that Welch's method already applies windowing and averages windowed periodograms. In the present work, the apodization-based expressions are evaluated using fixed classical window functions. As a consequence, the resulting formulation reduces to an equivalent representation of the standard windowed periodogram estimator, up to numerical precision.

The apodization framework proposed by Stoica and Moses [SM05], however, is based on the adaptive selection of the window parameter β as a function of both frequency and data. This adaptive optimization step is not considered here. Therefore, the present comparison corresponds to the non-adaptive limit of the apodization approach.
This approach shows its advantages more clearly when the choice of the window is adapted to specific sequences of the signal for which it is constructed. This leads to an optimization problem of selecting the most suitable window for a corresponding part of the signal.

For applications in neural networks, this suggests that the additional complexity introduced by adaptive window selection must be carefully weighed against its potential benefit. In particular, since the non-adaptive formulation already increases computational cost without improving accuracy, the practical advantage of the fully adaptive apodization approach remains uncertain in this context.

In the following section, the spectral methods are validated using Welch's method as the underlying spectral estimate for the Dirlik and Tovo–Benasciutti approaches.

3.2.2. Validation of the Spectral Methods

In this section, the spectral methods by Dirlik and Tovo–Benasciutti introduced in Section 3.1.3 and Section 3.1.4 are validated by comparison with the RFC method. Since these spectral methods are intended to act as a surrogate for RFC in neural network training, their ability to accurately reproduce fatigue damage must be assessed. In addition, a brief investigation is conducted in Section 3.3 to determine whether the spectral estimates obtained using the windowed periodogram according to Welch, selected in the previous section, or the simple periodogram provide a better approximation of the cycle-counting characteristics of the Rainflow method.

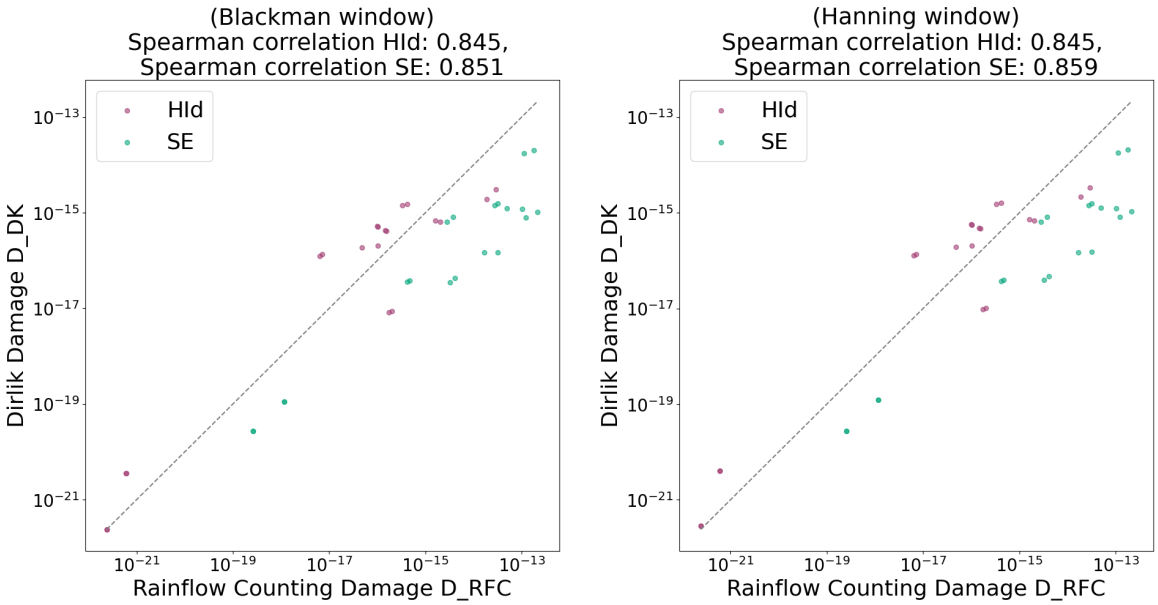
In this context, it is important to note that the spectral methods are formulated in the frequency domain, whereas RFC operates in the time domain, such that the resulting damage values are not directly comparable without further consideration. In particular, the damage obtained from spectral methods represents a constant damage rate D, which implies that the expected accumulated damage $\bar{D}$ over a time interval T is given by

$$\bar{D}_{DK}(T) = T\, D_{DK}, \qquad \bar{D}_{TB}(T) = T\, D_{TB}, \tag{3.25}$$

as described in [BT05, p. 289]. By incorporating this relation into the implementation of the spectral methods, the resulting damage obtained by the Dirlik and Tovo–Benasciutti approaches can be directly compared with the Rainflow damage used in the post-processing step of the virtual sensor.

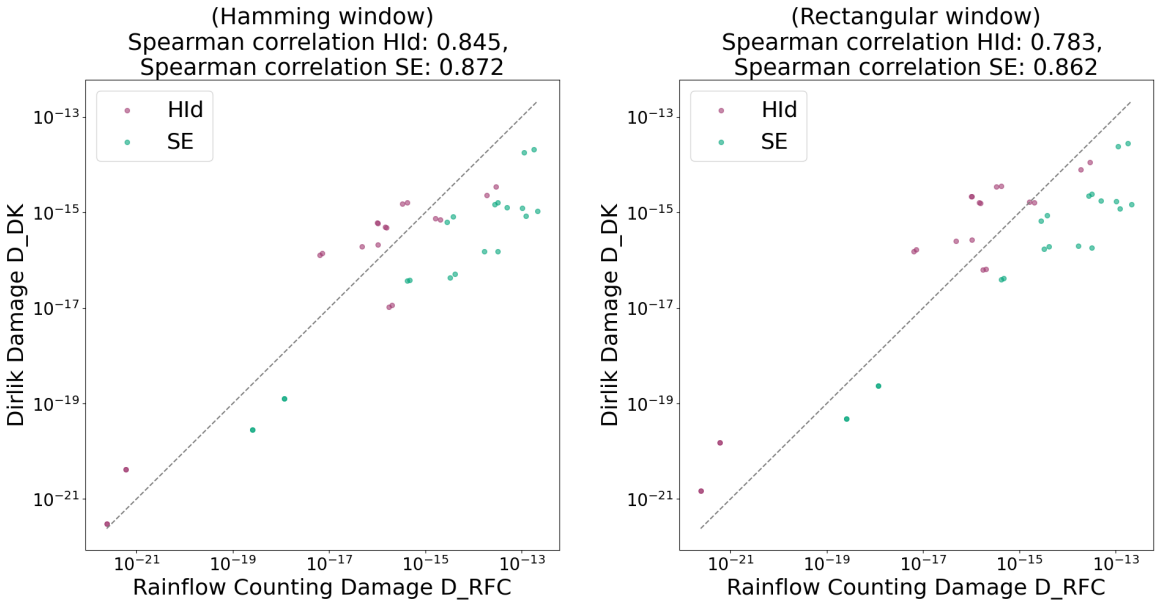
Dirlik's Method

For the validation, the damages of different time signals were computed, once with the RFC method in combination with Miner's rule and secondly with the Dirlik method. The total damages of the Dirlik method were then derived based on Equation 3.25 to make both methods comparable. Subsequently, scatter plots with the damages of each method were determined, as shown below in Figure 3.3. In addition to the plots, the Spearman correlation between the different damage sets was computed and displayed above each corresponding image.



(a) Blackman window

(b) Hanning window



(c) Hamming window

(d) Rectangular window

Figure 3.3.: Dirlik damage compared to Rainflow damage and their correlation for different window types used in the PSD estimation of Dirlik.

The different windows are four of the ones introduced in the previous sections. Among these window types used within the Dirlik method for the calculation of the spectral moments, and thus the PSD, the Hamming window shows the highest correlation, indicating that the reduced smoothing effect preserves the spectral characteristics relevant for fatigue damage estimation more accurately.

For the data set 'SE', the correlation is higher than for the other data set, which is an interesting observation in relation to the signal characteristics. Since the data set 'SE' contains signals with stronger irregularities and is therefore more non-stationary, the accuracy of the spectral fatigue damage estimate is theoretically expected to be reduced compared to RFC. It is therefore noteworthy that the corresponding damage values obtained from the spectral methods still show a similar behaviour compared to the Rainflow results, as 'SE' exhibits a strong linear correlation, even though the data set 'HId' is distributed more closely around the identity line and thus closer to the damage values derived via RFC. Making use of the various window functions implemented in `scipy`, their influence on the Dirlik damage is further illustrated in the plots in Section A.1.5.

In conclusion, these results show a strong agreement between the spectral method by Dirlik and the damage computed through RFC, confirming its suitability as a surrogate for fatigue damage estimation.

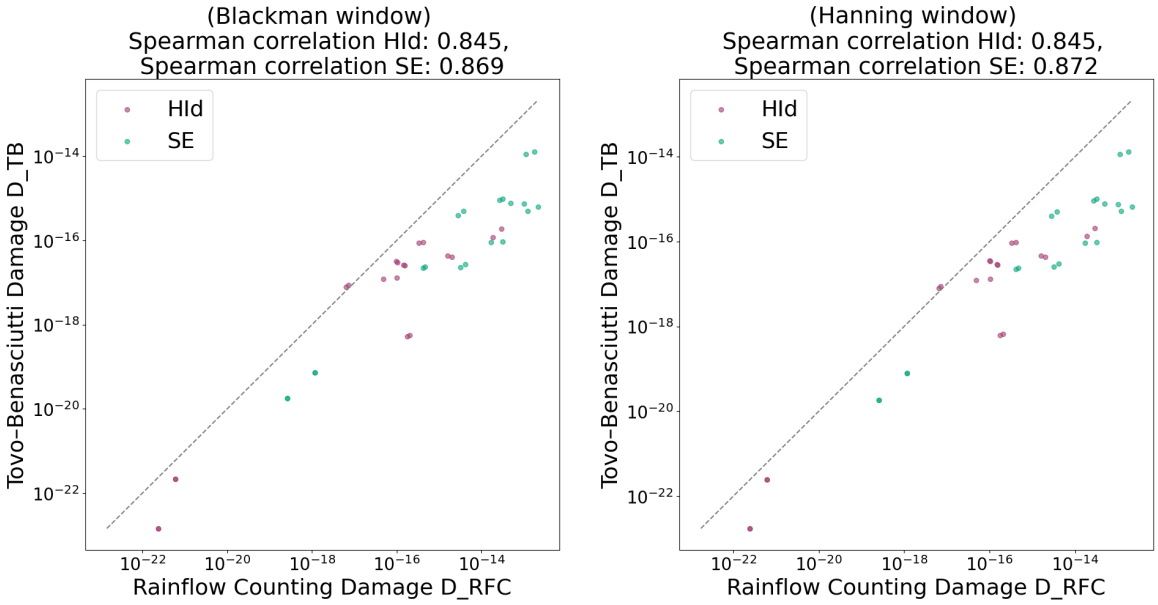
Tovo–Benasciutti's Method

The results for the spectral method by Tovo–Benasciutti [BT05] are obtained by following the same procedure as for the Dirlik method. In this context, the parameter b, which is determined as a function of the spectral moments, plays a central role. For certain window types, the resulting correlation with respect to the Rainflow method for the 'SE' data is higher than that obtained using the Dirlik method (Figure 3.4), although the corresponding alignment with the identity line is worse.

This is particularly noteworthy, as the gradient of the Tovo–Benasciutti method is generally simpler to derive than that of the Dirlik method, which may have a beneficial influence on computational complexity and runtime, especially with regard to integration into neural network training procedures. However, this simplification only holds for a specific choice of b. In the general case, b must be determined as a function of the spectral moments, which requires its calibration or computation depending on the data. For example, an approximation for b is given by

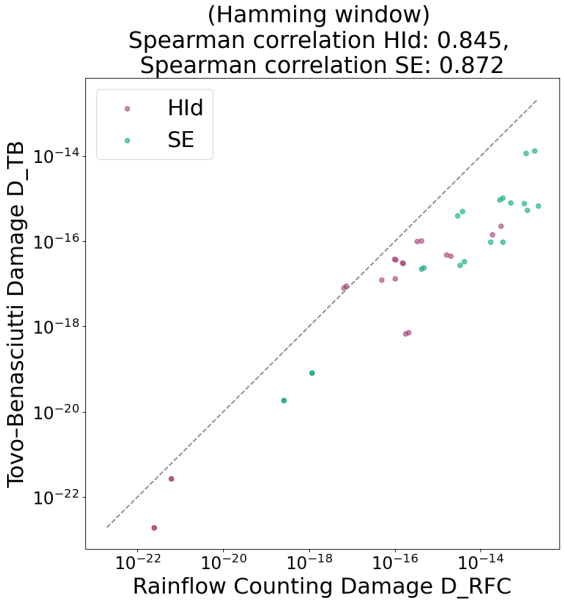
$$b_{\text{app}} = \frac{(\alpha_1 - \alpha_2)\left[1.112\Big(1 + \alpha_1\alpha_2 - (\alpha_1 + \alpha_2)\Big)e^{2.11\alpha_2} + (\alpha_1 - \alpha_2)\right]}{(\alpha_2 - 1)^2}, \tag{3.26}$$

as proposed by Benasciutti and Tovo [BT05].

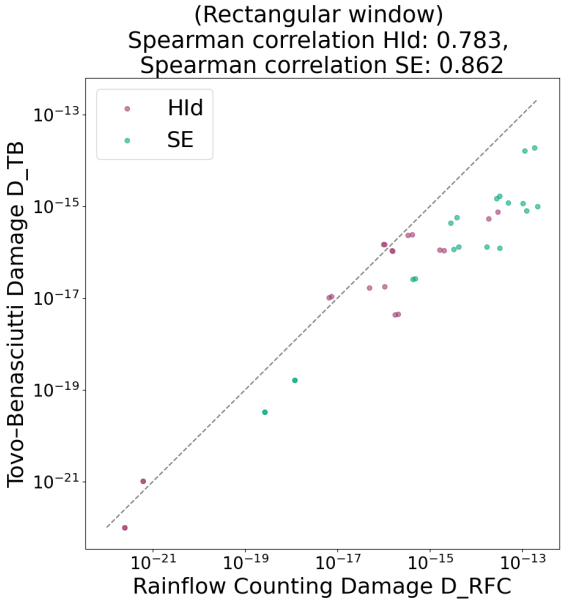


(a) Blackman window

(b) Hanning window



(c) Hamming window



(d) Rectangular window

Figure 3.4.: Tovo–Benasciutti damage compared to Rainflow damage and their correlation for different window types within the PSD of Tovo–Benasciutti.

As mentioned before, the observations are similar to those obtained for the Dirlik method. In particular, for the Tovo–Benasciutti method, the rectangular window provides among the best results, depicted in Figure 3.4. However, despite these observations, the Tovo–Benasciutti method is not further considered in this work. Although it exhibits comparable or locally improved correlation, the dependence of the parameter b on the spectral moments introduces additional complexity, especially for implementations requiring efficient and stable gradient computation.

3.3. Comparison of Periodogram to Welch and their Influence on the Fatigue Damage

Building on the validation of the methods and their implementations, another aspect of interest is whether the chosen spectral estimate in combination with the Dirlik method reflects the cycle-counting behavior of the RFC method. In particular, it is investigated to what extent the spectral approach reproduces the underlying cycle distributions identified by Rainflow, as examined in the following section by comparing the corresponding cycle density distributions obtained using the simple and the windowed periodogram.

In the theory of stationary processes, the Welch method is generally considered the preferable choice for estimating the PSD of a signal, as it reduces the spectral leakage of the periodogram [SM05]. With this in mind, the influence of the underlying PSD estimators, namely the Welch method and the unwindowed periodogram, on the resulting fatigue damage is investigated in the following.

Halfpenny compared the cycle counts of the Dirlik and Rainflow methods, among other aspects, by means of *range–mean histograms* [Hal99]. Following this approach, the range–mean histograms of the Rainflow method and the Dirlik method are generated, once using the Welch estimator and once using the simple periodogram.

Figure 3.5 displays the considered force–time history together with the corresponding PSD estimates obtained using Welch and the simple periodogram, while the associated range–mean histograms are shown in Figure 3.6.

Note that, although the example data sets ('SE', 'HId') correspond to force–time histories, the theoretical framework underlying the considered fatigue damage models is formulated in terms of stress. In the present context, the applied methods can be used equivalently, such that the terminology of stress–time histories is adopted where convenient.

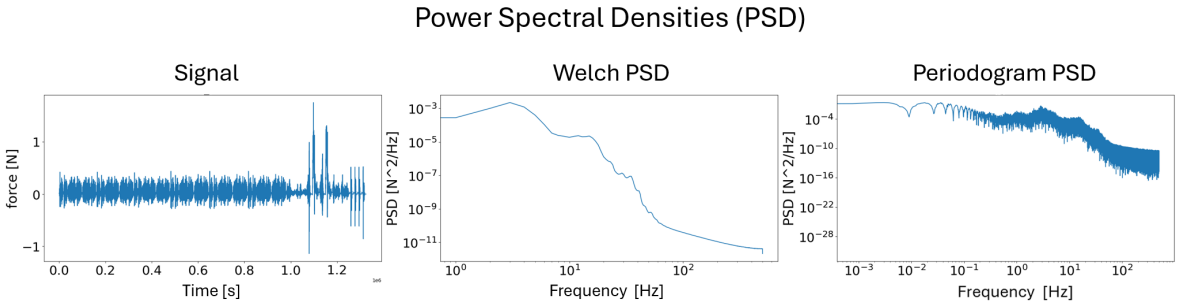


Figure 3.5.: Signal (left), PSD via Welch (middle), PSD via periodogram (right)

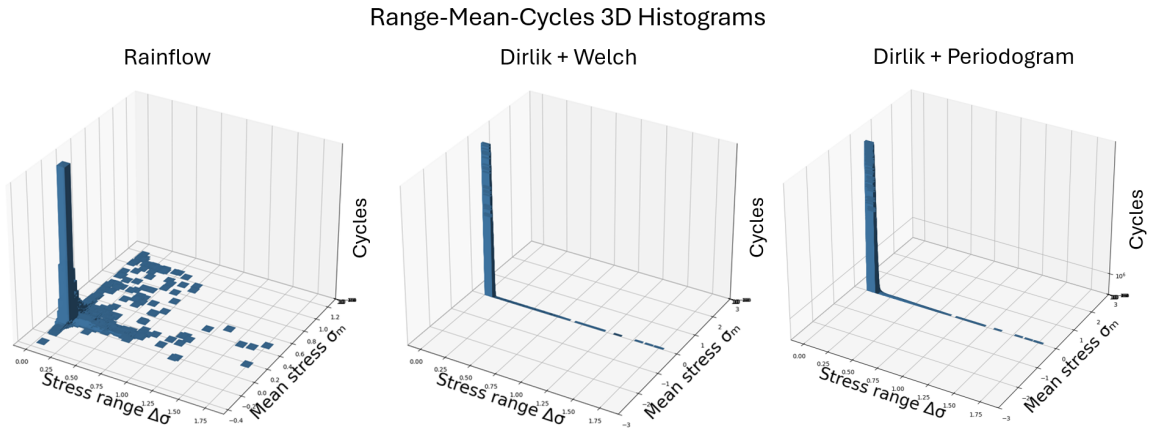


Figure 3.6.: Range-mean-cycle counts of Rainflow (left), Dirlik with underlying Welch (middle), Dirlik with underlying periodogram spectral estimate (right)

For the Rainflow method, the cycles and mean stresses are obtained directly by definition, whereas for the Dirlik method additional computations are required. Since Dirlik is a spectral method based on a probability density function depending on the parameters introduced in Subsection 3.1.3, the cycle amplitudes are described by

$$p_{RFC}^{DK}(s) = \frac{1}{\sigma_X}\left[\frac{C_1}{Q}e^{-\frac{Z}{Q}} + \frac{C_2 Z}{R^2}e^{-\frac{Z^2}{2R^2}} + C_3 Z e^{-\frac{Z^2}{2}}\right], \tag{3.27}$$

as introduced in Equation 3.20 (see also [BT05; Hal99]), where $Z = s/\sigma_X$ denotes the normalized stress amplitude. The expected number of stress cycles is then obtained as

$$N(s) = \nu_p \, T \, p_{RFC}^{DK}(s),$$

where $N(s)$ denotes the expected number of cycles of range s within time T [Hal99]. This formulation allows the reconstruction of cycle density distributions for the Dirlik method, depending on the chosen PSD estimator.

Since Dirlik does not provide information about the mean stress, the three-dimensional histograms are reduced to a two-dimensional representation for improved visualization, as shown in Figure 3.7. This figure reveals that, in contrast to the general expectations from spectral analysis, the simple periodogram (Figure 3.7(right)) reproduces the cycle distribution with respect to stress ranges more closely than the Welch method (Figure 3.7 (middle)). While this effect is less visible in the three-dimensional plots, it becomes apparent when focusing on the Dirlik-based representations. This indicates that the stronger smoothing introduced by the Welch method may reduce the fidelity of the underlying cycle representation, even though it improves the statistical properties of the PSD estimate.

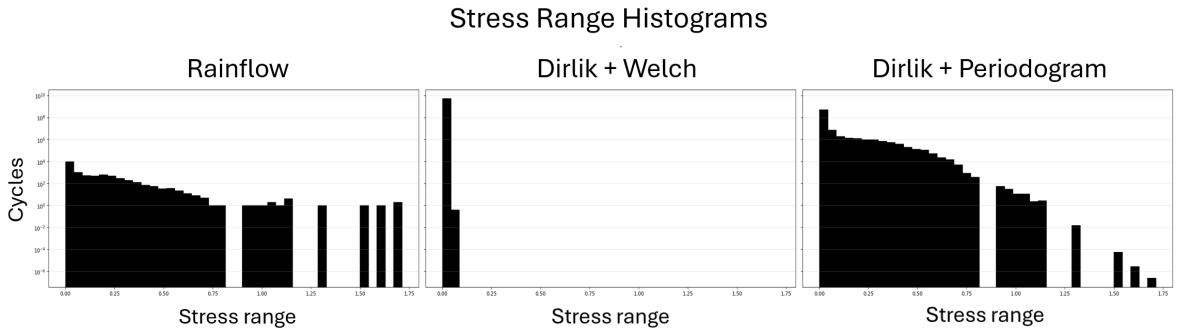


Figure 3.7.: Range-mean-cycle counts (2D) of Rainflow (left), Dirlik with underlying Welch (middle), Dirlik with underlying periodogram spectral estimate (right)

To further evaluate this observation, the resulting fatigue damages are directly compared with those obtained from the Rainflow method, as already performed in Section 3.2.

The Figures 3.8 and 3.9 show the fatigue damage for the entire data set as well as for the individual subsets.

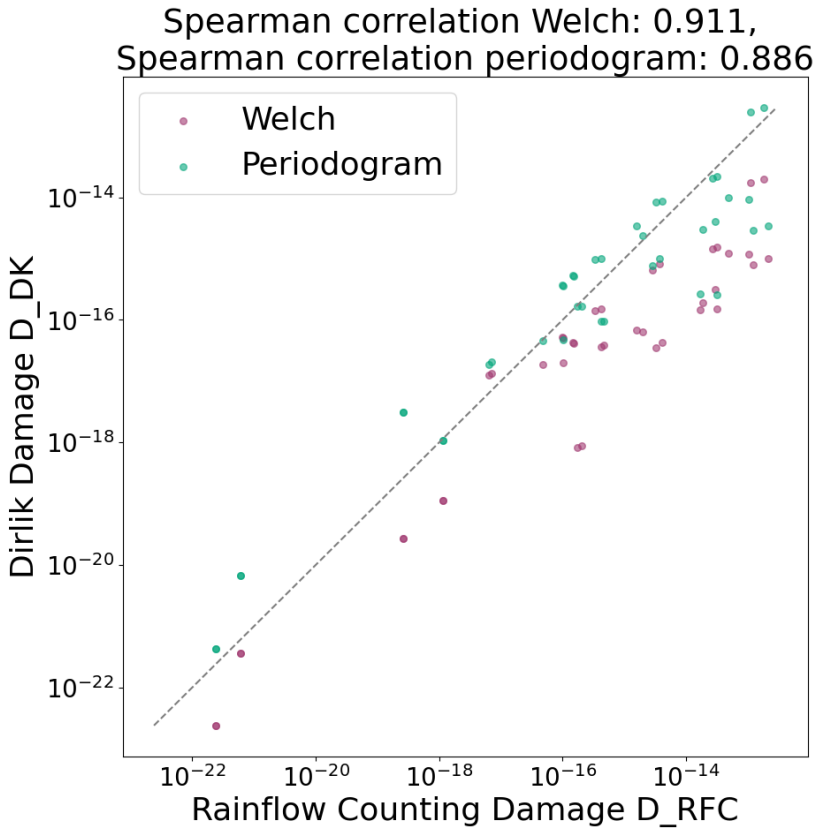


Figure 3.8.: Comparison of Welch- and periodogram-based Dirlik damage estimates relative to Rainflow counting

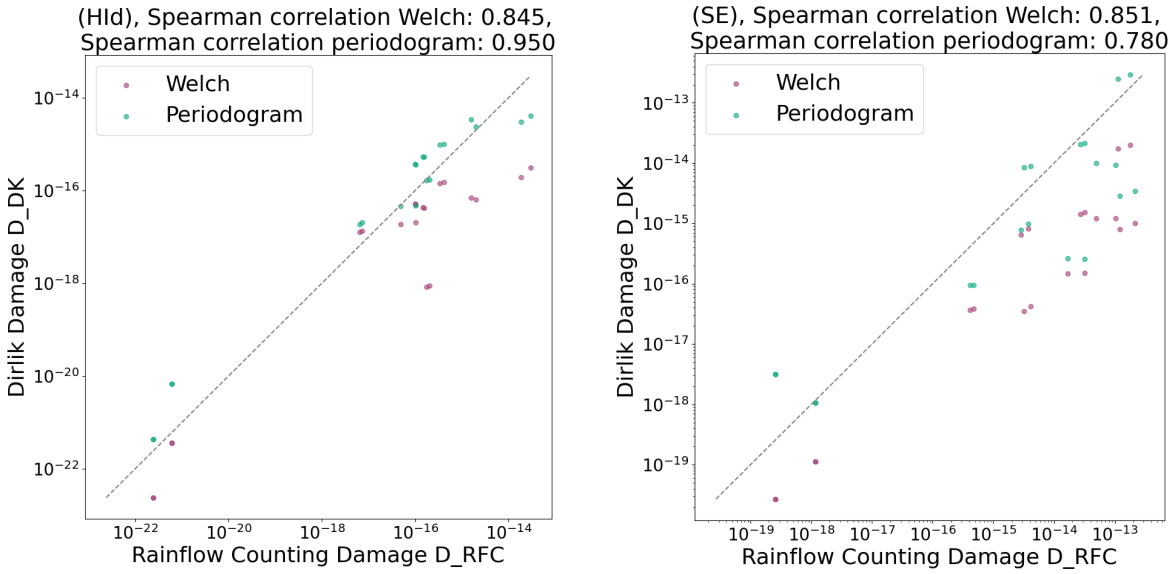


Figure 3.9.: Comparison of Welch- and periodogram-based Dirlik damage estimates relative to Rainflow counting, HId (left), SE (right)

For all cases except 'HId', the Welch-based Dirlik damage exhibits a higher correlation with Rainflow, while the periodogram-based Dirlik damage shows a closer alignment with respect to the absolute damage values. This indicates a trade-off between statistical stability and physical fidelity: while Welch improves correlation due to variance reduction, the periodogram appears to preserve cycle-related characteristics more accurately. Therefore, the question arises whether the periodogram provides a more suitable basis for the Dirlik method when used as a surrogate in a loss function. This aspect is further investigated in the following chapters.

The next chapter is dedicated to the theory behind deep learning frameworks, to further motivate the need for a differentiable fatigue damage surrogate, as well as to derive the gradients of all methods presented in this section. Subsequently, their integration into the loss function of the neural network is considered, including potential limitations such as vanishing gradients resulting from the physical formulation of the spectral methods.

4. Deep Neural Networks and Challenges of their Training

As motivated in the previous chapters, this work aims to incorporate fatigue damage into the training objective of a compact deep learning model in order to improve the fatigue damage obtained from the model predictions.

Training a deep learning model can be formulated as an optimization problem in which a loss function is minimized [Agg23]. Since this minimization is typically carried out using gradient-based methods (e.g., stochastic gradient descent), any quantity included in the training objective, defined by the loss, must be differentiable with respect to the network parameters. Consequently, a fatigue damage formulation that is integrated into the loss must be differentiable in order to be compatible with gradient-based training.

The commonly used fatigue damage estimation method of RFC cannot be directly used for this purpose, as it is computationally expensive and, more importantly, non-differentiable. The non-differentiability of RFC results from the discrete nature of the algorithm. In particular, the extraction of peaks and valleys, the division into levels, and the decision criteria of the four-point counting rule introduce discontinuities. Furthermore, the selection of extrema prevent a differentiable formulation.

Therefore, an alternative fatigue damage surrogate is required that is differentiable while still approximating the damage obtained from RFC, such as the spectral method by Dirlik introduced previously.

This chapter begins by formulating the learning objective through a mathematically precise definition of the loss function and its extension to incorporate fatigue damage. The following sections then summarize the deep learning background required for the subsequent gradient analysis.

4.1. The Loss Function

In this section, the learning objective is formalized and extended to account for fatigue damage. Let $\mathcal{X}$ denote the input space and Θ the space of neural network parameters (i.e., weights and biases). A neural network model defines a mapping

$$f : \mathcal{X} \times \Theta \to \mathbb{R}^N, \qquad (x; \theta) \mapsto f(x; \theta),$$

where $\theta \in \Theta$ are the trainable network parameters and $f(x; \theta)$ represent the predicted force–time history [GBC16]. The ground truth signal is denoted by $y \in \mathbb{R}^N$.

The fatigue damage surrogate is defined as a mapping

$$D : \mathbb{R}^N \to [0, 1],$$

where the critical damage value is normalized to unity [BT05]. Thus, $D = 0$ corresponds to the beginning of fatigue life, while $D = 1$ indicates failure.

To avoid ambiguity in notation, we distinguish a signal-level loss and a damage-level loss:

$$L_1 : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}, \tag{4.1}$$

$$L_2 : [0,1] \times [0,1] \to \mathbb{R}, \tag{4.2}$$

both of which are measured by the mean squared error (MSE) [GBC16], while later in Section 5.3 a different loss for L_2 is chosen.

The baseline model (without fatigue information) minimizes the signal-level loss

$$\mathcal{L}_y = L_1\Big(f(x;\theta), y\Big), \tag{4.3}$$

as illustrated in Figure 4.1. To incorporate fatigue damage into training, an additional

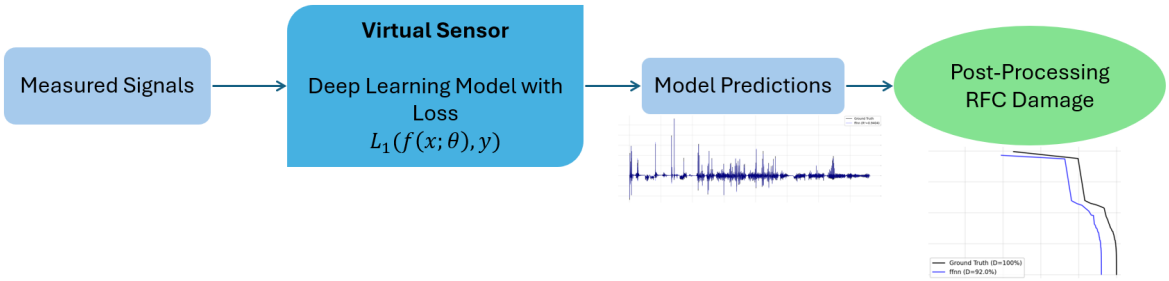


Figure 4.1.: Model without considering fatigue damage in the loss.

damage-based loss term is introduced:

$$\mathcal{L}_D = L_2\left(D(f(x;\theta)),\ D(y)\right). \tag{4.4}$$

The combined loss is then defined as

$$\mathcal{L}_{\text{total}} = \mathcal{L}_y + \alpha\,\mathcal{L}_D = L_1\left(f(x;\theta), y\right) + \alpha\, L_2\left(D(f(x;\theta)), D(y)\right), \tag{4.5}$$

where $\alpha \geq 0$ is a weighting parameter. The modified training setup is illustrated in Figure 4.2.

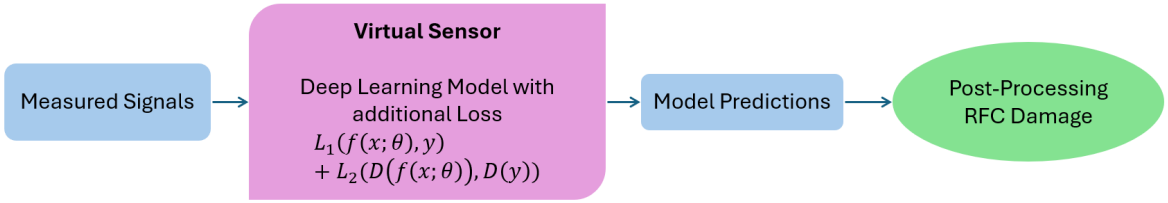


Figure 4.2.: Adapted model with an additional damage-based loss term.

Training corresponds to minimizing $\mathcal{L}_{\text{total}}$ with respect to θ

$$\min_{\theta\in\Theta} \quad \mathcal{L}_{total}(\theta) := L_1(f(x;\theta), y) + \alpha L_2(D(f(x;\theta)), D(y))$$

using gradient-based methods. The required gradients are computed via backpropagation and propagated through the network layers. In the present setting, the inclusion of a physics-informed loss may introduce very small gradient contributions, which can act as a bottleneck in the gradient chain and lead to vanishing-gradient-like behavior. The implications of this effect and the corresponding gradient analysis are addressed in the next chapter.

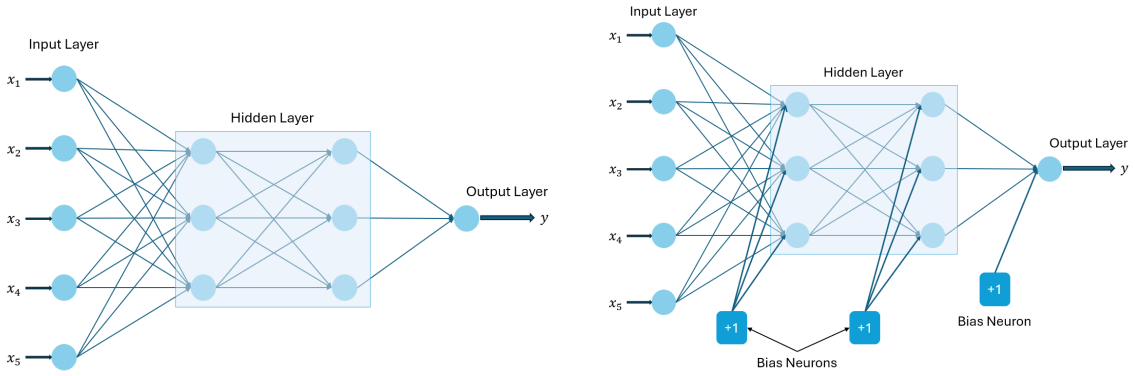
4.2. Backpropagation in Deep Neural Networks

Before considering the algorithms for forward and backward propagation, it is necessary to define the overall structure of multilayered neural networks in order to understand the origin of the underlying formulas.

In this section, the architecture of such networks and the connections between different layers are introduced, including activation functions and the distinction between networks with and without bias. This is followed by the presentation of the forward and backward propagation algorithms, with a particular focus on the computation of loss gradients.

4.2.1. Architecture of Multilayered Neural Networks

Multilayered (deep) neural networks consist of groups of units, called layers. These layers are typically arranged sequentially, where each layer represents a function of the previous one [GBC16]. The structure of such neural networks is illustrated in Figure 4.3.



(a) (b)

Figure 4.3.: Neural Networks, (a) wihout bias neurons, (b) with bias neurons (compare [Agg23])

The input vector x is transformed into the output $f(x;\theta)$ through a sequence of intermediate representations (forward propagation). Here, $h^{(k)}$ denotes the hidden activations, obtained by applying the activation function ψ to the pre-activation values $a^{(k)}$. The pre-activation $a^{(k)}$ is given by a linear combination of the activations from the previous layer, together with the weights $W^{(k)}$ and biases $b^{(k)}$ [GBC16].

The initial state is defined as

$$h^{(0)} = x \qquad \text{(initial input)}.$$

For a network with l layers, the following relations hold:

$$h^{(1)} = \psi(a^{(1)}) \qquad \text{(input to hidden layer)} \tag{4.6}$$

$$h^{(k)} = \psi(a^{(k)}) \qquad \text{(hidden to hidden layer)} \tag{4.7}$$

$$f(x;\theta) = h^{(l)} = \psi(a^{(l)}) \qquad \text{(hidden to output layer)}, \tag{4.8}$$

where the pre-activations are given by

$$a^{(k)} = b^{(k)} + W^{(k)} h^{(k-1)}.$$

For networks without bias, the condition $b^{(k)} = 0$ holds for all $k \in \{1, \ldots, l\}$ (Figure 4.3a), whereas for networks with bias the general form above applies (Figure 4.3b).

4.2.2. Forward- and Backwardpropagation

After introducing the basic architecture of deep neural networks, where the depth is given by the number of layers l, backpropagation is explained. Furthermore, the discussion of backpropagation provides the motivation for using a differentiable fatigue damage formulation, as gradients are required for training neural networks.

Backpropagation is a key component in the training process and is therefore discussed to analyze the challenges arising during training. One of the main challenges is the vanishing and exploding gradient problem, which will be described in more detail in the next chapter. Since the spectral fatigue damage method considered in this thesis is included in the modified loss function (Equation 4.5), it directly influences the training process. Consequently, it must be examined whether the derived and implemented gradient of the Dirlik method leads to vanishing or exploding gradient behavior.

Backpropagation is solely a method for computing the gradient, whereas a separate optimization algorithm, such as stochastic gradient descent (SGD), is used to perform learning based on the gradient obtained via backpropagation [GBC16].

Before backpropagation comes into play, a forward pass through the neural network is performed, as described in the following pseudocode, in which the for loop of Algorithm 2 displays the way the hidden layers interact with each other, as described in Equations 4.6 - 4.8.

Algorithm 2 Forward Propagation [GBC16]

Function: 'Computes target output and its loss'
Input: $W^{(i)}$, $i \in \{1, \ldots, l\}$, weight matrices; $b^{(I)}$, $i \in \{1, \ldots, l\}$, bias parameters; x, input; y, target output
Output: $f(x, \theta)$ output, $\mathcal{L}$ cost (loss)
$h^{(0)} = x$
for $k = 1, \ldots, l$ **do**
    $a^{(k)} = b^{(k)} + W^{(k)} h^{(k-1)}$
    $h^{(k)} = \psi(a^{(k)})$
end for
$f(x; \theta) = h^{(l)}$
$\mathcal{L} = L(f(x, \theta), y) + \lambda \Omega(\theta)$

After a forward pass like this, all layers beginning at the output layer are propagated backwards as shown in Algorithm 3 of the backward propagation. Essential about this algorithm is that only the gradients of the loss function $\mathcal{L}$ are derived, while the update of the states is performed through an optimization algorithm like stochastic gradient descent or Adam.

Algorithm 3 Backward Propagation for Algorithm 2 [GBC16]

Function: 'Computes the gradient of the loss/cost function'
$g \leftarrow \nabla_{f(x,\theta)}\mathcal{L} = \nabla_{f(x,\theta)} L(f(x,\theta), y)$
for $k = l, l-1, \ldots, 1$ **do**
    $g \leftarrow \nabla_{a^{(k)}}\mathcal{L} = g \circ \psi'(a^{(k)})$
    $\nabla_{b^{(k)}}\mathcal{L} = g + \lambda \nabla_{b^{(k)}}\Omega(\theta)$
    $\nabla_{W^{(k)}}\mathcal{L} = g\, {h^{(k)}}^T + \lambda \nabla_{W^{(k)}}\Omega(\theta)$
    $g \leftarrow \nabla_{h^{(k-1)}}\mathcal{L} = {W^{(k)}}^T g$
end for

For the loss function of the total loss Equation 4.5 for the here considered neural network holds

$$\mathcal{L}_{total}(\theta) = L_1(f(x;\theta), y) + \alpha L_2(D(f(x;\theta)), D(y))$$

The gradient of the loss function with respect to the hidden states is obtained by stepwise application of the chain rule. First, the gradient of the loss with respect to θ is computed:

$$\frac{\partial \mathcal{L}_{total}}{\partial \theta} = \left(\frac{\partial L_1(f(x;\theta), y)}{\partial f(x;\theta)} + \alpha \frac{\partial L_2(D(f(x;\theta)), D(y))}{\partial D(f(x;\theta))} \frac{\partial D(f(x;\theta))}{\partial f(x;\theta)} \right) \frac{\partial f(x;\theta)}{\partial \theta} \tag{4.9}$$

The next step involves combining the gradient in Equation 4.9 with the derivative of $f(x;\theta) = \psi(a^{(l)})$ with respect to the activation:

$$\frac{\partial \mathcal{L}_{total}}{\partial a^{(k)}} = \frac{\partial \mathcal{L}_{total}}{\partial \theta} \circ \psi'(a^{(k)})$$

By continuing this procedure, the gradients are propagated through the network by repeated application of the chain rule. Thus, the gradient of the loss with respect to the hidden state is given by:

$$\nabla_{h^{(k-1)}}\mathcal{L} = {W^{(k)}}^T \frac{\partial \mathcal{L}_{total}}{\partial a^{(k)}} \tag{4.10}$$

$$\Rightarrow \nabla_{h^{(k-1)}}\mathcal{L} = {W^{(k)}}^T \left[\left(\frac{\partial L_1(f(x;\theta), y)}{\partial f(x;\theta)} + \alpha \frac{\partial L_2(D(f(x;\theta)), D(y))}{\partial D(f(x;\theta))} \frac{\partial D(f(x;\theta))}{\partial f(x;\theta)} \right) \frac{\partial f(x;\theta)}{\partial \theta} \right] \circ \psi'(a^{(k)}).$$

The gradients of the loss with respect to weights and bias are derived respectively and are of the form shown in Algorithm 3.

The gradient of the total loss with respect to the hidden activations is lastly retained for the subsequent analysis of vanishing and exploding gradients in the next chapter.

5. Gradient Analysis of the Spectral Fatigue Loss Function

5.1. Gradient of Fatigue Damage for Neural Network Training

After introducing and validating several methods for fatigue damage estimation in Chapter 3, their differentiability is investigated. The motivation for this analysis arises from the gradient-based training of neural networks discussed in Chapter 4. In this context, backpropagation computes the gradient of the loss function, whose gradient chain, illustrated in Equation 4.10, includes the derivative of the fatigue damage estimate with respect to the model output $f(x;\theta)$. These gradients are subsequently used in gradient-based optimization algorithms to update the parameters θ to minimize the loss [GBC16].

This requirement necessitates replacing the fatigue damage obtained via RFC by a formulation that admits a closed-form and differentiable representation, as discussed in the previous chapter. Among the methods that provide reliable fatigue damage estimates in comparison to RFC and experimental observations [BT05], the previously introduced spectral methods are particularly suitable. In the following, their derivatives are derived and analyzed with respect to their numerical implementation in terms of accuracy and computational efficiency.

Initially, the gradients of the two presented spectral estimators are derived with respect to the target signal, i.e., the load history $y(t)$. For this purpose, both the Welch method and the periodogram were introduced in analytical form in Section 3.1.1. Then subsequently, the fatigue damage models by Dirlik and Tovo–Benasciutti are differentiated with respect to $y(t)$.

5.1.1. Gradient of Periodogram Spectral Estimate

Since the subsequent methods are differentiated with respect to the target signal y, and the spectral methods depend on the estimation of the PSD via their spectral moments, the differentiation propagates down to the PSD itself. Therefore, the gradient of the PSD with respect to $y(t)$ has to be derived in order to guarantee that the overall formulation is fully differentiable.

This requirement is also the motivation for the analytical implementation and validation of the Welch and periodogram method in the previous Chapter 3. Since standard `Python` packages do not provide gradients of PSD estimators, and a purely numerical differentiation for long signals would be computationally too expensive.

The gradient of the periodogram (Equation 3.1) used to estimate the PSD is obtained

by rewriting the complex exponential via Euler's identity and using $|z|^2 = z\bar{z}$ to obtain a real-valued representation, as shown in Section A.3.1, the gradient then follows from the product rule and is given by

$$\frac{\partial \hat{\Phi}_p(\omega)}{\partial y(t)} = \frac{2}{N}((\sum_{t=1}^{N} \tilde{y}(t)\cos(\omega t))(\sum_{t=1}^{N} \frac{\partial \tilde{y}(t)}{\partial y(t)}\cos(\omega t)) + (\sum_{t=1}^{N} \tilde{y}(t)\sin(\omega t))(\sum_{t=1}^{N} \frac{\partial \tilde{y}(t)}{\partial y(t)}\sin(\omega t))).$$

Here, $\tilde{y}(t)$ denotes the zero-mean representation of the full data signal $y(t)$, i.e., $\tilde{y}(t) = y(t) - \mathbb{E}[y(t)]$, defined analogously to the windowed segments in Equation 3.3.

5.1.2. Gradient of Welch Spectral Estimate

The gradient of the Welch method (Equation 3.2 - Equation 3.6) is defined as follows. First, the gradient of the Welch PSD estimate is obtained through

$$\frac{\partial \hat{\Phi}_W(\omega)}{\partial y(\hat{t})} = \frac{1}{S}\sum_{j=1}^{S} \frac{\partial \hat{\Phi}_j(\omega)}{\partial y(\hat{t})}.$$

Applying the chain rule, the next step is the derivative of the windowed periodogram. By applying the Euler representation in Equation 3.4, the complex exponential is decomposed into sine and cosine terms, and the squared magnitude eliminates the imaginary part. This transformation is carried out analogously to Section A.3.1, yielding an expression that is differentiable with respect to the signal

$$\begin{aligned}\frac{\partial \hat{\Phi}_j(\omega)}{\partial y(\hat{t})} = \frac{2}{MP}\Big(&(\sum_{t=1}^{M} v(t)\tilde{y}_j(t)\cos(\omega t))(\sum_{t=1}^{M} v(t)\tfrac{\partial \tilde{y}_j(t)}{\partial y(\hat{t})}\cos(\omega t)) \\ &+ (\sum_{t=1}^{M} v(t)\tilde{y}_j(t)\sin(\omega t))(\sum_{t=1}^{M} v(t)\tfrac{\partial \tilde{y}_j(t)}{\partial y(\hat{t})}\sin(\omega t))\Big).\end{aligned} \tag{5.1}$$

Here, each windowed segment $\tilde{y}_j(t)$ is a function of the full signal $y(t)$ through the segmentation of the data (Equation 3.2), such that the derivative is taken with respect to the global signal. The windowing introduces a distinction between the discrete time points t within each segment and the global discrete time points $\hat{t}$ of the full signal. Then, the gradient of the zero-meaned data segments $\tilde{y}_j(t)$ is derived, resulting in

$$\frac{\partial \tilde{y}_j(t)}{\partial y(\hat{t})} = \begin{cases} 1 - \dfrac{1}{M}, & \text{if } (j-1)K + t = \hat{t} \\ -\dfrac{1}{M}, & \text{if } (j-1)K + t \neq \hat{t} \end{cases} \tag{5.2}$$

The index ranges are given by $t = 1, \ldots, M$, $j = 1, \ldots, S$, and $\hat{t} = 1, \ldots, N$. Consequently, the derivative of the one-sided PSD is given by the following cases based on the gradient of the periodogram by Welch:

$$\frac{\partial W_X(\omega)}{\partial y(\hat{t})} = \begin{cases} 2\,\frac{\partial \hat{\Phi}_W(\omega)}{\partial y(\hat{t})}, & 0 < \omega < \infty \\ \frac{\partial \hat{\Phi}_W(0)}{\partial y(\hat{t})}, & \omega = 0 \end{cases}$$

which follows directly from the product rule.

The gradient of the spectral method by Dirlik is derived in the following.

5.1.3. Gradient of Dirlik's Spectral Method

The first of the spectral methods for which the derivative is derived is the Dirlik method (Equation 3.21).

To simplify the differentiation, the fatigue damage function of the Dirlik method given in Section 3.1.3 is decomposed into the following product

$$D_{DK}(y) = f(y)\, g(y)\, h(y), \tag{5.3}$$

where the dependence on y arises implicitly through the spectral moments $\lambda_n(y)$, the resulting quantities $\nu_p(y)$, $\sigma_X(y)$, the Dirlik parameters $C_1(y)$, $C_2(y)$, $C_3(y)$, $Q(y)$, and $R(y)$ defined in Section 3.1.3. This decomposition into the factors $f(y)$, $g(y)$ and $h(y)$ defined in (5.4)-(5.6) allows for a structured application of the product rule, where each component is differentiated separately using the chain rule. The corresponding factors are defined as

$$f(y) = \frac{\nu_p(y)}{C} \tag{5.4}$$

$$g(y) = \sigma_X^m(y) \tag{5.5}$$

$$h(y) = C_1(y)Q^m(y)\Gamma(1+m) + 2^{\frac{m}{2}}\Gamma(1+\frac{m}{2})(C_2(y)|R(y)|^m + C_3(y)). \tag{5.6}$$

Applying the product rule yields

$$\frac{\partial D_{DK}(y)}{\partial y} = \frac{\partial f(y)}{\partial y}(g(y)\, h(y)) + f(y)(\frac{\partial g(y)}{\partial y}\, h(y) + g(y)\frac{\partial h(y)}{\partial y}) \tag{5.7}$$

For the factors $f(y)$ and $g(y)$, which admit a direct representation in terms of the spectral moments, the following derivatives are obtained:

$$\frac{\partial f(y)}{\partial y} = \frac{1}{C}\frac{\partial \nu_p(y)}{\partial y} = \frac{1}{C}\frac{1}{2\sqrt{\frac{\lambda_4(y)}{\lambda_2(y)}}}\frac{\frac{\partial \lambda_4(y)}{\partial y}\lambda_2(y) - \lambda_4(y)\frac{\partial \lambda_2(y)}{\partial y}}{\lambda_2^2(y)}$$

with $\nu_p(y) = \sqrt{\frac{\lambda_4(y)}{\lambda_2(y)}}$, and

$$\begin{aligned}\frac{\partial g(y)}{\partial y} &= \frac{\partial \sigma_X^m(y)}{\partial \sigma_X(y)} \cdot \frac{\partial \sigma_X(y)}{\partial \lambda_0(y)} \cdot \frac{\partial \lambda_0(y)}{\partial y} \\ &= m\ \sigma_X^{m-1}(y)\ \frac{\partial \sigma_X(y)}{\partial \lambda_0(y)} \cdot \frac{\partial \lambda_0(y)}{\partial y} \overset{\sigma_X = \sqrt{\lambda_0(y)}}{=} m\ \sigma_X^{m-1}(y)\ \frac{1}{2\sqrt{\lambda_0(y)}}\ \frac{\partial \lambda_0(y)}{\partial y}.\end{aligned}$$

For the remaining term $h(y)$, all expressions defining the Dirlik method in Section 3.1.3 are differentiated analogously. The resulting derivatives are summarized in Section A.3.3, as they involve multiple gradient chains down to the bandwidth parameters α_1, α_2 and the spectral moments λ_n with respect to y (Section 3.1.2).

Since the spectral moments $\lambda_n(y)$ depend on the PSD, the computation of these gradients ultimately requires differentiation of the PSD with respect to the input signal y, as derived in Section 5.1.1 and 5.1.2.

5.1.4. Gradient of Tovo–Benasciutti's Method

The derivative of the second spectral method by Tovo and Benasciutti (Equation 3.24) is obtained by differentiating the individual summands.

$$\frac{\partial D_{TB}}{\partial y} = b\,\frac{\partial D_{NB}}{\partial y} + (1-b)\,\frac{\partial D_{RC}}{\partial y}.$$

First, the derivative of the narrow-band approximation of the rainflow damage D_{NB} as defined in Equation 3.22 is considered with respect to the input signal y. The dependence on y arises implicitly through the parameters ν_0 and σ_X, which depend on the spectral moments λ_n

$$\begin{aligned}\frac{\partial D_{NB}}{\partial y} = \frac{\partial \nu_0}{\partial y}\,\frac{1}{C}(\sqrt{2}\,\sigma_X)^m \Gamma\Big(1+\frac{m}{2}\Big) \\ + \nu_0\,\frac{1}{C}\,m\,(\sqrt{2}\,\sigma_X)^{m-1}\left(\sqrt{2}\,\frac{\partial \sigma_X}{\partial y}\right)\Gamma\Big(1+\frac{m}{2}\Big).\end{aligned}$$

Analogously, the gradient of the range-mean damage D_{RC} is obtained as

$$\begin{aligned}\frac{\partial D_{RC}}{\partial y} = \frac{\partial \nu_p}{\partial y}\,\frac{1}{C}(\sqrt{2}\,\sigma_X\alpha_2)^m \Gamma\Big(1+\frac{m}{2}\Big) \\ + \nu_p\,\frac{1}{C}\,m\,(\sqrt{2}\,\sigma_X\alpha_2)^{m-1}\left(\sqrt{2}\,\frac{\partial \sigma_X}{\partial y}\alpha_2 + \sqrt{2}\,\sigma_X\,\frac{\partial \alpha_2}{\partial y}\right)\Gamma\Big(1+\frac{m}{2}\Big).\end{aligned}$$

To complete the gradient chain, the derivative of the parameter ν_0 is given by

$$\frac{\partial \nu_0}{\partial y} = \frac{1}{4\pi\sqrt{\frac{\lambda_2}{\lambda_0}}}\cdot\frac{\lambda_0\frac{\partial \lambda_2}{\partial y} + \lambda_2\frac{\partial \lambda_0}{\partial y}}{\lambda_0^2}.$$

The derivative of ν_p, representing the peak rate, follows analogously from the corresponding derivation in the previous section on the Dirlik method.

Remark: If the approximation of b defined in Equation 3.26 is used, its derivative with respect to $y(t)$ must also be computed, as b depends on the spectral moments, which are all derived from the PSD.

5.2. Validation of the Implementation

The implementation of the analytically derived gradients is validated by comparison with central finite differences and the automatic differentiation provided by `TensorFlow` [ABC+16]. The central difference approach is only feasible for signals of limited length, as its computational cost in terms of time and memory increases significantly otherwise.

For validation, randomly generated signals with values drawn from a uniform distribution in the interval $[-10, 10]$ and increasing signal length N were considered. The resulting errors between the analytically derived gradients and the reference methods are shown in Figure 5.1. Since the values of the gradients are small, as shown in Section 5.3.4, the error metrics are chosen accordingly to yield values that are representative for assessing differences between quantities of very small magnitude.

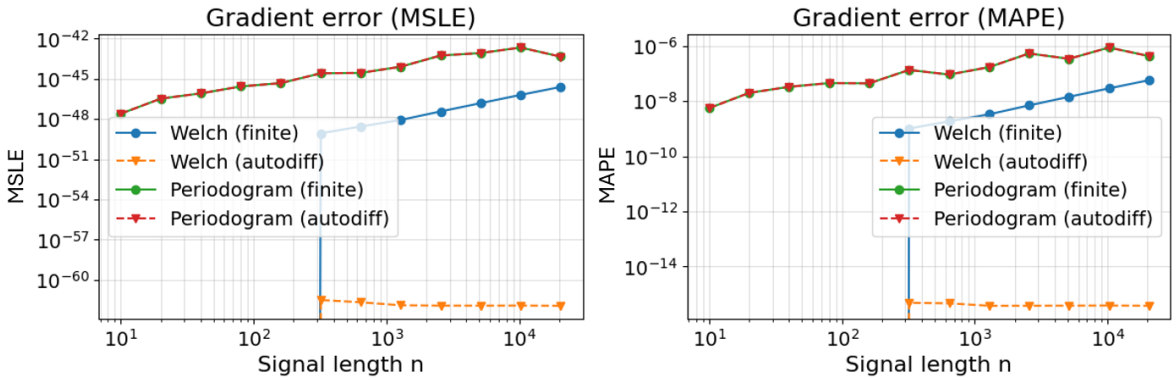


Figure 5.1.: Errors of the Dirlik gradient for increasing signal length compared to finite differences and automatic differentiation

As shown in Figure 5.1, the errors measured in terms of mean squared logarithmic error (MSLE) and mean absolute percentage error (MAPE) remain sufficiently small, indicating the correctness of the analytical gradient implementation. The gradient based on the periodogram exhibits slightly larger, but still comparable, errors relative to the reference methods. In contrast, the Welch-based gradient shows a larger deviation from TensorFlow's automatic differentiation. Nevertheless, for sufficiently long signals, both implementations provide accurate approximations of the true gradients.

An additional aspect of the validation concerns the computational efficiency of the different approaches. As discussed throughout this thesis, central finite differences are computationally expensive compared to analytically derived gradients. This behavior is confirmed by the results shown in Figure 5.2.

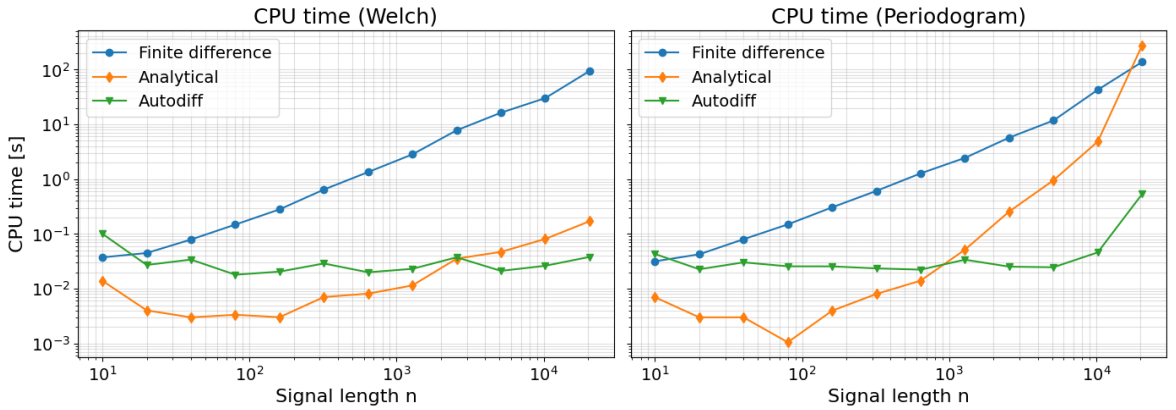


Figure 5.2.: Computation time of the Dirlik gradient for increasing signal length compared to finite differences and automatic differentiation

The runtime comparison demonstrates that central differences quickly become computationally prohibitive as the signal length increases. While for very long signals the analytical implementation of the periodogram-based gradient may approach or exceed the cost of finite differences, the analytical approach remains more efficient than TensorFlow's automatic differentiation for moderate signal lengths. Consequently, for shorter signals or training configurations with smaller batch sizes, the analytical

gradient provides a computationally advantageous alternative.

The validation is presented only for the Dirlik method, as the underlying spectral estimation and gradient computation form the critical components influencing the overall accuracy. Since both finite differences and automatic differentiation operate directly on the implemented function, agreement with these reference methods confirms the correctness of the entire gradient computation. Given the sensitivity of the Dirlik method to the spectral estimates, the observed small errors provide sufficient evidence for the validity of the implementation.

Remark: The computational efficiency of the Welch-based Dirlik gradient may be further improved by exploiting sparse tensor structures in the implementation of the Welch method. In particular, depending on the choice of window length and overlap, the corresponding tensor representation contains a large proportion of zero entries (cf. Subsection A.3.2).
Remark: To ensure a fair comparison of computational costs, a warm-up call is executed prior to each timing measurement. This eliminates the influence of TensorFlow's initial graph tracing and compilation overhead, which would otherwise bias the runtime evaluation.

5.3. The Vanishing and Exploding Gradient Problem

One of the challenges that make it difficult to train a neural network is the phenomenon of vanishing or exploding gradients. In the literature, this problem is widely referred to as the phenomenon where gradients shrink or grow exponentially during backpropagation [QWZ23]. A common approach to identify this behavior within a deep learning network is to examine the spectral radius $\rho(J)$ of the Jacobian J, i.e. the eigenvalue of J with largest magnitude, of the hidden-to-hidden mapping $(\frac{\partial h^{(k)}}{\partial h^{(k-1)}})$ [PMB13; GBC16], or the stronger condition of *dynamical isometry*, which considers the largest singular value of the Jacobian $\mu(J)$ [PSG17]. In both cases, training is expected to exhibit vanishing gradients if $\rho(J) < 1$ ($\mu(J) < 1$), and exploding gradients if $\rho(J) > 1$ ($\mu(J) > 1$) [PMB13; PSG17].

However, not only the gradient propagation through the hidden states influences the training process. The overall gradient of the loss function with respect to the weights and biases is determined by a chain of derivatives (see Equation 4.10), and extreme gradient behavior in any part of this chain can negatively affect training. Deep learning frameworks are formulated as optimization problems in which the loss function serves as the objective [GBC16]. Therefore, very small gradients (flat regions) or very large gradients (cliffs) can pose significant difficulties for the optimization process [GBC16]. As discussed in [KS24], such behavior may lead to non-robust predictive functions and unstable training.

The norm of the loss gradient provides an indication of whether the gradient is too small or too large [PMB13]. Since the gradient of the loss can be expressed as a chain of derivatives,

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial D} \frac{\partial D}{\partial h^{(l)}} \sum_{1 \leq k \leq l} \frac{\partial h^{(l)}}{\partial h^{(k)}} \frac{\partial h^{(k)}}{\partial \theta}, \tag{5.8}$$

each component of this product contributes to whether the gradient vanishes[1] or explodes [PMB13]. Consequently, even if the Jacobian satisfies $\rho(J) \approx 1$ ($\mu(J) \approx 1$), the overall gradient may still be dominated by other factors in the chain, which can negatively affect the training process [GBC16; GB10].

In the present context, the Dirlik method is expected to compute fatigue damage values in the range from 0 (no damage) to 1 (end of fatigue life), provided that the fatigue life of the considered component is not exceeded. For short input signals representing force–time histories over only a few seconds, the resulting damage values are therefore typically very small and close to 0. Consequently, the corresponding gradients are also expected to be small, while large variations that would indicate exploding gradients are unlikely.

Since the Dirlik method is introduced as an additional loss component to an already stable deep learning model for the virtual sensor, the Jacobian of the hidden states in the gradient chain (i.e., the term $\sum_{1\leq k\leq l} \frac{\partial h^{(l)}}{\partial h^{(k)}} \frac{\partial h^{(k)}}{\partial \theta}$) can be assumed to satisfy $\rho(J) \approx 1$. Therefore, the remaining factors in the gradient chain (Equation 5.8) must be analyzed to determine whether they induce very small or very large gradients, which could lead to vanishing or exploding gradient behavior, even though such effects are not expected from the network structure itself.

5.3.1. Lipschitz Continuity

Lipschitz continuity is used here as a tool to relate the magnitude of function variations to the magnitude of derivatives. In particular, bounds on Lipschitz constants for the Dirlik damage and for the loss function provide an indication of whether the associated landscape is locally flat (small gradients) or steep (large gradients).

Throughout this section, $\|\cdot\|_p$ denotes the p-norm on the relevant Euclidean spaces for a fixed $p \in [1, \infty]$. For matrices, $\|\cdot\|_p$ denotes the operator norm induced by $\|\cdot\|_p$, i.e.

$$\|A\|_p := \sup_{\|u\|_p=1} \|Au\|_p.$$

Definition 5.1 (Lipschitz continuity) A function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is Lipschitz continuous (K_0-Lipschitz) under a chosen p-norm $\|\cdot\|_p$ in the variable x if there exists a constant $K_0 \geq 0$ such that for all x_1 and x_2 in the domain of f ($\mathrm{dom}(f)$), the following inequality is always satisfied,

$$\|f(x_1) - f(x_2)\|_p \leq K_0 \|x_1 - x_2\|_p. \tag{5.9}$$

An equivalent definition holds for *local Lipschitz continuity* at a point $x_2 = x$, where (5.9) is required to hold for all x_1 in a neighborhood of x [QWZ23]. From the first-order condition of Lipschitz continuity, the Lipschitz constant K_0 can be determined as an upper bound on the gradient of the function f.

Lemma 5.2 (First-order condition for Lipschitz continuity [QWZ23])**.** *A continuous and differentiable function f is K_0-Lipschitz continuous if and only if the norm of its gradient is bounded by K_0,*

$$\|\nabla f(x)\|_p \leq K_0.$$

[1] A gradient close to 0 does not necessarily indicate an extremum of the loss function, but may also correspond to a flat region.

Thus, a valid Lipschitz constant is given by $K_0 = \sup_{x \in \mathrm{dom}(f)} \|\nabla f(x)\|_p$. The definition of Lipschitz gradient continuity is defined equivalently.

Definition 5.3 (Lipschitz gradient continuity) A function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is said to have a Lipschitz continuous gradient (or K_1-Lipschitz) under a choice of p-norm $\|\cdot\|_p$ in the variable x if there exists a constant $K_1 \geq 0$ such that for all x_1 and x_2 in the domain of f, the following inequality is always satisfied,

$$\|\nabla f(x_1) - \nabla f(x_2)\|_p \leq K_1 \|x_1 - x_2\|_p.$$

Lipschitz continuity provides a bound on the rate at which the gradient of the function can change, ensuring that the function's slope does not change too abruptly [QWZ23]. Similar to the first-order condition there exists the smoothness Lemma.

Lemma 5.4 (Smoothness Lemma [QWZ23])**.** *A continuous and twice differentiable function f is K_1-smooth if and only if*

$$\|\nabla^2 f(x)\|_p < K_1.$$

Accordingly, one may estimate K_1 by $K_1 = \mu_1(\nabla^2 f(x))$, where μ_1 denotes the largest singular value of the Hessian [HJ13].

In the following, these bounds are used to assess whether the gradients introduced by the Dirlik-based loss are expected to be extremely small (flat regions) and may therefore contribute to vanishing-gradient-like behavior during training.

5.3.2. Analysis of the Lower and Upper Bounds of the Welch Method

Part of deriving the Lipschitz constant of the Dirlik method is to derive a corresponding bound for the Welch method and then substitute it into the overall expression. To estimate such a bound, the gradient of the Welch method (Equation 5.1) is simplified in the following way, since the sine and cosine terms have the same structure. Thus,

$$\frac{\partial \hat{\Phi}_j(\omega)}{\partial y(\hat{t})} = \frac{2}{MP}\Big(A_{\cos} B_{\cos} + A_{\sin} B_{\sin}\Big),$$

with

$$A_{\cos} = \sum_{t=1}^{M} v(t)\, \tilde{y}_j(t) \cos(\omega t), \qquad B_{\cos} = \sum_{t=1}^{M} v(t)\, \frac{\partial \tilde{y}_j(t)}{\partial y(\hat{t})} \cos(\omega t),$$

$$A_{\sin} = \sum_{t=1}^{M} v(t)\, \tilde{y}_j(t) \sin(\omega t), \qquad B_{\sin} = \sum_{t=1}^{M} v(t)\, \frac{\partial \tilde{y}_j(t)}{\partial y(\hat{t})} \sin(\omega t),$$

$$A_{cs} = \begin{pmatrix} A_{\cos} \\ A_{\sin} \end{pmatrix}, \qquad B_{cs} = \begin{pmatrix} B_{\cos} \\ B_{\sin} \end{pmatrix}.$$

First, consider the term $A_{\cos} B_{\cos} + A_{\sin} B_{\sin}$, which equals the scalar product $\langle A_{cs}, B_{cs} \rangle$. By Cauchy–Schwarz (equivalently Hölder) one obtains

$$\Big|A_{\cos} B_{\cos} + A_{\sin} B_{\sin}\Big| \leq \|A_{cs}\|_2 \, \|B_{cs}\|_2 \leq \|v\|_2^2 \, \|\tilde{y}_j\|_2 \left\| \frac{\partial \tilde{y}_j}{\partial y(\hat{t})} \right\|_2. \tag{5.10}$$

Assuming the window is normalized such that $\|v\|_2^2 = MP$, it remains to bound $\|\tilde{y}_j\|_2$ and $\left\|\frac{\partial \tilde{y}_j}{\partial y(\hat{t})}\right\|_2$.

The gradient $\frac{\partial \tilde{y}_j}{\partial y}$ can be written in block form as

$$\frac{\partial \tilde{y}_j}{\partial y} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ A \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{N\times M}, \qquad A = \begin{pmatrix} 1-\frac{1}{M} & -\frac{1}{M} & \cdots & -\frac{1}{M} \\ -\frac{1}{M} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -\frac{1}{M} \\ -\frac{1}{M} & \cdots & -\frac{1}{M} & 1-\frac{1}{M} \end{pmatrix} \in \mathbb{R}^{M\times M}.$$

The factor $\left\|\frac{\partial \tilde{y}_j}{\partial y(\hat{t})}\right\|_2$ is constant with respect to the data, and we note that $\|\cdot\|_2$ denotes the induced spectral norm (the Frobenius norm $\|\cdot\|_F$ would be looser). One obtains

$$\left\|\frac{\partial \tilde{y}_j}{\partial y(\hat{t})}\right\|_F = \sqrt{M\left|1-\frac{1}{M}\right|^2 + M(M-1)\left|\frac{1}{M}\right|^2} = \sqrt{M-1},$$

$$\left\|\frac{\partial \tilde{y}_j}{\partial y(\hat{t})}\right\|_2 = \max \sigma\left(\frac{\partial \tilde{y}_j}{\partial y(\hat{t})}\right) = 1.$$

An upper bound for $\|\tilde{y}_j\|_2$ is obtained from the trivial estimate

$$\|\tilde{y}_j\|_2 \le \|y_j\|_2 \le \sqrt{M}\, \max_t |y_j(t)|. \tag{5.11}$$

Combining (5.10) and (5.11) yields the gradient bound

$$\left|\frac{\partial \hat{\Phi}_j(\omega)}{\partial y(\hat{t})}\right| \le 2\sqrt{M}\, \max_t |y_j(t)|.$$

For the one-sided PSD (density) this implies

$$\left|\frac{\partial W_X(\omega)}{\partial y(\hat{t})}\right| \le \frac{4}{fs}\sqrt{M}\, \max_t |y_j(t)|.$$

To determine bounds for the gradient of the Dirlik method, bounds for the Welch PSD itself (and hence for the spectral moments) are also required. Using again Cauchy–Schwarz,

$$\hat{\Phi}_j(\omega) = \frac{1}{MP}\left|\sum_{t=1}^{M} v(t)\tilde{y}_j(t)e^{-i\omega t}\right|^2 \le \frac{1}{MP}\Big(\|v\|_2\|\tilde{y}_j\|_2\Big)^2 = \|\tilde{y}_j\|_2^2 \le M\max_t |y_j(t)|^2,$$

and therefore

$$\hat{\Phi}_W(\omega) = \frac{1}{S}\sum_{j=1}^{S}\hat{\Phi}_j(\omega) \le M\max_t |y_j(t)|^2, \qquad W_X(\omega) \le \frac{2}{fs}\, M\max_t |y_j(t)|^2.$$

The derived bounds are loose general estimations of the magnitude of the spectral estimates obtained from the periodogram or, similarly, from the Welch approach. They show the dependence of the magnitude on the window length M, the sampling frequency f_s, and the maximum signal amplitude, i.e. $\max_t |y_j(t)|^2$, such that increasing M and $\max_t |y_j(t)|^2$ leads to an increase in the magnitude, while f_s acts as a scaling factor. Based on these bounds, deriving an upper bound for the Dirlik method provides the information required to assess the stated problem of vanishing and exploding gradients as described in Section 5.3. A similar derivation is carried out for the periodogram in the next section.

5.3.3. Periodogram Method for PSD Estimation

Equivalently, for the gradient of the periodogram approach one obtains

$$\left|\frac{\partial \hat{\Phi}_p(\omega)}{\partial y(\hat{t})}\right| \leq \left|\frac{2}{N}\right| \; \|\tilde{y}\|_2 \left\|\frac{\partial \tilde{y}}{\partial y(\hat{t})}\right\|_2 \leq \left|\frac{2}{N}\right| \sqrt{N} \max_t |y(t)|,$$

$$\Rightarrow \quad \left|\frac{\partial W_X(\omega)}{\partial y(\hat{t})}\right| \leq \left|\frac{4}{N\,fs}\right| \sqrt{N} \max_t |y(t)|.$$

Additionally,

$$|\hat{\Phi}_p(\omega)| = \frac{1}{N}\left|\sum_{t=1}^{N} y(t) e^{-i\omega t}\right|^2 \leq \frac{1}{N}\Big(N \max_t |y(t)|\Big)^2 = N \max_t |y(t)|^2,$$

$$|W_X(\omega)| \leq \frac{2N}{fs} \max_t |y(t)|^2.$$

Analogously to the bounds derived for the Welch approach, the bounds for the periodogram depend on the signal length N, the sampling frequency f_s, and the maximum signal magnitude.

5.3.4. Analysis of the Lipschitz Continuity of the Dirlik Method

Based on the upper bounds derived in the preceding sections, one could, in principle, attempt to obtain an analytical upper bound for the Dirlik method by substituting the PSD formulation together with all parameter functions of the Dirlik method into the final gradient expression and repeatedly applying norm inequalities (e.g., the triangle inequality and Hölder's inequality). Such a derivation would yield a *global* (worst-case) Lipschitz constant, i.e., a bound that is valid over the entire input domain. However, in practice this type of bound is expected to be highly conservative and therefore not informative for the training behavior of the neural networks considered in this work, where only a specific region of the input space is encountered during optimization.

For this reason, the focus is placed on the *local* behavior that is relevant for training, i.e., on the magnitude of gradients observed for typical predicted signals $f(x, \theta)$ throughout optimization. This empirical analysis provides practical insight into the effective influence of the Dirlik-based loss term during backpropagation, albeit without

guaranteeing a strict global bound. An attempt was made to derive an analytical bound using symbolic computation (in particular `SymPy`). However, due to the nested substitutions required by the Dirlik formulation, the resulting highly complex expressions, and the subsequent application of norm inequalities, no tractable closed-form bound could be obtained. This highlights the analytical intractability of the Dirlik gradient in the present setting. Consequently, the norm of the Dirlik gradient is assessed empirically, as illustrated in the following plots (Figure 5.3 - 5.5).

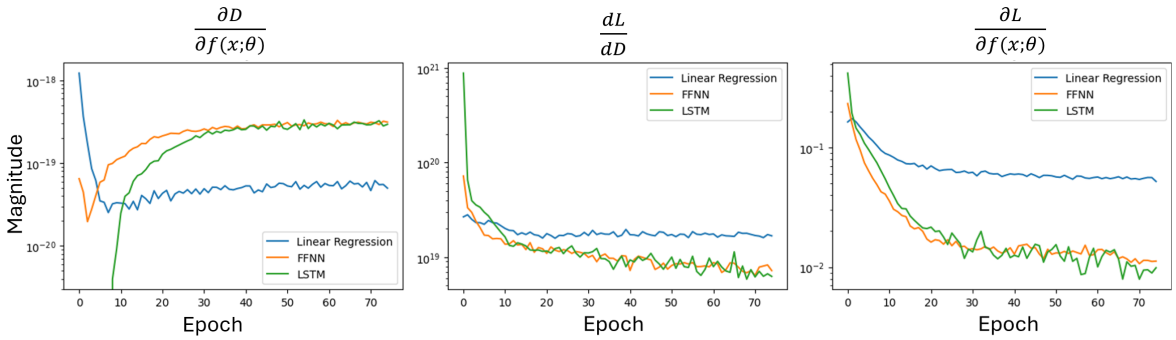


Figure 5.3.: Gradient norms across training epochs for three models (Linear Regression, FFNN, LSTM). Left: $\|\frac{\partial D}{\partial f(x;\theta)}\|_2$, norm of the Dirlik damage gradient with respect to the model output $f(x,\theta)$. Center: $\|\partial \mathcal{L}_D/\partial D\|$, magnitude of the derivative of the damage loss (here MSLE) with respect to the predicted damage. Right: $\|\frac{\partial \mathcal{L}_D}{\partial f(x;\theta)}\|$, norm of the damage-loss gradient with respect to $f(x;\theta)$.

Figure 5.3 shows gradient norms for three model classes: Linear Regression, a feedforward neural network (FFNN), and an LSTM. Across all three model types, the Dirlik gradient norm $\|\frac{\partial D}{\partial f(x;\theta)}\|_2$ remains extremely small (on the order of 10^{-19} to 10^{-18}) over many epochs. Such values are effectively near-zero and form a flat plateau rather than indicating convergence to a meaningful optimum. In particular, a near-zero gradient norm in this context should be interpreted as an extremely flat region of the loss landscape, which prevents the optimizer from making progress, rather than as evidence of having reached a true minimizer.

In contrast, the derivative of the damage loss with respect to the predicted damage, $\|\partial \mathcal{L}_D/\partial D\|$, can take substantially larger values depending on the chosen loss function. This behavior is a direct consequence of the chosen loss formulation, where the damage loss is defined using the mean squared logarithmic error (MSLE). As a result, the product structure implied by the chain rule yields damage-loss gradients with respect to the model output, $\|\frac{\partial \mathcal{L}_D}{\partial f(x;\theta)}\|$, that are still small but no longer numerically negligible. This provides a nonzero training signal and allows gradient-based optimizers to propagate updates through the network.

The case in which the damage loss is defined using a standard mean squared error (MSE) is shown in Figure 5.4.

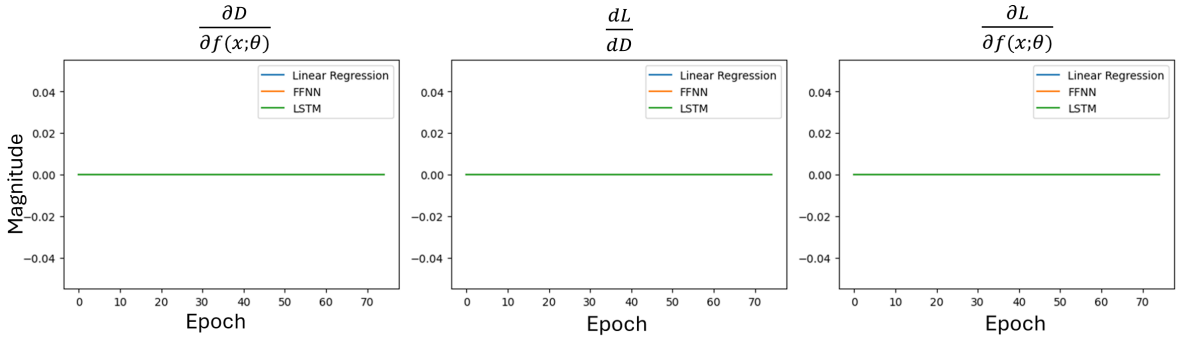


Figure 5.4.: Gradient norms across training epochs for three models (Linear Regression, FFNN, LSTM) using MSE as the damage loss. Left: $\|\frac{\partial D}{\partial f(x;\theta)}\|$. Center: $\|\partial \mathcal{L}_D/\partial D\|$ for MSE. Right: $\|\frac{\partial \mathcal{L}_D}{\partial f(x;\theta)}\|$.

As illustrated in Figure 5.4, the gradients relevant for the damage-based learning signal become effectively zero throughout training. This indicates that the optimizer receives no meaningful gradient information from the damage loss term, and training stagnates with respect to the damage objective. These results emphasize that an appropriate loss formulation (or rescaling) is necessary to avoid numerically vanishing gradients while preserving the physical meaning introduced by the Dirlik damage model.

The effect of using a MSLE is highlighted more directly in Figure 5.5.

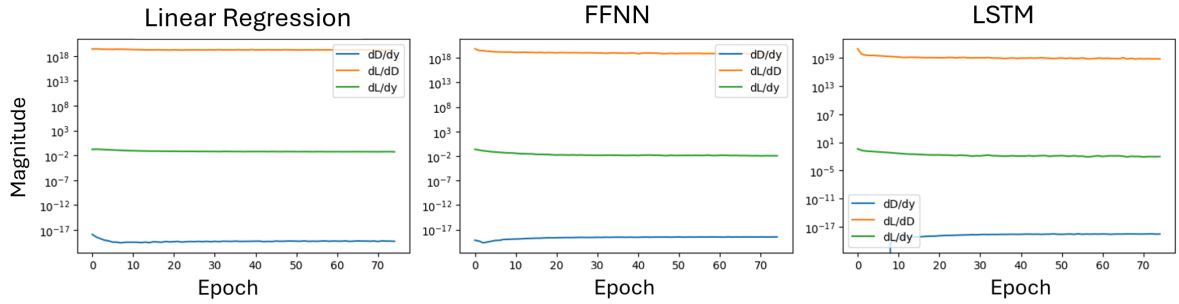


Figure 5.5.: Gradients during training under MSLE-based damage loss: $\|\frac{\partial D}{\partial y}\|$, $\|\partial \mathcal{L}_D/\partial D\|$, and $\|\frac{\partial \mathcal{L}_D}{\partial y}\|$ shown separately for Linear Regression (left), FFNN (center), and LSTM (right).

Figure 5.5 illustrates that MSLE provides a rescaling effect that increases $\|\partial \mathcal{L}_D/\partial D\|$ and thereby yields a nonzero $\|\frac{\partial \mathcal{L}_D}{\partial f(x;\theta)}\|$. While the Dirlik gradient $\|\frac{\partial D}{\partial f(x;\theta)}\|$ itself remains extremely small, the damage-loss gradient with respect to the network output is lifted to values that are small but practically usable, enabling gradient-based learning to proceed.

From the viewpoint of Lipschitz analysis, these empirical findings should be interpreted as follows. The first-order condition in Theorem 5.2 relates a Lipschitz constant to an upper bound on the gradient norm. In the region of the input space explored during training, the observed Dirlik gradient norms suggest that the *effective local* Lipschitz constant associated with the Dirlik contribution is extremely small. Consequently, the

[1]Here $y = f(x;\theta)$, while y elsewhere in Chapter 4 denotes the ground truth.

corresponding loss landscape is locally very flat. In the backpropagation chain (cf. Equation 5.8), the factor $\frac{\partial D}{\partial f(x;\theta)}$ therefore acts as a *bottleneck*: even if other terms in the chain are well behaved, multiplying by an almost-zero factor yields an overall gradient that is too small for optimization to progress. This constitutes a vanishing-gradient-like effect induced by the Dirlik-based loss component.

Notably, similar issues have been discussed in the context of physics-informed neural networks, where additional physics-based loss terms can flatten the loss landscape and lead to early stagnation of training. For example, Rathore et al. [Rat+24] report that composite loss functions may introduce flat regions and inhibit optimization progress, which aligns with the present observation that an unmodified Dirlik term can yield a near-zero gradient contribution and therefore requires careful loss design.

Beyond the gradient norm (which corresponds to first-order information), second-order information is captured by the Lipschitz constant of the gradient (smoothness) as defined in the smoothness Lemma 5.4. In principle, this constant provides insight into how rapidly the gradient can change locally. However, deriving analytical expressions for the Dirlik Hessian is infeasible due to the complexity of the closed-form formulation and its gradient. Consequently, second-order quantities must also be approximated numerically.

Since computing Hessians via finite differences is computationally intensive for full-length signals, representative signal segments are considered to obtain indicative results. While this does not yield a strict global characterization, it provides practical insight into the local curvature behavior of the Dirlik-based loss in scenarios relevant for training. Based on Theorem 5.4, approximate Lipschitz constants were obtained by computing Hessians for selected signal sequences from the datasets considered in the previous chapters and extracting their largest singular values. The resulting ranges are given by

$$\text{HId: } (10^{-30},\, 10^{-19}), \qquad \text{SE: } (10^{-32},\, 10^{-21}).$$

These values support the previously observed behavior of extremely small Dirlik gradients. In particular, such small smoothness constants indicate that the gradient changes only minimally in local regions of the input space, reinforcing the interpretation of a very flat loss landscape.

It can therefore be concluded that the choice of the damage loss function L_2 as defined in Section 4.1) is crucial. As demonstrated in the present case, selecting a MSLE effectively compensates for the extremely small gradients induced by the Dirlik term and enables stable gradient-based optimization.

6. Results

In this final chapter, the results of training runs of the deep learning models for the virtual sensor are presented and analyzed.

It is observed that the effectiveness of incorporating the additional damage-based loss depends strongly on the choice of hyperparameters, including the loss weighting, batch size, number of training epochs, and, in the case of the Welch method, parameters such as window size and overlap. A systematic hyperparameter optimization is beyond the scope of this work. However, representative parameter configurations were identified that allow for a meaningful evaluation of the effect of the additional loss term. Across the considered setups, incorporating fatigue damage directly into the training objective leads, in most cases, to an improvement in the prediction of cumulative fatigue damage during postprocessing. This confirms that integrating damage information into the optimization process can positively influence model performance.

The figures (6.1 - 6.3) display six different plots (a–f) considered for the virtual sensor and fatigue damage assessment and evaluation. The image in the upper left corner (a) illustrates the considered force-time history, comparing the ground truth and the prediction derived through the virtual sensor, while (b) depicts the class count of both. The plot of the normalized damage (c) is of main interest, alongside the cumulative fatigue damage plot (e), the latter showing the cumulative damage, and the former showing the normalized damage value per time t, derived in post-processing via RFC. In (d) and (f), the stress range pairs derived through RFC and the PSD are compared between prediction and ground truth.

Overall, the largest improvement is observed for the FFNN, while the LSTM model shows smaller but still noticeable gains. Linear regression models perform significantly worse compared to both neural network architectures, independent of the chosen loss, and are therefore not considered in further detail, as they are only able to capture linear relationships and thus fail for the nonlinear data considered here.

For the considered test data, the FFNN consistently provides the most reliable improvements. The employed architectures consist of two hidden layers with either $(8, 16)$ or $(16, 32)$ units. Depending on the choice of batch size and weighting parameter α, the improvement in cumulative fatigue damage ranges from approximately 5 to 30 percentage points, as shown in plots (c) and (e) of Figure 6.1–Figure 6.3. Similar improvements are observed for the normalized damage measures. Figure 6.1 illustrates in (a) that the prediction of the force-time history is improved, as the dominant peaks and valleys are no longer underestimated, as can be seen in Figure 6.3 for the loss without the additional damage term. This result is reflected in figures (b)–(f), depending on the RFC post-processing. In particular, the key damage-related quantities, whether in normalized or cumulative representation, demonstrate the improvement achieved by the combined loss.

These improvements are further illustrated in Figure 6.2 when compared to the

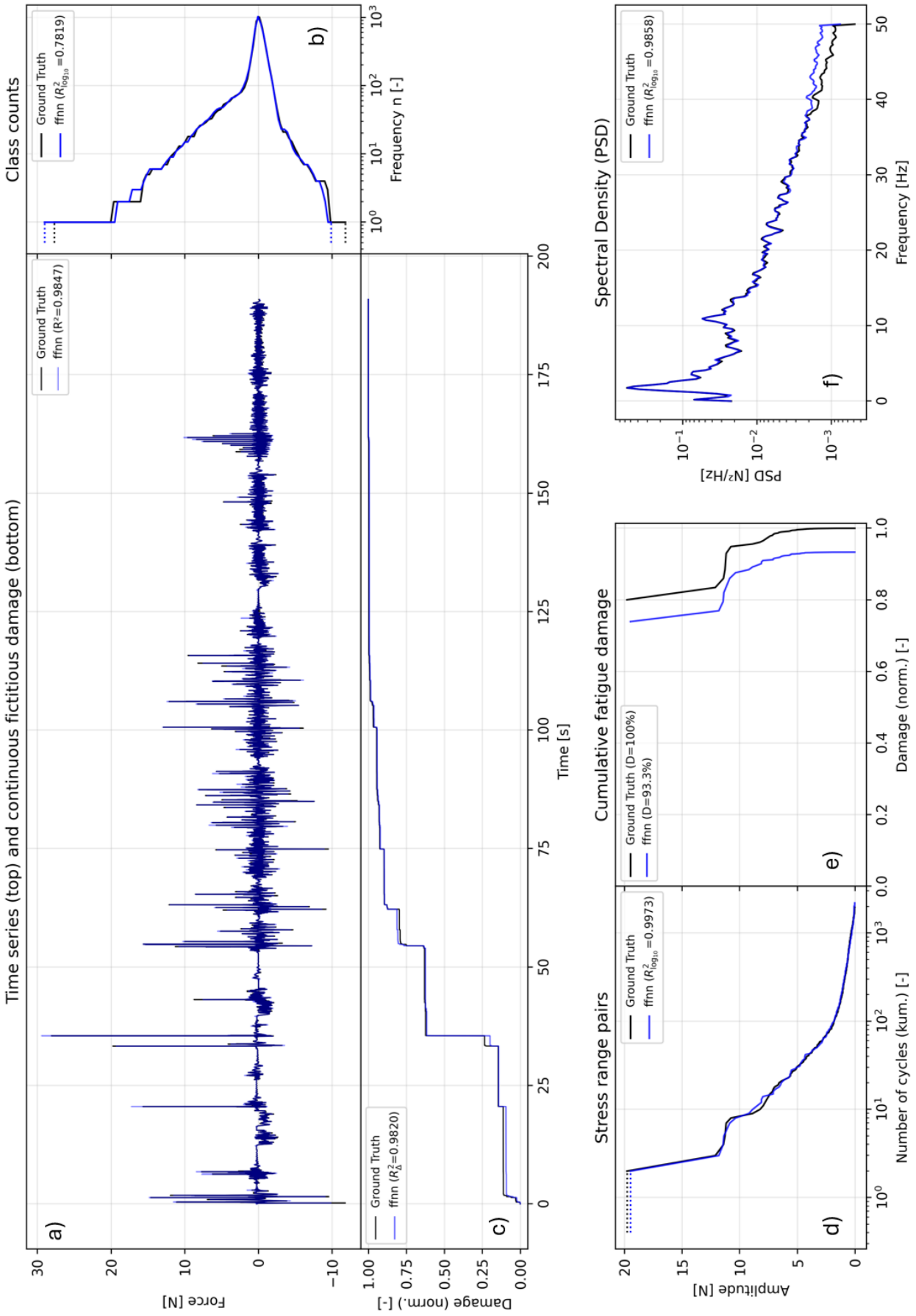


Figure 6.1.: Results for combined loss using the periodogram (FFNN)

corresponding results without damage-based loss shown in Figure 6.3.

Figure 6.3 illustrates the aforementioned underestimation of peaks and valleys in the force-time history in (a). This demonstrates the problem addressed in this thesis, namely that such underestimation prevents the correct computation of fatigue damage via RFC. Due to the underestimation, RFC cannot correctly compute the corresponding fatigue damage based on the prediction, as observed in (c) and (e). Consequently, a fatigue damage-based loss is incorporated into the training of the deep learning model to resolve this issue.

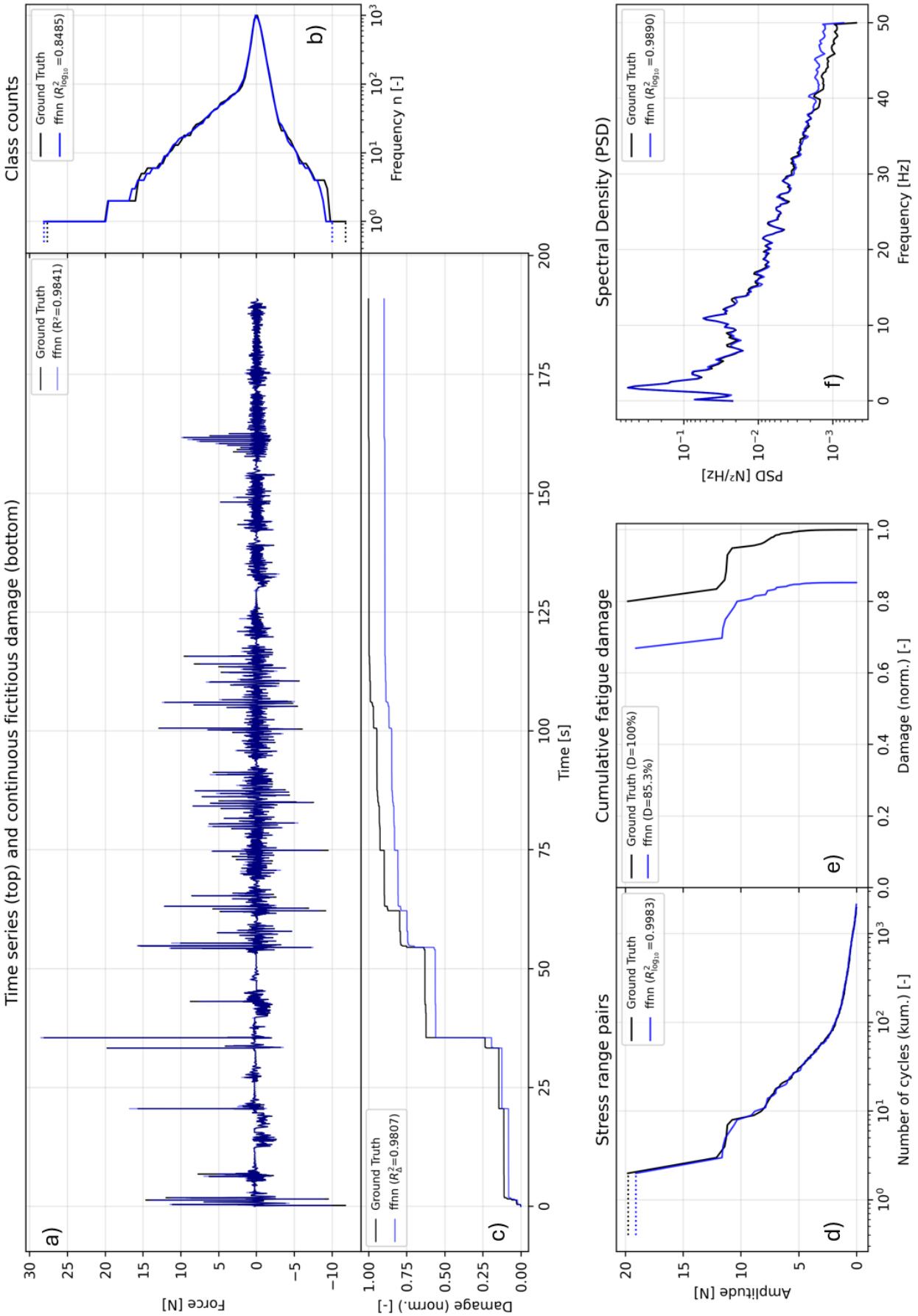


Figure 6.2.: Results for combined loss using the periodogram (extended run) (FFNN)

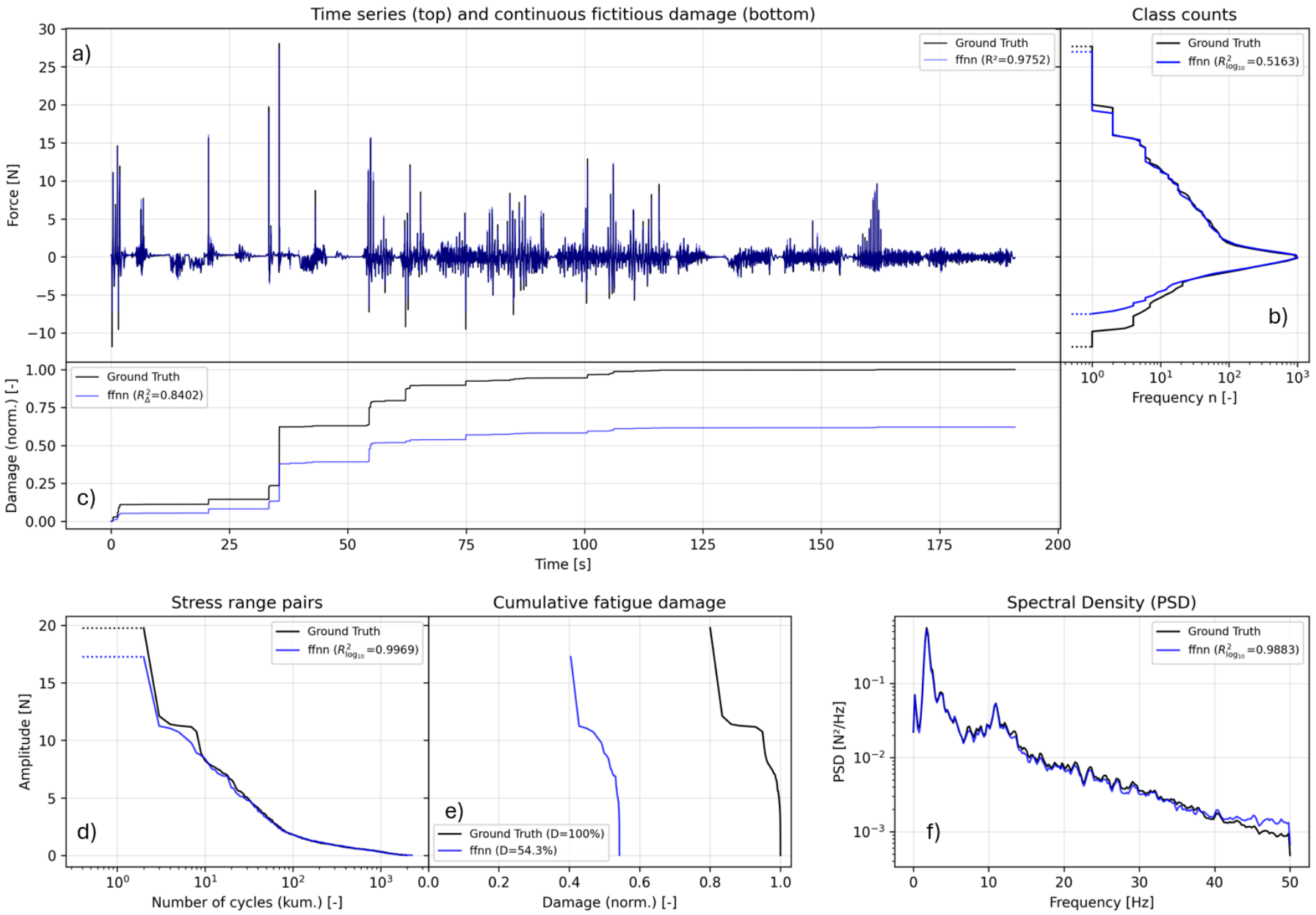


Figure 6.3.: Results without damage-based loss for periodogram (FFNN)

A minor dependence on the chosen spectral estimation method is observed. Using the Welch method yields slightly smaller improvements (up to approximately 25 percentage points), as shown in Figure 6.4 and Figure 6.5 compared to the periodogram (up to approximately 30 percentage points), shown in Figure 6.2 and Figure 6.3 . This observation may be related to the findings discussed in Section 3.3, where the periodogram estimate was found to resemble the Rainflow-counted cycles more closely. In particular, the damage estimates computed using the Dirlik method (in combination with the periodogram) showed a closer alignment with the Rainflow reference (identity line) than those based on the Welch estimate.

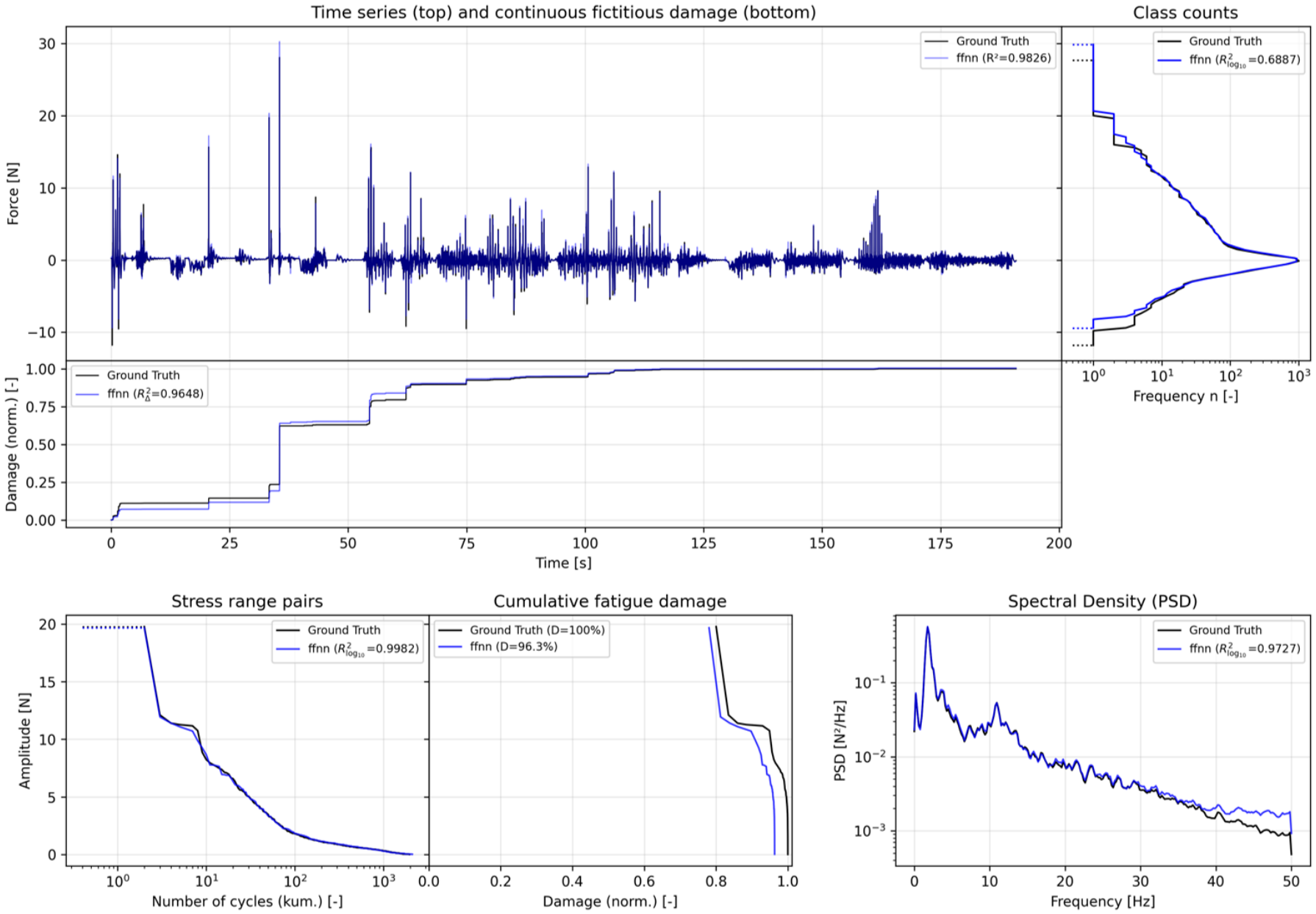


Figure 6.4.: Results for combined loss using Welch (FFNN)

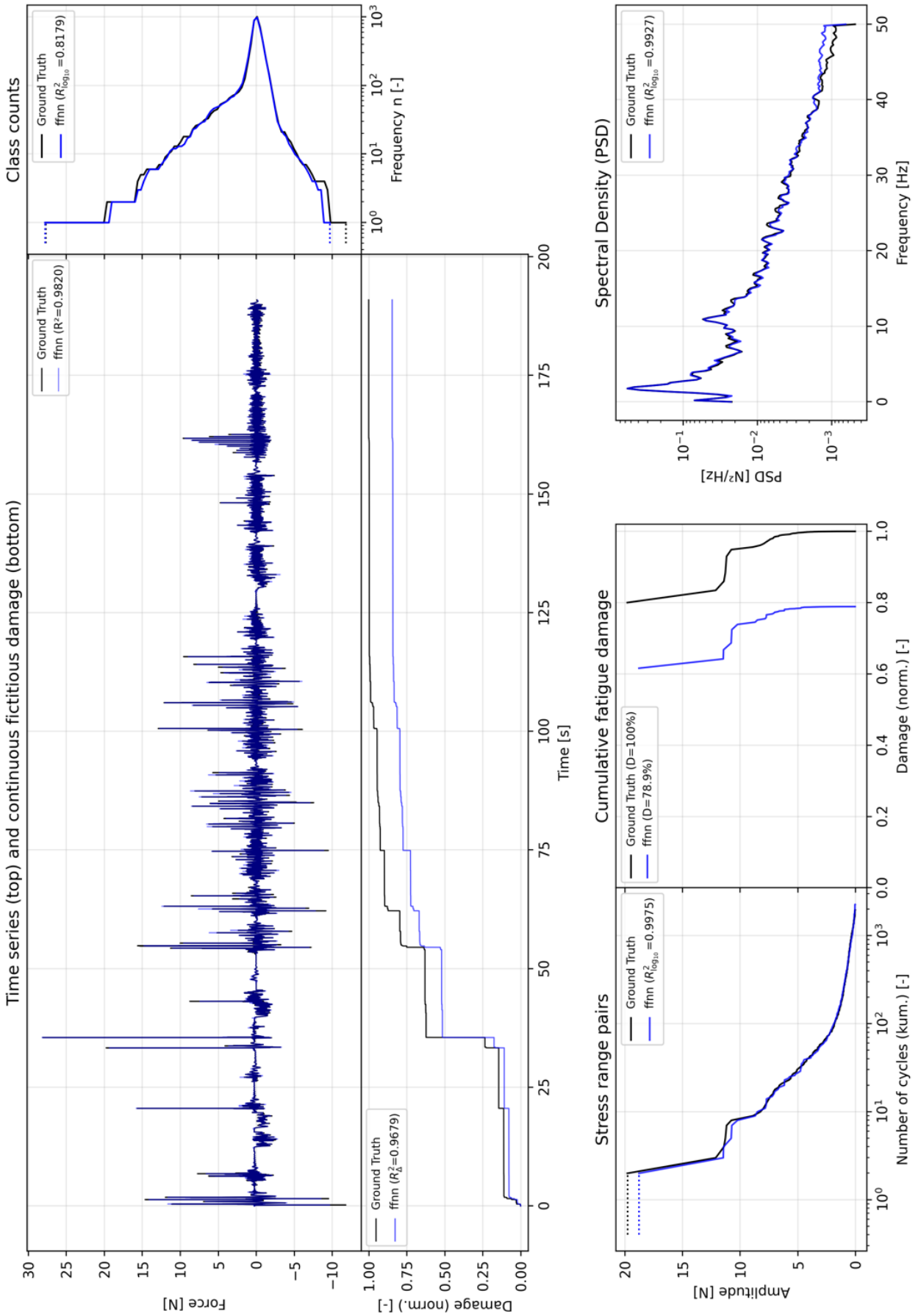


Figure 6.5.: Results without damage-based loss using Welch (FFNN)

It should be noted that due to the stochastic nature of training, these differences cannot be attributed solely to the theoretical properties of the spectral estimators and may partly reflect variability in the optimization process.

A key observation underlying these results is that, although the predicted load histories achieve small errors in conventional metrics, models trained without damage-aware loss tend to over- and underestimate peaks and valleys. Since fatigue damage is highly sensitive to extrema (Section 2.2), these deviations lead to significant discrepancies in damage estimation. Incorporating damage information directly into the training objective mitigates this effect and improves alignment with the ground truth.

However, fatigue damage is represented by a single scalar value typically in the range of $[0, 1]$. Consequently, different load histories may yield similar damage values over a given time span. This non-uniqueness contributes to variability in training outcomes and suggests that further investigation of loss design and parameter selection is required to achieve consistently optimal results.

Overall, the results demonstrate that incorporating a physics-informed damage loss can improve fatigue damage prediction, while also highlighting the sensitivity of the approach to model and training design choices. These findings motivate a more systematic discussion of the implications, which is provided in the following chapter.

7. Conclusion and Outlook

7.1. Conclusion

The main objective of this work was to establish a differentiable surrogate for fatigue damage that enables its integration into the training of virtual sensors. Such sensors aim to estimate forces and resulting structural damage in vehicles, where direct physical measurements are impractical due to cost and hardware constraints. Consequently, lightweight deep learning models are required, which often lack sufficient physical guidance during training.

To address this limitation, the spectral fatigue damage model proposed by Dirlik was introduced as a differentiable surrogate for Rainflow Counting. By enabling end-to-end differentiation through the spectral estimation step, fatigue damage can be incorporated directly into the loss function and thus influence the optimization process during training. The analysis of the underlying spectral estimators, including the periodogram and the Welch method, together with the derived gradient expressions, provides the foundation for integrating fatigue damage into gradient-based learning frameworks.

A central finding of this work is that the Dirlik-based loss introduces extremely small gradients, leading to vanishing-gradient-like behavior during training. This challenge was addressed through an appropriate choice of the loss formulation, in particular the MSLE, which effectively rescales the gradients and enables stable optimization. The experimental results demonstrate that incorporating the damage-based loss improves the prediction of cumulative fatigue damage. At the same time, the observed variability across training runs highlights that the effectiveness of the approach depends on hyperparameter choices as well as on the inherent non-uniqueness of fatigue damage representations.

7.2. Outlook

While the proposed approach demonstrates clear potential, several challenges remain for future work. A fundamental limitation arises from the fact that fatigue damage is represented as a scalar quantity typically in the range $[0, 1]$. As a result, different load histories may correspond to similar damage values, introducing ambiguity into the learning objective and contributing to variability in training outcomes. Addressing this issue requires further investigation of alternative loss formulations and data representations that better capture the underlying load characteristics.

As the combined loss considered here, as defined in Section 4.1, can be interpreted as a multitask loss function in the sense of Chen et al. [Che+18], their gradient normalization algorithm (GradNorm) could provide more consistent results when applied in this context. GradNorm proposes an adaptive approach in which the weighting parameter,

here denoted by α, is adjusted dynamically at each training step. This allows the contributions of the individual tasks to be balanced throughout training, thereby supporting an optimized training [Che+18].

Since real load histories in vehicle applications are highly nonstationary and vary significantly across different scenarios, such an adaptive approach could further improve the multitask formulation used here for predicting both the signal and the corresponding damage. In addition, the algorithm regulates extreme gradients by penalizing both excessively large and excessively small backpropagated gradients [Che+18]. Consequently, the bottleneck effect discussed in Section 5.3, where extremely small Dirlik gradients introduce vanishing-gradient-like behavior, could be mitigated within the optimization process itself. This would enable the consideration of alternative loss functions beyond the mean squared logarithmic error used in this work, for example probabilistic formulations that more explicitly account for extreme events or outliers.

From a computational perspective, the efficiency of the gradient computation, particularly for the Welch-based method, can be further improved. As discussed in Section 5.2, exploiting sparse tensor structures in the implementation of the windowed gradient could significantly reduce computational cost.

Overall, the integration of physics-informed damage models into deep learning has been shown to be both feasible and beneficial. Future developments in loss design, optimization strategies, and efficient implementations are expected to further enhance the performance and robustness of virtual sensors for fatigue analysis.

A. Appendix

A.1. Power Spectral Density Implementations

A.1.1. Periodogram Method

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.1.2. Welch Method

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.1.3. Apodization Method

The next code snippet shows the implementation of the in Table 3.2 derived data and frequency depentent temporal windowed periodograms.

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.1.4. Plots of Welch and Apodization Comparison for Different Windows

These are the plots corresponding to the comparison of Welch and apodization approach in Section 3.2 for the two more windows of *Hamming* and *Rectangular*.

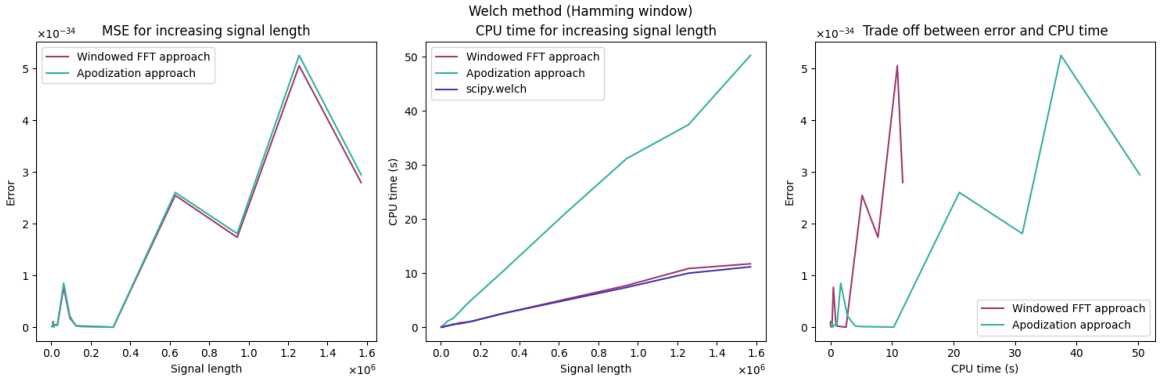


Figure A.1.: Plots of the Welch method with Hamming window, error for increasing signal length (left), CPU time for increasing signal length (centre), error and corresponding CPU time (right)

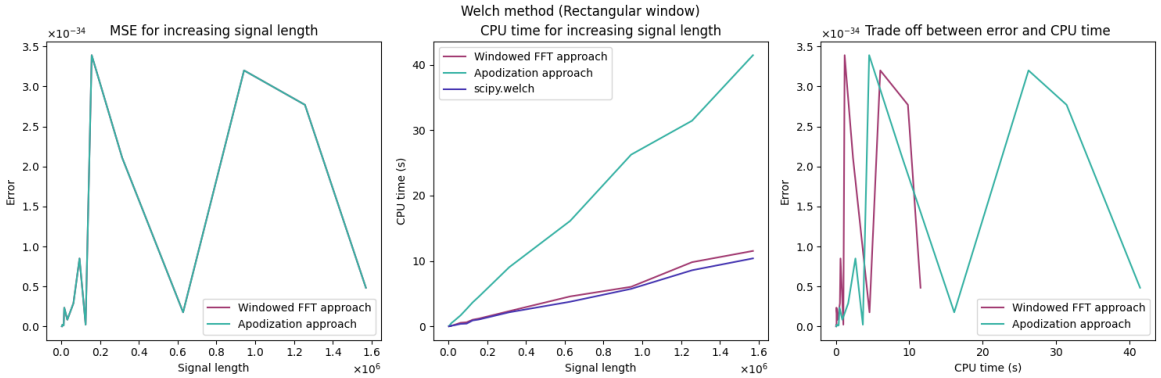
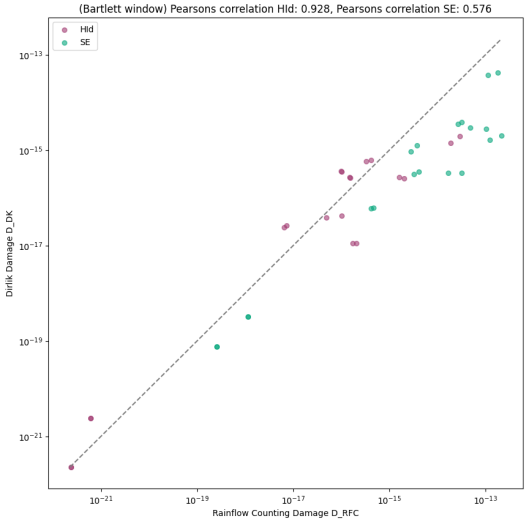


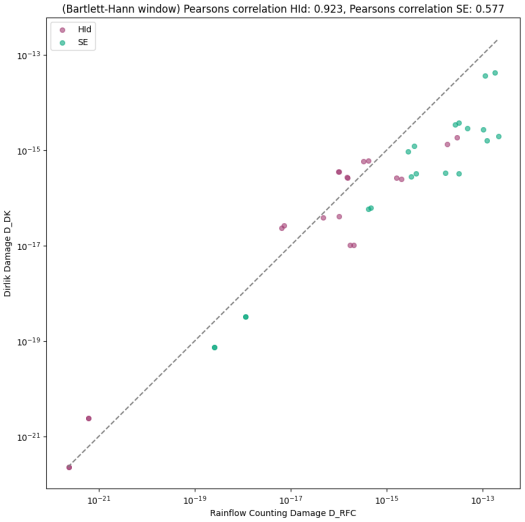
Figure A.2.: Plots of the Welch method with Rectangular window, error for increasing signal length (left), CPU time for increasing signal length (centre), error and corresponding CPU time (right)

A.1.5. Plots of Dirlik Damage under the Choice of Different Window Types

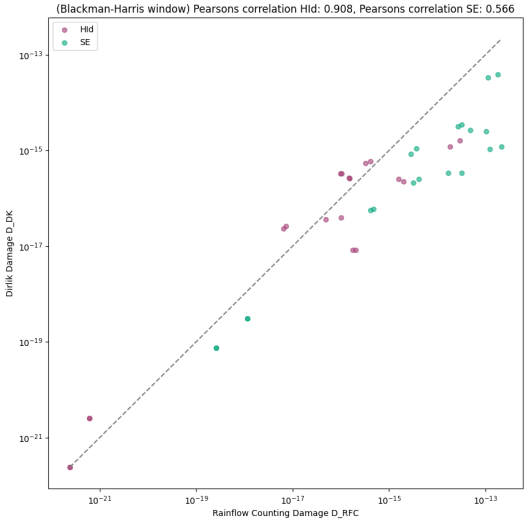
Dirlik damage in relation to Rainflow damage for different windows.



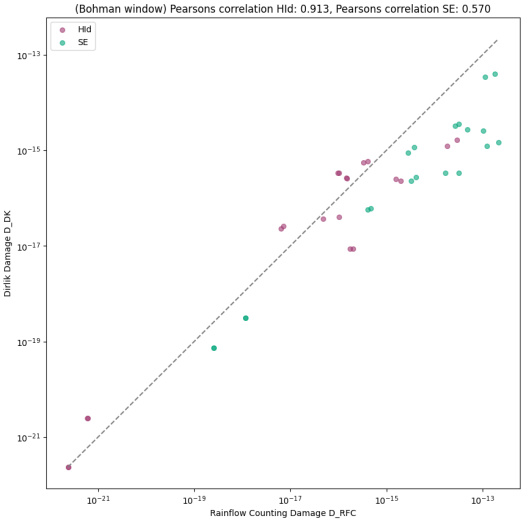
(a) Bartlett



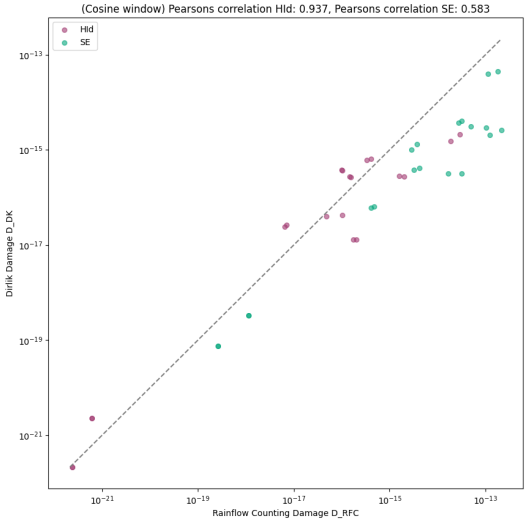
(b) Bartlett-Hanning



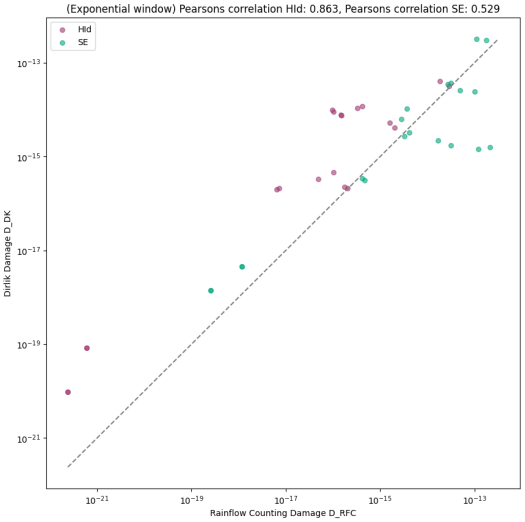
(c) Blackman-Harris



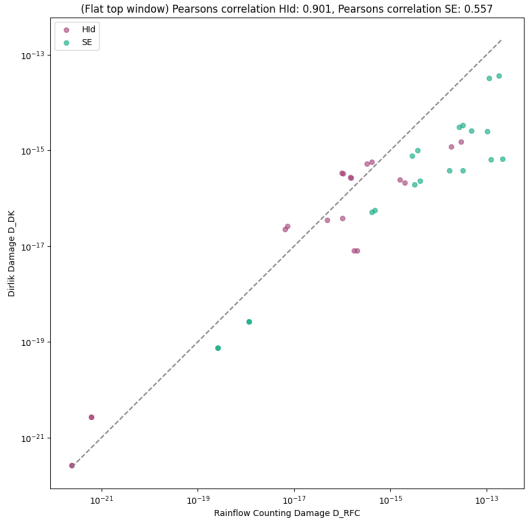
(d) Bohman



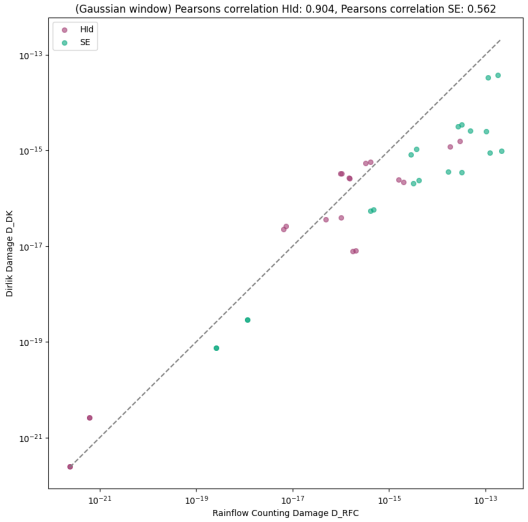
(e) Cosine



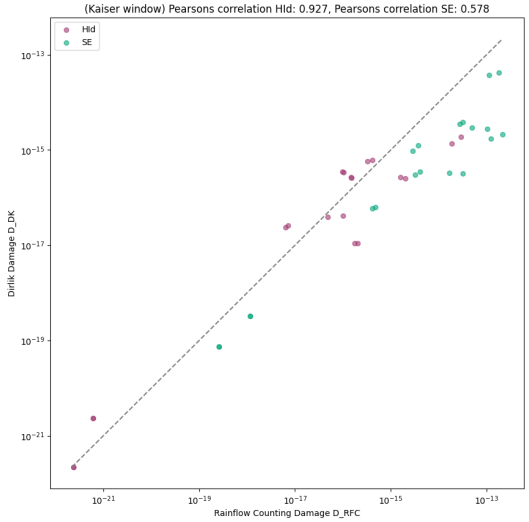
(f) Exponential



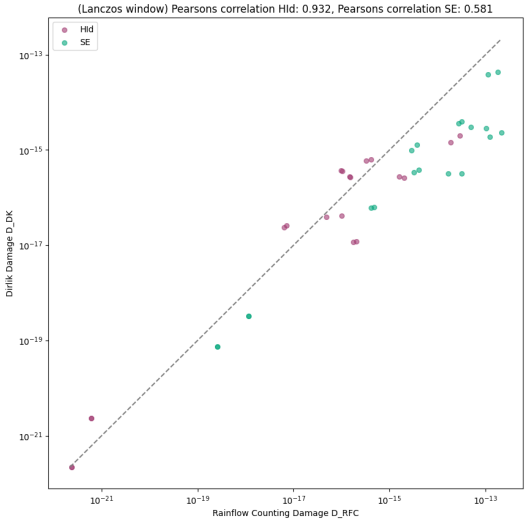
(a) Flat top



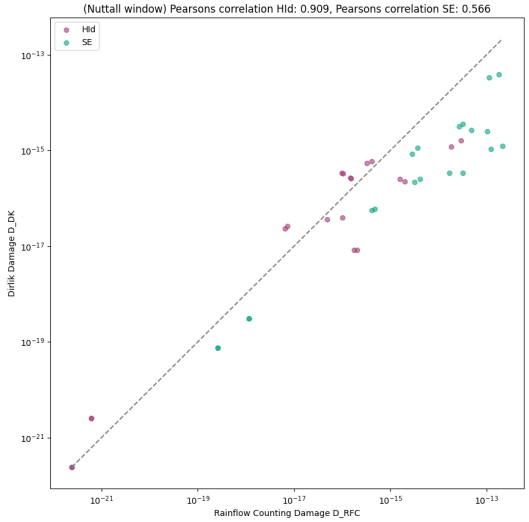
(b) Gaussian



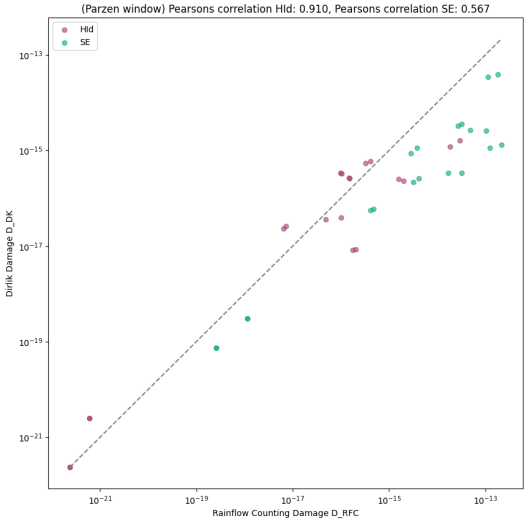
(c) Kaiser



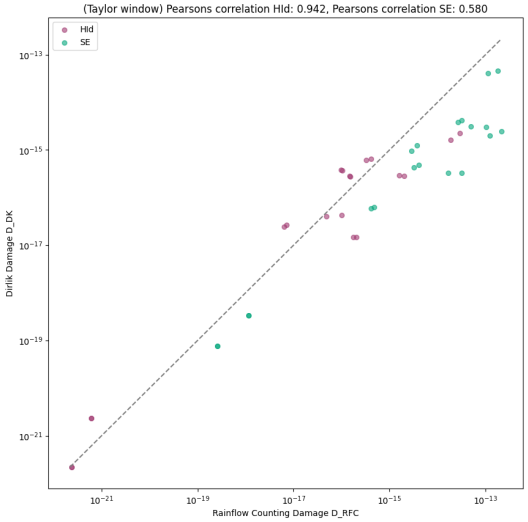
(d) Lanczos



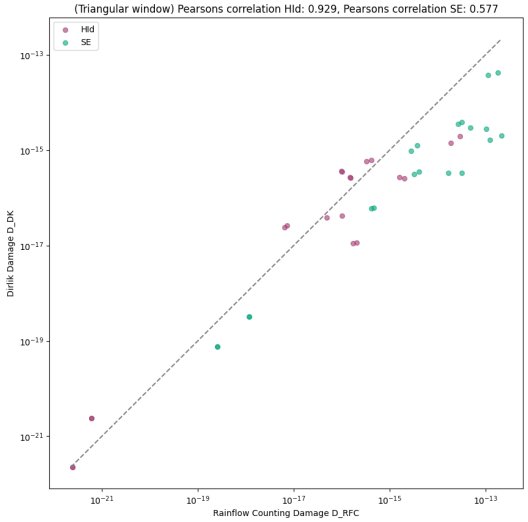
(e) Nuttall



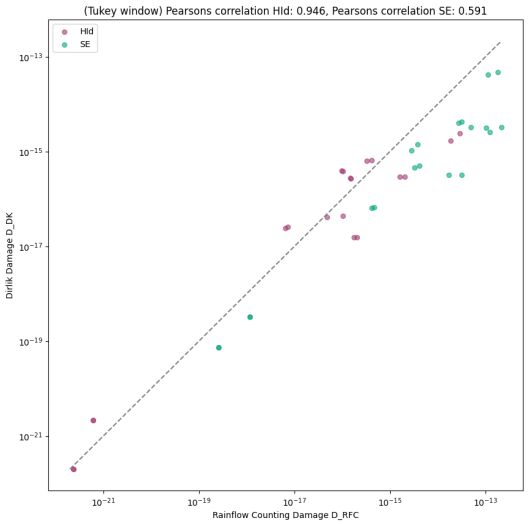
(f) Parzen



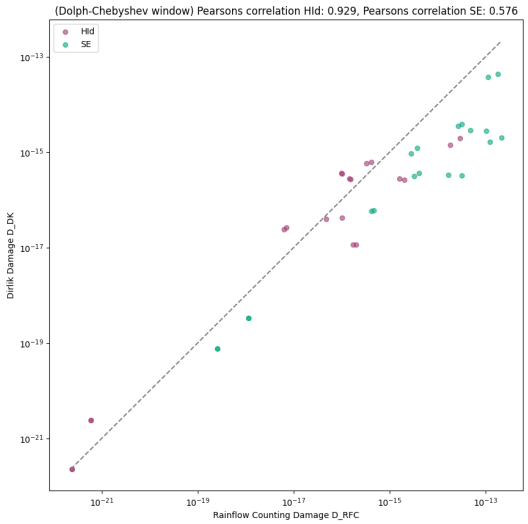
(a) Taylor



(b) Triangular



(a) Tukey



(b) Dolph-Chebyshev

A.2. Graph Safe Implementation of the Spectral Estimates and Spectral Method by Dirlik

In the following section, the methods are all implemented in TensorFlow. Their implementation is ensured to be graph safe for seemles application in training deep learning frameworks in TensorFlow and keras.

A.2.1. Periodogram Method (TensorFlow)

In the case of the periodogram spectral estimate to derive the PSD, this approach was only implementend in TensorFlow, so the corresponding code is the one in Subsection A.1.1.

A.2.2. Welch Method (TensorFlow)

Welch function defined in TensorFlow with TensorFlow Tensors and corresponding operations for integration in neural network training.

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.2.3. Implementation of the Dirlik Method (Welch)

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.2.4. Implementation of the Dirlik Method (Periodogram)

Since the only thing changing in the Dirlik method for PSD through the periodogram spectral estimate, the definintion of the function is identical to the one in Subsection A.2.3 except for the followning lines of code, that need to be replaced.

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.3. Derivation of the Gradients and their Implementation

A.3.1. Gradient of the Periodogram Method

The gradient of the periodogram estimate

$$\hat{\Phi}_p(\omega) = \frac{1}{N}\left|\sum_{t=1}^{N} y(t)\,\mathrm{e}^{-\mathrm{i}\omega t}\right|^2 .$$

is derived by making use of the Euler identity $e^{-i\omega t} = \cos(\omega t) - i\sin(\omega t)$. Substituting this identity into the DFT definition yields

$$z := \sum_{t=1}^{N} y(t)e^{-i\omega t} = \sum_{t=1}^{N} y(t)(\cos(\omega t) - i\sin(\omega t)) = \underbrace{(\sum_{t=1}^{N} y(t)\cos(\omega t))}_{\Re(z)} \underbrace{-i\,(\sum_{t=1}^{N} y(t)\sin(\omega t))}_{\Im(z)} \in \mathbb{C},$$

where $\Re(z)$ denotes the real part and $\Im(z)$ the imaginary part of z.

Using $|z|^2 = z\bar{z}$, the periodogram can be written as

$$\hat{\Phi}_p(\omega) = \frac{1}{N}\,|z|^2 = \frac{1}{N}(\sqrt{z\bar{z}})^2 = \frac{1}{N}(\Re(z)^2 + \Im(z)^2) \in \mathbb{R}.$$

By application of the product rule, the gradient of this equivalent real-valued representation is obtained as

$$\frac{\partial \hat{\Phi}_p(\omega)}{\partial y(t)} = \frac{2}{N}((\sum_{t=1}^{N} \tilde{y}(t)\cos(\omega t))(\sum_{t=1}^{N} \frac{\partial \tilde{y}(t)}{\partial y(t)}\cos(\omega t))+(\sum_{t=1}^{N} \tilde{y}(t)\sin(\omega t))(\sum_{t=1}^{N} \frac{\partial \tilde{y}(t)}{\partial y(t)}\sin(\omega t))).$$

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.3.2. Gradient of the Welch Method

To implement the gradient of the Welch method, one can consider each of the components in their matrix form intead of defining them as functions and thus computing the sum over t.
Thus, Equation 5.2 can be considered as: Set

$$\partial \tilde{Y}_j = \frac{\partial \tilde{y}_j(t)}{\partial y(\hat{t})} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ A \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{N\times M} \; \forall \hat{t} \in \{1,\dots,N\}, \; \forall t \in \{1,\dots,M\},$$

$$\text{with} \begin{pmatrix} 1-\frac{1}{M} & -\frac{1}{M} & \cdots & -\frac{1}{M} \\ -\frac{1}{M} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -\frac{1}{M} \\ -\frac{1}{M} & \cdots & -\frac{1}{N} & 1-\frac{1}{M} \end{pmatrix} \in \mathbb{R}^{M\times M},$$

Set

$$\begin{aligned} V = v(t) &\in \mathbb{R}^M, \forall t \in \{1,\dots,M\} \\ \tilde{Y}_j = \tilde{y}_j(t) &\in \mathbb{R}^M, \forall t \in \{1,\dots,M\} \\ Cos = \cos(\omega t) &\in \mathbb{R}^{N\times U}, \forall t \in \{1,\dots,M\} \\ Sin = \sin(\omega t) &\in \mathbb{R}^{N\times U}, |\omega| = U, \forall t \in \{1,\dots,M\}. \end{aligned}$$

Thus,

$$\sum_{t=1}^{M} v(t)\tilde{y}_j(t)\cos(\omega t) = \underbrace{(V \circ \tilde{Y}_j)}_{\in\mathbb{R}^M}\ \underbrace{Cos}_{\in\mathbb{R}^{M\times U}} \in \mathbb{R}^U,$$

$$\sum_{t=1}^{M} v(t)\frac{\partial\tilde{y}_j(t)}{\partial y(\hat{t})}\cos(\omega t) = \underbrace{(V \circ \partial\tilde{Y}_j)}_{\in\mathbb{R}^{N\times M}}\ \underbrace{Cos}_{\in\mathbb{R}^{M\times U}} \in \mathbb{R}^{N\times U}$$

($\circ$ stands for elementwise-multiplication)

$$\underbrace{(\sum_{t=1}^{M} v(t)\tilde{y}_j(t)\cos(\omega t))}_{\in\mathbb{R}^{N\times U}} \circ \underbrace{(\sum_{t=1}^{M} v(t)\frac{\partial\tilde{y}_j(t)}{\partial y(\hat{t})}\cos(\omega t))}_{\in\mathbb{R}^{N\times U}} \in \mathbb{R}^{N\times U}$$

equivalently for $/sin$ terms.

$$\Rightarrow \frac{\partial\hat{\Phi}_j(\omega)}{\partial y(\hat{t})} = \underbrace{\frac{2}{MP}}_{\in\mathbb{R}} \underbrace{(((V \circ \tilde{Y}_j)Cos)((V \circ \partial\tilde{Y}_j)Cos) + ((V \circ \tilde{Y}_j)Sin)((V \circ \partial\tilde{Y}_j)Sin))}_{\in\mathbb{R}^{N\times U}} \in \mathbb{R}^{S\times N\times U}$$

$$\Rightarrow \frac{\partial\hat{\Phi}_W(\omega)}{\partial y(\hat{t})} \in \mathbb{R}^{N\times U}.$$

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.3.3. Gradient of the Dirlik method

The gradient of the factor $h(y)$ of the product chain to derive the gradient of the Dirlik spectral method is given by

$$\begin{aligned}\frac{\partial h(y)}{\partial y} = {} & \frac{\partial C_1}{\partial y}Q^m\Gamma(1+m) + C_1 m Q^{m-1}\frac{\partial Q}{\partial y}\Gamma(1+m) \\ & + \sqrt{2}^m\Gamma(1+\frac{m}{2})(\frac{\partial C_2}{\partial y}|R|^m + C_2 m|R|^{m-1}\frac{R}{|R|}\frac{\partial R}{\partial y} + \frac{\partial C_3}{\partial y})\end{aligned}$$

$$\frac{\partial \nu_p}{\partial y} = \frac{1}{2\sqrt{\frac{\lambda_4}{\lambda_2}}} \frac{\frac{\partial \lambda_4}{\partial y}\lambda_2 - \lambda_4 \frac{\partial \lambda_2}{\partial y}}{\lambda_2^2}$$

$$\frac{\partial x_m}{\partial y} = \frac{\partial \frac{\lambda_1}{\lambda_0}}{\partial y}\sqrt{\frac{\lambda_2}{\lambda_4}} + \frac{\lambda_1}{\lambda_0}\frac{1}{2\sqrt{\frac{\lambda_2}{\lambda_4}}}\frac{\partial \frac{\lambda_2}{\lambda_4}}{\partial y}$$

$$\frac{\partial \frac{\lambda_1}{\lambda_0}}{\partial y} = \frac{\frac{\partial \lambda_1}{\partial y}\lambda_0 - \lambda_1 \frac{\partial \lambda_0}{\partial y}}{\lambda_0^2}$$

$$\frac{\partial \frac{\lambda_2}{\lambda_4}}{\partial y} = \frac{\frac{\partial \lambda_2}{\partial y}\lambda_4 - \lambda_2 \frac{\partial \lambda_4}{\partial y}}{\lambda_4^2}$$

$$\frac{\partial C_1}{\partial y} = \frac{2(\frac{\partial x_m}{\partial y} - 2\alpha_2 \frac{\partial \alpha_2}{\partial y})(1+\alpha_2^2) - 4(x_m - \alpha_2^2)\alpha_2 \frac{\partial \alpha_2}{\partial y}}{(1+\alpha_2^2)^2}$$

$$\frac{\partial C_2}{\partial y} = \frac{-(\frac{\partial \alpha_2}{\partial y} + (1-2C_1)\frac{\partial C_1}{\partial y})(1-R) - (1-\alpha_2 - C_1 + C_1^2)(-\frac{\partial R}{\partial y})}{(1-R)^2}$$

$$\frac{\partial C_3}{\partial y} = -(\frac{\partial C_1}{\partial y} + \frac{\partial C_2}{\partial y})$$

$$\frac{\partial Q}{\partial y} = \frac{(1.25(\frac{\partial \alpha_2}{\partial y} - \frac{\partial C_3}{\partial y} - (\frac{\partial C_2}{\partial y}R + C_2\frac{\partial R}{\partial y})))C_1 - (1.25(\alpha_2 - C_3 - (C_2 R))\frac{\partial C_1}{\partial y})}{C_1^2}$$

$$\frac{\partial R}{\partial y} = \frac{(\frac{\partial \alpha_2}{\partial y} - \frac{\partial x_m}{\partial y} - 2C_1\frac{\partial C_1}{\partial y})(1-\alpha_2 - C_1 + C_1^2) - (\alpha_2 - x_m - C_1^2)(-(\frac{\partial \alpha_2}{\partial y} + (1-2C_1)\frac{\partial C_1}{\partial y}))}{(1-\alpha_2 - C_1 + C_1^2)^2}$$

$$\frac{\partial \alpha_n}{\partial y} = \frac{\frac{\partial \lambda_n}{\partial y}\sqrt{\lambda_0 \lambda_{2n}} - \lambda_n \frac{1}{2\sqrt{\lambda_0 \lambda_{2n}}}\left(\frac{\partial \lambda_0}{\partial y}\lambda_{2n} + \lambda_0 \frac{\partial \lambda_{2n}}{\partial y}\right)}{\lambda_0 \lambda_{2n}}$$

$$\frac{\partial \lambda_n}{\partial y} = \frac{\partial}{\partial y}\int_0^\infty \omega^n W_X(\omega)\, d\omega \overset{Leibniz\ integral\ rule}{=} \int_0^\infty \omega^n \frac{\partial W_X(\omega)}{\partial y}\, d\omega$$

A.3.4. Implementation of the Gradient of the Dirlik Method (Welch)

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

A.3.5. Implementation of the Gradient of the Dirlik Method (Periodogram)

In this case, as the spectral estimate is the only thing changing in the implementation of the Dirlik gradient, the code is identical to the one in Subsection A.3.4 except that the following lines have to be changed accordingly.

Source code omitted for proprietary compliance with Mercedes-Benz AG; the underlying mathematical logic and implementation details remain fully described throughout the entire thesis.

Bibliography

[ABC+16] Martín Abadi, Paul Barham, Zhifeng Chen, et al. "TensorFlow: A System for Large-Scale Machine Learning". In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 2016, pp. 265–283.

[Agg23] Charu C. Aggarwal. *Neural Networks and Deep Learning: A Textbook*. Second. Cham: Springer, 2023. ISBN: 978-3-031-29641-3. DOI: 10.1007/978-3-031-29642-0. URL: https://link.springer.com/book/10.1007/978-3-031-29642-0.

[BT05] D. Benasciutti and R. Tovo. "Comparison of spectral methods for fatigue analysis of broad-band Gaussian random processes". In: *Department of Engineering, University of Ferrara* (2005).

[Che+18] Zhao Chen et al. "GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks". In: *Proceedings of the 35th International Conference on Machine Learning (ICML)* 80 (2018). arXiv preprint arXiv:1711.02257v4. URL: https://arxiv.org/abs/1711.02257.

[DB21] Turan Dirlik and Denis Benasciutti. "Dirlik and Tovo–Benasciutti Spectral Methods in Vibration Fatigue: A Review with a Historical Perspective". In: *Metals* 11.9 (2021), p. 1333. DOI: 10.3390/met11091333. URL: https://www.mdpi.com/1996-1944/11/9/1333.

[Dir85] Turan Dirlik. "Application of Computers in Fatigue Analysis". Ph.D. Thesis. PhD thesis. Coventry, England: Department of Engineering, University of Warwick, 1985.

[GB10] Xavier Glorot and Yoshua Bengio. "Understanding the Difficulty of Training Deep Feedforward Neural Networks". In: *arXiv preprint arXiv:1001.4842* (2010). arXiv:1001.4842 [cs.LG]. URL: https://arxiv.org/abs/1001.4842.

[GB20] Aboubaker Gherbi and Mourad Belgasmia. "Stochastic Analysis Basics and Application of Statistical Linearization Technique on a Controlled System with Nonlinear Viscous Dampers". In: *Handbook of Probabilistic Models* (2020). Chapter 21, pp. 505–525. DOI: 10.1016/B978-0-12-816514-0.00021-7. URL: https://doi.org/10.1016/B978-0-12-816514-0.00021-7.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[Hai06] Erwin Haibach. *Betriebsfestigkeit: Verfahren und Daten zur Bauteilberechnung*. 3rd ed. Berlin, Heidelberg, New York: Springer-Verlag, 2006. ISBN: 978-3-540-29363-7.

[Hal99] Andrew Halfpenny. "A Frequency Domain Approach for Fatigue Life Estimation from Finite Element Analysis". In: *Proceedings of the International Conference on Damage Assessment of Structures (DAMAS 99)* (1999). URL: http://www.vibrationdata.com/tutorials2/Whitepaper_nCode_frequency_domain_fatigue-Halfpenny.pdf.

[HJ13] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. English. 2. ed. Cambridge ; New York: Cambridge University Press, 2013.

[JS20] Ricardo Jauregui and Ferran Silva. "Numerical Validation Methods". In: *Universitat Politècnica de Catalunya (UPC), Barcelona, Spain* (2020). URL: https://upc.edu/.

[Köh+12] Michael Köhler et al. *Zählverfahren und Lastannahme in der Betriebsfestigkeit*. Heidelberg, Dordrecht, London, New York: Springer-Verlag, 2012. ISBN: 978-3-642-13163-9. DOI: 10.1007/978-3-642-13164-6.

[KP85] W. Krüger and J. Petersen. *Simulation und Extrapolation von Rainflow-Matrizen*. Tech. rep. Bericht Nr. 8. Kaiserslautern, Germany: Universität Kaiserslautern, Fachbereich Mathematik, May 1985.

[KS24] Grigory Khromov and Sidak Pal Singh. "Some Fundamental Aspects About Lipschitz Continuity of Neural Networks". In: *Proceedings of the 12th International Conference on Learning Representations (ICLR)*. Vienna, Austria, 2024.

[PMB13] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. "On the Difficulty of Training Recurrent Neural Networks". In: *arXiv preprint arXiv:1211.5063* (2013). arXiv:1211.5063 [cs.LG]. URL: https://arxiv.org/abs/1211.5063.

[Pro05] D. Prokhorov. "Virtual Sensors and Their Automotive Applications". In: *2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*. 2005, pp. 411–416. DOI: 10.1109/ISSNIP.2005.1595614.

[PSG17] Jeffrey Pennington, Samuel S. Schoenholz, and Surya Ganguli. "Resurrecting the Sigmoid in Deep Learning through Dynamical Isometry: Theory and Practice". In: *arXiv preprint arXiv:1711.04735* (2017). arXiv:1711.04735v1 [cs.LG], submitted 13 November 2017. URL: https://arxiv.org/abs/1711.04735.

[QWZ23] Xianbiao Qi, Jianan Wang, and Lei Zhang. "Understanding Optimization of Deep Learning via Jacobian Matrix and Lipschitz Constant". In: *arXiv preprint arXiv:2306.09338* (2023). URL: https://arxiv.org/abs/2306.09338.

[Rat+24] Pratik Rathore et al. "Challenges in Training PINNs: A Loss Landscape Perspective". In: *arXiv preprint arXiv:2402.01868* (2024). DOI: 10.48550/arXiv.2402.01868. URL: https://arxiv.org/abs/2402.01868.

[SM05] Petre Stoica and Randolph Moses. *Spectral Analysis of Signals*. Upper Saddle River, New Jersey: Prentice Hall, 2005. ISBN: 0-13-113956-8.

[Vir+20] Pauli Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[Wel67] P. Welch. “The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms”. In: *IEEE Transactions on Audio and Electroacoustics* 15.2 (1967), pp. 70–73. DOI: 10.1109/TAU.1967.1161901.

Master's Theses in Mathematical Sciences 2026:E79
ISSN 1404-6342

LUNFNA-3049-2026

Numerical Analysis
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lu.se/