TISMIR

A Basic Tutorial on Novelty and Activation Functions for Music Signal Processing

EDUCATIONAL ARTICLE

MEINARD MÜLLER
CHING-YU CHIU

*Author affiliations can be found in the back matter of this article

]u[ubiquity press

ABSTRACT

In Music Information Retrieval (MIR), a general goal is to recognize times of novelty within music recordings. This includes estimating structural boundaries through the detection of changes in harmony, tempo, or instrumentation and identifying onsets of note and sound events by capturing changes in the music signal's energy or spectral content. These tasks leverage novelty functions, which are one-dimensional, time-dependent functions characterized by sharp local maxima that indicate significant musical and acoustical changes. From a given music recording, novelty functions can be derived using a variety of methods, ranging from traditional signal-processing techniques to modern data-driven approaches, where they are often termed "activation functions." In this tutorial, we explore the concept of novelty functions and some of their essential properties. We discuss methods to enhance these functions and improve their distinctive peak-like structures. These improvements are crucial for simplifying the identification of specific musical events using post-processing methods, from basic peak picking to more sophisticated approaches like periodicity analysis. We also assess novelty functions through commonly used metrics such as precision, recall, and F-measure but with an emphasis on error tolerance. Aimed at Bachelor's degree and beginning Master's degree students with basic knowledge of signal processing and mathematics, this tutorial uses illustrative figures to clarify key concepts, thereby broadening its accessibility to a wider MIR audience and enriching their comprehension of this significant subject. Furthermore, Jupyter notebooks, including Python source code for the core algorithms and audio examples that allow for reproducing the tutorial's figures, are provided at https://github.com/groupmm/edu_novfct.

CORRESPONDING AUTHOR:

Meinard Müller

International Audio Laboratories Erlangen, Germany

meinard.mueller@audiolabs-erlangen.de

KEYWORDS:

Novelty function, activation function, onset, beat, downbeat, structure, music, audio

TO CITE THIS ARTICLE:

Müller, M., & Chiu, C.-Y. (2024). A Basic Tutorial on Novelty and Activation Functions for Music Signal Processing. *Transactions of the International Society for Music Information Retrieval,* 7(1), 179–194. DOI: https://doi.org/10.5334/tismir.202

1 INTRODUCTION

In the field of Music Information Retrieval (MIR), segmenting music into meaningful sections is a fundamental task. Generally, in multimedia processing, segmentation involves dividing a digital document into parts to simplify analysis and facilitate understanding. For example, in image processing, segmentation is the process of dividing an image into regions based on characteristics such as color, intensity, or texture, with boundaries typically marked by significant changes in these properties. Similarly, music segmentation divides a music recording into musically and acoustically meaningful sections, each defined by its start and end points. This can include tasks from a detailed level, such as identifying the start positions of individual notes or beats, to a broader level, such as detecting changes in instrumentation or harmony. In the latter scenario, these points of change or novelty often indicate structural boundaries within the music, such as the transition from a verse to a chorus or from one musical theme to another.

A fundamental concept for detecting changes in music recordings is the *novelty function*. Originally applied in the context of music structure analysis by Foote (2000), novelty functions have since been extensively used for various MIR tasks, including onset detection (Bello et al., 2005), beat and pulse tracking (Grosche and Müller, 2011), chord recognition (Pauwels et al., 2019), and music transcription (Benetos et al., 2019). Mathematically, a novelty function is a time-dependent function $\Delta : \mathbb{R} \to \mathbb{R}$ that yields a real value $\Delta(t) \in \mathbb{R}$ for each time point $t \in \mathbb{R}$. The value $\Delta(t)$ represents the degree of novelty (or a change) at each point in time. Intuitively, the local maxima or peaks of a novelty function correspond to changes in specific properties of the underlying music recording. Illustrative examples are shown in Figure 1 for various MIR tasks.

In practice, the methods for computing novelty functions and identifying relevant peaks depend on the particular MIR task being addressed and the initial music input representation. Starting with a music recording, the first step typically involves transforming the audio signal into a feature representation, such as a chromagram or spectrogram (Müller, 2021), to capture important musical and acoustical properties, such as those related to harmony or frequency. Subsequently, this representation undergoes conversion into an appropriate novelty function using filtering and differentiation techniques. To identify significant peaks, the final step involves post-processing, which can range from simple peak picking to more complex methods like periodicity analysis and dynamic programming. The positions of these peaks act as potential markers for estimating the timing of musical events under analysis.

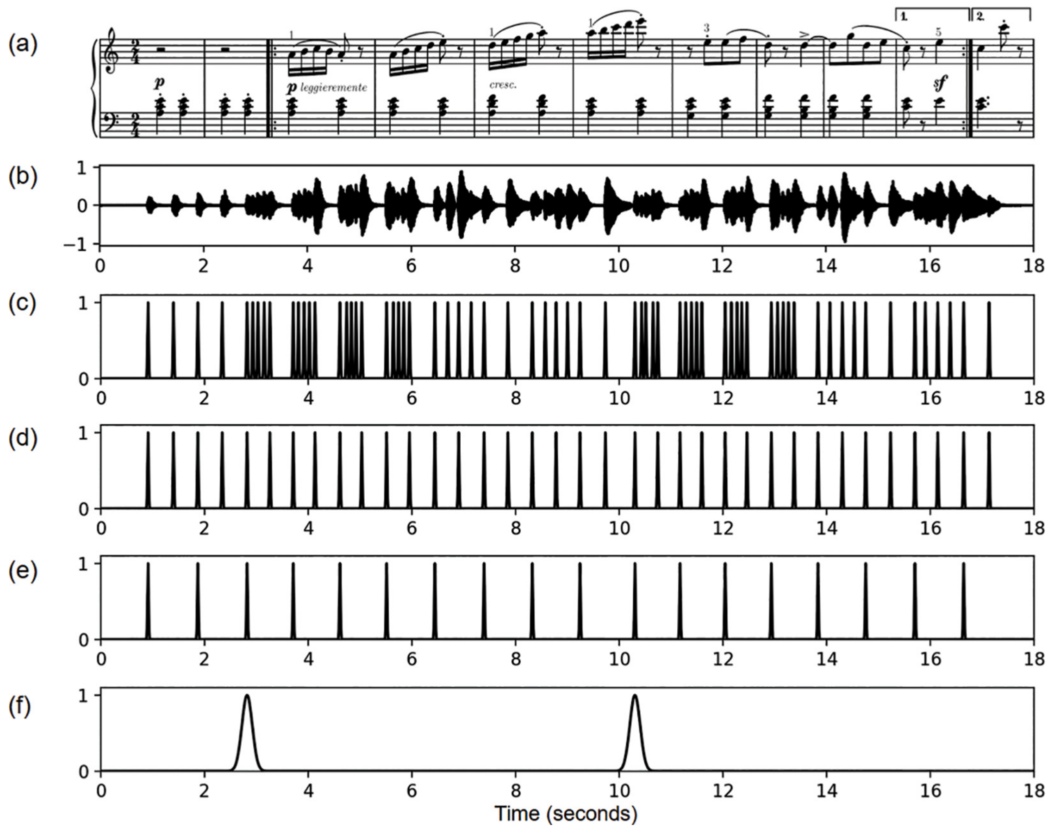

Figure 1 Idealized novelty functions for various MIR tasks. **(a)** Musical score of the beginning of the second piano etude from Op. 100 by Burgmüller. **(b)** Audio recording of the corresponding excerpt. **(c)** Onset detection. **(d)** Beat tracking. **(e)** Downbeat (measure) tracking. **(f)** Boundary estimation for music structure analysis.

In recent years, data-driven machine learning methods have been applied to compute functions similar to novelty functions, commonly referred to as *activation functions* in this context. These approaches necessitate a training dataset comprising music recordings and target annotations that identify the time points when relevant musical events occur. Despite activation functions being used for a broader range of tasks, including the detection of properties over longer time periods (e.g., singing voice detection), the terms "activation function" and "novelty function" are frequently used interchangeably within the context of beat tracking, musical boundary detection, and related MIR tasks. In this article, the term "novelty function" is used to highlight our focus on functions characterized by their peak structures.

Clearly, the success of the final segmentation heavily relies on the characteristics of the novelty function. The post-processing step is crucial, requiring the novelty function to have distinct peaks at targeted locations (to capture true positives) while suppressing local maxima at irrelevant positions (to avoid false positives). Enhancing the properties of novelty functions often includes methods like smoothing and normalization. Specifically, when incorporating novelty functions into probabilistic frameworks, it is beneficial to constrain novelty values to between 0 and 1, allowing them to be interpreted and treated as pseudoprobabilities. In summary, ideally, novelty functions are sensitive to relevant musical changes and are robust against irrelevant aspects and noise. They should remain stable amidst signal variations, offer precise timing, be computationally efficient, and be adaptable across different music genres.

After this general introduction, the remainder of this tutorial is organized as follows: In Section 2, we provide basic mathematical definitions and, to make them more concrete, consider two specific MIR applications: onset detection and structure analysis. We demonstrate the derivation of novelty functions for both tasks using traditional signal processing techniques. Additionally, we describe general principles of data-driven methods that produce activation functions to achieve similar goals. Then, in Section 3, we examine the fundamental properties of novelty functions and explore strategies to improve them. In Section 4, we cover post-processing strategies, with a focus on simple peak picking, and discuss their evaluation based on precision, recall, and F-measure, taking into account error tolerance. Finally, in Section 5, we summarize the tutorial and provide guidance for educational integration, recommended software, and relevant datasets for hands-on experimentation.

We conclude this introduction by stating that this tutorial is primarily aimed at Bachelor's degree and beginning Master's degree students with basic knowledge of signal processing and mathematics. However, through the use of illustrative figures to explain key concepts, we strive to make this tutorial accessible to a broader, non-technical MIR audience. Furthermore, along with this article, we provide Jupyter notebooks[1] that include Python source code for the core algorithms. Additionally, by providing music and audio examples, the notebooks allow for a step-by-step reproduction of all results and illustrations shown in the article's figures.

2 BASIC DEFINITIONS AND NOVELTY COMPUTATION

In this section, we discuss various ways to compute novelty functions. To make things concrete, we consider two representative MIR tasks known as onset detection and structure analysis. Before doing so, we establish some basic mathematical definitions that will be used in the remainder of the tutorial.

2.1 BASIC MATHEMATICAL DEFINITIONS

As mentioned in the introduction, a novelty function can be modeled as a time-dependent, real-valued function. While considering continuous-time functions with time points $t \in \mathbb{R}$ being intuitive, the use of computers necessitates discrete-time modeling for practical implementations and computability. Therefore, for the remainder of this tutorial, we will assume that the time axis is discretized and defined by a regularly sampled time grid. Let F_s denote the sampling rate (given in Hertz), which specifies the number of samples per second defined by this grid. Furthermore, we assume that the time axis covers only a finite time interval, starting with time position $t = 0$. Then the discrete-time axis can be encoded by a finite set of integers $[0 : N-1] := \{0, 1, \ldots, N-1\}$, where $N \in \mathbb{N}$ is the total number of grid points and the index $n \in [0 : N-1]$ corresponds to the time point

$$t = \frac{n}{F_s} \tag{1}$$

given in seconds. In the following, we consider a novelty function to be a discrete-time function given by

$$\Delta : [0 : N-1] \to \mathbb{R}, \tag{2}$$

which specifies, for each time index $n \in [0 : N-1]$, the novelty value $\Delta(n)$. For the sake of easier understanding and intuition, we continue to display a discrete-time novelty function in our visualizations, with the time axis specified in seconds and using Equation (1) to convert indices to physical time.

2.2 ONSET DETECTION

The objective of *onset detection* is to identify the physical start times of musical events in a recording, which are often characterized by a sudden increase in energy from percussive sound components, such as those produced by a drum or the hammer strike of a piano. This is exemplified in Figure 2, with an excerpt from "Another One Bites the Dust" by Queen, featuring sound events of various instruments. Time positions where the amplitude envelope begins to rise can indicate note or sound onsets, although finding such positions is more challenging with non-percussive musical sounds, due to soft transitions and complex polyphonic music occurring because of masking effects.

Instead of directly tracking energy changes, many onset-detection methods detect changes across different frequency regions, leading to an audio feature referred to as *spectral flux*. This involves converting the music recording into a spectrogram using a short-time Fourier transform (STFT), as detailed in (Müller, 2021). While the STFT parameters, including the window size, the window type, and the hop size, can be crucial for the properties of the resulting novelty function, these aspects are not covered further in this tutorial. In the following paragraphs, we assume that the STFT is represented by a matrix $\mathbf{X} \in \mathbb{C}^{N \times K}$ that encodes the phase and magnitude at each time index $n \in [0 : N-1]$ and at each frequency index $k \in [0 : K-1]$, with $K \in \mathbb{N}$ denoting the number of frequency bands. The magnitude spectrogram $|\mathbf{X}|$ of our Queen example is shown in Figure 2(c).

To enhance the small, yet acoustically significant, frequency components that are barely visible in the spectrogram, one can apply *logarithmic compression*. This involves applying a logarithm to the magnitude spectrogram $|\mathbf{X}|$, yielding

$$\mathbf{Y} = \log(1 + \gamma \cdot |\mathbf{X}|), \tag{3}$$

with a hyperparameter $\gamma > 0$ used to control the degree of compression; see also (Klapuri et al., 2006). This compression not only accounts for the human perception of sound intensity but also equalizes the signal's dynamic range.

Finally, one computes the discrete temporal derivative of the compressed spectrogram $\mathbf{Y}$, and accumulates the resulting frequency-dependent changes. Here, only positive changes are considered, under the assumption that onsets correspond to energy increases rather than decreases. This process yields the novelty function $\Delta : [0 : N-2] \to \mathbb{R}$, which is defined by

$$\Delta(n) := \sum_{k=0}^{K-1} |\mathbf{Y}(n+1,k) - \mathbf{Y}(n,k)|_{\geq 0} \tag{4}$$

for $n \in [0 : N-2]$. In this definition, we use an operation known as *half-wave rectification*, which is defined for a real number $x \in \mathbb{R}$:

$$|x|_{\geq 0} := \begin{cases} x & \text{for } x \geq 0, \\ 0 & \text{for } x < 0. \end{cases} \tag{5}$$

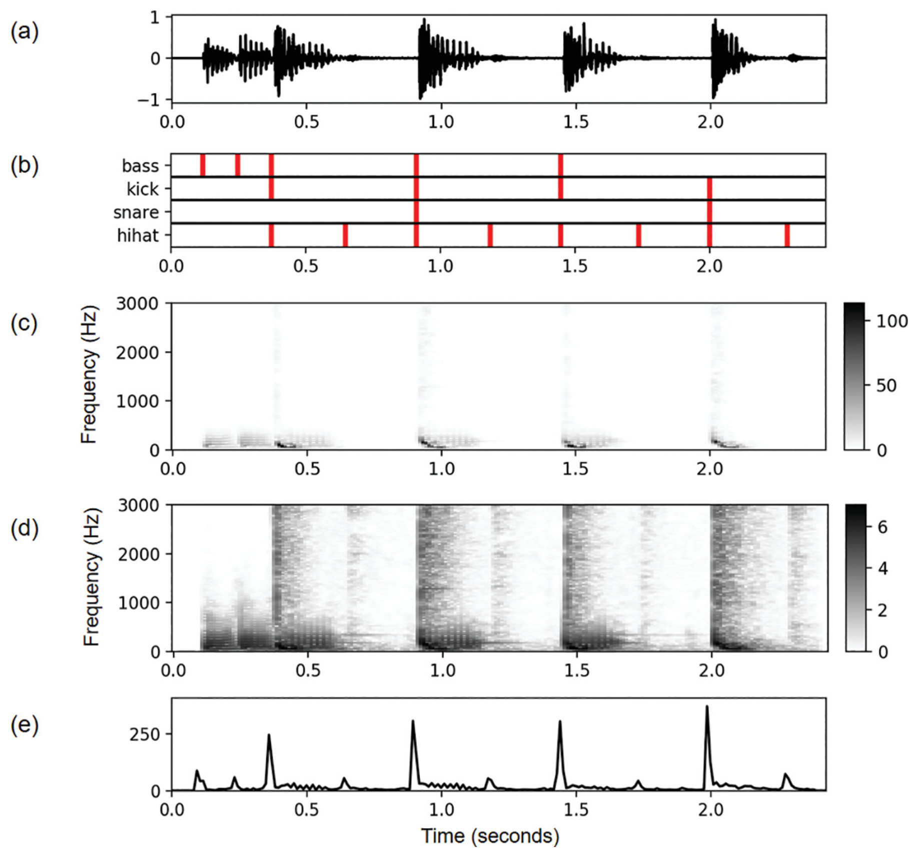

Figure 2 Computation of a novelty function based on spectral flux used for onset detection. The music excerpt corresponds to the beginning of the song "Another One Bites the Dust" by Queen. **(a)** Music signal (shown as waveform). **(b)** Annotation of target onsets for the applicable instruments. **(c)** Magnitude spectrogram. **(d)** Compressed magnitude spectrogram. **(e)** Novelty function.

Figure 2(e) shows the resulting novelty function for our example, where the peaks accurately indicate the onset positions of all sound events. Notably, the four most dominant peaks correspond to drum hits. Even at time points where only the hihat is active, one can observe four smaller peaks, although these sound events are barely visible in the waveform. This detection is possible due to the weak but still characteristic broad-band transients that are enhanced by the compression.

At this point, we want to emphasize that spectral flux, as introduced in this section, serves only as an illustrative and simple example of how one may obtain a novelty function for onset detection. In many situations, spectral flux fails to reliably detect note onsets, thus necessitating more refined methods. For instance, when singing a tone or playing a note on a violin, musicians often use a technique called vibrato, which involves a pulsating change in frequency (*frequency modulation*), usually in the range of 5 to 7 oscillations per second. Spectral flux is sensitive to these modulations, typically resulting in a large number of spurious peaks in the resulting novelty function that are not related to the actual note beginning. In these scenarios, techniques such as logarithmic compression may even further amplify fluctuating and noise-like sound components, thus increasing the number of spurious peaks. Here, more refined techniques are required that, for example, suppress vibrato effects (Böck and Widmer, 2013) or reduce the effect of non-stationary background noise (Lostanlen et al., 2019).

2.3 MUSIC STRUCTURE ANALYSIS

The general goal of *music structure analysis* is to divide a music representation into temporal segments that correspond to musical parts and to categorize these segments into meaningful groups. An example is illustrated in Figure 3 with a recording of the "Hungarian Dance No. 5" by Johannes Brahms, structured as $A_1A_2B_1B_2CA_3B_3B_4$. This structure includes three A-parts and four B-parts that repeat, along with a unique C-part. Repetition is key to identifying musical form, often with each section being homogeneous in aspects such as instrumentation, tempo, or harmony. For instance, contrasting sections might be homogeneous in harmony but differ in musical key, as seen in the Brahms example with A-parts and B-parts in G minor and the C-part in G major, as illustrated by Figure 3(b).

Rather than recovering the entire musical form, a subtask within music structure analysis is known as *boundary detection*. Here, the objective is to identify the time points where one musical part ends and the next musical part starts (e.g., the end of the B_2 part in G minor and the beginning of the C-part in G major; see Figure 3(b)). In this context, the concept of novelty-based boundary detection is designed to identify transitions between two contrasting yet homogeneous segments. Following the classical approach by Foote (2000), we now proceed to derive a novelty function for this segmentation task.

The first step in boundary detection is converting the music signal into a feature representation that captures essential musical information like melody, harmony,

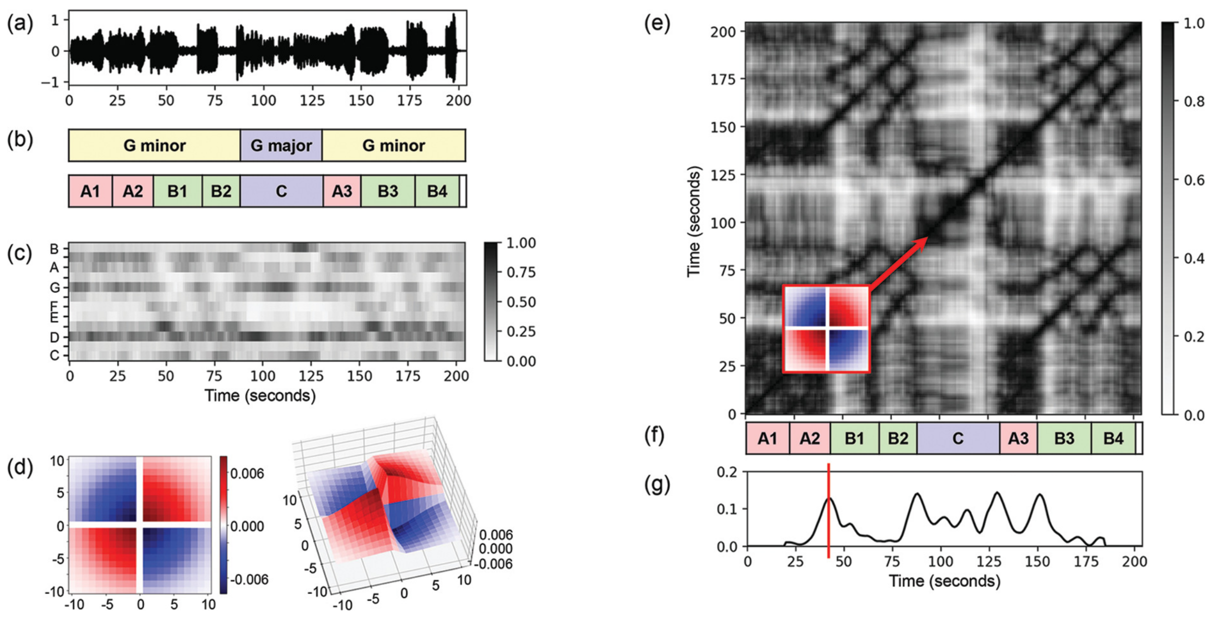

Figure 3 Novelty-based boundary detection for music structure analysis. The example is based on a recording of the "Hungarian Dance No. 5" by Johannes Brahms. **(a)** Music signal (shown as waveform). **(b)** Annotation of local key segments and the musical form. **(c)** Chromagram. **(d)** Checkerboard kernel shown as 2-dimensional and 3-dimensional plot. **(e)** Self-similarity matrix with kernel shifted along the main diagonal. **(f)** Annotation of the musical form. **(g)** Novelty function with a vertical line indicating the position of the kernel shown in (e).

rhythm, or timbre. For instance, when identifying boundaries associated with key changes, chroma-based audio features are particularly effective. These features utilize the 12 -tone equal-tempered scale, splitting pitch into tone height (octave number) and chroma, which corresponds to the twelve pitch-spelling attributes $C, C^{\#}, D, \ldots,$ B. Chroma features are computed for each time point by aggregating the spectral information of all pitches that share the same chroma across octaves into a single coefficient. This results in a time-chroma representation, also known as a *chromagram*, which offers robustness against timbre variations and closely aligns with the harmonic aspects of music; see Figure 3(c) for an example.

Mathematically, a feature representation like a chromagram is modeled as a sequence $(\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{N-1})$ with feature vectors $\mathbf{v}_n$ for each time index $n \in [0 : N-1]$. Often, the cosine similarity measure, defined as the cosine of the angle between two vectors, is used to compare two feature vectors. Assuming that all components of the feature vectors are non-negative (which is, e.g., the case when using magnitude- or probability-based feature values), a cosine similarity value is high (close to 1) if the two vectors are similar (i.e., if they point in the same direction) and low (close to 0) if they are dissimilar (i.e., if they point in orthogonal directions). Comparing all feature vectors pairwise results in a self-similarity matrix (SSM), which we denote as $\mathbf{S} \in \mathbb{R}^{N \times N}$. The values $\mathbf{S}(n, m)$ of this matrix are defined as the cosine similarity between vectors $\mathbf{v}_n$ and $\mathbf{v}_m$ for $n, m \in [0 : N-1]$.

Self-similarity matrices are an important tool for displaying structural properties of the feature representation: Repetitions appear as path-like structures, and homogeneous segments appear as block-like regions; see Figure 3(e) for an example. In particular, the block structures of the SSM along its main diagonal represent consecutive contrasting homogeneous segments, and the corner points on the main diagonal of such blocks characterize musical boundaries. These corner points can be captured as follows: A checkerboard-like kernel (as shown in Figure 3(d)) is shifted along the main diagonal of the SSM, and its correlation with the corresponding submatrix of the SSM is computed; see Figure 3(e). This process defines a novelty function in which the time index corresponds to the kernel's position and the function's value corresponds to the correlation result. High values in the novelty function occur when the shifted kernel is positioned at the corner point of a checkerboard-like structure in the SSM, indicating significant changes. Otherwise, the correlation, and thus the value of the novelty function, is low. For mathematical details, we refer to (Müller, 2021; Paulus et al., 2010). As can be seen in Figure 3(g), the novelty function of our Brahms example yields a particularly high value at the transition from the B_2-part to the C-part, which is characterized by a key change from G minor to G major.

2.4 DATA-DRIVEN NOVELTY COMPUTATION

Both methods of computing novelty functions for onset and boundary detection rely on traditional signal-processing techniques and follow a similar computational pipeline: Initially, the music signal is converted into a feature representation, and a novelty measure is applied to detect changes in these feature sequences. In recent years, data-driven approaches based on deep learning (DL) have provided an alternative by integrating feature extraction and novelty measurement within a single framework. These approaches often treat novelty detection as a frame-wise binary classification problem in which the network learns to predict the probability or likelihood of a relevant musical event occurring at a given time point. These predicted values yield what is commonly referred to as an *activation function* in the deep learning literature. In our scenario, this activation function serves as a novelty function that assigns each index $n \in [0 : N-1]$ to a pseudoprobability value ranging from 0 to 1.

Data-driven approaches to novelty detection offer several advantages: First, they often provide a novelty function within a probabilistic framework (e.g., when rephrasing the segmentation problem as a framewise classification problem with a softmax output layer), thus allowing for natural normalization and interpretation. Second, neural networks can be designed to implicitly learn signal properties relevant to novelty detection (e.g., in their hidden layers) without the need for explicit modeling of musical or acoustical properties. Even though one may argue that data-driven deep learning techniques are also prone to inductive biases (stemming from design choices made in, e.g., the network architecture, data preparation, and input feature computation), these approaches tend to be versatile for applications to different music types and various MIR tasks.

However, such data-driven approaches also come with limitations. First, they typically require training data with highly accurate reference annotations, such as audio excerpts with annotated note onsets or musical boundaries. Such annotations are often not available in large numbers, and their creation may require significant manual labor by domain experts. Secondly, although novelty occurrences are defined locally, their identification and understanding may require both temporal context and long-term musical knowledge. This is typically addressed by providing networks with entire input patches (Schlüter and Böck, 2014) or by using recurrent networks such as bidirectional LSTM to capture longer-term dependencies (Eyben et al., 2010). Thirdly, the rarity of relevant changes in novelty detection leads to class imbalance, where the network may frequently predict non-activity (i.e., probability 0), potentially resulting in an all-zero novelty function. Additionally, minor inaccuracies in annotations can degrade network performance. One solution to this is a technique called target smearing, in which neighboring frames around active frames are also marked as active. This strategy may be combined with a Gaussian weighting strategy during the learning stage to emphasize the relevance of the originally annotated frames (Ullrich et al., 2014).

In summary, deep learning has enabled significant advancements in MIR tasks involving novelty detection, thanks to deep learning's ability to learn relevant structures directly from training data. In practice, numerous design choices remain, including network architecture, data preparation, input representation, target smearing, and class imbalance compensation, among others. Additionally, training these deep neural networks is an art in itself, often requiring complicated optimization approaches and sometimes leading to inconsistent and unpredictable outputs. In Section 3, we will discuss strategies that address some of these issues, including the use of ensembling approaches to help reduce model uncertainty. Since deep learning is not the focus of this tutorial, we refer to the cited literature mentioned above for further reading on this topic.

3 ENHANCEMENT STRATEGIES

As previously discussed, novelty functions should ideally capture relevant musical changes while being robust against irrelevant signal variations, background noise, and random fluctuations. However, in practice, novelty functions computed via signal processing methods or obtained through activation functions using data-driven machine-learning techniques often fall short of these abilities. In this section, we explore general enhancement strategies that aim to improve the prominence of relevant peaks, suppress small fluctuations, and ensure that the areas between relevant peaks remain close to zero. To illustrate some of these strategies, Figure 4(c) displays a spectral-based novelty function computed for an audio excerpt (orchestral version) from Dimitri Shostakovich's "Waltz No. 2, Suite for Variety Orchestra No. 1." For context, Figure 4(a) shows the musical score of a piano-reduced version of this audio excerpt, and Figure 4(e) shows the annotations for onsets, beats, and downbeats. The first beats, which are played softly by nonpercussive instruments, leading to weak onsets, correspond to less pronounced peaks, while the sharp "staccato" notes played with percussive support in the second and third beats produce higher peaks. The excerpt will serve as our running example in the next two sections.

3.1 SMOOTHING

A simple but important strategy to reduce local fluctuation is smoothing the novelty function. This can be done in many ways (e.g., already at the stage of computing the novelty function). For example, when calculating spectral flux (see Section 2.2), increasing the STFT window size will smooth the spectrogram, which in turn has a smoothing effect on the resulting novelty function. Similarly, in the context of boundary detection (see Section 2.3), smoothing the input feature representation (such as the chromagram) or increasing the size of the checkerboard kernel will smooth the novelty function. Furthermore, when computing an activation function using deep

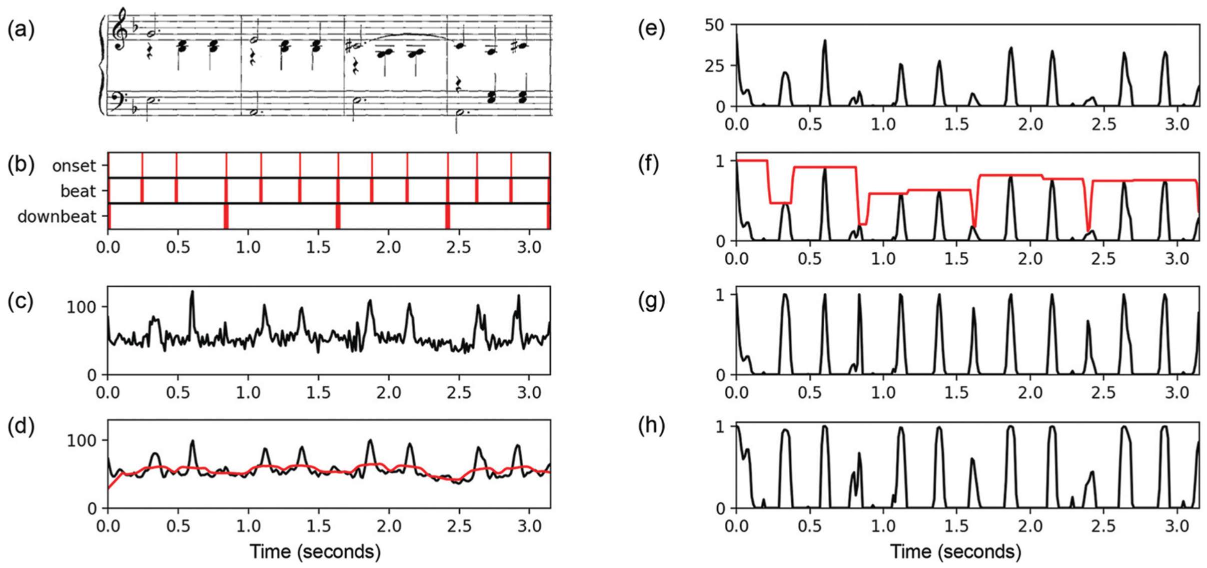

Figure 4 Enhancement strategies. The example is based on an audio excerpt (orchestral version) from Dimitri Shostakovich's "Waltz No. 2, Suite for Variety Orchestra No. 1." **(a)** Musical score of piano-reduced version. **(b)** Annotations of onset, beat, and downbeat positions. **(c)** Novelty function based on spectral flux (see Figure 2). **(d)** Smoothed novelty function and local average function μ (red curve). **(e)** Enhanced novelty function $\overline{\Delta}$. **(f)** Max-normalized novelty function and local max function (red curve). **(g)** Novelty function after local-max normalization. **(h)** Novelty function from (f) after applying the hyperbolic tangent function.

learning (Section 2.4), methods such as target smearing may also have smoothing effects.

Alternatively, direct smoothing of a novelty function can be achieved by applying a low-pass filter (e.g., via convolution with a Gaussian function), as illustrated in Figure 4(d). The degree of smoothing is a trade-off between removing small artifacts and retaining sufficient information. Finally, we want to emphasize that smoothing is crucial when applying resampling strategies to change the novelty function's sampling rate. In particular, downsampling may distort or even remove sharp, impulse-like structures in the novelty function without prior smoothing.

3.2 LOCAL THRESHOLDING

Furthermore, to compensate for a constant offset or bias from zero as well as to reduce the effect of random noise–like fluctuations in novelty functions, one may remove all values below a specified threshold and suitably shift the novelty function. Instead of using a global threshold, employing a local threshold can be more effective. To achieve this, one first determines a local average function $\mu : [0 : N-1] \to \mathbb{R}$ as follows:

$$\mu(n) := \frac{1}{2M+1} \sum_{m=-M}^{M} \Delta(n+m), \tag{6}$$

where $n \in [0 : N-1]$, and the parameter $M \in \mathbb{N}$ sets the size of the averaging window. To prevent summation over invalid index regions (e.g., negative indices) in this equation, one typically applies padding strategies (e.g., zero padding, extending the novelty function with zeros to the left and right). These approaches, however, may introduce additional artifacts that need to be handled with care. The enhanced novelty function $\overline{\Delta}$ is then obtained by subtracting this local average from Δ and keeping only the positive values (half-wave rectification; see Equation (5)):

$$\overline{\Delta}(n) := |\Delta(n) - \mu(n)|_{\geq 0} \tag{7}$$

for $n \in [0 : N-1]$. Figure 4(d) illustrates the local average function μ, and Figure 4(e) illustrates the resulting enhanced novelty function $\overline{\Delta}$.

3.3 NORMALIZATION

Normalization is a crucial enhancement strategy for novelty functions because it standardizes values within a specific range, typically 0 to 1. This standardization balances the values, making them comparable and treatable as pseudoprobabilities within probabilistic frameworks. The simplest method is *max normalization*, in which the entire novelty function is divided by its maximum value by scaling all values to the range 0 to 1, as shown in Figure 4(f). However, this procedure is sensitive to outliers, as a single large value may push most of the other values close to 0. Additionally, the normalization process is sensitive to the section chosen, meaning that using a subsection of the novelty function can result in a different normalization.

To address this issue, one might apply local max normalization by first deriving a local max function (computed similarly to the local average function in Equation (6)); see Figure 4(f). Pointwise division by this function leads to local normalization, as illustrated in Figure 4(g). However, this procedure must be handled with care to avoid dividing by zero, which can be mitigated by setting the minimum value of the local max function to a positive threshold. Additionally, local normalization can have unintended effects, such as amplifying very small, noise-like components to larger values

Finally, another common normalization method, also frequently used in deep learning, involves applying a function like the hyperbolic tangent (tanh), which maps the value $x = 0$ to zero and compresses positive values $x > 0$ between 0 and 1, as defined by

$$\tanh(x) := 1 - \frac{2}{1 + \exp(2x)}. \tag{8}$$

The effect is illustrated in Figure 4(h). Similar to the logarithmic compression defined in Equation (3), a hyperparameter can be introduced to control the degree of compression.

3.4 ENSEMBLING

We conclude this section by discussing another general enhancement strategy that involves combining several independently computed novelty functions. In general, *ensembling* refers to the process of combining multiple models to produce a stronger and more accurate model. By aggregating the predictions of several models, the ensemble can reduce errors and improve performance compared to any single model. This strategy can be applied to novelty functions that are computed based on different feature representations or enhancement strategies, with aggregation methods that extend beyond simple averaging, such as using median values or voting approaches. A specific type of ensemble method known as *bagging* (short for "bootstrap aggregating") in the machine learning literature (Breiman, 1996) combines multiple versions of a model trained on different subsets of the training data. This ensembling technique enhances the stability and accuracy of data-driven approaches by reducing variance and preventing overfitting. Although bagging increases computational load due to the need to train multiple models, it often yields more accurate and robust novelty functions, especially in complex prediction tasks.

4 POST-PROCESSING AND EVALUATION

Understanding and assessing the properties of a novelty function is not straightforward and is typically conducted within the context of a specific MIR application, such as onset detection, beat tracking, or music structure analysis. We first introduce how to evaluate estimated peak positions against reference positions using threshold-dependent precision, recall, and F-measure (Section 4.1). Then we demonstrate how the peak positions can be selected using simple peak-picking strategies (Section 4.2). These methods are instructive, rely on only a few assumptions, and highlight the fundamental characteristics of novelty functions. More advanced post-processing strategies are outlined in Section 4.3.

4.1 PRECISION, RECALL, F-MEASURE

In music segmentation, a common evaluation approach involves comparing a set of *estimated* time positions (e.g., derived from the peak positions of a novelty function) with a list of *reference* time positions (also known as *ground truth* or *target* positions). In onset detection, for example, these reference positions may be provided by a human annotator or by sensor technology, such as Disklavier technology[2] for pianos, which captures this information during a performance. The objective of the evaluation measure is to assess how well the reference positions are covered by the estimated positions. Depending on the application's requirements, small inaccuracies may be acceptable as long as they are within a defined tolerance (expressed by a tolerance parameter $\tau > 0$). For instance, in the evaluation of onset detection approaches, a tolerance of up to 50 milliseconds is often chosen (which is roughly twice the duration a human requires to distinguish between two separate complex or real-world sounds). In music structure analysis, a tolerance of more than 500 milliseconds (which corresponds to a beat duration at a tempo of 120 beats per minute) may still be considered acceptable.

In the following, we assume that the estimated time indices $e_\ell \in [0 : N-1]$ for $\ell \in [1 : L]$ are provided by a list $(e_1, e_2, \ldots, e_L)$ of length $L \in \mathbb{N}$, sorted by increasing indices. Similarly, we assume that the reference time positions are given by a list $(r_1, r_2, \ldots, r_M)$ of length $M \in \mathbb{N}$. As illustrated in Figure 5, an estimated position e_ℓ is considered *correct* and is called a *true positive* (TP) if it falls within the τ-neighborhood of a reference boundary r_m:

$$|e_\ell - r_m| \leq \tau. \tag{9}$$

If this is not the case, then e_ℓ is called a *false positive* (FP). Additionally, a reference position r_m is termed a *false negative* (FN) if there is no estimated position within its τ-neighborhood.

Due to the tolerance parameter τ, multiple estimated positions may fall within the τ-neighborhood of a single reference position, and, conversely, a single estimated position may cover multiple reference positions. This can lead to anomalies in evaluation measures, an issue that can be resolved by requiring that the distance between consecutive boundaries must exceed twice the tolerance parameter:

$$|e_{\ell+1} - e_\ell| > 2 \cdot \tau \quad \text{and} \quad |r_{m+1} - r_m| > 2 \cdot \tau \tag{10}$$

for $\ell \in [1 : L-1]$ and $m \in [1 : M-1]$, respectively. This requirement also holds semantic significance, as boundaries that are too close together might indicate that the defined segments lack substantial musical meaning or distinctiveness. For example, considering structural parts, a section lasting less than a second is unlikely to occur. Similarly, two subsequent beats should be spaced more than 100 milliseconds apart, implying a tempo of less than 600 beats per minute. For alternatives to using the constraints expressed in Equation (10) (e.g., based on bipartite graph matching), we refer to (Raffel et al., 2014).

With these definitions and requirements in place, the *precision* P_τ of the estimation is determined by dividing

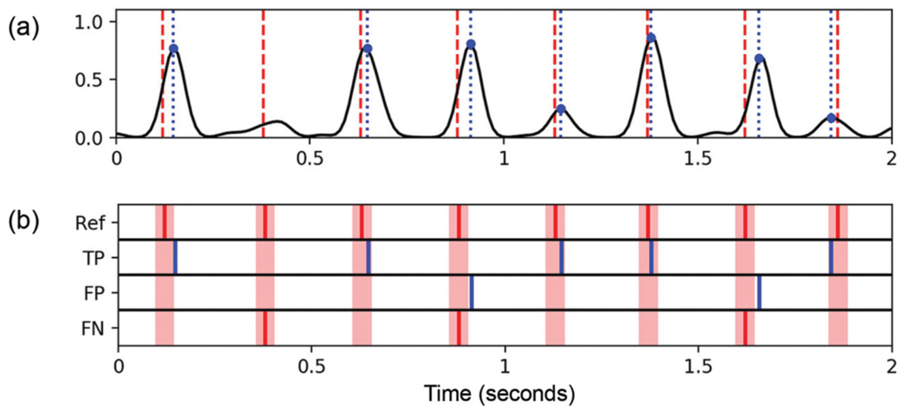

Figure 5 Illustration of precision, recall, and F-measure with error tolerance. **(a)** Novelty function with estimated positions (blue dotted lines) and reference positions (red dashed lines). **(b)** Illustration of reference positions (Ref), true positives (TP), false positives (FP), and false negatives (FN), with error tolerance $\tau = 0.025$ (indicated by red-shaded regions). One has #TP = 5, #FP = 2, and #FN = 3, thus yielding $\mathrm{P}_\tau = 5/7 = 0.714, \mathrm{R}_\tau = 5/8 = 0.625$, and $\mathrm{F}_\tau = 0.667$.

the number of true positives by the total number of estimated positions:

$$\mathrm{P}_\tau = \frac{\#\mathrm{TP}}{\#\mathrm{TP} + \#\mathrm{FP}} = \frac{\#\mathrm{TP}}{L}. \tag{11}$$

In contrast, the *recall* R_τ is defined as the number of true positives divided by the total number of reference positions:

$$\mathrm{R}_\tau = \frac{\#\mathrm{TP}}{\#\mathrm{TP} + \#\mathrm{FN}} = \frac{\#\mathrm{TP}}{M}. \tag{12}$$

Note that both precision and recall have values in the range between 0 and 1. A perfect precision, $\mathrm{P}_\tau = 1$, means that every estimated position is correct. In contrast, a perfect recall, $\mathrm{R}_\tau = 1$, means that every reference position is matched to an estimated position. Precision and recall are often combined by taking their harmonic mean to form a single measure, commonly referred to as the *F-measure*:

$$\mathrm{F}_\tau = \frac{2 \cdot \mathrm{P}_\tau \cdot \mathrm{R}_\tau}{\mathrm{P}_\tau + \mathrm{R}_\tau}. \tag{13}$$

For an example, we refer to Figure 5. A key property of the F-measure is that it lies in the range between 0 and 1, with $\mathrm{F}_\tau = 1$ if and only if $\mathrm{P}_\tau = 1$ and $\mathrm{R}_\tau = 1$. The tolerance parameter τ can be viewed as a hyperparameter that can be adjusted to meet specific requirements or to provide different perspectives on accuracy by considering sweeps over this parameter. Note that the threshold-dependent versions generalize the notion of the standard definitions of precision, recall, and F-measure used in information retrieval. For further details, we refer to (Lukashevich, 2008; Müller, 2021). For an implementation of the most common metrics used in general MIR research, we refer to the open-source Python library `mir_eval` (Raffel et al., 2014).

4.2 PEAK PICKING

Peak picking is a critical step in identifying significant points in a novelty function. It can be challenging due to noise and irregularities that may generate spurious peaks. However, effective strategies for peak picking often involve enhancing the novelty function to reduce noise, applying adaptive thresholding to suppress insignificant peaks, and setting constraints on the minimum distance between peaks to prevent selecting those peaks that are too closely spaced. For illustration, we continue with our running Shostakovich example, as shown in Figure 6, which we consider to be of a medium-difficulty level for peak picking.

Intuitively, a peak is a local maximum characterized by the property that the novelty function shifts from an increasing state (positive derivative) to a decreasing state (negative derivative). Therefore, a simple peak picking approach searches for all such local maxima. As illustrated by Figure 6(b), this method may lead to many spurious peaks. Using a global threshold to discard small, noise-like peaks that fall below it can eliminate many of these spurious peaks, as seen in Figure 6(c). However, this may also lead to losing relevant peaks, potentially increasing the number of false negatives. As an alternative, as is illustrated by Figure 6(d), one may apply adaptive thresholding to select a peak only if its value exceeds a local average of the novelty function, which can be computed as shown in Equation (6).

Another simple but very effective heuristic used in peak picking involves introducing a distance parameter that specifies the required minimum temporal distance between neighboring peaks, as depicted in Figure 6(e). One straightforward approach to achieving this is to successively disregard the smallest peaks until the condition is met for all remaining peaks. The distance condition is particularly beneficial for MIR tasks that inherently

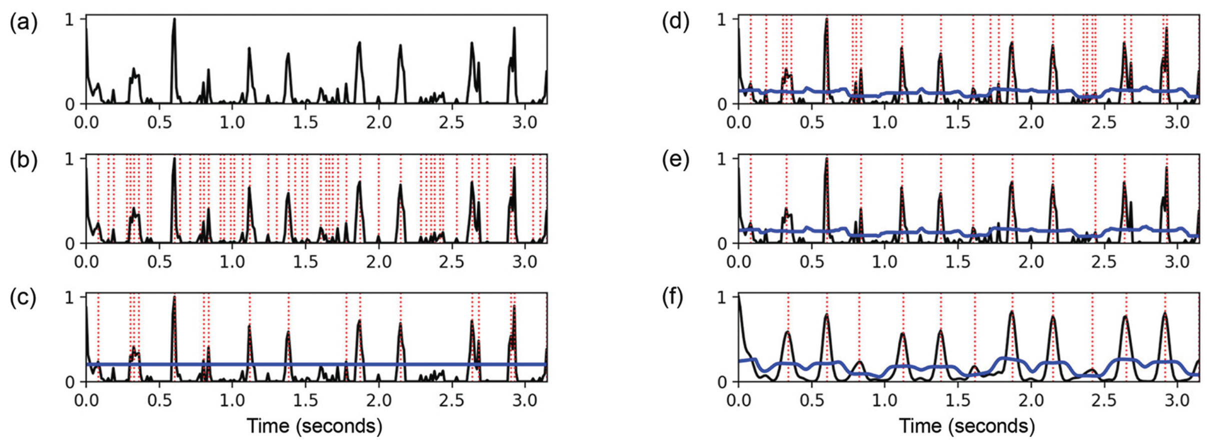

Figure 6 Illustration of peak picking based on different heuristics. **(a)** Novelty function. **(b)** Simple peak picking (selected peak positions are indicated by red vertical lines). **(c)** Usage of a global threshold (shown as a blue horizontal line). **(d)** Usage of a local threshold (shown as a blue curve). **(e)** Usage of an additional distance constraint. **(f)** Application of Gaussian smoothing, max normalization, and a local threshold.

demand minimum distance bounds. For example, in structure analysis, the expected minimum duration of musical sections defines a minimum distance between adjacent structural boundaries. Similarly, in beat tracking, the expected maximum tempo necessitates a minimum distance between successive beat positions.

There are many additional heuristics and refinements for peak picking, such as applying prominence to highlight peaks that stand out distinctly from their surrounding peaks. More sophisticated approaches may involve global optimization strategies that jointly consider multiple criteria such as peak height, prominence, distance, and regularity. We will address one such approach in Section 4.3.

As can be seen, novelty enhancement and peak picking heuristics are closely related. We want to emphasize at this point that before applying an overly complicated peak picking strategy based on several heuristics, it can be highly beneficial to apply a Gaussian smoothing filter to the novelty function (with the Gaussian variance informed, for example, by the tolerance parameter used in the evaluation). As demonstrated in Figure 6(f), this can significantly reduce the impact of noise-like fluctuations, simplify the peak picking process, and substantially improve the final analysis results.

At this point, we want to emphasize that we have only touched on some common strategies for peak picking and that we do not intend to propose a single solution. Peak picking must be done with care, and the combination of heuristics and parameters will depend on the requirements and prior knowledge specific to the application. For experimenting with basic peak picking strategies, we recommend the scipy.signal package from the Python library SciPy (Virtanen et al. 2020). More advanced techniques, especially for onset detection, are detailed in (Böck et al., 2012). A variety of peak picking methods are also compared in Part 6 of the FMP notebooks[3]; see (Müller and Zalkow, 2019).

4.3 FURTHER POST-PROCESSING APPROACHES

As mentioned earlier, various heuristics can be used to guide the selection of peaks from a novelty function, with many peak picking strategies designed to balance several criteria at the same time. These criteria are typically tailored to the needs of specific MIR applications In beat tracking; for example, it is practical to assume that beat positions coincide with note onsets and are uniformly spaced, mirroring the pulse a person taps while listening to music. Thus, a beat sequence is typically characterized by evenly spaced onset positions. Indeed, many music genres feature a strong and steady beat, with a relatively constant tempo throughout the recording.

4.3.1 *Tempo-informed optimal beat sequence*

Based on these assumptions, Ellis (2007) introduces a beat tracking procedure that starts with a novelty function indicating note onsets (e.g., computed using spectral flux). This method involves selecting time positions corresponding to strong peaks that are also regularly spaced in time, matching the expected tempo. In particular, a score function is introduced to assess how well a beat sequence meets the criteria of novelty strength and temporal regularity. Among all possible beat sequences, it computes the optimal beat sequence that maximizes this score, effectively constituting the final beat tracking result. Although computing scores for all potential beat sequences may seem daunting, dynamic programming facilitates efficient computation by minimizing redundant calculations. This approach makes it feasible to identify the optimal beat sequence with a runtime that is at most quadratic in the length N of the novelty function. The runtime can be further reduced by limiting the computation at each time step to a small search window informed by the expected tempo (e.g., double the period of the expected tempo). This heuristic is applied, for example, in the implementation provided by the Python package librosa (McFee et al., 2015).

4.3.2 *Hidden Markov models*

The main limitation of the beat tracking procedure previously described is its reliance on a single, predefined tempo. More advanced approaches, many based on concepts related to Hidden Markov Models (HMMs), accommodate potential tempo changes and also address other musical elements such as meter and rhythm. In these models, various aspects are represented by hidden states that can be inferred from audio data but are not directly observable. In (Krebs et al., 2015), an HMM is employed that incorporates state-space discretization and a tempo transition model, with a novelty function serving as the observation sequence. State-space discretization converts continuous musical features into discrete states, thus simplifying the model and reducing computational demands. The tempo transition model primarily maintains consistent tempi, permitting changes only at beat positions. Again, dynamic programming is employed to compute an optimal beat sequence that meets constraints related to tempo, meter, and other musical aspects.

4.3.3 *Predominant local pulse*

While targeting similar objectives as beat tracking, the method introduced by Grosche and Müller (2011) takes a conceptually different approach to deal with tempo changes and weak or even missing note onsets. The main idea of the computational pipeline is illustrated in Figure 7, which continues our Shostakovich example in Figure 4. First, one computes a novelty function that captures note onset information, as shown in Figure 7(c). The intuitive ideas are to locally compare the novelty function with windowed sinusoids of different frequencies and to consider, for each time position, the windowed

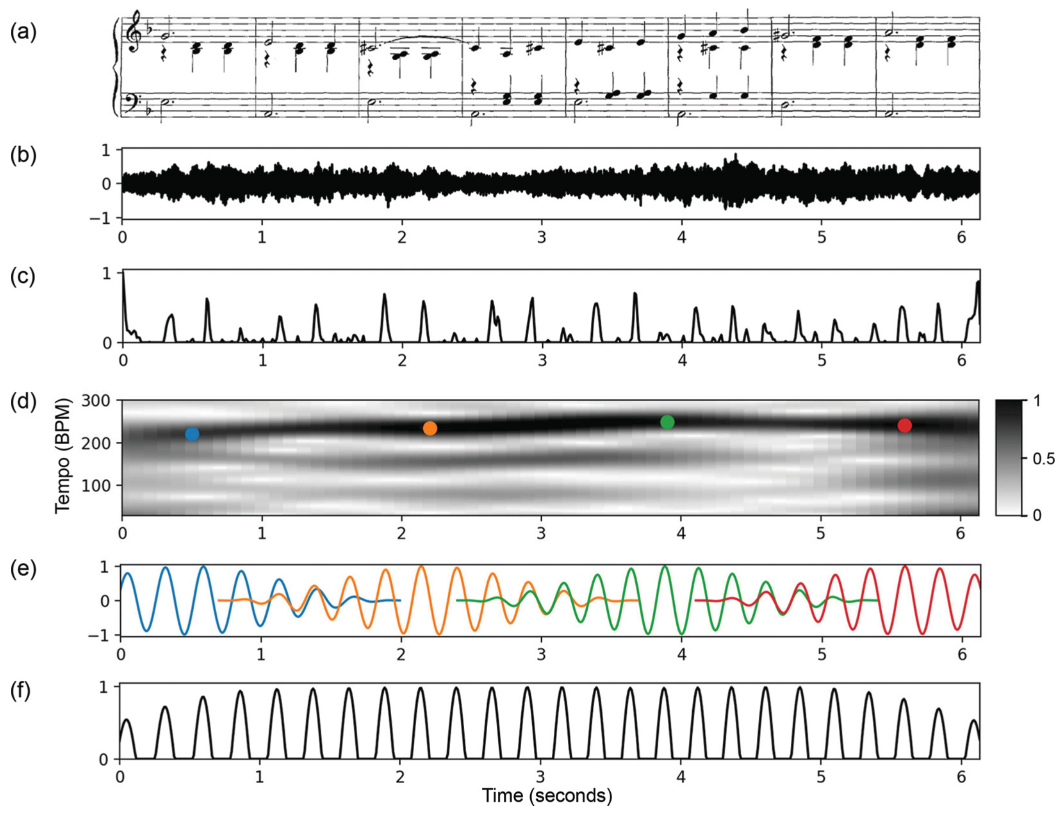

Figure 7 Illustration of the computational pipeline for deriving the predominant local pulse (PLP) function, continuing the Shostakovich example introduced in Figure 4, **(a)** Musical score of piano-reduced version. **(b)** Audio recording of the corresponding excerpt. **(c)** Novelty function. **(d)** Tempogram showing timedependent tempo maxima, indicated by colored dots. These dots are displayed at only four selected time positions for better visibility. **(e)** Optimal windowed sinusoids corresponding to the maxima. **(f)** PLP function obtained by applying overlap-add and half-wave rectification techniques.

sinusoid with the frequency and phase that best capture the local peak structure of the novelty function. To implement this idea, an STFT is employed to convert the novelty function into a time-frequency representation. Note that the frequency axis can be reinterpreted as a tempo axis, where 1 Hertz (one oscillation per second) corresponds to 60 beats per minute (BPM). This results in a time-tempo representation, also called a *tempogram*, as seen in Figure 7(d). For each time position, one then considers the maximal tempo value and derives from this the aforementioned optimal windowed sinusoid (where required frequency and phase information is obtained by the STFT); see Figure 7(e). Finally, an overlap-add technique is employed to accumulate all these windowed sinusoids over time, and half-wave rectification is used to retain only the positive parts. As a result, a single function is obtained that enhances the local periodicity of the original novelty function; see Figure 7(f). This function, referred to as the PLP function, reveals predominant local pulse (PLP) information. The term "predominant pulse" is loosely used to describe the strongest pulse level measurable in the novelty function. Intuitively, the PLP function acts as a pulse tracker, which is capable of adjusting to both continuous and sudden tempo changes, provided that the underlying novelty function displays locally periodic patterns.

5 EDUCATIONAL INTEGRATION AND RECOMMENDATIONS

We conclude this tutorial with a summary and some thoughts on the intended audience and the potential curriculum integration for this material. Additionally, we offer recommendations for software and datasets that facilitate hands-on experience and further exploration.

5.1 SUMMARY

In this tutorial, we explored the concept of novelty functions, which are time-dependent functions with peak-like structures that are used to identify significant changes in a time series or in a feature sequence representation. Using concrete MIR tasks as examples, we demonstrated how to construct and enhance these functions. We also discussed peak identification through basic peak picking and more sophisticated post-processing procedures and also discussed their evaluation using precision, recall, and the F-measure, including error tolerance.

The concept of novelty functions extends beyond MIR and is applicable to general time series analysis and multimedia processing. We considered the context of music to motivate constructions such as spectral flux using spectrograms and kernel-based novelty detection using self-similarity matrices. We hope that these examples are straightforward, explicit, and easy to understand, also motivating the most important characteristics that should be captured by novelty functions in general. Once a novelty function is constructed, the enhancement and peak-picking strategies discussed in this tutorial are generic and can be applied to any novelty or activation function with a peak-like structure.

5.2 TARGET READERSHIP AND CURRICULUM

As mentioned in the introduction, this tutorial is geared toward students who have basic signal processing and mathematics skills typically acquired at the Bachelor's degree level in computer science or engineering. In particular, students should be familiar with concepts including audio signals, Fourier transforms, spectral representations, and filtering techniques. We also assume some exposure to or interest in music, with a basic understanding of concepts such as notes, beats, downbeats, and musical form, although no deeper musicological knowledge is required.

At some points in this tutorial, we did not shy away from using mathematical notation to demonstrate how concepts can be expressed mathematically. Along with textual and mathematical descriptions, we provided illustrations of computational pipelines using concrete music examples, including score representations and annotations to give context. Thus, while the tutorial is technically oriented, we also aim for it to be accessible to a broader, even non-technically oriented, readership.

This basic tutorial, focusing on classical concepts of general relevance, is designed primarily for intermediate Bachelor's degree or beginning Master's degree courses. It provides sufficient material for a 90-minute lecture unit within a course on music information retrieval or music processing and covers classical MIR tasks such as beat tracking and music structure analysis. Additionally, this tutorial is suitable for general courses in computer science, multimedia engineering, information science, and digital humanities, for which the music domain serves as a practical example for studying complex and multifaceted time series.

5.3 SOFTWARE

This tutorial provides both a theoretical foundation for novelty functions and concrete applications such as beat tracking and boundary detection. Complementary lectures and hands-on experience are essential for deepening the understanding of these concepts and their practical relevance (Müller et al., 2021). Therefore, along with this tutorial, we provide Jupyter notebooks at https://github.com/groupmm/edu_novfct that allow for reproducing and experimenting with all the examples discussed in this article. Furthermore, suitable software packages like the Python package librosa (McFee et al., 2015), which is standard in MIR research, and libfmp (Müller and Zalkow, 2021), with its educational lens, enhance interactive learning by providing functions for various music segmentation tasks discussed in this tutorial.

Much of this tutorial's material is included in the FMP notebooks (Müller and Zalkow, 2019), which bridge theory and practice by integrating technical concepts, mathematical details, code examples, illustrations, and sound examples in a unified Jupyter notebook framework. This tutorial expands on the FMP notebooks by providing a compact account of novelty functions and their properties, offering new perspectives on this material.

Figures in the tutorial were generated using Python functions from `librosa` and `libfmp`. Alongside the code, we provide audio excerpts integrated into Jupyter notebooks that allow for reproducing all figures. The accompanying material demonstrates how the figures were generated and serves as a practical entry point for students to engage in hands-on experimentation using their own examples and to start conducting their own research.

For evaluation metrics, we recommend the Python package `mir_eval` (Raffel et al., 2014), and for simple peak picking strategies, we recommend the scipy.signal package from the SciPy library (Virtanen et al., 2020). Additionally, the Python library `madmom` (Böck et al., 2016) is a good starting point for exploring deep learning methods for computing and experimenting with activation functions for onset, beat, and downbeat estimation. The Python package `MSAF`[4] provides computational tools for music structure analysis (Nieto and Bello, 2016), including an implementation of the kernel-based novelty detection approach discussed in this article. Finally, we want to point to the ISMIR homepage, which includes an overview website of further MIR software tools.[5]

5.4 DATASETS

In addition to software, one requires datasets with suitable annotations for conducting systematic experiments in music segmentation and assessing methods across various music genres. Starting with beat tracking, the `Ballroom` dataset (Gouyon et al., 2006) and the `GTZAN` dataset (Tzanetakis and Cook, 2002) offer audio excerpts and annotations for diverse styles. Classical and non-Western music genres often pose specific challenges in beat tracking due to their variations in tempo and rhythmic complexity. The `Mazurka` dataset provides detailed annotations for beat tracking in classical music (Grosche et al., 2010).

Beyond beat tracking, the `RWCPop` dataset (Goto et al., 2002) features 100 pop songs, and the Beatles dataset (Harte et al., 2005) includes 180 songs from twelve Beatles albums, both providing extensive annotations for beats, structure, and other musical elements. The `SALAMI` database, one of the largest for music structure analysis, provides more than 2400 annotations on 1356 recordings (Smith et al., 2011), highlighting issues such as structural ambiguities and cross-annotator differences (Flexer, 2014). The Jazz Structure Dataset (`JSD`) (Balke et al., 2022) further enriches the field with more than 3000 segments annotated for structure and instrumentation, supporting tasks such as structure analysis and instrument recognition. Furthermore, we want to point out the open-source Python library `mirdata`,[6] which provides tools for working with common MIR datasets (Bittner et al., 2019).

Many of the datasets and music examples in this article are from Western music cultures. However, the concepts of novelty functions can also apply to other genres and types of music, given the principles of onset and musical boundary detection. Music examples from non-Western music cultures (including Hindustani, Carnatic, Turkish-makam, Arab-Andalusian, and Beijing Opera) can be found as part of the Dunya[7] research prototype that has been developed as part of the CompMusic[8] project (Serra, 2014).

Finally, we point to the Music Information Retrieval Evaluation Exchange (MIREX), an annual evaluation of MIR algorithms between 2005 and 2021, providing results, datasets, and reference implementations (Downie, 2008). For more details, we refer readers to the MIREX homepage.[9] Further pointers to dataset resources can be found on the ISMIR homepage.[10]

ACKNOWLEDGMENTS

We thank Peter Meier, Johannes Zeitler, and Simon Schwär for proofreading the manuscript and improving the Jupyter notebooks that accompany this article. Additionally, we thank the anonymous reviewers and the section editor for their helpful and constructive comments.

FUNDING INFORMATION

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grant No. 500643750 (MU 2686/15-1) and the National Science and Technology Council (NSTC), Taiwan, under Grant No. NSTC 112-2917-I-564-011. The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Fraunhofer Institute for Integrated Circuits IIS.

COMPETING INTERESTS

The authors have no competing interests to declare.

AUTHOR CONTRIBUTION

Meinard Müller was the primary contributor to the conceptualization and writing of this article. Ching-Yu Chiu contributed to the writing and to setting up the Jupyter notebooks that accompany the article.

NOTES

1. https://github.com/groupmm/edu_novfct
2. https://en.wikipedia.org/wiki/Disklavier
3. https://audiolabs-erlangen.de/FMP
4. https://pythonhosted.org/msaf/
5. https://ismir.net/resources/software-tools/
6. https://mirdata.readthedocs.io/
7. https://dunya.compmusic.upf.edu/
8. https://compmusic.upf.edu/
9. https://www.music-ir.org/mirex/wiki/MIREX_HOME
10. https://ismir.net/resources/datasets/

AUTHOR AFFILIATIONS

Meinard Müller https://orcid.org/0000-0001-6062-7524
International Audio Laboratories Erlangen, Friedrich-Alexander Universität Erlangen-Nürnberg, Am Wolfsmantel 33, 91058 Erlangen, Germany

Ching-Yu Chiu https://orcid.org/0000-0002-3671-8474
International Audio Laboratories Erlangen, Germany

REFERENCES

Balke, S., Reck, J., Weiß, C., Abeßer, J., and Müller, M. (2022). JSD: A dataset for structure analysis in jazz music. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, *5*(1), 156–172.

Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., and Sandler, M. B. (2005). A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, *13*(5), 1035–1047.

Benetos, E., Dixon, S., Duan, Z., and Ewert, S. (2019). Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, *36*(1), 20–30.

Bittner, R. M., Fuentes, M., Rubinstein, D., Jansson, A., Choi, K., and Kell, T. (2019). mirdata: Software for reproducible usage of datasets. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 99–106).

Böck, S., and Widmer, G. (2013). Maximum filter vibrato suppression for onset detections. In *Proceedings of the International Confenference on Digital Audio Effects (DAFx)*.

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., and Widmer, G. (2016). madmom: A new Python audio & music signal processing library. In *Proceedings of the ACM International Conference on Multimedia (ACM-MM)* (pp. 1174–1178).

Böck, S., Krebs, F., and Schedl, M. (2012). Evaluating the online capabilities of onset detection methods. In *Proceedings of* *the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 49–54).

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140. https://doi.org/10.1007/bf00058655

Downie, J. S. (2008). The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, *29*(4), 247–255. https://doi.org/10.1250/ast.29.247

Ellis, D. P. (2007). Beat tracking by dynamic programming. *Journal of New Music Research*, *36*(1), 51–60. https://doi.org/10.1080/09298210701653344

Eyben, F., Böck, S., Schuller, B., and Graves, A. (2010). Universal onset detection with bidirectional long short-term memory neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 589–594).

Flexer, A. (2014). On inter-rater agreement in audio music similarity. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 245–250).

Foote, J. (2000). Automatic audio segmentation using a measure of audio novelty. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)* (pp. 452–455).

Goto, M., Hashiguchi, H., Nishimura, T., and Oka, R. (2002). RWC music database: Popular, classical and jazz music databases. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 287–288).

Gouyon, F., Klapuri, A. P., Dixon, S., Alonso, M., Tzanetakis, G., Uhle, C., and Cano, P. (2006). An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(5), 1832–1844. https://doi.org/10.1109/tsa.2005.858509

Grosche, P., and Müller, M. (2011). Extracting predominant local pulse information from music recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, *19*(6), 1688–1701. https://doi.org/10.1109/tasl.2010.2096216

Grosche, P., Müller, M., and Sapp, C. S. (2010). What makes beat tracking difficult? A case study on Chopin Mazurkas. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 649–654).

Harte, C., Sandler, M. B., Abdallah, S., and Gómez, E. (2005). Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 66–71).

Klapuri, A. P., Eronen, A. J., and Astola, J. (2006). Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(1), 342–355. https://doi.org/10.1109/tsa.2005.854090

Krebs, F., Böck, S., and Widmer, G. (2015). An efficient state-space model for joint tempo and meter tracking. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 72–78).

Lostanlen, V., Palmer, K., Knight, E., Clark, C. W., Klinck, H., Farnsworth, A., Wong, T., Cramer, J., and Bello, J. P. (2019). Long-distance detection of bioacoustic events with per-channel energy normalization. In *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)* (pp. 144–148).

Lukashevich, H. (2008). Towards quantitative measures of evaluating song segmentation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 375–380).

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O. (2015). Librosa: Audio and music signal analysis in Python. In *Proceedings of the Python Science Conference* (pp. 18–25).

Müller, M. (2021). *Fundamentals of music processing using Python and Jupyter Notebooks* (2nd ed.). Springer Verlag.

Müller, M., and Zalkow, F. (2019). FMP Notebooks: Educational material for teaching and learning fundamentals of music processing. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 573–580).

Müller, M., and Zalkow, F. (2021). libfmp: A Python package for fundamentals of music processing. *Journal of Open Source Software (JOSS)*, *6*(63), 1–5.

Müller, M., McFee, B., and Kinnaird, K. (2021). Interactive learning of signal processing through music: Making Fourier analysis concrete for students. *IEEE Signal Processing Magazine*, *38*(3), 73–84. https://doi.org/10.1109/msp.2021.3052181

Nieto, O., and Bello, J. P. (2016). Systematic exploration of computational music structure research. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 547–553).

Paulus, J., Müller, M., and Klapuri, A. (2010). Audiobased music structure analysis. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 625–636).

Pauwels, J., O'Hanlon, K., Gómez, E., and Sandler, M. B. (2019). 20 years of automatic chord recognition from audio. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 54–63).

Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., and Ellis, D. P. W. (2014). `mir_eval`: A transparent implementation of common MIR metrics. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 367–372).

Schlüter, J., and Böck, S. (2014). Improved musical onset detection with convolutional neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 6979–6983).

Serra, X. (2014). Creating research corpora for the computational study of music: The case of the CompMusic project. In *Proceedings of the AES International Conference on Semantic Audio*.

Smith, J. B. L., Burgoyne, J. A., Fujinaga, I., Roure, D. D., and Downie, J. S. (2011). Design and creation of a large-scale database of structural annotations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 555–560).

Tzanetakis, G., and Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, *10*(5), 293–302. https://doi.org/10.1109/tsa.2002.800560

Ullrich, K., Schlüter, J., and Grill, T. (2014). Boundary detection in music structure analysis using convolutional neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)* (pp. 417–422).

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., and Larson, E. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-020-0772-5

TO CITE THIS ARTICLE:
Müller, M., & Chiu, C.-Y. (2024). A Basic Tutorial on Novelty and Activation Functions for Music Signal Processing. *Transactions of the International Society for Music Information Retrieval*, *7*(1), 179–194. DOI: https://doi.org/10.5334/tismir.202

Submitted: 7 May 2024 **Accepted:** 12 August 2024 **Published:** 19 September 2024