

Online Driveline Fatigue Data Acquisition Method

Hussein Dourra
Chrysler Group LLC

ABSTRACT

Two on-line algorithms have been developed to acquire driveline component loads in terms of revolutions at torque and rainflow cycle counting matrix. These algorithms have been implemented in real-time on a standard engine controller unit and have been optimized for fast run-time and low memory requirements. The revolutions at torque algorithm is intended to count the number of driveshaft revolutions in each torque level for each gear and store the number of counts in the engine controller memory. The rainflow cycle counting algorithm is intended to count driveshaft torque cycles and to store the number of counts in a two dimensional "from-to" matrix format in the engine controller memory. The revolutions at torque histogram data and the rainflow cycle counting matrix are then downloaded from the vehicle using the data collection device. Download occurs when the vehicle is serviced at a dealership. This data based on the real customer usage information can then be used to develop durability passing criteria for vehicle, system and component tests.

CITATION: Dourra, H., "Online Driveline Fatigue Data Acquisition Method," *SAE Int. J. Passeng. Cars – Electron. Electr. Syst.* 6(1):2013, doi:10.4271/2013-01-1270.

INTRODUCTION

The current durability validation process for powertrain systems and components has three different levels:

1. Vehicle level testing
2. System level testing
3. Component level testing

For vehicle level testing, vehicles are tested at proving grounds based on specific powertrain endurance schedules. For system level testing, offline system data such as torque at speed and gears is utilized to create transmission and differential dynamometer testing schedules. For component level testing, offline component data such as rainflow cycle techniques [1, 2, 3, 4, 5, 6, 7, 8] is used to develop bench test criteria.

The current vehicle durability design and validation target is based on a proving ground powertrain endurance schedule which was derived based on a 95th percentile customer usage information. The proving ground powertrain schedules were derived by analyzing the polled data from a pool of vehicles whose owners volunteered to participate in answering the survey questionnaires.

These voices of customers are translated to engineering requirements as part of the proving ground schedule. The way to create such a proving ground powertrain schedule is very subjective due to the subjective nature of the customer responses and the interpretation of the voices of customers to engineering criteria.

The other design and validation test target uses historical data collected from various sources and the acquired data from a test vehicle running on a proving ground based on the above mentioned proving ground schedules. However, historical data often lacks context and may not correlate with actual customer usage patterns, and the acquired proving ground data tends to be subjective and prone to error. As a result, the use of these data is limited because of a lack of confidence in its accuracy. When it is used, design targets are set to be artificially high in order to compensate for the low confidence rating.

Thus, there is a need to acquire the real-time customer usage data to substantiate and define the truly representative of 95th percentile customer usage in an objective manner. There is a need to improve the quality of data used to construct vehicle simulations and models for prototype durability testing. There is also a need to improve the efficiency of obtaining and processing component data so as to improve the design and testing stages of powertrain development.

The objective of the current work is to collect real-time vehicle drive torque data from customer vehicles, process the data efficiently, and utilize the data to set the proving ground schedules for actual requirements. The two main customer data sets that need to be collected and processed are the rainflow cycle count decomposition of the driveshaft torque loading cycles which stores essential fatigue information in a compact matrix form, and the revolutions at torque

histogram. Knowledge of prop shaft loading cycles and their statistical distribution among the customer population is useful for:

1. Understanding customer usage.
2. Driveline and axle design and sizing.
3. Fatigue and durability analysis.
4. Development and validation of proving grounds durability schedules.
5. Development and validation of dynamometer and bench-testing.

Commercial fatigue analysis software is available which incorporates efficient algorithms to count rainflow cycles, but the code is written for post-processing applications. In this approach, the two algorithms are implemented into the engine controller unit and executed in real-time.

ALGORITHM DESCRIPTION

The engine controller software calculates the real-time driveshaft torque time history. The driveshaft torque is calculated based on acquired data from the vehicle's throttle position, axle ratio, weight and calibrated powertrain performance curves. The time domain data required to implement this system is massive, and requires a large memory space in the engine controller memory map. This is one of the reasons why this approach was not adapted by other OEMs. Due to this constraint, a new method is required to present the data. The time domain data process can't be used, instead, the data is converted to the forms such as revolutions-at-torque histogram and a two dimensional rainflow cycle counting matrix.

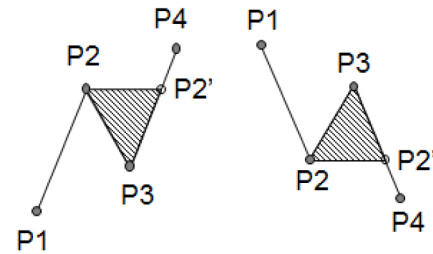
RAINFLOW CYCLE COUNTING

Rainflow cycle counting method is a process of extracting cycles from a complicated loading history, where each cycle is associated with a closed stress-strain hysteresis loop. The output of this cycle counting method can be represented by a compact matrix format that retains essential fatigue loading information.

The four-point rainflow cycle counting rule [4, 7, 8] is described below:

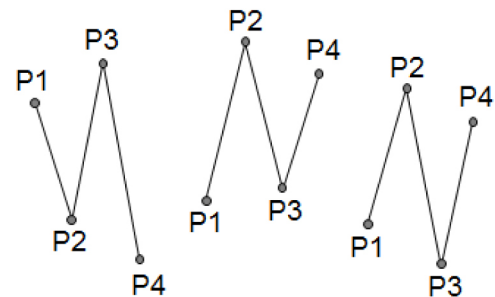
Consider four consecutive load points P1, P2, P3, and P4. If P2 and P3 are contained between P1 and P4, then a cycle is counted from P2 to P3 (and back to P2') as shown in Figure 1, else no cycle is counted as shown in Figure 2. The four-point rule is summarized by the following statement:

If $|P2-P1| \geq |P3-P2|$ and $|P4-P3| \geq |P3-P2|$, then FROM = P2 and TO = P3.



Cycle counted

Figure 1. Counted Cycles



No Cycle

Figure 2. Uncounted cycles

The steps in the algorithm are:

1. Initialize buffer.
2. Read next incoming point, x .
3. Check if x is a peak or valley point.
4. If x is not a peak or valley point, then go to 2, else
 - 4.1. Update buffer count, $k = k + 1$
 - 4.2. Temporarily store x in buffer, $y_k = x$.

5. If buffer has less than 4 points, then go to 2, else apply the four-point rule to the 4 most recent buffer points, $y_{k-3}, y_{k-2}, y_{k-1}, y_k$.

5.1. If the four-point rule passes, remove y_{k-2} and y_{k-1} from buffer, and update the from-to matrix. Update buffer count, $k = k - 2$. Go to 5.

5.2. Else the four-point rule fails, go to 2.

The buffer is a temporary storage for data points that have not been counted as cycles by the four-point rule. Only peak and valley points are stored in the buffer. This method minimizes the buffer size as well as the overall memory storage space. All buffer points should eventually be counted as cycles and removed from the buffer. The buffer must be initialized by storing the first incoming point and the next peak or valley point. This will establish the initial search