LabVIEW™ ポイントバイポイント VI 入門

目次

370434A-01

2002 年 01 月版

LabVIEW は、ポイントバイポイント解析を行う VI セットを提供しています。ポイントバイポイント解析は、リアルタイムのデータ集録に適しています。データ集録システムがリアルタイムで確定的な性能を必要とする場合、配列ベースの LabVIEW 解析 VI のポイントバイポイントバージョンを使用したプログラムを作成することができます。

リアルタイム性は、データ集録にとって理想です。LabVIEW のポイントバイポイント解析を使用して、データ解析においてもリアルタイムを利用することができます。バッファの準備、解析、および出力など、配列ベースの解析を段階的に行うと、高速で確定的なリアルタイムシステムには時間がかかり過ぎるおそれがあります。またリアルタイムシステムには、各データ点に対して 1 点ずつ連続して解析を行う必要がある場合もあります。

本書では、ポイントバイポイントデータ解析の概念およびプログラミングの詳細を説明します。LabVIEW のポイントバイポイント VI には、以下の利点があります。

- リアルタイムイベントのトラッキングとリアルタイムイベントへの応答が可能
- 解析プロセスを信号に直接つないで、速度を向上させ、データ損失を最小限に抑える
- 配列を割り当てる必要がなく、サンプリングレートの調整の必要も減るため、プログラミング作業が簡素化
- 単一の信号を即座に処理するので、解析が自動的にデータ集録と同期を取る

LabVIEW のポイントバイポイント VI ライブラリを使用する

LabVIEW のポイントバイポイント VI は、連続データ集録関連の、各配列ベースの解析 VI と対応しています。ただし、プログラミング上の相違を考慮する必要があります。通常、ポイントバイポイント VI を使用すると、プログラミングタスクの数が減少します。以下の表は、LabVIEW のポイントバイポイント VI の入出力の特徴について説明しています。

表 1 ポイントバイポイント VI の入出力の特徴

パラメータの種類	説明
入力データ	着信データ
出力データ	解析済みの送信データ
初期化	VI の内部状態をリセットするルーチン
サンプル長	データ集録システムまたは演算システムのデータの重要部分を最もよく表わす設定

ポイントバイポイント解析システムのサンプルについては、「ポイントバイポイント解析のケーススタディ」を参照してください。

ポイントバイポイント VI を初期化する

このセクションでは、多くのポイントバイポイント VI でどのような場合に、またどのようにしてポイントバイポイント**初期化**パラメータを使用するかを説明します。また、LabVIEW 6.0 以降のバージョンにある LabVIEW First Call? 関数についても説明します。

ポイントバイポイント VI での初期化の目的

初期化パラメータを使用すると、データフローや演算を妨げずに VI の内部状態をリセットすることができます。以下のようなイベントに応じて VI をリセットすることができます。

- ユーザがパラメータの値を変更
- アプリケーションが特定のイベントを生成するかしきい値に到達

たとえば、**関数→ポイントバイポイント→その他の関数 PtByPt** パレットにある Value has Changed PtByPt VI は、以下のようにイベントを変更することができます。

- 入力データを受信
- 変更を検出
- 別の VI で初期化をトリガするブールの TRUE 値を生成
- 処理のために入力データを別の VI に転送

図 1 は、Value has Changed PtByPt VI が別の VI での初期化をトリガし、その VI にデータを転送している場合を示しています。ここでは、入力データはパラメータ、つまりターゲット VI のための値です。

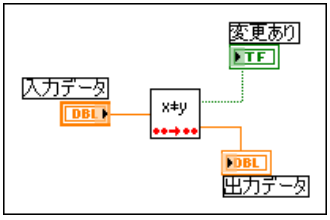


図 1　Value has Changed PtByPt VI の通常の役割

First Call? 関数を使用する

オペレータがアプリケーションを停止および起動する場合は常に初期化が自動的に行われるため、多くのポイントバイポイントアプリケーションでは**初期化**パラメータを使用する必要はありません。たとえば、アプリケーションの電源が 24 時間ごとに停止するとします。電源を停止すると、システムが適切に初期化されます。

ポイントバイポイント VI には、必要な場合 LabVIEW First Call? 関数が含まれています。First Call? 関数を含む VI では、最初に VI が呼び出されたときに VI の内部状態がリセットされます。VI の最初の呼び出しでは、First Call? 関数の**初期化**パラメータ値は常に TRUE です。それ以外の VI 実行期間は、パラメータ値は FALSE のままです。**関数→上級→同期**パレットにある First Call? 関数を使用して、ポイントバイポイント VI を作成します。図 2 は、LabVIEW While ループを使った First Call? 関数の使用例を示しています。

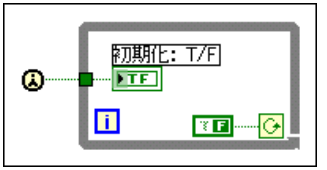


図 2　While ループ内で First Call? 関数を使用する

エラーのチェックおよび初期化

ポイントバイポイント VI は、作成したアプリケーションの構成における欠陥の識別に役立つエラーを生成します。標準の LabVIEW エラーコードに加えて、ポイントバイポイントのエラーコードがいくつかあります。

エラーコードは、通常、無効なパラメータおよび設定を識別します。より高いレベルのエラーチェックを行うために、データ集録または演算における不規則性を監視し、不規則性に対応するようにプログラムを構成します。たとえば、データの範囲検査をするときにある形式のエラーチェックを作成する場合などです。

ポイントバイポイント VI は、VI が最初に呼び出されたときに、またはアプリケーションの初期化後 1 回目に呼び出されたときにエラーコードを生成します。ポイントバイポイント VI は、エラーコードを一度だけ生成するため、リアルタイムの確定的なアプリケーションにおいて最適な動作が可能です。

ポイントバイポイント VI は、最初の呼び出しでエラーが検出された場合にエラーコードを生成して、無効なパラメータまたは設定をすべて知らせます。次の呼び出しでは、ポイントバイポイント VI はエラーコードをゼロに設定し、エラーコードを生成せずに実行を続けます。ユーザは、最初のエラーに対して以下のいずれかの動作をするようにアプリケーションをプログラミングすることができます。

- エラーをレポートして実行を続ける
- エラーをレポートして停止する
- エラーを無視して実行を続ける（デフォルト）

以下のプログラミングシーケンスでは、**エラー**パラメータを持つポイントバイポイント VI のエラーチェックメカニズムを作成するために Value has Changed PtByPt VI を使用する方法を説明します。

1. エラーを監視するパラメータを選択します。
2. パラメータ値を**入力データ**として Value has Changed PtByPt VI に配線します。
3. **出力データ**（Value has Changed PtByPt VI において常に**入力データ**と同じもの）をターゲット VI に転送します。
4. また、Value has Changed PtByPt VI は、入力パラメータ値が変更したときに TRUE 値を出力します。図 1 のように、TRUE イベントをターゲット VI に渡し、初期化をトリガします。
5. この初期化の直後の呼び出しでは、LabVIEW はエラーをチェックします。この初期化およびエラーチェックループは、入力パラメータが変更するたびに実行します。

FAQ (よくある質問)

このセクションでは、ポイントバイポイント解析について頻繁に寄せられる質問に対して解答します。

LabVIEW のポイントバイポイント解析と配列ベースの解析の相違点

表 2 および表 3 は、配列ベースの LabVIEW 解析とポイントバイポイント解析をさまざまな観点から比較しています。表 2 にある 2 つの自動車燃料配送システム（気化および燃料噴射）の相違を例として、配列ベースのデータ解析とポイントバイポイント解析の相違点を説明しています。

表 2 従来のパラダイムと新しいパラダイムの比較

従来のパラダイム	新しいパラダイム
自動車技術	
気化システム • フロート室内で燃料が蓄積される。 • エンジンの負圧によって吸い込まれた空気の流れによってキャブレター内部で負圧が発生し、燃料が吸い出される。 • 若干効率の良い燃焼が起こる。	**燃料噴射システム** • 燃料がガスタンクから連続流動。 • 燃焼の開始直後に、燃料が各燃焼室に直接噴射される。 • 即反応する精密な燃焼が起こる。

表 2 従来のパラダイムと新しいパラダイムの比較（続き）

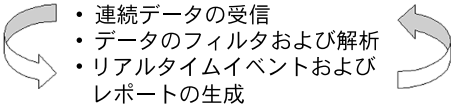
従来のパラダイム	新しいパラダイム
データ解析の技術	
配列ベースの解析 • データのバッファ構成単位の準備 • データ解析 • 解析データのバッファを生成 • レポートを生成	ポイントバイポイント解析  • 連続データの受信 • データのフィルタおよび解析 • リアルタイムイベントおよびレポートの生成

表 3 は、配列ベースの解析とポイントバイポイント解析のその他の比較を示しています。

表 3 配列ベースのデータ解析とポイントバイポイントデータ解析の比較

	配列ベースの解析	ポイントバイポイント VI を使用したデータ集録および解析
互換性	制限されたリアルタイムシステムとの互換性	リアルタイムシステムに互換：配列ベースのシステムと下位互換
データタイプ	配列	スカラ
割り込み	影響が大きい	影響が小さい
操作	ユーザによる観察：オフライン	ユーザによる制御：オンライン
性能およびプログラミング	複雑なステートマシンで起動データ損失（4～5 秒) を補正	起動データの損失なし。データ集録システムを一度初期化してから連続
アプローチ	プロセスの鏡映	自然で直接的なプロセスのフロー
プログラミング	バッファを指定	指定バッファなし
結果	レポートを出力	レポートおよびイベントをリアルタイムで出力
ランタイム動作	遅延プロセス	リアルタイム
ランタイム動作	停止	続行
ランタイム動作	待機	実行
動作スタイル	非同期	同期

ポイントバイポイント解析を使用する理由

ポイントバイポイント解析は、コンピュータベースのリアルタイムのデータ集録に適しています。配列ベースの解析では、入力、解析、出力のプロセスをより大きいデータセットのサブセットに実行します。ポイントバイポイント解析では、入力、解析、出力のプロセスを連続してリアルタイムで実行します。

ポイントバイポイント解析で新たに導入された概念

ポイントバイポイント解析を行う際は、まず以下の概念を理解してください。

- 初期化：以前のデータ解析セッションでの設定の介入を防止するため、ポイントバイポイント解析のアプリケーションを初期化する必要があります。
- 再入実行：ポイントバイポイント解析では LabVIEW の再入実行を有効にする必要があります。再入実行は、同じ解析関数を使用する 2 つのプロセスが互いに干渉しないことを保証する固定メモリを単一の解析プロセスに割り当てます。

メモ VI をカスタマイズして独自のポイントバイポイントアプリケーションで使用する場合は、再入実行を必ず有効にしてください。再入実行は、ほとんどのポイントバイポイント VI でデフォルトで有効になっています。

- 確定性：ポイントバイポイント解析は、リアルタイムのデータ信号のフローを効率よく統合するため、多くの確定的システムで使用されます。

ポイントバイポイント解析を使用しやすい理由

ほとんどの解析操作に対するアプローチが同じであるため、LabVIEW のポイントバイポイント解析は利用しやすくなっています。配列ベースのデータ解析でこうしたオペレーションを使用するのと同じような場合に、フィルタ、積分、平均値アルゴリズム等を使用してください。これとは対照的に、多項式関数におけるゼロの演算はポイントバイポイント解析には適さないため、こうした配列ベースの VI のポイントバイポイントバージョンは必要ありません。

データバッファなしで解析可能な理由

解析関数は、データセットの性質を表す解答を算出します。配列ベースのデータ集録および解析では、データを 10 個の小さいバッファに分けることにより大きなデータセットを解析する場合もあります。この 10 個のデータセットを解析すると、10 個の解答が得られます。さらに、この 10 個の解答から、全データセットの性質を表す 1 つの解答を出すことができます。

ポイントバイポイント解析では、全データセットをリアルタイムで解析します。特定の長さのサンプルユニットがバッファを置き換えます。ポイントバイポイントサンプルユニットは、解析しているデータセットの中の有効なイベントの長さと一致する長さにすることができます。欠陥のある車輪を検出するために 1 秒に数千個のサンプルを集録する例については、「ポイントバイポイント解析のケーススタディ」のセクションを参照してください。この場合、信号は時速 60 ～ 70 km で運行している列車から

送信されます。このアプリケーションのサンプルの長さは、車輪間の最低距離に対応しています。

通常のポイントバイポイントアプリケーションは、長い一連のサンプルユニットを解析します。しかし、ユーザが関心を持つのは大抵、こうしたサンプルユニットの中のいくつかのみです。そうしたサンプルを識別するため、アプリケーションは**関連信号の終点**などの遷移に焦点をおきます。

「ポイントバイポイント解析のケーススタディ」セクションのサンプル、列車車輪の検出アプリケーションでは、対象となるサンプルを識別するのに、**信号の終点**を使用します。アプリケーションがこの遷移点を識別する瞬間、現在のサンプルユニットの振幅の最大値を読み取ります。特にこの振幅の値は、信号が停止した列車車輪の完了信号に対応します。このリアルタイム振幅値は、その車輪と列車に関するイベントまたはレポートを生成するために使用できます。

リアルタイムアプリケーションでポイントバイポイント解析が効果的な理由

概して、連続した高速なデータフローをプロセスする必要がある場合は、ポイントバイポイント解析で対応できます。たとえば、工業オートメーションの場合、制御データフローは連続的で、コンピュータはさまざまな解析関数や転送関数を使用して実際のプロセスを制御します。こうした工学的タスクでは、ポイントバイポイント解析はリアルタイムで行われます。

リアルタイムアプリケーションの中には、高速データ集録および解析の代わりに簡潔で信頼性のあるプログラムを必要とするものもあります。ポイントバイポイント解析は、ユーザが配列を割り当てる必要がなく、データ解析フローも自然で連続しているため、簡潔で信頼性があります。

ポイントバイポイント解析の必要性

高速で確定的なポイントバイポイントデータ集録なしでプロセスの制御が可能な限りにおいては、ポイントバイポイント解析を必ずしも使用する必要はありません。ただし、リアルタイムデータアプリケーション用にリソースをとる場合は、アプリケーションを最大限に活用するためにポイントバイポイント解析を使用してください。サンプル／秒のレートを 10 の指数的に増加するごとに、ポイントバイポイント解析の必要性が増加します。

ポイントバイポイントアプローチにより、アプリケーションのフローが監視および制御する実際のプロセスの自然なフローに合うようになるため、設計、実装、テストプロセスが簡素化されます。

ポイントバイポイント解析の長期的重要性

リアルタイムでのデータアプリケーションおよびデータ解析は、合理的で安定したアプリケーションを必要とします。ポイントバイポイント解析は、集録および解析のプロセスに直接連結するため、より合理的で安定し

ています。合理的で安定したポイントバイポイント解析により、集録および解析のプロセスは、FPGA (フィールドプログラマブルゲートアレイ) チップ、DSP チップ、組み込み式コントローラ、専用の CPU、そして ASIC における制御点に近づきます。

例

リアルタイム解析および制御は、ポイントバイポイントパラダイムの利点のひとつです。以下の例は、ポイントバイポイント VI および関数を使用することの利点を示しています。

ポイントバイポイントおよび配列ベースのフィルタの例

図 3 に示すように、PtbyPt and Array Based Filter VI は、ノイズを含んだ信号を生成します。VI は、信号をフィルタ処理してノイズを取り除き、実際の信号をより良く表現します。フィルタは、信号中の不要な周波数をブロックし、重要な周波数を通します。この VI は、ノイズを信号から取り除くフィルタ処理に、2 つの方法を採用しています。1 つめの方法では、ポイントバイポイントバージョンのフィルタを使用してノイズを取り除きます。2 つめの方法では、フィルタの配列ベースのバージョンを使用します。

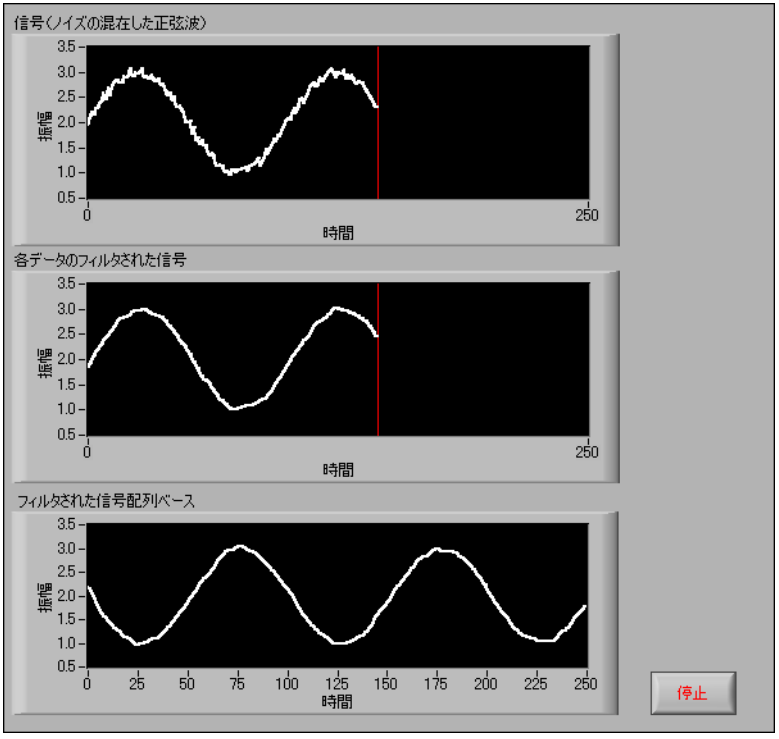


図 3 ポイントバイポイントと配列ベースのフィルタの対比

ポイントバイポイントをベースとしたフィルタ

ポイントバイポイントの LabVIEW のプログラミングでは、VI は単一データ点を集録、解析し、レポート作成へと送ります。同時に、VI は別の単一データ点を集録して 1 点ずつプロセスを繰り返します。ポイントバイポイントフィルタでは、VI は入力信号の各点を解析してリアルタイムで 1 つずつレポートを生成します。

配列ベースのフィルタ

配列ベースの LabVIEW のプログラミングでは、VI はデータセットを集録してレポートを生成します。配列ベースのフィルタでは、VI は一定の長さの信号を集録して解析します。解析後、フィルタは不要な周波数をデータセットから取り除き、VI はグラフ化されたデータセットのレポートを表示します。グラフは、フィルタ処理された信号全体を示しています。VI は、また一定の長さの信号が集録されるまで待機し、プロセスを続行します。

リアルタイム振幅スペクトルの例

図 4 のように、Realtime Amplitude Spectrum VI は、サンプル信号を生成してランダムノイズを加えます。VI はその信号を解析し、パワースペクトルをグラフに表示します。

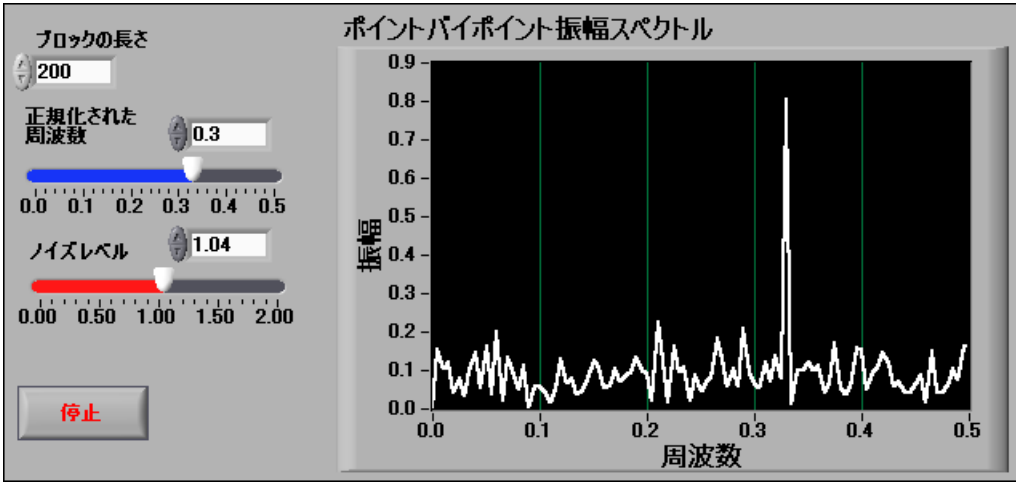


図 4　リアルタイム制御器を使用したパワースペクトル解析

この VI は、ポイントバイポイント VI を使用するため、周波数、ノイズレベル、およびサンプル数をリアルタイムで変更することができます。こうした変更は、パワースペクトルにリアルタイムで影響します。ポイントバイポイントパラダイムは、リアルタイムレベルの詳細を要するデータ集録および解析において非常に役立ちます。

移動ヒストグラム PtByPt

図 5 のように、Moving Histogram PtByPt VI は、信号を生成してそれを解析し、ヒストグラムを作成します。リアルタイムでサンプル間の間隔とサンプルの長さを調節することができます。

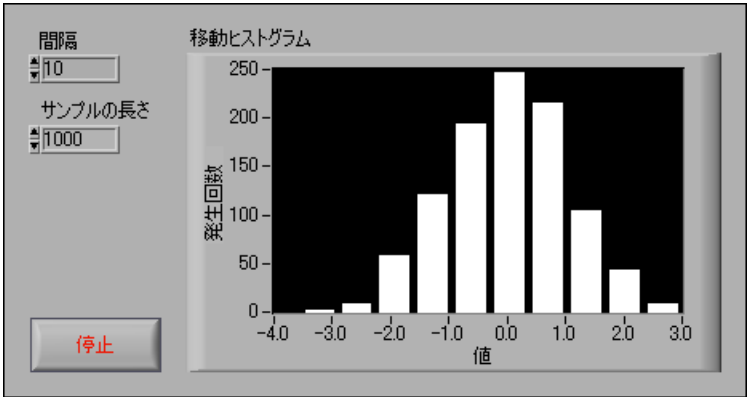


図 5　ポイントバイポイント VI を使用して作成した移動ヒストグラム

LabVIEW の配列ベースのパラダイムで移動ヒストグラムを作成することは可能ですが、より困難です。ポイントバイポイントパラダイムにより、リアルタイム動作の VI を簡単に作成することができます。

ポイントバイポイント解析のケーススタディ

このケーススタディでは、LabVIEW で作成したポイントバイポイント解析のアプリケーションの例を示します。列車車輪の欠陥を検出するリアルタイムデータ集録アプリケーションは、ポイントバイポイントデータ解析の簡潔性および柔軟性を示しています。Train Wheel PtByPt VI は、ポイントバイポイント VI を使用しています。

列車車輪のポイントバイポイント解析

この例では、操車場の整備員は、列車車輪の欠陥を検出しなければなりません。鉄道作業員は欠陥のある車輪をハンマーで打ち共鳴音の違いで欠陥を識別します。テストを手動で行うと非常に時間がかかる上、エラーの発生する可能性が高く、また、大ざっぱ過ぎて小さな欠陥を検出することができないため、手動テストに代わって自動検査の採用が必要となります。自動化されたソリューションでは、汽車の車輪を停止せずに使用中のままテストでき、ダイナミックテストのパワーも加わります。

欠陥のある可能性がある列車車輪を検出するソリューションでは、微妙な欠陥も迅速に検出する必要があります。このアプリケーションは、列車の通常運行中にデータを収集します。このアプリケーションにより、プログラミングを簡素化し、テスト結果の出力速度および正確性の向上が可能なため、リアルタイムでデータ収集および解析を行うことができます。

Train Wheel PtByPt VI は、LabVIEW のポイントバイポイント解析機能を使用して、欠陥のある列車車輪を検出するためのソリューションを構築します。

図 6 および図 7 は、Train Wheel PtByPt VI のフロントパネルおよびブロックダイアグラムを示しています。このマニュアルで使用されている Train Wheel PtByPt VI およびポイントバイポイントプログラミングのその他のサンプル VI は、National Instruments Developer Zone、`ni.com/zone` からダウンロードすることができます。

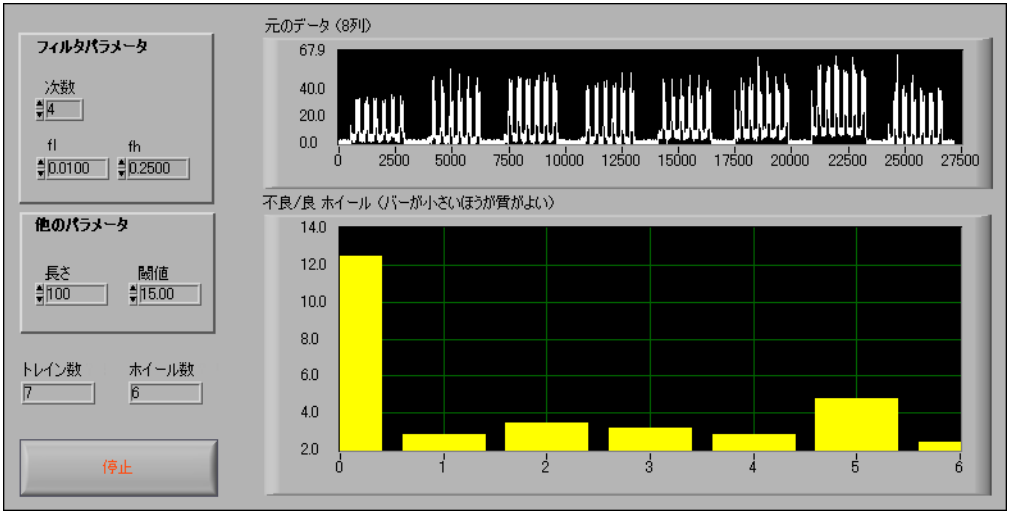


図 6 Train Wheel PtByPt VI のフロントパネル

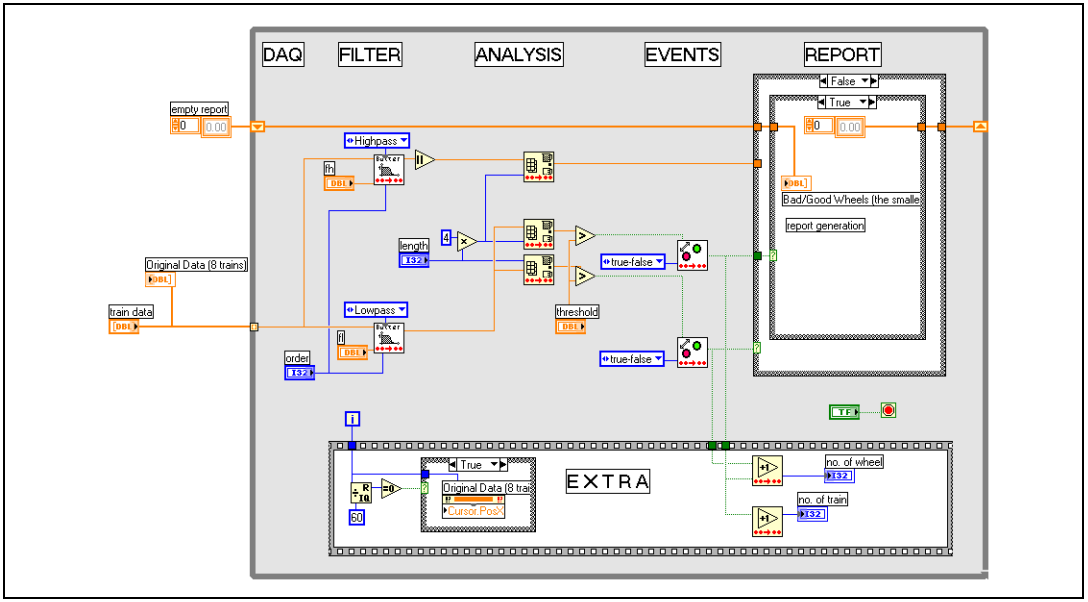


図 7 Train Wheel PtByPt VI

メモ この例では、LabVIEW のポイントバイポイント解析プログラムを実装することに焦点を当てています。理想的なサンプリングの間隔および信号調節へのアプローチについては、この例では扱いません。

LabVIEW ポイントバイポイントソリューションの概要

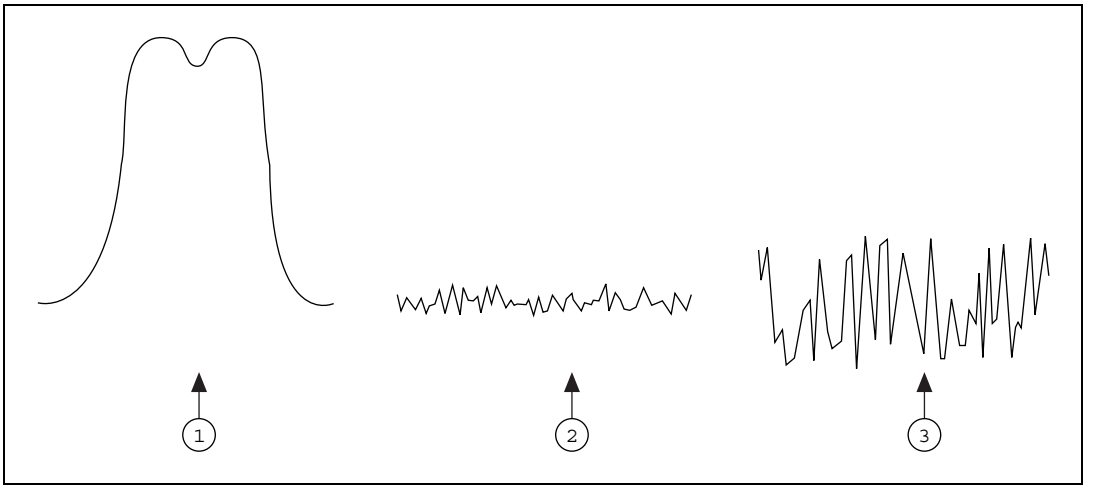
列車車輪検出アプリケーションは While ループから連続してデータを集録します。以下に、ループ内で起きるプロセスを説明します。

1. DAQ：波形データのフロー
2. フィルタ処理：低周波数成分と高周波数成分の分離
3. 解析：列車、車輪、各車輪の波形のエネルギーレベルの検出
4. イベント：車輪および列車車輪の信号遷移への応答
5. レポート：列車、車輪、および車輪に欠陥のある可能性がある列車のロギング

検出アプリケーションには、Case ストラクチャ、While ループ、数値制御器、数値演算子などの標準的な LabVIEW のプログラミングオブジェクトが必要です。Train Wheel PtByPt VI プログラムには、以下のセクションに挙げる VI も必要です。

列車車輪波形の特徴

列車車輪から発生する波形の特徴により、その波形信号を一点ずつどのようにフィルタ処理および解析するかが決定されます。動作中の列車車輪は、低周波数成分および高周波数成分を含む信号を発生します。鉄道路線にひずみゲージを取り付けると、正規分布曲線に近い、ノイズを含む信号が検出されます。図は、この曲線の低周波数成分および高周波数成分を示しています。



1 一般的な列車車輪信号のローパス成分	2 一般的な列車車輪信号のハイパス成分	3 欠陥のある列車車輪信号のハイパス成分

図 8　列車車輪の信号の低周波数成分および高周波数成分

列車車輪動作の低周波数成分は、運行中の通常のノイズを表しています。欠陥車輪および正常車輪は、信号の同じ低周波数成分を生成します。曲線のピークは、車輪がひずみゲージの真上を動いている瞬間を表しています。正規分布曲線の最低点はそれぞれ、車輪がひずみゲージの上を通過する際の車輪の始点と終点を表しています。

列車車輪の信号は、車輪の品質を反映する高周波数成分も含んでいます。運行中、欠陥のある車輪は、より高いエネルギーを発生します。つまり、欠陥のある車輪の高周波数成分の振幅のほうが大きいということになります。

Train Wheel PtByPt VI の波形のパラメータ

欠陥車輪を含むすべての列車車輪の波形は、予測域に入ります。この予測可能な動作により、表 4 に示すパラメータを選ぶことができます。こうしたパラメータは、「LabVIEW ポイントバイポイントソリューションの概要」セクションで説明されている 5 つの段階に該当します。

メモ 表にあるパラメータは、ポイントバイポイントアプローチを使用したプログラミングの詳細を理解するのに役立ちます。ただし、ポイントバイポイント集録および解析を理解するには、この表のパラメータを参照する必要はありません。Train Wheel PtByPt VI を使用する場合は常に、パラメータを調節するようにしてください。これは、各データ集録システムにより特徴が異なるためです。

表 4 Train Wheel PtByPt VI の列車の車輪の波形のパラメータとして使用できるもの

パラメータの種類	パラメータ名	段階および影響を受ける VI	Train Wheel PtByPt での目的
データ入力	**入力データ**	データ集録 /DAQ In.VI	データソース、デバイス、チャネル、毎秒ごとのサンプル数、およびスケーリング
フィルタの分解能	**次数**	フィルタ処理 / Butterworth Filter PtByPt	VI が一定時間内にフィルタ処理をする波形データ量。Train Wheel PtByPt では、2 を指定できます。 **次数**は、Train Wheel PtByPt での両方のバタワースフィルタに該当します。
低域遮断周波数	**fl**	フィルタ処理 / Butterworth Filter PtByPt	列車車輪のひずみゲージからの離脱を識別する最低限の信号強度。Train Wheel PtByPt では、0.01 を指定できます。
高域遮断周波数	**fh**	フィルタ処理 / Butterworth Filter PtByPt	高域周波数の波形情報の終了を識別する最低限の信号強度。Train Wheel PtByPt では、0.25 を指定できます。
解析する波形の長さ	**長さ**	解析 / Array Max & Min PtByPt	Train Wheel PtByPt が解析する波形部分のサイズ。サンプルの理想的な長さを計算するには、列車速度、最小車輪間隔、および 1 秒間に受け取るサンプル数を考慮します。Train Wheel PtByPt では、100 を指定できます。 3 つの Array Max & Min PtByPt VI の値をすべて計算する際に、Train Wheel PtByPt VI は、**長さ**を使用します。
初期化	**初期化**	すべてのポイントバイポイント VI	連続データ集録の新しいセッションのために VI をリセットするルーチン。

表 4 Train Wheel PtByPt VI の列車の車輪の波形のパラメータとして使用できるもの（続き）

パラメータの種類	パラメータ名	段階および影響を受ける VI	Train Wheel PtByPt での目的
（乗数）	（なし）	解析 / Array Max & Min PtByPt	より長い波形部分を解析するように設定します。この長い部分が列車車輪の信号動作を表示するのに失敗した場合、VI は列車の終点を識別します。Train Wheel PtByPt では、4 を指定できます。
（しきい値）	（なし）	解析 / Array Max & Min PtByPt	集録している信号内に列車車輪の信号が存在しない場合を識別するための、比較点を提供します。Train Wheel PtByPt では、3 を指定できます。

Train Wheel PtByPt VI でのサンプリングレート

ポイントバイポイント検出アプリケーションは、運行している列車車輪からの一連の連続波形データを操作します。時速 60 ～ 70 km で走行している列車から、欠陥のある車輪を検出するのに十分な情報を得るには、1 秒に数百個から数千個のサンプルが適当です。

Train Wheel PtByPt VI のフィルタ処理要件

ポイントバイポイント検出アプリケーションは、列車車輪波形の低周波数成分および高周波数成分をフィルタ処理する必要があります。2 つのポイントバイポイントバタワースフィルタは、以下のタスクを行います。

- 波形の低周波数成分の抽出
- 波形の高周波数成分の抽出

Train Wheel PtByPt VI の解析

ポイントバイポイント検出のアプリケーションは、低周波数成分および高周波数成分を別々に解析する必要があります。この場合、Array Max & Min VI のポイントバイポイントバージョンは、各車輪、列車終端部、車輪の終端部の波形におけるエネルギーレベルを表す波形データを抽出します。

それぞれ 3 つのポイントバイポイント Array Max & Min VI は、以下の個別タスクを行います。

- 各車輪の最高周波数値の識別
- 各列車終端部の識別
- 各車輪終端部の識別

メモ Array Max & Min PtByPt VI という VI 名には、単に配列ベースのこの VI の名前と一致させるために Array という言葉が付いています。この VI の PtByPt バージョンに配列を割り当てる必要はありません。

解析段階で最大値および最小値を識別した後、別のイベント段階で値がしきい値設定に達するのを検出します。

Train Wheel PtByPt VI のイベント

ポイントバイポイント検出のアプリケーションは、検出する車輪および列車をすべて記録します。Boolean Crossing VI は、Array Max & Min PtByPt VI が列車車輪の終端部を検出するたびにイベントを生成します。データ集録システムも、Array Max & Min PtByPt VI が列車の終端部を検出するたびにイベントを生成します。高周波数信号の解析により、どの車輪に欠陥のある可能性があるかを識別します。VI が欠陥のある可能性がある車輪を見つけた場合、その VI はレポートが車輪の終端部イベントを受け取った際にイベントを直接レポートに渡します。

2 つのポイントバイポイント Boolean Crossing VI は、以下のタスクを行います。

- 車輪終端部を示す信号における遷移点の検出
- 列車終端部を示す信号における遷移点の検出

Boolean Crossing PtByPt VI は、遷移に対応します。単一車輪波形の振幅がある一定のレベルを下回れば、車輪終端部はひずみゲージに到達しています。このデータ集録システムでは、車輪終端部を識別するには、3 が適切なしきい値レベルです。信号強度がこのレベルを下回ると、Boolean Crossing VI は遷移イベントを認識し、そのイベントをレポートへと渡します。

Train Wheel PtByPt VI のレポート

ポイントバイポイント検出アプリケーションは、データ集録システムを通して通過する全列車の全車輪についてレポートを生成します。このシステムは、欠陥がある可能性のある車輪すべてについてレポートを生成します。

車輪がひずみゲージを通過するたびに、アプリケーションはその波形を捕らえて解析し、イベントのレポートを生成します。表 5 は、単一列車車輪についてのレポートの構成要素を説明しています。

表 5 列車の単一車輪についてのレポートの例

情報源	結果の意味
波形イベントのカウンタメカニズム	段階 1：車輪 No. 4 は、ひずみゲージを通過しました。
ハイパスフィルタデータの解析	段階 2：車輪 No. 4 は、ひずみゲージを通過しました。車輪に欠陥のある可能性があります。
列車終端部のイベントのためのカウンタのメカニズム	段階 3：列車 No. 8 の車輪 No. 4 は、ひずみゲージを通過しました。車輪に欠陥のある可能性があります。

Train Wheel PtByPt VI がポイントバイポイント解析を使用するのは工業プロセスを制御するためではなく、レポートを生成するためです。ただし、このデータ集録システムは、リアルタイムでデータを集録するので、欠陥のある可能性がある車輪を見つけたときに列車を停止するなど、リアルタイム制御応答を生成するようにこのアプリケーションを変更することができます。

まとめ

リアルタイムでデータ集録を行う場合、ポイントバイポイント解析は、リアルタイムでデータを解析するのに役立ちます。ポイントバイポイント解析は、連続的および瞬間的に行われます。データ集録中、そのフィルタ処理および解析を 1 点ずつ行い、必要な情報を抽出し、適切な応答をします。このケーススタディは、イベントおよびレポートの両方をリアルタイムで生成するためのポイントバイポイントアプローチの効果を示しています。