APPROACHES TO INCORPORATE LARGE DATA SERIES INTO **ADAMS** MODELS

Renguang Dong
Centre for Surface Transportation Technology
National Research Council Canada
Ottawa, Canada

Abstract
Modellers are often required to use measured data such as road or track profiles, accelerations, and dynamic forces as inputs for the simulation of road or railway vehicles. Each series of data often has more than 10,000 points. Such a large number of points cannot be directly taken into models built in ADAMS because the maximum number of points for any data element is limited to 1,250. To overcome this limitation, several possible approaches have been studied and tested. Two of them have been successfully applied to several projects. They are summarized in this paper.

1. Introduction

To simulate vehicle responses, modellers often need to use measured displacements, accelerations, or dynamic forces as excitation inputs into a model. In the current version of ADAMS (8.2), measured data such as displacements may be incorporated into the model using the following procedures:

Step 1. Read the data into the ADAMS model
MAINMENU ⇒ POSTPROCESSING ⇒ NUMERIC_RESULTS
⇒ READ_FROM_FILE

Step 2. Create a data element such as a spline using the data file in ADAMS model
MAINMENU ⇒ PREPROCESSING ⇒ DATA_ELEMENT ⇒ CREATE ⇒ SPLINE

Step 3. Use the data element to create the required input function (such as a motion generator)
MAINMENU ⇒ PREPROCESSING ⇒ CONSTRAINT ⇒ CREATE
⇒ MOTION_GENERATOR

Accelerations and forces can be input in a similar manner. Any intermediate point between any two points can be automatically calculated as a function of time or distance travelled, using the "AKISPL", "CURVE" or "CUBSPL" interpolator built into ADAMS. The spline can usually represent the original data very well.

However, the number of data points in any ADAMS data element is limited to 1,250. This becomes a problem in simulations we have done of road and railway vehicles, where the number

of input points can be greater than 1,250, 100,000 points in some cases. One could calculate the system responses piece by piece using the approach mentioned. However, this would be time consuming, especially if a large number of simulations are required, as occurs in many projects. In addition, the transition process at the broken points may introduce some unexpected components in the responses, and it is not convenient to use the calculated results to compare with experimental data. To resolve this problem, several possible approaches have been studied and tested. Two of them have been successfully applied to our simulations. They are general approaches and can be applied to any other case where a large number of data points are required as inputs to an ADAMS model. The approaches are described below.

2. Approach I

The first approach may be called the "cutting and merging" method. The basic idea is shown in Figure 1. First, the original data series is divided into several pieces, with each piece having fewer than 1,250 points. Thus, each piece can be input into the ADAMS model, and represented by a spline, using Step 1 and Step 2 described above. A short overlap of the data points between two neighboring pieces should be included to ensure a good spline shape and continuity at the broken point.

Once all the required splines are ready, there are two methods to link the splines to form a continuous function that represents the original data series. The first method is to use the arithmetic "IF" built into ADAMS. As noted in ADAMS/SOLVER REFERENCE MANUAL, when using an arithmetic "IF" it is important that the resulting function be continuous. Since an overlap is considered, the values of some points at the vicinity of the broken points are exactly the same in the two neighboring sections. If a local interpolation method is used, the function should not be discontinuous. However, if a global interpolation method is used, continuity at switch points may not be guaranteed. As recommended in ADAMS/SOLVER - AN OVERVIEW OF HOW TO USE ADAMS/SOLVER, one may use equivalent "STEPs" built into ADAMS to replace the "IF" FUNCTIONs. This results in the second method for the connections of the splines.

By utilizing the "STEPs", the function expression that represents a given data series may be written as

$$Function = \sum_{i=1}^{n} SPLINE_i \times STEP_i$$

where n is the total number of splines to be required, *SPLINE* is the spline created in ADAMS model and *STEP* is the step function built into ADAMS.

The step function(s) for each spline is(are) designed so that the requested portion of the spline is kept intact and the other part is totally suppressed after the operation in the above equation. The transition period from "1" to "0" in the step function should be short and it should be the same for any two neighboring splines. Several samples of the step functions are shown in Figure 1. Only one step function is required for the first and last splines but two step functions are necessary to those between. With the total number of points less than 1,250, a few extra points

should be added to each section of data, which form the added portion(s) of the splines, as shown in the Figure 1. This is to make the spline cover the full range of the distance or time in order to ensure a successful interpolation. The added portions do not have to be the same as shown in the Figure. Any limited real value is acceptable because they will actually be eliminated in the computing operations.

As an example, a model of a truck is shown in Figure 2. To simulate the road input, an actuator was used and connected to a wheel by a force or a spring-damper system. The road input was represented by the vertical displacement of the actuator, which was assigned to the actuator by the MOTION_GENERATOR in CONSTRAINT. Using the parameters shown in Figures 1 and 2, the function for the motion at Wheel 1 was expressed as:

```
Function =  STEP(TIME*85/3.6, 310.0, 1.0, 310.0001, 0.0)
            *CUBSPL(TIME*85/3.6, 0,.truck7.SPL1)
            +STEP(TIME*85/3.6, 310.0, 0.0, 310.0001, 1.0)
            *CUBSPL(TIME*85/3.6, 0,.truck7.SPL2)
            *STEP(TIME*85/3.6, 617.5, 1.0, 617.5001, 0.0)
            + .....
```

where 85/3.6 is the vehicle travel speed in meters per second, 310 and 617.5 are the distance for the first and second broken points, respectively.

At Wheel 2, it may be expressed as:

```
Function =  STEP(TIME*85/3.6-3.97, 310.0, 1.0, 310.0001, 0.0)
            *CUBSPL(TIME*85/3.6-3.97, 0,.truck7.SPL1)
            +STEP(TIME*85/3.6-3.97, 310.0, 0.0, 310.0001, 1.0)
            *CUBSPL(TIME*85/3.6-3.97, 0,.truck7.SPL2)
            *STEP(TIME*85/3.6-3.97, 617.5, 1.0, 617.5001, 0.0)
            + .....
```

where 3.97 is the distance between the first wheel and the second wheel. The function at Wheel 3 was made by simply replacing 3.97 by (3.97+10.35) in the above expression.

Figure 3 shows the comparison of the original data of a road profile with the displacement output from an actuator. The two lines coincide almost perfectly. This demonstrates that the "cutting and merging" approach described above works very well.

This approach takes advantage of the data element such as the spline provided in ADAMS, which can usually give a very good interpolation as well good the first and second derivatives if the global interpolators are used. It is also convenient if the total number of the data points is only a few multiples of 1,250. However, if the data size is larger than 10 multiples of 1,250, the function expression may become very lengthy and the preparation for the splines will then be highly time consuming. In such a case, another approach may be used, which is presented in the next section.

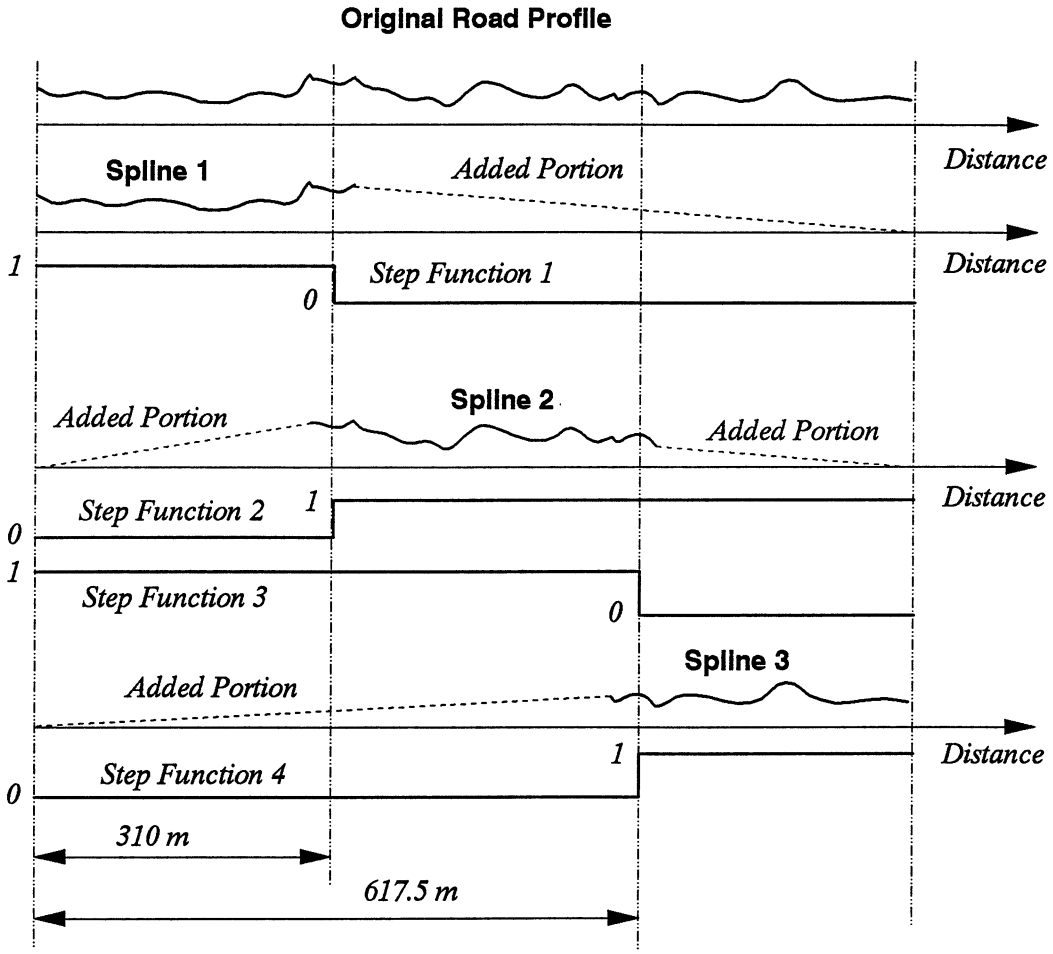

Figure 1. Breaking down the profile

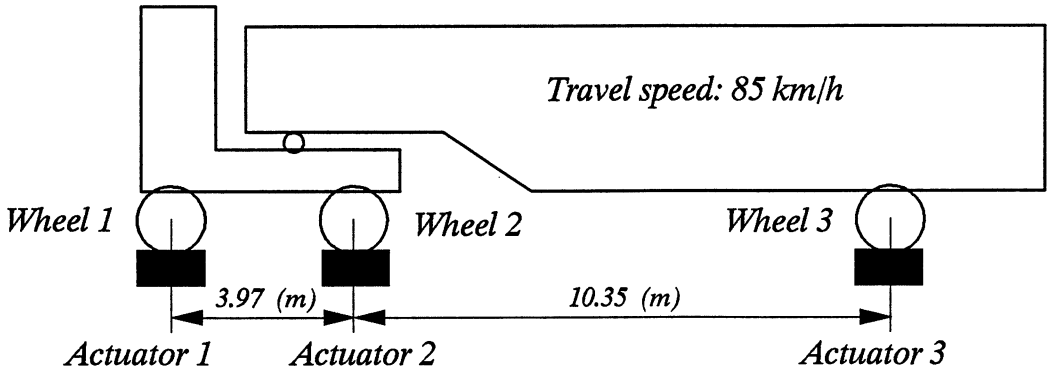

Figure 2. A truck model

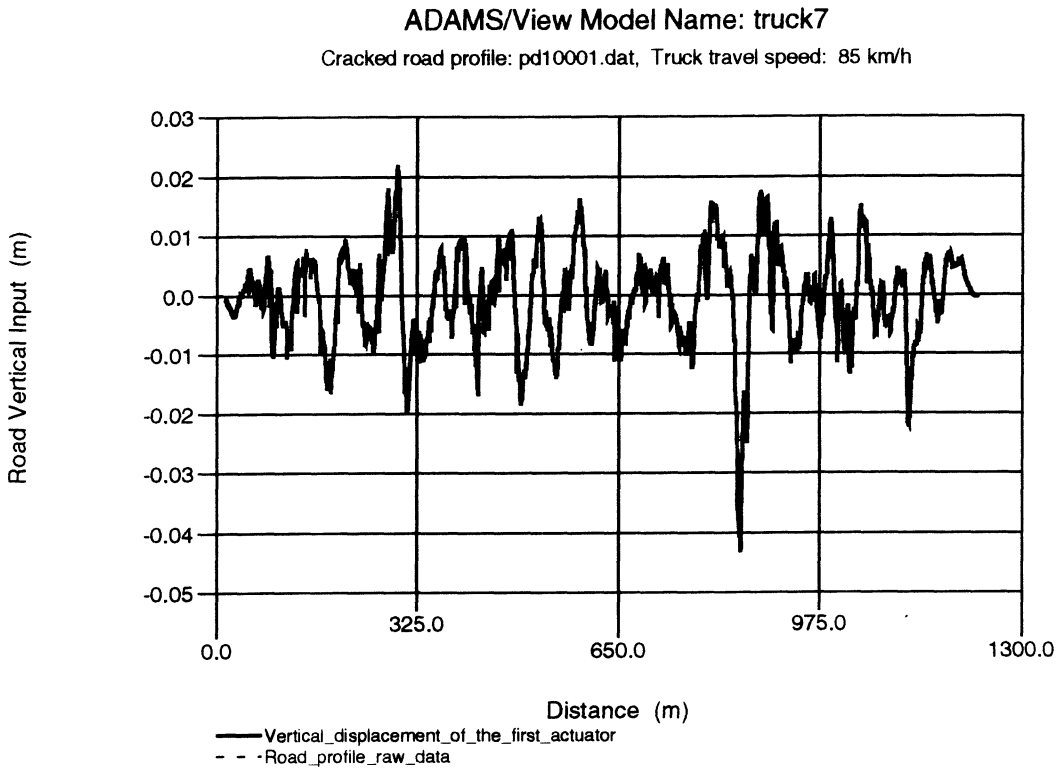

Figure 3. Comparing original data of road profile with displacement output from ADAMS model

3. Approach II

It is possible to access the data element such as a spline created in the model from a user_written subroutine. This provides us with an alternative approach to that described above. However, this approach also requires a considerable amount of data preparation and occupies many memories if the data series is very large.

A simple approach to incorporating a large data series into a model is to open a file and defined an array as a local variable in the subroutine. The data are read in and stored in the array at the beginning of the computation. The current value is calculated in the subroutine using a local interpolation technique such as parabolic interpolation. The simulation time corresponding to the last successful simulation is used to determine the locations of the base points.

If memory is a problem, the data may be read in once per piece and kept updated at certain given points of time or distance. We have successfully input the data once per point. The simulation time corresponding to the last successful simulation step was used to determine when the file should be opened and the next data be read in. In the case of the parabolic interpolation, only three base points are required and it is not necessary to define the array mentioned. These points were stored in the array of passed statement parameters (PAR), which has the maximum size of

30. This is usually sufficient to accommodate this number of data elements. However, the computation may not be efficient because the computer has to visit the hard disk constantly to read the input file. It is better to read in the data at a relatively long interval if there is sufficient memory to permit it.

This "subroutine" approach has been applied to a railway vehicle simulation. Dynamic forces measured from instrumented wheelsets were used as inputs to the vehicle model to calculate the accelerations of wheelset axle bearings. The parabolic interpolation was used in this case. A portion of original data for a vertical dynamic force and the corresponding force output from the ADAMS model are plotted in Figure 4. As shown, the two lines match very well, except that there are some minor difference at very few locations.

Because a local interpolation technique is used in this approach, the first and second derivatives may not be accurate and continuous, but there is minimal effort in preparing the data file, regardless of how large the data file would be.

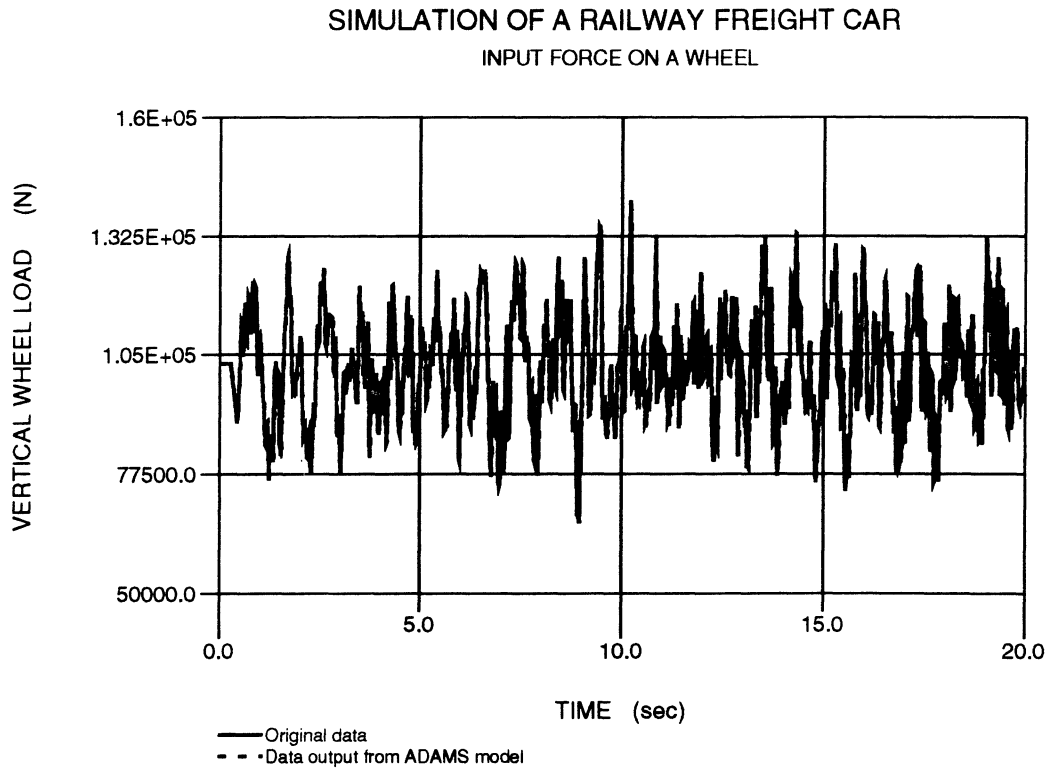

Figure 4. Comparing original data of dynamic force with the force output from ADAMS model

4. Summary

Even though a large data series cannot be directly incorporated into an ADAMS model because of a limitation of data elements, it is not difficult to overcome this limitation using the tools already provided in ADAMS.

Two basic approaches for incorporating large data series into ADAMS models are proposed and have been tested. The results demonstrate that they are valid and reliable in the cases we have used, namely to several models of road and railway vehicles. We may also generalize to say that they can be used to input any displacement, acceleration or dynamic forces.

The first approach, the "cutting and merging" method, is efficient and convenient if the data size is only within a few multiples of 1,250. The second approach, or the subroutine method, is recommended if the size of the data file is much larger than 1,250.