Optimal Model-Based Decomposition of Powertrain System Design

Optimal design of large engineering systems modeled as nonlinear programming problems remains a challenge because increased size reduces reliability and speed of numerical optimization algorithms. Decomposition of the original model into smaller coordinated submodels is desirable or even necessary. The article presents a methodology for optimal model-based decomposition of design problems, whether or not initially cast as optimization models. The overall model is represented by a hypergraph that is optimally partitioned into weakly-connected subgraphs satisfying partitioning constraints. The formulation is robust enough to account for computational demands and resources, and the strength of interdependencies between the design relations contained in the model. This decomposition methodology is applied to a vehicle powertrain system design model consisting of engine, torque converter, transmission, and wheel-tire assemblies, with 87 design relations and 119 design and state/behavior variables.

Introduction

Optimization methods have been applied with practical success to automotive vehicle powertrain components using well-developed and calibrated simulations and have now become common product development tools. Difficulties arise when we start considering design of the powertrain system. In general, the size of the problem becomes too large (more than one hundred variables) to expect reliable results from numerical optimization algorithms given known model nonlinearities. Different computational modules (simulations) coupled through design and state/behavior variables must be used. Even when numerical results are successfully obtained, one may not be able to readily interpret the engineering trade-offs implied. These observations, which are not peculiar to only automotive systems, have motivated the present work.

Breaking up the optimization model into smaller submodels must employ rigorous decomposition and coordination strategies. Besides improved coordination and communication, decomposition allows for conceptual simplification of the design, parallel computation, modularity, simpler and more efficient computational procedures, and different solution techniques for individual subsystems.

Decomposition strategies in the design and optimization literature are commonly classified as object (by physical components), aspect (by knowledge domains), and sequential (by directed flow of elements or information) decomposition. Object and aspect decomposition assume a "natural" decomposition of the problem. However, drawing "boundaries" around physical components and subassemblies is very subjective, while division by specialties (knowledge domains) may be dictated by management considerations that fail to account for disciplinary coupling. Sequential decomposition presumes unidirectionality of design information flow that contradicts the cooperative behavior desirable in concurrent engineering. Finally, computational resources often dictate in practice design strategies for large systems whose simulations may require days of computation on workstations or use of massively parallel machines—a requirement difficult to address in the decompositional strategies above.

Optimal model-based decomposition (OMBD) aims at the solution of large-scale design problems (with 100 or more variables and constraints). Model-based decomposition allows identification of weakly-connected structures implicit in the mathematical design model that can satisfy high-level design requirements such as concurrency, modularity, and robustness, as well as availability of computational resources. A functional representation of all design relations is required although the precise form of the functions is not needed for decomposition. The term design relation is very general and may correspond to either an algebraic or differential equation, a discretized continuum equation, a response surface, or a simulation used to evaluate state/behavior variables or constraint functions. A black-box simulation may be treated as a single design relation or as a collection of relations, each one corresponding to an output from the simulation. The design model is then represented by a hypergraph and the optimal model-based decomposition problem is formulated as a hypergraph partitioning problem. The representation and formulation are robust enough to account for computational demands of the modules in the system model, the strength of their interdependencies, and the available computational resources, so constraints on the decomposition problem itself can be imposed. This approach makes use of recent advances common to such diverse areas as graph theory, VLSI design, computational mechanics, and parallel computing that are very relevant to the solution of the OMBD problem and of the attendant large scale optimal design one.

How to solve already partitioned problems for increased computational efficiency and robustness has been extensively studied in operations research, but not how to partition a model in the first place. A recent review on the former topic can be found in Wagner (1993) and Wagner and Papalambros (1993a). Design researchers have studied decomposition for improving coordination and information transfer across multiple disciplines (Ballinger and Sobieszczanski-Sobieski, 1994; Kroo et al., 1994) and for streamlining the design process (Eppinger et al., 1994; Kusiak and Wang, 1993; Rogers and Bloebaum, 1994) applying sequential decomposition to the design sequence.

The present work was also motivated by the undirected graph representation of the optimal design problem used by Wagner and Papalambros (1993a, b). Mathematical relations and design and state/behavior variables were depicted by the vertices and edges of the graph, respectively. Identification of "linking" or

"coupling" variables y leads to independent design subproblems that correspond to connected components in the graph when linking variables are deleted. The remaining variables in each subgraph are "local" variables x of the corresponding subproblem k. Heuristic acceptability criteria were used to select appropriate linking variables (resulting in a heuristic decomposition strategy). A mathematical programming coordination strategy is then used to solve the original problem as a set of smaller subproblems solved independently but coordinated by a master problem. In hierarchically partitioned problems, the master problem is solved for the linking variables y which are then input as parameters to the subproblems. Information on the dependence of the local variables with respect to the linking variables is fed back to the master problem. Subproblems may be recursively partitioned to generate a multilevel hierarchy. In nonhierarchical decomposition, bidirectional interaction between subproblems may exist and the master problem is replaced by a nonhierarchical coordination strategy.

The heuristic decomposition above can be made more rigorous by modeling the decomposition problem as a network optimization problem (Michelena and Papalambros, 1995c). Design relations are modeled as processing units of a communication network and design and state/behavior variables are the communication links between these units. The optimal decomposition problem is then formulated as one of finding the communication links whose failure lessens the most the network reliability. This form of the OMBD problem has led to the hypergraph-based partitioning methodology given in the next section.

The remainder of the article offers a discussion of the hypergraph representation and formulation of the OMBD problem as a hypergraph partitioning one. Strategies for solving the latter problem and the overall methodology for large scale optimal design are then presented. An application of the decomposition methodology to automotive powertrain systems, based on a model by Wagner (1993), concludes the article. Further details on the decomposition method and an extended version of this paper can be found in Michelena and Papalambros (1995a, b), respectively.

### Optimal Model-Based Decomposition as Hypergraph Partitioning

Optimal decomposition of a design problem calls for (i) minimizing the interconnection between subproblems and (ii) balancing the size of the subproblems. The former is aimed at reducing the size of the master problem and/or the effort to coordinate individual subproblems, and the latter at matching available computational resources. Hence, the OMBD problem may be formulated as the following hypergraph partitioning problem in which vertices represent design relations, and hyperedges depict design and state/behavior variables.

#### Hypergraph K-Partitioning Problem

Given a hypergraph \( \mathcal{H} = (V, E) \) containing \( N \) vertices \( V = \{v_1, v_2, \ldots, v_N\} \) with positive weights \( \omega(v_i) \), and \( M \) hyperedges \( E = \{e_1, e_2, \ldots, e_M\} \) with positive weights \( \omega(e) \), a constant \( 2 \leq K \leq N \), and a partition load (or size) vector \( \mathbf{m} = (m_1, \ldots, m_K) \) such that \( \sum_{i=1}^{N} \omega_i = \sum_{k=1}^{K} m_k \), find a partition of \( V \) into \( K \) disjoint subsets \( P^k = \{V_1, V_2, \ldots, V_K\} \) that minimizes (1) the total weight of the hyperedges cut by \( P^k \), \( C(P^k) \), and (2) \( \sum_{v \in V_k} \omega(v) - m_k \) for every \( k \in \{1, 2, \ldots, K\} \). The set of hyperedges cut by \( P^k \) is \( E_k^c(P_k) = \{e_j \in E : \text{there exist } \nu_i, \nu_j \text{ in } e_j \text{ and } V_i \in P^k, \nu_i \in V_i \in P^k, \text{and } j_1 \neq j_2 \} \).
fore, the total weight of the hyperedges cut by \( P^* \) is \( C(P^*) = \sum_{e \in E_P(P^*)} \omega(e) \).

When this formulation is applied to the OMBD problem, weights represent computational costs (e.g., CPU time or memory) for the design relations, edge weights depict coupling strength or amount of transferred data between computational (e.g., simulation) modules, and partition loads represent processing capabilities in a distributed computational environment. In this article, the terms hypergraph and graph \( K \)-partitioning imply some sort of constraint on the partition subset sizes (loads), as opposed to the unrestricted common meaning of the terms.

Some partitioning methods found in the literature are only applicable to graphs. A graph, also known as a linear graph, is a hypergraph in which the cardinality of every hyperedge is equal to two. Specifically, spectrum-based methods have only been developed for graphs, their extension to hypergraphs being a major challenge. Thus, partitioning a hypergraph representation may require approximating the hypergraph by a graph.

Hypergraphs are also used to model circuit netlists in VLSI design, where the vertices of the hypergraph represent modules or cells, and the hyperedges represent signal nets. A method used by VLSI designers to approximate a hypergraph \( H \) by a graph \( G \) consists in defining the vertex set of \( G \) the same as the vertex set of \( H \). The edge set of \( G \) is obtained by replacing each hyperedge of \( H \) by the edge set of a clique containing the vertices of the hyperedge. That is, a clique of \( G \) that contains every vertex of the hyperedge is created. Resulting parallel edges in \( G \) are replaced by a single edge whose weight is determined by adding the weights of the parallel edges. Weighted edges are needed to estimate the number of hyperedges of \( H \) cut by a vertex set partition from the weights of the edges of \( G \) cut by the same partition. Several hyperedge models can be found in the circuit and graph partitioning literature, each assigning different edge weights to the cliques (see Michelen and Papalambros 1995a, for a summary). Here we consider the “standard” clique model (Lenzner and Skillicorn 1990) that replaces every \( p \)-hyperedge by a clique with edge weights \( \omega = 1/(p - 1) \).

A graph representation of a design problem may be constructed using a hyperedge model. The OMBD problem can then be formulated as a graph partitioning problem with vertices \( V \) representing design relations and (cliques of) edges \( E_G \) representing variables. In the graph partitioning formulation, an edge weight is computed by multiplying the weight defined by the hyperedge model and the weight of the associated hyperedge (which depicts coupling strength or amount of transferred data between computational modules). As mentioned above, minimizing the total weight of cut hyperedges is equivalent to minimizing the total weight assigned to linking variables. However, minimizing of the weight of cut edges in a graph representation results in the minimization not only of the weight of linking variables but also of the number of functions on which these variables depend.

Graph and Hypergraph \( K \)-Partitioning Techniques

Both hypergraph and graph \( K \)-partitioning problems are NP-hard, even if edge and vertex weights are one, and the number of partitions is two. If the number of partitions \( K \) is fixed and there is no restriction on the size of the partitions, then the problem is solvable in polynomial time \( O(N^K) \), where \( N \) is the number of vertices in the graph (Goldschmidt and Hochbaum 1994). However, this case has no practical application since the resulting partitions may be very unbalanced.

Partitioning methods include iterative improvement and global techniques. Iterative improvement algorithms, also known as local search algorithms, make local changes to an initial partition to minimize the total weight of the edges cut, while keeping the parts balanced. These algorithms are quite robust because they can deal with graphs and hypergraphs, and arbitrary vertex and edge weights and balance criteria. However, unless they are initialized with a “good” partition, the local optimum may not be global. For global methods, the partitioning problem is formulated as an optimization problem and solved using approximation techniques.

Iterative Improvement Methods. Kernighan and Lin (1970) first proposed a local search method to solve the graph bisection problem. Their algorithm (KL) starts with an initial (arbitrary) partition and iteratively exchanges pairs of vertices to minimize the cut of the bisection. To reduce the risk of being trapped in a local minimum, the KL procedure determines the vertex pair whose exchange results in the largest decrease of the cut size or in the smallest increase, if no decrease is possible. The original KL algorithm can handle edge weights. Dunlop and Kernighan (1985) extended the KL algorithm to hypergraph bisection.

Fiduccia and Mattheyses (1982) proposed a bipartition algorithm based on the KL algorithm that could handle vertex and edge weights, unequal partitions, and hypergraphs. In the Fiduccia-Mattheyses (FM) algorithm a single vertex is moved across the cut in a single move, so that the algorithm can deal with partitions of different sizes and nonuniform vertex weights. Partition balance is enforced by bounds on the partition sizes.

Variants to the KL and FM algorithms continue to appear in the graph and hypergraph partitioning literature. Krishnamurthy (1984) introduced the concept of level gains into the FM heuristics. Sanchis (1989) adapted Krishnamurthy’s bipartition algorithm to solve the hypergraph \( K \)-partitioning problem. Hendrickson and Leland (1993a) have implemented an iterative improvement graph partitioning algorithm patterned after the FM algorithm but generalized in several ways: First, \( K \)-partitioning is possible. Second, the algorithms can handle an arbitrary inter-set cost metric. Third, robustness is improved by including an element of randomness.

Global Methods. Global partitioning methods start with an encoding of a problem instance, such as a graph adjacency matrix or list or a hypergraph incidence matrix, and compute an approximation to the optimal partition that can be used as input to an iterative improvement algorithm.

Simulated annealing (SA) has been used for graph partitioning by Johnson et al. (1989). They showed that SA usually needs much more time than iterative improvement methods, specifically when used on graphs generated with a built-in geometric structure. Bui and Moon (1994) presented a hybrid genetic algorithm that combines a variation of the Fiduccia-Mattheyses algorithm with genetic space exploration to give competitive performance.

Spectral partitioning methods identify a good approximation to a \( K \)-partition from global information about the structure of the graph extracted from the spectral properties of the graph. Typically, a \( K \)-dimensional geometric representation of the graph is constructed from the \( K \) eigenvectors that correspond to the smallest eigenvalues of the graph Laplacian matrix. A drawback of these methods is that they cannot be directly applied to hypergraphs, so a hyperedge model is needed to approximate the hypergraph by a graph. The relation between the spectrum of a graph and
other graph properties has been an area of active research, but only recently spectrum-based methods have been successfully applied to graph partitioning (Alpert and Yao, 1995; Hendrickson and Leland, 1995; Pothen et al., 1990). Michalena and Papalambros (1995a) have extended formulations given by Renderl and Wolfovicz (1990) and Falkner et al. (1994) to account for weighted vertices.

In the powertrain application below, the software package Chaco (Hendrickson and Leland, 1993b) is used to identify optimal partitions of the model. Chaco implements several methods for finding small edge separators in weighted (nodes and edges) graphs. We have selected a spectral method together with a variant of the FM algorithm for global partitioning and its afterward refinement, respectively. The methods contained in Chaco are limited to equal-load partitions (i.e., the case when \(m_i = m_j\) for all \(i = j\)) and recursive bisection and octa-section (so Chaco can only generate 2- and 4-partitions for a whole number \(l\)). A forthcoming version of Chaco is expected to generate an arbitrary number of equal- or unequal-load partitions.

A Decomposition-Based Methodology for Large-Scale Design Problems

The proposed methodology is aimed at advancing the use of nonlinear optimization techniques in the solution of large-scale design problems. Before model decomposition is attempted, the designer must identify the context in which the design process will take place. The design context includes: simulation capabilities, computational environment and cost, existing organizational structures and design decomposition strategies, fidelity of component models, and design objectives such as modularity, robustness, and concurrency. The design context defines decomposition parameters needed to complete the following design steps:

1. Formulate system model, identifying and characterizing system variables, constraints, design criteria, and simulation modules and response surfaces implicit in the model.
2. Identify constraints for model decomposition, including requirements of decomposition-by-components, decomposition-by-disciplines, linearity and monotonicity of subsystem models, computational resources, and organizational structure of the design team.
3. Estimate computational effort to evaluate each model function and to execute each simulation contained in the model.
4. Generate a hypergraph representation of system model. The representation considers decomposition constraints (identified in (2)) modeled with hyperedge weights, and computational efforts (estimated in (3)) modeled with vertex weights. Computational resources determine partition loads.
5. Apply global and local search partitioning techniques to the hypergraph generated in (4) to produce a \(K\)-partition of the system model, for different possible configurations defined by, e.g., number of partitions and partition loads.
6. Carry out system optimization for each model partition obtained in (5).
7. Select best partition(s) from optimization results. Stop if load is sufficiently balanced; otherwise, re-evaluate computational effort for functions and simulation modules, and return to (4).

Decomposition conditions such as decomposition-by-disciplines or decomposition-by-components may require that groups of design constraints or variables belong to the same partition. In design model decomposition, large or small edge weights can force design relations to belong to the same submodel or can delete variables from the model, respectively. Load balancing is achieved when the processing time and memory space demanded by each design subproblem match the available computational resources.

The general methodology postulated above has not been fully implemented to an actual application. The powertrain system application described below has been limited to steps (1), (4) and (5) of the proposed methodology. Step 5, in particular, is fully automated.

Model-Based Decomposition of a Powertrain Design Problem

Synthesis of powertrain components into an optimal system is a difficult task. In practice, this task is completed incrementally by trial and error and is costly and time consuming. As reliable and more accurate models become available, there is greater demand to reduce design cycle time and cost by performing system synthesis using a variety of simulation models. Most of these advances, however, are proprietary in nature. Wagner (1993) presented one of the most comprehensive powertrain system studies available in the open literature with a relatively complete and tested mathematical model having 87 design relations and 127 variables, which include 57 possible degrees of freedom. This model, which considers engine, torque converter, transmission, and wheel-tire assemblies, will be used to illustrate the proposed hypergraph-based decomposition methodology.

Seven design criteria are considered by Wagner: (1) Fuel consumption and (2) emissions directly affect profits since a vehicle that cannot be certified to meet emissions cannot be sold. (3) The distance a vehicle travels from rest in four seconds and (4) the 5-20 mph time correlate with initial acceleration. (5) The 0-60 mph time correlates with average vehicle acceleration over the speed range of the engine. (6) Starting gradeability is important for markets with hilly or mountainous terrain. (7) Cruising velocity at grade is the speed at which a vehicle can climb a six percent grade in fourth gear.

Design variables describe either geometry or a control strategy. Powertrain design and state/behavior variables are summarized in Table A-1 in the Appendix. Dependence of the design relations on the variables can be found in Michalena and Papalambros (1995b). A design relation may be an equality or inequality constraint or the objective function in the powertrain optimization model, and its evaluation could entail some kind of simulation or using a response surface.

Figure 3 shows the functional dependence table for the powertrain optimization model. No reordering of columns and rows results in a block-diagonal FDT, meaning that the optimization model is completely connected.

Before partitioning, the "standard" hyperedge model was used to generate a graph adjacency list from the FDT shown in Fig. 3. The hypergraph representation of the powertrain problem contains 87 nodes and 119 hyperedges since eight variables (with indices: 15, 46, 52, 69, 71, 73, 104, and 127) were assumed to be constant parameters. The graph representation of the powertrain model contains 87 nodes and 560 edges.

Bisection using unit weights for functions (nodes) and variables (hyperedges) resulted in two subproblems having 43 and 44 functions. Partitioning time was 0.33 sec. on a Sun SPARC-station. The reordered FDT is shown in Fig. 4. The linking variables have indices: 3, 6, 16, and 17, which correspond to the intake manifold pressure, engine speed, engine torque, and torque converter impeller torque. Subproblem 2 contains all engine related design relations in addition to the fuel consumption, emissions, and anti-lug constraints, and the accessories torque and torque converter impeller rotational subproblem 1 contains the remaining design relations, including the acceleration, starting gradeability and cruising velocity at grade constraints. The anti-lug (minimum torque) constraint, \#64, which only depends on linking variables, may be included in the master problem for hierarchical decomposition.
Wagner’s bisection of the problem (Wagner, 1993), using
heuristic acceptability criteria for partitions, is not directly com-
parable to this result because Wagner’s model included a time
discretization parameter as design variable that we have opted
to leave out of the model. Wagner identified a set of eleven
candidate linking variables that by exhaustive enumeration and
analysis of its subsets was reduced to four—with indices 3, 6,
16 and the time discretization parameter. Wagner’s decompo-
sition consisted of two subproblems having 41 and 46 functions,
with the subproblem having the non-engine design relations also
containing the accessories torque and torque converter impeller
torque relations.

Quadrisection using unit weights for functions and variables
resulted in three subproblems containing 22 functions and one
subproblem containing 21 functions. Partitioning time was 0.51
sec. The reordered FDT is shown in Fig. 5. The (18) linking
variables have indices: 3, 6, 10, 11, 16, 17, 19, 47, 50, 58, 66,
76, 77, 78, 82, 108, 110, and 116. Subproblems 2 and 4 contain
all engine related design relations in addition to the fuel con-
sumption, emissions and anti-lug constraints, and the accesso-
ries torque and torque converter impeller torque relations. Sub-
problem 1 contains wheel-tire model, powertrain geometry and
vehicle geometry design relations, and the acceleration, starting
gradeability and cruising velocity at grade constraints. Subpro-
blem 3 contains torque converter, transmission and powertrain
geometry design relations. Recursive bisection to generate a
model quadrisection resulted in a decomposition with 19 linking
variables instead.

Octasection using unit weights for functions and variables
resulted in seven subproblems containing 11 and one sub-
problem containing 10 functions. The optimal partition was
achieved by quadrisection followed by bisection. Partitioning
time was 0.66 sec. The reordered FDT is shown in Fig. 6. The
(33) linking variables have indices: 3, 4, 5, 6, 8, 10, 11, 16,
17, 19, 20, 22, 26, 28, 29, 34, 47, 48, 50, 53, 58, 66, 76, 77,
78, 79, 82, 108, 109, 110, 111, 116, and 119. Subproblems 2,
4, 6, and 8 contain all engine related design relations in addition
to the fuel consumption, emissions and anti-lug constraints,
and the accessories torque and torque converter impeller torque
relations. Subproblem 1 contains vehicle geometry design rela-
tions and the acceleration constraints. Subproblem 3 contains
the transmission design relations. Subproblem 5 contains wheel
model, powertrain geometry, and vehicle geometry design rela-
tions, and the starting gradeability and cruising velocity at grade
constraints. Subproblem 7 contains torque converter, transmis-
sion, and powertrain geometry design relations. Recursive bi-
section to generate a model octasection resulted in a decomposi-
tion with 34 linking variables instead. Direct octasection gen-
erated a decomposition with 33 linking variables too; however,
the number of edges cut in the associated graph representation
was slightly greater than for the decomposition described above.

**Discussion and Conclusions**

Spectral graph-partitioning methods together with iterative
improvement hypergraph-partitioning techniques were used to
optimally partition a design model. The methodology was applied to a powertrain problem having 87 design relations and 119 variables. Whereas the partitions obtained by problem bisection may appear obvious, validating an object decomposition based on engine and driveline, assignment of the seven design criteria to each partition and quadrisection and octasection of the problem are not intuitive. Partition of the engine design relations among two and four subproblems in the cases of model quadrisection and octasection, respectively, may not be evident to an experienced designer. The same observation applies to the powertrain geometry design relations for problem quadrisection and octasection, and to the vehicle geometry and transmission design relations for problem octasection.

The partitioning formulation and implementation are robust enough to account for partition loads, function evaluation and simulation times/memory requirements, and the strength of function coupling. Thus, the optimal problem partition may be forced to meet an existing analysis and simulation environment. Finally, the proposed decomposition method can also be used to partition and organize the associated product development process by substituting design tasks and flows of design information for the design relations and variables, respectively, as described in Papalambros and Michela (1995).

Acknowledgment

This work has been partially supported by research grants from Ford Motor Company and the U.S. Army Automotive Research Center for Modeling and Simulation of Ground Vehicles. This support is gratefully acknowledged.

References


Fig. 6 FDT for powertrain problem after optimization model octasection (33 linking variables)
<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>Appendix</th>
<th>Description</th>
<th>Appendix</th>
<th>Description</th>
<th>Appendix</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Vehicle velocity</td>
<td>Appendix</td>
<td>Coefficient of adhesion</td>
<td>Appendix</td>
<td>Intake valve closing timing</td>
<td>Appendix</td>
</tr>
<tr>
<td>2</td>
<td>Vehicle acceleration</td>
<td>Appendix</td>
<td>Traction ratio</td>
<td>Appendix</td>
<td>Intake valve maximum lift</td>
<td>Appendix</td>
</tr>
<tr>
<td>3</td>
<td>Intake manifold pressure</td>
<td>Appendix</td>
<td>Rotational inertia of driving wheel</td>
<td>Appendix</td>
<td>Exhaust valve maximum lift</td>
<td>Appendix</td>
</tr>
<tr>
<td>4</td>
<td>Normal force at driving wheels</td>
<td>Appendix</td>
<td>Rotational inertia of non-driving wheel</td>
<td>Appendix</td>
<td>Cylinder diameter</td>
<td>Appendix</td>
</tr>
<tr>
<td>5</td>
<td>Normal force at non-driving wheels</td>
<td>Appendix</td>
<td>Power conversion efficiency of gearbox</td>
<td>Appendix</td>
<td>Diameter of rod bearing</td>
<td>Appendix</td>
</tr>
<tr>
<td>6</td>
<td>Engine speed</td>
<td>Appendix</td>
<td>Speed reduction ratio of gearbox</td>
<td>Appendix</td>
<td>Width of rod bearing</td>
<td>Appendix</td>
</tr>
<tr>
<td>7</td>
<td>Engine acceleration</td>
<td>Appendix</td>
<td>Span ratio of gearbox</td>
<td>Appendix</td>
<td>Diameter of main bearing</td>
<td>Appendix</td>
</tr>
<tr>
<td>8</td>
<td>Torque converter turbine speed</td>
<td>Appendix</td>
<td>Fourth gear ratio</td>
<td>Appendix</td>
<td>Width of main bearing</td>
<td>Appendix</td>
</tr>
<tr>
<td>9</td>
<td>Torque converter turbine acceleration</td>
<td>Appendix</td>
<td>Rotational inertia of gear box</td>
<td>Appendix</td>
<td>Number of main bearings</td>
<td>Appendix</td>
</tr>
<tr>
<td>10</td>
<td>Drive shaft speed</td>
<td>Appendix</td>
<td>Indent of gear</td>
<td>Appendix</td>
<td>Induction system discharge coefficient</td>
<td>Appendix</td>
</tr>
<tr>
<td>11</td>
<td>Drive shaft acceleration</td>
<td>Appendix</td>
<td>Rotation inertia of differential</td>
<td>Appendix</td>
<td>Manifold discharge coefficient</td>
<td>Appendix</td>
</tr>
<tr>
<td>12</td>
<td>Driveway axle speed</td>
<td>Appendix</td>
<td>Reduction ratio at differential</td>
<td>Appendix</td>
<td>Post discharge coefficient</td>
<td>Appendix</td>
</tr>
<tr>
<td>13</td>
<td>Driveway axle acceleration</td>
<td>Appendix</td>
<td>Rotation inertia of differential</td>
<td>Appendix</td>
<td>Intake manifold runner length</td>
<td>Appendix</td>
</tr>
<tr>
<td>14</td>
<td>Tire slip ratio</td>
<td>Appendix</td>
<td>Rotation inertia of drive shaft</td>
<td>Appendix</td>
<td>Exhaust manifold runner length</td>
<td>Appendix</td>
</tr>
<tr>
<td>15</td>
<td>Speed ratio across torque converter</td>
<td>Appendix</td>
<td>First gear ratio</td>
<td>Appendix</td>
<td>Exhaust manifold runner diameter</td>
<td>Appendix</td>
</tr>
<tr>
<td>16</td>
<td>Engine torque</td>
<td>Appendix</td>
<td>Second gear ratio</td>
<td>Appendix</td>
<td>Exhaust gas recirculation</td>
<td>Appendix</td>
</tr>
<tr>
<td>17</td>
<td>Torque converter impeller torque</td>
<td>Appendix</td>
<td>Third gear ratio</td>
<td>Appendix</td>
<td>Spark timing</td>
<td>Appendix</td>
</tr>
<tr>
<td>18</td>
<td>Torque entering gear box</td>
<td>Appendix</td>
<td>Torque converter impeller angle</td>
<td>Appendix</td>
<td>Temperature of charge in intake manifold</td>
<td>Appendix</td>
</tr>
<tr>
<td>19</td>
<td>Torque entering final drive</td>
<td>Appendix</td>
<td>Rotational inertia of torque converter impeller</td>
<td>Appendix</td>
<td>Tuning pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>20</td>
<td>Churning losses in gear box</td>
<td>Appendix</td>
<td>Rotational inertia of torque converter impeller</td>
<td>Appendix</td>
<td>Tuning manifold pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>21</td>
<td>Torque at driving axle</td>
<td>Appendix</td>
<td>Capacity factor of torque converter at stall</td>
<td>Appendix</td>
<td>Indicated mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>22</td>
<td>Force at driving wheels/road interface</td>
<td>Appendix</td>
<td>Capacity factor of torque converter</td>
<td>Appendix</td>
<td>Total friction mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>23</td>
<td>Grade angle</td>
<td>Appendix</td>
<td>Torque ratio of torque converter at stall</td>
<td>Appendix</td>
<td>Pumping mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>24</td>
<td>Starting grade angle</td>
<td>Appendix</td>
<td>Torque ratio of torque converter</td>
<td>Appendix</td>
<td>Piston ring friction mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>25</td>
<td>Cruising velocity at I.S. grade</td>
<td>Appendix</td>
<td>Torsion converter stall speed</td>
<td>Appendix</td>
<td>Main bearing friction mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>26</td>
<td>Vehicle frontal area</td>
<td>Appendix</td>
<td>Engine torque</td>
<td>Appendix</td>
<td>Valvetrain friction mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>27</td>
<td>Vehicle body drag coefficient</td>
<td>Appendix</td>
<td>Speed at peak power</td>
<td>Appendix</td>
<td>Oil pump friction mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>28</td>
<td>Rolling resistance force</td>
<td>Appendix</td>
<td>Engine speed at peak torque</td>
<td>Appendix</td>
<td>Water pump friction mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>29</td>
<td>Coefficient of rolling resistance</td>
<td>Appendix</td>
<td>Stroke/Surge</td>
<td>Appendix</td>
<td>Valve train type</td>
<td>Appendix</td>
</tr>
<tr>
<td>30</td>
<td>Axle resistance force</td>
<td>Appendix</td>
<td>Burn duration</td>
<td>Appendix</td>
<td>Brake mean effective pressure</td>
<td>Appendix</td>
</tr>
<tr>
<td>31</td>
<td>Weight of aerodynamic effort</td>
<td>Appendix</td>
<td>Smoke</td>
<td>Appendix</td>
<td>Total thermal fraction in engine</td>
<td>Appendix</td>
</tr>
<tr>
<td>32</td>
<td>Height of center of gravity</td>
<td>Appendix</td>
<td>Stroke</td>
<td>Appendix</td>
<td>Fuel flow rate</td>
<td>Appendix</td>
</tr>
<tr>
<td>33</td>
<td>Mass of vehicle</td>
<td>Appendix</td>
<td>Stroke/Surge</td>
<td>Appendix</td>
<td>Non-flow rate</td>
<td>Appendix</td>
</tr>
<tr>
<td>34</td>
<td>Distance from center of gravity to driving axle</td>
<td>Appendix</td>
<td>Compression ratio</td>
<td>Appendix</td>
<td>Thermal capacity</td>
<td>Appendix</td>
</tr>
<tr>
<td>35</td>
<td>Weight base</td>
<td>Appendix</td>
<td>Engine vacuum</td>
<td>Appendix</td>
<td>Surface-to-volume ratio</td>
<td>Appendix</td>
</tr>
<tr>
<td>36</td>
<td>Length of axle</td>
<td>Appendix</td>
<td>Intake valve diameter</td>
<td>Appendix</td>
<td>Volume efficiency</td>
<td>Appendix</td>
</tr>
<tr>
<td>37</td>
<td>Weight distribution factor at driving wheel</td>
<td>Appendix</td>
<td>Intake valve opening time</td>
<td>Appendix</td>
<td>Torque of accessories</td>
<td>Appendix</td>
</tr>
<tr>
<td>38</td>
<td>Weight distribution factor at non-driving wheel</td>
<td>Appendix</td>
<td>Intake valve closing time</td>
<td>Appendix</td>
<td>Rotational inertia of accessories</td>
<td>Appendix</td>
</tr>
<tr>
<td>39</td>
<td>Dynamic weight</td>
<td>Appendix</td>
<td>Intake valve opening time</td>
<td>Appendix</td>
<td>Catalyst NOX efficiency</td>
<td>Appendix</td>
</tr>
<tr>
<td>40</td>
<td>Effective tire radius</td>
<td>Appendix</td>
<td>Intake valve opening time</td>
<td>Appendix</td>
<td></td>
<td>Appendix</td>
</tr>
</tbody>
</table>