U.S. Department
of Transportation

**National Highway
Traffic Safety
Administration**

DOT HS **January 2006**
Final Report

The SISAME Program: Structural Crash Model Extraction and Simulation

Stuart G. Mentzer
Information Systems and Services, Inc.
8601 Georgia Ave., Suite 708
Silver Spring, MD 20910

Prepared for:

National Highway Traffic Safety Administration
400 7th Street, S.W.
Washington, DC 20590

Under Contract No. DTNH22–01–C–07317

Technical Report Documentation Page

1. Report No. DOT HS	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle THE SISAME PROGRAM: Structural Crash Model Extraction and Simulation		5. Report Date January 2006
		6. Performing Organization Code
7. Author(s) Stuart G. Mentzer		8. Performing Organization Report No.
9. Performing Organization Name and Address Information Systems and Services, Inc. 8601 Georgia Ave., Suite 708 Silver Spring, MD 20910		10. Work Unit No. (TRAIS)
		11. Contract or Grant No. DTNH22–01–C–07317
12. Sponsoring Agency Name and Address National Highway Traffic Safety Administration 400 7th Street, S.W. Washington, DC 20590		13. Type of Report and Period Covered Final Report October 2001 – January 2006
		14. Sponsoring Agency Code
15. Supplementary Notes Office of Crashworthiness Research/NHTSA, NRD-11		

16. Abstract

This report contains the users manual and formulation for the SISAME v.2.0 structural impact modeling program. SISAME extracts one-dimensional lumped-parameter structural models from crash test data, and performs simulations of such models.

SISAME uses a combination of a user-specified model configuration and crash test motion and, optionally, force data in a global approximation scheme to find optimal models. Any subset of the load-paths and mass element weights can be extracted from sufficiently complete and accurate sets of test data. The full time span of the test data is used in extracting the parameters for a robust solution. The approximation is based on the full (overdetermined) set of equations of motion for the model's instrumented masses and target equations for specified force histories. The load-paths are parameterized to obtain a tractable problem with a modest number of unknown parameters. Linear inequality constraints are used to assure physically meaningful load-paths. Weighted estimates and/or constraints can be used for any extracted parameter.

SISAME provides a range of static and dynamic load-path elements, extensive defaults and error checking, and a range of output quantities and formats, and has a friendly, self-documenting user-interface. Sophisticated algorithms are incorporated to ensure efficient reliable extractions and simulations.

The SISAME source code is available upon request from the National Highway Traffic Safety Administration, Office of Crashworthiness Research (NRD-11).

17. Key Words Lumped-parameter model, computer simulation, least squares, constrained optimization.	18. Distribution Statement This document is available to the public from the National Technical Information Service, Springfield, Virginia 22161.

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of pages 116	22. Price

Form DOT F 1700.7 (8-72)

Reproduction of completed page authorized

Contents

List of Figures

1 Introduction

The SISAME (**S**tructural **I**mpact **S**imulation **A**nd **M**odel **E**xtraction) program is a general purpose tool for extracting and simulating one-dimensional lumped-parameter structural models in large deformation crash and impact events. SISAME was developed for the U.S. Department of Transportation, National Highway Traffic Safety Administration, Office of Crashworthiness for modeling nonoblique vehicle impacts.

The goal of the SISAME methodology is to develop vehicle crash models of modest complexity that can accurately and efficiently predict the structural behavior in a range of similar events. The ability to simulate these models in a range of single and multiple vehicle crash events provides a capability that would not be practical with crash testing. The ability to examine the effects of modified structural properties and even to find the structural changes needed to achieve safety-modified crash data is an additional benefit for vehicle safety research.

SISAME uses specialized algorithms to find optimal models directly from crash test data, bypassing the difficulties in getting accurate large deformation models from structural analysis, static testing, and finite element approaches. SISAME models are high-level, with compound yet physically meaningful load-paths and a moderate number of parameters. By allowing any mix of user-specified and extracted parameters, and estimates for and/or constraints on any extracted parameter, SISAME uses all information available to obtain the most accurate model possible. SISAME also provides a wide range of features with a friendly self-documenting interface, making it a suitable tool for serious modeling projects.

Section 2 presents the basic SISAME modeling concepts. Section 3 contains the SISAME Users Guide. Section 4 describes the load-path static types available. Section 5 describes the load-path dynamic types available. Section 6 presents the SISAME methodology formulation. Appendix A presents a simple SISAME modeling example. Appendix B provides guidelines for SISAME modeling.

1.1 Overview

SISAME models are one-dimensional structures containing rigid mass elements connected by nonlinear, path-dependent, energy-dissipating load-path (spring) elements, such as the 3-mass, 6-spring passenger vehicle model in Figure 1.1. To simulate such a model, the model configuration, mass weights, attachments, spring characteristics, and initial mass velocities

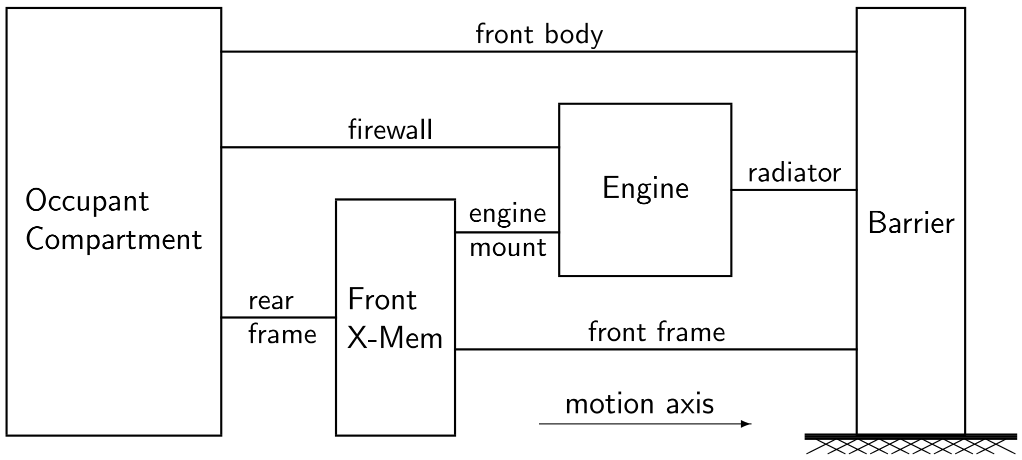


Figure 1.1: A simple SISAME vehicle model.

must be specified. Numerical simulation of the model can then be performed by SISAME to determine the model's behavior.

Extracting a structural model from test data is intrinsically much more difficult than simulation. In general, extraction is the process of solving for some of the structural characteristics of the model when given some motion and/or force data from the test. Since realistic vehicle models have more load-paths than mass elements, the instantaneous equations of motion are typically nonsquare and thus a determinate solution for the load-path forces at each time step is not possible. Instead, SISAME performs a constrained least squares solution for the optimal model parameters using the entire time span of the instrument data. This turns the underdetermined instantaneous problem into an overdetermined full-span optimization problem.

This global approach and the use of constraints to eliminate nonphysical behavior make the extraction process fairly robust. If the motions are accurate and the actual system can be accurately represented by the chosen model configuration, then the model will be able to accurately reproduce the test event and to predict events that are sufficiently similar to the test event. If, in addition, the data is sufficiently complete and the model configuration is sufficiently accurate that only a small range of models can accurately match the data, then the model will be an accurate representation of the actual structure.

SISAME is able to extract load-path characteristics and mass weight values. The inputs are the model configuration, defined mass weights, the motion histories of some or all of the masses, and, possibly, measured force histories. The motion and force data are augmented by constraints on the load-path characteristics (based on *a priori* knowledge of load-path behavior) and, optionally, user-specified estimate and/or constraint information.

In most cases a basic model extraction relying on SISAME defaults will produce a good model. When greater control over the extraction is desired the user can adjust the target confidence bands and/or add supplementary information such as weight estimates, constraints on load-path force levels, or target forces for load-path subsets from load-cell data. SISAME is designed to use available information about a structure to refine its solution in a balanced way.

1.2 History

The historical approach to developing vehicle crash simulation models has been to use a combination of engineering analysis, experience, and static crush testing to specify the characteristics of the major load-paths. Models developed in this way have achieved only limited success because of the difficulty in capturing the dynamic crush behavior in a static test.

The original approach for extracting models directly from crash test data was the Fiat methodology [2, 3, 6], which was effectively restricted to static models with only as many load-paths as mass elements (square systems), and performed a determinate solution for the load-path forces at each instant that did not necessarily produce physically meaningful structures.

A preliminary version of SISAME [7] also performed a determinate extraction that limited the number of extracted load-paths to the number of mass motions provided, but supported supplementary user-defined load-paths and dynamic effects, and provided data preprocessing tools and accurate numerical techniques to produce more realistic models.

The first extraction method using approximation techniques to allow an arbitrary number of extracted load-paths (nonsquare systems) is presented in the thesis of Dr. W.T. Hollowell [1]. This approach solved for optimal chronological linearizations of the static load-paths with automatic time segmentation, and was successful in identifying the characteristics of nonsquare models.

This approach was modified to the global method now used in SISAME with the aims of providing greater robustness and directly generating simulation-ready models. SISAME v.2 adds many capabilities to this basic approach to provide a production-level modeling tool.

1.3 Capabilities

SISAME can be used to model a range of essentially nonoblique impact events. This includes full frontal and non-angled offset vehicle impacts into fixed and moving barriers, two vehicle full frontal and offset impacts, and right-angled side impacts of a moving barrier or pole into a vehicle.

SISAME provides a large enough set of load-path properties to further support a broader range of modeling tasks. SISAME vehicle models can be extended with simple biomechanical models to study the timing, severity, and effect of interaction of occupants with the vehicle structures. SISAME can also be used to perform more general one-dimensional lumped-parameter modeling of impact and non-deformation events.

SISAME can perform weight extractions to help define a model's weight distribution when vehicle tear-down data is incomplete or unavailable.

SISAME can be used in high-level vehicle design. Comparing a model extracted from safety-enhanced (based on occupant modeling) instrument signals to the original model provides safety-based redesign guidance. SISAME could be readily extended to include material or construction cost functions in selecting the new design.

SISAME can be used to distill small, efficient models from complex finite element (FEM) models for simulation of events sufficiently similar to the FEM-simulated "test" event.

Multiple-event modeling can be performed with a companion program, SISAMEM, to obtain models that are a balanced match to an optionally weighted combination of events having some components in common.

A number of features make SISAME easy to work with. The input file format is flexible and self-documenting, provides many defaults, and supports powerful referencing relationships between model components and parameters. Model extractions can incorporate any available supplementary information about the structure and the knowledge of the modeler into estimates and bounds on the extracted parameters that refine the model solution. The modeler can also control the relative weighting of the different types of target equations used in the extraction process. Known portions of a model can also be simulated while other portions are being extracted. The filtering specifications for each instrument signal can be controlled separately. Extraction runs directly generate the signals and measure of fit report needed to assess the extracted model, and the model file needed to simulate it.

The motion filtering method developed for SISAME modeling can be used for other applications and is available separately in the SimFil program [9, 10]. This filtering approach provides a uniform treatment of the acceleration, velocity, and displacement domain motion signals that maintains their analytic relationships while providing non-distorting filtered representations of each.

1.4 Future

SISAME was initially developed for nonoblique full frontal crash modeling. The version of SISAME presented here has been extended with a number of features to support additional applications, including offset and side impact modeling. In non-angled offset and side impact events the vehicle(s) generally experience some rotation later in the event, which may limit the applicability of current one-dimensional SISAME modeling. Other features of these events can be successfully modeled with a one-dimensional model. For example, the bending across the front of a vehicle during an offset impact can be modeled by deformable shear-coupling between frontal interface mass elements. But to fully capture the structural behavior in these events, and to model angled impacts, two- or three-dimensional modeling is required. A three-dimensional version, SISAME-3D, is under development to address these applications.

1.5 General Information

SISAME is driven by a self-documenting input file. This version uses NHTSA UDS data files for input and output of time series and load-path data but is readily adaptable to other data file formats. SISAME can operate in either metric or English dimensional system. Extensive error checking is performed on the SISAME inputs to assure that the program input requirements are met and that the results are meaningful.

SISAME version 2.0 is written in portable, object-oriented C++ and is available for a number of computing platforms. The earlier SISAME version 1 release was written in Fortran 77 and had fixed model size limits.

A companion program, SISAMEM, can be used for multiple-event SISAME modeling.

The associated VeCor [11] program can be used to assess/refine the data prior to modeling. The filtering used in SISAME is taken from the SimFil program [9, 10]. These programs are also integrated into the NHTSA computing environment.

1.6 Documentation

This document applies to SISAME version 2.0. The SISAMEM program is documented separately. The VeCor program is documented in [11]. The SimFil program is documented in [9, 10].

2 SISAME Modeling Concepts

This section presents the basic terminology, conventions, and concepts used in SISAME modeling. The terminology and conventions are described in Section 2.1. The output types available are described in Section 2.2. The extraction process is discussed in Section 2.3.

2.1 SISAME Models

SISAME models are lumped-parameter **structures** consisting of rigid **mass** elements connected by massless **spring** elements corresponding to the load-paths of the model. SISAME models are one-dimensional: all motion is along a single **motion axis**.

Simulation models are models with all mass weights and load-path characteristics defined.

Extraction models are models with undefined mass weights and/or load-path characteristics. Extractions are based on finding the unknown model parameters to best satisfy a number of **target equations**. Target equations may include the equations of motion for mass elements, equations targeting sums of load-path and/or mass inertia forces to a given force history, and modeler-supplied estimates for the unknown parameters.

2.1.1 Model Structures

SISAME **structures** are composites of model elements and nested structures. The **model** structure consists of the entire model and can contain both model elements and **vehicle** structures.

2.1.2 Model Elements

There are two classes of SISAME masses. The motions of **simulated masses** are determined entirely by the applied forces, integrating the differential equations of motion from an initial displacement and velocity. **Instrumented masses** have their motions specified by input motion histories, and are sub-classified as target or driven masses. **Target mass** motion data is used to provide target equations for the model extraction process. Target masses correspond to motions obtained from the test event and their target equations of motion contribute to the solution for adjacent extracted springs. **Driven mass** motions are either determined by forces external to the model, such as a sled with a prescribed motion profile, or are approximate motions for which target equations are not appropriate, such as displacements taken from test camera films. Instrumented masses can also be **extracted-weight masses**, which have their weight determined by the extraction process. Since SISAME can simulate defined substructures of a model while other substructures are being extracted, an extraction run can include masses of any class. A simulation run can include simulated and/or driven masses, but no target masses. The relevant quantities for each mass are:

Mass Quantity	Symbol	Formula
acceleration	a	
velocity	v	
displacement	d	
weight (in mass units)	m	
weight (in force units)	w	
inertia force	g	$w\,a$
exerted force	$\tilde{g}$	

where the exerted force is the net force acting on the mass.

Fixed **barrier** and **ground** model-level elements are provided for modeling impacts with a fixed structure and loads transmitted to ground. They behave identically but both are provided to allow partitioning of such loads into corresponding barrier and ground forces.

There are two classes of SISAME springs. **Defined spring** characteristics are fully specified in the model description. **Extracted spring** characteristics are partially or fully determined by the extraction process. An extraction run can have all extracted springs or can include some defined springs. A simulation run includes only defined springs.

A spring element can have a **static aspect** and a **dynamic aspect**. The static aspect determines the static force generated by the spring as a function of the current and, possibly, previous deflections. The dynamic aspect determines the crush rate-dependent contribution to the static force that gives the total spring force. SISAME provides both elastic (non-dissipating) and inelastic (energy dissipating) static aspect types, and both damper and magnifier dynamic aspect types. A spring can have only a static aspect and be a purely static element, or only a damper dynamic aspect and be a simple damper.

Each spring is connected to two masses, $m^{\ominus}$ and $m^{\oplus}$, where $m^{\oplus}$ is treated as farther forward along the motion axis. The relevant quantities for each spring are:

Spring Quantity	Symbol	Formula
deflection	x	$d(m^{\ominus}) - d(m^{\oplus})$
relative velocity	r	$v(m^{\ominus}) - v(m^{\oplus})$
static force	f^s	f^s
dynamic magnifier	M	$M(r, \sigma(f^s))$
damping force	δ	$\delta(r)$
force	f	$Mf^s + \delta$
energy	e	$\int f \, \partial x$

where $\sigma(\cdot)$ is the sign function. The spring quantities use a **compression–positive sign convention**.

There are no **contact forces** or **friction forces** other than those generated by appropriate spring elements.

SISAME also provides three classes of **force** elements. A **group force** is a sum of user-specified spring forces and/or mass inertia forces, optionally having an associated comparison force time series. A **target force** is a group force with an associated target force time series (for targeting a group force that depends on extracted parameters). A **driving force** is a force time series applied to one or more masses.

A **simulation model** contains only simulated or driven masses, defined springs, and group or driving forces. Each simulated mass and group force can have an associated **comparison signal** for use in comparing the model behavior with known data.

The mass, spring, and force elements can be grouped into **vehicles** having vehicle-local coordinate systems, properties, and default parameters. The elements, vehicles, and the model as a whole are considered the **components** of a SISAME model.

2.1.3 Model Configuration Limitations

In general, any combination of mass, spring, and force elements is an acceptable SISAME model. Two limitations occur when extraction runs include simulated masses (allowing known portions of the structure to be included in the extraction process). The first limitation is that **extracted springs cannot be connected to simulated masses**; the simulated masses must be connected to the rest of the model with defined springs. The other limitation is that **simulated masses cannot have extracted weights**.

2.2 SISAME Outputs

Tabular history listings, time series files, spring history files, parametric spring files, fit measure reports, and model file outputs are available from a SISAME run. The model file obtained from an extraction run is, in general, ready for use in simulation.

SISAME provides both filtered instrument and effective solution outputs from extraction runs to help assess the extracted model. The effective outputs show the motions and forces that the extracted model would experience at each instantaneous state of the event, and comparing them to the filtered instrument signals shows the quality of the model's fit in the domain that SISAME works in to extract the model. If this comparison is satisfactory the extracted model should be used to resimulate the event to verify that its simulation still tracks the instrument signals satisfactorily.

Because extraction is an approximation process, the forces on a mass will not precisely sum to its measured $w\,a$ value in extraction run outputs as they would in a simulation. Resimulation of an extracted model will give exact consistency for simulated masses, but will not exactly match the original motion data.

2.3 The Extraction Process

This section outlines the steps required in the extraction process. The relevant NHTSA software is indicated for each step.

1. Prepare input data.
   (a) Assess/refine test motion and force histories. VeCor
   The VeCor program or another approach should be used to verify and, if necessary, refine the instrument signals to sufficient accuracy to capture the load-path deflections and other quantities important in the extraction process.
   (b) Assign motions to correspond to instrumented model masses.
2. Build extraction model.
   (a) Assign known weights to target masses.
   (b) Select filtering for instrumented masses and forces.
   (c) Extract unknown weights using target force data (optional). SISAME
   (d) Choose load-paths to include as defined and extracted springs.
   (e) Add known sub-structures (defined springs and simulated masses).
   (f) Add any defined static or dynamic parameters to extracted springs.
   (g) Add extracted spring and/or weight specifications including any desired parameter estimates and/or constraints.
3. Perform extraction run. SISAME
   (a) Request SISAME outputs of filtered and effective motions for target masses, parametric springs, filtered and effective forces for target forces, the fit measure report, and the extracted model file.
   (b) Assess model by comparing fit for target masses and forces, and examining parametric extracted springs for unacceptable features. Refine model inputs and re-extract, if desired.
   (c) The sensitivity of the extracted model parameters can be roughly gauged by repeating the extraction with increased parameter conditioning (a smaller *ConPC*) and seeing how much the solution changes.
4. Perform resimulation run. SISAME
   (a) Use the extracted model file to resimulate the extraction event and compare with the filtered motions produced from the extraction run to assess the model's baseline simulation accuracy.

3 SISAME Users Guide

SISAME runs are controlled by a self-documenting input file. The input files are divided into three sections containing information about the run as a whole, the description of the model, and the list of outputs requested. Each section is made up of a number of fields of the form *Tag=Value* .

Here is a very simple SISAME input file:

```
SISAME Input File

Run Information

  RunID=OneMassMdl
    Title=One-Mass Linear Elastic Model
    DelTOut=.0001  FinTOut=.1

Model Information

  MdlID=OneMassMdl
    DimSys=Metric

    MassID=Mass
      Wt=1000  IniVel=50

    SprID=Spring
      NegMass=Mass  PosMass=Barrier
      StaType=LE  S=10000

Output Information

  OutClass=MassTS  Qty=AVD  Mass=Mass
```

This input file describes a simulation model with one mass element, `Mass`, connected by a spring element, `Spring`, to the built-in `Barrier`. The model uses the metric dimensional system. The `Mass` weighs 1000 kilograms and the `Spring` has a linear elastic static aspect with a stiffness of 10000 newtons/millimeter. The initial conditions have the `Mass` moving at 50 kilometers/hour. Time series file output of the `Mass` acceleration, velocity, and displacement are specified.

The remainder of this Users Guide describes the construction and usage of SISAME input files. The spring static and dynamic aspects are described in Sections 4 and 5.

3.1 Running SISAME

SISAME can be executed with the syntax

`SISAME` [*Input*.*Ext*]

where [*Input*.*Ext*] is an optional input file name. If no input file is specified on the command line SISAME will prompt for one. The extension *Ext* is typically `sim` for simulation files and `ext` for extraction files.

SISAME begins by processing and checking the input file. A log file with a name of the form *Input*`.log` is created. All warning and error messages go to the log file (error messages are also written to the console). A summary of the run and the model is written to the log file. SISAME then performs the extraction or simulation, sending progress messages to the log file and the console. Finally SISAME generates the requested outputs sending tabular output information about the output files to an output file (which defaults to the log file). Note that SISAME will not perform the simulation or extraction for a test run (*Run*`.Class=T`) or if no requested outputs require it.

After successful completion of the SISAME run the user should check the log file for warnings, assess the results by plotting output files and, for extractions, examine the fit measure report. If desired, the input file can then be refined and the model run again.

3.2 Dimensional System

SISAME can operate in either of the dimensional systems shown below:

Data Type	**Metric**	**English**
Time	SECONDS	SECONDS
Distance	MILLIMETERS	INCHES
Velocity	KILOMETERS/HOUR	MILES/HOUR
Acceleration	G'S	G'S
Mass	KILOGRAMS	POUNDS
Force	NEWTONS	POUNDS
Energy	JOULES	FOOT-POUNDS
Frequency	HZ	HZ

3.3 Model Limitations

When designing SISAME models remember that

- Extracted springs cannot be connected to simulated masses.
- Simulated masses cannot have extracted weights.

Model sizes are limited only by the available system memory.

3.4 IDs, Names, and References

The components and parameters of a SISAME model are referred to by name in the input file. The constituents and construction of these references is described below.

3.4.1 Component IDs

The components (elements, vehicles, and the model as a whole) of a SISAME model are identified by the **IDs** *`MdlID`*, *`VehID`*, *`MassID`*, *`SprID`*, and *`FrcID`*. These IDs are used elsewhere in the input file to form names that reference model components and parameters. For example, a spring's *`NegMass`* and *`PosMass`* identify the mass elements (or barrier) that it is connected to.

Here are the rules for component IDs:

- Up to ten characters of an ID are significant.
- IDs are case-sensitive.
- IDs can be made up entirely of numeric characters.
- IDs can be the same as a parameter tag.
- IDs may not contain internal space or tab characters or any of the characters { `.` `,` `+` `*` `=` `#` `?` ~ `!` `<` `>` `[` `]` `(` `)` `{` `}` }.
- IDs may not begin with a `-` (dash) character.
- IDs must be unique among sibling IDs within the same structure and distinct from containing vehicle or model structure IDs. IDs in different structures can be the same but this can require greater care in specifying references to those components.
- IDs cannot case-insensitively match `Sta` or `Dyn` or any static or dynamic aspect type code: `LE`, `EE`, `AE`, `BE`, `SE`, `AI`, `BI`, `SI`, `LD`, `AD`, `LM`, or `AM`.
- `Barrier`, `BarrierFrc`, `Ground`, and `GroundFrc` are reserved model-level IDs.

3.4.2 Component Names and References

Component Names

Components are referred to by their **component names**. The table below shows the symbolic component names used in this document and how they are constructed in SISAME from the component IDs.

Component Names			
Symbolic			Constructed
Component	*Structure*	*Model*	`MdlID`
		Vehicle	*Model*`.VehID`
	Element	*Mass*	*Structure*`.MassID`
		Spring	*Structure*`.SprID`
		Force	*Structure*`.FrcID`

For example, the name of a mass with ID `Engine` that is in a vehicle with ID `Car` that is in a model with ID `Model` is `Model.Car.Engine`.

Component References

Component references are used in SISAME input files to specify spring attachment masses, reference springs, force members, output elements, and as the prefix of parameter references. IDs and component reference matching are case-sensitive.

Shorthand component references are supported, omitting the *Model* or *Vehicle* prefix when the reference resolution rules (see below) will match the desired component. References are taken to be in the local context where they occur. Output specifications are in the model-level context.

Component references are resolved using a “first match” rule that searches up the containment hierarchy from the local context for the first component matching the leading ID in the reference in this order:

1. The local context component.
2. Member components if the local context is a structure.

3. For each structure up the containment tree from the local context up to the model structure:
   (a) The containing structure.
   (b) Member components of the containing structure.

So, for example, within vehicle `Car1` a spring attachment of `NegMass=Engine` suffices as a shorthand reference to the engine in `Car1`, but `NegMass=Car2.Engine` is required to reference the engine in vehicle `Car2`.

3.4.3 Element Set References

Element set references can be used in the force member lists and output element lists. An element set reference is of the form

[*VehIDSet*.] *ElementIDSet* or [`MdlID.`] *ElementIDSet*

where the *VehIDSet*`.` or `MdlID.` prefixes are needed to distinguish references from elements with the same ID in the local context and where an *IDSet* is of the form

$$\mathit{ID} \quad \text{or} \quad \{\mathit{ID}1, \mathit{ID}2, \ldots, \mathit{ID}n\}$$

where the braces surrounding the lists of IDs is optional. Each *ID* can be a component ID or one of the wildcards IDs:

`*` or `[*]`

SISAME constructs an element set from a set reference according to the following rules:

- Each possible combination of *ID*s is used to generate a potential element reference that is then is resolved using the "first match" rule given in Section 3.4.2 above.
- Valid elements of the appropriate type are added to the list. Nonexistent elements or elements of the wrong type are discarded without triggering a warning or error.
- Wildcard *ID* `*` selects all contained components.
- Wildcard *ID* `[*]` selects all immediate member components (but not members of the members).
- A leading wildcard *ID* is resolved in the local context, not by the "first match" rule, as follows:

- In an element context (such as a force member list) the leading wildcard selects all sibling components of the containing structure and, for wildcard `*`, all of their contained components.
- In a structure context (such as an output element list) the leading wildcard selects all member components of the structure and, for wildcard `*`, all of their contained components.

The following are examples of valid element set references:

```
Car.*      *.{FrontFrame,RearFrame}      {Car1,Car2}.{OccComp,Engine}
```

3.4.4 Spring Aspect Names and References

Each spring element can have a static aspect and a dynamic aspect, and these aspects are referred to by their **aspect names** in references to their parameters. The table below shows the symbolic aspect names used in this document and how they are constructed in SISAME from the spring names.

Spring Aspect Names		
Symbolic		Constructed
Aspect	*Static*	*Spring*`.`*StaType*
	Dynamic	*Spring*`.`*DynType*

StaType can be `Sta` or the actual static aspect type: `LE`, `EE`, `AE`, `BE`, `SE`, `AI`, `BI`, or `SI`. So, for example, a shorthand reference to a static `SI` aspect in a spring with ID `FrontEnd` is `FrontEnd.Sta` or `FrontEnd.SI`.

DynType can be `Dyn` or the actual dynamic aspect type: `LD`, `AD`, `LM`, or `AM`. So, for example, a shorthand reference to a dynamic `LM` aspect in a spring with ID `FrontEnd` is `FrontEnd.Dyn` or `FrontEnd.LM`.

Aspect references are resolved by first resolving the component portion using the rules specified in Section 3.4.2 and then identifying the specified aspect. The *StaType* and *DynType* part of aspect references are matched case-insensitively.

Shorthand aspect references are supported, using a shorthand component reference to the aspect's spring. For references to a parameter in the same spring the entire *Spring* portion of the name can be omitted. For references to a parameter in the same aspect the entire *Aspect* can be omitted, starting the reference with the parameter *Tag*.

3.4.5 Parameter Names and References

Parameters are referred to by their **parameter name**. The table below shows the symbolic parameter name used in this document and how it is constructed in SISAME from the Component name or, for static and dynamic aspect parameters, the Aspect name.

Parameter Names	
Symbolic	**Constructed**
Parameter	*Component . Tag*
Parameter	*Aspect . Tag*

Parameter blocks are arrays of parameters indexed from 1 to N. Parameter blocks are referred to by their **parameter block name**, and their sub-blocks and parameters are referred to by **block references**, constructed as shown in the table below.

Parameter Block Names	
Symbolic	**Constructed**
Block	*Component . Tag*
Block	*Aspect . Tag*
Sub-Block	*Block* $i : j$
Sub-Block	*Block* $: j$
Sub-Block	*Block* $i :$
Parameter	*Block* i

Sub-Block references include the block parameters indexed from i to j. If omitted, the value of i defaults to 1 and the value of j defaults to the upper index, N.

Parameter and parameter block references are resolved by first resolving the component portion using the rules specified in Section 3.4.2 and then identifying the specified aspect and parameter or block.

Shorthand references that use a shorthand component or aspect reference (see Sections 3.4.2 and 3.4.4) can be used. For example, references to a parameter in the same component or aspect can omit the *Component* or *Aspect* portion. A spring (non-aspect) parameter can be referenced from its static or dynamic aspect using a shorthand reference of the form *`SprID.`Tag*.

Transformed references of the form

Reference $* m + c$ or $-$ *Reference* $* m + c$

are supported, where *Reference* can be a *Parameter*, *Block*, or *Sub-Block* reference. The unary minus form on the right is equivalent to replacing m replaced by $-m$. Parameters specified with zero-scaled ($m = 0$) references to extracted parameters are treated as defined (nonextracted) parameters with a value of c.

Transformed default references of the form

`#` $* m + c$ or $-$`#` $* m + c$

are supported for numeric parameters with default values. The `#` symbol is a place holder for the default value, as in `DelTSim=#*10`. Omitting the `#` is supported but deprecated. The unary minus form on the right is equivalent to replacing m replaced by $-m$.

See Section 3.5 below for a full description of transform specifiers.

3.5 Transform Specifiers

Transformed parameter references and defaults use the **transform syntax**

$* m + c$

where m is a scale factor and c is an offset value. Either the $* m$ or $+ c$ portions can be used alone. If both are present the scale factor is applied before the offset, giving the effect of (*Value* $* m$)$+ c$. Negative values of m or c can be used as in `*-1` or `+-100` (the `+` is still required before a negative c value). Values of m entered in scientific format should not use a `+` before the exponent to avoid having the exponent treated as the $+ c$ portion (*e.g.*, use `1E4` instead of `1E+4`). The m and c values should not be enclosed in parens and no whitespace should appear within the transform specifier.

3.6 Extracted Parameter Specifiers

An extracted parameter is specified with a question mark. The basic form is simply

Tag `=?` .

Estimates for and/or bounds on the extracted parameter's value can be optionally appended in parens using the form

Tag `=?(` *Specifiers* `)`.

No whitespace should be left between the `?` and the `(`, but whitespace can be used to separate the estimate and bound specifiers within the parens.

The estimate and bound specifiers are of the form

Estimate: `~`*Estimate*`[`*ConfidenceBand*`]`

Lower Bound: `>`*LowerBound*

Upper Bound: `<`*UpperBound*

where any number of each of these (within the line length limit) can appear in any order. The confidence band for an estimate must be a positive number. The bounds are implemented as $\geq$ and $\leq$ constraints despite the use of the `>` and `<` characters.

The estimates and bounds (but not the confidence bands) can be references or transformed references to other parameters (see Sections 3.4.5 and 3.5). (To extract equal-valued parameters use direct *Tag* `=`*Reference* references rather than using distinct `=?` extracted parameters coupled by estimate or bound references.) Note that spring parameter estimate and bound values are not affected by spring transform parameters.

For example, the specifier

```
S=?( ~FrontFrame.BI.SY*2[1000] <5000 )
```

indicates a stiffness that is estimated to be twice that of the `FrontFrame` spring's yield stiffness with a confidence band of 1000, and is no more than 5000.

A sequence of n extracted block parameters is specified using an entry of the form *Tag* `=?`n or *Tag* `=?`n`(` *Specifiers* `)`. Block references in the *Specifiers* estimates or bounds are supported, and the referenced parameters are assigned sequentially to each of the n extracted parameter specifiers (the block reference must contain at least n parameters). For example, the force block specification `F= 0 ?5( >F1:5 ) ?10` could be used to enforce nonnegative slopes on the first five of fifteen segments.

3.7 Input File Overview

Figure 3.1 presents a template for the SISAME input file and Figure 3.2 presents the spring static and dynamic aspect parameter templates. These templates are distributed with SISAME to aid in creating input files.

General notes on the input file are given below.

```
SISAME Input File


Run Information

  RunID=  Descr=  Class=
    Title=
    MaxIter=  ConPC=  ConPD=  MultPD=  Relax=  IterCon=  ConvTol=
    WtgTime=  WtgFac=
    DelTOut=  FinTOut=  DelTSim=  OutFile=


Model Information

  MdlID=  Descr=
    DimSys=  WtMag=
    Wt=  IniVel=  IniDisp=  IniPosn=
    Cutoff=  ZeroSm=  EndSm=  ConV=  ConD=  ConSS=

  VehID=  Descr=  Make=  Model=  Year=  CoordSys=
    Wt=  IniVel=  IniDisp=  IniPosn=
    Cutoff=  ZeroSm=  EndSm=  ConV=  ConD=  ConSS=

    MassID=  Descr=  Class=
      Wt=  IniVel=  IniDisp=  IniPosn=
      File=  CompFile=  Cutoff=  ZeroSm=  EndSm=  ConIF=  ConV=  ConD=

    SprID=  Descr=  Class=
      NegMass=  PosMass=
      Ref=  XScl=  FScl=  RScl=  DScl=  MScl=
      StaType=  (static parameters)
      DynType=  (dynamic parameters)

    FrcID=  Descr=  Class=
      MemSpr=  MemMass=
      File=  CompFile=  Cutoff=  ZeroSm=  EndSm=  ConF=


Output Information

  OutClass=MassTbl  Sub=          Mass=
  OutClass=SprTbl   Sub=          Spr=
  OutClass=FrcTbl   Sub=          Frc=
  OutClass=MassTS   Qty=AVDPavdp  Mass=
  OutClass=SprTS    Qty=XRSDFE    Spr=
  OutClass=FrcTS    Qty=Ff        Frc=
  OutClass=SprYvX   Qty=SDFE      Spr=
  OutClass=SprPar   Qty=SD        Spr=
  OutClass=FitRep   File=
  OutClass=Model    File=


Comments
```

Figure 3.1: SISAME input file template.

Static parameters:

```
StaType=LE  S=

StaType=EE  SO=  C=  P=  XMag=

StaType=AE  SO=  FMax=

StaType=BE  SO=  XC=  FC=  SC=  XT=  FT=  ST=

StaType=SE  XiScl=  FiScl=  Symmetric=  ConSS=  AnySlope=
  X=
  F=

StaType=AI  SO=  FMax=

StaType=BI  SO=  FYC=  FYT=  SY=

StaType=SI  SU=  ST=  XSlk=  XiScl=  FiScl=  Symmetric=  ConSS=  AnySlope=
  X=
  F=
```

Universal static parameters:

```
OneWay=  XScl=  FScl=  XDel=  FDel=  XFailMin=  XFailMax=
```

Dynamic parameters:

```
DynType=LD  DSlp=  DSlk=

DynType=AD  DSlp=  DMax=  DSlk=

DynType=LM  MSlp=

DynType=AM  MSlp=  MMax=
```

Universal dynamic parameters:

```
OneWay=  XScl=  RScl=  DScl=  MScl=
```

Figure 3.2: Spring parameter template.

3.7.1 Input File Structure

- The file header `SISAME Input File` and the section headings `Run Information`, `Model Information`, and `Output Information` are required and must appear on separate lines in that order.
- The `Run Information` section provides information about the run, event, and model as a whole.
- The `Model Information` section describes the vehicles and the mass, spring, and force elements that make up the model.
- The `Output Information` section describes the desired outputs from the run.
- The tags `RunID`, `MdlID`, `VehID`, `MassID`, `SprID`, and `FrcID` are **leading tags** that begin the run, model, vehicle, and element specifications and must be the first tag on a line. The other fields in each section can appear in any order and on as many lines as needed.
- The masses, springs, and forces appearing before any vehicle specification are **model-level elements**.
- A vehicle contains all mass, spring, and force elements following its specifications until the next vehicle, if any, is specified.
- Elements within the model or vehicle sections can appear in any order.
- Blank lines can be used to separate portions of the input file.
- An exclamation point (`!`) character (outside of a quoted string) causes the rest of the line to be treated as a comment.
- `Comment=`*comment* fields can be included in any section of the input file.
- Everything below the optional `Comments` heading is ignored.

3.7.2 Input File Fields

- Input file fields are of the form *Tag*=*Value* for single-valued fields or *Tag*=*Value1 Value2 ...* for **block fields** .
- Individual field values cannot be broken over multiple lines but block fields can use multiple lines.
- IDs are case-sensitive but the aspect type and parameter tag portion of references are case-insensitive.
- Fields and block values can be separated by **whitespace** (spaces and tabs) and commas.
- **String** values that contain `=` or `!` characters must be enclosed within double quote characters (`" "`). Any string value can be enclosed in double quotes if desired.
- A field is considered **unspecified** if it is not present in the input file or if *Tag*= appears with no value.
- If a field is repeated the last value specified is used.

3.7.3 Parameter Specifications

- The numeric, Boolean (True/False), and flag character fields that define the model, event, and extraction/simulation specifications are considered **parameters**.
- **Extracted parameters** are specified with a value of `?` optionally followed by an estimate and/or bound specifier (see Section 3.6).
- Parameters values can be specified as a **reference** to another parameter (see Section 3.4.5). The referenced parameter can be a defined value, extracted, suppressed, or itself a reference to another parameter. The type (numeric, Boolean, or flag) and value/state of the referenced parameter must be compatible with the referencing parameter (*e.g.*, an nonextractable parameter cannot reference an extracted parameter). Acyclic reference chains are resolved by SISAME (circular reference chains and self-references are detected as errors).
- Numeric parameters can be specified with a **transformed reference** that modifies the referenced parameter's value (see Sections 3.4.5 and 3.5).
- A reference to a parameter with different units will cause a warning message that can be safely ignored if the reference has a transform that includes necessary units conversion effects.
- A reference to an extracted parameter sets up a relationship where multiple model parameters share a common internal extracted parameter. The implicit and explicit constraints on all of those model parameters must be satisfied by the internal parameter. SISAME may perform an initial Phase I procedure (see Section 7.8) to obtain parameter values that satisfy all constraints. If no such values are possible SISAME will terminate and report that no feasible solution exists.
- **Weight** parameter values in the input file are in mass units but are converted to force units internally for simulating the equations of motion and setting up default confidence bands. These values are the same (pounds) for the English dimensional system but in the metric system 1 kilogram of mass has a weight in force units of 9.80665 newtons.

3.7.4 Input Data Files

- Relative file name paths are automatically translated to the local operating system format to allow the same input file to be used on different computer platforms.
- The time step of all input time series files must be the same but it need not match the output time step. The time step should be small enough to capture the full frequency content of interest. (SimFil [9, 10] can be used to change the time step of a time series with minimal loss of frequency content.)
- The time span of all input time series must at least equal the final output time, ***`FinTOut`***.

3.8 Input File Specifications

The SISAME input file specifications for each section of the input file are described in the following sections. The specifications are presented in tables having the columns and usage described below.

Tag	Description	Value	Default	N	?

Tag The *Tag* for the field that is used with the syntax *Tag*=*Value* to assign a value to the field. Tags are shown with this typeface: `Tag`.

Description A description of the field.

Value Indicates the type and allowable values and/or range for the field. The value type is one of:

$\mathcal{R}$ Real (floating point). Range, if any, is shown, for example $\mathcal{R} \geq 0$.
Floating point (123.456) or scientific (1.23456E2) formats are accepted.
The internal precision is approximately 15 decimal digits.

$\mathcal{I}$ Integer. Range, if any, is shown, for example $\mathcal{I} \geq 0$.

$\mathcal{B}$ Boolean (True/False). Accepts any of { `True T Yes Y On 1` } for "True" and any of { `False F No N Off 0` } for "False". Case-insensitive.

$\mathcal{C}$ Character. Accepts single character values from the given list, for example $\mathcal{C}$ = `A`, `B`.

$\mathcal{S}$ Character string. A maximum length of L characters is indicated as $\mathcal{S} \leq L$. A list of accepted strings may be shown, for example $\mathcal{S}$ = `StringA`, `StringB`.

Parameters can be specified as references to other parameters (see Section 3.4.5).

Numeric parameters can be specified as transformed references to other parameters (see Sections 3.4.5 and 3.5).

Numeric parameter blocks marked with the $\nearrow$ symbol must have monotonically increasing values.

The value of a field with tag `Tag` is indicated by ***Tag***.

Default The default value, if any, or possibly one of the entries:

> ***computed*** The default value is computed from other model specifications, as described in the notes for that field.
>
> ***hierarchy*** The first specified value from a hierarchy of values is the default value, as described in the notes for that field.
>
> ***none*** By default the field is inactive and is assigned no value (unlike a field with a blank entry, which has no default value).

The default value is used if the parameter value is not specified or implied.

All parameters with a default entry can be specified using the syntax *Tag* = `#` to cause the default value to be used and to appear in the output model file.

Numeric parameters with a default entry can be specified using the **transformed default** syntax *Tag* = `#` $* \, m + c$ (see Section 3.5).

N An `N` in this column indicates that the parameter is **suppressible**. Suppressed parameters are specified by using a value of `N`. A suppressed field causes the associated model feature to be inactive, as described in the notes for that field.

? A `?` in this column indicates that the parameter is **extractable**. Extracted parameters are specified by using a value of or starting with `?` (see Section 3.6) or a reference to an extracted parameter (see Section 3.4.5).

3.8.1 Run Information

Run Information Specifications

Tag	Description	Value	Default	N	?
`RunID`	Run ID	$\mathcal{S} \leq 10$	`Run1`		
`Descr`	Description	$\mathcal{S}$			
`Class`	Run class	$\mathcal{C} =$ `S`, `E`, `T`	*see Notes*		
`Title`	Run title	$\mathcal{S}$			
`MaxIter`	Max number of extraction iterations	$\mathcal{I} \geq 0$	50	`N`	
`ConPC`	Parameter conditioning confidence band factor	$\mathcal{R} > 0$	1	`N`	
`ConPD`	Parameter damping confidence band factor	$\mathcal{R} > 0$	1	`N`	
`MultPD`	Parameter damping multiplier	$\mathcal{R} > 0$	0.5	`N`	
`Relax`	Iteration relaxation control	$\mathcal{B}$	`True`		
`IterCon`	Iteration constraint control	$\mathcal{B}$	`True`		
`ConvTol`	Convergence tolerance control factor	$\mathcal{R} > 0$	1		
`WtgTime`	Time-based weighting times	$\mathcal{R} \geq 0 \ \nearrow$			
`WtgFac`	Time-based weighting factors	$\mathcal{R}$			
`DelTOut`	Output time step	$\mathcal{R} > 0$			
`FinTOut`	Final output time	$\mathcal{R} \geq 0$			
`DelTSim`	Simulation time step	$\mathcal{R} > 0$	*computed*		
`OutFile`	File for run output report and tables	$\mathcal{S}$	*log file*		

Run Information Notes

- `RunID` is the leading tag for the run information section. The ***`RunID`*** appears in all outputs to identify the run.
- The Run ***`Class`*** defaults to `S` for a simulation run and `E` for an extraction run (when extracted parameters are present). Setting `Class=T` requests a test run, with no simulation or extraction performed, where the input file will be processed and the run and model summaries and the output model file can be examined.
- ***`MaxIter`*** is the maximum number of extraction iterations SISAME can use to converge on a self-consistent solution when model features require iteration. `MaxIter=0` produces a single pass extraction. `MaxIter=N` suppresses all extraction passes and causes run outputs to display the initial solution generated by SISAME.
- ***`ConPC`*** is the confidence band factor for the parameter conditioning target equations. These confidence bands are of the form ***`ConPC`*** $\cdot 10^5 \cdot \bar{p}$ where $\bar{p}$ is a baseline magnitude for the type and units of the specific parameter. A smaller ***`ConPC`*** will generally improve numerical conditioning of the extraction computations, possibly at some cost to the solution quality. Setting `ConPC=N`

suppresses parameter conditioning. If the model has extracted parameters that are never active during the event, or other singular or ill-conditioned features, parameter conditioning is required to avoid a singular or ill-conditioned least squares system.

- ***ConPD*** is a confidence band factor for the parameter damping target equations, which target iterative solutions to the previous solution to damp large parameter changes and speed convergence. Parameter damping begins after ($^{\texttt{MaxIter}}/_2$) iterations or when solution cycling is detected. These confidence bands are of the form ***ConPD*** $\cdot 10^3 \cdot \bar{p}$ where $\bar{p}$ is a baseline magnitude for the type and units of the specific parameter. A smaller ***ConPD*** will facilitate convergence of difficult iterative extractions. Setting `ConPD=N` suppresses parameter damping.
- ***MultPD*** is a multiplier applied cumulatively to ***ConPD*** on each iteration pass to speed solution convergence (with ***MultPD*** ≤ 1). A smaller ***MultPD*** will facilitate convergence of difficult iterative extractions. Setting `MultPD=N` suppresses changes to ***ConPD***.
- ***Relax*** controls whether relaxation is enabled during extraction iterations. Relaxation limits the change in the parameter solutions between iterations in an effort to speed convergence. By default ***Relax*** is `True` and SISAME will begin and gradually increase the amount of relaxation after ($^{\texttt{MaxIter}}/_2$) iterations. No relaxation occurs if ***MaxIter*** < 3. Relaxation is also used if solution cycling is detected regardless of ***Relax***.
- ***IterCon*** controls whether iteration constraints are enabled during extraction. Iteration constraints limit the solution change in each iteration to control the error of the linearized approximants used to represent nonlinear parameter contributions. Iteration constraints prevent solution overshooting/oscillation and may be needed to assure convergence, and thus should generally be enabled. By default ***IterCon*** is `True` enabling the iteration constraints.
- ***ConvTol*** is a factor that controls the extraction iteration convergence stopping criteria tolerances. A smaller ***ConvTol*** causes tighter convergence criteria to be used.
- Time-based weighting can be used to emphasize portions of the event by specifying an equal number of ***WtgTime*** and ***WtgFac*** values. An unweighted solution is equivalent to a constant weighting value of 1. The ***WtgFac*** are linearly interpolated and the first and last ***WtgFac*** are extrapolated as constants out to the time span tails to determine the weight at each time step.
- By default a simulation time step is selected by SISAME based on an analysis of the model's frequency content. Specifying a ***DelTSim*** overrides the default value, but care should be exercised to avoid degrading the accuracy of the results. The simulation time step is adjusted, if necessary, to evenly divide ***DelTOut***. Default-based ***DelTSim*** values are set after an analysis of the model so they cannot be used as the target of a parameter reference. ***DelTSim*** specifications are not carried over to extracted output model files.

3.8.2 Model Information

The `Model Information` section contains specifications for the model and vehicle structures and their containing mass, spring, and force elements. The model structure contains all components of the model. Each vehicle structure contains all the elements following the vehicle specifications up to the next vehicle, if any. Elements can appear within the model and vehicle sections in any order.

3.8.2.1 Model Specifications

The model specifications apply to all vehicles, masses, springs, and forces in the model.

Model Specifications

Tag	Description	Value	Default	N	?
`MdlID`	Model ID	$\mathcal{S} \leq 10$	`Mdl1`		
`Descr`	Description	$\mathcal{S}$			
`DimSys`	Dimensional system	$\mathcal{S} =$ `Metric`, `English`			
`WtMag`	Weight magnitude (in mass units)	$\mathcal{R} > 0$	$\overline{m}$		
`Wt`	Model weight (in mass units)	$\mathcal{R} > 0$	*computed*		?
`IniVel`	Model default initial velocity	$\mathcal{R}$			
`IniDisp`	Model default initial displacement	$\mathcal{R}$			
`IniPosn`	Model default initial position	$\mathcal{R}$			
`Cutoff`	Model default cutoff frequency	$\mathcal{R} \geq 0$			
`ZeroSm`	Model default time zero tail smoothing frequency	$\mathcal{R} > 0$		`N`	
`EndSm`	Model default final time tail smoothing frequency	$\mathcal{R} > 0$		`N`	
`ConV`	Model default velocity confidence band factor	$\mathcal{R} > 0$	1	`N`	
`ConD`	Model default displacement confidence band factor	$\mathcal{R} > 0$	1	`N`	
`ConSS`	Model default smoothness confidence band factor	$\mathcal{R} > 0$	1	`N`	

Model Specification Notes

- `MdlID` is the leading tag for the model information section.
- `DimSys=Metric` selects the metric dimensional system. `DimSys=English` selects the English dimensional system. The ***`DimSys`*** can be abbreviated to as little as one character.

- *`WtMag`* is the weight magnitude value (in mass units), $\tilde{m}$, that is converted to $\tilde{w}$ (in force units) for use in setting default confidence bands. *`WtMag`* defaults to the average defined mass weight in the model. A *`WtMag`* value is used if there are any instrumented or driven masses, target forces, comparison signals, or extracted parameters in the model. If there are no defined-weight masses in the model SISAME cannot generate a default for *`WtMag`* and a value must be specified in the input file.
- *`Wt`* is the total model weight.
  - If the model does not contain extracted-weight masses, a specified model *`Wt`* must equal the default computed weight (the sum of the model's defined mass weights excluding any driven masses with unspecified weights). Specifying the *`Wt`* is optional but provides a check on the mass weights. Specifying `Wt=#` causes the computed model weight to appear in the output model file (transformed defaults are not supported).
  - If the model contains extracted-weight masses, specifying a model *`Wt`* adds a total model weight constraint to the extraction (the specified weight must exceed or equal the total of the model's defined mass weights, if any). If the model *`Wt`* is unspecified or if `Wt=?` is specified no total weight constraint is generated and the model weight is an implicitly extracted value (which cannot be referenced). Specifying `Wt=?` causes the extracted model weight value to appear in the output model file.
- *`IniVel`*, *`IniDisp`*, and *`IniPosn`* are used in the mass defaulting hierarchies. Their values are taken to be in the model (global) coordinate system.
- *`ConV`* and *`ConD`* are used in the mass defaulting hierarchies.
- *`Cutoff`*, *`ZeroSm`*, and *`EndSm`* are used in the mass and force defaulting hierarchies.
- *`ConSS`* is used in the spring defaulting hierarchy.

3.8.2.2 Vehicle Specifications

The vehicle specifications apply to all masses, springs, and forces that follow it until the next vehicle specification or the end of the `Model Information` section. (The elements before the first vehicle specification are considered model-level elements.)

Vehicle Specifications

Tag	Description	Value	Default	N	?
`VehID`	Vehicle ID	$\mathcal{S} \leq 10$	`Veh`*`v`*		
`Descr`	Description	$\mathcal{S}$			
`Make`	Make of vehicle	$\mathcal{S}$			
`Model`	Model of vehicle	$\mathcal{S}$			
`Year`	Model year of vehicle	$\mathcal{I}$			
`CoordSys`	Vehicle coordinate system orientation	$\mathcal{C} =$ `+`, `-`	`+`		
`Wt`	Vehicle weight (in mass units)	$\mathcal{R} > 0$	*computed*		?
`IniVel`	Vehicle default initial velocity	$\mathcal{R}$	*hierarchy*		
`IniDisp`	Vehicle default initial displacement	$\mathcal{R}$	*hierarchy*		
`IniPosn`	Vehicle default initial position	$\mathcal{R}$	*hierarchy*		
`Cutoff`	Vehicle default cutoff frequency	$\mathcal{R} \geq 0$	*hierarchy*		
`ZeroSm`	Vehicle default time zero tail smoothing frequency	$\mathcal{R} > 0$	*hierarchy*	N	
`EndSm`	Vehicle default final time tail smoothing frequency	$\mathcal{R} > 0$	*hierarchy*	N	
`ConV`	Vehicle default velocity confidence band factor	$\mathcal{R} > 0$	*hierarchy*	N	
`ConD`	Vehicle default displacement confidence band factor	$\mathcal{R} > 0$	*hierarchy*	N	
`ConSS`	Vehicle default smoothness confidence band factor	$\mathcal{R} > 0$	*hierarchy*	N	

Vehicle Specification Notes

- `VehID` is the leading tag for the vehicle specification. *`VehID`* defaults to `Veh`*`v`*, where *`v`* is the sequential vehicle number.
- Setting `CoordSys=-` specifies a vehicle with a local coordinate system opposite to the global system, which applies to all of its mass motions and forces and their input time series. The $m^{\ominus}$ and $m^{\oplus}$ mass attachments are also reversed for springs within the vehicle. This allows two independent vehicle models to be merged into a frontal impact vehicle–to–vehicle model without reversing all the internal mass attachments of one vehicle, for example. *`CoordSys`* cannot be specified as a parameter reference.

- ***Wt*** is the total vehicle weight.
  - If the vehicle does not contain extracted-weight masses, a specified vehicle ***Wt*** must equal the default computed weight (the sum of the vehicle's defined mass weights excluding any driven masses with unspecified weights). Specifying the ***Wt*** is optional but provides a check on the mass weights. Specifying `Wt=#` causes the computed vehicle weight to appear in the output model file (transformed defaults are not supported).
  - If the vehicle contains extracted-weight masses, specifying a vehicle ***Wt*** adds a vehicle weight constraint to the extraction (the specified weight must exceed or equal the total of the vehicle's defined mass weights, if any). If the vehicle ***Wt*** is unspecified or if `Wt=?` is specified no total weight constraint is generated and the vehicle weight is an implicitly extracted value (which cannot be referenced). Specifying `Wt=?` causes the extracted vehicle weight value to appear in the output model file.
- ***IniVel***, ***IniDisp***, and ***IniPosn*** are used in the mass defaulting hierarchies. Their values are taken to be in the vehicle coordinate system and they default to $\pm$ the corresponding model parameter's value, depending on the vehicle's orientation.
- ***ConV*** and ***ConD*** are used in the mass defaulting hierarchies. They default to the corresponding model parameter's value.
- ***Cutoff***, ***ZeroSm***, and ***EndSm*** are used in the mass and force defaulting hierarchies. They default to the corresponding model parameter's value.
- ***ConSS*** is used in the spring defaulting hierarchy. It defaults to the model ***ConSS*** value.

3.8.2.3 Mass Specifications

Mass Specifications

Tag	Description	Value	Default	N	?
`MassID`	Mass ID	$\mathcal{S} \leq 10$	`Mass`*m*		
`Descr`	Description	$\mathcal{S}$			
`Class`	Class	$\mathcal{C} =$ `S`, `T`, `D`, `d`	*see Notes*		
`Wt`	Weight (in mass units)	$\mathcal{R} > 0$	*none*		?
`IniVel`	Initial velocity	$\mathcal{R}$ *or* `FILE`	*hierarchy*		
`IniDisp`	Initial displacement	$\mathcal{R}$ *or* `FILE`	*hierarchy*		
`IniPosn`	Initial position	$\mathcal{R}$	*hierarchy*		
`File`	Motion file	$\mathcal{S}$	*none*		
`CompFile`	Comparison motion file	$\mathcal{S}$	*none*		
`Cutoff`	Cutoff frequency	$\mathcal{R} \geq 0$	*hierarchy*		
`ZeroSm`	Time zero tail smoothing frequency	$\mathcal{R} > 0$	*hierarchy*	N	
`EndSm`	Final time tail smoothing frequency	$\mathcal{R} > 0$	*hierarchy*	N	
`ConIF`	Mass inertia force confidence band	$\mathcal{R} > 0$	$10\sqrt{w_i\,\tilde{w}}$		
`ConV`	Velocity confidence band factor	$\mathcal{R} > 0$	*hierarchy*	N	
`ConD`	Displacement confidence band factor	$\mathcal{R} > 0$	*hierarchy*	N	

Mass Specification Notes

- `MassID` is the leading tag for the mass specification. ***`MassID`*** defaults to `Mass`*m* where *m* is the sequential mass number within the containing structure.
- The mass ***`Class`*** defaults to `S` (simulated) for a mass without a motion ***`File`***. If a ***`File`*** is specified, the ***`Class`*** defaults to `T` (target) if the mass target equations would be meaningful and to `D` (driven) otherwise. A driven mass can be specified by `Class=D` or `Class=d`: using `D` causes the mass to remain a driven mass in the output model file whereas using `d` causes the mass to become a simulated mass in the output model file.
- A ***`Wt`*** value must be specified for simulated masses. The ***`Wt`*** can be defined or extracted for instrumented masses. A ***`Wt`*** is optional for driven masses.
- `Wt=?` specifies an extracted mass weight value. Extracted weight masses are usually included in target forces to provide SISAME the clearest information about the weight. Alternately, they can appear implicitly by having extracted springs to which they connect in target forces, although this additional coupling of extracted parameters can lead to a significantly different model (increasing the target force ***`ConF`*** value may be helpful). SISAME can return a zero extracted weight if that provides the optimal fit to the test data although that is not a legal weight for simulation. Estimates and/or bounds may be needed to supplement the test data and refine such weight extractions.

- *`IniVel`* defaults to the containing *Vehicle*`.`*`IniVel`* and then $\pm$*Model*`.`*`IniVel`*. `IniVel=FILE` specifies that the associated motion file initial velocity be used.
- *`IniDisp`* defaults to the containing *Vehicle*`.`*`IniDisp`*, $\pm$*Model*`.`*`IniDisp`*, and then zero. `IniDisp=FILE` specifies that the associated motion file initial displacement be used.
- *`IniPosn`* allows mass position outputs that differ from the displacement by the constant *`IniPosn`*$-$*`IniDisp`*. *`IniPosn`* defaults to the containing *Vehicle*`.`*`IniPosn`*, $\pm$*Model*`.`*`IniPosn`*, and then *`IniDisp`*.
- The sign used for the *Model*-based motion parameter defaults depends on the orientation of the vehicle containing the mass: $+$ is used in positively oriented vehicles and $-$ is used in negatively oriented vehicles.
- *`File`* is the name of a file containing the acceleration, velocity, or displacement time series defining the mass motion.
- *`CompFile`* is the name of a file containing the acceleration, velocity, or displacement time series for a motion to be compared with the simulated motion. The initial velocity and displacement of the comparison motion must match those of the simulated mass.
- *`Cutoff`*, *`ZeroSm`*, and *`EndSm`* control the motion filtering for masses with an associated motion file. See the SimFil documentation [9, 10] for more information on the filtering used in SISAME.
- *`Cutoff`* defaults to the containing *Vehicle*`.`*`Cutoff`* and then *Model*`.`*`Cutoff`*. The number of DFT coefficients needed for the filtered signal representation is given by $k = \lfloor 5T_{\mathrm{DFT}}\,\omega_c \rfloor$ where ω_c is the cutoff frequency in Hz and T_{DFT} is the DFT time span in seconds selected by SISAME (generally 10% more than *`FinTOut`* if this motion data is available).
- *`ZeroSm`* defaults to the containing *Vehicle*`.`*`ZeroSm`*, *Model*`.`*`ZeroSm`*, and then *`Cutoff`*. Time-zeo tail smoothing can be suppressed by specifying a value of `N`.
- *`EndSm`* defaults to the containing *Vehicle*`.`*`EndSm`*, *Model*`.`*`EndSm`*, and then *`Cutoff`*. End tail smoothing can be suppressed by specifying a value of `N`.
- *`ConIF`* is an RMS confidence band magnitude for the mass inertia force of target and comparison masses, and of driven masses with defined weights. *`ConIF`* defaults to $10\sqrt{w_i\,\tilde{w}}$ ($10\tilde{w}$ for extracted-weight masses), where w_i is the weight of the mass (in force units) and $\tilde{w}$ is the weight magnitude (in force units). This scaled geometric mean weight force is designed to balance the solution weighting between small and large masses.
- *`ConV`* and *`ConD`* are confidence band factors for the velocities and displacements of target and comparison masses. The weighting of these target equations is based on *`ConIF`*. *`ConV`* defaults to the containing *Vehicle*`.`*`ConV`*, *Model*`.`*`ConV`*, and then 1. *`ConD`* defaults to the containing *Vehicle*`.`*`ConD`*, *Model*`.`*`ConD`*, and then 1. *`ConV`* and/or *`ConD`* can be set to `N` to suppress velocity and/or displacement domain targeting (this will usually speed up the extraction). Decreasing *`ConV`* and/or *`ConD`* will generally improve the extracted model's velocity and/or displacement fit but at some cost to the acceleration fit.

3.8.2.4 Spring Specifications

Spring Specifications

Tag	Description	Value	Default	N	?
`SprID`	Spring ID	$\mathcal{S} \leq 10$	`Spr`*`S`*		
`Descr`	Description	$\mathcal{S}$			
`Class`	Class	$\mathcal{C} =$ `D`, `E`	*see Notes*		
`NegMass`	Negative side mass reference	$\mathcal{S}$			
`PosMass`	Positive side mass reference	$\mathcal{S}$			
`Ref`	Reference spring	$\mathcal{S}$	*none*		
`XScl`	Deflection scale factor ($X_\otimes$)	$\mathcal{R} > 0$	1		
`FScl`	Static force scale factor ($F_\otimes$)	$\mathcal{R} \geq 0$	1		
`RScl`	Relative velocity scale factor ($R_\otimes$)	$\mathcal{R} > 0$	1		
`DScl`	Damping force scale factor ($D_\otimes$)	$\mathcal{R} \geq 0$	1		
`MScl`	Dynamic magnifier scale factor ($M_\otimes$)	$\mathcal{R} \geq 0$	1		
`StaType`	Static aspect type	$\mathcal{S} =$ `LE`, `EE`, `AE`, `BE`, `SE`, `AI`, `BI`, `SI`	*none*		
`DynType`	Dynamic aspect type	$\mathcal{S} =$ `LD`, `AD`, `LM`, `AM`	*none*		

Spring Specification Notes

- `SprID` is the leading tag for the spring specification. *`SprID`* defaults to `Spr`*`s`* where *`s`* is the sequential spring number within the containing structure.
- The spring *`Class`* defaults to `D` for defined springs and `E` for extracted springs. The *`Class`* is uniquely determined by default so explicit specification of the spring *`Class`* is optional.
- *`NegMass`* and *`PosMass`* are references to the mass elements or the fixed `Barrier` or `Ground` masses to which the spring is attached with respect to the vehicle coordinate system. The relative position of the two masses determines whether the spring is in compression or tension for given mass displacements. Note that extracted springs cannot be connected to simulated masses.
- *`Ref`* identifies a spring whose static and dynamic aspects this spring's aspects, if not specified, will refer to. Scaling relative to the referenced spring can be applied by using spring transform scale factors. Acyclic chains of `Ref` references are supported. Note that the referenced spring's deflection range determines any automatic tail deflections.
- **Spring transforms** provide an easy way to modify spring static and dynamic characteristics. **Spring-level** scale factor transforms $X_\otimes$, $F_\otimes$, $R_\otimes$, $D_\otimes$, and $M_\otimes$ (those specified before the static and dynamic aspect specifications) apply to static and dynamic parameters specified or in a referenced *`Ref`* spring. Additional transform parameters are available in the static (see Section 4.1) and dynamic (see Section 5.1) aspect specifications.

  - Transforms are applied to the affected **transforming parameters** with defined values or referencing parameters in other springs. Transforms are *not* applied to parameters specified as extracted (*Tag* `=?(` *Specifiers* `)`) or to parameters referencing other parameters in the same spring.
  - Scale factor transforms are incorporated into the value of affected transforming parameters and do not appear in the output model file.
  - Transforms are incorporated before references to a transforming parameter are resolved.

- `StaType` is the leading tag for the static spring aspect specifications, if any. The static aspect specifications must follow the general spring specifications. The static aspect types are described in detail in Section 4. The parameter templates for each static type are presented in Figure 3.2.
- `DynType` is the leading tag for the dynamic spring aspect specifications, if any. The dynamic aspect specifications must follow the general spring specifications. The dynamic aspect types are described in detail in Section 5. The parameter templates for each dynamic type are presented in Figure 3.2.

3.8.2.5 Force Specifications

Force Specifications

Tag	Description	Value	Default	N	?
`FrcID`	Force ID	$\mathcal{S} \leq 10$	`Frc`f		
`Descr`	Description	$\mathcal{S}$			
`Class`	Class	$\mathcal{C}$ = `G`, `T`, `D`	*see Notes*		
`MemSpr`	Force member spring set list	$\mathcal{S}$	*none*		
`MemMass`	Force member mass set list	$\mathcal{S}$	*none*		
`File`	Force file	$\mathcal{S}$	*none*		
`CompFile`	Comparison force file	$\mathcal{S}$	*none*		
`Cutoff`	Cutoff frequency	$\mathcal{R} \geq 0$	*hierarchy*		
`ZeroSm`	Time zero tail smoothing frequency	$\mathcal{R} > 0$	*hierarchy*	`N`	
`EndSm`	Final time tail smoothing frequency	$\mathcal{R} > 0$	*hierarchy*	`N`	
`ConF`	Force confidence band	$\mathcal{R} > 0$	$10\,\tilde{w}$		

Force Specification Notes

- `FrcID` is the leading tag for the force specification. *`FrcID`* defaults to `Frc`f where f is the sequential force number within the containing structure.
- The force *`Class`* defaults to `G` (group) for a force without a *`File`* and to `T` (target) for a force with a *`File`*. A driving force must be specified by `Class=D`.
- The force member lists support element set reference notation (see Section 3.4.3).
- A group force has member springs and/or masses and no *`File`*, but may have a *`CompFile`*.
- A target force member springs and/or masses and a force time series in the *`File`* field. The force data is used as a target for the corresponding group force value. Target forces are not carried over to the extracted model file.
- The default spring contribution to a group/target force is $(+f)$. The default group/target force contribution from masses is $(-\mathit{FrcOr} \cdot g_g)$, where g_g is the inertia force taken in the global coordinate system and *FrcOr* is the ± 1 orientation of the vehicle containing the force. The inertia force of target masses is based on the actual acceleration not the effective acceleration. Group and target force mass weights must be specified (`Wt=?` masses are allowed).
- The sign of the spring or mass contribution to the group/target force can be user-specified to override the defaults. Placing `(+)` or `(-)` in the member list assigns the corresponding sign to all subsequent elements until another `(+)` or `(-)`, if any, is encountered. The sign flags should be separated from adjacent element names by spaces, as in

```
MemMass= (+) A.Occ A.Eng (-) B.Occ.
```

- A driving force has a list of member masses and a force time series named by ***`File`***. The force orientation is assumed to be that of the local vehicle and the force is applied to *each* member mass. A driving force has no effect on a driven mass except to change its effective motion output.
- The default driving force contribution to its masses in the global coordinate system is ($\mathit{FrcOr} \cdot f^d$), where *FrcOr* is the ± 1 orientation of the vehicle containing the force, and f^d is the driving force time series value.
- A model-level **barrier force** element with the ID `BarrierFrc` is created automatically as the net force of all spring elements attached to the model's `Barrier`. Likewise, a model-level **ground force** element with the ID `GroundFrc` is created automatically as the net force of all spring elements attached to the model's `Ground`. Target or comparison force files can be specified for these force elements within explicit model-level forces with ID `FrcID=BarrierFrc` or `FrcID=GroundFrc` (the force members cannot be changed and the class cannot be set to `D`). These forces are only present if springs are attached to the corresponding fixed masses or if they are explicitly specified.
- ***`File`*** is the name of a file containing the force time series.
- ***`CompFile`*** is the name of the file containing the force time series to be compared with the group force.
- `File=ZERO` can be used to specify a constant target force value of zero. `CompFile=ZERO` can be used to specify a constant comparison force value of zero. Constant `ZERO` forces do not require filtering parameters.
- ***`Cutoff`***, ***`ZeroSm`***, and ***`EndSm`*** control the filtering of forces with an associated force file. See the SimFil documentation [9, 10] for more information on the filtering used in SISAME.
- ***`Cutoff`*** defaults to the containing *Vehicle.`Cutoff`* and then *Model.`Cutoff`*. The number of DFT coefficients needed for the filtered signal representation is given by $k = \lfloor 5T_{\mathrm{DFT}}\,\omega_c \rfloor$ where ω_c is the cutoff frequency in Hz and T_{DFT} is the DFT time span in seconds selected by SISAME (generally 10% more than ***`FinTOut`*** if this motion data is available).
- ***`ZeroSm`*** defaults to the containing *Vehicle.`ZeroSm`*, *Model.`ZeroSm`*, and then *`Cutoff`*. Time-zeo tail smoothing can be suppressed by specifying a value of `N`.
- ***`EndSm`*** defaults to the containing *Vehicle.`EndSm`*, *Model.`EndSm`*, and then ***`Cutoff`***. End tail smoothing can be suppressed by specifying a value of `N`.
- ***`ConF`*** is an RMS confidence band magnitude for the force of target and comparison forces. ***`ConF`*** defaults to $10\,\tilde{w}$, where $\tilde{w}$ is the weight magnitude (in force units). This confidence band is equivalent to 10 g's for a mass weighing $\tilde{w}$ (in force units).

3.8.3 Output Information

Output Specifications

Tag	Description	Value	Default	N	?
`OutClass`	Output class	*see below*			
`Sub`	Subsampling rate	$\mathcal{I} \geq 1$	1		
`Qty`	Output quantity flags	*see below*			
`Mass`	Mass set list	$\mathcal{S}$	*none*		
`Spr`	Spring set list	$\mathcal{S}$	*none*		
`Frc`	Force set list	$\mathcal{S}$	*none*		
`File`	File name for the output	$\mathcal{S}$	*see Notes*		

The `Output Information` template is repeated here with descriptions of each output class:

```
Output Information
  OutClass=MassTbl  Sub=          Mass=       Mass Table listing
  OutClass=SprTbl   Sub=          Spr=        Spring Table listing
  OutClass=FrcTbl   Sub=          Frc=        Force Table listing
  OutClass=MassTS   Qty=AVDPavdp  Mass=       Mass Time Series file
  OutClass=SprTS    Qty=XRSDFE    Spr=        Spring Time Series file
  OutClass=FrcTS    Qty=Ff        Frc=        Force Time Series file
  OutClass=SprYvX   Qty=SDFE      Spr=        Spring Y vs. X History file
  OutClass=SprPar   Qty=SD        Spr=        Spring Parametric file
  OutClass=FitRep   File=                     Fit Measure Report file
  OutClass=Model    File=                     Model file
```

Output Information Notes

- `OutClass` is the leading tag for an output specification.
- Output specifications can appear in any order.
- SISAME pre-scans the output specifications to check for errors before performing the model extraction or simulation.
- Extraction runs perform a final simulation pass only if outputs other than parametric spring or model file are requested.
- The table listings print the relevant time series using the specified subsampling rate, if any.
- A mass *`Qty`* contains any subset of the letters { `A`, `V`, `D`, `P`, `a`, `v`, `d`, `p` }. These request acceleration, velocity, displacement, and position outputs. The upper case letters request the actual motion of the mass. For a simulated mass with a comparison motion, the lower case letters request the (filtered) comparison motions. For an instrumented mass, the lower case letters requests the effective motions: the effective acceleration is the net spring and driving force divided by

the weight, and this is integrated to obtain the effective velocity and displacement. For target masses, the actual and effective motions can be compared to assess the fit in the solution domain.

- A spring *`Qty`* contains any subset of the letters { `X`, `R`, `S`, `D`, `F`, `E` }, which request deflection, relative velocity, static force, dynamic force, force, and energy output, respectively. Only `S`, `D`, `F`, and `E` can be used in `SprYvX` outputs. Only `S` and `D` can be used in `SprPar` outputs.
- A force *`Qty`* contains any subset of the letters { `F`, `f` }, which request group force and filtered (target, driving, or comparison) force output, respectively. For a target force the group force output is the effective force of the extracted components, and this can be compared to the filtered target force to assess the solution fit in the solution domain. There is no group force output corresponding to a driving force.
- The *`Mass`*, *`Spr`*, and *`Frc`* fields contain lists of element and element set references (see Sections 3.4.2 and 3.4.3). The local context for these references is the model structure.
- Multiple *`Qty`* and element specifiers can be used within a single `OutClass` specifier. Each *`Qty`* applies to the subsequent elements specified until the next *`Qty`*, if any.
- Element output files get DOS-compatible "8.3" format names of the form:

  `s` *RunCode Type VehCode* `.` *ElementCode*

  where *RunCode* is up to four characters of the *`RunID`*, *Type* is a two character code for the data type of file, *VehCode* is a unique one-character code generated for the vehicle, and *ElementCode* is a unique three-character code generated for the element. The log file shows the file names used for each requested output.
- Extracted spring force and energy outputs (excluding parametric output) are determined by driving the extracted springs with the instrumented mass motions, not by a resimulation of the model.
- `SprPar` static output files are in the form of force vs. deflection points that give a visual representation of the static aspect when displayed. Failure deflections are not shown in these files.
- `SprPar` dynamic output files are in the form of force or magnification factor vs. relative velocity points that give a visual representation of the dynamic aspect when displayed. Dynamic magnifiers outputs are generated for the $\sigma(f^S) > 0$ case.
- The `SprPar` *`Qty`* defaults to `S` if not specified.
- The **barrier force**, defined as the net spring force on the `Barrier`, will be available for output as a model-level force element with the ID `BarrierFrc` if the barrier force is active in the model.
- The **ground force**, defined as the net spring force on the `Ground`, will be available for output as a model-level force element with the ID `GroundFrc` if the ground force is active in the model.
- The `FitRep` request produces a fit measure report file showing the unweighted and weighted RMS difference between target mass and force instrument and effective extracted signals. Fit measures are also shown for non-target masses and forces that have comparison files, and for driven masses with nonzero weights and adjacent springs or driving forces. The fit reports do not include measures of fit to parameter conditioning, parameter damping, static segment smoothness, and user-specified estimate targets. The `FitRep` output file name defaults to *Input*`.fit` where *Input* is the name of the input file with the path and extension removed.

- The `Model` output file name defaults to *Input*`.mdl` where *Input* is the name of the input file with the path and extension removed.
- `Model` files contain *processed* versions of the springs, incorporating any transforms into the spring parameters where appropriate.
- The model file output from a test run (*`Run.Class=T`*) is the processed input model including the extracted parameter specifications. To obtain a model file with the initial solution parameter values perform an extraction run with `MaxIter=N`.
- Computed or extracted numeric values are formatted with about seven digits of decimal precision in output model files.
- Model files are given the default output specification: `OutClass=MassTS Qty=AVD Mass=*` except for test run (*`Run.Class=T`*) model files, which receive no output specifications.
- Comments are not carried over to the output model file.

4 Static Aspects

SISAME provides eight spring static aspect types, including five elastic and three inelastic (energy dissipating) types.

A compression–positive sign convention is used for the static aspects. The static forces are denoted by f^S. The static stiffnesses are denoted by s^S.

Section 4.2 describes the elastic static aspect types and Section 4.3 describes the inelastic static types.

4.1 Universal Static Parameters

A number of universal static parameters are available for all of the static aspect types. These parameters can be included anywhere within the static aspect specifications (after the `StaType` field). The effects of these parameters are not illustrated in the static aspect figures that follow.

The `OneWay` parameter allows specification of compression-only or tension-only static aspects. A compression-only static aspect (`OneWay=C`) generates only nonnegative static forces. A tension-only (`OneWay=T`) static aspect generates only nonpositive static forces.

Static transforms provide an easy way to modify static aspect characteristics. **Scale factor** transforms $X_\otimes$ and $F_\otimes$ scale the static characteristic along the deflection and force axes. **Offset** transforms X_Δ and F_Δ add offset values to the static characteristic deflections and forces.

Static failure can be specified to occur if the deflection reaches or goes below a given $X^{\mathrm{Fail}}_{\mathrm{Min}}$ value or if it reaches or goes above a given $X^{\mathrm{Fail}}_{\mathrm{Max}}$ value. A failed static aspect generates zero static force for the remainder of the event.

Universal Static Parameters

Tag	Description	Value	Default	N	?
`OneWay`	One-way static aspect flag	`C` *or* `T`	*none*	`N`	
`XScl`	Deflection scale factor ($X_\otimes$)	$\mathcal{R} > 0$	1		
`FScl`	Force scale factor ($F_\otimes$)	$\mathcal{R} \geq 0$	1		
`XDel`	Deflection offset ($X_\triangle$)	$\mathcal{R}$	0		
`FDel`	Force offset ($F_\triangle$)	$\mathcal{R}$	0		`?`
`XFailMin`	Tension failure deflection ($X^{\text{Fail}}_{\text{Min}}$)	$\mathcal{R}$	*none*	`N`	
`XFailMax`	Compression failure deflection ($X^{\text{Fail}}_{\text{Max}}$)	$\mathcal{R}$	*none*	`N`	

Universal Static Parameter Notes

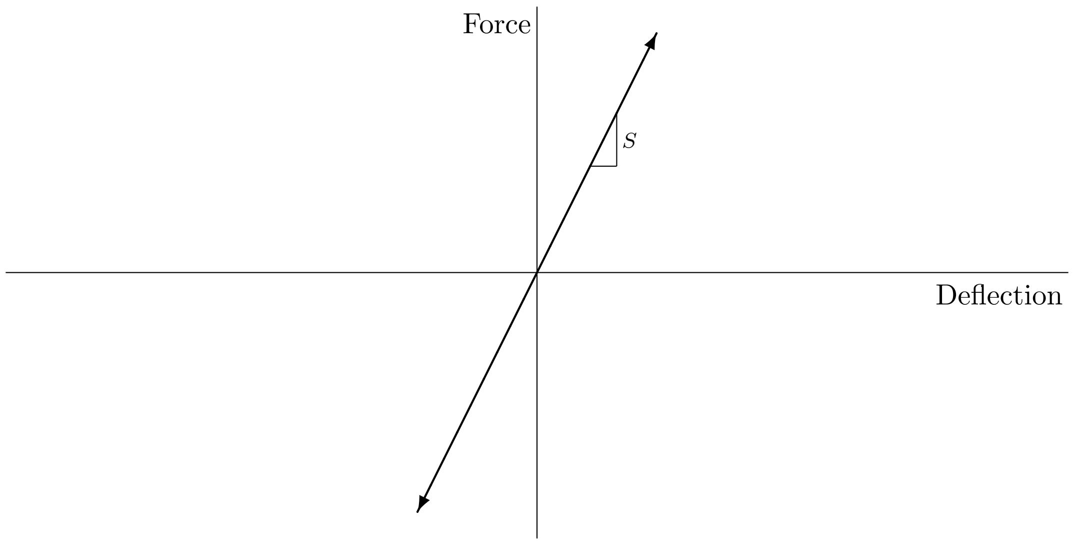
- Extracted one-way static aspects can hamper the solution convergence. Reducing the ***`ConPC`***, ***`ConPD`***, ***`MultPD`***, and/or (for segmented static aspects) ***`ConSS`*** values may improve convergence in such cases.
- Transforms are applied to the affected **transforming parameters** with defined values or referencing parameters in other springs. Transforms are *not* applied to parameters specified as extracted (***`Tag`***`=?(` *Specifiers* `)`) or to parameters referencing other parameters in the same spring.
- Scale factor transforms are incorporated into the value of affected transforming parameters and do not appear in the output model file.
- Transforms are incorporated before references to a transforming parameter are resolved.
- $X_\triangle$ and $F_\triangle$ are transformed by spring-level (but not static) scale factor transforms and are not incorporated into the value of transforming static parameters.
- The effect of $X_\triangle$ and $F_\triangle$ on an inelastic static aspect is to offset the initial static characteristic.
- The combination of a one-way static aspect with $X_\triangle \neq 0$ and/or $F_\triangle \neq 0$ can cause some static parameters to be inactive in the first pass solution. The use of parameter bounds to assure initial activity may be beneficial.
- References to static scale factors should include the ***`StaType.`*** portion to distinguish them from references to corresponding spring-level scale factors.
- $X^{\text{Fail}}_{\text{Min}}$ and $X^{\text{Fail}}_{\text{Max}}$ are scaled by both the static and spring-level $X_\otimes$ values, if any, but the failure deflections are not affected by the value of $X_\triangle$.

4.2 Elastic Static Aspects

SISAME provides five elastic static aspect types. Each of the elastic static aspect types has a static force that is a true (non-path-dependent) function of deflection.

The properties and parameters for each elastic static aspect type are described below.

4.2.1 Static Type LE : Linear Elastic



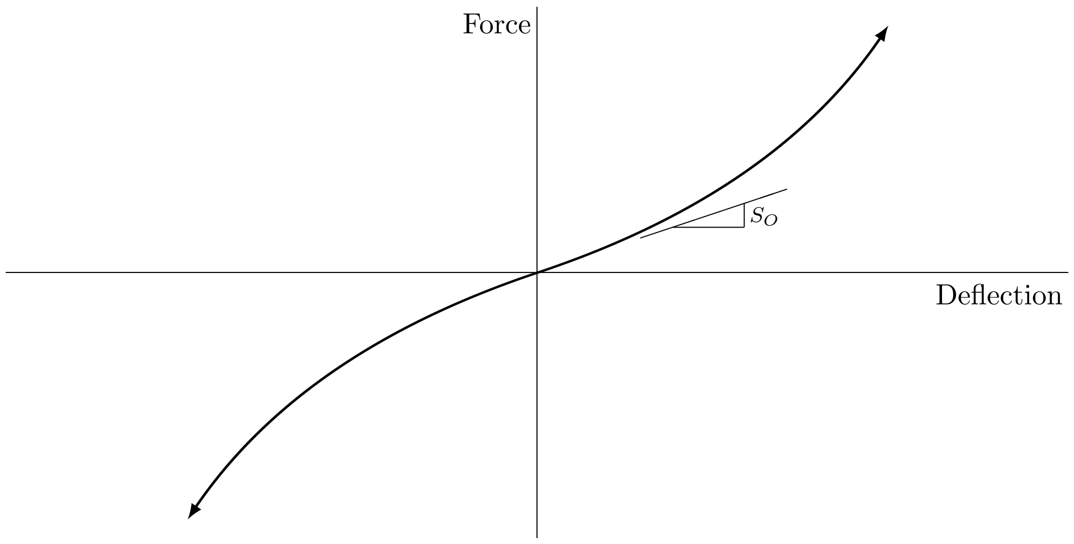
Static type LE is linear elastic with **force** and **stiffness**

$$f^S = Sx \qquad\qquad s^S = S\,.$$

Static Type LE Parameters

Tag	Description	Value	Default	N	?
`S`	Stiffness (S)	$\mathcal{R} \geq 0$			?

4.2.2 Static Type **EE** : Exponential Elastic



Static type EE is exponential elastic with **force** and **stiffness**

$$f^S \;=\; S_O\,x \;+\; C\,\sigma(x)\,|x|^P \qquad\qquad s^S \;=\; S_O \;+\; C\,P\,|x|^{P-1}$$

where S_O is the stiffness at the origin, $\sigma(x) = \pm 1$ is the sign of x, and C and P are the constant and exponent of the exponential term, respectively. If $C > 0$ and $P > 1$ the stiffness increases with $|x|$.

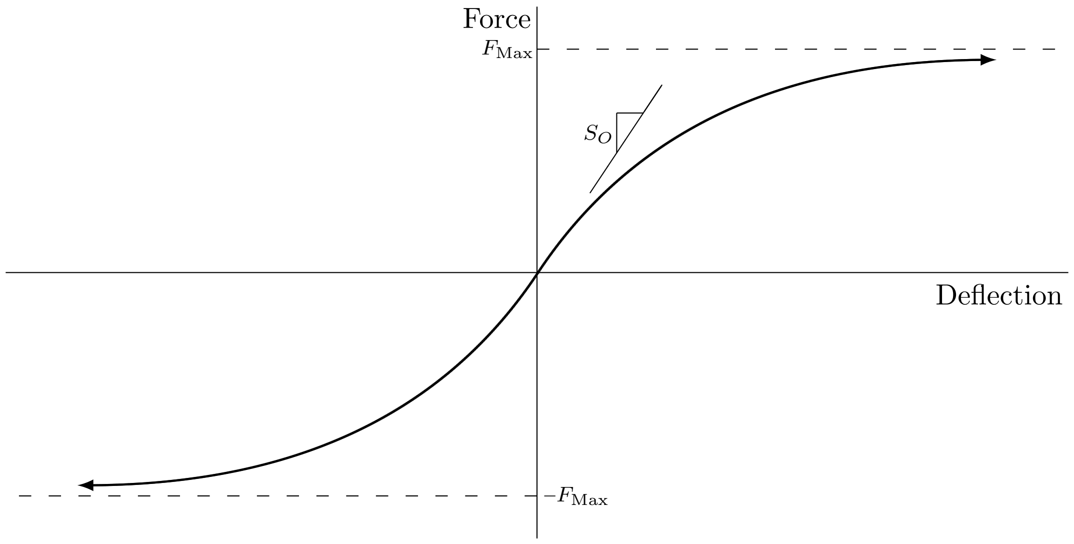
Static Type EE Parameters

Tag	Description	Value	Default	N	?
SO	Origin stiffness (S_O)	$\mathcal{R} \geq 0$			?
C	Constant (C)	$\mathcal{R} \geq 0$			?
P	Exponent (P)	$\mathcal{R} \geq 1$			?
XMag	Deflection magnitude estimate	$\mathcal{R} \geq 0$	0		

Static Type EE Notes

- Extraction of the exponent P can be hindered by the inherent exponential volatility. Despite the default presence of iteration constraints, convergence may benefit from the use of bounds and/or estimates on P.
- *XMag* is used to bound the stiffness for simulation time step selection if the spring is attached to a simulated mass. Increasing *XMag* can force a small time step. A warning is generated if *XMag* is less than the actual deflection magnitude.
- Deflection scaling ($X_\otimes$) cannot be used with an extracted exponent P unless C is a non-reference extracted parameter or a reference to another parameter in the same spring.

4.2.3 Static Type **AE** : Arctangent Elastic



Static type AE is arctangent elastic with **origin stiffness** S_O and asymptotic **maximum force** magnitude F_{Max}.

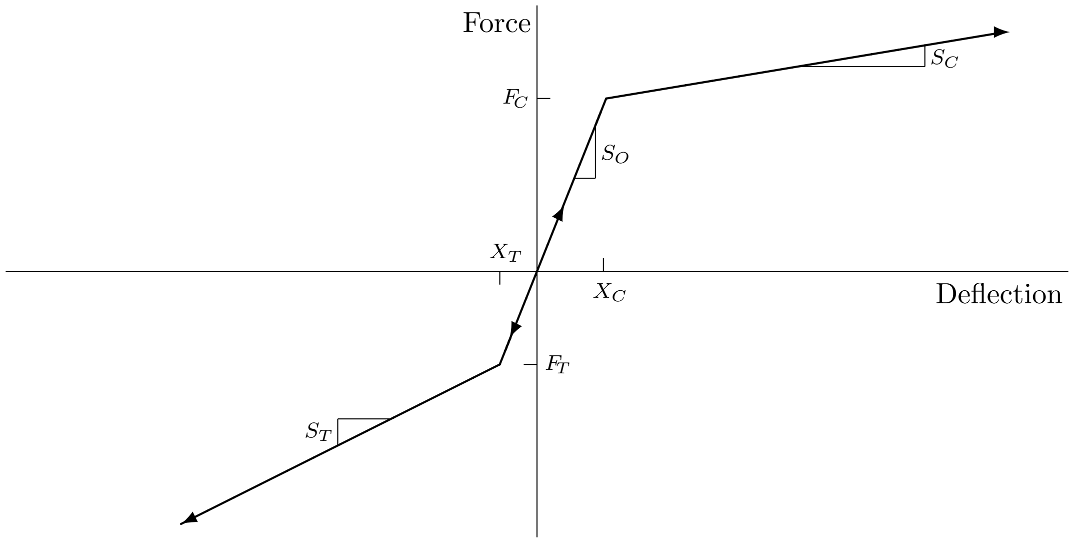
The **force** and **stiffness** are

$$f^S = \left(\frac{2}{\pi}F_{\text{Max}}\right)\text{Arctan}\left(\frac{\pi}{2}\frac{S_O}{F_{\text{Max}}}x\right) \qquad s^S = \frac{S_O}{1+\left(\frac{\pi}{2}\frac{S_O}{F_{\text{Max}}}x\right)^2}.$$

Static Type **AE** Parameters

Tag	Description	Value	Default	N	?
`SO`	Origin stiffness (S_O)	$\mathcal{R} \geq 0$			?
`FMax`	Asymptotic maximum force magnitude (F_{Max})	$\mathcal{R} \geq 0$			?

4.2.4 Static Type BE : Bilinear Elastic



Static type BE is bilinear elastic.

The origin line **force** and **stiffness** are

$$f^S_O = S_O\, x \qquad s^S_O = S_O\,.$$

The compression line **force** and **stiffness** are

$$f^S_C = F_C + [x - X_C]\, S_C \qquad s^S_C = S_C\,.$$

The tension line **force** and **stiffness** are

$$f^S_T = F_T + [x - X_T]\, S_T \qquad s^S_T = S_T\,.$$

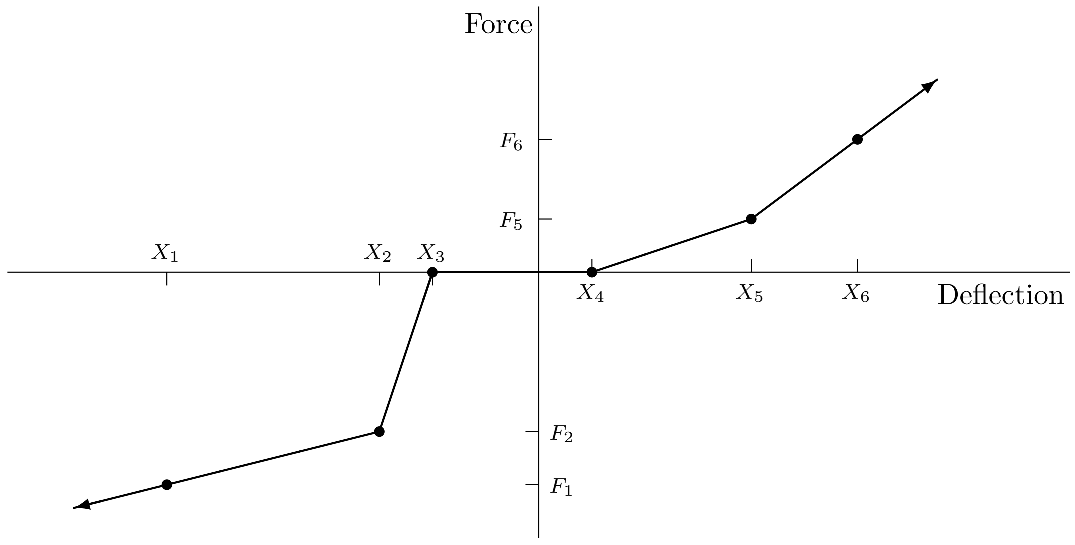
Static Type BE Parameters

Tag	Description	Value	Default	N	?
SO	Origin stiffness (S_O)	$\mathcal{R} \geq 0$			?
XC	Compression breakpoint deflection (X_C)	$\mathcal{R} \geq 0$	$-X_T$	N	?
FC	Compression breakpoint force (F_C)	$\mathcal{R} \geq 0$	$-F_T$	N	?
SC	Compression stiffness (S_C)	$\mathcal{R} \geq 0$	S_T	N	?
XT	Tension breakpoint deflection (X_T)	$\mathcal{R} \leq 0$	$-X_C$	N	?
FT	Tension breakpoint force (F_T)	$\mathcal{R} \leq 0$	$-F_C$	N	?
ST	Tension stiffness (S_T)	$\mathcal{R} \geq 0$	S_C	N	?

Static Type BE Notes

- Static BE parameters are related by $F_C = S_O X_C$ and $F_T = S_O X_T$, which imply:
  - $S_O = 0 \implies F_C = F_T = 0$.
  - $X_C = 0 \implies F_C = 0$ and $X_T = 0 \implies F_T = 0$.
  - $F_C = 0 \implies S_O = 0$ and/or $X_C = 0$ and $F_T = 0 \implies S_O = 0$ and/or $X_T = 0$.
- Breakpoints can be specified using either the deflection or force parameter but not both. Unspecified (implicit) breakpoint parameters are determined by the above relationships. If neither deflection nor force parameters are specified for a breakpoint they are determined by the defaults from the opposite breakpoint.
- Deflection parameters must be used if $S_O = 0$ is specified or to allow extraction of $S_O = 0$. Deflection parameters are also recommended if an extracted S_O may be small.
- Force parameters can be used with known breakpoint deflections by specifying `FC=SO*`X_C or `FT=SO*`X_T, or alternatively, `SO=FC*`$1/X_C$ or `SO=FT*`$1/X_T$.
- Implicit breakpoint parameters can be referenced if their implicit value is defined or is linear in the explicit extracted parameters.
- Breakpoint parameters without an explicit or implicit value are assigned the defaults shown.
- The compression line can be suppressed by using `XC=N` or `FC=N` or `SC=N`. The tension line can be suppressed using by `XC=N` or `FT=N` or `ST=N`. For BE aspects with extracted breakpoints, suppressing the compression or tension lines is more efficient than specifying `SC=SO` or `ST=SO`.
- $F_T \neq -F_C$ and/or $S_T \neq S_C$ can be used to model asymmetric compression–tension behavior.

4.2.5 Static Type SE : Segmented Elastic



Static type SE is segmented elastic with a force defined by linear segments between specified deflection–force points.

The **force** is defined by the points (X_i, F_i) for $i = 1, \ldots, N$ where X_i is increasing with i. The outermost segments are extrapolated to define the force at deflections outside the range of the points. **Preload** occurs if the force at $x = 0$ is nonzero.

Symmetric static type SE is a variant with the force for positive deflections defined by the points (X_i, F_i) where $X_i + X_\Delta \geq 0$ and the symmetric force for negative deflections defined by the points $(–X_i, –F_i)$.

Static Type SE Parameters

Tag	Description	Value	Default	N	?
`XiScl`	Deflection block scale factor ($X_{i\otimes}$)	$\mathcal{R} > 0$	1		
`FiScl`	Force block scale factor ($F_{i\otimes}$)	$\mathcal{R} \geq 0$	1		
`ConSS`	Segment smoothness confidence band factor	$\mathcal{R} > 0$	*hierarchy*	`N`	
`AnySlope`	Extracted segment slope control	$\mathcal{B}$	`False`		
`Symmetric`	Symmetric SE variant selector	$\mathcal{B}$	`False`		
`X`	Deflection (X_i) block	$\mathcal{R} \nearrow$			
`F`	Force (F_i) block	$\mathcal{R}$			?

Static Type SE Notes

- $X_{i\otimes}$ and $F_{i\otimes}$ can be used to scale the X_i and F_i block values before applying any spring transforms.
- *ConSS* is the confidence band factor for extracted segment smoothness target equations. These equations target no slope change between adjacent segments, so reducing *ConSS* tends to smooth the static characteristic. The segment smoothness confidence band is of the form *ConSS*$\cdot 5000(\tilde{w}/\tilde{x})$ where $\tilde{w}$ is the weight magnitude (in force units) and $\tilde{x}$ is a unit-invariant baseline deflection. *ConSS* defaults to the containing *Vehicle.ConSS*, *Model.ConSS*, and then 1. Segment smoothness targets can be suppressed by using `ConSS=N`. `ConSS` is suppressed in aspects defined by a `Ref` reference. *ConSS* is not affected by spring transforms.
- *AnySlope* controls the allowable sign of extracted segment slopes. A value of `True` allows unconstrained slope signs. By default the extracted segment (those with extracted forces at either end) slopes are constrained to be nonnegative. If `AnySlope=True` is used segment slopes can still be constrained to be nonnegative by using specifiers of the form `?( >F`*i*` )` for the entry for F_{i+1}, or `?`*n*`( >F`*i*`: )` for a section of n forces starting at F_{i+1}.
- An equal number of X_i and F_i values are specified following the `X` and `F` tags. These are block parameters and both single and block reference entries are supported (see Section 3.4.5).
- `X` block deflections can be selected automatically by entering `#` instead of a value. For springs between instrumented masses SISAME can use the actual deflection range to set automatic X_1 or X_N tail deflections. For symmetric SE aspects the automatic tail deflection values are $X_1 = 0$ and $X_N = \max(|x|)$. Sequences of automatic deflections are evenly spaced. Automatic deflection values are affected by spring transforms. The form `#`*n* can be used to indicate n successive automatic deflections.
- The form `?`*n* can be used to indicate n successive extracted forces.
- The (X_i, F_i) points are offset by (X_Δ, F_Δ). For symmetric SE aspects the $(-X_i, -F_i)$ negative deflection points are offset by $(-X_\Delta, -F_\Delta)$ to preserve symmetry.
- For symmetric SE aspects $X_1 + X_\Delta \geq 0$ is required. Note that if $X_1 + X_\Delta = 0$ but $F_1 + F_\Delta \neq 0$ there will be a force discontinuity at the origin. A segment smoothness target is applied to an extracted origin segment between $(X_1 + X_\Delta, F_1 + F_\Delta)$ and $(-(X_1 + X_\Delta), -(F_1 + F_\Delta))$ only if $X_1 + X_\Delta > 0$.
- Negative segment slopes are allowed although they deviate from idealized physical lumped-parameter behavior. They may be useful for modeling load-paths of greater size and/or complexity that are actually composites of multiple load-paths when using a coarse model discretization. The `AnySlope` parameter can be used to allow negative extracted segment slopes. Negative-sloped tail segments are extrapolated across the zero force line, which will give nonphysical behavior. SISAME issues a warning if negative slopes are obtained for an extracted tail segment.

4.3 Inelastic Static Aspects

SISAME provides three inelastic static aspect types. Inelastic static aspects provide a static force that is dependent on the current and past deflections (path-dependent).

Yielding (inelastic deformation) occurs during increasing deflection traversal of a compression boundary or decreasing deflection traversal of a tension boundary.

Unloading occurs when the deflection reverses direction while on a compression or tension boundary. The force follows a linear elastic unloading line from the last boundary point. The unloading path, consisting of the unloading line and any slack or reloading (in the opposite direction) line behaves linear elastically and is traversed until a boundary is reached or a new domain entered (such as one-way clipping at zero force).

For bidirectionally yielding aspects the opposite boundary shifts as necessary so that the unloading path intersects it: the compression boundary shifts by negative deflections and the tension boundary shifts by positive deflections. The boundary shifts create an expanding energy envelope.

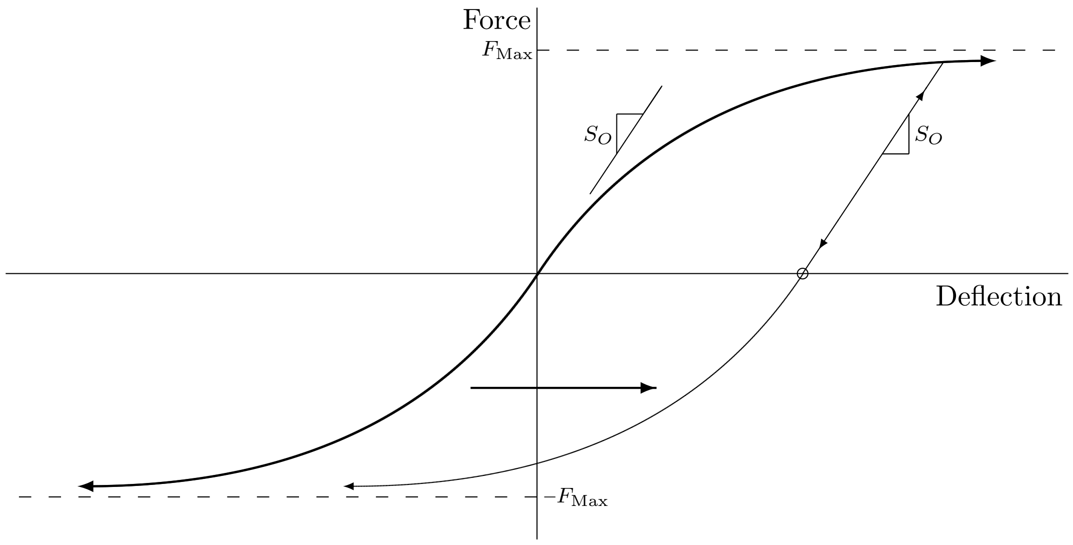
One-way aspects do not yield (thus the unloading path and opposite boundary do not shift) while the force is clipped at zero.

Energy is dissipated through cyclic hysteresis during clockwise traversals in the deflection–force plane.

If the initial deflection–force point differs from the aspect's natural origin the aspect behaves as if the spring's deflection had started at the origin and moved continuously to the initial deflection. (Some assumption is necessary as these aspects are path-dependent.)

The properties and parameters for each inelastic static aspect type are described below.

4.3.1 Static Type **AI** : Arctangent Inelastic



Static type AI is arctangent inelastic with **origin and unloading stiffness** S_O and asymptotic **maximum force** magnitude F_{Max}.

The initial **boundary force** and **stiffness** are

$$f^S = \left(\frac{2}{\pi}\,F_{\text{Max}}\right)\text{Arctan}\left(\frac{\pi}{2}\frac{S_O}{F_{\text{Max}}}\,x\right) \qquad s^S = \frac{S_O}{1+\left(\frac{\pi}{2}\frac{S_O}{F_{\text{Max}}}\,x\right)^2}\,.$$

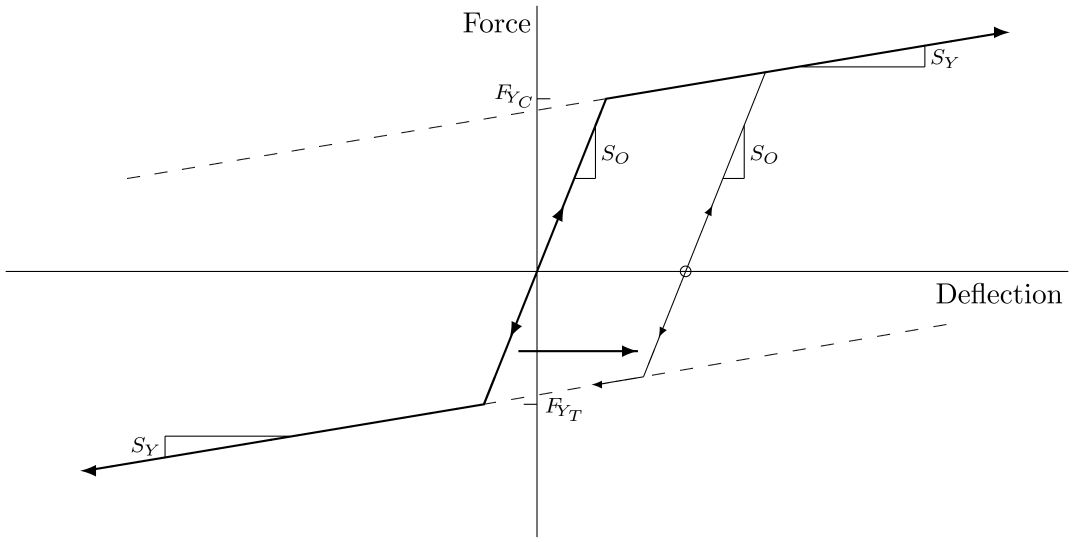
Yielding occurs during traversal of the compression or tension boundaries. The opposite boundary shifts as necessary so that the unloading line intersects it.

Unloading traverses an elastic line with slope S_O from the last boundary point until either boundary is reached.

Static Type **AI** Parameters

Tag	Description	Value	Default	N	?
`SO`	Origin and unloading stiffness (S_O)	$\mathcal{R} \geq 0$			?
`FMax`	Asymptotic maximum force magnitude (F_{Max})	$\mathcal{R} \geq 0$			?

4.3.2 Static Type BI : Bilinear Inelastic



Static type BI is bilinear inelastic with **origin and unloading stiffness** S_O, initial **compression yield force** F_{Y_C}, initial **tension yield force** F_{Y_T}, and **yielding stiffness** of S_Y.

The origin line **force** and **stiffness** are

$$f^S_O \;=\; S_O\,x \qquad\qquad s^S_O \;=\; S_O\,.$$

The compression boundary **force** and **stiffness** are

$$f^S_C \;=\; F_{Y_C} \;+\; [x - X_{Y_C}]\,S_Y \qquad\qquad s^S_C \;=\; S_Y\,.$$

The tension boundary **force** and **stiffness** are

$$f^S_T \;=\; F_{Y_T} \;+\; [x - X_{Y_T}]\,S_Y \qquad\qquad s^S_T \;=\; S_Y\,.$$

Yielding occurs during traversal of the compression or tension boundaries. The opposite boundary shifts as necessary so that the unloading line intersects it.

Unloading traverses an elastic line with slope S_O from the last boundary point until either boundary is reached.

Static Type BI Parameters

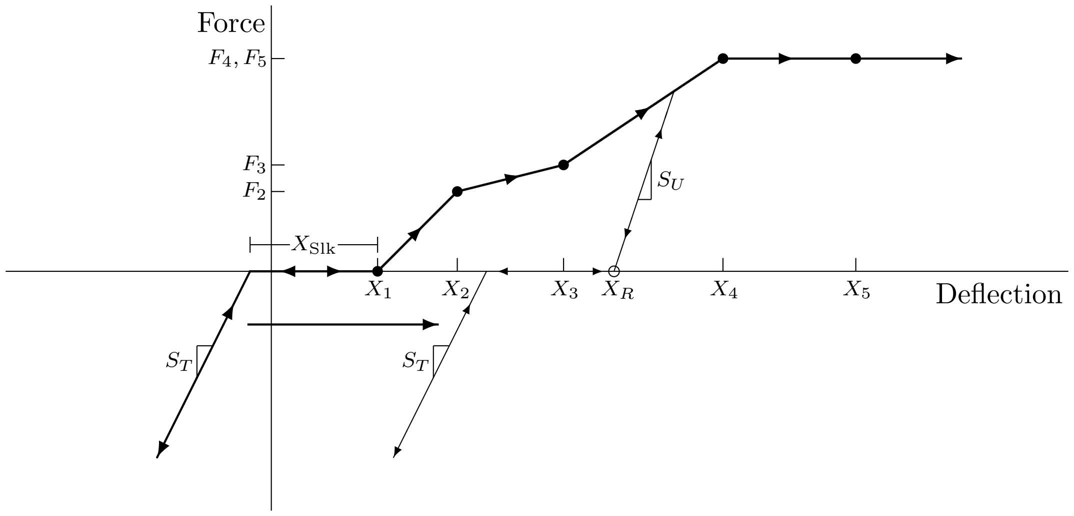
Tag	Description	Value	Default	N	?
SO	Origin and unloading stiffness (S_O)	$\mathcal{R} \geq S_Y$			?
FYC	Compression yield force (F_{Y_C})	$\mathcal{R} \geq 0$	$-F_{Y_T}$	N	?
FYT	Tension yield force (F_{Y_T})	$\mathcal{R} \leq 0$	$-F_{Y_C}$	N	?
SY	Yield stiffness (S_Y)	$S_O \geq \mathcal{R} \geq 0$		N	?

Static Type BI Notes

- If *SO* = 0 the force is constant for all deflections and the other BI parameters are inactive and are suppressed by default.
- Yield deflection values, X_{Y_C} or X_{Y_T}, can be implicitly defined by specifying FYC=SO*X_{Y_C} or FYT=SO*X_{Y_T}.
- Yielding in either direction can be suppressed using FYC=N or FYT=N.
- Specifying SY=N suppresses yielding in both directions (giving linear elastic behavior) and suppresses the inactive F_{Y_C} and F_{Y_T} by default.
- In One-way BI aspects with $S_Y > 0$ both F_{Y_C} and F_{Y_T} can contribute to the behavior at large deflections where unloading may reach the opposite boundary before reaching zero force. Suppressing the opposite boundary (using FYT=N or FYC=N) assures unloading continuously to zero force from all deflections.
- $F_{Y_T} \neq -F_{Y_C}$ can be used to model asymmetric compression–tension behavior.

4.3.3 Static Type SI : Segmented Inelastic

Nonsymmetric Static Type SI



Nonsymmetric static type SI is segmented inelastic with a compression boundary defined by linear segments between specified deflection–force points.

The **compression boundary** is defined by the points (X_i, F_i) for $i = 1, \ldots, N$ where X_i is increasing with i and $F_i + F_\Delta \geq 0$. The last compression segment is extrapolated to define the boundary for $x > X_N + X_\Delta$.

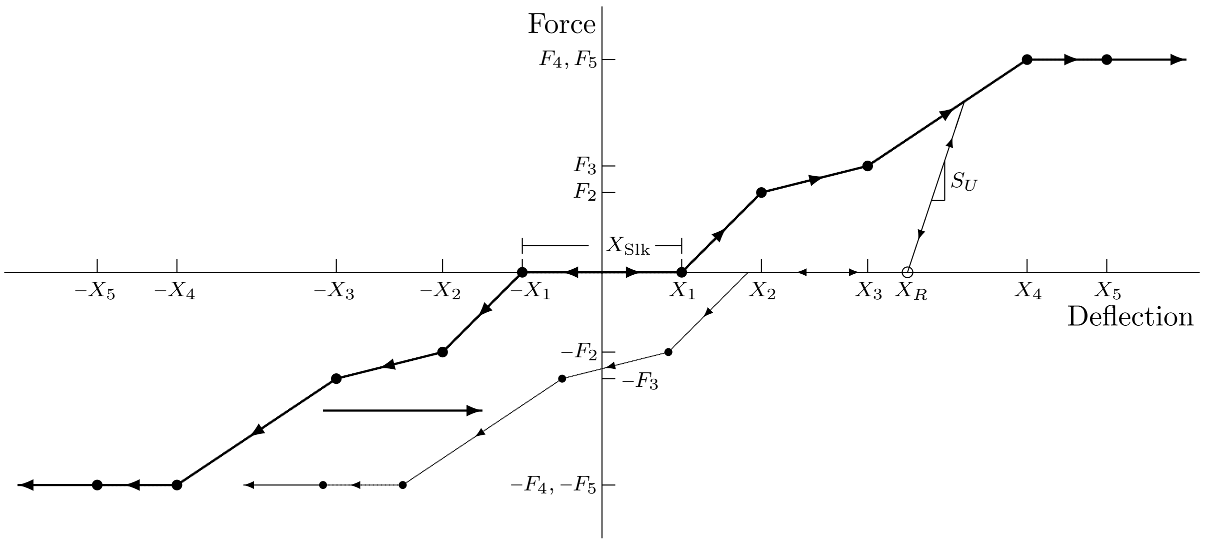
Yielding occurs during increasing deflection traversal of the compression boundary.

Unloading traverses an elastic line with slope S_U from the last boundary point down to zero force. If the deflection increases again to the last boundary deflection yielding continues on the compression boundary.

Slack (zero force) occurs for a distance of X_{Slk} below the point, X_R, where the unloading line intersects the zero force axis.

Tension traverses an elastic line with slope S_T for all deflections below $X_R - X_{\text{Slk}}$.

Symmetric Static Type SI



Symmetric static type SI is segmented inelastic with compression and tension boundaries defined by linear segments between deflection–force points.

The initial **compression boundary** is defined by the points (X_i, F_i) for $i = 1, \ldots, N$ where X_i is increasing with i, $X_i + X_\Delta \geq 0$, and $F_i + F_\Delta \geq 0$. The symmetric initial **tension boundary** is defined by the points $(-X_i, -F_i)$. The last segments are extrapolated to define the boundary for deflections outside the boundary point range.

Yielding occurs during traversal of the compression or tension boundaries. The opposite boundary shifts as necessary so that the unloading path intersects it.

Unloading traverses an elastic line with slope S_U from the last boundary point to zero force. If the deflection again reaches the last boundary deflection yielding continues on the boundary.

Slack (zero force) occurs for the (implied) slack distance, X_{Slk}, between the point, X_R, where the current unloading line intersects the zero force axis and the point where **reloading** towards the opposite boundary along a line with slope S_U begins.

Static Type SI Parameters

Tag	Description	Value	Default	N	?
`SU`	Unloading slope (S_U)	$\mathcal{R} \geq 0$			?
`ST`	Tension slope (S_T) for nonsymmetric SI	$\mathcal{R} \geq 0$	S_U		?
`XSlk`	Slack (X_{Slk}) for nonsymmetric SI	$\mathcal{R} \geq 0$	0		?
`XiScl`	Deflection block scale factor ($X_{i\otimes}$)	$\mathcal{R} > 0$	1		
`FiScl`	Force block scale factor ($F_{i\otimes}$)	$\mathcal{R} \geq 0$	1		
`ConSS`	Segment smoothness confidence band factor	$\mathcal{R} > 0$	*hierarchy*	N	
`AnySlope`	Extracted segment slope control	$\mathcal{B}$	`False`		
`Symmetric`	Symmetric SI variant selector	$\mathcal{B}$	`False`		
`X`	Boundary deflection (X_i) block	$\mathcal{R} \nearrow$			
`F`	Boundary force (F_i) block	$\mathcal{R}$			?

Static Type SI Notes

- If S_U is less than any of the boundary segment slopes it is possible for energy to be created in the deflection–force plane if the unloading line crosses outside of the boundary. SISAME issues a warning if an SI aspect is defined with this property. Extracted S_U values are constrained to be at least as large as any segment slope.
- S_T applies only to nonsymmetric SI aspects. For symmetric SI aspects the unloading slope, S_U, is used for unloading/reloading in both compression and tension.
- X_{Slk} can be specified only for nonsymmetric SI aspects. For symmetric SI aspects the **implied slack** value is $\max(2(X_1 - ({}^{F_1}\!/_{S_U})), 0)$.
- $X_{i\otimes}$ and $F_{i\otimes}$ can be used to scale the X_i and F_i block values before applying any spring transforms and without affecting other SI parameters.
- ***ConSS*** is the confidence band factor for extracted segment smoothness target equations. These equations target no slope change between adjacent segments, so reducing ***ConSS*** tends to smooth the static characteristic. The segment smoothness confidence band is of the form ***ConSS***$\cdot 5000({}^{\tilde{w}}\!/_{\tilde{x}})$ where $\tilde{w}$ is the weight magnitude (in force units) and $\tilde{x}$ is a unit-invariant baseline deflection. ***ConSS*** defaults to the containing *Vehicle*.***ConSS***, *Model*.***ConSS***, and then 1. Segment smoothness targets can be suppressed by using `ConSS=N`. `ConSS` is suppressed in aspects defined by a `Ref` reference. ***ConSS*** is not affected by spring transforms.
- ***AnySlope*** controls the allowable sign of extracted segment slopes. A value of `True` allows unconstrained slope signs. By default the extracted segment (those with extracted forces at either end) slopes are constrained to be nonnegative. If `AnySlope=True` is used segment slopes can still be constrained to be nonnegative by using specifiers of the form `?( >F`*i* `)` for the entry for F_{i+1}, or `?`*n*`( >F`*i*`: )` for a section of n forces starting at F_{i+1}.
- An equal number of X_i and F_i values are specified following the `X` and `F` tags. These are block parameters and both single and block reference entries are supported (see Section 3.4.5).

- `X` block deflections can be selected automatically by entering `#` instead of a value. For springs between instrumented masses SISAME can use the actual deflection range to set automatic X_1 or X_N tail deflections. For symmetric SI aspects the automatic tail deflection values are $X_1 = 0$ and $X_N = \max(|x|)$. Sequences of automatic deflections are evenly spaced. Automatic deflection values are affected by spring transforms. The form `#`n can be used to indicate n successive automatic deflections.
- $F_i + F_\Delta \geq 0$ is required for defined and extracted forces.
- Extracted forces are constrained to keep segment slopes below S_U.
- The form `?`n can be used to indicate n successive extracted forces.
- The (X_i, F_i) compression boundary points are offset by (X_Δ, F_Δ). For symmetric SI aspects the $(-X_i, -F_i)$ tension boundary points are offset by $(-X_\Delta, -F_\Delta)$ to preserve symmetry.
- If $F_1 + F_\Delta > 0$ the unloading line from $(\hat{X}_1 + X_\Delta, F_1 + F_\Delta)$, where $\hat{X}_1$ is the current shifted compression X_1 value, is traversed between the zero force axis and the compression boundary when $X_R < x < \hat{X}_1$. For symmetric SI aspects the tension boundary is treated symmetrically. (In the degenerate $S_U = 0$ case the transition to the compression boundary occurs discontinuously at $\hat{X}_1 + X_\Delta$ and subsequent unloading occurs on a zero slope line.)
- Negative segment slopes are allowed although they deviate from idealized physical lumped-parameter behavior. They may be useful for modeling load-paths of greater size and/or complexity that are actually composites of multiple load-paths when using a coarse model discretization. And a steep negative slope may be appropriate to represent a buckling event. The `AnySlope` parameter can be used to allow negative extracted segment slopes. Negative-sloped last segments are extrapolated to their zero force crossing and remain at zero force beyond that deflection (yielding continues while at zero force). SISAME issues a warning if negative slopes are obtained for an extracted last segment.

5 Dynamic Aspects

SISAME provides four spring dynamic aspect types, including two damper and two magnifier types. Any dynamic aspect type can be combined with any static aspect type in a SISAME spring element.

The two damper types provide a force that is a function of the spring's relative velocity. The damping force is independent of the associated static force and can be used with no static aspect to obtain a pure damping element.

The two magnifier types scale the static force by a magnification function of the spring's relative velocity.

A compression–positive sign convention is used for the dynamic aspects.

The dampers are described in Section 5.2 and the magnifiers are described in Section 5.3.

5.1 Universal Dynamic Parameters

A number of universal dynamic parameters are available for all of the dynamic aspect types. These parameters can be included anywhere within the dynamic aspect specifications (after the `DynType` field). The effects of these parameters are not illustrated in the dynamic aspect figures that follow.

The `OneWay` parameter allows specification of compression-only or tension-only dynamic aspects. A compression-only dynamic aspect (`OneWay=C`) is inactive when the relative velocity is negative. A tension-only dynamic aspect (`OneWay=T`) is inactive when the relative velocity is positive.

Dynamic transforms provide an easy way to modify dynamic aspect characteristics. **Scale factor** transforms $X_{\otimes}$, $R_{\otimes}$, $D_{\otimes}$, and $M_{\otimes}$ scale the dynamic characteristic along the corresponding axes.

Universal Dynamic Parameters

Tag	Description	Value	Default	N	?
`OneWay`	One-way dynamic aspect flag	`C` *or* `T`	*none*	`N`	
`XScl`	Deflection scale factor ($X_\otimes$)	$\mathcal{R} > 0$	1		
`RScl`	Relative velocity scale factor ($R_\otimes$)	$\mathcal{R} > 0$	1		
`DScl`	Damping force scale factor ($D_\otimes$)	$\mathcal{R} \geq 0$	1		
`MScl`	Dynamic magnifier scale factor ($M_\otimes$)	$\mathcal{R} \geq 0$	1		

Universal Dynamic Parameter Notes

- Transforms are applied to the affected **transforming parameters** with defined values or referencing parameters in other springs. Transforms are *not* applied to parameters specified as extracted (`*Tag*=?( *Specifiers* )`) or to parameters referencing other parameters in the same spring.
- Scale factor transforms are incorporated into the value of affected transforming parameters and do not appear in the output model file.
- Transforms are incorporated before references to a transforming parameter are resolved.
- $M_\otimes$ scales the parametric contribution to the dynamic magnification that is added to one to get the magnification: M_{Slp} and, for dynamic type AM, $(M_{\text{Max}} - 1)$ are scaled by $M_\otimes$.
- $X_\otimes$ affects only the damping slack deflection, D_{Slk}.
- References to the dynamic scale factors should include the `*DynType*.` portion to distinguish them from references to corresponding spring-level scale factors.

5.2 Dampers

Dampers generate a force that is a function of a spring's relative velocity. Dampers act independently of and in parallel with a spring's static aspect, if any.

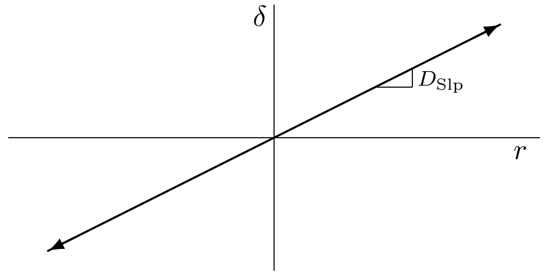
A damper slack, D_{Slk}, can be specified to prevent damping at deflections less than D_{Slk}.

A compression-only (`OneWay=C`) damper generates zero force at negative relative velocities. A tension-only (`OneWay=T`) damper generates zero force at positive relative velocities.

The damping function and a picture of the damper characteristic are shown below for both damper types. Both of the damping functions are of the form $f = f^s + \delta(r)$.

5.2.1 Dynamic Type **LD** : Linear Damper

$$f \;=\; f^S \;+\; D_{\text{Slp}}\, r$$

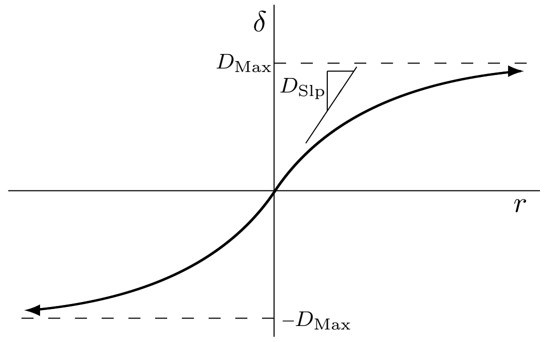


Dynamic Type LD Parameters

Tag	Description	Value	Default	N	?
`DSlp`	Damping slope (D_{Slp})	$\mathcal{R} \geq 0$			?
`DSlk`	Damping slack (D_{Slk})	$\mathcal{R}$	*none*	`N`	

5.2.2 Dynamic Type **AD** : Arctangent Damper

$$f \;=\; f^S \;+\; \left(\tfrac{2}{\pi} D_{\text{Max}}\right) \text{Arctan}\!\left(\tfrac{\pi}{2} \frac{D_{\text{Slp}}}{D_{\text{Max}}} r\right)$$



Dynamic Type AD Parameters

Tag	Description	Value	Default	N	?
`DSlp`	Damping slope at $r = 0$ (D_{Slp})	$\mathcal{R} \geq 0$			?
`DMax`	Maximum damping force magnitude (D_{Max})	$\mathcal{R} \geq 0$			?
`DSlk`	Damping slack (D_{Slk})	$\mathcal{R}$	*none*	`N`	

5.3 Magnifiers

Magnifiers scale the static force based on a function of the relative velocity. This is used to approximately compensate for various inertial lag strain rate effects. These effects arise from the model discretization error (the actual load-paths are not massless) and from using one-dimensional models that do not capture transverse motions [12]. Transverse motions are particularly significant during buckling/collapse events.

The magnifiers appear complicated because of the $\sigma(r f^S)$ exponent ($\sigma(\cdot)$ is the sign function). The exponent is applied to a term that is at least 1 and depends only on the relative velocity magnitude. If the static force and the relative velocity have the same sign, then the magnifier effect increases the force magnitude, and so the static force is multiplied by the term. If they have opposite signs then the magnifier effect decreases the force magnitude, and so the static force is divided by the term. This is consistent with the intuition that the dynamic effects should resist the motion and dissipate energy.

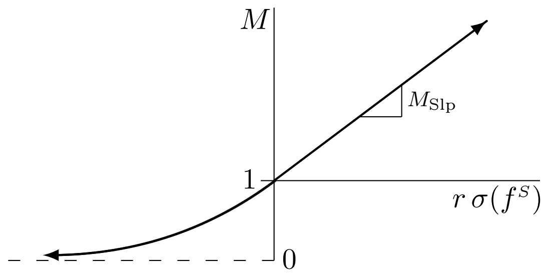
A compression-only (`OneWay=C`) magnifier has a factor of one at negative relative velocities. A tension-only (`OneWay=T`) magnifier has a factor of one at positive relative velocities.

Applying a magnifier to extracted static aspects which may have zero force crossings determined by extracted parameters requires that SISAME iterate for the model, since the sign of the force affects the magnification factor, and may hamper the solution convergence. Reducing the ***`ConPC`***, ***`ConPD`***, ***`MultPD`***, and/or (for segmented static aspects) ***`ConSS`*** values may improve convergence in such cases.

The magnifier functions and picture are shown below for both magnifier types. Both of the magnifiers are of the form $f = M(r, \sigma(f^S))\, f^S$.

5.3.1 Dynamic Type LM : Linear Magnifier

$$f \;=\; \left[1 + M_{\mathrm{Slp}}\,|r|\,\right]^{\sigma(r f^S)}\, f^S$$

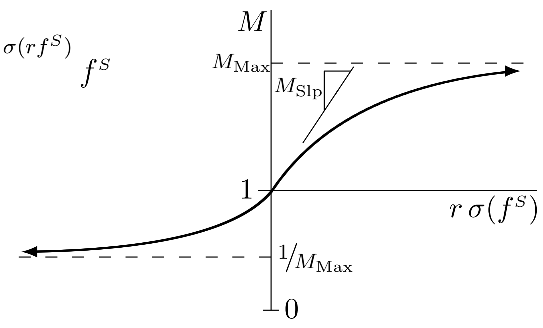


Dynamic Type LM Parameters

Tag	Description	Value	Default	N	?
`MSlp`	Magnifier slope (M_{Slp})	$\mathcal{R} \geq 0$			?

5.3.2 Dynamic Type AM : Arctangent Magnifier

$$f \;=\; \left[1 + \left(\tfrac{2}{\pi}(M_{\mathrm{Max}} - 1)\right) \mathrm{Arctan}\!\left(\tfrac{\pi}{2}\,\tfrac{M_{\mathrm{Slp}}}{M_{\mathrm{Max}}-1}\,|r|\,\right)\right]^{\sigma(r f^S)}\, f^S$$



Dynamic Type AM Parameters

Tag	Description	Value	Default	N	?
`MSlp`	Magnifier slope at $r = 0$ (M_{Slp})	$\mathcal{R} \geq 0$			?
`MMax`	Maximum magnification factor (M_{Max})	$\mathcal{R} \geq 1$			?

6 SISAME Formulation

The key theoretical and algorithmic aspects of the SISAME methodology are presented below. First, the target equations that form the basis of the model extraction are presented. This is followed by a description of the construction and solution of the least squares problem for the optimal model. Finally, some algorithmic aspects of the computations are described.

6.1 Partitioned Equations of Motion

SISAME uses partitioned equations of motion that distinguish **simulated masses** from **instrumented masses**, and **defined springs** from **extracted springs** as the basis for simulation and to construct the mass target equations. This section presents these partitioned equations of motion and the mass and force target equations.

Consider a SISAME model with M masses and S springs. At any instant, the spring force, $\mathbf{f}$, is related to the force exerted on the masses, $\tilde{\mathbf{g}}$, by

$$\mathbf{\Phi}\,\mathbf{f} + \boldsymbol{f}^d \;=\; \tilde{\mathbf{g}} \tag{6.1}$$

where $\mathbf{\Phi}$ is the $M{\times}S$ spring force mass influence matrix (determined by the configuration of the model), and $\boldsymbol{f}^d$ is the vector of driving forces on the masses. For the one-dimensional models used in SISAME, $\mathbf{\Phi}$ is constant with entries of 0, 1, and –1.

The instantaneous spring force is related to the static spring force by

$$\mathbf{f} \;=\; \mathbf{M}\,\mathbf{f}^S + \boldsymbol{\delta} \tag{6.2}$$

where $\mathbf{M}$ is the diagonal matrix of dynamic magnifiers, $\mathbf{f}^S$ is the static spring force vector, and $\boldsymbol{\delta}$ is the damping force vector.

The partitioned version of (6.1) is

$$\begin{bmatrix} \mathbf{\Phi}_{SD} & \mathbf{\Phi}_{SE} \\ \mathbf{\Phi}_{ID} & \mathbf{\Phi}_{IE} \end{bmatrix} \begin{pmatrix} \mathbf{f}_D \\ \mathbf{f}_E \end{pmatrix} + \begin{pmatrix} \boldsymbol{f}_S^d \\ \boldsymbol{f}_I^d \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{g}}_S \\ \tilde{\mathbf{g}}_I \end{pmatrix} \tag{6.3}$$

where the upper portion corresponds to the simulated masses ($\Box_S$) and the lower portion to the instrumented masses ($\Box_I$), and the left half of the $\mathbf{\Phi}$ matrix corresponds to the defined springs ($\Box_D$) and the right half to the extracted springs ($\Box_E$). This subscript notation is followed below.

For simulated masses, the inertia force equals the force exerted by the springs, or

$$\mathbf{g}_S \;=\; \tilde{\mathbf{g}}_S \;=\; \mathbf{\Phi}_{SD}\,\mathbf{f}_D \,+\, \mathbf{\Phi}_{SE}\,\mathbf{f}_E \,+\, \boldsymbol{f}_S^d\,. \tag{6.4}$$

Using $\mathbf{W}_S\,\mathbf{a}_S = \mathbf{g}_S$, where $\mathbf{W}_S$ is the diagonal matrix of simulated mass weights (which cannot be extracted), gives

$$\mathbf{a}_S \;=\; \mathbf{W}_S^{-1}\left(\mathbf{\Phi}_{SD}\,\mathbf{f}_D \,+\, \mathbf{\Phi}_{SE}\,\mathbf{f}_E \,+\, \boldsymbol{f}_S^d\right). \tag{6.5}$$

If the right hand side is known at each instant, the simulated mass motions can be numerically integrated and, since the instrumented mass motions are known, the defined spring forces can be computed at each time step during the extraction. The global approach used in SISAME does not produce the force in the extracted springs, $\mathbf{f}_E$, until after the simulation is complete, so $\mathbf{\Phi}_{SE}=\mathbf{0}$ is required. Thus simulated masses cannot be connected to extracted springs. The simulated mass motions are then obtained by integrating

$$\mathbf{a}_S \;=\; \mathbf{W}_S^{-1}\left(\mathbf{\Phi}_{SD}\,\mathbf{f}_D \,+\, \boldsymbol{f}_S^d\right). \tag{6.6}$$

SISAME treats the simulated portion of an extraction model precisely as it treats a pure simulation model.

Instrumented mass motions are simply computed at each instant from a filtered representation of the supplied motion data. The set of instrumented masses is made up of driven and target masses. For the driven masses no further computations are required.

The instrumented masses are either target or driven masses, and any instrumented mass may also be an extracted-weight mass. Let the target masses be denoted by ($\Box_T$) and the extracted-weight masses (which may be target or driven masses) by ($\Box_E$). Suppose there are M_T target masses and M_E extracted-weight masses, and let $\mathbf{w}_E$ be the extracted weight vector. For target masses, at each instant the inertia force, $\mathbf{g}_T$, is a sum of contributions from defined-weight and extracted-weight masses

$$\mathbf{g}_T \;=\; \bar{\mathbf{g}}_T \,+\, \mathbf{A}_{TE}\,\mathbf{w}_E \tag{6.7}$$

where $\bar{\mathbf{g}}_T$ is the defined-weight mass inertia vector, with zero entries for extracted-weight target masses, and $\mathbf{A}_{TE}$ is the $M_T \times M_E$ extracted-weight target mass influence matrix, with a single nonzero acceleration entry in each row corresponding to an extracted-weight target mass.

The extraction process attempts to approximately satisfy the equations of motion for the target masses, that is, to best satisfy $\tilde{\mathbf{g}}_T \approx \mathbf{g}_T$, or

$$\mathbf{\Phi}_{TD}\,\mathbf{f}_D \,+\, \mathbf{\Phi}_{TE}\,\mathbf{f}_E \,+\, \boldsymbol{f}_T^d \;\approx\; \bar{\mathbf{g}}_T \,+\, \mathbf{A}_{TE}\,\mathbf{w}_E \tag{6.8}$$

or

$$\mathbf{\Phi}_{TE}\,\mathbf{f}_E \,-\, \mathbf{A}_{TE}\,\mathbf{w}_E \;\approx\; \bar{\mathbf{g}}_T \,-\, \mathbf{\Phi}_{TD}\,\mathbf{f}_D \,-\, \boldsymbol{f}_T^d \tag{6.9}$$

where the defined spring force $\mathbf{f}_D = \mathbf{M}_D\,\mathbf{f}_D^S + \boldsymbol{\delta}_D$ and the other elements of the right side are known at each instant.

If $[\,\boldsymbol{\Phi}_{TE}\ \ \text{-}\mathbf{A}_{TE}\,]$ is square (*i.e.*, $S_E = M_T$) and nonsingular this system can be solved at each instant for $\mathbf{f}_E$ and $\mathbf{w}_E$. Such a determinate extraction (without extracted weights) was used in the Fiat methodology [2, 3, 6] and in the preliminary version of SISAME [7]. The restriction to models with only as many extracted springs as target masses does not allow the extraction of multiple load-path systems without additional defined springs. Also, the resulting spring deflection–force paths may not correspond to physically meaningful load-paths. The one advantage to the determinate approach is that $\boldsymbol{\Phi}_{SE} = \mathbf{0}$ is not required.

To allow nonsquare (underdetermined) models with more extracted load-paths than target masses to be extracted, SISAME combines the target equations from the entire event into a single overdetermined system that can then be used to find the optimal model. First the springs are parameterized to avoid a huge number of unknowns and constrained to physically meaningful behavior.

6.2 Mass Target Equations

At each instant, the extracted spring force $\mathbf{f}_E = \mathbf{M}_E\,\mathbf{f}_E^S + \boldsymbol{\delta}_E$ is unknown. In general, the extracted spring force is the sum of a portion that depends on unknown spring parameters and a known value, $\bar{\mathbf{f}}_E = \mathbf{M}_E\,\bar{\mathbf{f}}_E^S + \bar{\boldsymbol{\delta}}_E$.

To assure a tractable approximation problem, the springs are parameterized so that the force is linear in the parameters. Suppose there are P_S spring parameters and $P = P_S + M_E$ parameters total. The instantaneous extracted spring force can be represented in the form

$$\mathbf{f}_E \;=\; \boldsymbol{\Pi}_E\,\mathbf{p}_S \;+\; \bar{\mathbf{f}}_E \tag{6.10}$$

where $\boldsymbol{\Pi}_E$ is the $S_E \times P_S$ extracted spring parameter force influence matrix, and $\mathbf{p}_S$ is the unknown spring parameter vector. $\boldsymbol{\Pi}_E$ is known since all extracted springs are connected to instrumented masses ($\boldsymbol{\Phi}_{SE} = \mathbf{0}$).

Expressing the mass target equations in terms of the parameters, the instantaneous target system (6.9) becomes

$$\boldsymbol{\Phi}_{TE}\,\boldsymbol{\Pi}_E\,\mathbf{p}_S \;-\; \mathbf{A}_{TE}\,\mathbf{w}_E \;\;\approx\;\; \bar{\mathbf{g}}_T \;-\; \boldsymbol{\Phi}_{TD}\,\mathbf{f}_D \;-\; \boldsymbol{\Phi}_{TE}\,\bar{\mathbf{f}}_E \;-\; \boldsymbol{f}_T^d\,. \tag{6.11}$$

Letting

$$\boldsymbol{\Gamma}_M \;=\; \begin{bmatrix} \boldsymbol{\Phi}_{TE}\,\boldsymbol{\Pi}_E & \text{-}\mathbf{A}_{TE} \end{bmatrix} \quad \text{and} \quad \mathbf{p} \;=\; \begin{pmatrix} \mathbf{p}_S \\ \mathbf{w}_E \end{pmatrix} \quad \text{and} \quad \boldsymbol{\gamma}_M \;=\; \bar{\mathbf{g}}_T - \boldsymbol{\Phi}_{TD}\,\mathbf{f}_D - \boldsymbol{\Phi}_{TE}\,\bar{\mathbf{f}}_E - \boldsymbol{f}_T^d$$

where the dimensions of $\mathbf{\Gamma}_{M}$ are $M_{\mathcal{T}} \times P$, the instantaneous mass target equations become

$$\mathbf{\Gamma}_{M}\,\mathbf{p} \;\approx\; \boldsymbol{\gamma}_{M}\,. \tag{6.12}$$

A solution, $\mathbf{p}$, that provides a good fit to the (acceleration domain) mass target equations does not assure that the resulting model's simulations will not drift in the velocity and displacement domains, since the acceleration errors may not have near-zero first and second integrals. The model's simulation accuracy can be improved by including target systems for the velocity and displacement domains. This is achieved by target equations that are the first and second integral over time of (6.11)

$$\mathbf{\Phi}_{TE}\left[\int \mathbf{\Pi}_{E}\,d\tau\right]\mathbf{p}_{S} \;-\; \left[\int \mathbf{A}_{TE}\,d\tau\right]\mathbf{w}_{E} \;\approx\; \int\left(\bar{\mathbf{g}}_{T} \;-\; \mathbf{\Phi}_{TD}\,\mathbf{f}_{D} \;-\; \mathbf{\Phi}_{TE}\,\bar{\mathbf{f}}_{E} \;-\; \boldsymbol{f}_{T}^{d}\right)d\tau \tag{6.13}$$

$$\mathbf{\Phi}_{TE}\left[\iint \mathbf{\Pi}_{E}\,d\tau\,dv\right]\mathbf{p}_{S} - \left[\iint \mathbf{A}_{TE}\,d\tau\,dv\right]\mathbf{w}_{E} \;\approx\; \iint\left(\bar{\mathbf{g}}_{T} - \mathbf{\Phi}_{TD}\,\mathbf{f}_{D} - \mathbf{\Phi}_{TE}\,\bar{\mathbf{f}}_{E} - \boldsymbol{f}_{T}^{d}\right)d\tau\,dv\,. \tag{6.14}$$

These are actually in units of impulse and the integral of impulse target systems since the extracted weights, $\mathbf{w}_{E}$, cannot be factored out. Extending (6.12) with these target equations gives the full mass target system

$$\hat{\mathbf{\Gamma}}_{M}\,\mathbf{p} \;\approx\; \hat{\boldsymbol{\gamma}}_{M} \tag{6.15}$$

where the dimensions of $\hat{\mathbf{\Gamma}}_{M}$ are $3M_{\mathcal{T}} \times P$.

6.3 Force Target Equations

Additional target equations are created for each target force at each instant. Let the target forces be denoted by $(\square_{\mathcal{T}})$. Suppose there are $F_{\mathcal{T}}$ target forces in the model. Let $\mathbf{f}_{\mathcal{T}}$ be the instantaneous target force vector. The associated target system matches this against the sums of spring and mass inertia forces corresponding to the target force elements. The instantaneous force target equations can then be expressed as

$$\mathbf{\Pi}_{\mathcal{T}}\,\mathbf{p}_{S} \;+\; \mathbf{A}_{\mathcal{T}E}\,\mathbf{w}_{E} \;\approx\; \mathbf{f}_{\mathcal{T}} \;-\; \bar{\mathbf{f}}_{\mathcal{T}} \;-\; \bar{\mathbf{g}}_{\mathcal{T}} \tag{6.16}$$

where $\mathbf{\Pi}_{\mathcal{T}}$ is the $F_{\mathcal{T}} \times P_{S}$ extracted spring target force parameter influence matrix, $\mathbf{A}_{\mathcal{T}E}$ is the $F_{\mathcal{T}} \times M_{E}$ extracted weight target force influence matrix, with signed acceleration entries corresponding to the extracted-weight masses in the target forces, $\bar{\mathbf{f}}_{\mathcal{T}}$ is the vector of known (defined or extracted) spring force contributions to the target forces, and $\bar{\mathbf{g}}_{\mathcal{T}}$ is the vector of defined-weight mass inertia force contributions to the target forces.

Letting

$$\mathbf{\Gamma}_{F} \;=\; \begin{bmatrix}\mathbf{\Pi}_{\mathcal{T}} & \mathbf{A}_{\mathcal{T}E}\end{bmatrix} \qquad \text{and} \qquad \mathbf{p} \;=\; \begin{pmatrix}\mathbf{p}_{S}\\ \mathbf{w}_{E}\end{pmatrix} \qquad \text{and} \qquad \boldsymbol{\gamma}_{F} \;=\; \mathbf{f}_{\mathcal{T}} \;-\; \bar{\mathbf{f}}_{\mathcal{T}} \;-\; \bar{\mathbf{g}}_{\mathcal{T}}$$

where the dimensions of $\boldsymbol{\Gamma}_F$ are $F_T \times P$, the instantaneous force target equations become

$$\boldsymbol{\Gamma}_F \, \mathbf{p} \;\approx\; \boldsymbol{\gamma}_F \,. \tag{6.17}$$

6.4 Weighted Target System

Combining the rows of the mass and force target equations together, the instantaneous target system can be expressed as

$$\boldsymbol{\Gamma} \, \mathbf{p} \;\approx\; \boldsymbol{\gamma} \,. \tag{6.18}$$

Supplementary target equations (see Section 7.5) are included, for example to specify estimated values for some of the parameters. Let these other target equations be denoted by $\boldsymbol{\Psi} \, \mathbf{p} \approx \boldsymbol{\psi}$.

If time steps $0, \ldots, N$ are used to generate the instantaneous target equations, the complete target system is

$$\begin{bmatrix} \boldsymbol{\Gamma}^{(0)} \\ \boldsymbol{\Gamma}^{(1)} \\ \vdots \\ \boldsymbol{\Gamma}^{(N)} \\ \boldsymbol{\Psi} \end{bmatrix} \mathbf{p} \;\approx\; \begin{bmatrix} \boldsymbol{\gamma}^{(0)} \\ \boldsymbol{\gamma}^{(1)} \\ \vdots \\ \boldsymbol{\gamma}^{(N)} \\ \boldsymbol{\psi} \end{bmatrix} \tag{6.19}$$

which is written more concisely as

$$\mathbf{p} \;\approx\; \boldsymbol{\upsilon} \,. \tag{6.20}$$

Letting the number of target equations be L $(L \geq (N+1)(M_T + F_T))$, has dimensions $L \times P$. In practice $L \gg P$ so the parameters are overdetermined (barring singularity) and the target equations can only be approximately satisfied. (It is not uncommon for the target equations to be somewhat ill-conditioned when the data is not sufficient to clearly indicate a unique optimal model.)

To obtain a reasonable approximate solution, the target equations are weighted so that an error of one unit in each target equation is of equal importance. This requires pre-multiplying by an $L \times L$ weighting matrix, $\boldsymbol{\Omega}$.

$$\boldsymbol{\Omega} \; \mathbf{p} \;\approx\; \boldsymbol{\Omega} \, \boldsymbol{\upsilon} \tag{6.21}$$

or

$$\mathbf{T} \, \mathbf{p} \;\approx\; \boldsymbol{\tau} \tag{6.22}$$

with the natural correspondence. $\mathbf{T}$ is the $L \times P$ weighted target matrix and $\boldsymbol{\tau}$ is the weighted target vector. In practice $\boldsymbol{\Omega}$ is diagonal; no coupling between the equation errors is considered.

The complete weighted target system is an overdetermined set of equations, the optimal solution of which gives the desired model. By considering all of the target information together, SISAME can find the globally optimal model parameters, making the most use of the often limited information available.

A summary of the computations performed by SISAME is that the model is first time-stepped through the event, numerically integrating for the motions of simulated masses and accumulating the target information, and then the target information is solved, subject to constraints on the parameters, for the optimal model. The structure and solution of this optimization problem is treated below.

6.5 Constrained Approximation Problem

To optimally solve an overdetermined system first requires selecting the error norm to be minimized. SISAME uses the least squares (L_2) error norm for the weighted target equations as the basis for the optimization. The choice of this error measure is driven primarily by the fact that least squares problems are tractable and readily solved. Alternatives such as an L_1 error norm may also be practical.

The L_2 error of the target equations is

$$\begin{aligned} e &= \|\mathbf{T}\,\mathbf{p} - \boldsymbol{\tau}\|_2^2 \\ &= \mathbf{p}^T\,\mathbf{T}^T\,\mathbf{T}\,\mathbf{p} - 2\,\mathbf{p}^T\,\mathbf{T}^T\,\boldsymbol{\tau} + \boldsymbol{\tau}^T\boldsymbol{\tau}\,. \end{aligned} \tag{6.23}$$

Letting

$$\mathbf{G} = \mathbf{T}^T\,\mathbf{T} \qquad \text{and} \qquad \mathbf{h} = \mathbf{T}^T\boldsymbol{\tau} \qquad \text{and} \qquad b = \boldsymbol{\tau}^T\boldsymbol{\tau}$$

this becomes

$$e = \mathbf{p}^T\,\mathbf{G}\,\mathbf{p} - 2\,\mathbf{p}^T\mathbf{h} + b \tag{6.24}$$

where $\mathbf{G}$ is symmetric, positive semidefinite with dimensions $P\times P$.

The gradient of the error is

$$\nabla e = \mathbf{2}\,\mathbf{G}\,\mathbf{p} - \mathbf{2}\,\mathbf{h} \tag{6.25}$$

where $\mathbf{2}$ is the diagonal matrix with entries of 2. The optimal *unconstrained* parameters are obtained by setting $\nabla e = \mathbf{0}$, giving $\mathbf{p} = \mathbf{G}^{-1}\,\mathbf{h}$. The system $\nabla e = \mathbf{0}$ is known as the **normal equations**.

Constraints are used in SISAME to enforce valid behavior of the springs and to satisfy bounds and specified total model or vehicle weights for the extracted weights. Suppose the equality constraints are given by $\mathbf{C}_=\,\mathbf{p} = \mathbf{c}_=$ and the inequality constraints by $\mathbf{C}\,\mathbf{p} \leq \mathbf{c}$. To incorporate the constraints in the solution process the Lagrangian is formed

$$\ell = e + \boldsymbol{\lambda}^T\,(\mathbf{C}_=\,\mathbf{p} - \mathbf{c}_=) + \boldsymbol{\mu}^T\,(\mathbf{C}\,\mathbf{p} - \mathbf{c}) \tag{6.26}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the vectors of Lagrange multipliers associated with the equality and inequality constraints, respectively. The necessary and sufficient conditions [5] for the globally optimal parameters are

$$\begin{aligned}
\nabla_{\mathbf{p}}\,\ell \;=\; \nabla e + \mathbf{C}_{=}^{T}\,\boldsymbol{\lambda} + \mathbf{C}^{T}\,\boldsymbol{\mu} &= \mathbf{0} \\
\mathbf{C}_{=}\,\mathbf{p} &= \mathbf{c}_{=} \\
\mathbf{C}\,\mathbf{p} &\leq \mathbf{c} \\
\boldsymbol{\mu} &\geq \mathbf{0} \\
\boldsymbol{\mu}^{T}\,(\mathbf{C}\,\mathbf{p} - \mathbf{c}) &= 0\,.
\end{aligned} \tag{6.27}$$

This is a Linear Complementary Programming (LCP) [13] problem that can be solved by techniques based on the simplex method for Linear Programming (see Section 7.1).

7 SISAME Implementation

7.1 LCP Solution

The Linear Complementary Programming (LCP) optimality conditions (6.27) can be solved directly by standard LU factorization techniques. The instantaneous contributions to the **G** matrix and **h** vector can be accumulated during the time stepping procedure, so the very large **T** matrix is never explicitly formed.

Finding the optimal solution is made difficult by the need to determine which inequality constraints are active equalities at the optimum point. For activity set A the active inequality constraints become $\mathbf{C}_A\mathbf{p} = \mathbf{c}_A$ and the inactive constraint Lagrange multipliers are set to $\boldsymbol{\mu}_I = 0$ where the inactivity set is I. The optimality conditions for the tentative activity set A reduce to

$$\begin{bmatrix} \mathbf{G} & \mathbf{C}_=^T & \mathbf{C}_A^T \\ \mathbf{C}_= & \mathbf{0} & \mathbf{0} \\ \mathbf{C}_A & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{p} \\ \mathbf{2}\boldsymbol{\lambda} \\ \mathbf{2}\boldsymbol{\mu}_A \end{pmatrix} = \begin{pmatrix} \mathbf{h} \\ \mathbf{c}_= \\ \mathbf{c}_A \end{pmatrix} \tag{7.1}$$

with the additional conditions

$$\begin{aligned} \mathbf{C}_I\,\mathbf{p} &\leq \mathbf{c}_I \\ \boldsymbol{\mu}_A &\geq \mathbf{0}\,. \end{aligned} \tag{7.2}$$

Thus an activity set is optimal if the inactive constraints are satisfied and the active multipliers are nonnegative. These are primal and dual feasibility requirements, respectively. Testing for optimality is performed by solving the system in (7.1) and checking the conditions in (7.2).

Procedures for finding the (globally) optimal solution are similar to the simplex method for Linear Programming [13] except that the number of active constraints is not fixed at the number of variables. The method used in SISAME pivots between active constraint sets while maintaining a primal feasible solution that is (barring degeneracy) strictly improving. At each stage a feasible point on the current active constraint set is maintained. If an inactive constraint is violated by the solution of (7.1), the current working feasible point can be moved towards the solution until the first inactive constraint becomes a tight equality. It is not hard to show that the error measure at the new feasible point is reduced. If the solution violates no inactive constraints the feasible point can be updated to this new point. An active inequality constraint with a negative multiplier can also be dropped from the active set at any stage. It can be shown that the next solution of (7.1) will have a smaller error measure and will not violate the dropped constraint. When the solution satisfies both conditions in (7.2) the global optimum has been found.

While much more efficient than exhaustively trying all possible active sets, this process can still require a large number of pivots in models of modest complexity. Efficient computation of the $\mathbf{p}$ and $\boldsymbol{\mu}_A$ values is therefore critical to the efficiency of the extraction process.

SISAME uses a partial LU factorization from the previous pivot in computing the solution. The partial factorization for active set A is

$$\begin{bmatrix} \mathbf{PS} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \left[\begin{array}{ccc|c} \mathbf{G} & \mathbf{C}_=^T & \mathbf{C}_A^T & \mathbf{h} \\ \mathbf{C}_= & \mathbf{0} & \mathbf{0} & \mathbf{c}_= \\ \mathbf{C}_A & \mathbf{0} & \mathbf{0} & \mathbf{c}_A \end{array}\right] = \begin{bmatrix} \mathbf{L} & \mathbf{0} \\ \mathbf{Y}_A & \mathbf{I} \end{bmatrix} \left[\begin{array}{cc|c} \mathbf{U} & \hat{\mathbf{C}}_A^T & \hat{\mathbf{h}} \\ \mathbf{0} & \mathbf{Z}_A & \hat{\mathbf{c}}_A \end{array}\right] \tag{7.3}$$

where $\mathbf{PS}$ is a permutation and scaling matrix to allow for row scaling and partial pivoting during the reduction of $\mathbf{G}$ and the equality constraints ($\mathbf{PS}$, $\mathbf{L}$, and $\mathbf{U}$ have dimensions of P plus the number of equality constraints), $\mathbf{L}$ is unit lower triangular, $\mathbf{U}$ is upper triangular, and $\mathbf{Z}_A$ is not reduced to triangular form. The LU portion includes the equality constraint rows and columns since these do not change during the pivoting process and $\mathbf{G}$ may be singular without them in some cases. Since $\mathbf{G}$ is symmetric positive-definite, in theory a more efficient Cholesky factorization that avoids partial pivoting and scaling could be used. However, the condition number of $\mathbf{G}$ is often not small enough to make this a safe method, and the efficiency of this factorization is not critical since it is performed only once in each iteration.

Note that solving the partially factored system involves solving $2\mathbf{Z}_A\boldsymbol{\mu}_A = \hat{\mathbf{c}}_A$ then back-substitution on

$$\mathbf{U}\begin{pmatrix} \mathbf{p} \\ 2\boldsymbol{\lambda} \end{pmatrix} = \hat{\mathbf{h}} - 2\hat{\mathbf{C}}_A^T\boldsymbol{\mu}_A \tag{7.4}$$

which is considerably less work than solving the full system from scratch. The active constraint pivoting scheme used in SISAME attempts to remove a constraint with a negative Lagrange multiplier after adding each constraint to A. The removed constraint is selected based only on the values of $\boldsymbol{\mu}_A$, so back-substitution for $\mathbf{p}$ is avoided completely in a constraint removal pass.

The partially factored system must be efficiently updated for the addition or removal of a constraint from the active set. Such update formulas are not difficult to develop. Since $\mathbf{Z}_A$ is not row reduced, no cumulative error occurs when $\mathbf{Z}_A$ is updated, and the solution for $\boldsymbol{\mu}_A$ can be performed with row scaling and partial pivoting. These accuracy benefits come at the cost of lower efficiency than methods that update a factorization of $\mathbf{Z}_A$.

7.2 Spring Force Representation

At each instant the SISAME formulation requires a linear representation for the spring forces in terms of the extracted static and dynamic parameters (see Sections 6.2 and 6.3). In some cases this is straightforward.

A linear elastic static LE aspect with a linear LD damper has the exact, linear force representation

$$f \;=\; S\,x \;+\; D_{\text{Slp}}\,r\,.$$

A segmented elastic static SE aspect has the static force representation

$$f^S \;=\; \left(\tfrac{X_{i+1}-x}{X_{i+1}-X_i}\right) F_i \;+\; \left(\tfrac{x-X_i}{X_{i+1}-X_i}\right) F_{i+1}$$

where the $[X_i, X_{i+1}]$ segment is active at x, which is an exact, linear representation since the X_i values are defined.

Other static and dynamic types and combinations can have nonlinear extracted parameter contributions. Linearized local approximants are employed to address these nonlinearities (see Section 7.2.1). Some static and dynamic types can have domain transitions that depend on extracted parameters and must be estimated from the previous solution (see Section 7.2.2). The use of approximants or estimates requires SISAME to use an iterative procedure to obtain a solution (see Section 7.3).

7.2.1 Linear Approximants

Nonlinear spring force expressions are linearized in SISAME using first order Taylor expansions about the previous parameter solution. This yields approximants of the form

$$\overline{f}(\mathbf{p}) \;=\; f(\widetilde{\mathbf{p}}) \;+\; \nabla f(\mathbf{p})\,(\mathbf{p}-\widetilde{\mathbf{p}})$$

where $\overline{f}$ is the approximant for f and $\widetilde{\mathbf{p}}$ is the previous solution for the parameter vector $\mathbf{p}$. Some examples of the approximants are given below.

The static EE type has static force

$$f^S \;=\; S_O\,x \;+\; C\,\sigma(x)\,|x|^P$$

which is nonlinear if P is extracted. The EE approximant is

$$\overline{f^S} \;=\; [x]\,S_O \;+\; \left[\sigma(x)\,|x|^{\widetilde{P}}\right] C \;+\; \left[\widetilde{C}\sigma(x)\,|x|^{\widetilde{P}}\ln|x|\right] P \;-\; \left[\widetilde{C}\sigma(x)\,|x|^{\widetilde{P}}\ln|x|\widetilde{P}\right]\,.$$

The static AE type has static force

$$f^S \;=\; \left(\tfrac{2}{\pi}\,F_{\text{Max}}\right)\text{Arctan}\!\left(\tfrac{\pi}{2}\tfrac{S_O}{F_{\text{Max}}}\,x\right)$$

which is nonlinear unless the value of $\frac{S_O}{F_{\text{Max}}}$ is defined. Letting $\mathcal{X} = \frac{\pi}{2}\frac{S_O}{F_{\text{Max}}}\,x$ the AE approximant is

$$\overline{f^S} \;=\; \left[\tfrac{x}{1+\widetilde{\mathcal{X}}^2}\right] S_O \;+\; \left[\tfrac{2}{\pi}\left(\text{Arctan}\!\left(\widetilde{\mathcal{X}}\right) - \tfrac{\widetilde{\mathcal{X}}}{1+\widetilde{\mathcal{X}}^2}\right)\right] F_{\text{Max}}\,.$$

The static AI type boundary force approximant is similar with an additional term for the effect of boundary shifts.

The static BE type has a linear origin line static force of $f^S = S_O\,x$ and compression and tension line static forces of the form

$$f^S_\square \;=\; F_\square \;+\; [x - X_\square]S_\square$$

where $\square$ is C or T, which is nonlinear if both $X_\square$ and $S_\square$ are extracted. If $X_\square$ is the explicit parameter then, using $F_\square = S_O X_\square$, the BE approximant is

$$\overline{f^S_\square} \;=\; \left[\widetilde{X}_\square\right] S_O \;+\; \left[\widetilde{S}_O - \widetilde{S}_\square\right] X_\square \;+\; \left[x - \widetilde{X}_\square\right] S_\square \;-\; \left[\left(\widetilde{S}_O - \widetilde{S}_\square\right)\widetilde{X}_\square\right]\,.$$

If $F_\square$ is the explicit parameter and $S_O > 0$ the BE approximant is

$$\overline{f^S_\square} \;=\; \left[\tfrac{\widetilde{F}_\square \widetilde{S}_\square}{\widetilde{S}_O^2}\right] S_O \;+\; \left[1 - \tfrac{\widetilde{S}_\square}{\widetilde{S}_O}\right] F_\square \;+\; \left[x - \widetilde{X}_\square\right] S_\square$$

(when $\widetilde{S}_O = 0$ a different approximant is used). The static BI type boundary force approximant is similar to the explicit $F_\square$ parameter case.

The nonsymmetric static SI type has the tension line static force

$$f^S_T \;=\; S_T\left(x - \left(X_B - \tfrac{F_B}{S_U} - X_{\text{Slk}}\right)\right)$$

where (X_B, F_B) is the compression boundary point that the unloading path connects to, which has two nonlinear terms. The SI tension line approximant is

$$\overline{f^S_T} \;=\; \left[\tfrac{\widetilde{S_T}}{\widetilde{S_U}}\right] F_B \;-\; \left[\tfrac{\widetilde{F_B}\widetilde{S_T}}{\widetilde{S}_U^2}\right] S_U \;+\; \left[x - \left(X_B - \tfrac{\widetilde{F_B}}{\widetilde{S_U}} - \widetilde{X_{\text{Slk}}}\right)\right] S_T \;+\; \left[\widetilde{S_T}\right] X_{\text{Slk}} \;-\; \left[\widetilde{X_{\text{Slk}}}\widetilde{S_T}\right]\,.$$

The dynamic AD damper type has force

$$f \;=\; f^S \;+\; \left(\tfrac{2}{\pi}D_{\text{Max}}\right)\text{Arctan}\!\left(\tfrac{\pi}{2}\tfrac{D_{\text{Slp}}}{D_{\text{Max}}}r\right)$$

which is nonlinear unless the value of $\frac{D_{\text{Slp}}}{D_{\text{Max}}}$ is defined. Letting $\mathcal{R} = \frac{\pi}{2}\frac{D_{\text{Slp}}}{D_{\text{Max}}}\,r$ the AD approximant is

$$\overline{f} \;=\; f^S \;+\; \left[\tfrac{r}{1+\widetilde{\mathcal{R}}^2}\right] D_{\text{Slp}} \;+\; \left[\tfrac{2}{\pi}\left(\text{Arctan}\!\left(\widetilde{\mathcal{R}}\right) - \tfrac{\widetilde{\mathcal{R}}}{1+\widetilde{\mathcal{R}}^2}\right)\right] D_{\text{Max}}\,.$$

Extracted dynamic magnifiers require approximants because their parameters appear in a product/ratio with the (possibly extracted) static parameters and because their parameters appear in the denominator for the $\sigma(rf^S) = -1$ case.

The dynamic LM magnifier type has force

$$f \;=\; \left[1 + M_{\text{Slp}}\,|r|\,\right]^{\sigma(rf^S)} f^S$$

which is nonlinear if M_{Slp} is extracted and either f^S is extracted or $\sigma(rf^S) = -1$. Letting $\mathcal{M} = M_{\text{Slp}}\,|r|$ the LM approximants are

$$\overline{f}|_{\sigma=1} \;=\; \left[1 + \widetilde{\mathcal{M}}\right] f^S \;+\; \left[\widetilde{f^S|r|}\right] M_{\text{Slp}} \;-\; \left[\widetilde{f^S\mathcal{M}}\right]$$

$$\overline{f}|_{\sigma=-1} \;=\; \left[\tfrac{1}{1+\widetilde{\mathcal{M}}}\right] f^S \;-\; \left[\tfrac{\widetilde{f^S|r|}}{(1+\widetilde{\mathcal{M}})^2}\right] M_{\text{Slp}} \;+\; \left[\tfrac{\widetilde{f^S\mathcal{M}}}{(1+\widetilde{\mathcal{M}})^2}\right]$$

The dynamic AM magnifier type has force

$$f \;=\; \left[1 + \left(\tfrac{2}{\pi}(M_{\text{Max}} - 1)\right) \operatorname{Arctan}\!\left(\tfrac{\pi}{2}\tfrac{M_{\text{Slp}}}{M_{\text{Max}}-1}\,|r|\,\right)\right]^{\sigma(rf^S)} f^S$$

which is nonlinear unless the value of $\frac{M_{\text{Slp}}}{M_{\text{Max}}-1}$ is defined or if M_{Slp} or M_{Max} is extracted and either f^S is extracted or $\sigma(rf^S) = -1$. Letting $\mathcal{M} = \frac{2}{\pi}(M_{\text{Max}} - 1)$ and $\mathcal{R} = \frac{\pi}{2}\frac{M_{\text{Slp}}}{M_{\text{Max}}-1}\,|r| = \frac{M_{\text{Slp}}}{\mathcal{M}}\,|r|$ and $\mathcal{A} = \operatorname{Arctan}(\mathcal{R})$ the AM approximants are

$$\overline{f}|_{\sigma=1} \;=\; \left[1 + \widetilde{\mathcal{M}}\widetilde{\mathcal{A}}\right] f^S \;+\; \left[\widetilde{f^S}\tfrac{|r|}{1+\widetilde{\mathcal{R}}^2}\right] M_{\text{Slp}} \;+\; \left[\widetilde{f^S}\tfrac{2}{\pi}\left(\widetilde{\mathcal{A}} - \tfrac{\widetilde{\mathcal{R}}}{1+\widetilde{\mathcal{R}}^2}\right)\right](M_{\text{Max}} - 1) \;-\; \left[\widetilde{f^S}\widetilde{\mathcal{M}}\widetilde{\mathcal{A}}\right]$$

$$\overline{f}|_{\sigma=-1} \;=\; \left[\tfrac{1}{1+\widetilde{\mathcal{M}}\widetilde{\mathcal{A}}}\right] f^S \;-\; \left[\tfrac{\widetilde{f^S}\frac{|r|}{1+\widetilde{\mathcal{R}}^2}}{(1+\widetilde{\mathcal{M}}\widetilde{\mathcal{A}})^2}\right] M_{\text{Slp}} \;-\; \left[\tfrac{\widetilde{f^S}\frac{2}{\pi}\left(\widetilde{\mathcal{A}}-\frac{\widetilde{\mathcal{R}}}{1+\widetilde{\mathcal{R}}^2}\right)}{(1+\widetilde{\mathcal{M}}\widetilde{\mathcal{A}})^2}\right](M_{\text{Max}} - 1) \;+\; \left[\tfrac{\widetilde{f^S}\widetilde{\mathcal{M}}\widetilde{\mathcal{A}}}{(1+\widetilde{\mathcal{M}}\widetilde{\mathcal{A}})^2}\right]$$

The approximants are, in general, inexact representations of the spring force (since $\mathbf{p} \neq \widetilde{\mathbf{p}}$) so an iterative procedure (see Section 7.3) is used to converge on a self-consistent solution: as $(\mathbf{p} - \widetilde{\mathbf{p}}) \to 0$ the approximation error goes to zero.

To control the approximation error and facilitate robust convergence SISAME uses **iteration constraints** that limit the change to $\mathbf{p}$ in each iteration. Constraints specific to each approximant form are implemented to achieve a balance between convergence speed and robustness.

7.2.2 Domain Transitions

Many of the static types can have domain transitions that depend on extracted parameters, including

- Inelastic static transitions between the unloading line and the opposite boundary or slack domains.
- One-way clipping transitions.
- Static BE and BI transitions between the origin line and compression/tension lines/boundaries.
- Static SI transitions between unloading, slack, and reloading domains.

The dynamic magnifier exponent, $\sigma(rf^s)$, depends on the sign of the static force, which can depend on extracted static parameters.

When domain transitions depend on extracted parameters SISAME uses the previous solution to select the active domain at each instant. This inexact domain identification requires the use of an iterative procedure (see Section 7.3) to converge on a self-consistent solution.

7.3 Extraction Iteration

When approximants (see Section 7.2.1) are used or when domain transitions depend on extracted parameters (see Section 7.2.2) an iterative procedure is used to seek a self-consistent (fixed point) solution. The iteration starts with an initial solution (generated automatically) and uses this solution to determine the static force sign and other state information necessary to set up a tractable approximation problem that tends to improve the solution. The resulting solution is used in the same way, and the iterations continue until convergence yields a self-consistent solution.

SISAME automatically detects and reports the presence of features requiring iterative solutions. Most real-world models will contain such features.

SISAME uses an automatic **relaxation** process to speed convergence as the maximum number of iterations is approached or when the iterations are cycling (usually near a self-consistent solution). During relaxation the solution changes by a diminishing fraction of the distance from the last solution to the newly extracted solution.

Despite the lack of guaranteed optimality for iteratively obtained models, trials indicate that optimal or near-optimal models are obtained in almost all cases. With an appropriate initial feasible solution, it is not hard to show that, for any one feature, the iterations generally tend to improve successive solutions. Nevertheless, the fewer the "difficult" features the faster the iterations can be expected to converge, and the more likely the resulting model will be at least near-optimal.

7.4 Velocity and Displacement Targets

The velocity and displacement domain target equations of (6.13) and (6.14) are implemented by numerical integration. In particular, the influence matrix integrals

$$\int \mathbf{\Pi}_{E}\, d\tau \qquad \text{and} \qquad \int\!\!\int \mathbf{\Pi}_{E}\, d\tau\, dv$$

must be accumulated to form the target system at each time step. In practice, a weighted sum is used to approximate the velocity and displacement target equations; a higher-order integration method would be inappropriate given the spring static characteristic slope discontinuities and would require storing $S_E \times P_S$ matrices for a number of time steps.

The integrated matrices include parameter contributions from time-zero onward, yielding target systems that are less sparse than the acceleration target systems. Thus the use of velocity and displacement domain targets can cause some slowdown of the extraction. Since the velocities and displacements are smoother than accelerations, SISAME applies these targets only at every eighth time step to minimize the computational burden. The velocity and displacement targeting can also be suppressed by the user, but this is not recommended for more sensitive modeling applications.

7.5 Supplementary Target Equations

The extracted spring solution can be refined by the introduction of supplementary target equations. These might be used to condition the least squares problem in cases that are inherently ill-conditioned, such as models with many load-path loops. Or target equations might be used to provide estimates for some of the parameters to assist SISAME in partitioning the forces among multiple load-paths. In general, providing any additional information known about the parameters helps in narrowing the sometimes large space of near-optimal models.

SISAME uses a lightly weighted diagonal **parameter conditioning** target system of the form

$$\mathbf{I}\,\mathbf{p} \;\approx\; \mathbf{0} \tag{7.5}$$

to prevent singularity when there are parameters that are never active in the given motion data or, for example, when two extracted springs are connected in parallel to the same masses. This also improves the condition of extractions that are otherwise ill-conditioned.

If SISAME detects pairs of parameters, $\mathbf{p}_i$ and $\mathbf{p}_j$, with identical contributions to the least squares system (6.18) weighted equations of the form $\mathbf{p}_i \approx \mathbf{p}_j$ are included in a **parameter equivalence conditioning** target system of the form

$$\mathbf{E}\,\mathbf{p} \;\approx\; \mathbf{0} \tag{7.6}$$

to improve the condition of the extraction in the presence of these singularities.

SISAME also uses a diagonal **parameter damping** target system of the form

$$\mathbf{I}\,\mathbf{p} \;\approx\; \mathbf{p}_f \tag{7.7}$$

where $\mathbf{p}_f$ is the current feasible solution. This target system tends to damp changes in the parameters and thus assists in the final solution convergence.

Another lightly weighted target system is used to make SISAME prefer smooth segmented static characteristics. For each junction between two segments, either of which include at least one extracted force, there is a target equation of the form

$$\tfrac{\Delta_i}{\Delta_i^{(2)}}\,F_{i-1} \;-\; F_i \;+\; \tfrac{\Delta_i}{\Delta_i^{(2)}}\,F_{i+1} \;\approx\; 0 \tag{7.8}$$

where $\Delta_i \doteq X_{i+1} - X_i$ and $\Delta_i^{(2)} \doteq X_{i+1} - X_{i-1}$. The target error is the amount (in force units) that F_i differs from the value at X_i of the line between F_{i-1} and F_{i+1}.

All of these supplementary target equations are included in the $\boldsymbol{\Psi}\,\mathbf{p} \approx \boldsymbol{\psi}$ system of (6.19).

7.6 Target Equation Weighting

The target equations are weighted by the matrix $\boldsymbol{\Omega}$. The weights are needed to assure that the least squares solution balances the errors with respect to their significance and not their numerical size. SISAME uses a diagonal $\boldsymbol{\Omega}$, a common simplifying approach although it treats the errors as independent random variables, which they are not.

For truly independent errors, weights can be derived from confidence bands. Fixing a common level of confidence (*e.g.*, 90%), a band of $\pm\Delta$ can be assigned to each equation. The corresponding weight would be $1/\Delta$. This approach is used to set the weights for the supplementary target equations.

To select weights for the mass acceleration and force target equations that compensate for the number of time steps and the dependencies among errors at nearby time points, a confidence band of $\pm\Delta_{\mathrm{RMS}}$ on the root mean square error for each equation is used. The corresponding weight is

$$\frac{1}{\Delta_{\mathrm{RMS}}\sqrt{N+1}}$$

where $N+1$ is the total number of time steps, $0,\ldots,N$, at which each target equation is applied. SISAME provides user-controllable Δ_{RMS} values for the mass acceleration targets, ***ConIF***, and force targets, ***ConF***.

For the velocity and displacement targets SISAME uses weighting of

$$\frac{1}{\mathit{ConIF}\sqrt{N+1}}\,\frac{\eta}{\mathit{ConV}\,(N+1)} \qquad \text{and} \qquad \frac{1}{\mathit{ConIF}\sqrt{N+1}}\,\frac{\eta}{\mathit{ConD}\,(N+1)^2}$$

where η includes compensation for applying these targets at only a subset of the time steps and a heuristic factor designed to let the default *ConV* and *ConD* factors of 1 give a reasonable balance between the acceleration, velocity, and displacement target weighting for most applications.

7.7 Extracted Parameter Constraints

Constraints are used in SISAME to ensure physically meaningful extracted components, to meet user-specified extracted parameter bounds (see Section 3.6), and to control approximant errors (see Section 7.2.1). The SISAME solution procedure supports linear equality and inequality constraints.

Automatically generated constraints ensure

- Extracted mass weights are nonnegative and meet any specified vehicle or run weights.
- Extracted static and dynamic parameters meet the requirements of their types.

For example, an extracted nonsymmetric static SI aspect has constraints

- $S_U \geq 0$
- $S_T \geq 0$
- $X_{\text{Slk}} \geq 0$
- $F_i + F_\Delta \geq 0$ for $i = 1, \ldots, N$
- $F_i \geq F_{i-1}$ for $i = 2, \ldots, N$ ensuring nonnegative segment slopes (unless `AnySlope=True`)
- $F_i - F_{i-1} \leq (X_i - X_{i-1})\, S_U$ for $i = 2, \ldots, N$ ensuring segment slopes are less than S_U

User-specified extracted parameter bounds allow partial parameter information to be exploited to reduce the space of feasible solutions. (Constraints increase the computational effort more than target equations so extracted parameter estimates should be preferred when acceptable.)

SISAME automatically eliminates redundant constraints for efficiency.

7.8 Phase I Procedure

The extraction process requires an initial feasible solution from which to begin the optimization (see Section 7.1). SISAME constructs an initial solution that satisfies the inherent constraints on model component behavior and the user-specified numeric parameter bounds. References between extracted parameters may cause the constructed solution to be infeasible. In this case a Phase I procedure is automatically activated to obtain a feasible solution.

The Phase I procedure can also be triggered in iterations subsequent to the first pass if ill-conditioning causes the solution to drift sufficiently far from precise feasibility.

For the Phase I procedure an additional parameter, p_0, is introduced, the error function is augmented with $\frac{1}{2}Mp_0^2 + Mp_0$ for some large value of M, and the constraint $p_0 \geq 0$ is included. The existing constraints are also modified to

$$\mathbf{C}_= \mathbf{p} + \mathbf{s}\, p_0 \;=\; \mathbf{c}_= \qquad \text{and} \qquad \mathbf{C}\,\mathbf{p} + \mathbf{u}\, p_0 \;\leq\; \mathbf{c}\,.$$

The values of $\mathbf{s}$ and $\mathbf{u}$ and an initial feasible solution to this augmented problem are constructed as follows, starting with an arbitrary parameter vector $\mathbf{p}$. The elements of $\mathbf{u}$ are zero for the original constraints that are not violated by $\mathbf{p}$ and -1 for those that are. The initial value of p_0 is simply any value large enough so that all the violated inequality constraints, with the additional $-p_0$ term, are satisfied. Then $\mathbf{s}$ is set to whatever is needed to satisfy the augmented equality constraints with this p_0.

The Phase I optimality equations (7.1) become

$$\begin{bmatrix} M & \mathbf{0} & \mathbf{s}^T & \mathbf{u}_A^T \\ \mathbf{0} & \mathbf{G} & \mathbf{C}_=^T & \mathbf{C}_A^T \\ \mathbf{s} & \mathbf{C}_= & \mathbf{0} & \mathbf{0} \\ \mathbf{u}_A & \mathbf{C}_A & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{pmatrix} p_0 \\ \mathbf{p} \\ 2\boldsymbol{\lambda} \\ 2\boldsymbol{\mu}_A \end{pmatrix} = \begin{pmatrix} -M \\ \mathbf{h} \\ \mathbf{c}_= \\ \mathbf{c}_A \end{pmatrix}. \tag{7.9}$$

If M is large enough and a feasible solution exists, the solution to this augmented problem will have $p_0 = 0$ and thus will be feasible for the original problem. If $p_0 \neq 0$ SISAME will increase M automatically and repeat the Phase I solution. If M becomes so large that ill-conditioning is severe, SISAME terminates and reports that no feasible solution exists.

This Phase I procedure involves minimal additional storage and has the benefit of producing the desired solution at the same time since, with $p_0 = 0$, the solution for $\mathbf{p}$ will minimize the original error function. In practice the Phase I solution for $\mathbf{p}$ is used as the initial feasible solution in a verification solution of the unaugmented problem to eliminate any numerical distortion that the large M value may have introduced.

7.9 Filtered Data Representation

The instrumented motion and force data is transformed to a special Discrete Fourier Transform (DFT) representation for reconstruction of the filtered motions at each simulation time step. A baseline function with the correct acceleration, velocity, and displacement tail values is removed before compaction to a DFT. This facilitates a DFT-based filter that simultaneously approximates the motion in all three domains without drift errors in any of them. The complete description of this filtering approach can be found in the documentation for the SimFil program [9, 10]. The filtering in SISAME is equivalent to SimFil filtering using a DFT final time of, if available, 1.1× the final SISAME output time (***FinTOut***) and with the specified SISAME tail smoothing (***ZeroSm*** and ***EndSm***, which default to the cutoff frequency ***Cutoff***).

7.10 Time Step Selection

The simulated mass motion state is determined from the current defined spring state by integrating the equations of motion from the previous time step. SISAME selects a simulation time step based on the frequency content, damping, and the output and DFT timing. First, an approximation to the highest frequency of the simulated mass subsystem is computed. Also, the highest frequency in the filtered instrument data is considered. Finally the maximum damping to weight ratio is considered. SISAME selects a time step with enough steps per maximum frequency cycle to assure accurate integration. Let this simulation time step be denoted Δ_ω.

The actual simulation time step, Δ is selected as the largest value less than or equal to Δ_ω that divides the output time step, Δ_{o}.

If the model has instrumented masses or forces, SISAME selects a final time for the DFT data representation, T_I, which is at least the requested final output time, T_{o}. To allow the fast reconstruction method, Δ and T_I must have a sufficiently large common divisor. To see if this is possible, SISAME first computes

$$\Delta_{\mathrm{gcd}} = \mathrm{gcd}(T_I, \Delta_{\mathrm{o}})$$

If this is greater than or equal to Δ, then Δ is reduced to the largest value that divides Δ_{gcd}.

If Δ_{gcd} is less than Δ, SISAME looks for a slightly smaller Δ sharing a sufficiently large common divisor with Δ_{gcd}. This divisor determines the number of sine/cosine values needed to cover the DFT time span T_I. The maximum size of these trigonometric arrays and the overhead of accessing elements of large arrays are considered in selecting the best combination of divisor and Δ. If no good choice within the array size limitation is found, SISAME uses a slower reconstruction method that computes the trigonometric values as they are needed.

7.11 Numerical Integration

The simulated mass motions are numerically integrated using a low-order predictor–corrector method. Higher order methods are not appropriate because of the slope discontinuities that may be present in the static spring characteristics. SISAME uses a predictor step followed by two corrector steps. The combination is a second order method for the velocity and a third order method for the displacement.

The predictor step is based on the state of the simulated masses at the previous simulation time step. The predictor step at the i^{th} time step for each simulated mass is

$$\begin{aligned} v_i^* &= v_{i-1} + \Delta a_{i-1} && \text{(Euler)} \\ d_i^* &= d_{i-1} + \tfrac{\Delta}{2}\,(v_{i-1} + v_i^*) && \text{(Trapezoid)} \end{aligned} \tag{7.10}$$

Given the velocities and displacements, the predicted force in the defined springs $\mathbf{f}_D^*$ is computed and the predicted simulated mass accelerations, a_i^* are obtained from

$$\mathbf{a}_S^* = \mathbf{W}_S^{-1}\,\mathbf{\Phi}_{SD}\,\mathbf{f}_D^* \tag{7.11}$$

The first corrector step is

$$\begin{aligned} v_i^{**} &= v_{i-1} + \tfrac{\Delta}{2}\,(a_{i-1} + a_i^*) && \text{(Trapezoid)} \\ d_i^{**} &= d_{i-1} + \Delta v_{i-1} + \tfrac{\Delta^2}{6}\,(2a_{i-1} + a_i^*) && \text{(Parabolic)} \end{aligned} \tag{7.12}$$

The spring force $\mathbf{f}_D^{**}$ is computed and the accelerations, a_i^{**}, are obtained from

$$\mathbf{a}_S^{**} = \mathbf{W}_S^{-1}\,\mathbf{\Phi}_{SD}\,\mathbf{f}_D^{**} \tag{7.13}$$

The second corrector step has the same form as the first

$$\begin{aligned} v_i &= v_{i-1} + \tfrac{\Delta}{2}\,(a_{i-1} + a_i^{**}) && \text{(Trapezoid)} \\ d_i &= d_{i-1} + \Delta v_{i-1} + \tfrac{\Delta^2}{6}\,(2a_{i-1} + a_i^{**}) && \text{(Parabolic)} \end{aligned} \tag{7.14}$$

The spring force $\mathbf{f}_D$ is computed and the accelerations, a_i, are obtained from

$$\mathbf{a}_S = \mathbf{W}_S^{-1}\,\mathbf{\Phi}_{SD}\,\mathbf{f}_D \tag{7.15}$$

Appendices

A SISAME Modeling Example

This appendix presents a simple SISAME modeling example based on a 56 kilometer/hour full frontal, nonoblique, vehicle–to–barrier crash test of a four door sedan (NHTSA Vehicle Test number 1890).

The instrument signals were preprocessed with VeCor to prepare them for modeling use. For this uniaxial event, the symmetric right–left instrument signals were then averaged to obtain the occupant compartment, engine, and front wheel accelerations and the barrier force shown in Figure A.1.

A.1 Weight Extraction

A simple weight extraction run was performed to help select mass weights and to assess the completeness of the set of accelerations. The weight extraction input and output files are shown in Figures A.2.

The weight extraction was performed with a 60 Hz cutoff frequency since the barrier force is a sum of signals and contains little higher frequency content. Time-series outputs with a 0.0001 second time step and a final output time of 0.15 second (approximately when the vehicle separates from the barrier) were specified. The vehicle weight of 1711 kilograms was specified to constrain the total of the extracted mass weights.

The weight extraction selects weights for the masses to match the vehicle inertia force to the measured barrier force. Typically, the barrier–inertia force match alone does not clearly identify accurate mass weights, and estimates or bounds based on engineering knowledge are used. In this case only a lower bound on the weight of the front wheel assemblies was used.

The barrier–inertia force comparison (see Figure A.3) shows how well the three accelerations, when combined using the extracted weights, can capture the vehicle inertia force as measured at the barrier. The match obtained in this case is quite good for a three-mass model and indicates that the set of accelerations is an acceptably complete representation of the vehicle's inertia profile.

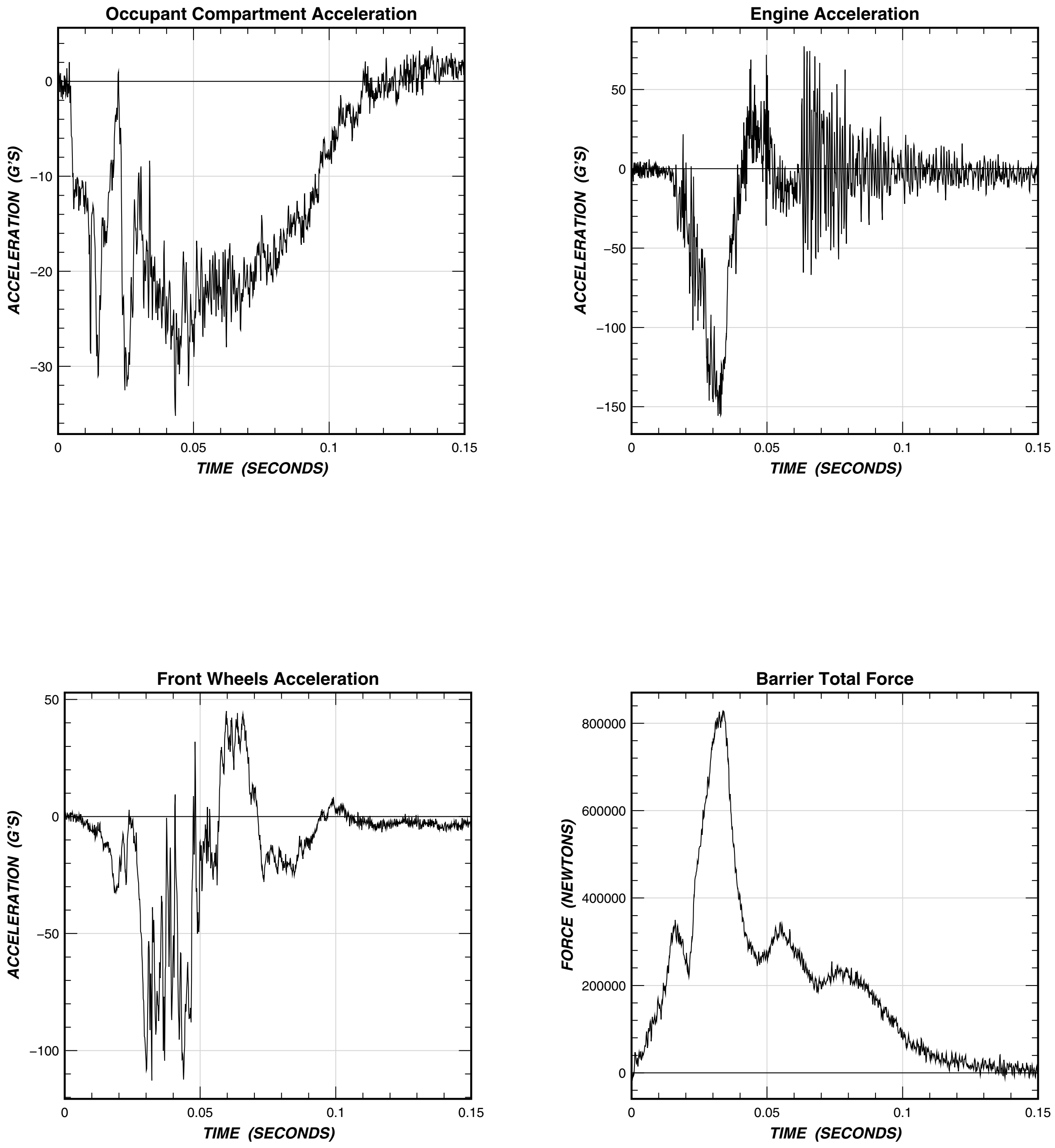


Figure A.1: Instrument signals.

```
SISAME Input File

Run Information

  RunID=1890_Wt
    Title=Full Frontal Test 1890 Weight Extraction
    DelTOut=.0001  FinTOut=.15

Model Information

  MdlID=1890_Wt
    DimSys=Metric
    Cutoff=60

    FrcID=InertiaFrc
      MemMass=*
      File=..\..\Data\Model\v1890fv0.Bar

  VehID=Vehicle
    Wt=1711  IniVel=56.3

    MassID=OccComp  Descr=Occupant Compartment
      Class=D
      Wt=?
      File=..\..\Data\Model\v1890av1.Occ

    MassID=Engine  Descr=Engine
      Class=D
      Wt=?
      File=..\..\Data\Model\v1890av1.Eng

    MassID=Wheels  Descr=Front Wheels/Suspension
      Class=D
      Wt=?( >120 )
      File=..\..\Data\Model\v1890av1.Whe

Output Information

  OutClass=FrcTS  Qty=Ff  Frc=InertiaFrc
  OutClass=FitRep
  OutClass=Model
```

```
SISAME Input File

Run Information

  RunID=1890_Wt
    Title=Full Frontal Test 1890 Weight Extraction
    DelTOut=.0001  FinTOut=.15

Model Information

  MdlID=1890_Wt
    DimSys=Metric
    Cutoff=60

  VehID=Vehicle
    Wt=1711  IniVel=56.3

    MassID=OccComp  Descr=Occupant Compartment
      Class=D
      Wt=1203.171
      File=..\..\Data\Model\v1890av1.Occ

    MassID=Engine  Descr=Engine
      Class=D
      Wt=387.8291
      File=..\..\Data\Model\v1890av1.Eng

    MassID=Wheels  Descr=Front Wheels/Suspension
      Class=D
      Wt=120
      File=..\..\Data\Model\v1890av1.Whe

Output Information

  OutClass=MassTS  Qty=AVD  Mass=*
```

Figure A.2: Weight extraction input and output files.

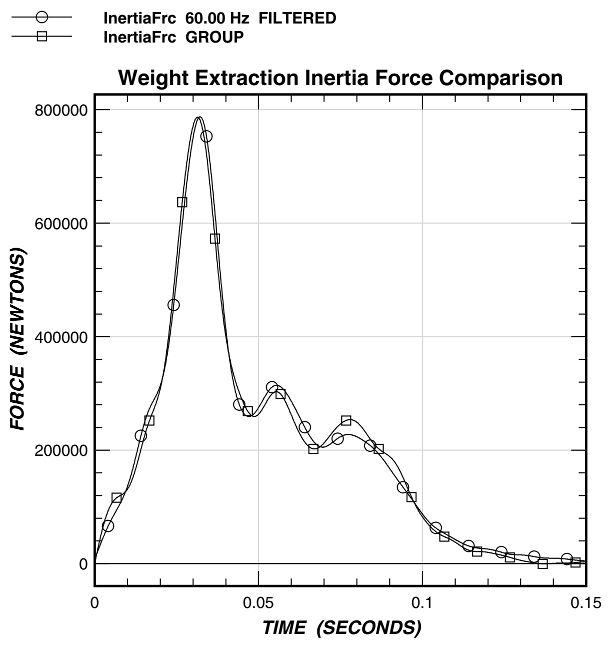
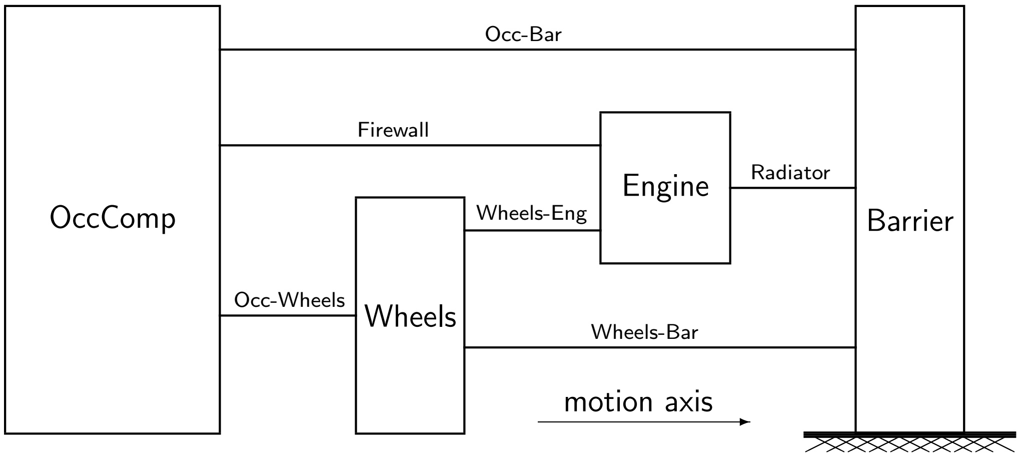


Figure A.3: Weight extraction inertia force comparison.

A.2 Model Extraction

The weight extraction output file was turned into a model extraction input file by rounding the weight values and adding the six extracted load-paths shown in the model schematic below, yielding the model extraction input file shown in Figure A.4.



A cutoff frequency of 60 Hz was selected to focus this coarsely discretized model on the low-frequency structural behavior. The final time was set at 0.125 second to cover the essential portion of the impact.

A uniform load-path type was selected for simplicity: segmented inelastic (SI) static aspects with linear magnifier (LM) dynamic aspects. The static forces at zero deflection were set to zero to correspond to the unloaded state of the structure prior to impact. This simple model lacks an interface so the load-paths contacting the barrier were assigned a zero tension slope (`ST=0`). The other load-paths were given extracted tension slopes and slack distances.

`MassTS` outputs of the filtered and effective motions were requested for model assessment. The `SprPar` outputs provide a visual representation of the extracted static and dynamic aspects. The `FitRep` output presents a quantitative summary of the extraction fit measures. The `Model` output file is used for examination and simulation of the extracted model.

SISAME was run on the input file producing the log file and requested outputs. The log file run and model summaries are shown in Figure A.5. The log file will also contain any warning and error messages, a history of the extraction iterations, and a list of the generated outputs.

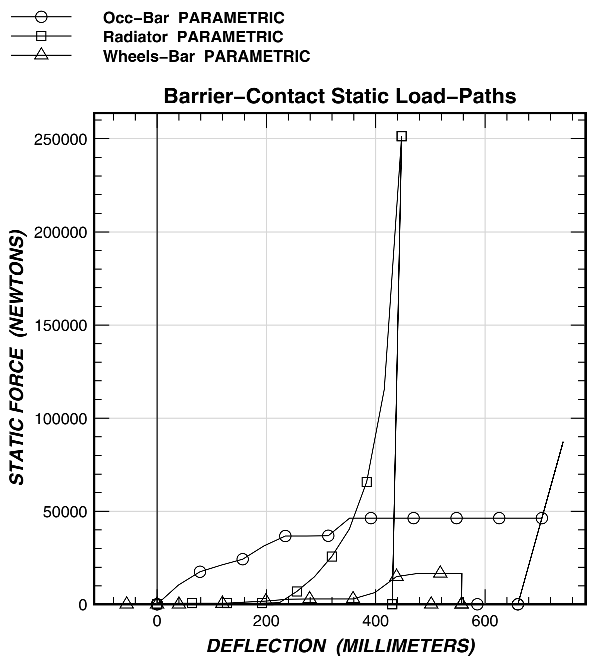
The extracted static load-path characteristics are shown in Figure A.6. The load-paths exhibit some typical features of simple full frontal models. For example, the engine–to–barrier (`Radiator`) load-path exhibits significant slack followed by rapidly stiffening loading and a very stiff unloading, as expected for a large rigid mass separated from the barrier by clearance and relatively light, low-strength components.

```
SISAME Input File

Run Information

  RunID=1890
    Title=Full Frontal Test 1890 Model
    DelTOut=.0001  FinTOut=.125

Model Information

  MdlID=1890
    DimSys=Metric
    Cutoff=60

  VehID=Vehicle
    Wt=1711  IniVel=56.3

    MassID=OccComp  Descr=Occupant Compartment
      Wt=1203
      File=..\..\Data\Model\v1890av1.Occ

    MassID=Engine  Descr=Engine
      Wt=388
      File=..\..\Data\Model\v1890av1.Eng

    MassID=Wheels  Descr=Front Wheels/Suspension
      Wt=120
      File=..\..\Data\Model\v1890av1.Whe

    SprID=Occ-Bar
      NegMass=OccComp  PosMass=Barrier
      StaType=SI  SU=?  ST=0
        X=  0  #19
        F=  0  ?19
      DynType=LM  MSlp=?( ~0[1] )

    SprID=Radiator
      NegMass=Engine  PosMass=Barrier
      StaType=SI  SU=?  ST=0
        X=  0  #14
        F=  0  ?14
      DynType=LM  MSlp=?( ~0[1] )

    SprID=Wheels-Bar
      NegMass=Wheels  PosMass=Barrier
      StaType=SI  SU=?  ST=0
        X=  0  #14
        F=  0  ?14
      DynType=LM  MSlp=?( ~0[1] )

    SprID=Firewall
      NegMass=OccComp  PosMass=Engine
      StaType=SI  SU=?  ST=?  XSlk=?
        X=  0  #14
        F=  0  ?14
      DynType=LM  MSlp=?( ~0[1] )

    SprID=Occ-Wheels
      NegMass=OccComp  PosMass=Wheels
      StaType=SI  SU=?  ST=?  XSlk=?
        X=  0  #14
        F=  0  ?14
      DynType=LM  MSlp=?( ~0[1] )

    SprID=Wheels-Eng
      NegMass=Wheels  PosMass=Engine
      StaType=SI  SU=?  ST=?  XSlk=?
        X=  0  #14
        F=  0  ?14
      DynType=LM  MSlp=?( ~0[1] )

Output Information

  OutClass=MassTS  Qty=AVDavd  Mass=*
  OutClass=SprPar  Qty=SD  Spr=*
  OutClass=FitRep
  OutClass=Model
```

Figure A.4: Model extraction input file.

```
------------------------------------------------------------------------------------------------------------------------------
 SISAME 2.0.0  Log File  ||||||||||||||||||||||||||||||||||||||||||||||||||||||  Run: 1890       Date: 2003/03/01  Time: 15:20:12
------------------------------------------------------------------------------------------------------------------------------


------------------------------------------------------------------------------------------------------------------------------
                                             |||||||  SISAME Run Summary  |||||||
------------------------------------------------------------------------------------------------------------------------------


Run Title: Full Frontal Test 1890 Model

Extraction run:   107 extracted parameters

               Timing Values
------------------------------------------
      Quantity              Time (sec)
--------------------  --------------------

Freq-Based Time Step       0.0003333333
Simulation Time Step       0.0001000000
Output     Time Step       0.0001000000
Final Output Time          0.1250000

         Metric Dimensional System
------------------------------------------
        Type                 Units
--------------------  --------------------

TIME                  SECONDS
DISTANCE              MILLIMETERS
VELOCITY              KILOMETERS/HOUR
ACCELERATION          G'S
WEIGHT                KILOGRAMS
FORCE                 NEWTONS
ENERGY                JOULES


------------------------------------------------------------------------------------------------------------------------------
                                            |||||||  SISAME Model Summary  |||||||
------------------------------------------------------------------------------------------------------------------------------


------------------------------------------------------------------------------------------------------------------------------
                                             |||||  Model Element Breakdown  |||||
------------------------------------------------------------------------------------------------------------------------------

                         Number  of  Masses               Number  of  Springs          Number  of  Forces
              ----------------------------------------   ------------------------   ---------------------------

Mdl/Veh ID    Total  Simulated Target Driven Extracted    Total  Defined Extracted    Total  Group Target Driving       Weight
----------    -----  --------- ------ ------ ---------    -----  ------- ---------    -----  ----- ------ -------   --------------

1890            3        0        3      0       0          6       0        6          1      1      0       0       1711.000
Vehicle         3        0        3      0       0          6       0        6          0      0      0       0       1711.000

------------------------------------------------------------------------------------------------------------------------------
                                                |||| 1890 Model ||||
------------------------------------------------------------------------------------------------------------------------------

  Mdl ID          Model Description                                CoordSys                   IniVel            Weight
----------   ------------------------------                       ----------               --------------    --------------

1890                                                                  +                                        1711.000

------------------------------------------------------------------------------------------------------------------------------
                                             |||  1890 Force Elements  |||
------------------------------------------------------------------------------------------------------------------------------

 Force ID          Force Description          Class        Cutoff             ConF              No. of Springs    No. of Masses
----------   ------------------------------   -----    --------------    --------------         --------------    -------------

BarrierFrc   Fixed Barrier Force                G                                                      3                0

------------------------------------------------------------------------------------------------------------------------------
                                                |||| Vehicle Vehicle ||||
------------------------------------------------------------------------------------------------------------------------------

  Veh ID         Make                  Model              Year    CoordSys                   IniVel            Weight
----------   ---------------   -------------------------  ------  ----------              --------------    --------------

Vehicle                                                               +                    56.30000          1711.000

------------------------------------------------------------------------------------------------------------------------------
                                             |||  Vehicle Mass Elements  |||
------------------------------------------------------------------------------------------------------------------------------

 Mass ID          Mass Description           Class        Cutoff             ConIF              IniVel            Weight
----------   ------------------------------   -----    --------------    --------------    --------------    --------------

OccComp      Occupant Compartment               T       60.00000          81230.26          56.30000          1203.000
Engine       Engine                             T       60.00000          46131.85          56.30000          388.0000
Wheels       Front Wheels/Suspension            T       60.00000          25655.22          56.30000          120.0000

------------------------------------------------------------------------------------------------------------------------------
                                            |||  Vehicle Spring Elements  |||
------------------------------------------------------------------------------------------------------------------------------

Spring ID         Spring Description          Class     Neg-Global Mass       Pos-Global Mass       Static Type   Dynamic Type
----------   ------------------------------   -----   ---------------------  ---------------------  -----------   ------------

Occ-Bar                                         E     OccComp                1890.Barrier               SI             LM
Radiator                                        E     Engine                 1890.Barrier               SI             LM
Wheels-Bar                                      E     Wheels                 1890.Barrier               SI             LM
Firewall                                        E     OccComp                Engine                     SI             LM
Occ-Wheels                                      E     OccComp                Wheels                     SI             LM
Wheels-Eng                                      E     Wheels                 Engine                     SI             LM
```

Figure A.5: Model extraction log file.



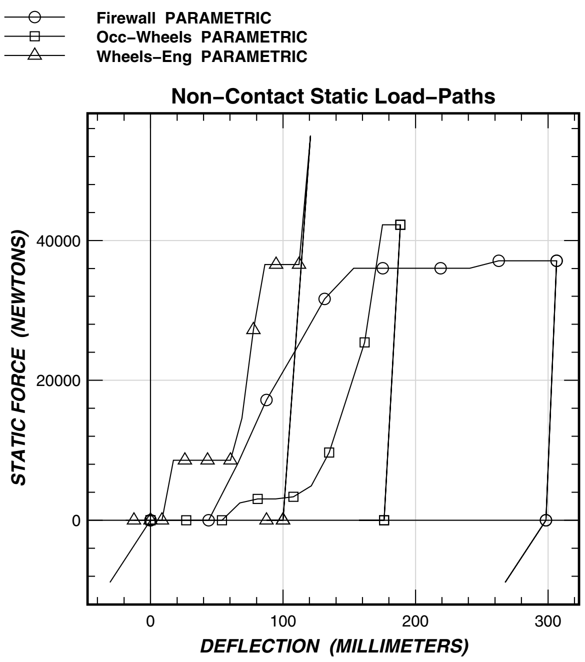
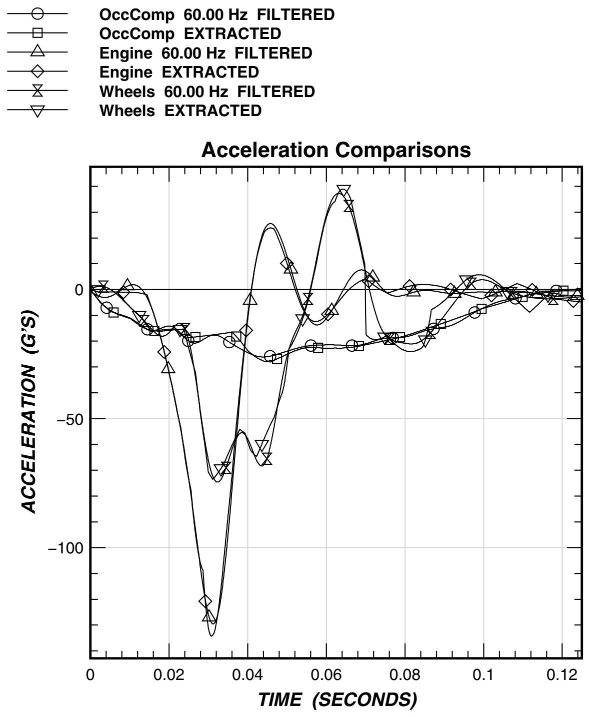
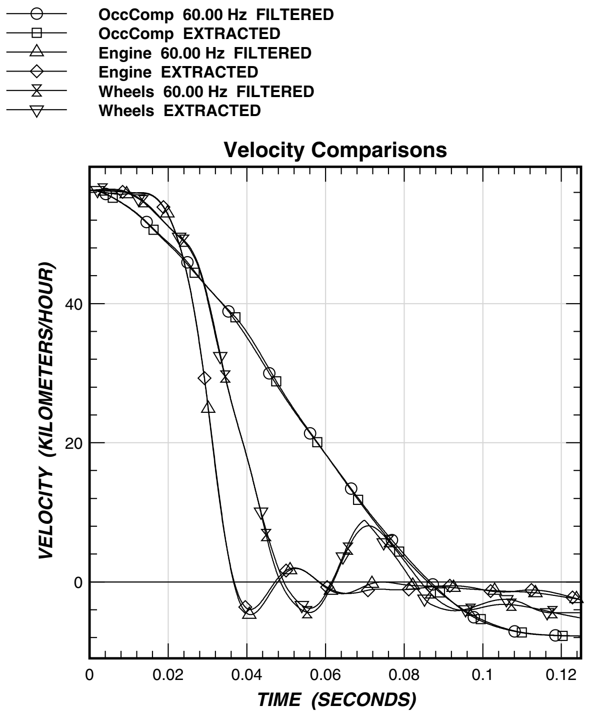


Figure A.6: Extracted static load-path characteristics.

The extracted model's effective motions are compared to those of the filtered input signals in Figure A.7. The resimulation (not shown) is very close to the effective motions, confirming that the model can accurately reproduce the test motions. This model should be predictively accurate for events sufficiently similar to the extraction event, such as full frontal vehicle–to–barrier or vehicle–to–vehicle impacts at the same or lower effective speed.

The extracted model file is shown in Figure A.8. The fit measure report (Figure A.9) shows a breakdown of the absolute and weighted differences between the instrument and effective motions for the three target masses. As is typical, the inertia force fit measure dominates the velocity and displacement domain contributions. The total fit measure is small: measures under 0.25 usually indicate a good fit for simple full frontal models.





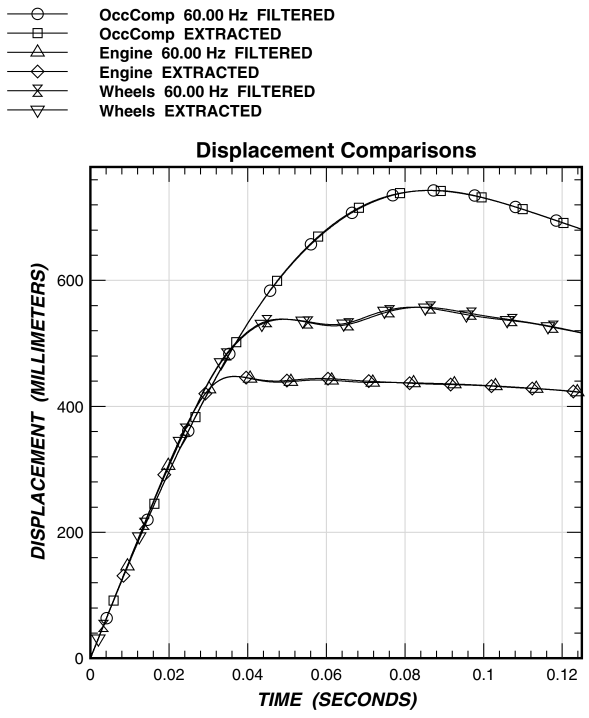


Figure A.7: Extracted model motion comparisons.

```
SISAME Input File

Run Information

  RunID=1890
    Title=Full Frontal Test 1890 Model
    DelTOut=.0001  FinTOut=.125

Model Information

  MdlID=1890
    DimSys=Metric

  VehID=Vehicle
    Wt=1711  IniVel=56.3

    MassID=OccComp  Descr=Occupant Compartment
      Wt=1203

    MassID=Engine  Descr=Engine
      Wt=388

    MassID=Wheels  Descr=Front Wheels/Suspension
      Wt=120

    SprID=Occ-Bar
      NegMass=OccComp  PosMass=1890.Barrier
      StaType=SI  SU=1050.81  ST=0
        X=            0      39.10879      78.21759      117.3264      156.4352
                195.544      234.6528      273.7616      312.8703      351.9791
               391.0879      430.1967      469.3055      508.4143      547.5231
               586.6319      625.7407      664.8495      703.9583      743.0671
        F=            0      10421.55      17497.46      21083.66      24243.73
               31494.36       36731.4       36731.4      36891.58      46276.54
               46276.54      46276.54      46276.54      46276.54      46276.54
               46276.54      46276.54      46276.54      46276.54      87372.47
      DynType=LM  MSlp=.08360393

    SprID=Radiator
      NegMass=Engine  PosMass=1890.Barrier
      StaType=SI  SU=14665  ST=0
        X=            0      31.95099      63.90199      95.85298       127.804
                159.755       191.706       223.657      255.6079      287.5589
               319.5099      351.4609      383.4119      415.3629      447.3139
        F=            0      647.6113      647.6113      647.6113      647.6113
               647.6113      647.6113      928.2786      6952.647      14876.83
               25763.73      40489.77      65714.42        115414      251252.8
      DynType=LM  MSlp=.08159942

    SprID=Wheels-Bar
      NegMass=Wheels  PosMass=1890.Barrier
      StaType=SI  SU=28037.96  ST=0
        X=            0      39.82942      79.65883      119.4882      159.3177
               199.1471      238.9765      278.8059      318.6353      358.4647
               398.2942      438.1236       477.953      517.7824      557.6118
        F=            0             0             0      441.7641      1192.533
               1805.586      2819.831      2914.142      2914.142      2914.142
                6379.86      14955.15      16684.52      16684.52      16684.52
      DynType=LM  MSlp=.1053051

    SprID=Firewall
      NegMass=OccComp  PosMass=Engine
      StaType=SI  SU=4531.36  ST=289.2645  XSlk=0
        X=            0      21.90127      43.80253       65.7038      87.60507
               109.5063      131.4076      153.3089      175.2101      197.1114
               219.0127      240.9139      262.8152      284.7165      306.6177
        F=            0             0             0      8064.777      17192.06
               24309.83      31643.37      36020.38      36020.38      36020.38
               36020.38      36020.38      37099.66      37099.66      37099.66
      DynType=LM  MSlp=.1185908

    SprID=Occ-Wheels
      NegMass=OccComp  PosMass=Wheels
      StaType=SI  SU=3416.312  ST=0  XSlk=0
        X=            0      13.47399      26.94797      40.42196      53.89594
               67.36993      80.84392       94.3179      107.7919      121.2659
               134.7399      148.2138      161.6878      175.1618      188.6358
        F=            0             0             0             0             0
               2466.701      3050.603      3050.603      3371.625      4931.658
               9674.798      17589.27      25424.31      42244.54      42244.54
      DynType=LM  MSlp=.03821379

    SprID=Wheels-Eng
      NegMass=Wheels  PosMass=Engine
      StaType=SI  SU=2655.382  ST=0  XSlk=.4303424
        X=            0       8.62403      17.24806      25.87209      34.49612
               43.12015      51.74418      60.36821      68.99224      77.61627
                86.2403      94.86433      103.4884      112.1124      120.7364
        F=            0             0      8579.553      8579.553      8579.553
               8579.553      8579.553      8579.553      14615.76      27210.37
               36543.24      36543.24      36543.24      36543.24      54980.86
      DynType=LM  MSlp=0
```

Figure A.8: Extracted model file.

```
-----------------------------------------------------------------------------------------------------------------------------------
  SISAME 2.0.0  Fit Measure Report  ||||||||||||||||||||||||||||||||||||||||||||  Run: 1890        Date: 2003/03/01  Time: 15:20:25
-----------------------------------------------------------------------------------------------------------------------------------



Fit Measure Units:

                         FORCE                NEWTONS
                         VELOCITY             KILOMETERS/HOUR
                         DISPLACEMENT         MILLIMETERS



Mass Fit Measure Legend:

                     IF  INERTIA FORCE
                      V  VELOCITY
                      D  DISPLACEMENT



-----------------------------------------------------------------------------------------------------------------------------------
                                                |||  Target Mass Fit Measures  |||
-----------------------------------------------------------------------------------------------------------------------------------



      Mass Name                Mass Description                Cutoff            Con Band           RMS diff         Wtd RMS diff
---------------------   ------------------------------    ----------------   ----------------   ----------------   ----------------

Vehicle.OccComp         Occupant Compartment                 60.00000      IF   81230.26           13313.62           0.1638998
                                                                            V   6.077077           0.2811865          0.04627003
                                                                            D   211.0096           0.8695274          0.004120795

Vehicle.Engine          Engine                               60.00000      IF   46131.85           8552.299           0.1853882
                                                                            V   10.70069           0.3966962          0.03707203
                                                                            D   371.5517           1.173806           0.003159201

Vehicle.Wheels          Front Wheels/Suspension              60.00000      IF   25655.22           4651.849           0.1813218
                                                                            V   19.24141           0.6487439          0.03371602
                                                                            D   668.1046           1.683922           0.002520447
                                                                                                                   ----------------

                                                                             Combined weighted RMS differences:  IF   0.1771153
                                                                                                                  V   0.03937859
                                                                                                                  D   0.003332374
                                                                                                                   ----------------

                                                                                                              Total   0.1814707
```

Figure A.9: Fit measure report.

B SISAME Modeling Guidelines

SISAME can be applied to a range of predictive, parametric, and design modeling tasks, but model extraction is a fundamentally different process than simulation and requires different skills from the modeler. Some basic SISAME modeling notes and guidelines are presented below.

B.1 Understanding Extraction

- A number of factors influence the extraction process and can limit the accuracy of a model. Accurate instrument data for all the significant mass motions are necessary because the computations SISAME performs are inherently sensitive to motion data errors. The mass weights, spring configuration, and static and dynamic aspect types must also be appropriate. The validity of the underlying assumptions of a one-dimensional event on a discretizable structure affect the usefulness of the extracted model.
- SISAME can produce predictively useful models from a set of accelerometer signals with little knowledge of the structure. Without supplementary information about the structure or additional instrument data (such as barrier forces) the model produced may not accurately capture the load distribution in the structure. The load-path "loops" needed to model vehicle structures make the instantaneous load distribution indeterminate given the accelerations of the masses. Even though SISAME considers the entire span of the event in its extractions, some indeterminacy may remain, and there may be a large space of near-optimal models that can reproduce the acceleration signals with almost equal accuracy. All of these models may produce predictively accurate motions in the range of events of interest. A small amount of supplementary information, or extraction from more than one event, may suffice to remove the indeterminacy and yield an accurate structure.
- To obtain the most complete and accurate models, instrument locations should, ideally, capture all significant masses that will undergo distinct motions.
- Simulations of events more severe than the extraction event require extrapolated load-path behavior and are of lower predictive accuracy, in general.
- Simulations of events involving significantly different failure modes than the extraction event are of lower accuracy.

B.2 Preparing the Instrument Data

- Careful assessment and refinement of the test instrument data using VeCor or a comparable tool is vital to the extraction of quality models, especially for more complex models. The load-path deflections are typically computed from the difference between doubly integrated accelerometer signals making them sensitive to even modest signal errors.
- The filtering method uses a portion of the motion data outside the extraction time span, if available, for acceleration/force tail smoothing. The best results will be obtained if the entire span of the motion/force data is input to SISAME.

B.3 Model Configuration

- When the set of active load-paths is not completely known additional load-paths can be included in preliminary extractions to help identify the important load-paths. Nonintuitive load-paths are sometimes important to the model because the composite nature of the masses and load-paths can obscure their interconnections and interactions. A common mistake is to attempt to channel a number of loads through a mass when load-paths that bypass the mass are also active.
- Load-paths representing large or composite structures in coarser models may benefit from the use of `AnySlope=True` and will generally benefit from larger dynamic contributions. This will give less physically-idealized models that may be more predictively accurate for the range of events of interest.
- More complex load-path behaviors than those provided directly by SISAME static and dynamic aspect types can be obtained by combining multiple springs in parallel, or in series connected by low-weight mass nodes.
- In keeping with the nature of lumped-parameter modeling, SISAME is intended for relatively low frequency extractions. Extractions at high frequencies can greatly increase the computation time, usually with little or no benefit to model quality. For vehicle crash test modeling applications, a 0.1–0.2 second time span and a 100 Hz or less cutoff frequency are generally appropriate.
- Stiffer springs and higher cutoff frequencies cause SISAME to select a smaller default simulation time step, thus slowing the extraction and/or simulation.

B.4 Weight Extraction

- Mass element weights can be extracted in a separate **weight extraction** run containing only the masses and the target force(s) containing them.
- The simplest weight extractions use a single target force for the set of all masses. For barrier impacts this target force is the total barrier force. For vehicle–to–vehicle impacts a zero target force is used for the set of all masses in both vehicles.

- Weight extractions without weight estimates or bounds provide an assessment of the completeness of the motion instrumentation and assist in the selection of model extraction cutoff frequencies. If significant masses experiencing distinct motions are not captured by the instrumentation the force match will be of lower accuracy, and a lower frequency model extraction will be appropriate.
- A single target force match alone cannot, in general, clearly identify accurate mass weights, and the use of weight estimates based on engineering knowledge is recommended. The single target force system, even in the presence of complete and accurate instrument data, is best used as an adjunct to weight estimates.
- Specified run and vehicle weights are used to generate equality constraints on the extracted mass weights.
- Combining weight and spring extraction in the same SISAME run provides some coupling of the weights and springs parameters, which may or may not be desirable. SISAME can find a somewhat better fit to the cumulative data but it may excessively skew the spring solution in doing so if there is not sufficient information to clearly imply a unique model. The balance between force and motion target fits may need to be adjusted via ***ConF*** and/or ***ConIF***. This type of run also allows the inertia force of mass elements to be split over multiple target forces containing the extracted springs attached to the extracted-weight masses.

B.5 Dynamic Extraction

- Simultaneous extraction of both dynamic and static spring aspects is supported. A crash data set may not contain sufficient information to clearly distinguish the dynamic and static contributions, in which case estimates and/or bounds on the extracted dynamic parameters may improve the quality of the model. Multiple-event extractions (using SISAMEM) with different speeds events provide a better basis from which to obtain robust dynamic parameters.
- Dynamic effects can partially compensate for the inertial lag not captured in a coarsely discretized model. This can cause larger dynamic contributions to be extracted for larger, composite load-paths.

B.6 Controlling and Refining Extractions

- Extractions can be refined by improving the data preparation or altering the model configuration, mass weights, defined static and dynamic load-path parameters, filtering frequencies, or target confidence bands. Extractions can also be improved by using supplementary information such as additional defined parameters, additional target data, or estimates and/or bounds for extracted parameters.
- Experience has shown that a poor extraction fit is less likely to be caused by the mass weight distribution or dynamic effects than it is to the data quality and preparation or the lack of key load-paths.

- The use of parameter estimates and bounds is valuable when partial information about the extracted structures is known. This can help SISAME identify the physical structure more accurately when the instrument data alone has many near-optimal models of the chosen configuration.
- Target equations used in the constrained, least squares solution procedure for the extracted parameters are weighted to balance the various errors. Weighting is based on confidence bands on the target data that should be assigned so that target equation errors are within the band with some uniform certainty level, for example 90%. Some target equations, such as the target mass inertia force and target force matches, are repeated at each simulation time step and are weighted with respect to a root mean square (RMS) error over the full time span. In these cases, the confidence bands should also be selected with respect to the RMS error.
- Target forces are particularly useful for helping the partition of forces in the "looped" load-path models that are required for vehicle modeling. The closed loops of extracted springs in such models make an instantaneous force solution problem singular, and in practice the mass motions alone may not strongly identify a unique force partition within the structure. Providing a target force for one load-path in each loop would completely remove the singularities. Extraction from multiple events (using SISAMEM) is another approach to improving the structure identification.
- Certain features of extraction models do not allow direct solution for the optimal model, and SISAME uses a fixed-point iteration procedure to attempt to converge on a self-consistent that, in almost all cases, will be a near-optimal model.
- For **faster extractions** a larger than default ***`DelTSim`*** can be used. Smaller than default ***`MaxIter`***, ***`ConPD`***, and ***`MultPD`*** values and a larger than default ***`ConvTol`*** value can reduce the number of iterations. Decreasing the ***`Cutoff`*** value(s) will speed each iteration. Setting `ConV=N` and `ConD=N` will speed up each iteration but may worsen the velocity and displacement domain fits and the drift of the resimulation.
- For **better extracted models** a default or smaller ***`DelTSim`*** should be used. Larger than default ***`MaxIter`***, ***`ConPD`***, and ***`MultPD`*** may be helpful by allowing more iterations. A smaller than default ***`ConvTol`*** may slightly improve the solution. The ***`ConIF`***, ***`ConV`***, ***`ConD`***, and ***`ConF`*** confidence band values can be adjusted to better balance the fit among the target motion domains and forces.
- Extracted static SE and SI aspects can be "smoothed" by decreasing the ***`ConSS`***. This may be particularly helpful for `AnySlope=True` aspects.

B.7 Assessing the Extracted Model

- The model is extracted to best fit the filtered target signals, so the filtered outputs produced by SISAME are the best basis for comparison with model-derived outputs. The output sampling rate may not be sufficient for accurate integration so obtain acceleration, velocity, and displacement outputs directly from the SISAME extraction run, as needed.
- The effective motion outputs are based on the accelerations that would be produced by the extracted and defined spring forces if driven through the extraction event deflection histories. Comparing these to the filtered instrumented motions shows how closely the SISAME model matches the extraction event in the domain that the solution is performed and without the

cumulative drift of resimulation. If this fit is acceptable the extracted model should be exercised in a resimulation run to assess its baseline accuracy before conducting predictive simulations of other events. A poor resimulation match indicates that the data, configuration, weights, defined static and dynamic behavior, target confidence bands, and/or discretization assumptions need to be reexamined.

- Extracted spring force and energy outputs, group force outputs, and effective motions are based on the springs being driven by the instrumented mass motions. The extracted model is an optimal but inexact fit to the model/data combination, so the forces from an extraction run do not sum exactly to the instrumented mass inertia forces.
- Even small deflection errors can lead to a poor motion fit prior to and during inelastic load-path unloading. Further refining the instrument data can improve the extraction results.

B.8 Common Problems

- SISAME catches numerous model specification errors and issues detailed messages to assist in correcting a problem. The log file should be checked for warnings even when a run completes without error.
- If SISAME reports that an ill-conditioned least squares system occurred during an extraction possible causes include:
  - Revered positive–negative mass attachments on a spring with a nonsymmetric static SI aspect such that the spring experiences primarily tension instead of compression (a warning about this condition is generated). Symmetric SI aspects eliminate the possibility of reversed attachments and are appropriate when yielding may occur in either direction.
  - Inactive extracted parameters, possibly due to model configuration errors.
  - Attempting to extract static and dynamic parameters on load-paths where the data does not sufficiently distinguish the static and dynamic contributions.
  - Inherent ill-conditioning in the model–data combination.

  Decreasing `ConPC` can improve the conditioning, but the extracted model should be scrutinized. In many cases the ill-conditioning is inherent in the extraction problem and will not seriously degrade the resulting model.

B.9 Predictive Simulations

- Simulating extracted models in less severe conditions than the extraction event may not exercise some of the extracted parameters, using, in effect, a less complex model.
- Simulating extracted models in more severe conditions than the extraction event may entail significant extrapolation of the extracted springs, thus reducing the reliability of the simulation's accuracy.

- Validation extractions using data generated by simulating analytic models are also subject to limitations which can distort the extracted springs and, usually to a lesser extent, reduce the predictive accuracy of the extracted model. Some of the causes of these problems are:
  - Extracted springs with different deflection breakpoints than the corresponding simulated spring.
  - Cutoff frequencies low enough to eliminate or distort model features (particularly "sharp" features like failures).
  - Signal tail smoothing (alters the input signal tail values). `ZeroSm=N` and `EndSm=N` should be used with analytically generated signals.
  - Spring parameters that are inactive during the simulated event (so that constraints and conditioning determine their extracted values).
- For **faster simulations** a larger ***`DelTSim`*** than default can be used (up to a factor of 100 can generally be used without serious loss of precision).
- Very small ***`DelTSim`*** values can cause cumulative numeric drift errors that degrade the simulation accuracy.
- Artificially large load-path stiffness parameters, such as S_U or S_T in static SI aspects, can be extracted in some cases and will force a small default ***`DelTSim`*** value in subsequent simulations of the extracted model. Controlling these parameters (see Section 3.6) usually has little cost to the overall fit of the model and can greatly speed the simulations.

References

[1] Hollowell, W.T. (1986). *Adaptive Time Domain, Constrained System Identification of Nonlinear Structures.* Doctoral thesis, University of Virginia.

[2] Larsen, S. and Shaw, L.M. (1980 Revision). *Technical Justification and User's Manual for Fiat Automotive Structural Crashworthiness Computer Program A (Revision No. 3).* Dynamic Science Inc. Report No. 4890–76–51.

[3] Larsen, S. and Shaw, L.M. (1980 Revision). *Technical Justification and User's Manual for Fiat Automotive Structural Crashworthiness Computer Program B (Revision No. 3).* Dynamic Science Inc. Report No. 4890–76–52.

[4] Lawson, C.L. and Hanson, R.J. (1974). *Solving Least Squares Problems.* Prentice–Hall, Inc., Englewood Cliffs, N.J.

[5] Luenberger, D.G. (1973). *Introduction to Linear and Nonlinear Programming.* Addison–Wesley, Reading, Mass.

[6] Mentzer, S.G. (1982). *Analysis and Enhancement of the Fiat Methodology for Vehicle Crash Modeling and Simulation.* Automated Sciences Group, Inc. Report No. ASG–TR–82–18.

[7] Mentzer, S.G. (1984). *Structural Impact Simulation and Model Extraction: The SISAME and MDOP Programs.* Automated Sciences Group, Inc. Report No. ASG–TR–A140.05.

[8] Mentzer, S.G. *et al* (1992). *The SISAME Methodology for Extraction of Optimal Structural Crash Models.* SAE Report No. 920358.

[9] Mentzer, S.G. (1991). *The SIMFIL Program: Simultaneous Filtering.* U.S. Department of Transportation Report No. DOT HS 807 731.

[10] Mentzer, S.G. (1995). *The SimFil Program v.2: Simultaneous Filtering.* U.S. Department of Transportation Draft Report.

[11] Mentzer, S.G. (1995). *The VeCor Program.* U.S. Department of Transportation Draft Report.

[12] Prasad, P. and Padgaonkar, A.J. (1981). *Static–to–Dynamic Amplification Factors for Use in Lumped-Mass Vehicle Crash Models.* SAE Report No. 810475.

[13] Shapiro, J.F. (1979). *Mathematical Programming: Structures and Algorithms.* Wiley, New York.

Index