AN EFFICIENT METHOD FOR THE OPTIMIZATION OF VISCOPLASTIC CONSTITUTIVE MODEL CONSTANTS

by

ERIK A. HOGAN

A thesis submitted in partial fulfillment of the requirements
for the Honors in the Major Program in Aerospace Engineering
in the College of Engineering and Computer Science
and in The Burnett Honors College
at the University of Central Florida
Orlando, FL

Spring Term 2009

Thesis Chair: Dr. Ali P. Gordon

Thesis Approval

University of Central Florida
The Burnett Honors College

I hereby recommend that the thesis prepared under my supervision by
Erik A. Hogan
Entitled: *An Efficient Method for the Optimization of Viscoplastic Constitutive Model Constants*
be accepted as partial fulfillment of the requirements for Honors in the Major

---

_____ _____

Ali P. Gordon, Ph.D. Date
Thesis Committee Chair
Department of Mechanical, Materials and Aerospace Engineering

Recommendation Concurred:

_____ _____

Alain Kassab, Ph.D. Date
Department of Mechanical, Materials and Aerospace Engineering

_____ _____

Christopher Geiger, Ph.D. Date
Department of Industrial Engineering and Management Systems

_____ _____

David Nicholson, Ph.D. Date
Department Chair
Department of Mechanical, Materials and Aerospace Engineering

_____ _____

Alvin Y. Wang, Ph.D. Date
Dean
The Burnett Honors College

The committee, the college, and the University of Central Florida are not liable for any use of the materials presented in this study.

ABSTRACT

Constitutive modeling is a method that is useful in providing precise predictions of material response in components subjected to a variety of operating conditions. A process for optimizing the material constants of the Miller constitutive model for uniaxial modeling was developed and implemented in an automated optimization routine. Generally, up to twenty experiments simulating a range of conditions are needed to identify the material parameters for the model. The research sought to minimize the amount of empirical data that is necessary for optimization, aiming to reduce the costs and time necessary to carry out this procedure for more expensive classes of materials. The ultimate goal was to develop a robust method for determining the material constants of a viscoplastic model using a minimum amount of experimental data. An automated optimization routine was implemented into a program, referred to as uSHARP, developed as part of the research to determine constitutive model parameters. Central to the method was the use of more complex stress, strain, and temperature histories than are traditionally used, allowing for the effects of all material parameters to be captured using as few tests as possible. By carrying out successive finite element simulations and comparing the results to simulated experimental test data, the material constants were determined from 75% fewer experiments. By reducing monetary costs and time required, this procedure will allow for a more widespread application of advanced constitutive models in industry, allowing for better life prediction modeling of critical components in high temperature applications.

DEDICATIONS

This work is dedicated to my parents; without their endless support and encouragement I never would have made it this far.

Also, to the guys from 5112: thanks for being there to help me wind down after long day of hard work. As we part ways after four years together, I wish you both the best and hope we remain close as the years pass.

ACKNOWLEDGEMENTS

I would like to express my gratitude for the assistance of my committee members, Dr. Gordon, Dr. Kassab, and Dr. Geiger, for their assistance and input during the completion of this project. A special thanks goes out to my committee chair, Dr. Gordon, for all of his patient guidance and frequent direction.

Much of this research would not have been possible without the assistance of my peers in the Mechanics of Materials Research Group at UCF. Thank you all.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Constitutive modeling is a field of engineering mechanics that has received considerable research attention over the past several decades, with much focus on the development of unified constitutive models. Recent advances in computer processing speed have enabled the implementation of these models into finite element analysis (FEA) software applications such as ANSYS and ABAQUS, yielding accurate and cost-effective predictions of stresses and strains in structural components under a non-generic variety of operating conditions, such as creep, fatigue, and thermal-mechanical fatigue. Under these conditions, certain alloys display inelastic behavior in the form of coupled creep and plasticity, commonly referred to as viscoplasticity. A variety of material models have been developed to correlate with mechanical test data. For this research, the Miller unified constitutive model (Miller, 1976a) will be used to run simulations using the ANSYS general purpose FEA software. The purpose of the research is to develop a robust method of parameter determination, using a minimum number of experimental tests, which can be easily applied to many different types of viscoplastic constitutive models.

Traditional methods for step-by-step parameter determination require data from upwards of twenty experimental tests (Chan et al., 1986; Rowley et al., 1994; Miller, 1976b). Often, a large amount of data is needed in order to determine only a portion of the total number of material constants. This trend is evident in Fig. 1, where a large

number of creep test data were used to determine only four of the eight material parameters in the Miller model. One test type used for parameter determination is a low cycle fatigue test; these tests are typically run until stress saturation occurs, sometimes approaching one hundred cycles (Ramaswamy et al., 1985). Examples of the standard tests used in parameter determination are shown in Fig. 2. These tests were conducted on the René 80 alloy and used to determine constitutive model parameters. Typically, combinations of creep, low cycle fatigue, and tensile data at various temperatures are used for material constant determination.

An efficient, easily applied optimization scheme could drastically reduce the amount of experimental data needed to determine inelastic constitutive model parameters, reducing both monetary costs and time required to implement advanced constitutive models in industry. This would allow for more accurate preliminary design of critical components, reducing the need for expensive design modifications as product development progresses. If a parameter determination scheme is designed in such a way so as not to be specific to any particular constitutive model, then a robust method will result which will be applicable to a broad range of constitutive models and experimental test types. A more advanced application of such a formulation is the capability to design materials to achieve a particular combination of mechanical properties. It is important to note that, for this research, the focus will be on determining the inelastic material constants. The elastic constants can be determined from simple tensile tests or slightly more complicated tests which will be identified in future work.

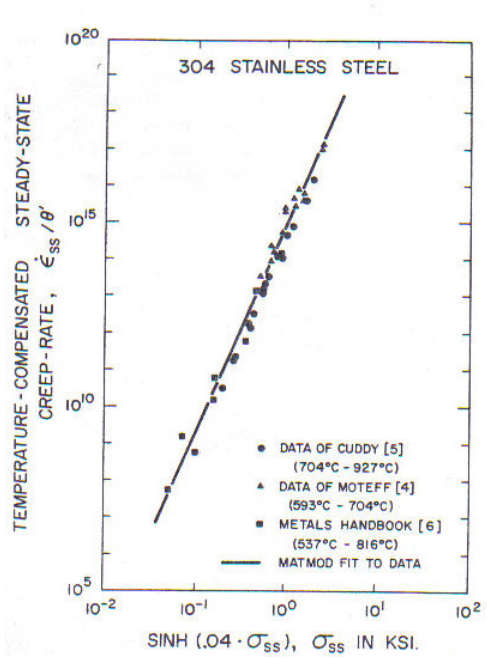


Figure 1: Large number of creep data used to determine only four of the eight material parameters in the Miller unified constitutive model (Miller, 1976b)

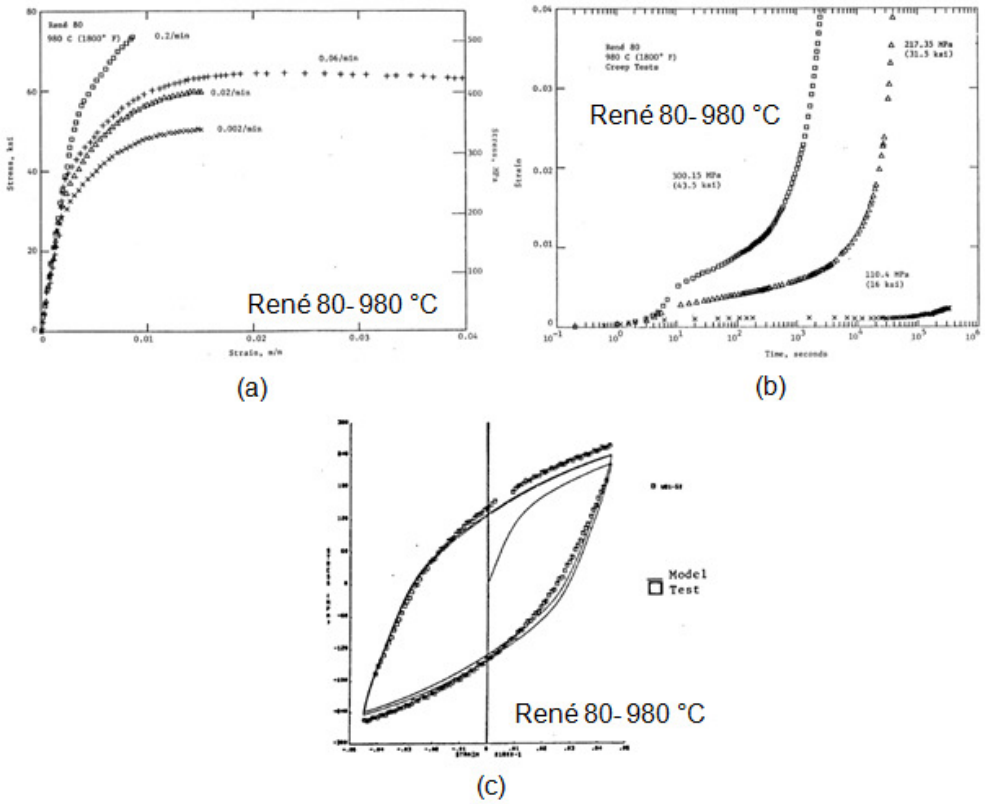


Figure 2: Standard tests types used for parameter determination include (a) tensile tests, (b) creep tests, and (c) low-cycle fatigue tests (Ramaswamy et al., 1985)

2. BACKGROUND

Viscoplastic constitutive modeling is an important technique used in the design of components in high temperature applications, such as in turbine engines (Ramaswamy et al., 1985). These material models allow design engineers to predict stresses and strains in critical components using finite element analysis software such as ANSYS, ABAQUS, etc. The results of such preliminary analyses allow for the optimization of shape geometry to improve factors of safety and reduce high stress concentrations. Finite element modeling can also be used to determine properties which allow for reasonable life prediction of components subject to a combination of cyclical loading and temperature cycling. The Miller model was designed for use in high temperature applications, and is capable of reasonably predicting material behavior under such conditions (Miller, 1976a; Miller, 1976b). Furthermore, the Miller constitutive model was extended to a multiaxial formulation for more complex geometries and loading conditions (Kagawa and Asada, 1983). Unfortunately, the use of advanced constitutive models is limited by the large amount of experimental data needed to determine material constants contained in the model and the often tedious step-by-step procedure used to determine these constants. This has confined the field of advanced constitutive modeling primarily to the realm of academia, rather than to widespread practical application in industry.

2.1 The Miller Unified Constitutive Model

The Miller model is most useful for predicting the behavior of materials in high temperature applications and is capable of modeling a wide range of material behavior including time-, rate-, and history-dependence. As a unified constitutive model, both creep and plastic strain are combined under the moniker of inelastic strain. The Miller model is convenient to use due to its low number of material constants, ease of implementation into finite element software packages, and its ability to accurately simulate a range of material behavior. There are difficulties in application of the Miller model, however. The formulation of the model includes three hyperbolic sine functions, as shown in Eqs. 2-4. As a result, numerical simulation of the Miller model is computationally expensive (Ramaswamy et al., 1985). Furthermore, the model is only capable of reasonably predicting behavior of materials that work-harden considerably (Miller, 1976a). This limits its applicability to certain classes of materials, such as steels.

The Miller unified constitutive model satisfies several important criteria that make it appropriate for testing an optimization procedure such as this one. Since the verification of the optimization procedure depends heavily on simulations conducted by ANSYS, it is important that the model be suitable for implementation into a finite element analysis program. Fortunately, the Miller model has already been coded to work with ANSYS by way of a user-programmable function (UPF) developed at UCF. This is a custom feature that allows for a user to define a constitutive model that will be used by ANSYS to predict material behavior. Another important factor is the dependence of material constants on temperature. If several of the constants changed with temperature, an attempt at optimization would be further complicated by the need for more isothermal tests at various temperatures. This is clearly in opposition with the overall goal of

reducing the number of tests required. In the Miller model, none of the material constants are temperature-dependent (Miller, 1976a), meaning fewer tests will be required. A third consideration that makes the Miller model suitable is that only cyclical and creep (or stress relaxation) data are needed to determine the constants. Both of these tests are relatively simple and can be conducted with equipment that is readily available.

The theory used in the derivation of the Miller model limits its application to alloys that work-harden considerably, such as aluminum alloys and steels (Miller, 1976a). More specifically, the model focuses on the effects of two particular types of hardening: isotropic and kinematic. These two phenomena reflect changes in material strength with respect to deformation history. Isotropic hardening implies that material strength remains constant in all directions. For example, straining a material specimen in the tensile direction will strengthen it both in the tensile and compressive directions. Kinematic hardening, on the other hand, reflects different material strengths with respect to different orientations in a material. For example, straining a specimen in the tensile direction may strengthen it in the tensile direction but weaken it in the compressive direction (Miller, 1976a). These effects are accounted for in the strain rate equation. Essentially, the model predicts that the strain rate of a material under loading is a function of the difference between the applied stress (σ) and the kinematic hardening variable, divided by the isotropic hardening variable. Kinematic hardening and isotropic hardening are denoted by the variables R and D, respectively, in the basic strain rate equation

$$\dot{\epsilon} = f\left(\frac{\sigma - R}{D}\right). \tag{1}$$

In this formulation, the variable *R* and *D* take the forms of internal state variables (ISVs) which describe the internal state of the candidate material. Even though these so-called ISVs cannot typically be measured through direct observation, they are necessary to describe the nature of the microstructure and its evolution. Since the values of *R* and *D* change during the deformation history of a material, equations are needed to model the behavior of these two factors. These equations are not solely dependent on the strain or strain rate, but rather depend on the entire previous deformation history (Miller, 1976a). It is through the evolution of *R* and *D,* called the back stress and drag stress respectively, that viscoplastic effects are captured by the Miller model.

An example of the evolution of the back and drag stresses for a constant amplitude strain controlled cyclic fatigue test can be seen in Fig. 3, as well as the corresponding stress history. As the number of cycles increases, the peak stress amplitude increases, indicating the occurrence of cyclic hardening. After approximately 6000 seconds, the peak amplitude remains nearly constant, indicating cyclic hardening has reached saturation. This cyclic hardening effect is captured by the drag stress. It is evident that throughout the duration of cyclic hardening, the drag stress increases until it reaches a steady state value after cyclic saturation has occurred. The back stress evolution is much more dynamic, following the cyclic nature of the stress response. As the number of cycles increases, the peak amplitude of the back stress decreases slightly, before reaching a steady state value.

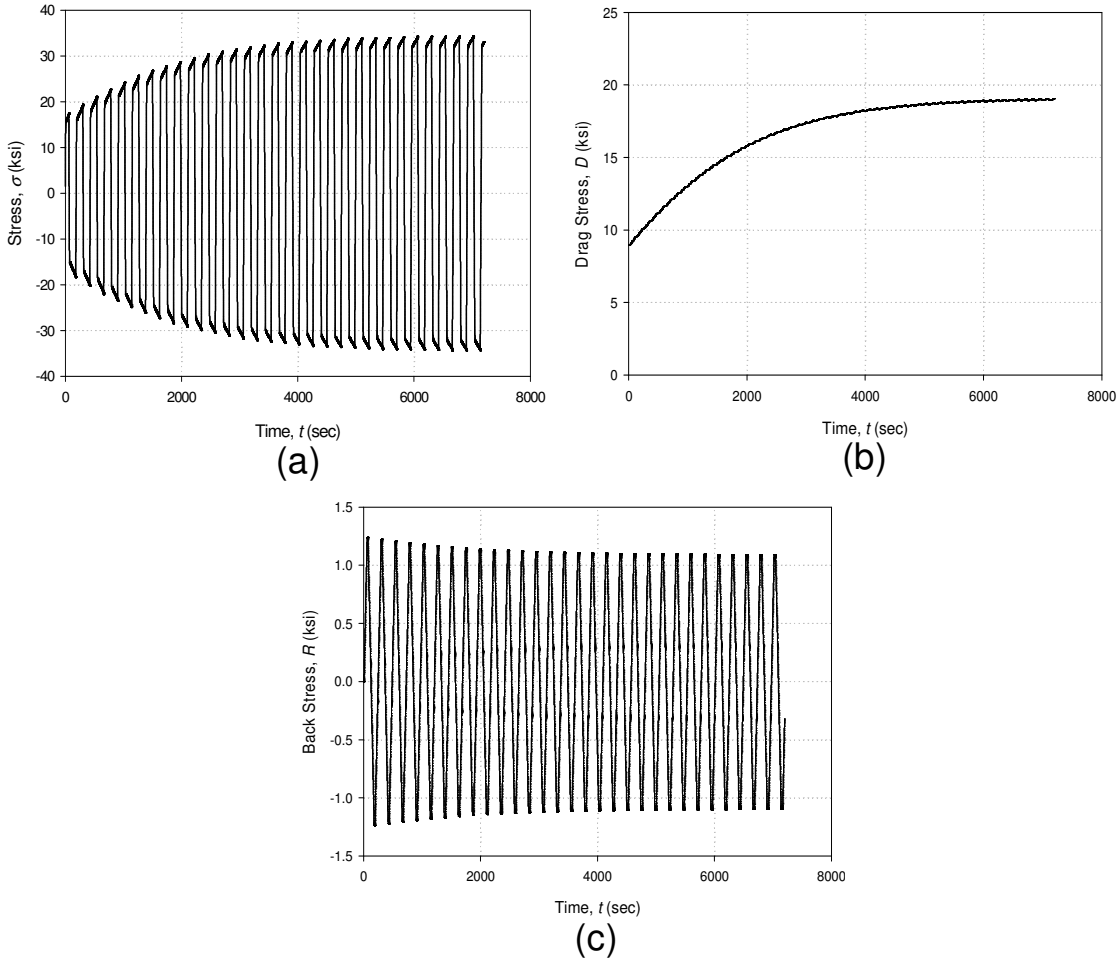


Figure 3: (a) Stress history, (b) drag stress evolution, and (c) back stress evolution for a constant amplitude strain-controlled cyclic fatigue test

In order to derive a usable function for the strain rate, steady state creep data were used as a starting point. Equations for the evolution of *R* and *D* were developed using warm working and work-hardening theory. The end result is the following set of three differential equations that are capable of predicting both steady state and transient material behavior (Miller, 1976a), i.e.,

$$\dot{\epsilon} = B\theta' \left\{ \sinh\left[\left(\frac{|\sigma - R|}{D} \right)^{1.5} \right] \right\}^n sgn(\sigma - R) \tag{2}$$

$$\dot{R} = H_1\dot{\epsilon} - H_1 B\theta'[\sinh\ (A_1|R|)\ ]^n\ sgn(R) \tag{3}$$

$$\dot{D} = H_2|\dot{\epsilon}|\ \left[C_2 + |R| - \left(\frac{A_2}{A_1}\right)D^3\right] - H_2 C_2 B\theta'[\ \sinh(A_2 D^3)]^n. \tag{4}$$

The θ' term is the only temperature-dependent factor in the Miller model. Its value is dependent on the current temperature of the material (*T)* and the melting temperature (T_m) of the material. It can be expressed as

$$\theta'(T) = \begin{cases} \exp\left\{\left[\frac{-Q}{0.6kT_m}\right]\left[ln\left(\frac{0.6T_m}{T}\right)+1\right]\right\}, & T \le 0.6T_m \\ \exp\left(\frac{-Q}{kT}\right), & T \ge 0.6T_m \end{cases} \tag{5}$$

where *k* is the ideal gas constant. Contained in these three differential equations are eight viscoplastic constants that must be determined, as seen in Table 1. Also contained in Table 1 are ranges for the constants available in literature (Miller, 1976b; Kagawa and Asada, 1983; Ramaswamy et al., 1985). None of these parameters are temperature-dependent; in fact, the only factor that temperature affects is the form of the θ' term. The testing procedure that has historically been used to determine these constants is outlined in Miller (1976a). It must be noted that two more variables, *A* and C_1, are introduced during the process of determining these variables. They do not factor into the above equations, but are used to determine A_1 and A_2.

Table 1: Material constants which need to be determined for the Miller model.

Material Constants	A_1	A_2	B	C_2	H_1	H_2	n	Q
Units	(ksi^{-1})	(ksi^{-3})	(sec^{-1})	*(ksi)*	*(ksi)*	*(unitless)*	*(unitless)*	*(cal/mol)*
Typical Value Ranges	0.8-0.93	$7.42x10^{-5}$-$5.94x10^{-3}$	$1.03x10^{14}$-$1.0x10^{15}$	0.1-50	280-10,000	100	1.60-5.8	91,000-104,600

The Miller model has been applied to several different materials, including type 304 stainless steel (Miller, 1976b; Kagawa and Asada, 1983) and Hastelloy X (Ramaswamy et al., 1985). The ability of the model to predict the behavior of type 304 stainless steel can be seen in Fig. 4. It is evident that the tensile simulations more closely match experimental data at higher temperatures. The reason for this is not immediately clear, and requires further investigation (Miller, 1976b). In the case of the cyclic fatigue simulation, the yield strength predicted by the model is somewhat higher than that of the experimental data, but the model still performs adequately in characterizing the response of the material. Fatigue data were taken from Miller (1976b) and the simulation was performed using an implementation of the Miller model with the ANSYS finite element software package.

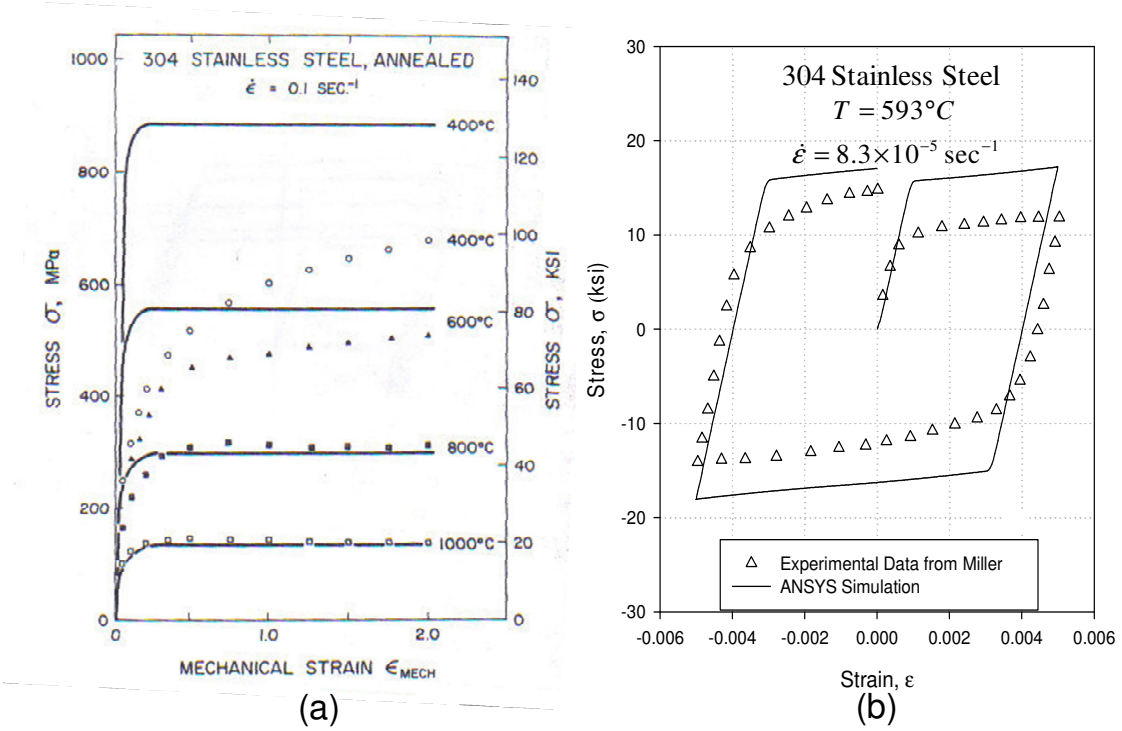


Figure 4: Application of the Miller model to predict behavior of type 304 stainless steel for (a) monotonic loading and (b) first cycle of fatigue test (Miller, 1976b)

2.2 Literature Review

Due to the importance of parameter determination in constitutive modeling, much research has been done towards solving the problem (Adebanjo, 1993; Cao and Lin, 2008; Kawasaki et al., 1998; Mustata and Hayhurst, 2005; Mulyadi et al., 2006). Upon review of the literature, it is evident that a widespread trend in parameter estimation is the development of an optimization procedure relevant only to a particular model or problem. Lacking is the existence of a procedure that is easily implemented to work with different constitutive models. Additionally, there seem to be significant faults when it comes to the development of objective functions for parameter determination. In nearly every case, the researchers have employed functions which compare simulated data with experimental data in order to determine the "best fit," without consideration to certain cases which might cause discrepancies during the process of parameter determination. In only one case (Cao and Lin, 2008) do the authors account for the differences in numbers of data points in different experimental data sets. The other investigators ignore the bias introduced into least squares calculations caused by higher numbers of data points when multiple curves are used simultaneously; this might lead to some of the fit discrepancies present in literature. Furthermore, minimal consideration is given to the application of an objective function which can allow for data with different orders of magnitude to be used simultaneously. Kawasaki and coworkers (1998) use a very basic weighted objective function of the form

$$f(\mathbf{x}) = \sum_{j=1}^{n} \sum_{i=1}^{m} w_{ij} \left| \sigma_{exp}(\varepsilon_{ij}) - \sigma_{pred}(\varepsilon_{ij}, \mathbf{x}) \right|, \tag{6}$$

where σ_{exp} and σ_{pred} are the experimental and predicted stress values, respectively. The i and j terms represent the number of data points in the dataset, and the number of datasets being used. The function employs weighting values, w_{ij}, to provide higher consideration to certain parts of the material response spectrum. Use of this objective function does not account for different numbers of data points in different datasets, nor does it allow for the simultaneous use of datasets with different orders of magnitude. Furthermore, the authors do not describe how the weighting values were determined, nor do they provide the weight values used during the parameter determination process.

In Adebanjo (1993) the author forgoes the use of an objective function altogether, relying on a visual "best fit" to determine the parameters. This removes any sense of objectivity and eliminates repeatability. It also means that much user involvement is required, introducing the possibility of human error. There is no way to quantify the caliber of the fit when using a visual method. Additionally, few of the papers address the need to normalize different data types, such as stress and strain, to the same order of magnitude so two different test types can be used concurrently in parameter determination.

Prior research has relied on the implementation of several different algorithms to minimize their respective objective functions and determine the material constants. Indeed, many optimization algorithms exist for solving these highly non-linear problems. What is not addressed is the quality of the experimental data used to determine the material parameters. No attempt is explicitly made to reduce the number of experimental tests needed to determine the material parameters. The authors do not examine or discuss whether or not the tests that were used to determine the material constants were broad

enough in scope to capture the effects of all the constants. It is possible that in some cases, a value was obtained for material parameters that would not be capable of accurately predicting material behavior in cases outside of the test matrix used in their determination. For example, in Cao and Lin (2008) only tensile test data were used to obtain a fit for the parameters of a unified constitutive model which was designed to predict material behavior under many different conditions. It may be possible that these tensile-based values do not accurately reflect the behavior of the material under cyclic loading, and this issue is not addressed by the authors.

It is evident from the literature review that constitutive modeling is an important research area, and the determination of material parameters is a very important consideration. The development of a truly robust method for parameter determination has remained elusive, but each paper contributes something to the cause. It seems, however, that too much focus is being placed on the performance of the optimization algorithms being used (evolutionary algorithms, downhill simplex method, etc.) and their performance, rather than on sound, widely applicable theory for parameter determination. The performances of such optimization algorithms are well established; they have been applied to many difficult-to-solve problems in the past (Corana et al., 1987; Hooke and Jeeves, 1961). Constitutive model parameter determination is essentially a highly non-linear minimization problem, and several different algorithms exist that perform adequately (Mustata and Hayhurst, 2005; Mulyadi et al., 2005; Kawasaki et al., 1998). Rather, more focus needs to be placed on developing a sound method - a simple yet effective objective function, automatic weighting routines, and a standardized procedure that can be used on multiple experimental test types simultaneously and without bias.

None of the researchers address the problem of minimizing necessary experimental test data to allow for a broader application of constitutive models. Traditionally, several types of experimental tests are used. Tensile tests conducted at various temperatures are commonly used. Cyclic fatigue tests are another standard test type used for parameter determination. Typically, these tests are run until cyclic saturation has occurred, and numerous tests at various temperatures are needed. Creep tests are a third standard test type. As shown in Fig. 1, the results from many different creep tests may be needed to determine constitutive model parameters. Certainly, automated parameter determination methods have successfully been implemented, but no focus has been made on developing a method that can be used easily and cost effectively by a wide range of users in industry, allowing for more accurate preliminary design using finite element analysis. The identification of a small number of tests capable of distinctly capturing the effects of the material constants, and as a result, allowing for the determination of accurate values without requiring an excessive amount of data would be a very significant advance in the field of parameter determination. Implementation of these complex test histories into an automated optimization routine using a robust objective function and automatic weighting values will greatly simplify the parameter determination process, resulting in a more efficient method for the optimization of material constants.

2.3 Type 304 Stainless Steel

Application of the Miller model will be limited to type 304 stainless steel in this research. The chemical composition of this alloy is shown in Table 2. Type 304 stainless steel is frequently used in high temperature applications due to its corrosion resistance

properties, which result from the inclusion of chromium and nickel in the chemical composition (*ASM Metals Handbook*, 1991). The dependence of ultimate tensile strength (UTS) and 0.2% yield strength on temperature is shown in Fig. 5. Both of the material properties decrease in value as temperature increases. Similarly, the elastic modulus decreases as temperature increases, as shown in Fig. 6. The ultimate tensile strength must be considered in the design of experimental tests for parameter determination; the Miller model does not include failure criteria, so the experimental tests designed for parameter determination must avoid approaching the UTS. Likewise, the fatigue life of type 304 stainless steel, shown in Fig. 7, must be considered during experimental design. If the necessary data is not collected before fatigue fracture occurs, the experimental data will be useless. This existing data for type 304 stainless steel will be kept under consideration while developing experiments for material parameter determination to ensure that excessive damage or premature failure will not occur in material specimens.

Table 2: Chemical composition of type 304 stainless steel (Steichen, 1973)

Carbon	Chromium	Nickel	Manganese	Silicon
0.052 (wt %)	18.92	9.52	1.1	0.52
Phosphorus	Sulfur	Molybdenum	Nitrogen	Iron
0.011	0.01	0.12	0.052	Balance

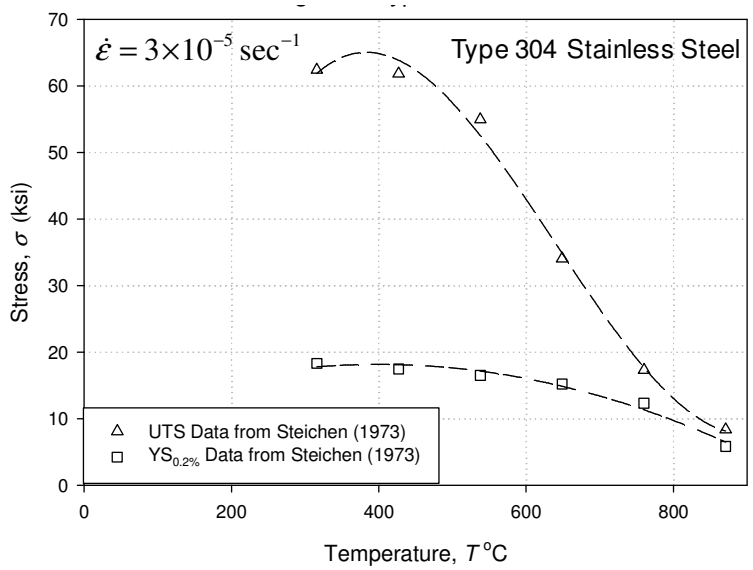


Figure 5: Temperature dependence of ultimate tensile strength and 0.2% yield strength for type 304 stainless steel

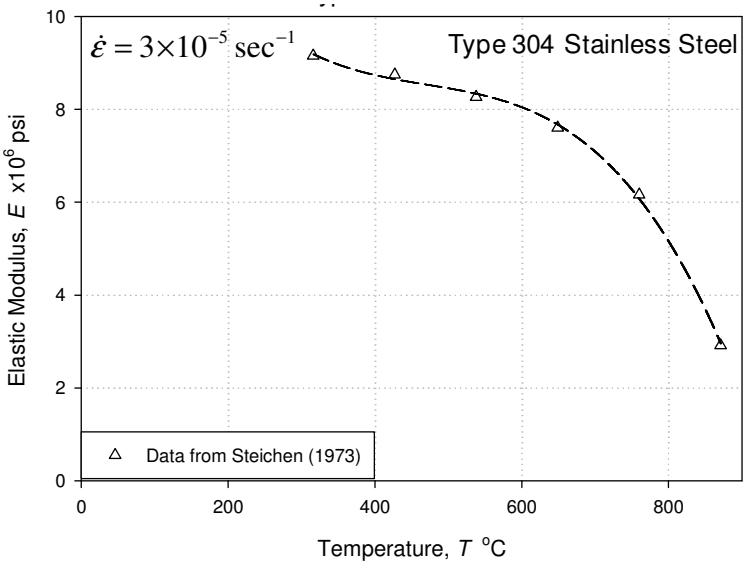


Figure 6: Temperature dependence of elastic modulus for type 304 stainless steel

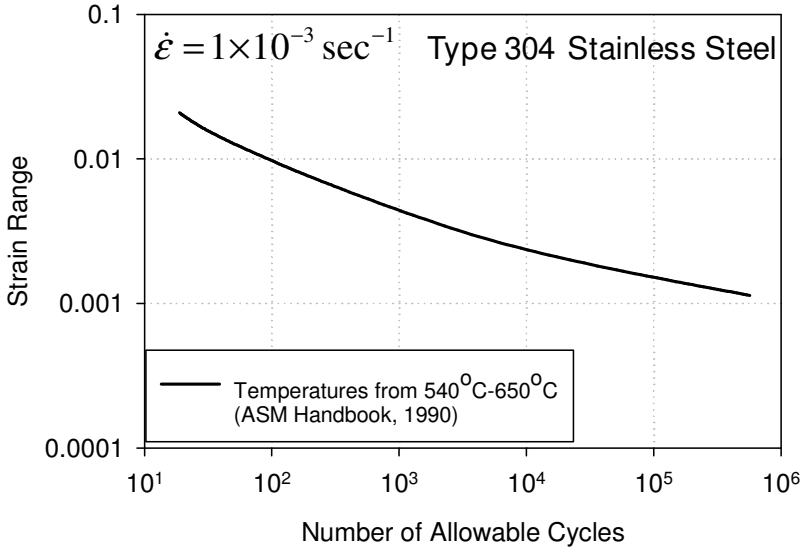


Figure 7: Fatigue life of type 304 stainless steel at elevated temperatures

3. COMPUTATIONAL METHODS

3.1 Finite Element Implementation

In order to compare theoretical results with experimental test data, the Miller model was implemented into a finite element program known as ANSYS by Patel and Stewart (2008). Since the tests conducted for parameter estimation are typically uniaxial the most appropriate model of an experimental test specimen is a single three dimensional SOLID185 element consisting of eight nodes, as presented in Fig. 8. All loads or displacements were applied to the top four nodes, while the bottom of the element had fixed displacement in the z-direction. The element cross sectional area was allowed to deform to account for the effects of Poisson's ratio. In this manner, uniaxial experimental tests were simulated.

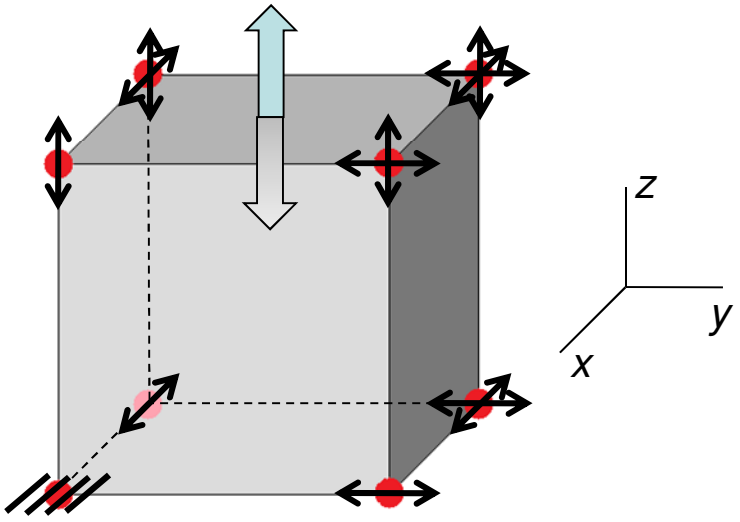


Figure 8: Eight-node element and boundary conditions used in ANSYS simulations

An ANSYS input file was required to drive the simulations that were done (see Appendix A). The command structure of the basic input file that was used is presented in

Figure 9. Since this file takes the form of a text-only document, sections of it can be modified easily by a simple FORTRAN code. When using an input file, the user must first define the test conditions and any constitutive model parameters necessary for the simulation. The input file then defines the coordinates of the nodes for the element and their degrees of freedom. These can be modified by the user to suit their needs. Since ANSYS performs the simulations using discrete time step approximations, $\Delta t^{(i)}$, there are several parameters which must be set to control the progression of time throughout the simulation. No load or displacement can be applied instantaneously, so a finite ramping time must be defined. As the solution progresses throughout the simulation, the time step that is taken by ANSYS varies depending on the conditions. This can cause problems in the solution if the time step is too large. To prevent this, the user must define a maximum time step size that is appropriate for the simulation being run. This is typically related to the strain rates that are experienced in the simulation, and should be verified before accepting any simulation data as accurate.

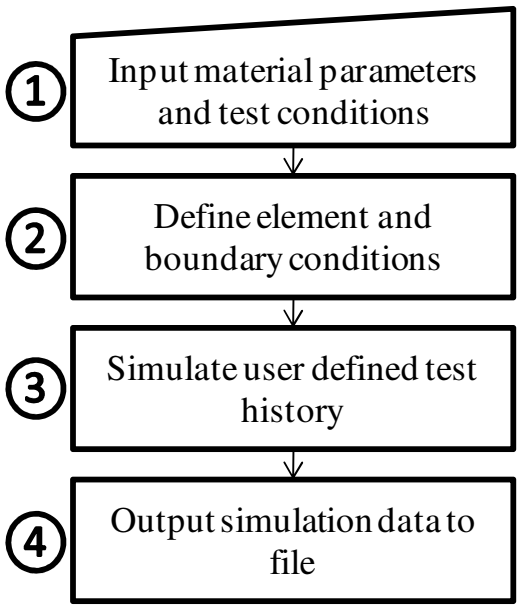


Figure 9: ANSYS input file process flowchart

3.2 ANSYS User Programmable Feature

In order to use the Miller viscoplasticity model with the ANSYS finite element software program, it was implemented into a user programmable feature (UPF). The UPF exchanges information with ANSYS in order to calculate the tensorial stress, tensorial strain, and the displacement vector, as outlined in Fig. 10. ANSYS provides the test parameters to the UPF, as well as the conditions of the stress or strain state at the current time step. The UPF uses this information to calculate the material elastic and inelastic Jacobian matrices, which are passed back to ANSYS and used to calculate the states of stress and strain for the subsequent time step. The convergence of the solution is checked by a Newton-Raphson line search method, and the process continues for the subsequent time step. Through this iterative process, the material behavior is simulated using the Miller model. It is important to note, that this procedure is not specific to the

Miller model. A UPF can be created for any compatible constitutive model and implemented into the ANSYS program.

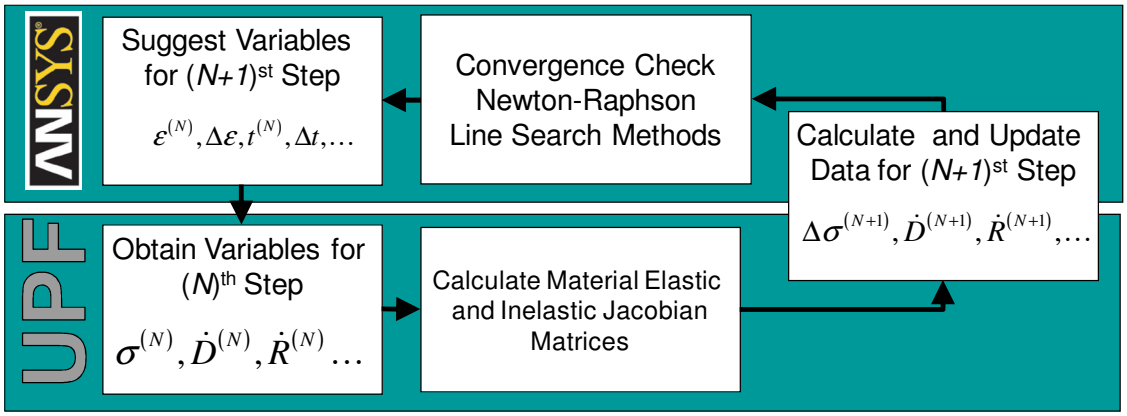


Figure 10: Simulation procedure using ANSYS and UPF

3.3 uSHARP Optimization Routine

Since the research objective was to develop a procedure for automatically determining the material constants of the Miller unified constitutive model with a minimum number of experimental tests, an automated optimization routine was needed that could compare simulations with experimental data and determine the parameters. For this purpose, a program called uSHARP was written. It is a robust program capable of automatically determining constitutive model parameters requiring only a single initial guess, which can be obtained from previously determined constants existing in literature. Any type of data can be used in the optimization process, including stress, strain, or even stress and strain rates. Before further discussion of the optimization procedure executed by uSHARP, it is pertinent to discuss the limitations of the software package.

First, uSHARP was written for constitutive models which are well understood, such as the Miller model. The program should only be used when the appropriate orders

of magnitude for the material parameters are known. The routine has demonstrated an inability to converge when the initial guess is not close to the proper order of magnitude. As such, the best initial starting point will be values that have previously been determined through the standard means for another material. This limitation, however, is a result of the optimization algorithm implemented into uSHARP, and not a problem with uSHARP itself. The application of a more robust optimization algorithm could mitigate this problem. Additionally, the constitutive model must accurately represent the behavior of the material for which it is being used. For example, the Miller model does not incorporate damage evolution; it would not be appropriate for predicting tertiary creep damage in a material.

If the uSHARP routine is used to optimize material constants, it is imperative that the test histories of the experimental and simulated data closely match. It is easy to simulate complicated test histories; it is difficult, however, to carry these out experimentally due to errors in equipment and natural imprecision present in the current usage of PID controllers. The emerging research being conducted with more advanced control systems indicates that in the near future these complicated test histories will be accurately replicated (Barnes et al., 2009).

3.3.1 Optimization Procedure

The uSHARP optimization routine makes use of an iterative procedure to determine the material constants through successive simulations. Simulations are conducted which match the test history of the experimental data and the material parameters are varied until a "best fit" is achieved. Key to this process is the use of a sophisticated objective function. This objective function relates the material constants

and the caliber of the fit between experiment and simulation. When the objective function is minimized, a best fit has been achieved and the material constants determined by the program are the constants which accurately reflect the behavior of the experimental data. The model and constants can then be used to simulate more complex service histories.

Since the uSHARP routine was designed so that multiple test types could be used simultaneously, a simple least squares function was not adequate. Different orders of magnitude present in different data types would naturally skew the value of the objective function towards higher orders of magnitude. Additionally, different data sets contain different numbers of data points, which would lead to skewing in the objective function by weighting tests with higher numbers of data points higher than tests with fewer data points. To account for these phenomena, a standard weighted least squares function was developed to allow for the simultaneous use of different experimental test types without unintended skewing (see Appendix B). The form of this objective function is as follows:

$$S = \frac{1}{n}\sum_{i=1}^{n} w(t_i)\left(\frac{y_{ex}(t_i) - y_{sim}(t_i)}{y_{ex,\max}} \times 100\right)^2 . \qquad (7)$$

Table 3: Objective function parameters

Symbol	**Meaning**
n	Number of data points in the FEM simulation
$w(t_i)$	Weight value for the data point at time index t_i
$y_{ex}(t_i)$	Experimental data value interpolated to the FEM time index t_i
$y_{sim}(t_i)$	Value of the FEM simulation data corresponding to time index t_i
$y_{ex,max}$	Maximum magnitude of the experimental dataset (positive or negative)

Equation 7 contains several different parameters, which are described in Table 3. Essentially, the least squares function accounts for the differences between the experimental and simulated data at each time index as a percentage of the maximum value of the experimental data set. This has the effect of scaling all test types to the same order of magnitude. Additionally, the total least squares value is normalized to the number of data points used in its calculation. This prevents test data containing more time indices from being weighted higher than data with fewer time indices. uSHARP can use multiple tests simultaneously in the determination of material parameters. This is done by adding the individual objective function values from each test type together, and minimizing the value of this combined objective function:

$$S_{total} = \sum_{j=1}^{m} S_j, \tag{8}$$

where m is the total number tests being combined into S_{total}.

It is important to note that the uSHARP objective function requires that the values between the data sets have the same time indices. Finite element simulations and experimental data will rarely, if ever, have identical time indices. For example, a low cycle fatigue experimental test may be sampled at 50 Hz, while a simulation of the same experiment may output data at approximately 7 Hz. To account for this, the experimental data points are linearly interpolated to the same time indices as the finite element simulation, as shown in Fig. 11. Although the different types of experiments, such as creep and low cycle fatigue tests, are not linear, the errors induced by using linear interpolation are small due to the proximity of data points. It is important to point out

that time indices are used for the *x*-values in the uSHARP objective function because each index is unique within the dataset. There are no cases where two *y*-values exist for the same time index in a single dataset. If another index for the *x*-values was used, such as strain in the case of a hysteresis loop, multiple *y*-values could exist for a single *x*-value. This would greatly complicate the process of determining the objective function, and is the reason why time is the preferred index.

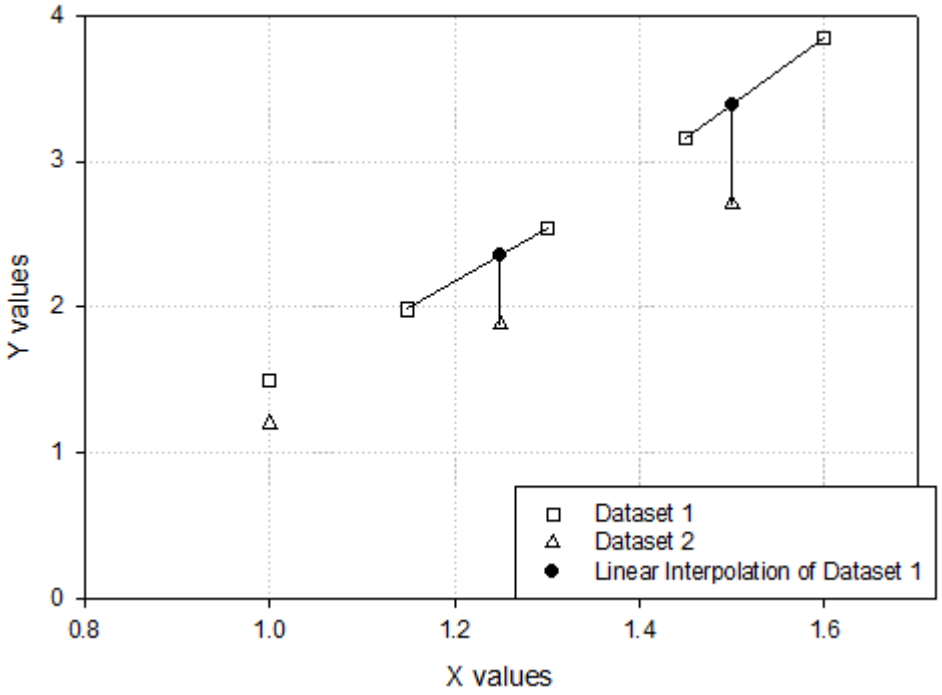


Figure 11: Example of linear interpolation used to calculate y values of Dataset 1 at x values of Dataset 2

The uSHARP objective function is a weighted least squares function. Depending on the test type being used, a different weighting function will be employed. Currently, weighting functions have been developed for two different test types: low cycle fatigue and stress relaxation. Each weighting function is designed to emphasize the more dynamic regions of material behavior, where the Miller model parameters influence the

material behavior most prominently. The simplest of the weighting functions is used for low cycle fatigue tests. All data points above a certain stress magnitude in tension and compression are weighted with a value of 2.5, while all points below this value are weighted as 1.0. This range of values was determined from practical experience and its utility was verified in the experimental results. The effect of the weighting scheme is an emphasis on the inelastic regions of material response where the Miller model, and not simple elastic theory, predicts the behavior. The weighting function developed for the stress relaxation is more complicated, using both the stress rate and the derivative of the stress rate:

$$w(t_i) = \frac{C_0 + C_1 \left|\dot{\sigma}(t_i)\right| + C_2 \left|\ddot{\sigma}(t_i)\right|}{\left|C_0 - C_1 \left|\dot{\sigma}(t_i)\right| - C_2 \left|\ddot{\sigma}(t_i)\right|\right|}, \tag{9}$$

where the dot and double dot refer to differentiation with respect to time. The C_1 and C_2 parameters are constants which eliminate nonequivalent units, and are equal to 1 sec/ksi and 1 sec^2/ksi, respectively. The effect of this weighting function is shown in Fig. 12.

The weighting function in Eq. 9 is designed to keep all values of $w(t_i)$ between 1.0 and 2.5. The lower value of 1.0 is set automatically as the stress relaxation approaches a steady state value. When this occurs, both $\dot{\sigma}(t_i)$ and $\ddot{\sigma}(t_i)$ approach zero, leaving only the C_0 term. The C_0 term is a constant which is calculated to set the upper limit of the weighting function at 2.5, and is defined as

$$C_0 = \frac{7}{3}\left(C_1 \left|\dot{\sigma}_{\max}\right| + C_2 \left|\ddot{\sigma}(t_{i,\max})\right|\right), \tag{10}$$

where $\dot{\sigma}_{\max}$ is the maximum rate of stress relaxation, and $\ddot{\sigma}(t_{i,\max})$ is the stress rate derivative at the time index where the maximum stress relaxation rate occurs. Equation 10 ensures that the maximum weighting value of 2.5 occurs at the location where the maximum rate of stress relaxation occurs. This is desirable in order to determine the hardening parameters present in the Miller model more accurately. The effects of the hardening parameters are more present in the regions of material behavior where relaxation is occurring most rapidly. Since experimental data do not represent a continuous function, and cannot be differentiated, a finite difference approximation was used to calculate the stress rate and derivative of the stress rate, using the form of

$$\dot{\sigma}(t_i) = \frac{\sigma(t_{i+1}) - \sigma(t_i)}{t_{i+1} - t_i} \tag{11}$$

$$\ddot{\sigma}(t_i) = \frac{\sigma(t_{i+1}) - 2\sigma(t_i) + \sigma(t_{i-1})}{(t_{i+1} - t_i)(t_i - t_{i-1})}. \tag{12}$$

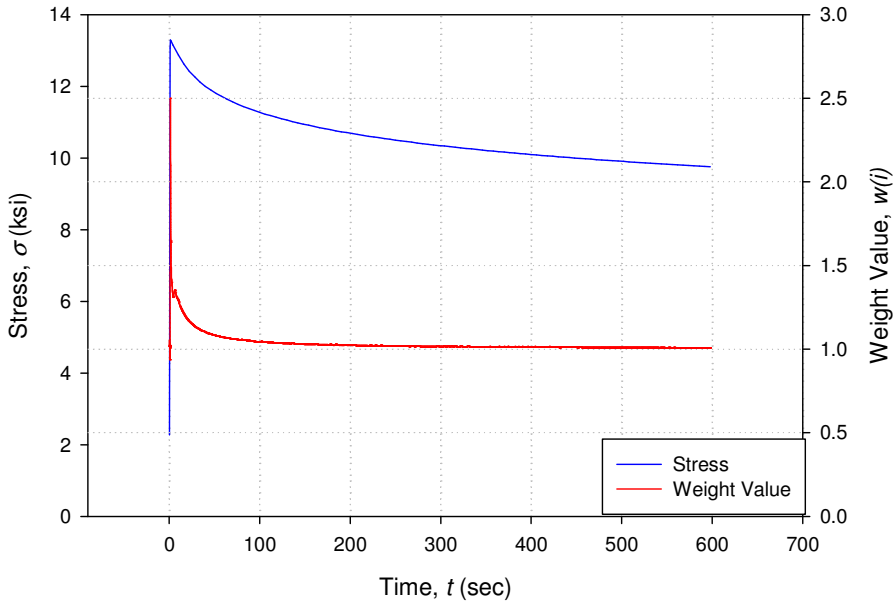


Figure 12: Stress relaxation, during strain control, weighting values determined by uSHARP

The uSHARP routine combines all of these features into an automated optimization procedure, requiring an initial guess for the material parameters and ANSYS input files matching the experimental tests being used in the optimization process. From there, simulations are run and the initial guess is modified until a best fit has been achieved, as presented in Fig. 13. Updating the guess for the material parameters at each iteration is handled by an optimization algorithm incorporated in the uSHARP routine.

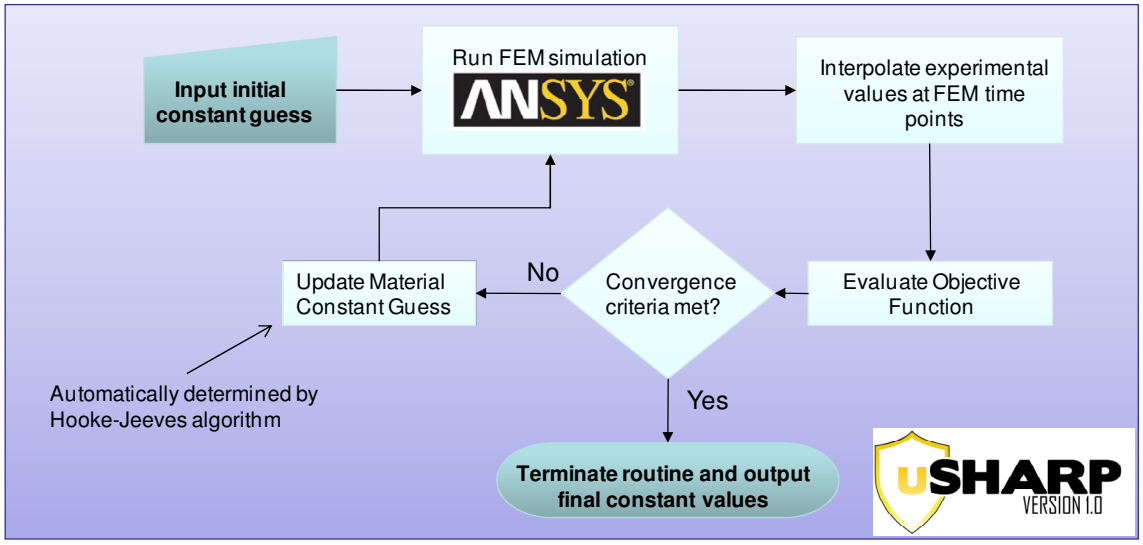


Figure 13: uSHARP optimization routine process flowchart

3.3.2 Optimization Algorithm

In order to minimize the objective function used in the uSHARP routine, an optimization algorithm was needed. Several criteria were used in the selection of this algorithm. First, it was desired that no derivatives or gradients be required for minimization. The complex nature of the Miller model would make such calculations difficult and would limit the robustness of the uSHARP routine by requiring modification

for use with any other constitutive model. To this effect, an optimization algorithm requiring only objective function evaluations was strongly desired. Second, due to the length of time required to carry out a single finite element simulation, an efficient optimization algorithm requiring a minimum number of iterations was desired. While this would almost certainly constrain the uSHARP routine to the use of a local optimizer, the length of time required for an optimization run would be practical and not too costly in terms of computation time. Lastly, an optimization algorithm requiring only a single initial guess was considered important to the goal of minimizing the amount of user input required to perform an optimization run. Often, a set of material constants for a particular constitutive model can be found in literature. These published constants would serve as a good initial guess for an optimization run. All of these desired characteristics are met by the Hooke-Jeeves direct search algorithm (Hooke and Jeeves, 1961), which is a robust local optimization algorithm. The logic process of the Hooke-Jeeves algorithm can be seen in Fig. 14. Its utility has been demonstrated on objective function minimization used for curve fitting with good results (Hooke and Jeeves, 1961). This is similar to the procedure used by the developed method for parameter determination, meaning the Hooke-Jeeves algorithm is well suited for incorporation into the uSHARP routine (see Appendix C).

It should be noted that any non-linear optimization algorithm requiring only objective function evaluations could be used with uSHARP. In fact, the structure of the uSHARP routine makes it very easy to implement other optimization algorithms into the program. Early on in the current study, the use of a simulated annealing algorithm

(Corana et al., 1987) was briefly investigated with promising results, but full validation will be saved for future study.

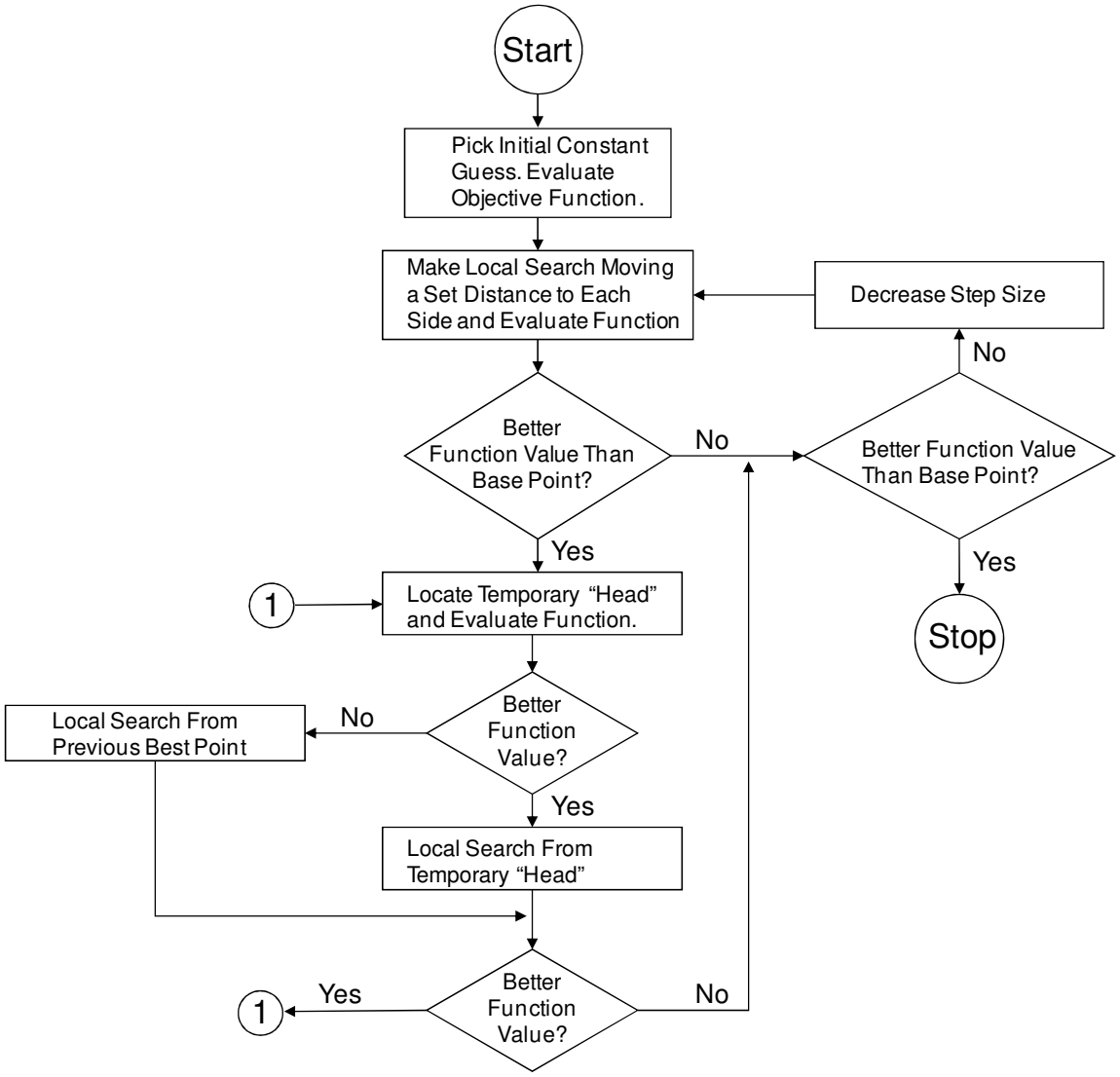


Figure 14: Hooke-Jeeves algorithm process diagram (Kuester and Mize, 1973)

4. EXPERIMENTAL PROCEDURE

4.1 Advanced Candidate Experiments

A major focus of the research was determining a small number of candidate experiments that would allow for the determination of the material parameters with a minimum amount of experimental data. These experiments take the form of numerically simulated mechanical test data generated from the UPF with the published Miller constants for type 304 stainless steel, shown in Table 4 (Miller, 1976b). These simulated experiments are useful in designing the actual mechanical experiments that will be performed with future research. After an analysis of the material parameters was performed based on Miller (1976a), the necessary test conditions were determined for the parameter determination.

The experimental behavior affected by each of the constants is shown in Table 5. The absence of the Q parameter must be noted. The determination of this parameter requires data at different temperatures. To maintain simplicity for the preliminary stages of the research, isothermal test conditions were used, eliminating the ability to determine Q. Future research will include non-isothermal test histories to enable the identification of this parameter.

Table 4: Values of Miller constants for type 304 stainless steel

Inelastic Constant	A_1 (ksi^{-1})	A_2 (ksi^{-3})	B (sec^{-1})	C_2 (ksi)	H_1 (ksi)	H_2	n
"Experimental" Value from Miller (1976b) for type 304 stainless steel	0.8	7.42×10^{-5}	1.0×10^{15}	1.0×10^{-1}	2.8×10^{2}	1.0×10^{2}	5.8

Two candidate experiments were identified for use with the aforementioned uSHARP optimization routine. In order to run the uSHARP routine, "experimental" tests were simulated using the ANSYS finite element program. The material constants from Miller (1976b) for type 304 stainless steel were used for these simulations. The value for Q was set at 91,000 cal/mole, as defined by Miller. In this way, the material constants of the "experimental" test data were already known, allowing for verification of the optimization routine once uSHARP had determined values for material parameters. Once the key simulated experiments have been identified and the parameter optimization process has been fully developed, experimental specimens will be used to obtain actual data. The uSHARP optimization routine can then be verified under real conditions.

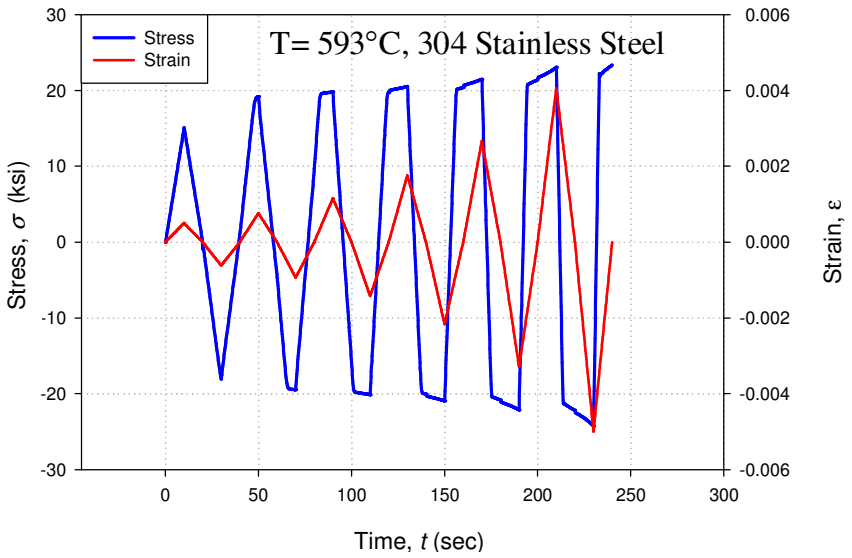
Table 5: Effect of constants on material behavior

Material Constants	Strongly Influences
A_1 (ksi^{-1})	Cyclic behavior at high strain ranges
A_2 (ksi^{-3})	Cyclic behavior at high strain ranges
B (sec^{-1})	Secondary creep
C_2 (ksi)	Cyclic behavior at low strain ranges
H_1 (ksi)	Cyclic behavior at intermediate strain ranges
H_2 $(unitless)$	Cyclic and isotropic hardening, primary creep
n $(unitless)$	Secondary creep

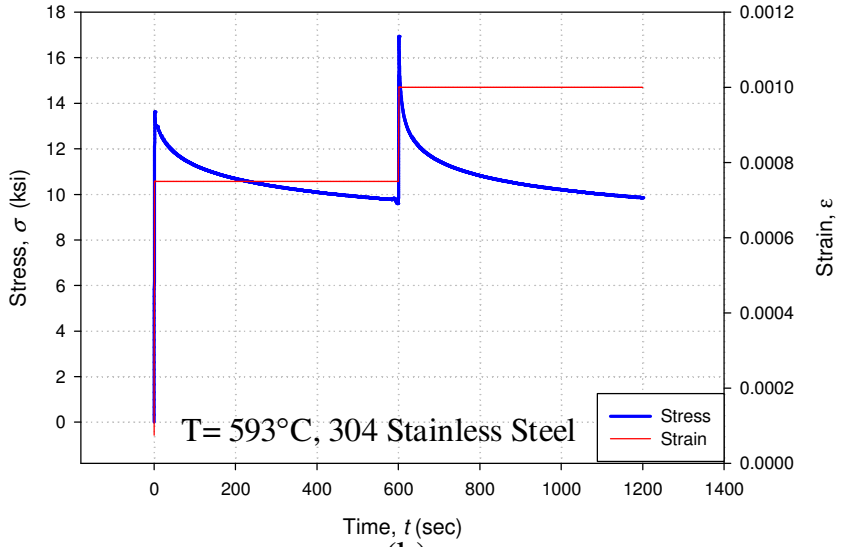
One simulated "experimental" test that was used was a strain-controlled low cycle fatigue test with ramping strain amplitude, as shown in Fig. 15a. The strain amplitude for the first cycle was 0.05%, and the amplitude of the last cycle was 0.5%. The time per quarter cycle was held at a constant 10 seconds, and the simulated temperature was

593°C. The strain rates experienced during the test ranged from $5.0x10^{-3}$ % sec^{-1} to $5.0x10^{-2}$ % sec^{-1}. This was done in order to provide material behavior at different strain rates, an important consideration with a strain rate sensitive model such as the Miller model (Miller, 1976a). Additionally, different strain amplitudes in a single test provide a broad spectrum of material behavior using only one test specimen. This low cycle fatigue test was used to simulate the non-hardening and cyclic material behavior so the corresponding constants could be determined. Six cycles were simulated in order to allow for reasonable simulation times.

The second "experimental" test type that was used to determine the material parameters was a strain-controlled stress relaxation test, also at 593°C. Two different strain amplitudes were applied in one test, as depicted in Fig. 15b. The first strain amplitude was 0.075%, followed by a ramping up to a level of 0.1%. Each of the amplitudes was held for 600 seconds. One second was used as the strain ramping time. This was done in order to determine the material hardening and creep properties and allow for the determination of the corresponding material constants. The two different strain amplitudes were used in order to gather material information under different conditions using only one test.



(a)



(b)

Figure 15: "Experimental" (a) low-cycle fatigue test at 593°C and (b) stress relaxation test at 593°C used for parameter determination

In case the previously described “experimental” tests were not sufficient to accurately determine the material parameters, a third test was created to be used along with the low cycle fatigue and stress relaxation tests. A strain-controlled ratcheting test was simulated at 593°C for fifteen cycles. The strain for the test was increased 0.05% under loading and decreased 0.025% during unloading, as depicted in Fig. 16. This

process was repeated fifteen times for a maximum amplitude of 0.4%. The time between successive peaks was held at a constant 40 seconds so that the strain rates varied. A high number of cycles were simulated in the ratcheting test to account for any shortcomings resulting from the low number of cycles present in the "experimental" low cycle fatigue test. This resulted in a significantly longer computation time; the compromise would come in the form of more accurate constants. The ratcheting test would only be employed for parameter determination if the stress relaxation and low cycle fatigue tests proved insufficient.

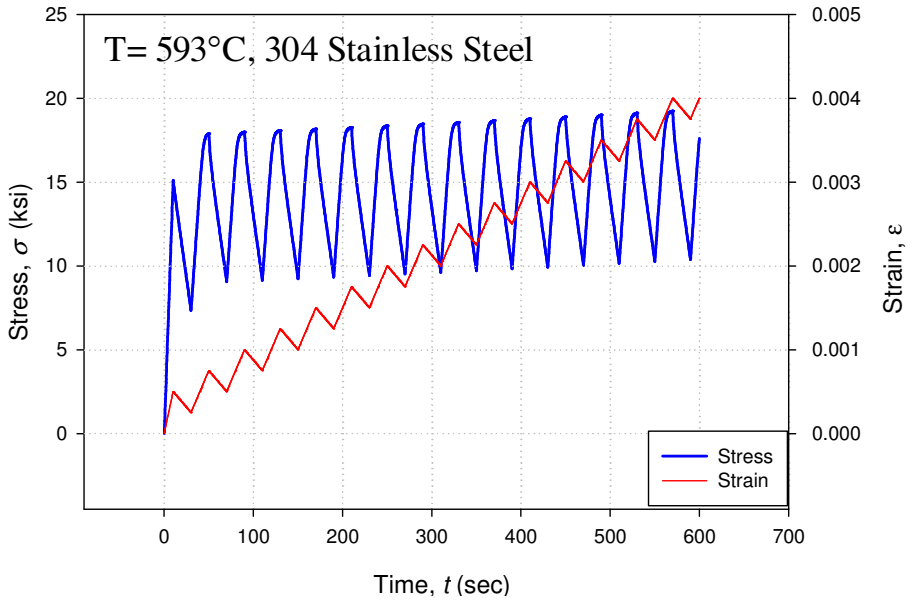


Figure 16: Ratcheting test used for parameter determination

Because the evolution of the internal state variables, R and D, is responsible for characterizing the material behavior, it is important to investigate how the ISVs evolve during the three simulated experiments. As a basis for comparison, evolution of the ISVs for standard test types are presented along with those of the designed experiments in Figs. 17 and 18. The stress relaxation test was compared with a creep test simulated at the

same temperature, under a stress of 15 ksi. The ratcheting and ramping amplitude low-cycle fatigue tests were both compared with simulated constant peak amplitude strain-controlled cyclic fatigue tests. For comparison with the ratcheting test, the peak strain amplitude of the standard cyclic test was set at 0.4%, the maximum amplitude experienced by the designed experiment. Similarly, the peak strain amplitude for the standard cyclic test corresponding to the ramping amplitude low-cycle fatigue test was set at 0.5%, the maximum amplitude experienced by the designed candidate experiment.

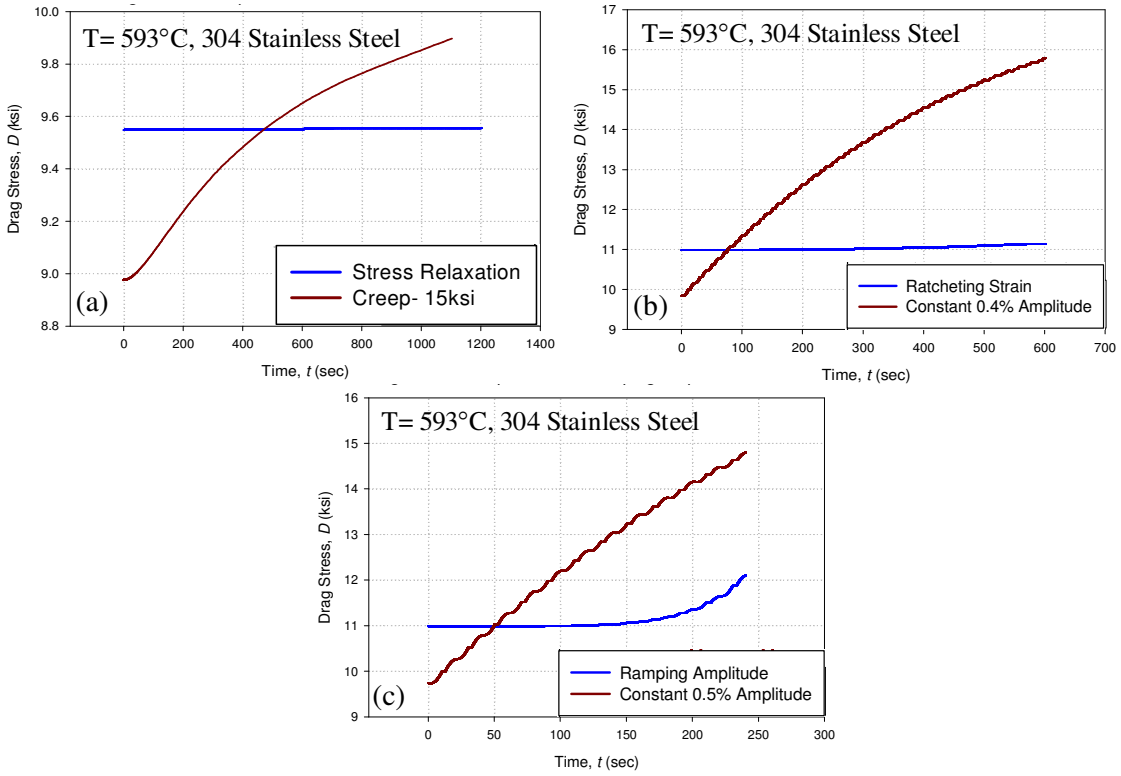


Figure 17: Drag stress comparison with standard experiments for (a) stress relaxation test, (b) ratcheting test, and (c) ramping amplitude low-cycle fatigue test

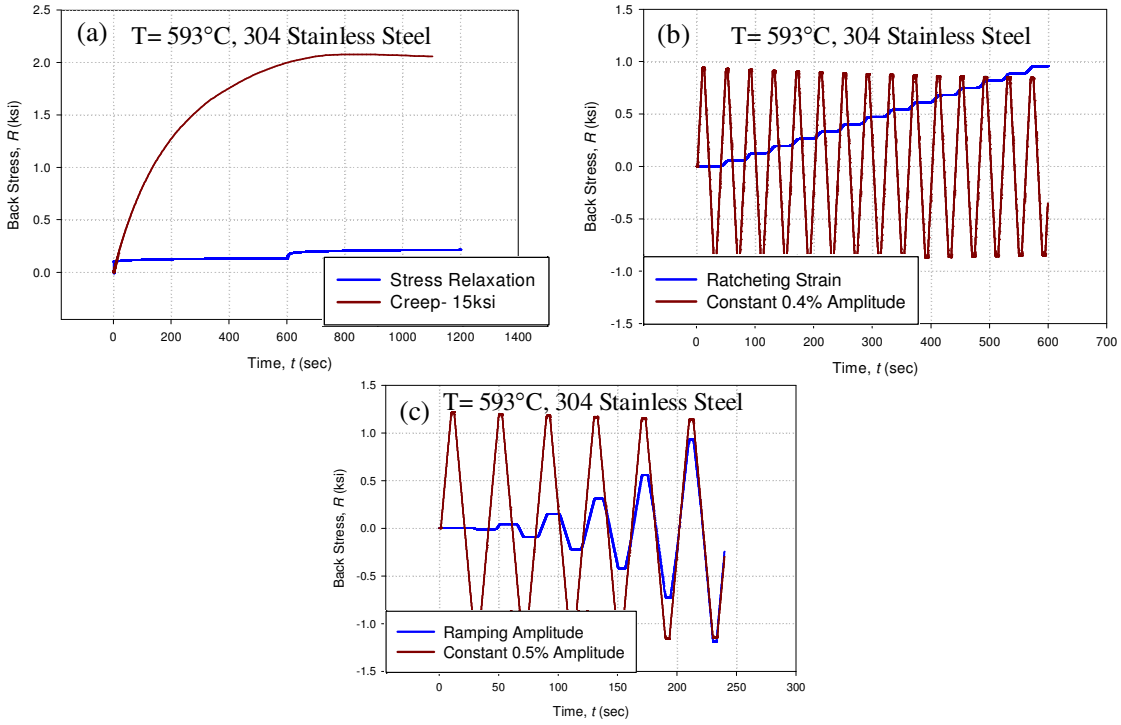


Figure 18: Back stress comparison with standard experiments for (a) stress relaxation test, (b) ratcheting test, and (c) ramping amplitude low-cycle fatigue test

None of the designed experiments demonstrate significant evolution of the drag stress compared to the standard test types. This means that very little isotropic hardening is occurring during the designed experiments. This is likely due to the low amount of inelastic deformation occurring during the designed experiments. The back stress evolution is much more dynamic for the proposed ratcheting and ramping amplitude low-cycle fatigue tests than the standard experiments. While the back stress evolution demonstrates cyclic behavior in the standard test, it is essentially steady state; there is very little change in the response from cycle to cycle. In contrast, the peak amplitude of the back stress evolution during the ramping amplitude low-cycle fatigue test increases with each successive cycle. The back stress evolution during the ratcheting test is not

cyclic in nature; instead, it steadily rises throughout the duration of the test. There is very minimal back stress evolution during the stress relaxation test.

All of the simulated "experimental" tests were designed so that they could be realistically conducted by the available hardware, thus allowing for actual experimental validation following the preliminary trial runs using the aforementioned synthetic data. None of the "experimental" tests feature instantaneous changes in strain level; there are no strain rates which are not achievable mechanically. While it was assumed that more complex control systems will enable more complicated experimental tests to be accurately conducted, the experimental tests that will be conducted for validation will be controlled by a PID controller. Correspondingly, none of the tests designed here should contribute too difficult of a challenge for the test frame; they are slightly more complex versions of standard experiments which are routinely conducted and are all isothermal. Furthermore, to ensure that the designed experiments could be conducted on specimens without premature failure, all of the strain rates and maximum strain amplitudes were kept to levels at or below those of the data found in Miller (1976b). There is no reason to believe that the candidate experiments designed here will accrue damage to a degree that would invalidate the data for use with the Miller model. Certainly, if the experiments were allowed to continue indefinitely, the specimens would rupture eventually; however, the needed quantity of experimental data will be acquired prior to specimen failure.

4.2 Parameter Determination using uSHARP

The "experimental" data that were used to validate the uSHARP optimization routine were created by finite element simulation using the constants for type 304

stainless steel determined by Miller (1976b) and applied to the previously outlined strain controlled experiments. As noted, an initial guess for each of the material constants is required for parameter determination. In order to validate the functionality of the uSHARP routine, a simulation was initiated using a guess for the material parameters that was biased and close to the "experimental" constants. Following successful validation, the initial guess was modified to a standard initial guess, seen in Table 6, and the optimization was executed again. The effects of the two initial guesses on the simulation of the ramping amplitude low-cycle fatigue test can be seen in Fig. 19. It is clear that the close initial guess provides a fairly close recreation of the "experimental" data. In contrast, the standard initial guess provides a stress amplitude that is nearly double that of the "experimental" data. As mentioned previously, the uSHARP routine performs best when the appropriate orders of magnitude for the constants are already known. This is evident in the selection of the standard guess. It is assumed that for any future parameter determination runs using a different alloy the constants developed for stainless steel will be used as a starting guess. This assumption is provided as justification for choosing an initial guess which is on the proper order of magnitude. During every optimization run, the evolution of the constants and the objective function value were recorded after each iteration. This enabled for real time updates of these values as the parameter determination progressed, and created a comprehensive log of these values for analysis following an optimization run.

Table 6: Material constants used to simulate "experimental" data and those used as an initial guess.

Parameter Set	Viscoplastic Constants						
	A_1 (ksi^{-1})	A_2 (ksi^{-3})	B (sec^{-1})	C_2 (ksi)	H_1 (ksi)	H_2	n
"Experimental" Value from Miller for type 304 SS	0.8	7.42×10^{-5}	1.0×10^{15}	1.0×10^{-1}	2.8×10^{2}	1.0×10^{2}	5.8
Close Guess	1.0	5.0×10^{-5}	1.0×10^{15}	1.0×10^{-1}	2.0×10^{2}	1.0×10^{2}	4.0
Standard Guess	1.0	1.0×10^{-5}	1.0×10^{15}	1.0×10^{-1}	1.0×10^{2}	1.0×10^{2}	1.0

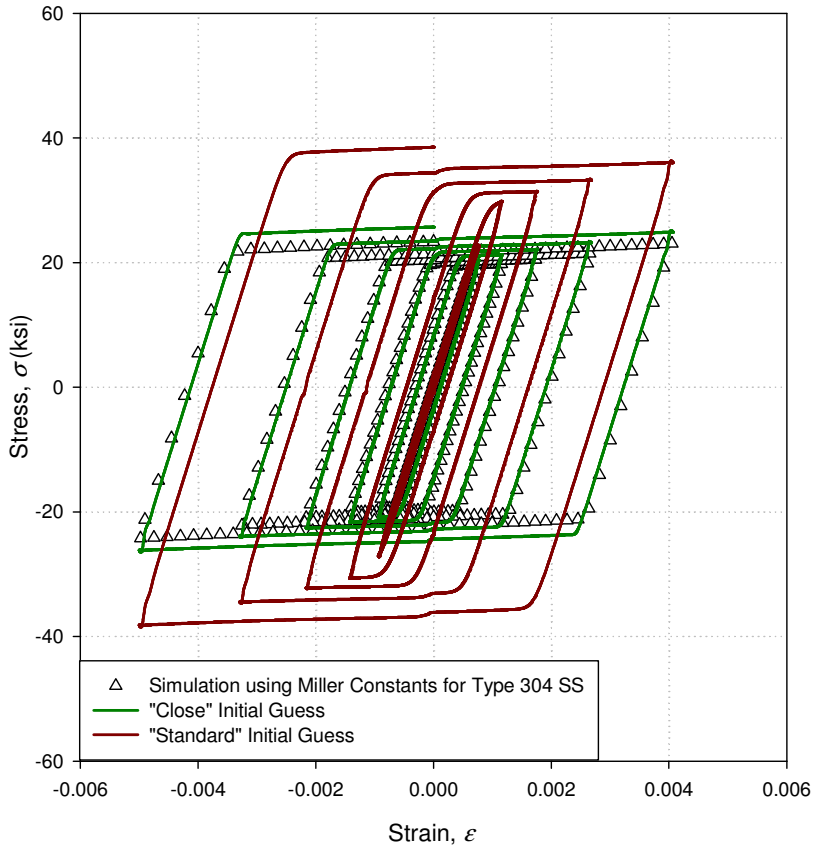


Figure 19: Effect of initial guesses on behavior during ramping amplitude low-cycle fatigue test

Using the weighting and objective functions described above, the compound stress relaxation test (Fig. 15b) and low cycle fatigue test with ramping amplitude (Fig. 15a) were used simultaneously to determine the material parameters using the uSHARP routine. The close guess was first used as a starting point, followed by the standard

guess. For both runs, the Hooke-Jeeves algorithm was used as the optimization algorithm. Once material parameters were determined by uSHARP, they were compared to the "experimental" set to check for accuracy. These results are presented in the following section. Once the constants had been determined using only the stress relaxation and ramping low cycle fatigue test, the ratcheting test was included and all three tests were used simultaneously to determine the material parameters using the Hooke-Jeeves algorithm. For the ratcheting test, the weighting function developed for low cycle fatigue tests was applied. In each case, the time versus stress data were used in the fit, with the stress values being used for y_{sim} and y_{ex} in Eq. 7.

During each optimization routine, the same input files that were used to generate the "experimental" data were used to generate data for each iteration and evaluate the objective function. The only difference between the input files was the material constants used. This was done under the assumption that with more advanced control systems, experimental and simulated test histories will closely match. There will, of course, be variances under real conditions but at this stage of the research, the goal is to develop a functional optimization routine. The uSHARP routine must first be validated under ideal conditions before the expenses of actual experimental testing should be incurred.

5. RESULTS AND DISCUSSION

5.1 Material Parameter Determination

The uSHARP routine was used three separate times to determine the Miller model material parameters. An outline of the initial guess used and the "experimental" tests used to determine the material constants is shown in Table 7. Two parameter determinations were done with the simultaneous use of the stress relaxation test and ramping amplitude low-cycle fatigue test to evaluate and minimize the objective function. The first determination was done using the close guess. The second was done using the standard guess. Once results had been obtained for both cases, the ratcheting test was added, and using the standard initial guess, the parameter determination was performed again. In each case, the "experimental" data was generated using the constants determined by Miller for type 304 stainless steel, as shown in Table 8 (Miller, 1976b).

Table 7: Initial guess and "experimental" tests used for parameter determination

Parameter Determination	Initial Guess	Stress Relaxation Test	Ramping Amplitude Low-Cycle Fatigue	Ratcheting Test
1	Close guess	X	X	
2	Standard guess	X	X	
3	Standard guess	X	X	X

The first parameter determination run was conducted using the close, biased guess and two of the "experimental" tests: the ramping amplitude low-cycle fatigue and stress relaxation tests. The close, biased guess was used to simulate a situation where the constants of a similar material are known and can serve as a starting point, or a premature

convergence of the optimization algorithm occurred. It was also done to test the functionality of the uSHARP routine under nearly ideal conditions. The values of the constants used for the initial guess and "experimental" test simulations are presented in Table 8, along with the final converged constant values. The percent differences between the constants of the "experimental" and solution sets are also presented as a way to verify the accuracy of the final converged constants. It is apparent that the "experimental" constants were not accurately determined during the optimization run.

Table 8: Values for material constants during first parameter determination

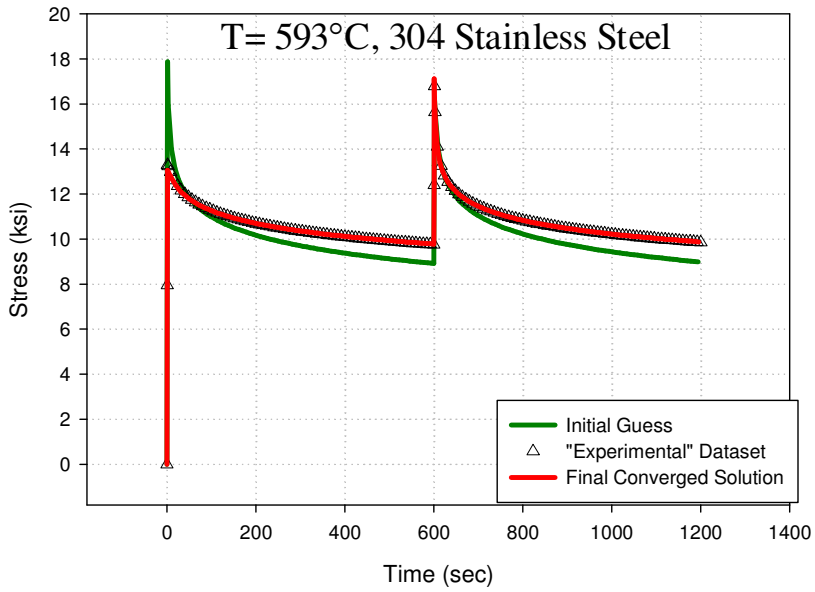
	A_1 (ksi^{-1})	A_2 (ksi^{-3})	B (sec^{-1})	C_2 (ksi)	H_1 (ksi)	H_2	n
"Experimental" Value from Miller for 304 SS	0.8	7.42×10^{-5}	1.0×10^{15}	1.0×10^{-1}	2.8×10^{2}	1.0×10^{2}	5.8
Initial Guess	1.0	5.0×10^{-5}	1.0×10^{15}	1.0×10^{-1}	2.0×10^{2}	1.0×10^{2}	4.0
Converged Values	1.53	3.28×10^{-5}	1.31×10^{15}	1.69×10^{-1}	2.28×10^{2}	1.03×10^{2}	6.0
Percent Difference	91.3	55.8	31.0	69.0	18.6	3.0	3.4

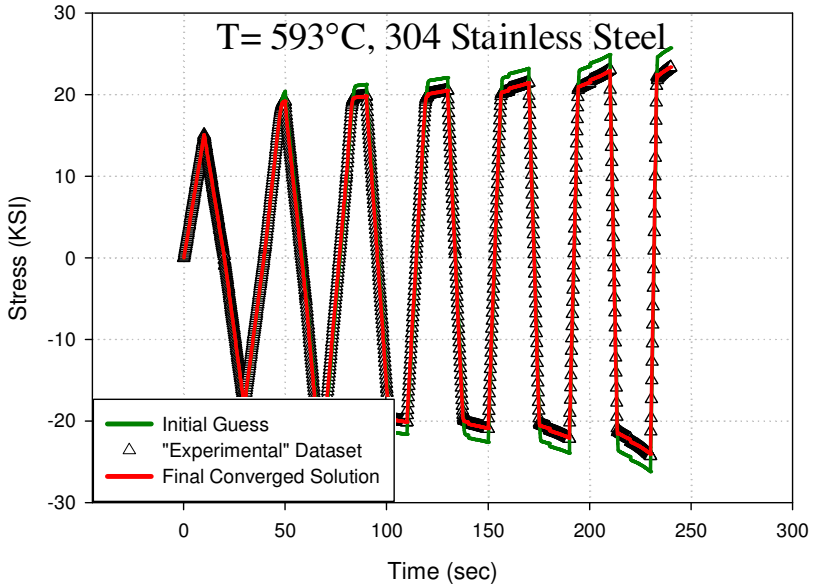
While the converged constants and the "experimental" constants did not match, the graphical fits between the simulations of the two datasets were very good, as shown in Fig. 20. The stress history of the final converged solution matches that of the experimental data very closely for both the stress relaxation test and the low cycle fatigue test. The stress histories for the initial guess are also quite close to those of the "experimental" tests, indicating a severe bias in the initial guess.

The evolution of the objective function throughout the optimization using the close initial guess is shown in Fig. 21. Starting with an initial value of 93.97, the objective function achieved a minimum value of 0.1614 where the best fit occurred. It is

apparent that the optimization process progressed very close to the minimum after approximately 100 iterations. Approximately 350 iterations are then spent with little change in the objective function value. This is most likely due to the initial guess being close to the minimum value. Because the initial guess was biased and close to the actual values of the "experimental" set, it did not take long for the uSHARP routine to find the close minimum. Very little alteration of the constants was needed to create a very good fit to the "experimental" data.

After the results of the close, biased guess indicated proper functionality of the uSHARP routine, a parameter determination was carried out again, this time with the standard initial guess. Similar to the initial run, several of the final converged constants were not close in value to those of the "experimental" set, as shown in Table 9. The uSHARP routine did determine four of the constants to within 4% of their corresponding "experimental" values: B, H_1, C_2, and n. A comparison with the results from the first optimization run shows a similar trend, with three of the four constants displaying relatively good matches with the Miller constants.





(b)

Figure 20: uSHARP convergence results using close initial guess for (a) stress relaxation test and (b) ramping amplitude low-cycle fatigue test

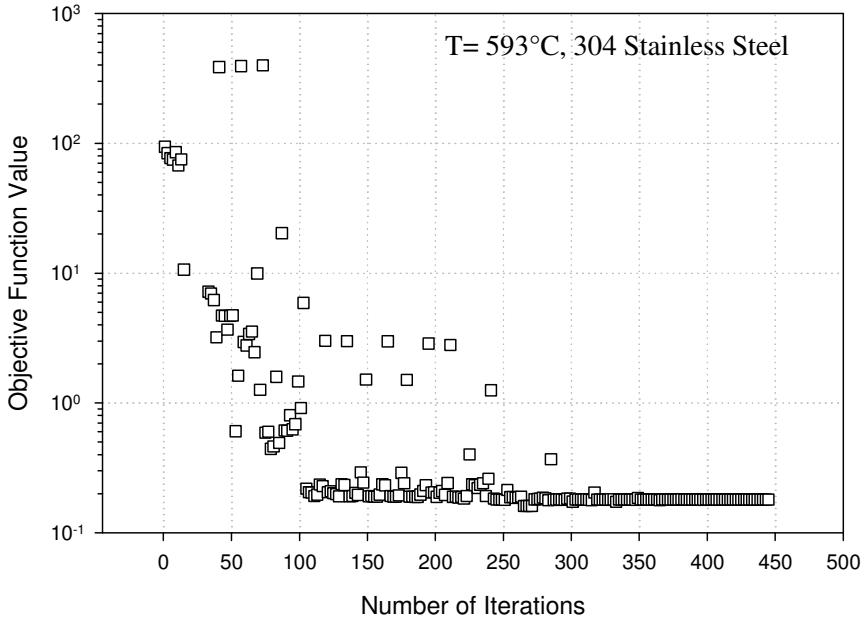


Figure 21: Objective function convergence for close, biased initial guess

Table 9: Values for material constants during second parameter determination

	A_1 (ksi^{-1})	A_2 (ksi^{-3})	B (sec^{-1})	C_2 (ksi)	H_1 (ksi)	H_2	n
"Experimental" Value from Miller for 304SS	0.8	7.42×10^{-5}	1.0×10^{15}	1.0×10^{-1}	2.8×10^{2}	1.0×10^{2}	5.8
Initial Guess	1.0	1.0×10^{-5}	1.0×10^{15}	1.0×10^{-1}	1.0×10^{2}	1.0×10^{2}	1.0
Converged Values	2.42	3.69×10^{-6}	9.79×10^{14}	9.72×10^{-2}	2.90×10^{2}	2.44×10^{1}	5.81
Percent Difference	202.5	95.0	2.1	2.8	3.57	75.6	0.17

The standard initial guess showed more significant differences with the "experimental" data than the close, biased initial guess. These starting values are much more indicative of using a published set of constants as a starting guess. In this case, the initial guess is a poor prediction of the "experimental" test history. Using the standard initial guess, the amplitude of the ramping low-cycle fatigue test was nearly double that of the "experimental" data, as shown in Fig. 22. Additionally, the amount of stress relaxation occurring in the initial guess is nearly five times that of the "experimental"

data. Again, the final fit of the converged values was very good, with nearly identical stress histories between the final converged values and the "experimental" data.

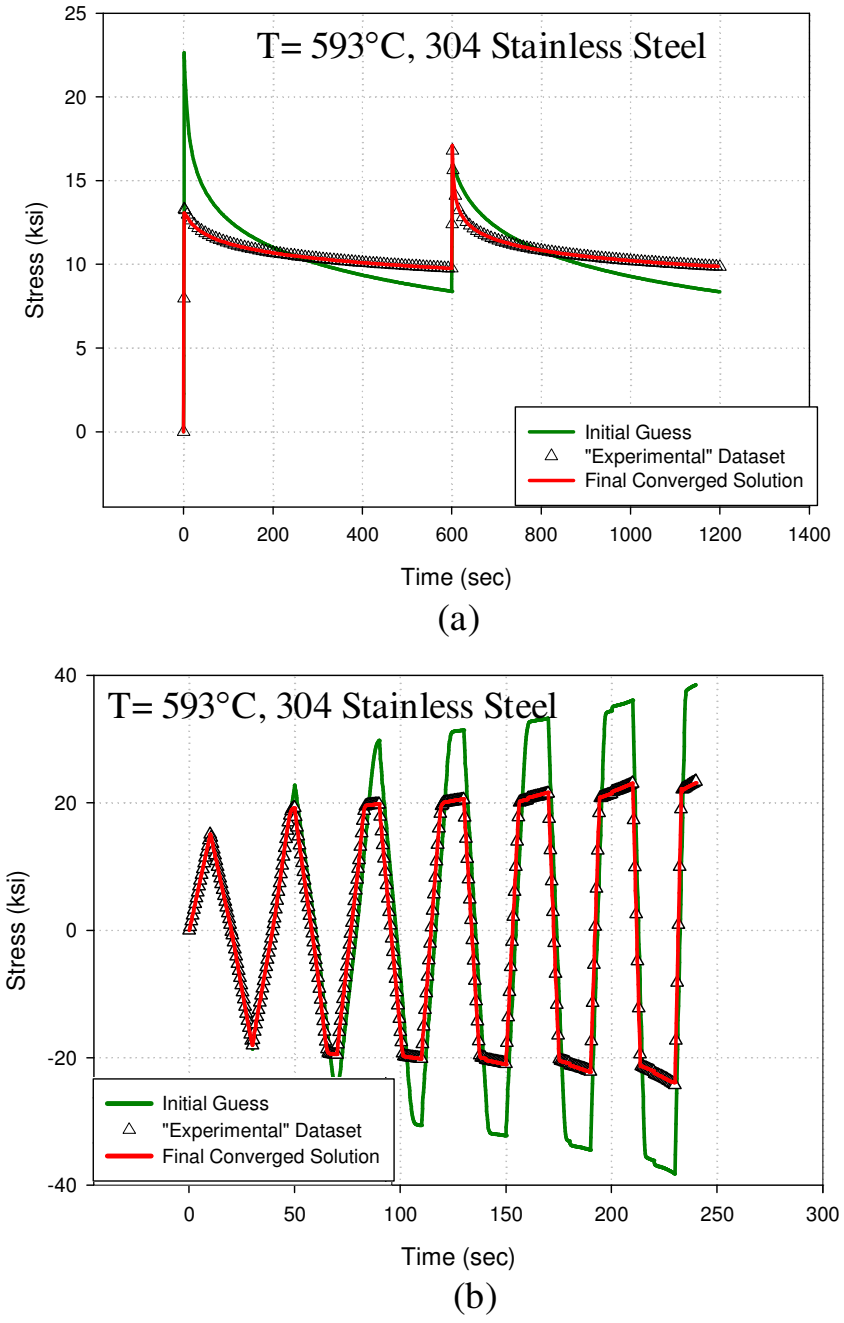


Figure 22: uSHARP convergence results using standard initial guess for (a) stress relaxation test and (b) ramping amplitude low-cycle fatigue test

Finding a solution using the standard initial guess required more than twice the number of iterations than the close, biased guess, as shown in Fig. 23. This is due to the need to search a larger portion of the solution space than was previously required to find a minimum. With a minimum objective function value of 0.2963, the solution fit for the standard initial guess was not quite as good as that of the close guess, which had a minimum objective function value of 0.1614. Even though the initial guess provided an objective function value of 3063.0, a much worse starting point than with the close initial guess, the uSHARP routine was still successful in determining a very good fit; the final objective function value was on the same order of magnitude as that of the close, biased guess. Relative to the scale of the problem, the difference between the two values is trivial.

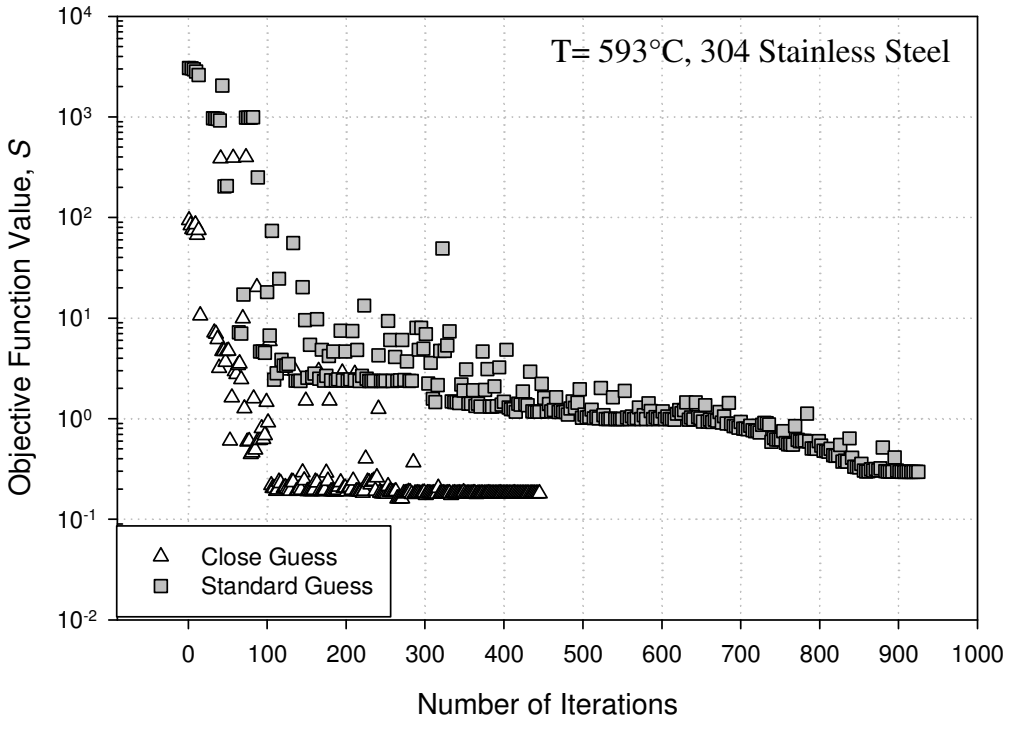


Figure 23: Comparison of objective function convergence between close and standard initial guesses.

The results of these two runs indicate good performance of the uSHARP routine. While the constants were not determined accurately in all cases, this is most likely due to the candidate experiments than the uSHARP routine itself. In order for the constants to be determined accurately, their effects on material behavior must be completely captured by the test matrix used during the optimization process. An analysis of the ill-determined constants yields insight into how the test matrix might be improved. In both cases, a poor solution for A_1 and A_2 was found. These two constants strongly influence the material behavior at high strain amplitudes. In order to determine their values, Miller (1976b) used results from more than 25 cyclic tests with constant strain amplitude. The highest strain amplitudes in the data were approximately 3%. Additionally, the tests were run until cyclic saturation occurred in all of the cases, meaning the peak stress values from cycle to cycle were constant. The maximum strain amplitude reached in the simulated ramping low-cycle fatigue test was 0.5%, much lower than that used by Miller (1976b). In the simulated "experimental" test, cyclic saturation had not occurred. Six cycles is not enough for saturation to occur in type 304 stainless steel; approximately 30 cycles are needed (Miller, 1976b). Additionally, with the ramping strain amplitude, cyclic saturation would be difficult to determine. The H_2 constant strongly influences isotropic hardening and drag stress evolution. As shown in Fig. 17, minimal drag stress evolution was present in the designed "experimental" tests. This indicates that very little isotropic hardening occurred, meaning the test matrix was insufficient to determine H_2.

For the H_1, B, n, and C_2 constants, the two "experimental" tests were adequate in capturing their effects on material behavior. Stress relaxation and creep behavior are strongly influenced by the B and n constants. This was the main reason for the inclusion

of a stress relaxation test. The effects of the H_1 constant are seen at intermediate strain ranges. It is apparent that the level of strain reached in the ramping amplitude low-cycle fatigue test was adequate in capturing the effects of this constant. Lastly, the C_2 constant is most influential on lower strain ranges, which seem well represented by the low-cycle fatigue test. Since four of the Miller constants can be determined from these two tests they will remain in the test matrix and will be augmented by other candidate experiments capable of capturing the effect of the other three constants.

With these considerations in mind, the "experimental" ratcheting test was combined with the stress relaxation and ramping amplitude low-cycle fatigue tests, and the uSHARP routine was again used to determine the model constants with these three tests. The standard initial guess was again used as the starting point to enable comparison with the previous parameter determination results. For a third time, the material parameters converged to values which did not match the Miller experimental constants, as shown in Table 10. Again, the B, H_1, and n constants are the closest in value to their "experimental" equivalents. The determination of A_1 improved significantly from the previous run using the standard initial guess, indicating that the inclusion of the ratcheting test helped to capture the effects of the constant better than the use of only the other two tests. The same can be said for the H_2 constant; it is likely that the additional number of cycles in the ratcheting test has helped to better highlight cyclic hardening effects influenced by the H_2 parameter. Unfortunately, these improvements are accompanied by poor performance in the determination of the A_2 and C_2 constants. This will be discussed further momentarily.

The stress histories of the "experimental" and final converged solution were, once again, nearly identical, as shown in Figs. 24 and 25. The initial guess used for the three tests was the same standard guess used in the previous optimization run, as reflected in the similarities between Figs. 22 and 25. The standard initial guess proved to be a poor predictor of the material behavior during the ratcheting test. Despite the low caliber of the initial fit, uSHARP was again successful in providing an excellent fit to the experimental data.

Table 10: Values for material constants during first third determination

	A_1 (ksi^{-1})	A_2 (ksi^{-3})	B (sec^{-1})	C_2 (ksi)	H_1 (ksi)	H_2	n
"Experimental" Value from Miller for 304SS	0.8	7.42×10^{-5}	1.0×10^{15}	1.0×10^{-1}	2.8×10^{2}	1.0×10^{2}	5.8
Initial Guess	1.0	1.0×10^{-5}	1.0×10^{15}	1.0×10^{-1}	1.0×10^{2}	1.0×10^{2}	1.0
Converged Values	0.46	3.01×10^{-4}	8.19×10^{14}	7.25×10^{-1}	3.15×10^{2}	1.31×10^{2}	5.68
Percent Difference	42.5	305.7	18.1	625.0	12.5	31.0	2.07

The number of iterations required for convergence using three tests was nearly five times that of the previous run using the standard initial guess and two tests. This is most likely due to an increased level of complexity introduced by the inclusion of a third test. Previously, the differences between only two datasets needed to be minimized. Adding a third test increased the size of the search space required before an acceptable minimum was found by the Hooke-Jeeves algorithm. Starting with an initial value of 4276.9, the objective function converged to a final value of 0.1352 as shown in Fig. 26. This is of the same order of magnitude as the previous two minimum objective function values, indicating that the caliber of the fits are nearly the same in all cases.

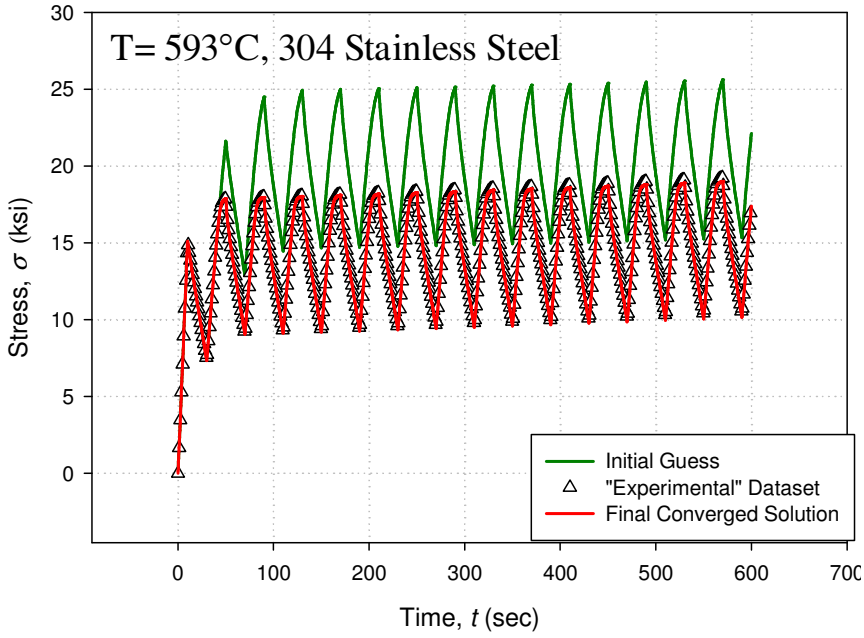
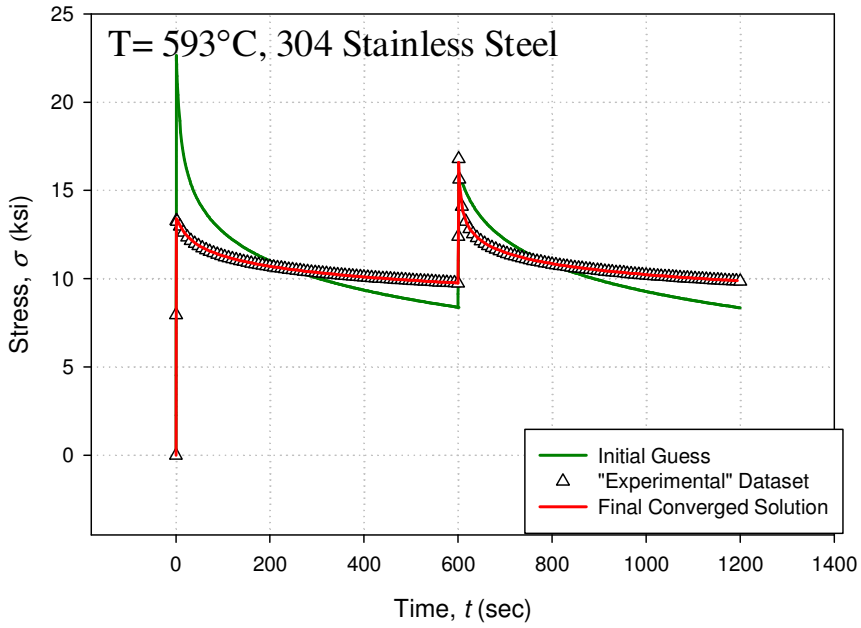


Figure 24: uSHARP convergence results using standard initial guess for ratcheting test used in conjunction with stress relaxation and low-cycle fatigue tests

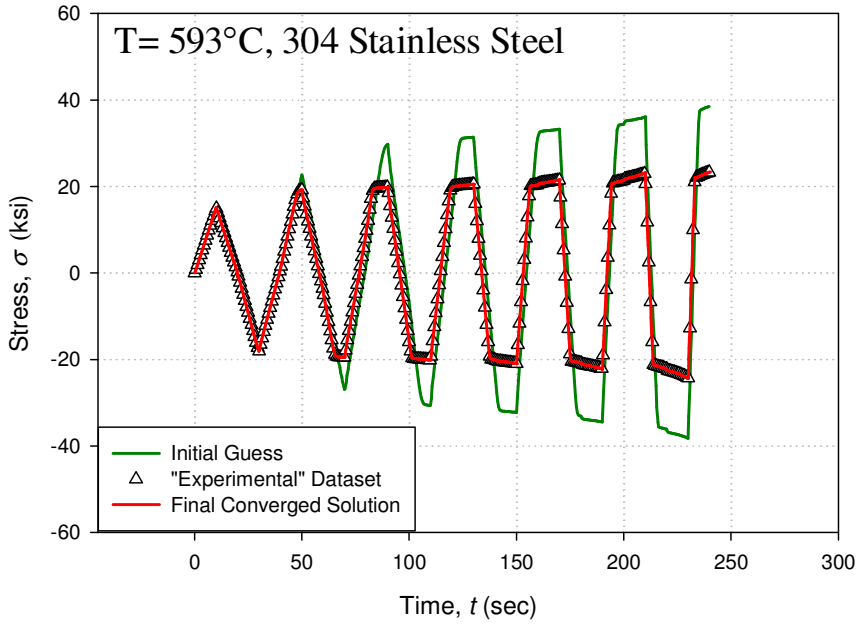
It was discovered after the third optimization run that the "experimental" stress relaxation and ratcheting tests were not created with the C_2 value from Miller (1976b). The values for the other material constants were correct, and all of the values for the constants used in the creation of the "experimental" ramping low-cycle fatigue test were also correct. This explains the discrepancies between the second and third optimization runs. Because the first two optimization runs were conducted with the stress relaxation test and the low-cycle fatigue tests, the effects of the correct value of C_2 were most prominent. The stress relaxation behavior simulated by the Miller model is insensitive to the value of C_2, as shown in Fig. 27. Therefore, the effects of C_2 were present primarily in the low-cycle fatigue "experimental" data, which was created with the correct value. This is the reason that C_2 was determined to within 3% of its actual value during the second optimization run, as shown in Table 9. When the ratcheting test was included, the

incorrect C_2 value was introduced into the experimental data, leading to the large error present between the value determined by Miller (1976b) and the final converged solution. It is not clear whether or not this error was the cause of the discrepancies between the converged values of the other constants and their "experimental" values, but it is likely due to the coupled nature of the equations which make up the Miller model. What is clear is that the parameter determination runs need to be repeated with the proper experimental data. This is currently in progress.

It must be noted that the "experimental" data used here were free from any noise and errors induced by use of actual experimental hardware. Essentially, the synthetic data represents a situation where the material response was perfectly measured experimentally, and the material properties were identical from specimen to specimen. In reality, this is never the case. Small errors introduced by the mechanical test frame and instrumentation contribute a very small amount of randomness into the results in the form of noise, and differences in manufacturing between two specimens will lead to variances in the results between two identical tests conducted on two different specimens. This variance, however, should remain at less than 5% when the developed experiments are conducted on real material specimens (J. Karl, personal communication, April 2009). This implies that useable results can still be obtained, in spite of the inherent discrepancies.



(a)



(b)

Figure 25: uSHARP convergence results using standard initial guess for (a) stress relaxation test and (b) ramping amplitude low-cycle fatigue test used in conjunction with ratcheting test to determine material parameters

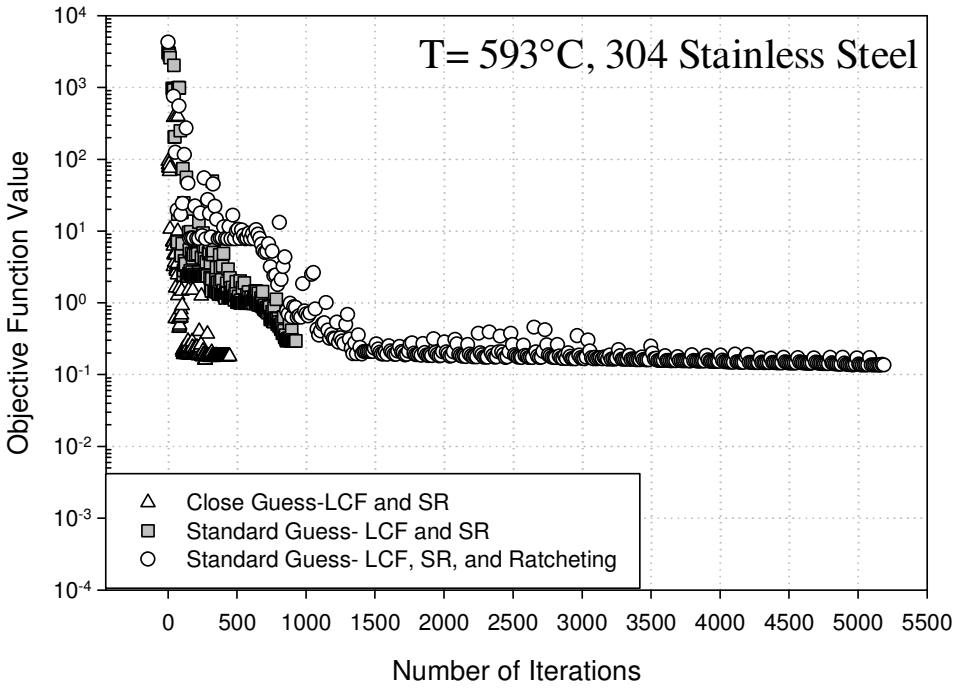


Figure 26: Comparison of objective function convergence for the three parameter determination runs

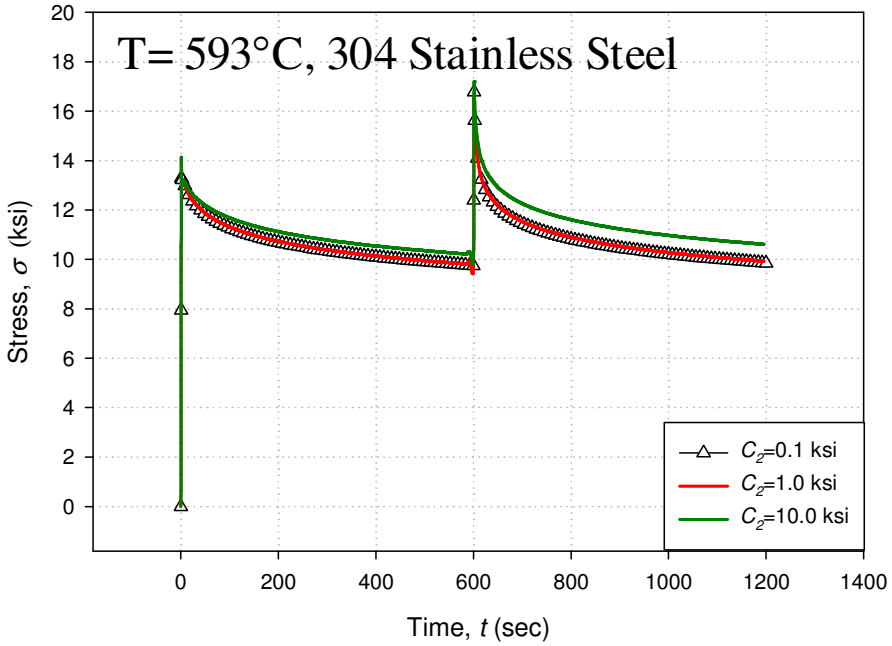


Figure 27: Effect of C_2 on stress relaxation behavior. All parameters were held constant while C_2 was varied.

5.2 Objective Function Analysis

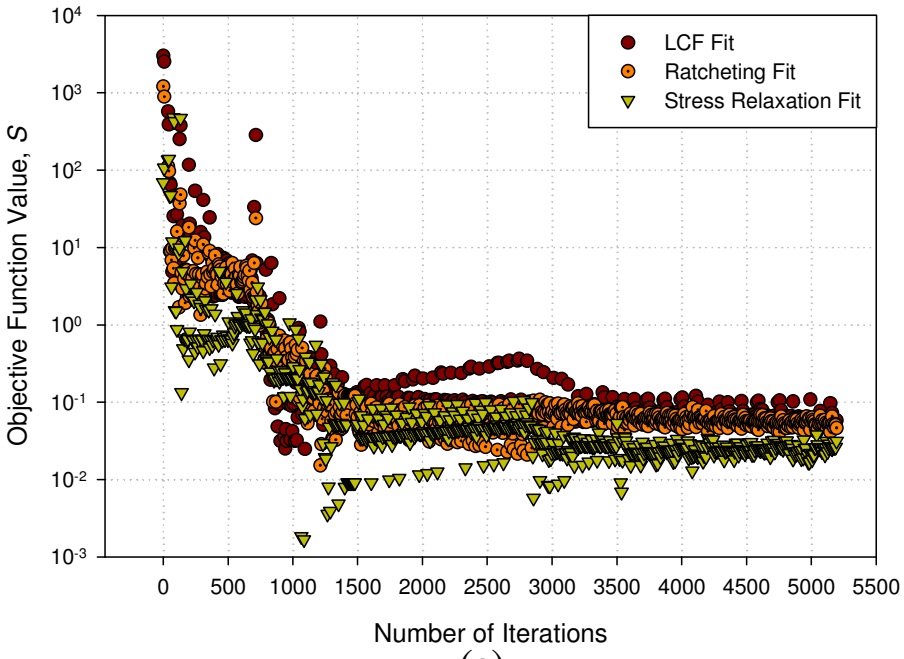
The objective function featured in the uSHARP routine was designed to enable the use of different test types and data sets simultaneously for parameter determination. It is pertinent to perform an analysis as to whether or not this goal has been achieved. The standard objective function commonly used for automated parameter determination is a simple, weighted least squares function (Mustata and Hayhurst, 2005; Mulyadi et al., 2006). This is typically expressed in the form of

$$S = \sum_{i=1}^{n} w(i)\left(y_{ex}(i) - y_{sim}(i)\right)^2, \tag{13}$$

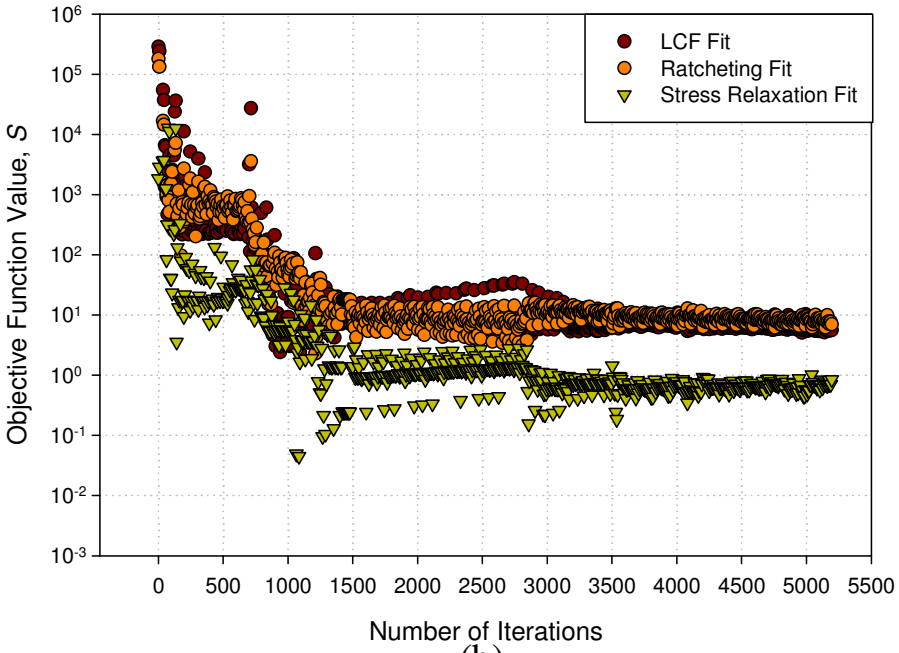
where n is the number of data points, $w(i)$ is the weight factor, y_{ex} is the experimental data point, and y_{sim} is the simulated data point. Multiple datasets are then combined in the form of Eq. 8. To provide an indication of the utility of Eq. 7, the uSHARP objective function, the individual S values of each of the three tests were recorded during the third optimization process. By comparing these results with S values calculated using Eq. 13 during the same optimization run, the proposed objective function can be compared with the standard from current literature. For both equations, the uSHARP weighting functions were used. The evolution of S throughout the optimization process for both cases is shown in Fig. 28. The results show that the use of the least squares error function would have been insufficient for the simultaneous use of the experimental data used during the parameter determination runs. Equal consideration would not have been given to each test. Use of the uSHARP objective function, however, eliminates this problem.

The use of Eq. 13 for the combination of the low-cycle fatigue and ratcheting test would have been sufficient in this case. The least-squares values for both tests are

approximately on the same order of magnitude, meaning that neither of the curves would receive more weight than the other during the optimization process. This trend can also be seen with use of Eq. 7, the uSHARP objective function. However, the inclusion of the stress relaxation test creates a problem when using Eq. 13. Throughout the duration of the optimization process, the value of *S* for the stress relaxation test is at least an order of magnitude lower than the values for the two other tests. This creates a natural bias during optimization, providing more consideration to the ratcheting and low-cycle fatigue tests. This trend is not evident with use of the uSHARP objective function. The individual *S* values remain on the same order of magnitude throughout the duration of the optimization process. This is proof that uSHARP has satisfied the goal of allowing for the use of several different test types without undue consideration being given to any single test over the others. This also represents an improvement over the standard objective function so commonly used for automated constitutive model parameter determination.



(a)



(b)

Figure 28: Individual objective function values for three "experimental" tests using (a) the uSHARP objective function (Eq. 7) and (b) standard objective function from literature (Eq. 13)

6. CONCLUSION

The uSHARP optimization routine was successful in determining values for the material constants which created excellent fits with the "experimental" data. While the final converged constants were not close to the "experimental" values in all cases, this is a reflection of an experimental test matrix which is not adequate in capturing the effects of all of the material constants rather than a failure in the functionality of uSHARP. These results highlight the need to verify the adequacy of the experimental test matrix used in any automated parameter determination routine before the results can be considered completely accurate. A procedure using synthetic data where the "experimental" constants are already known, similar to the method used here, would be a good initial step to verify the adequacy of any test matrix before the commitment to actual experiments using test specimens was made.

Use of the uSHARP routine shows great promise in simplifying the procedure of parameter determination. Four of the seven constants were determined to within 4% of their actual "experimental" values using only two tests, when typically more than twenty are used. The addition of a third test type seemed to enable a more accurate determination of two more constants, though these results are unclear at this time due to an error in the experimental data. It is predicted that once this error is accounted for, six of the seven constants will be determined accurately from only three tests. Furthermore, since the uSHARP optimization process is completely contained, calling on an external program (ANSYS) to provide the necessary finite element simulations, it is expected to be adaptable to any constitutive model. In this way, the developed parameter

determination method was designed to be broadly applicable to suit the needs of many different users, as its procedure is not specific to the Miller model.

The form of the objective function used in the uSHARP routine demonstrated superiority over the traditional form commonly found in literature. The goal of enabling the use of multiple test types simultaneously with minimal user input was satisfied, ensuring that every "experimental" test used had equal consideration during the parameter determination process. While the weighting functions were determined through a trial and error process, they have performed favorably. In each case, the minimum of the objective function corresponded to a very good fit.

7. FUTURE WORK

Overall, the development of the described method has been a success, though much work remains to be done. The forms of the weighting functions need to be optimized and additional weighting functions are needed for other test types, such as creep tests. A modification of one of the three test types used here would be favorable in order to better capture the effects of the A_2 constant. Furthermore, simple tests designed to enable the determination of the elastic material parameters will be developed. Once this has been done, the simulated test matrix will be conducted on actual test specimens to verify the functionality of uSHARP under non-ideal conditions. Because the goal of the research was to develop a method which can be applied to a wide range of constitutive models, use of the uSHARP routine must be proven for several different types of constitutive models to ensure broad applicability. The uSHARP routine is currently being used to determine material parameters for a tertiary creep damage model, and the initial results are promising. An analysis of the effects of the initial guess would be pertinent as well, to ensure that a final solution is not completely dependent on the quality of the initial guess. This analysis must consider, however, that the uSHARP routine was developed for application to cases where the orders of magnitude of the constants are already known.

References

Adebanjo, R. O., (1993) "Determination of the material constants for the MATMOD-ReX Constitutive Model." *ASME Winter Annual Meeting*, New Orleans, LA, November 2.

ASM Metals Handbook, Vol. 9, 2004.

Barnes, G., Mixa, M., and Gordon, A. P., (2009) "Design and Application Considerations for a High Speed Rotating Heat Pipe." *47th American Institute of Aeronautics and Astronautics (AIAA) Aerospace Sciences Meeting (ASM)*, Orlando, FL, January 5-8th.

Bell, M. and Pike, M., "Remark on Algorithm 178: Direct Search," *Communications of the ACM* 9(9): 684, September 1966.

Cao, J. and Lin, J., "A study on formulation of objective functions for determining material models," *International Journal of Mechanical Sciences* 50: 193-204, 2008.

Chan, K. S., Lindholm, U. S., Bodner, S. R., Hill, J. T., Weber, R. M., and Meyer, T. G., "Constitutive Modeling for Isotropic Materials (HOST)," National Aeronautics and Space Administration, Lewis Research Center, Ohio, NASA Contractor Report 179522, August 1986.

Corana, A., Marchesi, M., Martini, C. and Ridella, S., "Minimizing Multimodal Functions of Continuous Variables with the 'Simulated Annealing' Algorithm." *ACM Transactions on Mathematical Software* 8(3): 262-280, September 1987.

Hooke, R. and Jeeves, T. A., "Direct Search Solution of Numerical and Statistical Problems." *Journal of the ACM* 8(2): 212-229, April 1961.

Kagawa, H. and Asada, Y., (1983) "Extension of Miller's Constitutive Model for Multiaxial Deformation." *ASME International Conference on Advances in Life Prediction Methods,* Albany, NY, April 18-20.

Kawasaki, N., Felix, S., Kasahara, N., Furukawa, T., Yoshimura, S. and Yagawa, G., (1998) "Automated identification of material constants in inelastic constitutive equations by continuous evolutionary algorithm and massively parallel processors." *The ASME/JSME Pressure Vessels and Piping Conference*, San Diego, CA, July 26-30.

Kuester, J. L. and Mize, J. H., *Optimization Techniques with Fortran* (New York: McGraw Hill, Inc., 1973), 311.

Miller, A., "An Inelastic Constitutive Model for Monotonic, Cyclic, and Creep Deformation: Part I- Equations Development and Analytical Procedures," *Journal of Engineering Materials and Technology* 98: 97-105, April 1976a.

Miller, A., "An Inelastic Constitutive Model for Monotonic, Cyclic, and Creep Deformation: Part II- Application to Type 304 Stainless Steel," *Journal of Engineering Materials and Technology* 98: 106-113, April 1976b.

Mulyadi, M., Rist, M. A., Edwards, L., and Brooks, J. W., "Parameter optimization in constitutive equations for hot forging." *Journal of Materials Processing Technology* 177: 311-314, 2006.

Mustata, R. and Hayhurst, D. R., "Creep constitutive equations for a 0.5Cr 0.5Mo 0.25V ferritic steel in the temperature range 565 °C-675 °C," *International Journal of Pressure Vessels and Piping* 82: 363-372, 2005.

Patel, K. and Stewart, C., "Implementation of the Miller Constitutive Model into an ANSYS User Programmable Feature," UCF Mechanics of Materials Research Group, 2008.

Ramaswamy, V. G., Van Stone, R. H., Dame, L. T., and Laflen, J. H., "Constitutive Modeling for Isotropic Materials," National Aeronautics and Space Administration, Lewis Research Center, Ohio, NASA Contractor Report 174805, April 1984.

Rowley, M. A. and Thornton, E. A., "Constitutive Modeling of the Visco-plastic Response of High Temperature Alloys," *A collection of technical papers* 4: 2189-2206, 1994.

Steichen, J. M., "High Strain-Rate Tensile Properties of AISI Type 304 Stainless Steel," *Journal of Engineering Materials and Technology* 95(3): 182-185, July 1973.

APPENDIX A: SAMPLE ANSYS INPUT FILE

```
Finish
/Clear
!
/TITLE, Uniaxial Miller Model                          ! Title of File
/OUTPUT, C:\Miller_info, txt,,                                ! File which gives procedural
information
/CONFIG, NRES, 20000                                   ! Number of possible results
!
!
/prep7                                                 ! Enters Pre-processor phase
!
! Parameter Declaration
!
! Thermal/Mechanical Conditions
!
length = 1                                             ! Length of Block in mm
area   = length*length                                 ! area of square block mm^2
nodes  = 8                                             ! nodes of block
temp_c =   593                                         ! Temperature in Celcius
temp_k =   temp_c+273.15                               ! Celcius to Kelven Conversion
tmp_ref = 0.0                                          ! Reference temperature in K
temp_m=1800                                            ! Melting Temperature in K
!
! UPF Material Properties
!
*DIM,constants,,8
*VREAD, constants(1), C:\OPT\Constants, txt,,
(E12.5)
Young = 22.5E3
YS = 18.0                    !KSI - 0.2% Yield Strength
EE = 0.0005/10
Poisson = 0.3
A1 = 0.8
A2X= 7.42
A2 = A2X*1E-5
BX = 1.0
B = BX*1E15
C2X = 1.0
C2 = C1X*1E-1
k = 1.9859
H1X = 2.8
H1 = H1X*1E2
H2X = 1.0
H2 = H2X*1E2
n = 5.8
Q = 91000.0
! Determine Initial Drag Stress (ksi)
!
*IF,temp_k,LE,0.6*temp_m,THEN,,,,
AR=1*EXP( (-Q/(0.6*k*temp_m))*(LOG((0.6*temp_m)/temp_k)+1))
*ELSE
AR=EXP(-Q/(k*temp_k))
*ENDIF
tmp1=(EE/(B*AR))**(1/n)
D_o=(YS-0.002*H1)/LOG(tmp1+((tmp1**2)+1)**(1/2))**0.677
!
! Node locations of Block
!
N,1,0,0,0                                              ! Node,number,xcord,ycord,zcord

N,2,1,0,0
N,3,1,1,0
N,4,0,1,0
N,5,0,0,1
N,6,1,0,1
```

```
N,7,1,1,1
N,8,0,1,1
!
! Define Elements and Properties
!
ET, 1, Solid185, 0                                  ! Element Type used
E,1,2,3,4,5,6,7,8                                   ! Creating an element from the nodes
!
! Degress of Freedom
!
D,1,UX,0                                            ! Degree of Freedom, node,
                                                    ! direction, value
D,1,UY,0
D,1,UZ,0
D,2,UY,0
D,2,UZ,0
D,3,UZ,0
D,4,UX,0
D,4,UZ,0
D,5,UX,0
D,5,UY,0
D,6,UY,0
D,8,UX,0
!
! Data Parameters
!
data_freq      = 1                                  ! How often to record data, every 10
steps.
!
load_init_time = .1                                 ! Initial Load Time in seconds
load_mini_time = 1E-8                               ! Minimum Deltim step time in seconds
load_maxi_time = .15                                ! Maximum Deltim step time in seconds
load_ramp_time = 1E-2                               ! Ramp time used in Deltim in seconds
!
! Define Properties of Material 1 (for CMS-UPF)
!
TB,USER,1,1,13                                      ! TB, Lab, MAT, NTEMP, NPTS
TBTEMP, temp_k                                      ! TBTEMP, TEMP, KMOD
TBDATA,1,Young,Poisson,A1,A2,B,C1                   ! UPF Matl Props
TBDATA,7,C2,k,H1,H2,n,Q                             ! UPF Matl Props
TBDATA,13,temp_m                                    ! UPF Matl Props
!
TB,STATE,1,,31,                                     ! Define 4 state variables
TBDATA,1,0,0,0,0,0,0                                ! Assign values to ISV's
TBDATA,7,D_o,0,0,0,0,0                              ! Assign values to ISV's
TBDATA,13,0,0,0,0,0,0                               ! Assign values to ISV's
TBDATA,19,0,0,0,0,0,0                               ! Assign values to ISV's
TBDATA,25,0,0,0,0,0,0                               ! Assign values to ISV's
TBDATA,31,0,                                        ! Assign values to ISV's
!
! Temperature boundary conditions
!
TUNIF, temp_k
TREF, tmp_ref
TOFFST, 0.
!
FINISH
!
! Hystersis Loop Variables
!
n=6                                                 ! Number of Cycles
D_strain=0.005                                      ! Final Strain Level
D_s_ini=0.0005                                      ! Initial Cycle Strain Level
t0=10                                               ! Time Per 1/4th cycle (Quadrant)
Multi=(D_strain/D_s_ini)**(1/(2*n-1))
D_ref=D_s_ini/Multi
!
t_ini=0                                             ! Initial Time
t_inc=t0*4                                          ! Time Increment
t_fin=t_inc*n                                       ! Final Time
*DO,t1,t_ini,t_fin,t_inc
```

```
*IF,t1,EQ,t_fin,EXIT
/SOLU                                                ! Enters Solution Phase
!
! Step 1 (Strain QI)
!
t2 = t1+t0
D_ref=Multi*D_ref                                    ! Time at end of Step
d1 = D_ref                                           ! Displacement of step
Antype, static                                       ! ANTYPE, Antype, Status, LDSTEP,
              !SUBSTEP, Action
nropt,auto                                           ! Newton-Raphson set to auto for step
lnsrch,auto                                          ! Activates line search used with NR
!Nlgeom,0                                            ! Includes large deflection effects,
0=off
Solcontrol, 1                                        ! Optimizes nonlinear solutions
!Cnvtol, F, 1E-10,, 2, minref                        ! CNVTOL, Lab, VALUE, TOLER, NORM,
                             !MINREF
Time, t2                                             ! Time at end of step
Deltim, load_init_time, load_mini_time, load_maxi_time      ! DELTIM, DTIME, DTMIN,
DTMAX, Carry
Autots, 1                                            ! Auto Time Stepping
D, 5, UZ, d1                                         ! Apply displacment to node 5
D, 6, UZ, d1                                         ! Apply displacment to node 6
D, 7, UZ, d1                                         ! Apply displacment to node 7
D, 8, UZ, d1                                         ! Apply displacment to node 8
Outres, All, data_freq                               ! Outputs data to be read by ESOL
OUTRES,SVAR, data_freq
Crplim, 20, 1                                        ! CRPLIM, CRCR, Option, !Creep Ratio
Rate, 1                                              ! Activates Creep for step
Kbc, 0                                               ! Specifies stepped or ramped load,
1=stepped
Solve
!
! Step 2 (Strain - QIV)
!
t2 = t2+t0                                           ! Time at end of Step
d1 = 0                                               ! Displacement of step
Antype, static                                       ! ANTYPE, Antype, Status, LDSTEP,
SUBSTEP, Action
nropt,auto                                           ! Newton-Raphson set to auto for step
lnsrch,auto                                          ! Activates line search used with NR
!Nlgeom,0                                            ! Includes large deflection effects,
0=off
Solcontrol, 1                                        ! Optimizes nonlinear solutions
!Cnvtol, F, 1E-10,, 2, minref                        ! CNVTOL, Lab, VALUE, TOLER, NORM,
MINREF
Time, t2                                             ! Time at end of step
Deltim, load_init_time, load_mini_time, load_maxi_time      ! DELTIM, DTIME, DTMIN,
DTMAX, Carry
Autots, 1                                            ! Auto Time Stepping
D, 5, UZ, d1                                         ! Apply displacment to node 5
D, 6, UZ, d1                                         ! Apply displacment to node 6
D, 7, UZ, d1                                         ! Apply displacment to node 7
D, 8, UZ, d1                                         ! Apply displacment to node 8
Outres, All, data_freq                               ! Outputs data to be read by ESOL
OUTRES,SVAR, data_freq
Crplim, 20, 1                                        ! CRPLIM, CRCR, Option, !Creep Ratio
Limit
Rate, 1                                              ! Activates Creep for step
Kbc, 0                                               ! Specifies stepped or ramped load,
1=stepped
Solve
!
! Step 3 (Strain - QIII)
!
t2 = t2+t0
D_ref=Multi*D_ref                                    ! Time at end of Step
d1 = -D_ref                                          ! Displacement of step
Antype, static                                       ! ANTYPE, Antype, Status, LDSTEP,
SUBSTEP, Action
nropt,auto                                           ! Newton-Raphson set to auto for step
```

```
lnsrch,auto                                          ! Activates line search used with NR
!Nlgeom,0                                           ! Includes large deflection effects,
0=off
Solcontrol, 1                                        ! Optimizes nonlinear solutions
!Cnvtol, F, 1E-10,, 2, minref                        ! CNVTOL, Lab, VALUE, TOLER, NORM,
                              !MINREF
Time, t2                                             ! Time at end of step
Deltim, load_init_time, load_mini_time, load_maxi_time      ! DELTIM, DTIME, DTMIN,
DTMAX, Carry
Autots, 1                                            ! Auto Time Stepping
D, 5, UZ, d1                                         ! Apply displacment to node 5
D, 6, UZ, d1                                         ! Apply displacment to node 6
D, 7, UZ, d1                                         ! Apply displacment to node 7
D, 8, UZ, d1                                         ! Apply displacment to node 8
Outres, All, data_freq                               ! Outputs data to be read by ESOL
OUTRES,SVAR, data_freq
Crplim, 20, 1                                        ! CRPLIM, CRCR, Option, !Creep Ratio
Limit
Rate, 1                                              ! Activates Creep for step
Kbc, 0                                               ! Specifies stepped or ramped load,
1=stepped
Solve
!
! Step 4 (Strain - QII)
!
t2 = t2+t0                                           ! Time at end of Step
d1 = 0                                               ! Displacement of step
Antype, static                                       ! ANTYPE, Antype, Status, LDSTEP,
SUBSTEP, Action
nropt,auto                                           ! Newton-Raphson set to auto for this
step
lnsrch,auto                                          ! Activates a line search used with
Newton-Raphson
!Nlgeom,0                                           ! Includes large deflection effects,
0=off
Solcontrol, 1                                        ! Optimizes nonlinear solutions
!Cnvtol, F, 1E-10,, 2, minref                        ! CNVTOL, Lab, VALUE, TOLER, NORM,
MINREF
Time, t2                                             ! Time at end of step
Deltim, load_init_time, load_mini_time, load_maxi_time      ! DELTIM, DTIME, DTMIN,
                                                     !DTMAX, Carry
Autots, 1                                            ! Auto Time Stepping
D, 5, UZ, d1                                         ! Apply displacment to node 5
D, 6, UZ, d1                                         ! Apply displacment to node 6
D, 7, UZ, d1                                         ! Apply displacment to node 7
D, 8, UZ, d1                                         ! Apply displacment to node 8
Outres, All, data_freq                               ! Outputs data to be read by ESOL
OUTRES,SVAR, data_freq
Crplim, 20, 1                                        ! CRPLIM, CRCR, Option, !Creep Ratio
Rate, 1                                              ! Activates Creep for step
Kbc, 0                                               ! Specifies stepped or ramped load
Solve
!
*ENDDO
Finish
!
/Post26                                              ! Entering Post processor
/OUTPUT, C:\LCF, TXT,,               ! File where data results are outputed
NUMVAR,200
/GROPTS,View,1
!
ESOL,2,1,6,EPCR,Z,EPCRz                                    ! Stores creep in the Z for
node 6
ESOL,3,1,6,EPEL,Z,Elastic_z                          ! Stores elastic in Z for node 6
ESOL,4,1,6,EPPL,Z,Plastic_z                          ! Stores plastic in Z for node 6
ESOL,5,1,6,S,Z,Stress_z                                    ! Stores stress in Z for node
6
ESOL,6,1,6,SVAR,7,Ustatev
XVAR,3                                               ! Graphs in the X axis, time = 0
PLVAR,5                                              ! Graphs in the Y axis
```

```
LINES,300000                                    ! Maximum Number of lines in the data
results file
PRVAR,2,3,4,5,6                                 ! Stores data results in file
!
FINISH
```

APPENDIX B: uSHARP OBJECTIVE FUNCTION

```
!-----------------------------------------------------!
! THIS IS THE MAIN OBJECTIVE FUNCTION THAT WILL BE    !
! MINIMIZED BY THE HOOKE-JEEVES ALGORITHM.  IT SHOULD !
! CONTAIN ALL THE DATA SETS THAT ARE BEING FIT        !
! SIMULTANEOUSLY.                                     !
!-----------------------------------------------------!
! VERSION 1.0  12/17/2008                             !
! AUTHOR: ERIK HOGAN                                  !
!-----------------------------------------------------!
! REVISIONS:                                          !
!-----------------------------------------------------!

FUNCTION OBJECTIVE(X,N)

IMPLICIT NONE

  INTEGER N,I,J,STATUS,RSTATUS
  INTEGER NDIMA1,NDIMA2,NDIMA3
  INTEGER NVALS,NDIME1,NDIME2,NDIME3

  REAL, ALLOCATABLE, DIMENSION(:) :: ANSTIME1,ANSDATA1
  REAL, ALLOCATABLE, DIMENSION(:) :: ANSTIME2,ANSDATA2
  REAL, ALLOCATABLE, DIMENSION(:) :: ANSTIME3,ANSDATA3
  REAL, ALLOCATABLE, DIMENSION(:) :: EXTIME1,EXDATA1
  REAL, ALLOCATABLE, DIMENSION(:) :: EXTIME2,EXDATA2
  REAL, ALLOCATABLE, DIMENSION(:) :: EXTIME3,EXDATA3
  REAL SRFUNC,PRCRFUNC,LCFFUNC
  REAL SUM1,SUM2,SUM3
  REAL X(N),OBJECTIVE,SIG,TTOL

  !-------------------------------------------------------------!
  ! DEFINE STRESS TOLERANCE LEVEL FOR LCF TEST (SIG) AND TIME  !
  ! THRESHOLD (TTOL) FOR CREEP TEST.  SEE ACCOMPANYING         !
  ! FUNCTIONS FOR EXPLANATIONS AS TO WHAT THESE PARAMETERS DO. !
  !-------------------------------------------------------------!
  SIG=14.0
 !TTOL= 0.0  NOT USED


  !-------------------------------------------------------------!
  !Define the location of the constants file that is called by !
  !the ANSYS input file.                                       !
  !-------------------------------------------------------------!
  OPEN (UNIT=10, FILE='C:\OPT\Constants.txt', POSITION='REWIND')

  !-------------------------------------------------------------!
  !This file is used to save the current guess at each         !
  !iteration.  It helps in case you want to see the progression!
  !throughout the optimization routine.                        !
  !-------------------------------------------------------------!
  OPEN (UNIT=15, FILE='GUESS.TXT', POSITION='APPEND')
```

```
  !-------------------------------------------------------------!
  !Define the location of the experimental data file(s).       !
  !-------------------------------------------------------------!
  OPEN (UNIT=20, FILE='lcfex.txt', POSITION='REWIND')
  OPEN (UNIT=21, FILE='ratex.csv', POSITION='REWIND')
  OPEN (UNIT=22, FILE='srex.txt', POSITION='REWIND')

  !-------------------------------------------------------------!
  ! ALL VARIABLES THAT WILL BE DETERMINED BY THE PROGRAM MUST  !
  ! BE INITIALIZED TO 0.                                       !
  !-------------------------------------------------------------!
  NDIME1=0
  NDIME2=0
  NDIME3=0
  STATUS=0
  NDIMA1=0
  NDIMA2=0
  NDIMA3=0
  NVALS=0
  RSTATUS=0
  !-------------------------------------------------------------------!
  ! THE CURRENT GUESS NEEDS TO BE WRITTEN TO THE FILE THAT WILL   !
  ! BE CALLED BY THE INPUT DECK.                                  !
  !-------------------------------------------------------------------!
  DO I=1,N
    WRITE (10,11) X(I)
  ENDDO
  CLOSE (10)
  11 FORMAT(E12.5)
  !-------------------------------------------------------------!
  ! SAVE THE CURRENT GUESS TO THE 'GUESS.TXT' FILE FOR REVIEW. !
  !-------------------------------------------------------------!

  DO I=1,N
    WRITE (15,16) X(I)
  ENDDO
  WRITE (15,*)'------------------'
  CLOSE (15)
  16 FORMAT(E12.5)

 !-------------------------------------------------------------!
 ! FORCE THE GUESS TO REMAIN POSITIVE.                         !
 !-------------------------------------------------------------!
   DO I=1,N
    IF (X(I).LE.0) THEN
        SUM1=1E10
        SUM2=1E10
        SUM3=1E10
        GOTO 955
    ENDIF
   ENDDO

!-----------------------------------------------------!
! DETERMINE THE NUMBER OF DATA POINTS IN EACH DATASET   !
!-----------------------------------------------------!
    !DATASET 1!
    DO
```

```
        READ (20,*,IOSTAT=STATUS)
        IF (STATUS /= 0) EXIT
        NDIME1=NDIME1+1
    ENDDO
    REWIND (20)

    STATUS=0

    !DATASET 2!
    DO
        READ (21,*,IOSTAT=STATUS)
        IF (STATUS /= 0) EXIT
        NDIME2=NDIME2+1
    ENDDO
    REWIND (21)

    STATUS=0

    !DATASET 3!
    DO
        READ (22,*,IOSTAT=STATUS)
        IF (STATUS /= 0) EXIT
        NDIME3=NDIME3+1
    ENDDO
    REWIND (22)


  !ALLOCATE ARRAY SIZES!
  ALLOCATE (EXTIME1(NDIME1),EXDATA1(NDIME1))
  ALLOCATE (EXTIME2(NDIME2),EXDATA2(NDIME2))
  ALLOCATE (EXTIME3(NDIME3),EXDATA3(NDIME3))

!-----------------------------------------------------!
! READ IN THE EXPERIMENTAL DATASETS.                  !
!-----------------------------------------------------!
  DO I=1,NDIME1
    READ (20,23) EXTIME1(I),EXDATA1(I)
  ! PRINT *,EXTIME1(2),' ',EXDATA1(I)
  ENDDO

  CLOSE (20)

  DO I=1,NDIME2
   READ (21,695) EXTIME2(I),EXDATA2(I)
  ENDDO
  CLOSE (21)

  DO I=1,NDIME3
    READ (22,23) EXTIME3(I),EXDATA3(I)
  ! PRINT *,EXTIME3(I),' ',EXDATA3(I)
  ENDDO
  CLOSE (22)

23 FORMAT(E12.5,E12.5)
695 FORMAT(F8.2,F12.5)

!-----------------------------------------------------!
```

```
! AT THIS POINT, THE EXPERIMENTAL DATASETS HAVE BEEN  !
! DEFINED.  THE ANSYS SIMULATIONS FOR EACH DATASET    !
! NEED TO BE RUN.  THIS IS DONE HERE.                 !
!-----------------------------------------------------!
CALL INPROGRESS
call SYSTEM(' "C:\OPT\ANSYS1.BAT" ')
CALL INPROGRESS
call SYSTEM(' "C:\OPT\ANSYS2.BAT" ')
CALL INPROGRESS
call SYSTEM(' "C:\OPT\ANSYS3.BAT" ')

!-----------------------------------------------------!
!OPEN ANSYS OUTPUT FILES                              !
!----------------------!------------------------------!
! DEFINE THE LOCATION OF THE ANSYS OUTPUT FILES HERE.  !
! THESE WILL BE USED TO CALCULATE THE ARRAY SIZES AND  !
! DEFINE THE DATASETS FOR THE ANSYS SIMULATION VECTORS. !
!------------------------------=-----------------------!
   OPEN (UNIT=30, FILE='C:\LCF.TXT', POSITION='REWIND')
   OPEN (UNIT=31, FILE='C:\RAT.TXT', POSITION='REWIND')
   OPEN (UNIT=32, FILE='C:\SR.TXT', POSITION='REWIND')


!-------------------------------------------------------------------!
! THIS IS WHERE THINGS GET TRICKY.  EACH OUTPUT FILE                !
! WILL BE OF A DIFFERENT FORMAT, NECESSITATING AN                   !
! INDIVIDUAL FORMAT STATEMENT FOR EACH OUTPUT FILE.                 !
! HERE, THE SIZE OF THE SIMULATION ARRAYS ARE DETERMINED            !
! AND READ INTO THE PROGRAM.                                        !
!-------------------------------------------------------------------!

!DETERMINE SIZE OF ARRAYS!
    STATUS=0
    DO
        READ (30,*,IOSTAT=STATUS)
        IF (STATUS /= 0) EXIT
        NVALS=NVALS+1
    ENDDO
    REWIND (30)
    NDIMA1=NVALS-69
    NVALS=0
    !PRINT *,NDIMA1

    STATUS=0
    DO
        READ (31,*,IOSTAT=STATUS)
        IF (STATUS /= 0) EXIT
        NVALS=NVALS+1
    ENDDO
    REWIND (31)
    NDIMA2=NVALS-69
    NVALS=0
    !PRINT *,NDIMA2

    STATUS=0
    DO
        READ (32,*,IOSTAT=STATUS)
```

```
        IF (STATUS /= 0) EXIT
        NVALS=NVALS+1
    ENDDO
    REWIND (32)
    NDIMA3=NVALS-67
    NVALS=0
    !PRINT *,NDIMA3

!-----------------------------------------!
! ALLOCATE ARRAY SIZES.                   !
!-----------------------------------------!

    ALLOCATE (ANSTIME1(NDIMA1),ANSDATA1(NDIMA1))
    ALLOCATE (ANSTIME2(NDIMA2),ANSDATA2(NDIMA2))
    ALLOCATE (ANSTIME3(NDIMA3),ANSDATA3(NDIMA3))

!-----------------------------------------------------------!
! READ IN THE SIMULATED DATA SETS.  ENSURE THE FORMATTING   !
! STATEMENT FOR EACH DATASET IS CORRECTLY DEFINED OR THE    !
! SETS WILL NOT BE READ IN CORRECTLY.                       !
!-----------------------------------------------------------!
    !DATASET 1-LCF!
    READ (30,41) ANSTIME1(1),ANSDATA1(1)
     DO I=2,NDIMA1
        READ (UNIT=30,FMT=42,IOSTAT=RSTATUS) ANSTIME1(I),ANSDATA1(I)
        IF (RSTATUS /= 0) THEN
        SUM1=2E10
        SUM2=2E10
        SUM3=2E10
        GOTO 955
        ENDIF
     ENDDO
     CLOSE (30)
41 FORMAT(52/,F13.5,T59,F13.6)
42 FORMAT(F13.5,T59,F13.6)

    !DATASET 2-Ratcheting!
    RSTATUS=0
    READ (31,43) ANSTIME2(1),ANSDATA2(1)
     DO I=2,NDIMA2
        READ (UNIT=31,FMT=44,IOSTAT=RSTATUS) ANSTIME2(I),ANSDATA2(I)
        IF (RSTATUS /= 0) THEN
        SUM2=2E10
        SUM3=2E10
        GOTO 955
        ENDIF
     ENDDO
     CLOSE (31)
43 FORMAT(52/,F13.5,T59,E13.6)
44 FORMAT(F13.5,T59,E13.6)

    !DATASET 3-STRESS RELAXATION!
    RSTATUS=0
    READ (32,45) ANSTIME3(1),ANSDATA3(1)
     DO I=2,NDIMA3
        READ (UNIT=32,FMT=46,IOSTAT=RSTATUS) ANSTIME3(I),ANSDATA3(I)
        IF (RSTATUS /= 0) THEN
```

```
        SUM3=2E10
        GOTO 955
        ENDIF
      ENDDO
      CLOSE (32)
45 FORMAT(50/,F13.5,T61,F11.6)
46 FORMAT(F13.5,T61,F11.6)


!-------------------------------------------------------------!
! ALL NECESSARY DATASETS HAVE BEEN DEFINED.  THE OBJECTIVE    !
! FUNCTION WILL NOW BE EVALUATED.                             !
!-------------------------------------------------------------!

!LCF DATA!
SUM1=LCFFUNC(ANSDATA1,EXDATA1,SIG,NDIME1,NDIMA1,ANSTIME1,EXTIME1)

!CREEP DATA!
SUM2=LCFFUNC(ANSDATA2,EXDATA2,16.0,NDIME2,NDIMA2,ANSTIME2,EXTIME2)

!STRESS RELAXATION DATA!
SUM3=SRFUNC(ANSDATA3,ANSTIME3,EXDATA3,EXTIME3,NDIMA3,NDIME3)

DEALLOCATE (ANSTIME1,ANSDATA1)
DEALLOCATE (ANSTIME2,ANSDATA2)
DEALLOCATE (ANSTIME3,ANSDATA3)
DEALLOCATE (EXTIME1,EXDATA1)
DEALLOCATE (EXTIME2,EXDATA2)
DEALLOCATE (EXTIME3,EXDATA3)

955 CONTINUE

OBJECTIVE=SUM1+SUM2+SUM3


!-------------------------------------------------------------------!
! THE VALUE OF THE OBJECTIVE FUNCTION WILL NOW BE RECORDED TO       !
! FILE.                                                             !
!-------------------------------------------------------------------!

    OPEN (UNIT=50, FILE='FOM.TXT', POSITION='APPEND')
    WRITE (50,51) OBJECTIVE

    CLOSE (50)
51 FORMAT(E12.5)
52 FORMAT(E12.5,' ','+')
    OPEN (UNIT=100, FILE='SUMS.TXT', POSITION='APPEND')
    WRITE (100,*) SUM1, '  ',SUM2,'  ',SUM3

RETURN
END
```

```
!-----------------------------------------------------!
! THIS FUNCTION RETURNS THE NORMALIZED LEAST SQUARES  !
! VALUE BETWEEN TWO DATASETS.  THE USER IS REQUIRED   !
! TO INPUT THE TWO Y VALUE DATASETS, THE WEIGHT       !
! VECTOR, AND THE NUMBER OF DIMENSIONS OF THE DATA.   !
!-----------------------------------------------------!

FUNCTION NLSTSQR(YEX,YSIM,WEIGHT,NDIMA)

IMPLICIT NONE
INTEGER NDIMA,I
REAL YEX(NDIMA),YSIM(NDIMA),WEIGHT(NDIMA)
REAL NLSTSQR,YMAX,DIFF

!-----------------------------------------------------!
! DETERMINE MAX VALUE IN THE EXPERIMENTAL DATASET.    !
! THIS WILL BE USED TO NORMALIZE THE LEAST SQUARES    !
! CALCULATION.                                        !
!-----------------------------------------------------!

YMAX=0
DO I=1,NDIMA
    IF (ABS(YEX(I)).GT.YMAX) THEN
        YMAX=ABS(YEX(I))
    ELSE
        YMAX=YMAX
    ENDIF
ENDDO

!-----------------------------------------------------!
! PERFORM WEIGHTED LEAST SQUARES CALCULATION.         !
!-----------------------------------------------------!
DIFF=0
DO I=1,NDIMA
    DIFF=DIFF+(WEIGHT(I)*(100*(YEX(I)-YSIM(I))/YMAX)**2)
ENDDO

NLSTSQR=DIFF/NDIMA

RETURN
END
```

```
!-----------------------------------------------------!
! SRFUNC IS AN OBJECTIVE FUNCTION THAT AUTOMATICALLY  !
! WEIGHTS AND CALCULATES THE LEAST SQUARES VALUE FOR  !
! A STRESS RELAXATION TEST.  THE USER ONLY NEEDS TO   !
! SUPPLY THE EXPERIMENTAL DATA SET, FEM DATA SET, AND !
! THEIR CORRESPONDING DIMENSIONS. THE ROUTINE USES A  !
! COMBINATION OF THE FIRST AND SECOND DERIVATIVES OF  !
! THE STRESS VS TIME DATA TO GENERATE THE WEIGHTS.    !
!-----------------------------------------------------!
```

```
FUNCTION SRFUNC(ANSSTRESS,ANSTIME,EXSTRESS,EXTIME,NDIMA,NDIME)

IMPLICIT NONE
INTEGER NDIMA,NDIME,I
REAL ANSSTRESS(NDIMA),EXSTRESS(NDIME),STRESSINTERP(NDIMA)
REAL ANSTIME(NDIMA),EXTIME(NDIME)
REAL WEIGHT(NDIMA),SIGDOT(NDIMA),SIGDUB(NDIMA)
REAL DIFF,SRFUNC,NLSTSQR

!-----------------------------------------------------!
! FIRST, INTERPOLATE STRESS VALUES SO THEY CORRESPOND!
! TO THE SAME TIME INDICES.                           !
!-----------------------------------------------------!

CALL INTERP(ANSTIME,EXTIME,EXSTRESS,NDIMA,NDIME,STRESSINTERP)

!-----------------------------------------------------!
! PERFORM FIRST AND SECOND DERIVATIVE CALCULATIONS    !
! AND GENERATE THE WEIGHT FOR EACH TIME POINT. THE    !
! FIRST AND LAST POINTS ARE NEGLECTED FOR SIMPLICITY.!
!-----------------------------------------------------!

DO I=2,NDIMA-1
    !STRESS RATE CALCULATION!
    SIGDOT(I)=(STRESSINTERP(I+1)-STRESSINTERP(I))/(ANSTIME(I+1)-
ANSTIME(I))
    !SECOND DERIVATIVE CALCULATION!
    SIGDUB(I)=(STRESSINTERP(I+1)-2*STRESSINTERP(I)+STRESSINTERP(I-
1))/((ANSTIME(I+1)-ANSTIME(I))*(ANSTIME(I)-ANSTIME(I-1)))
    !GENERATE WEIGHTS!
    WEIGHT(I)=(0.5+ABS(SIGDOT(I))+ABS(SIGDUB(I)))/abs((0.5-
ABS(SIGDOT(I))-ABS(SIGDUB(I))))
ENDDO

WEIGHT(1)=0
WEIGHT(NDIMA)=0
!-----------------------------------------------------!
! DUE TO EXPERIMENTAL DISCREPANCIES, APPROXIMATE      !
! DERIVATIVES MAY LEAD TO VERY HIGH WEIGHTS FOR       !
! CERTAIN INCONSISTENT DATA POINTS.  THIS CORRECTS    !
! FOR THAT BY LIMITING THE MAXIMUM WEIGHT FOR ANY     !
! POINT TO 2.5.                                       !
!-----------------------------------------------------!

DO I=2,NDIMA-1
    IF (WEIGHT(I).GT.(2.5)) THEN
        WEIGHT(I)=2.5
    ELSE
        WEIGHT(I)=WEIGHT(I)
    ENDIF
ENDDO

!-----------------------------------------------------!
! CALCULATE NORMALIZED LEAST SQUARES VALUE.           !
!-----------------------------------------------------!
```

```
SRFUNC=NLSTSQR(STRESSINTERP,ANSSTRESS,WEIGHT,NDIMA)

RETURN
END




!-----------------------------------------------------------!
! LCFFUNC IS A FUNCTION TO AUTOMATICALLY WEIGHT AN LCF      !
! HYSTERESIS LOOP.  THE USER PROVIDES A STRESS VALUE, SIG,  !
! ABOVE WHICH THE FUNCTION WEIGHTS THE LEAST SQUARES        !
! VALUES HIGHER THAN THOSE OF THE ELASTIC DEFORMATION REGIONS.  !
! IDEALLY, SIG SHOULD REFLECT THE LOWEST STRESS VALUE AT WHICH  !
! PLASTIC DEFORMATION BEGINS.  THE ROUTINE WILL PENALIZE ANY          !
! GUESS THAT DOES NOT CAUSE STRESSES ABOVE SIG, ENSURING THE          !
! BEST FIT FALLS IN THE APPROPRIATE PLASTIC DEFORMATION REGION.       !
!---------------------------------------------------------------------!

FUNCTION LCFFUNC(ANSSTRESS,EXSTRESS,SIG,NDIME,NDIMA,ANSTIME,EXTIME)

IMPLICIT NONE
INTEGER I,NDIMA,NDIME
REAL ANSSTRESS(NDIMA),ANSTIME(NDIMA),EXSTRESS(NDIME),EXTIME(NDIME)
REAL STRESSINTERP(NDIMA),WEIGHT(NDIMA)
REAL SIG,NEGSIG,DIFFA,DIFFB,DIFFC,LCFFUNC,NLSTSQR

!-------------------------------------------------------------!
! FIRST, INTERPOLATE THE STRESS VALUES SO THEY CORRESPOND TO  !
! THE SAME TIME INDICES.                                      !
!-------------------------------------------------------------!

CALL INTERP(ANSTIME,EXTIME,EXSTRESS,NDIMA,NDIME,STRESSINTERP)


!-------------------------------------------------------------!
! BEGIN WEIGHTING FUNCTION, FIRST INITIALIZING DIFFERENCE VALUES!
! TO 0.                                                       !
!-------------------------------------------------------------!

NEGSIG=SIG*-1

DO I=1,NDIMA
    IF (STRESSINTERP(I).LT.NEGSIG) THEN
        WEIGHT(I)=2.5
    ELSEIF (STRESSINTERP(I).GT.SIG) THEN
        WEIGHT(I)=2.5
    ELSE
        WEIGHT(I)=1.0
    ENDIF
ENDDO
```

```
!-------------------------------------------------------------!
! CALCULATE TOTAL LEAST SQUARES VALUE AND AVERAGE IT ACROSS THE!
! NUMBER OF DATA POINTS USED.                                  !
!-------------------------------------------------------------!

LCFFUNC=NLSTSQR(STRESSINTERP,ANSSTRESS,WEIGHT,NDIMA)

RETURN

END
```

APPENDIX C: HOOKE-JEEVES ALGORITHM SOURCE

```
function hooke ( nvars, startpt, endpt, rho, eps, itermax, f )

!*******************************************************************************8
0
!
!! HOOKE seeks a minimizer of a scalar function of several variables.
!
!  Discussion:
!
!    This routine find a point X where the nonlinear objective function
!    F(X) has a local minimum.  X is an N-vector and F(X) is a scalar.
!    The objective function F(X) is not required to be differentiable
!    or even continuous.  The program does not use or require derivatives
!    of the objective function.
!
!    The user supplies three things:
!    1) a subroutine that computes F(X),
!    2) an initial "starting guess" of the minimum point X,
!    3) values for the algorithm convergence parameters.
!
!    The program searches for a local minimum, beginning from the
!    starting guess, using the Direct Search algorithm of Hooke and
!    Jeeves.
!
!    This program is adapted from the Algol pseudocode found in the
!    paper by Kaupe, and includes improvements suggested by Bell and Pike,
!    and by Tomlin and Smith.
!
!    The algorithm works by taking "steps" from one estimate of
!    a minimum, to another (hopefully better) estimate.  Taking
!    big steps gets to the minimum more quickly, at the risk of
!    "stepping right over" an excellent point.  The stepsize is
!    controlled by a user supplied parameter called RHO.  At each
!    iteration, the stepsize is multiplied by RHO  (0 < RHO < 1),
!    so the stepsize is successively reduced.
!
!    Small values of rho correspond to big stepsize changes,
!    which make the algorithm run more quickly.  However, there
!    is a chance (especially with highly nonlinear functions)
!    that these big changes will accidentally overlook a
!    promising search vector, leading to nonconvergence.
!
!    Large values of RHO correspond to small stepsize changes,
!    which force the algorithm to carefully examine nearby points
!    instead of optimistically forging ahead.  This improves the
!    probability of convergence.
!
!    The stepsize is reduced until it is equal to (or smaller
!    than) EPS.  So the number of iterations performed by
!    Hooke-Jeeves is determined by RHO and EPS:
```

```
!
!      RHO^(number_of_iterations) = EPS
!
!    In general it is a good idea to set RHO to an aggressively
!    small value like 0.5 (hoping for fast convergence).  Then,
!    if the user suspects that the reported minimum is incorrect
!    (or perhaps not accurate enough), the program can be run
!    again with a larger value of RHO such as 0.85, using the
!    result of the first minimization as the starting guess to
!    begin the second minimization.
!
!    Normal use:
!    (1) Code your function F() in the C language;
!    (2) Install your starting guess;
!    (3) Run the program.
!
!    If there are doubts about the result, the computed minimizer
!    can be used as the starting point for a second minimization attempt.
!
!    To apply this method to data fitting, code your function F() to be
!    the sum of the squares of the errors (differences) between the
!    computed values and the measured values.  Then minimize F()
!    using Hooke-Jeeves.
!
!    For example, you have 20 datapoints (T(i), Y(i)) and you want to
!    find A, B and C so that:
!
!      A*t*t + B*exp(t) + C*tan(t)
!
!    fits the data as closely as possible.  Then the objective function
!    F() to be minimized is just
!
!      F(A,B,C) = sum ( 1 <= i <= 20 )
!        ( y(i) - A*t(i)*t(i) - B*exp(t(i)) - C*tan(t(i)) )^2.
!
!  Modified:
!
!    12 February 2008
!
!  Author:
!
!    ALGOL original by Arthur Kaupe.
!    C version by Mark Johnson.
!    FORTRAN90 version by John Burkardt.
!
!  Reference:
!
!    M Bell, Malcolm Pike,
!    Remark on Algorithm 178: Direct Search,
!    Communications of the ACM,
!    Volume 9, Number 9, September 1966, page 684.
!
!    Robert Hooke, Terry Jeeves,
!    Direct Search Solution of Numerical and Statistical Problems,
!    Journal of the ACM,
```

```
!    Volume 8, Number 2, April 1961, pages 212-229.
!
!    Arthur Kaupe,
!    Algorithm 178:
!    Direct Search,
!    Communications of the ACM,
!    Volume 6, Number 6, June 1963, page 313.
!
!    FK Tomlin, LB Smith,
!    Remark on Algorithm 178: Direct Search,
!    Communications of the ACM,
!    Volume 12, Number 11, November 1969, page 637-638.
!
!  Parameters:
!
!    Input, integer ( kind = 4 ) NVARS, the number of spatial dimensions.
!
!    Input, real ( kind = 8 ) STARTPT(NVARS), the user-supplied
!    initial estimate for the minimizer.
!
!    Output, real ( kind = 8 ) ENDPT(NVARS), the estimate for the
!    minimizer, as calculated by the program.
!
!    Input, real ( kind = 8 ) RHO, a user-supplied convergence parameter
!    which should be set to a value between 0.0 and 1.0.  Larger values
!    of RHO give greater probability of convergence on highly nonlinear
!    functions, at a cost of more function evaluations.  Smaller
!    values of RHO reduce the number of evaluations and the program
!    running time, but increases the risk of nonconvergence.
!
!    Input, real ( kind = 8 ) EPS, the criterion for halting
!    the search for a minimum.  When the algorithm
!    begins to make less and less progress on each
!    iteration, it checks the halting criterion: if
!    the stepsize is below EPS, terminate the
!    iteration and return the current best estimate
!    of the minimum.  Larger values of EPS (such
!    as 1.0e-4) give quicker running time, but a
!    less accurate estimate of the minimum.  Smaller
!    values of EPS (such as 1.0e-7) give longer
!    running time, but a more accurate estimate of
!    the minimum.
!
!    Input, integer ( kind = 4 ) ITERMAX, a limit on the number of iterations.
!
!    Input, external real ( kind = 8 ) F, the name of the function routine,
!    which should have the form:
!      function f ( x, n )
!      integer ( kind = 4 ) n
!      real ( kind = 8 ) f
!      real ( kind = 8 ) x(n)
!
!    Output, integer ( kind = 4 ) HOOKE, the number of iterations taken.
!
  implicit none
```

```
integer nvars

real best_nearby
real delta(nvars)
real endpt(nvars)
real eps
real, external :: f
real fbefore
integer funevals
integer hooke
integer i
integer itermax
integer iters
integer j
integer keep
real newf
real newx(nvars)
real rho
real startpt(nvars)
real steplength
real tmp
logical, parameter :: verbose = .false.
real xbefore(nvars)

do i = 1, nvars
  newx(i) = startpt(i)
end do

do i = 1, nvars
  xbefore(i) = startpt(i)
end do

do i = 1, nvars
  if ( startpt(i) == 0.0D+00 ) then
    delta(i) = rho
  else
    delta(i) = rho * abs ( startpt(i) )
  end if
end do

funevals = 0
steplength = rho
iters = 0
fbefore = f ( newx, nvars )
funevals = funevals + 1
newf = fbefore

do while ( iters < itermax .and. eps < steplength )

  iters = iters + 1

  if ( verbose ) then

    write ( *, '(a)' ) ' '
```

```
      write ( *, '(a,i8,a,g14.6)' ) &
      '  FUNEVALS, = ', funevals, '  F(X) = ', fbefore

      do j = 1, nvars
        write ( *, '(2x,i8,2x,g14.6)' ) j, xbefore(j)
      end do

    end if
!
!  Find best new point, one coordinate at a time.
!
    do i = 1, nvars
      newx(i) = xbefore(i)
    end do

    newf = best_nearby ( delta, newx, fbefore, nvars, f, funevals )
!
!  If we made some improvements, pursue that direction.
!
    keep = 1

    do while ( newf < fbefore .and. keep == 1 )

      do i = 1, nvars
!
!  Arrange the sign of DELTA.
!
        if ( newx(i) <= xbefore(i) ) then
          delta(i) = - abs ( delta(i) )
        else
          delta(i) = abs ( delta(i) )
        end if
!
!  Now, move further in this direction.
!
        tmp = xbefore(i)
        xbefore(i) = newx(i)
        newx(i) = newx(i) + newx(i) - tmp
      end do

      fbefore = newf
      newf = best_nearby ( delta, newx, fbefore, nvars, f, funevals )
!
!  If the further (optimistic) move was bad...
!
      if ( fbefore <= newf ) then
        exit
      end if
!
!  Make sure that the differences between the new and the old points
!  are due to actual displacements; beware of roundoff errors that
!  might cause NEWF < FBEFORE.
!
      keep = 0
```

```
      do i = 1, nvars
        if ( 0.5D+00 * abs ( delta(i) ) < &
          abs ( newx(i) - xbefore(i) ) ) then
          keep = 1
          exit
        end if
      end do

    end do

    if ( eps <= steplength .and. fbefore <= newf ) then
      steplength = steplength * rho
      do i = 1, nvars
        delta(i) = delta(i) * rho
      end do
    end if

  end do

  do i = 1, nvars
    endpt(i) = xbefore(i)
  end do

  hooke = iters

  return
end

function best_nearby ( delta, point, prevbest, nvars, f, funevals )

!*******************************************************************************8
0
!
!! BEST_NEARBY looks for a better nearby point, one coordinate at a time.
!
!  Modified:
!
!    12 February 2008
!
!  Author:
!
!    The ALGOL original is by Arthur Kaupe.
!    C version by Mark Johnson
!    FORTRAN90 version by John Burkardt
!
!  Reference:
!
!    M Bell, Malcolm Pike,
!    Remark on Algorithm 178: Direct Search,
!    Communications of the ACM,
!    Volume 9, Number 9, September 1966, page 684.
!
!    Robert Hooke, Terry Jeeves,
!    Direct Search Solution of Numerical and Statistical Problems,
!    Journal of the ACM,
```

```
!    Volume 8, Number 2, April 1961, pages 212-229.
!
!    Arthur Kaupe,
!    Algorithm 178:
!    Direct Search,
!    Communications of the ACM,
!    Volume 6, Number 6, June 1963, page 313.
!
!    FK Tomlin, LB Smith,
!    Remark on Algorithm 178: Direct Search,
!    Communications of the ACM,
!    Volume 12, Number 11, November 1969, page 637-638.
!
!  Parameters:
!
!    Input, real ( kind = 8 ) DELTA(NVARS), the size of a step in each
direction.
!
!    Input/output, real ( kind = 8 ) POINT(NVARS); on input, the current
candidate.
!    On output, the value of POINT may have been updated.
!
!    Input, real ( kind = 8 ) PREVBEST, the minimum value of the function seen
!    so far.
!
!    Input, integer ( kind = 4 ) NVARS, the number of variables.
!
!    Input, external real ( kind = 8 ) F, the name of the function routine,
!    which should have the form:
!      function f ( x, n )
!      integer ( kind = 4 ) n
!      real ( kind = 8 ) f
!      real ( kind = 8 ) x(n)
!
!    Input/output, integer ( kind = 4 ) FUNEVALS, the number of function
evaluations.
!
!    Output, real ( kind = 8 ) BEST_NEARBY, the minimum value of the function
seen
!    after checking the nearby neighbors.
!
  implicit none

  integer nvars

  real best_nearby
  real delta(nvars)
  real, external :: f
  real ftmp
  integer funevals
  integer i
  real minf
  real point(nvars)
  real prevbest
  real z(nvars)
```

```
  minf = prevbest

  do i = 1, nvars
   z(i) = point(i)
  end do

  do i = 1, nvars

    z(i) = point(i) + delta(i)

    ftmp = f ( z, nvars )
    funevals = funevals + 1

    if ( ftmp < minf ) then

      minf = ftmp

    else

      delta(i) = - delta(i)
      z(i) = point(i) + delta(i)
      ftmp = f ( z, nvars )
      funevals = funevals + 1

      if ( ftmp < minf ) then
        minf = ftmp
      else
        z(i) = point(i)
      end if

    end if

  end do

  do i = 1, nvars
    point(i) = z(i)
  end do

  best_nearby = minf

  return
end
```