Kinetic Model Reduction using Integer and Semi-infinite Programming

by

Binita Bhattacharjee

Submitted to the Department of Chemical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Chemical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

December 2003

Author .....
Department of Chemical Engineering
December 22, 2003

Certified by .....
Paul I. Barton
Associate Professor, Department of Chemical Engineering
Thesis Supervisor

Certified by .....
William H. Green
Associate Professor, Department of Chemical Engineering
Thesis Supervisor

Accepted by .....
Daniel Blankshtein
Chairman, Committee on Graduate Students

Kinetic Model Reduction using Integer and Semi-infinite Programming

by

Binita Bhattacharjee

Submitted to the Department of Chemical Engineering
on December 22, 2003, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Chemical Engineering

Abstract

In this work an optimization-based approach to kinetic model reduction was studied with a view to generating reduced-model libaries for reacting-flow simulations. A linear integer formulation of the reaction elimination problem was developed in order to allow the model reduction problem to be solved cheaply and robustly to guaranteed global optimality. When compared with three other conventional reaction-elimination methods, only the integer-programming approach consistently identified the smallest reduced model which satisfies user-specified accuracy criteria. The proposed reaction elimination formulation was solved to generate model libraries for both, homogeneous combustion systems, and 2-D laminar flames. Good agreement was observed between the reaction trajectories predicted by the full mechanism and the reduced model library. For kinetic mechanisms having many more reactions than species, the computational speedup associated with reaction elimination was found to scale linearly with the size of the derived reduced model. Speedup factors of 4-90 were obtained for a variety of different mechanisms and reaction conditions. The integer-programming based reduction approach was tested successfully on large-scale mechanisms comprising upto ~ 2500 reactions.

The problem of identifying optimal (maximum) ranges of validity for point-reduced kinetic models was also investigated. A number of different formulations for the range problem were proposed, all of which were shown to be variants of a standard semi-infinite program (SIP). Conventional algorithms for nonlinear semi-infinite programs are essentially all lower-bounding methods which cannot guarantee the feasibility of an incumbent at finite termination. Thus, they cannot be used to identify rigorous ranges of validity for reduced kinetic models. In the second part of this thesis, inclusion functions were used to develop an inner approximation method which generates a convergent series of feasible upper bounds on the minimum value of a smooth, nonlinear semi-infinite program. The inclusion-constrained reformulation approach was applied successfully to a number of test problems in the SIP literature. The new upper-bounding approach was then combined with existing lower-bounding methods in a branch-and-bound framework which allows smooth nonlinear semi-infinite pro-

grams to be solved finitely to ϵ-optimality. The branch-and-bound algorithm was also tested on a number of small literature examples. In the final chapter of the thesis, extensions of the existing algorithm and code to solve practical engineering problems, including the range identification problem, were considered.

Thesis Supervisor: Paul I. Barton
Title: Associate Professor, Department of Chemical Engineering

Thesis Supervisor: William H. Green
Title: Associate Professor, Department of Chemical Engineering

To my parents, Rama Prasad and Mridula

Acknowledgments

I have had the privilege of being part of two excellent research groups at MIT. This thesis would not have seen its completion (or indeed, its inception) without the technical guidance, energy, patience, and optimism of my advisers, Paul Barton and Bill Green. I am also indebted to many past and present members of the Green Group and the Process Systems Engineering Laboratory for all the advice, camaraderie and encouragement volunteered over the years. In particular, I would like to thank Doug Schwer, Pisi Lu, Luwi Oluwoleo, Ed Gatzke, John Tolsma and Adam Singer for their generous contributions in the way of code, technical input and time.

The dedication in this thesis is a testament to the premium my parents have always placed on education, and to the innumerable sacrifices they have made on my account. I am also fortunate to have a wonderful sister and a very understanding extended family. They have always been supportive of my educational goals, despite my long absences from home, sometimes during important family events.

I cannot begin to imagine how different my graduate experience would have been without my friends, new and old, at and beyond MIT. In person and over thousands of miles, they have encouraged and coaxed me, gently bullied me, and occasionally provided much-needed distraction. I shall not attempt to mention everybody by name; I hope you know who you are, and how I grateful I am to you for making my time here so worthwhile, in and outside of Building 66.

Lastly, I would like to acknowledge financial support for this thesis from the following funding agencies: Alstom Power, the EPA Center for Airborne Organics the U.S.Department of Energy, and the National Science Foundation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation: Adaptive Chemistry for Reacting-flow Simulations

In recent years competitive markets and stricter environmental regulations have motivated increasingly comprehensive reactor modeling in the chemical process industries. Improved experimental techniques and the emergence of automated mechanism generation tools, e.g., [68], have enabled the development of complex kinetic mechanisms for modeling pollutant and byproduct formation. These large-scale mechanisms, often comprising hundreds of species and thousands of reactions, can now be simulated in the order of seconds for perfectly homogeneous systems. However, many exothermic industrial processes are significantly inhomogeneous. In such cases the chemical activity is strongly coupled to heat and mass transfer in the reacting-flow system. Unfortunately, detailed simulation of such complex reacting flows remains computationally prohibitive, despite rapid escalation in hardware capacity.

Simulating a reacting flow amounts to solving a system of partial differential equations (PDEs) comprising conservation equations for mass, momentum and energy, equations of state, and equations describing the kinetic source terms. The stiffness of the embedded kinetics often makes the chemical source term the most computationally intensive to solve. For example, in combustion applications, the range of

chemical time scales is typically two orders of magnitude larger than the range of diffusion-advection time scales [45, 62], and accounts for ∼90% of the computational load [71].

The conflicting needs for 'detailed' simulation results and reasonable CPU/memory loads have limited industry practice to the use of 'skeletal' models. Even when comprehensive mechanisms are available for the system of interest, a reduced model consisting of ∼10 species, ∼20 elementary reactions is selected by inspection to be included in the reacting-flow simulation. However, the accuracy compromised by this skeletal model in place of the full mechanism, is typically not quantified. Although a single such reduced model may represent chemical activity tolerably well in a limited region of the flow field, it is rarely (acceptably) accurate over the entire composition/temperature space of interest (where it is applied nevertheless). Consequently the predictive value of the reacting-flow simulation is potentially significantly lowered.

The adaptive chemistry approach proposed by Schwer *et al.* [60] addresses this issue by replacing the full mechanism with an entire library of locally-accurate, reduced kinetic models, instead of a single skeletal model. Over large regions of the flow field, the local chemistry is relatively simple. Thus many of these submodels are dramatically smaller and consequently cheaper to evaluate, than the full mechanism. Ideally, each submodel is associated with a range of validity in composition/temperature space, where it is known to approximate the full chemistry to within a quantified tolerance. Since only 'valid' submodels are applied, the accuracy of the simulation is preserved while speeding up the solution process.

The actual computational benefit derived from adaptive chemistry depends on the range of reaction conditions covered by the model library, and the sizes of the individual reduced models. The accuracy of an adaptive chemistry simulation (relative to the full chemistry simulation) depends on the numerical accuracy of the individual reduced models. Although model reduction and validation were done by inspection in [60], a more rigorous, automated approach is clearly desirable. In particular, a model reduction approach suited to implementation in adaptive chemistry should produce reduced models with quantified accuracies and ranges of validity; it should

identify the smallest reduced models that satisfy the user's accuracy/validity criteria; it should be computationally cheap and robust enough to construct model libraries a priori *and* supplement them on-the-fly; and it should be applicable with minimal system-specific customization to different mechanisms and reaction conditions. A number of different approaches to model reduction have been investigated previously in the literature. However, as discussed in Section 1.2, none of these are suitable for constructing model libraries comprising potentially hundreds of reduced kinetic models. Thus the main goal of this thesis was to develop a model reduction methodology which is computationally cheap, accurate, and system-independent enough to be implementable within an adaptive chemistry framework.

Although kinetic model reduction finds its primary application in split-operator simulations of reacting flows, it is also useful as a post-processing tool for automatic mechanism generation. All-inclusive computer-generated reaction mechanisms are becoming increasingly valuable as competitive and regulatory pressures increase. However, these automatically generated kinetic mechanisms are often conservative and contain redundant reactions and species, e.g., a butane pyrolysis mechanism generated by NetGen [68] generated 14 species and 260 reactions. Model reduction may be applied to identify the key species and reactions in such automatically-generated mechanisms.

1.2 Background on Existing Methods for Kinetic Model Reduction

The goal in any model reduction/simplification algorithm is to alleviate the computational cost associated with calculating $\Delta \boldsymbol{x}$ defined by the following ODE:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{\Gamma}(\boldsymbol{x}), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \Delta \boldsymbol{x} &= \int_{t_0}^{t_0+\Delta t} \boldsymbol{\Gamma} dt \end{aligned} \tag{1.1}$$

where $\boldsymbol{x}$ is the state vector and $\boldsymbol{\Gamma}$ is the kinetic source vector. Historically, constant-pressure, adiabatic homogeneous systems have been used to develop and test reduced models using $\boldsymbol{x} = (\boldsymbol{c}, \boldsymbol{T})$, where $\boldsymbol{c}$ is the N_S-dimensional vector of species concentrations and T is temperature. The kinetic ODE for this particular case is:

$$\begin{aligned} \dot{\boldsymbol{c}} &= \boldsymbol{f}(\boldsymbol{c}, T, \boldsymbol{k}), \quad \boldsymbol{c}(0) = \boldsymbol{c}_0 \\ \dot{T} &= f_T(\boldsymbol{c}, T, \boldsymbol{k}), \quad T(0) = T_0 \end{aligned} \tag{1.2}$$

where $\boldsymbol{k}$ is the vector of parameters asscoiated with the N_R reactions. The $N_S + 1$-dimensional system in (1.2) may be simplified by applying one or more of the following, such that a simpler/lower-dimensional closed-form expression for $\boldsymbol{\Gamma}$ is obtained:

- Species elimination, which decreases the dimensionality of the ODE system.
- Reaction elimination, which may decrease the stiffness, density and/or nonlinearity of the system. In some formulations species elimination is essentially an adjunct of reaction elimination, in that species which appear only in deleted reactions are deleted from the reduced model. Both reaction and species elimination result in a subset of the original reaction set which can (in principle) be solved more rapidly.
- Partial equilibrium/Quasi-steady state assumptions, which convert the ODE system into a DAE system by replacing some of the ODEs with algebraic constraints, in an attempt to reduce the overall stiffness of the system.

Many existing methods for mechanism reduction attempt to automate the process of identifying species and/or reactions which can be subjected to the above simplifications, e.g., sensitivity analysis and detailed reduction. As will be shown later, an optimization-based approach has the distinct advantage of performing all 3 reductions simultaneously. Parameterization methods replace the integration procedure with an approximate algebraic evaluation of $\Delta\boldsymbol{x}$ using the generated parameters. Finally, tabulation methods do not attempt reduce (1.2) at all, but instead store the solution so that it can be retrieved rather than calculated when a particular point in composition space is revisited by the reacting-flow calculation.

1.2.1 Sensitivity Analysis

Local Sensitivity Analysis

Although local sensitivities only reflect the effect of a local perturbation in rate parameters, they are widely used to identify 'redundant' reactions in a kinetic mechanism. Local sensitivity coefficients are obtained through a first order Taylor expansion around the initial value of the parameter e.g., a rate constant:

$$c_j(t, \boldsymbol{k} + \Delta\boldsymbol{k}) \equiv c_j(t, \boldsymbol{k}) + \sum_{i=1}^{N_R} \frac{\partial c_j}{\partial k_i}\Delta k_i, \quad s_{j,i} = \frac{\partial c_j}{\partial k_i}. \tag{1.3}$$

When the analytic solution to equation (1.2) is known, the local sensitivity coefficients are easily obtained by partially differentiating $\boldsymbol{c}$ to yield the matrix $\boldsymbol{S}$. Alternatively, an empirical model may also be used to obtain the sensitivities by differentiation. However, in most cases local sensitivities must be calculated numerically.

Finite Difference: The ODE system can be solved $N_R + 1$ times to obtain a finite difference approximation to the matrix $\boldsymbol{S}$:

$$\frac{\partial c_j(t)}{\partial k_i} = \frac{c_j(t, k_i + \Delta k_i) - c_j(t, k_i)}{\Delta k_i}, \quad i = 1, \ldots, N_R. \tag{1.4}$$

However, the finite difference approximation is impractical because it is computationally expensive compared to other methods and very sensitive to the Δk_i selected. Large Δk_i can result in high second-order approximation errors, whereas low Δk_i is susceptible to round-off error.

Direct Solution: To obtain a more accurate result, the sensitivities can also be obtained by solving an ODE system either simultaneously or sequentially with (1.2). Differentiating equation (1.2) with respect to k_i yields:

$$\frac{d}{dt}\frac{\partial \boldsymbol{c}}{\partial k_i} = \boldsymbol{J}(t)\frac{\partial \boldsymbol{c}}{\partial k_i} + \frac{\partial \boldsymbol{f}(t)}{\partial k_i}, \quad i = 1, \ldots, N_R. \tag{1.5}$$

Systems (1.2) and (1.5) are thus coupled through the rate sensitivity matrix $\boldsymbol{F} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{k}}$ and the Jacobian matrix $\boldsymbol{J} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{c}}$. They can be solved using a staggered direct method

[12], a simultaneous corrector method [51] or a staggered corrector [22] method. In the staggered direct method the discretised nonlinear DAE system is solved using a k-step general backward difference formula (BDF) :

$$\boldsymbol{g}(\boldsymbol{c}_{n+1}) = \boldsymbol{f}\left(t_{n+1}, \boldsymbol{c}_{n+1}, \sum_{l=0}^{k} \boldsymbol{\alpha}_l \boldsymbol{c}_{n+1-l}, \boldsymbol{k}\right) = \boldsymbol{0} \tag{1.6}$$

where $\boldsymbol{\alpha}_l$ contains the coefficients of the BDF formula. Equation (1.6) is solved for $\boldsymbol{c}_{n+1}$ using an iterative Newton-type process. The linear system below is solved at each iteration, m, using a suitable approximation to the Jacobian of $\boldsymbol{g}$.

$$\begin{aligned} \boldsymbol{G}\left(c_{n+1}^{m} - c_{n+1}^{m+1}\right) &= \boldsymbol{g}(\boldsymbol{c}_{n+1}^{m}) \\ \boldsymbol{G} &\approx \alpha_0 \frac{\partial \boldsymbol{f}}{\partial \dot{\boldsymbol{c}}_{n+1}^{m}} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{c}_{n+1}^{m}}. \end{aligned} \tag{1.7}$$

The solution to the sensitivity system is then obtained by solving (1.4) using BDF discretisation:

$$\begin{aligned} \boldsymbol{A}\begin{bmatrix} \dot{\boldsymbol{s}}_i \\ \boldsymbol{s}_i \end{bmatrix} &= -\frac{\partial \boldsymbol{f}}{\partial k_i} \\ \boldsymbol{A} &= \begin{bmatrix} \frac{\partial \boldsymbol{f}}{\partial \dot{\boldsymbol{c}}_{n+1}} & \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{c}_{n+1}} \end{bmatrix}. \end{aligned} \tag{1.8}$$

Although $\boldsymbol{A}$ is based on the same information as $\boldsymbol{G}$, it must be updated and factored at every step because $\boldsymbol{G}$ is updated only occasionally. The staggered corrector method improves computational efficiency by using a quasi-Newton iteration to solve the sensitivity system:

$$\boldsymbol{G}\left(\boldsymbol{s}_{i_{n+1}}^{m} - \boldsymbol{s}_{i_{n+1}}^{m+1}\right) = \boldsymbol{A}\begin{bmatrix} \boldsymbol{\alpha}_0 \boldsymbol{s}_{i_{n+1}}^{m} + \sum_{l=1}^{k} \alpha_l \boldsymbol{s}_{i_{n+1-l}} \\ \boldsymbol{s}_{i_{n+1}}^{m} \end{bmatrix} + \frac{\partial \boldsymbol{f}}{\partial k_i}. \tag{1.9}$$

In this case matrix $\boldsymbol{A}$ and the sensitivity residual vector need to be updated only once per step of the integrator and additional matrix factorisations are not required since $\boldsymbol{G}$ is already available as LU factors from the DAE corrector iteration. The simultaneous corrector method solves systems (1.2) and (1.5) simultaneously using (1.6-1.8) allowing the factored corrector matrix to be reused for multiple steps. However, matrix $\boldsymbol{A}$ must still be updated at every corrector iteration. This can be a

significant cost in large model reduction problems.

Green's Function: Since (1.5) is a linear, time-dependent, inhomogeneous equation, it can be solved by combining the particular solutions for each parameter with the homogenous solution [37]. The particular solution in this case is obtained as follows:

$$\frac{\partial \boldsymbol{c}(t_2)}{\partial k_i} = \int_{t_1}^{t_2} \boldsymbol{K}(t_2, s) \frac{\partial \boldsymbol{f}}{\partial k_i} ds \tag{1.10}$$

The kernel, or Green function matrix, $\boldsymbol{K}$, required to integrate (1.10) can be obtained by solving for the adjoint matrix, $\boldsymbol{K}^T$:

$$\frac{d}{dt}\boldsymbol{K}^T(t) = -\boldsymbol{K}^T(t)\boldsymbol{J}(t), \quad t \leq t_2, \quad \boldsymbol{K}^T(t_2, t_2) = \boldsymbol{I} \tag{1.11}$$

Polynomial Approximiation: In analyses comprising a comparatively large number of parameters but small number of species, it may be efficient to approximate sensitivity coefficients using Lagrange interpolation polynomials of degree L [36] such that:

$$\frac{\partial \boldsymbol{c}(t)}{\partial k_i} = \sum_{s=0}^{L} l_s(t) \frac{\partial \boldsymbol{c}}{\partial k_i}(t_s), \quad i = 1, 2, \ldots N_R. \tag{1.12}$$

The value of $\frac{\partial \boldsymbol{C}}{\partial k_i}$ at t_0 is known. The value of each coefficient $\frac{\partial \boldsymbol{C}}{\partial k_i}(t_s)$ is then obtained by requiring (1.12) to satisfy (1.5) at each point $t_s, s = 1, 2, \ldots L$.

Many of the local sensitivity methods described above, e.g., the finite difference and polynomial approximations, are calculated at specific time points. This can be misleading when trying to choose a reduced model since a particular sensitivity may be low at one point, but high at another point in time that is ignored by the sampling method. Solving the sensitivity system numerically resolves this problem by providing detailed time-dependent information.

Local sensitivity analysis is the most commonly used sensitivity method and is often used to justify mechanism reduction. However, local sensitivity refers to the effect of an infinitesimal parameter change and does not carry information about the consequences of setting the value of a parameter to 0 (which is the case in model reduction). A typical misinterpretation of local sensitivity coefficients is illustrated

below:

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C, \quad k_2 \gg k_1. \tag{1.13}$$

In this case $\frac{\partial c_A}{\partial k_2} = 0$, and $\frac{dc_C}{dt} \approx k_1 c_A$. Thus $\frac{\partial c_C}{\partial k_2}$ is also small. This suggests that the second reaction can be eliminated, which is not the case. Stochastic global sensitivity methods (e.g., FAST and WASP) have been developed to account for simultaneous, order-of-magnitude changes.

Global Sensitivity

Global sensitivity methods consider the reaction vector to be a vector of random variables with probability density function $p(\boldsymbol{k})$. The mean and variance in species concentrations are calculated using a physically reasonable distribution for the parameters and initial conditions. Fourier Amplitude Sensitivity Test (FAST) [15] is the most widely used global sensitivity method. Here the parameters k_i, are represented as periodic functions of the search variable, s:

$$k_i = G_i \sin(\omega_i s), \quad i = 1, 2, \ldots, N_R \tag{1.14}$$

such that ω_i is a frequency which belongs to the i^{th} parameter, and G_i is completely determined by the probability density function $p(k_i)$. If the frequencies are incommensurate, the curve defined by (1.14) will project s into all possible values of k_i. For computational reasons it is practical to use appropriate integer frequencies instead; the concentrations will then be (2π) periodic functions of s at time t and they can be Fourier analysed. The mean and variance of the instantaneous concentration are then:

$$\begin{aligned} \langle c_j(t) \rangle &= \int c_j(t, \boldsymbol{k}) p(\boldsymbol{k}) d\boldsymbol{k} \\ \sigma_j^2(t) &= 2 \sum_{l=1}^{+\infty} \left(A_{jl}^2(t) + B_{jl}^2(t) \right) \end{aligned} \tag{1.15}$$

where the Fourier coefficients $A_{jl}(t)$ and $B_{jl}(t)$ are given by:

$$\begin{aligned} A_{j,l}(t) &= \tfrac{1}{2\pi} \textstyle\int_{-\pi}^{\pi} c_j(t, s) \cos(ls)\, ds, \quad l = 0, 1, \ldots, \infty \\ B_{j,l}(t) &= \tfrac{1}{2\pi} \textstyle\int_{-\pi}^{\pi} c_j(t, s) \sin(ls)\, ds, \quad l = 0, 1, \ldots, \infty \end{aligned} \tag{1.16}$$

If the Fourier coefficients are evaluated with the fundamental frequencies of transformation (1.14) or with its harmonics ($l = r\omega_i,\ r = 1, 2, \ldots$) then the obtained variances:

$$\sigma_{ji}^2 = 2\sum_{r=1}^{+\infty}\left(A_{j,r\omega_i}^2(t) + B_{j,r\omega_i}^2(t)\right) \tag{1.17}$$

are part of the total variance $\sigma_j^2(t)$ and correspond to the variance of c_j arising from uncertainty in the i^{th} parameter. The ratio $S_{ji}(t) = \frac{\sigma_{ji}^2(t)}{\sigma_j^2(t)}$, called partial variance, is the basic measure of sensitivity in the FAST method.

The WASP method (Walsh Amplitude Sensitivity Procedure) [52] is similar to the FAST method except that 2-valued Walsh functions are used as the basis for decomposition. The effect of reaction reduction can be investigated by setting the lower value of the parameter to 0 (while using the nominal value of the parameter as the upper value). The major limitation of global sensitivity approaches is their high computational cost. The investigation of a 50-parameter model using the WASP method would require $2^{50} \approx 10^{15}$ runs.

1.2.2 Principal Component Analysis

The effect of a parameter change on the species concentrations, at a finite number of time points, may be represented as the following sum of deviations:

$$Q(\boldsymbol{\alpha}) = \sum_{n=1}^{L}\sum_{j=1}^{N_S}\left[\frac{c_{j,n}(\boldsymbol{\alpha}) - c_{j,n}(\boldsymbol{\alpha}^0)}{c_{j,n}(\boldsymbol{\alpha}^0)}\right] \tag{1.18}$$

where $c_{j,n}$ is the concentration of species j at time n, L is the total number of time points considered, and $\alpha_i = \ln k_i$ such that $\Delta\alpha_i = \alpha_i - \alpha_i^0 = \frac{\ln k_i}{k_i^0}$ for some nominal value k_i^0. A second order Taylor expansion of Q around the minimum $Q(\boldsymbol{\alpha}^0)$ yields the approximation:

$$Q(\boldsymbol{\alpha}) \approx (\Delta\boldsymbol{\alpha})^T \tilde{\boldsymbol{S}}^T \tilde{\boldsymbol{S}}(\Delta\boldsymbol{\alpha}), \tag{1.19}$$

where $\tilde{\boldsymbol{S}}$ is a matrix composed of the L sensitivity matrices $\boldsymbol{S}_n = \frac{\partial \ln \boldsymbol{c}}{\partial \ln \boldsymbol{k}}(t_n)$:

$$\tilde{\boldsymbol{S}} = \begin{bmatrix} \boldsymbol{S}_1 \\ \boldsymbol{S}_2 \\ \vdots \\ \boldsymbol{S}_L \end{bmatrix}. \tag{1.20}$$

In principal component analyses of concentration sensitivities (PCAS) [76], an eigenvalue-eigenvector decomposition of the symmetric matrix $\boldsymbol{S}^T\boldsymbol{S}$ is peformed such that:

$$\boldsymbol{S}^T\boldsymbol{S} = \boldsymbol{\zeta}\boldsymbol{\Lambda}\boldsymbol{\zeta}^T, \tag{1.21}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix formed by the eigenvalues, $\lambda_i, i = 1, \dots N_R$, of $\boldsymbol{S}^T\boldsymbol{S}$, and $\boldsymbol{\zeta}$ denotes the matrix of normed eigenvectors $\boldsymbol{\zeta}_i, i = 1, \dots N_R$. A reduced model is identified by retaining reactions which correspond to elements $\zeta_{j,i}$ larger than some threshold value ζ_S, associated with eigenvalues larger than some threshold λ_S, i.e., by retaining all reactions i for which there exists a j such that $\zeta_{j,i} \geq \zeta_S$ and $\lambda_j \geq \lambda_S$.

The PCAF [75] is similar to PCAS except that the eigenvalue-eigenvector analysis is now performed on the matrix product $\tilde{\boldsymbol{F}}^T\tilde{\boldsymbol{F}}$, where

$$\begin{aligned} \tilde{\boldsymbol{F}} &= \begin{bmatrix} \boldsymbol{F}_1 \\ \boldsymbol{F}_2 \\ \vdots \\ \boldsymbol{F}_L \end{bmatrix} \\ \boldsymbol{F}^T\boldsymbol{F} &= \boldsymbol{\xi}\boldsymbol{\Omega}\boldsymbol{\xi}^T. \end{aligned} \tag{1.22}$$

where $\boldsymbol{\Omega}$ is a diagonal matrix formed by the eigenvalues, $\gamma_i, i = 1, \dots N_R$, of $\boldsymbol{F}^T\boldsymbol{F}$, $\boldsymbol{\xi}$ denotes the matrix of normed eigenvectors $\boldsymbol{\zeta}_i, i = 1, \dots N_R$, and

$$\boldsymbol{F}_{n,j,i} = \frac{k_i}{f_j}\frac{\partial f_j}{\partial \boldsymbol{k_i}}(t_n) = \frac{k_i}{\dot{c}_j}\frac{\partial \dot{c}_j}{\partial k_i} = \frac{\nu_{j,i} r_i}{\dot{c}_j},\ j = 1 \dots N_S, \quad i = 1 \dots N_R$$

Key reactions are identified using threshold values γ_F and ξ_F for significant eigenvalues and normalized eigenvector elements respectively.

The PCAS and PCAF methods are useful for identifying not only which reactions

are dominant at particular points along the reaction trajectory, but also how these reactions are coupled together in N_R eigenvectors. However, they are susceptible to the same vulnerabilities as local sensitivity methods which 'sample' at discrete points along the reaction trajectory. Furthermore, the size of the reduced model identified by either method depends strongly on the threshold values selected by the user. Since these quantities are difficult to interpret physically, it is impossible for the user to select appropriate values for different mechanisms and reaction conditions without significant system-specific understanding.

1.2.3 Detailed Reduction

The detailed reduction method (DR) [24] eliminates reactions corresponding to low reaction rates and low enthalpy-weighted reaction rates. In other words, DR considers reactions which satisfy the following criteria to be negligible:

$$\begin{aligned} r_i &< \kappa \, |r_{lim}| \\ q_i &< \kappa \, \max_l q_l \\ q_l &= |r_l \, \Delta H_l| \end{aligned} \tag{1.23}$$

where r_{lim} is the rate of the rate-limiting reaction, ΔH_l is the enthalpy change of reaction l, and κ is some user-specified threshold tolerance. The DR method has the advantages of being extremely simple to implement, and requiring only relative tolerances to be set by the user. However, it is easy to find examples of situation where rate-based elimination is likely to preclude important reactions, e.g., Reaction 1 in (1.13).

1.2.4 Time Scale Based Reaction Reduction

The intrinsic low-dimensional manifolds (ILDM) [45] and the computational singular perturbation (CSP) approach [41] both perform time-scale based simplifications based on eigenvalue-eigenvector analyses of the Jacobian matrix $\boldsymbol{J} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{c}}$. In either

case, the goal is to eliminate eigenvectors associated with negative eigenvalues, which indicate that the system is insensitive to perturbations along a certain direction in the composition space. Neither method generates reduced models which are subsets of the original reaction and species sets. CSP yields pseudo-reaction rate vectors which may be used to construct closed-form approximations for $\boldsymbol{f}$, but are difficult to interpret phyiscally. ILDM does not yield any closed-form analytic expressions; it provides approximate solutions to (1.2) which are either generated a priori, for the entire composition space of interest, or on-the-fly, as a reacting-flow simulation visits new regions of composition space. In addition, both approaches suffer from the major limitation of the PCAS and PCAF methods; it is difficult for a user to select appropriate threshold values without extensive system-specific knowledge.

1.2.5 Parameterization

Evaluation of algebraic equations is much less computationally demanding than solution of an ODE. Parameterization techniques seek to exploit this advantage by fitting explicit functions to the numerically integrated solution of (1.2). Using orthonormal polynomial functions as the basis set further allows the coefficients of each polynomial to be determined independently of other coefficients. In [74] the approximating function is taken to be a high-order polynomial having only a few effective coefficients determined by a least-squares approximation from a highly overdetermined set of data. The data set is obtained by integrating the system for a fixed time interval at thousands of different initial conditions. The set of input functions were taken to be the mononomials of a general N_S-variable polynomial, i.e., products of a number of different species concentrations considered to be important in the reduced mechanism. From these, orthonormal polynomials were determined using the Gram-Schmidt process. Finally, the generated polynomial was evaluated using Horner equations.

The obvious advantages of polynomial fitting are that it is easy to implement and results in significant speedup since differential equations are replaced by explicit functions. However, polynomial fitting does not generate subsets of the original reaction and species sets which can be used be used to derive closed-form approximations

for the kinetic source term. Furthermore, when concentration variables are supplemented by numerous other physical quantities e.g., velocity, density, etc. it is often difficult to identify the key input variables to be used as the monomials that are used to construct the basis polynomial.

1.2.6 Adaptive Tabulation

Adaptive tabulation allows mechanisms of high dimensionality to be stored by parameterising only the accessed regions of chemical composition space. Two main approaches have been described in the literature: that of Tonse *et al.* [71] in which second order polynomials are fitted to the ODE solution, and that of Maas and Pope [45], in which a first order Taylor expansion about a point solution is used to approximate the solution of the full ODE.

Piecewise Reusable Implementation of Solution Mapping

In Piecewise Reusable Implementation of Solution Mapping (PRISM) [71] the chemical composition space is partitioned and indexed in $N_S + 2$ dimensional space (N_S species + time + temperature). Polynomial approximations to the ODE solution are derived when the trajectory enters a particular hypercube for the first time. Surface response theory is used to determine the number and location of hypercube points which are used to solve for polynomial coefficients which are stored along with the hypercube coordinates in a memory-resident list (or in a combination of memory-resident list and a disk file). When the hypercube is revisited during the course of the trajectory, its keys are located in the binary search tree and the associated polynomial coefficients are then retrieved. Mass and enthalpy conservation are treated by post-processing the mapped solution: $N_S - N_E - 1$ species concentrations are held constant, where N_E is the number of elements in the system. The remaining $N_E + 1$ concentrations are allowed to vary and solved for using conservation equations.

In-situ Adaptive Tabulation

The In-situ Adaptive Tabulation (ISAT) algorithm differs from PRISM in that the ODE solution is approximated by a Taylor expansion around a reference point, and the partitioning of the chemical space is not preset. Each table entry consists of a reference composition vector, the reaction mapping evaluated at the reference vector, the Jacobian of the reaction matrix evaluated at the reference composition, and the ellipsoid of accuracy (EOA) which specifies the range of applicability of the Taylor approximation around the given reference vector (depending on the user-supplied tolerance). A conservative estimate of the EOA is obtained from the singular values of the Jacobian at the reference point. When a query point is found to lie outside the EOA of the reference point, direct integration is used to compare the solution of the reference vector and the query point. If the deviation is found to be below the user provided tolerance, the EOA of the reference point is expanded to include the query point. Otherwise a new reference point in the accessed region is tabulated. The user-specified tolerance represents a compromise between the number of tabulation points, the accuracy of the method and the computational speedup achieved. It must be set empirically based on the user's needs by observing the global error accumulated for each species over the entire trajectory. In the early stages of a simulation most of the points visited must be tabulated, in the middle stages, the EOAs of revisited points undergo growth, and in the final stage most of the accessed points can simply be looked up. It is here that the computational savings over direct integration become most apparent.

The tabulation methods described here essentially improve on simple parameterization by storing the polynomial coefficients (so that they do no have to be regenerated), and extending the procedure to a larger region of composition space so that different polynomial coefficients are valid in different parts of the trajectory. However, such an approach quickly becomes memory-intensive for mechanisms of even moderate dimensionality. Furthermore, the problem of creating a physically interpretable reduced model remains.

1.2.7 Optimization-based Model Reduction

As discussed above, conventional model reduction techniques suffer from one or more the following limitations: they do not allow for simultaneous reaction and species elimination; they require excessive system-specific knowledge to be implemented intelligently; they do not provide the user control over either the size or the accuracy of the resulting model; rather reduction, simulation and error evaluation must be performed iteratively until an acceptable tradeoff between model size and accuracy is reached; they do not allow the user flexibility in defining a quantifiable metric of agreement between the full and reduced mechanisms; and they do not allow for parametric uncertainty in the reaction rate parameters. Many, if not all, of these concerns may be addressed using an optimization-based approach to model reduction. Naturally, the complexity of solving the optimization problem increases with the sophistication of the reduction formulation.

The work in this thesis is focussed on the most mathematically tractable reduction formulation, reaction elimination. Despite its simplicity, reaction elimination provides a roughly linear decrease in integration CPU time with model size. The problem of selecting the reactions to be included in a reduced kinetic model has been studied previously using both deterministic and stochastic optimization techniques [4, 79, 19, 20]. In an integer programming framework, the number of reactions in the reduced model is typically minimized subject to some constraint (G) that aims to ensure satisfactory agreement between the full and reduced mechanisms. This is represented as:

$$\begin{aligned} \min & \sum_{i=1}^{N_R} z_i \\ G[\boldsymbol{x}(t,\boldsymbol{z}), \boldsymbol{x}_{ref}(t)] & \leq 0 \end{aligned} \tag{1.24}$$

where $z_i \in \{0,1\}$ and $\boldsymbol{x}_{ref}(t)$ and $\boldsymbol{x}(t,\boldsymbol{z})$ are given by the solutions of the following

differential equations:

$$\left.\begin{aligned} \dot{x}_{j,ref} &= \Gamma_{j,ref}(\boldsymbol{x}_{ref}), \quad x_{j,ref}(t_0) = x_{j,ref,t_0} \\ \dot{x}_j &= \Gamma_j(\boldsymbol{x}, \boldsymbol{z}), \quad x_j(t_0) = x_{j,ref,t_0} \end{aligned}\right\} j = 1, \ldots, N_S + 1 \tag{1.25}$$

where

$\boldsymbol{x}_{ref}$ is the vector of state variables calculated using the full mechanism.

$\boldsymbol{x}$ is the vector of state variables calculated using the reduced model.

$\boldsymbol{z}$ is a binary vector of dimension N_R, such that $z_i = 1$ indicates that reaction i is included in the reduced model and $z_i = 0$ indicates that reaction i is deleted.

$\boldsymbol{\Gamma}$ is the vector of functions describing the system using the reduced model.

$\boldsymbol{\Gamma}_{ref}$ is the vector of functions describing the system using the full mechanism.

G is a user-defined functional measuring the model truncation error.

The binary vector $\boldsymbol{z}$ basically turns reactions 'on' and 'off' in the reduced model. Thus in the equations for the reduced model, the source term $\boldsymbol{\Gamma}$ is modified by $\boldsymbol{z}$ to exclude reactions that are deleted from the reduced model. The full mechanism is recovered by setting $z_i = 1,\ i = 1, \ldots, N_R$. The integer program (IP) in Eq. (1.24) is solved to determine the optimal binary vector $\boldsymbol{z}$. However, the particular form of G and $\boldsymbol{\Gamma}$ determine whether or not it is possible to solve Eq. (1.24) to global optimality using existing algorithms and hardware.

Integral error constraints of (almost) any form, combined with the nonlinear rate expressions result in nonconvex, dynamic-embedded integer programs. Although branch-and-bound [4] and constrained NLP methods [79] have been used successfully to identify reduced models by solving Eq. (1.24), such problems cannot be solved to guaranteed global optimality using existing algorithms (other than by enumerating all 2^{N_R} possible values for $\boldsymbol{z}$, which is usually impractical). In other words, a reaction set smaller than the obtained solution, $\sum z_i$, may also satisfy the constraints G. No bound is available on the possible discrepancy in size between the reaction

set found and the true optimal reaction set, and in certain cases these algorithms may fail to locate a feasible solution altogether (although at least one solution always exists). The problem in Eq. (1.24) has also been attacked stochastically using genetic algorithms [19]. However, stochastic methods also fail to provide any guarantee of optimality, and in particular genetic algorithms were found to be extremely computationally intensive for this problem. Alternative formulations of this problem have also been proposed in the literature. For example, in one case the error in the state variables was minimized subject to a constraint on the maximum number of reactions in the reduced model [79]. In another, the number of reactions in the reduced model and the weighted sum of species errors were both included in the objective function [19]. However, none of these formulations have the convexity property required to guarantee global optimality in existing deterministic algorithms.

Another disadvantage of existing integer programming formulations is the range of validity associated with the derived models. The formulation in ([79, 4]) follows the precedent set by conventional reduction methods by aiming to reproduce the constant-pressure batch reactor trajectory predicted by the full mechanism, for a fixed set of initial conditions, and a known time horizon. From the standpoint of an adaptive chemistry simulation, it is not necessarily helpful to generate reduced models which are valid for a fixed time horizon and set of initial conditions. Although the individual cells in a split operator simulation are typically modeled as constant-pressure batch reactors, it is unlikely that the same combination of $\boldsymbol{x}_0,\ \Delta t$ encountered in one cell, will be revisited later in another cell. Thus, the computational cost of generating a reduced model in the first place is not justified. Rather, it is more useful to identify reduced models which are associated with upper ($\boldsymbol{x}^u$) and lower ($\boldsymbol{x}^l$) limits of validity in composition/temperature space, such that once such a reduced model is generated, it can be reused at any cell which lies in the range $[\boldsymbol{x}^l,\ \boldsymbol{x}^u]$.

Such a reduced model may identified using one of two approaches: The reduced model may be generated subject to a postulated range of validity defined by an interval $[\boldsymbol{x}^l,\ \boldsymbol{x}^u]$. As shown in Chapter 3, this formulation produces an integer semi-infinite program, which cannot be solved to global optimality using known optimization algo-

rithms, and is not pursued here. Alternatively, the reduced model may be generated subject to point constraints such that the smallest possible model is derived. This latter problem is a finitely-constrained, algebraic integer program, which is easily solvable to global optimality, e.g., using a branch-and-cut algorithm. A range of validity is then identified a posteriori, using interval analysis methods. As shown in Chapter 3, a generalized semi-infinite program may be solved to identify an optimal (maximum) range of validity.

The next chapter describes a new, point-constrained integer programming formulation for reaction elimination. The linearity of the revised formulation allows it to be solved cheaply to global optimality using well-studied integer programming algorithms. The computational cost of generating and evaluating the reduced models, and the predictive value of the reduced reaction sets, are studied using two large-scale combustion mechanisms. In Chapter 3, extensions to species elimination and to reduction under parametric uncertainty are considered, and the use of interval analysis in calculating rigorous ranges of validity for point-constrained models is discussed; a generalized semi-infinite programming (GSIP) formulation for calculating optimal ranges of validity is also presented. Numerical methods for the global optimization of even standard semi-infinite programs (SIPs) were not previously known. A branch-and-bound framework for solving SIPs to global optimality is developed in Chapters 4 and 5 with a view to addressing this problem. Finally, the range problem and other avenues for future work in SIPs are considered in Chapter 6.

Chapter 2

Point-constrained Reaction Elimination

As discussed in the previous chapter, an optimization-based approach is particularly well suited to identifying reduced models for adaptive chemistry applications. However, the dynamic-embedded integer progams studied previously in the literature are too computationally intensive to be useful for generating model libraries from large-scale kinetic mechanisms. In this chapter a revised integer programming formulation for reaction elimination is proposed. The new linear formulation is solved to generate model libraries for methane and n-heptane combustion, and is also compared to a number of more conventional reaction elimination methods.

2.1 Formulation

The evolution of the state trajectory $\boldsymbol{x} = (T, \boldsymbol{y})$ for a constant-pressure, adiabatic batch reactor may be described by the following system of equations:

$$\begin{aligned} \Gamma_1 &= \frac{\sum_{j=1}^{N_S} h_j(T) M_j \sum_{i=1}^{N_R} \nu_{j,i} z_i r_i(\boldsymbol{y},T)}{\rho(\boldsymbol{y},T) C_p(\boldsymbol{y},T)} \\ \Gamma_{j+1} &= \frac{M_j \sum_{i=1}^{N_R} \nu_{j,i} z_i r_i(\boldsymbol{y},T)}{\rho(\boldsymbol{y},T)}, \qquad j = 1 \ldots N_S. \end{aligned} \tag{2.1}$$

Reduced models for such systems have been generated by solving the conventional integer programming formulation in Eq. (1.24). Agreement between the full and reduced chemistry profiles was enforced using the following integral (isoperimetric) constraint:

$$G(\boldsymbol{z}) = \sum_{j=1}^{N_S+1} \alpha_j \left(\int_0^{t_f} \frac{x_j(t,\boldsymbol{z}) - x_{j,ref}(t)}{x_{j,ref}(t)} dt \right)^2 - \delta_1 \leq 0 \tag{2.2}$$

where

α_j is a weighting factor for the state variable j

δ_1 is the error tolerance.

However, this constraint formulation results in a nonconvex, dynamic-embedded problem which is subject to the difficulties described in the previous section. In this chapter, a linear reformulation which allows the reaction-elimination problem to be solved to guaranteed global optimality, is introduced. This linearity is achieved by eliminating the embedded dynamic system (ODE) and nonlinearities in the general formulation of Eq. (1.24). Agreement is enforced between the (instantaneous) fluxes rather than the species mass fraction and temperature profiles. Mathematically, the isoperimetric constraint in Eq. (2.2) is replaced by the following set of algebraic point constraints:

$$G_{j,l} = \left|\Gamma_j\left(\boldsymbol{x}_{ref}(t_l), \boldsymbol{z}\right) - \Gamma_{ref,j}\left(\boldsymbol{x}_{ref}(t_l)\right)\right| - \left(atol_j + rtol_j \left|\Gamma_{ref,j}\right|\right) \leq 0,$$
$$j = 1 \ldots N_S + 1, \quad l = 1 \ldots N_t \tag{2.3}$$

where N_t is the number of points at which flux constraints are applied. The form of Eq. (2.3) is similar to the error constraint enforced in ODE solvers such as VODE [11]. For closest agreement with integral-based model reductions, the constrained points can be selected to lie along the trajectory of the full chemistry solution. The full chemistry profile is computed beforehand by integrating the original mechanism and saving the solution trajectory $(\boldsymbol{y}_{ref}(t),\, T_{ref}(t)),\ t_0 \leq t \leq t_f$ to file.

The new error constraints in Eq. (2.3) can be reformulated so that they are linear in the flux terms Γ. Since these flux terms (2.1) and the original objective function

are also linear in the decision variables z_i, the revised formulation yields a linear (algebraic) integer program:

$$\begin{aligned} & \min \textstyle\sum_{i=1}^{N_R} z_i \\ \Gamma_{j,l,ref} - \Gamma_{j,l} \quad & \leq atol_j + rtol_j \left|\Gamma_{j,l,ref}\right|, \qquad j = 1, \ldots, (N_S + 1), \quad l = 1, \ldots, N_t \\ \Gamma_{j,l} - \Gamma_{j,l,ref} \quad & \leq atol_j + rtol_j \left|\Gamma_{j,l,ref}\right|, \qquad j = 1, \ldots, (N_S + 1), \quad l = 1, \ldots, N_t \end{aligned} \tag{2.4}$$

where $\Gamma_{j,l} = \Gamma_j(\boldsymbol{x}_{ref}(t_l), \boldsymbol{z})$ and $\Gamma_{j,l,ref} = \Gamma_j(\boldsymbol{x}_{ref}(t_l))$, respectively. This linear IP is easily solved to global optimality using existing IP technology. This means that no reaction subset smaller than the solution $\sum_{i=1}^{N_R} z_i$ will satisfy the error constraints specified by the modeler in Eq. (2.3), i.e., the solution $\boldsymbol{z}$ defines an optimally-reduced kinetic model.

The revised reaction-elimination formulation is better suited to the construction of reduced model libraries than the conventional optimization formulations. The linearity of the integer program enables it to be solved to global optimality thus guaranteeing that the smallest possible valid reduced model is correctly identified. Furthermore, the simplicity of the problem formulation allows it to be solved cheaply and robustly for large-scale mechanisms ($N_R,\ N_S N_t \sim 0(10^3)$) which are generally beyond the scope of the state-of-the-art MINLP technology required to solve dynamic-embedded, nonconvex integer programs for even a feasible (and possibly suboptimal) solution.

However, an unresolved issue is the range of validity of the derived reduced model. Clearly, a reduced model generated by solving (2.3) is strictly valid only at the points $\boldsymbol{x}_{ref}(t_l), \quad l = 1, \ldots, N_t$ at which constraints were applied during reduction. However, to be useful in the context of adaptive chemistry, reduced models should be associated with continuous ranges of validity in composition space. One possible heuristic, previously used in [4], is to assume that the derived model is valid everywhere within the convex hull of the points $\boldsymbol{x}_{ref}(t_l), \quad l = 1, \ldots, N_t$. This is the approach taken in the numerical studies in Section 2.3.3. A more rigorous approach to identifying

ranges of validity is discussed in Chapter 3.

2.2 Computational Limitations

A key concern in developing model reduction methods is the efficiency and robustness with which large-scale mechanisms can be reduced. Indeed, it is these very mechanisms which undergo the most dramatic reduction and consequent increases in computational speed. The problem formulation in Eq. (2.3) allows one to solve for a reduced model using any one of a number of robust, commerical IP solvers, e.g., CPLEX [38], which implements a branch-and-cut strategy. The reader is referred to the optimization literature [23, 6, 5] for details on the solution algorithm. In this work the linear formulation was used successfully to reduce mechanisms as large as $N_{R,full} = 2446, N_S = 544$. The size of the mechanism which can be reduced is limited only by the capabilities of the IP solver. In principle, the speed and robustness of the IP solver may be significantly improved by customizing the branch-and-cut parameters for the reaction-elimination problem. However, all of the reduced models presented here were generated using default CPLEX parameters on a 733 MHz, 128MB Pentium-III PC running Linux.

2.3 Model Libraries for Adiabatic, Isobaric Homogeneous Reactors

2.3.1 Implementation

Before addressing the complicated task of constructing reduced model libraries for adaptive chemistry reacting-flow simulations, reaction elimination was applied to construct reduced models for constant-pressure, adiabatic batch reactors. In this section, the accuracy and computational speedup afforded by the reduced models is studied using batch reactor simulations of methane and n-heptane combustion.

Chemkin mechanism and thermodynamic data files for the original, detailed chem-

istry [64], [16] were first preprocessed using the Chemkin II interpreter [39]. A variant of the constant pressure batch reactor (CONP) module was used to set up the system in Eq. (2.1). The full mechanism was then numerically integrated using VODE [11], until combustion was essentially complete. Species mass fraction and temperature profiles, $\boldsymbol{x}_{ref}(t)$, were saved to file at each time step since this information is required to solve for the reduced model. The time points (t_l) at which each reduced model was required to hold were specified through a data file. Each of the reduced models shown in Fig. 2-1 was obtained by applying constraints at two preselected points, i.e., $N_t = 2$ in Eq. (2.4). The first reduced model shown in Fig. 2-1 was constructed using $t_1 = 0.05\ ms$ and $t_2 = 0.2\ ms$. In specifying the constraints in Eq. (2.3) a uniform relative error tolerance was applied for all species and temperature. For temperature, the absolute tolerance ($atol_T$) was set to $10^{-6}\ Ks^{-1}$. For all stable species the absolute tolerance ($atol_j$) was set to $atol_{ST}$, such that:

$$atol_{ST} = \max(10^{-6}\ s^{-1},\ \ \epsilon \max_l |\dot{y}_{ref,l}|) \tag{2.5}$$

and $\epsilon \ll 1$. A minimum $atol_j$ value of $10^{-6}\ s^{-1}$ was enforced to satisfy the numerical precision of the IP solver. The inclusion of the absolute tolerance in the species constraints allows effective model reduction by requiring only the major fluxes to be reproduced correctly (i.e., to within $rtol_j\,|\Gamma_{j,ref}|$) at each point, t_i. At the user's discretion, a constant value, e.g., $atol_{ST} = 10^{-5}s^{-1}$, may be used in place of Eq. (2.5). This is generally not advisable since the maximum species flux typically changes by several orders of magnitude over the course of a reaction. In such cases, imposing a constant $atol_{ST}$ yields inaccurate models at points where reaction fluxes are much lower than $atol_{ST}$, and unnecessarily large models wherever reaction fluxes are high.

Radicals play an important role in kinetic evolution, but occur only at low fluxes in the early stages of a combustion reaction. To ensure that these low but critical fluxes are predicted accurately by the reduced model, radicals are subjected to a smaller absolute tolerance ($atol_{RD}$) than stable species. The following approximation

to the free radical chain length is used to calculate $atol_{RD}$:

$$\frac{\text{rate of conversion}}{\text{net rate of free radical formation}} \approx \frac{\left|\dot{C}_{fuel}\right|}{r_{OH,ref}} = FRC \tag{2.6}$$

where $\dot{C}_{fuel}$ is the molar rate of consumption of fuel, and $r_{OH,ref}$ is rate of formation of hydroxyl in the rate-limiting initiation step. The absolute tolerance for radicals is then taken to be:

$$atol_{RD} = \max\left(10^{-6}\ \ s^{-1}, \quad \frac{atol_{ST}}{Ch}\right) \tag{2.7}$$

where $Ch = \max(1, FRC)$. In the early stages of combustion, the fuel is not yet completely reacted, hydroxyl formation is a good surrogate for free radical initiation, and Eq. (2.7) yields a good approximation to the true free radical chain length. Consequently $FRC \gg 1$ resulting in $atol_{RD} \ll atol_{ST}$ as desired. As the reaction progresses the fuel becomes depleted, and $atol_{RD} \rightarrow atol_{ST}$ as $FRC \rightarrow 0$. However, by then the radical fluxes are comparable to those of the stable species and no longer have to be subjected to special tolerances. Thus the FRC approximation serves the purpose of calculating a physically realistic scaling factor for $atol_{RD}$.

Once defined, the IP in Eq. (2.4) was solved using the CPLEX callable IP library [38]. Finally the solution to the IP was used to generate a reduced model in Chemkin compatible format.

2.3.2 Computational Gain From Reaction Elimination

The main goal of kinetic model reduction is to reduce the computational cost of reactor modeling. When this reduction is effected through reaction elimination, the computational saving is manifested in lower function and Jacobian evaluation costs. This is significant since previous work [62] has shown that Jacobian evaluations account for most of the CPU load during numerical integration of large stiff kinetic models. The effects of reaction elimination on function evaluation and Jacobian evaluation CPU times are quanitified here.

Function Evaluation

The cost of a function evaluation using the CONP batch reactor model in Chemkin II may be expressed as the sum of three terms:

$$CPU_{fe} = CPU_{re} + CPU_{se} + CPU_{te} \tag{2.8}$$

where

CPU_{fe} is the total CPU cost for a function evaluation of the kinetic model,

CPU_{re} is the CPU cost of evaluating the reaction rate vector $\boldsymbol{r}(T, \boldsymbol{y})$,

CPU_{se} is the CPU cost of multiplying the stoichiometric matrix, $\boldsymbol{\nu}$ with the reaction rate vector,

CPU_{te} represents all other costs including that of computing various thermodynamic properties.

The last of these terms, CPU_{te}, is essentially independent of N_R and hence unaffected by reaction elimination. Estimates for the first two terms' dependence on $N_{R,red}$ are as follows:

$$\begin{aligned} CPU_{re} &= N_{R,red} \langle CPU_{rr} \rangle \\ CPU_{se} &= \beta N_{R,red} N_S CPU_{mult} \end{aligned} \tag{2.9}$$

where

$N_{R,red}$ is the number of reactions in the reduced model,

$\langle CPU_{rr} \rangle$ is the mean cost of evaluating a single reaction rate term,

CPU_{mult} is the CPU cost of a multiplication operation,

β is the sparsity of the stoichiometric matrix.

N_S is clearly constant for reduced models that differ only in the number of reactions. When variations in β and $\langle CPU_{rr} \rangle$ are negligible, and $CPU_{te} << CPU_{se} + CPU_{re}$, CPU_{fe} decreases linearly with $N_{R,red}$. Alternatively, the speedup factor $\frac{CPU_{fe,full}}{CPU_{fe,red}}$ is expected to increase linearly with the model reduction factor $\frac{N_{R,full}}{N_{R,red}}$.

Jacobian Evaluation

The cost of constructing the dense, finite-difference Jacobian ($CPU_{je,den}$) (calculated using $N_S + 2$ function evaluations) simply scales as $(N_S + 2)CPU_{fe}$. As discussed in [62], the inherent sparsity of large-scale mechanisms such as the n-heptane mechanism can sometimes be exploited to evaluate Jacobians more efficiently. To employ sparse, analytical Jacobian evaluation, the dense ODE system defined by the conventional batch reactor equations in Eq. (2.1) must first be reformulated. One possible reformulation (referred to as the augmented density formulation in [62]) is obtained by supplementing the existing system with one additional independent variable, density, and one additional constraint:

$$\rho = \rho(\boldsymbol{y}, T, P) \tag{2.10}$$

where the expression on the right-hand side is the ideal gas law (or any suitable equation of state). The occurrence information and sparse analytical derivatives for this system must be suitably extracted, e.g., using an automatic differentiation tool such as DAEPACK [70]. This information is then provided to a sparse DAE solver, e.g., DSL48S [22], which is used to integrate the reformulated differential-algebraic system. The reader is referred to [62] for details. Finite differencing and sparse Jacobian evaluation of the full GRImech 3.0 and the n-heptane mechanisms have already been compared. The results in [62] indicate that the relative efficiency of either method depends on the size and sparsity of the particular mechanism at hand. Sparse evaluation incurs a certain amount of implementation-dependent computational overhead but becomes relatively more efficient as the size and sparsity of the mechanism increases. When comparing reduced models obtained by reaction elimination, the sparsity of the model tends to increase as the number of reactions decreases. The

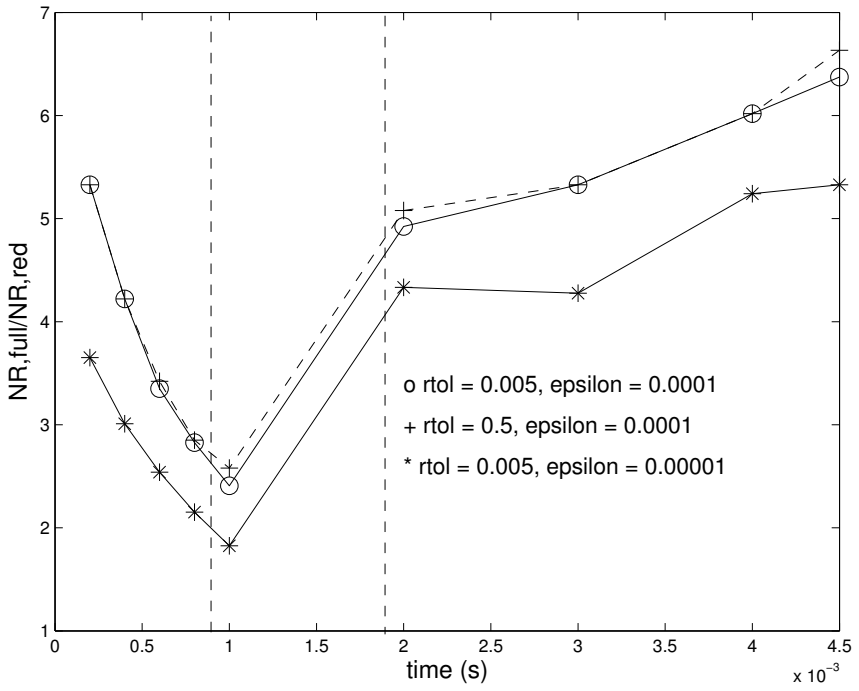

Figure 2-1: Optimal reduction factor at different points along the reaction trajectory for methane combustion in adiabatic, isobaric homogeneous reactor.The broken vertical lines show the time interval over which temperature rises from 1600K to 2400K. The full mechanism is GRImech 3.0 having $N_{R,full} = 325$. The optimally-reduced models have significantly fewer reactions, and vary in size between $51 \leq N_{R,red} \leq 135$ for $\epsilon = 0.0001$, $rtol_j = 0.005$. Larger models are needed to satisfy tighter error tolerances. Reaction conditions: 1 atm, $\phi = 0.5$, $T_0 = 1500K$.

size of the overhead and the degree of sparsity determine whether finite differencing or sparse Jacobian evaluation is more efficient in any particular case.

2.3.3 Reduced Models from GRImech 3.0

The first set of results presented here were obtained by reducing GRImech 3.0 [64], a methane combustion mechanism comprising 325 reactions and 53 species. The full mechanism was first integrated using an initial temperature of 1500 K and air-fuel feed mixture at atmospheric pressure with equivalence ratio $\phi = 0.5$. Fig. 2-1 shows the reduced models constructed at various points along the 5 ms reaction trajectory, using various tolerances ($\epsilon = 0.0001, rtol_j = 0.005$), ($\epsilon = 0.00001, rtol_j = 0.005$) and ($\epsilon = 0.0001, rtol_j = 0.5$). For the base case ($\epsilon = 0.0001$, $rtol_j = 0.005$) the

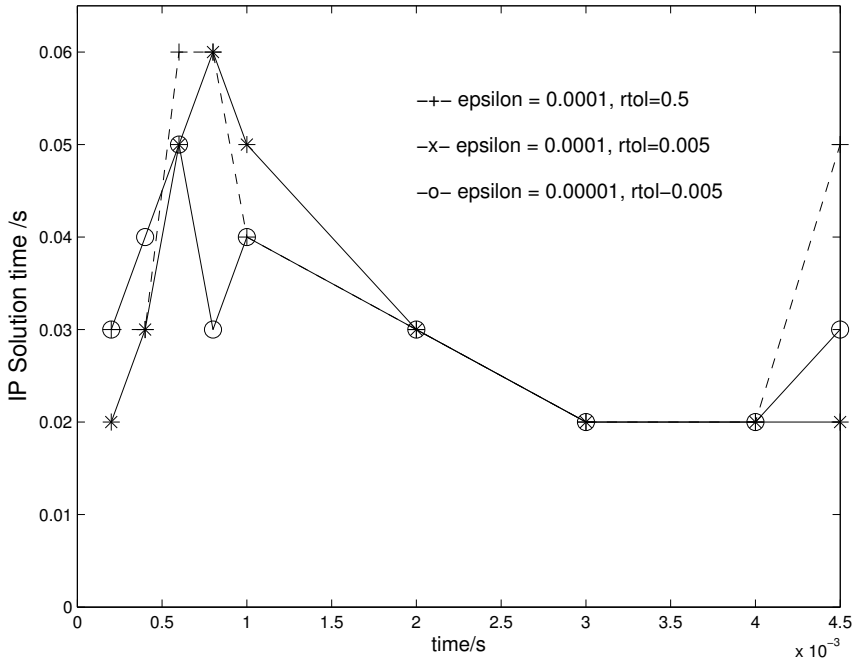

Figure 2-2: IP solution times for reducing GRImech 3.0 ($N_{R,full} = 325$) on a 733 MHz, 128 MB PC running Linux. The reaction conditions and tolerances imposed during model reduction have little effect on the time required to generate the optimally reduced models.

reduced models are 2-6 times smaller than the full model. As expected, the largest models are required at ignition where the chemistry is most complex. However, Fig. 2-1 shows that the choice of ϵ and $rtol_j$ can significantly influence the sizes of the reduced models obtained (and accordingly the expected speedup factor). Looser error tolerances clearly produce smaller, but less accurate reduced models. The particular tolerances applied should be guided by the user's minimum accuracy requirements, e.g., in a reacting flow implementation excessive error introduced by model truncation will not only erode accuracy, but may prevent convergence altogether. As shown in Fig. 2-2, the choice of ϵ and $rtol_j$ has little effect on the time required to construct the optimal reduced model. Rather the solution time for the branch-and-cut algorithm scales with $N_{R,full}$. The average IP solution time for all of the reduced models from GRImech 3.0 was 0.034 s using default CPLEX parameters.

As discussed earlier, a reduced model generated using the formulation in Eq. (2.4) is subject only to point constraints, and thus is known to be valid only at these

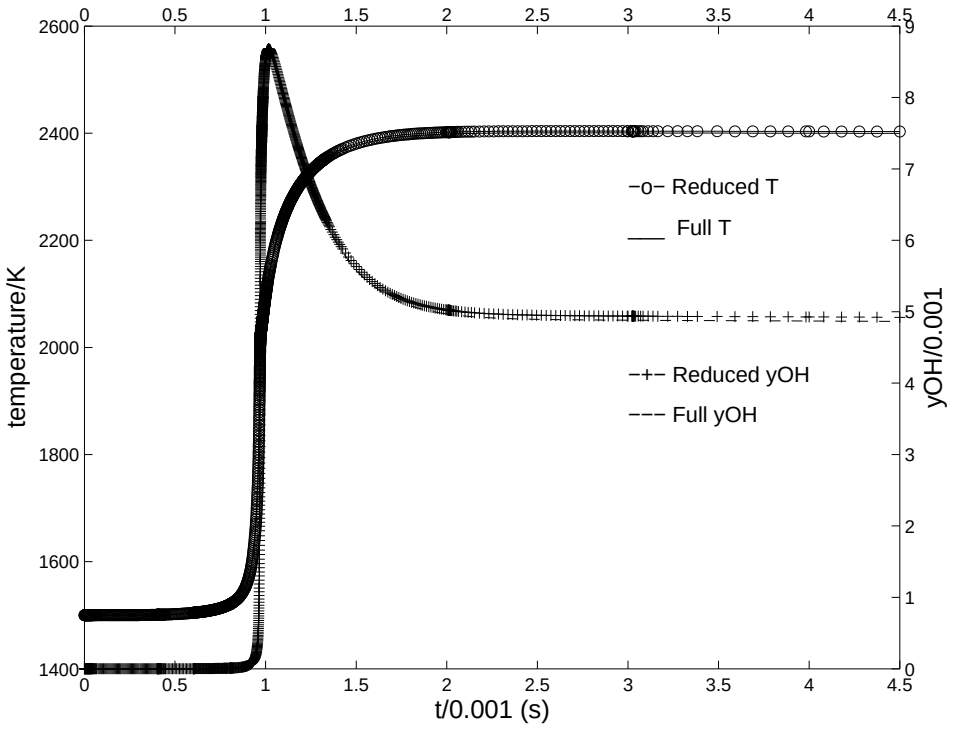

Figure 2-3: Temperature and OH mass fraction profiles using full and reduced models for methane combustion. Good agreement is achieved over the entire length of the reaction trajectory.

precise points. Each of the reduced models in Fig. 2-1 was generated by applying the constraints in Eq. (2.3) at two preselected points, $\boldsymbol{x}_{ref}(t_i)$ and $\boldsymbol{x}_{ref}(t_{i+1})$. By assuming each reduced model to be approximately valid over the entire interval $t_i \leq t \leq t_{i+1}$, the sequence of generated models can be used to integrate the initial value problem over the whole range. Since these intervals were selected back to back, a 'valid' reduced model was assigned to each portion (t_i, t_{i+1}) of the original reference trajectory. Using initial conditions identical to those of the reference profile, reaction trajectories were calculated using the sequence of reduced models in place of the full GRImech chemistry. Fig. 2-3 compares the temperature and hydroxyl mass fraction profiles obtained using the reduced models to those predicted using the full chemistry. The reduced models give results in very good agreement with the full chemistry. In the literature, reduced models derived using isoperimetric constraints have also shown good agreement with end point values, but often fail to predict transient properties

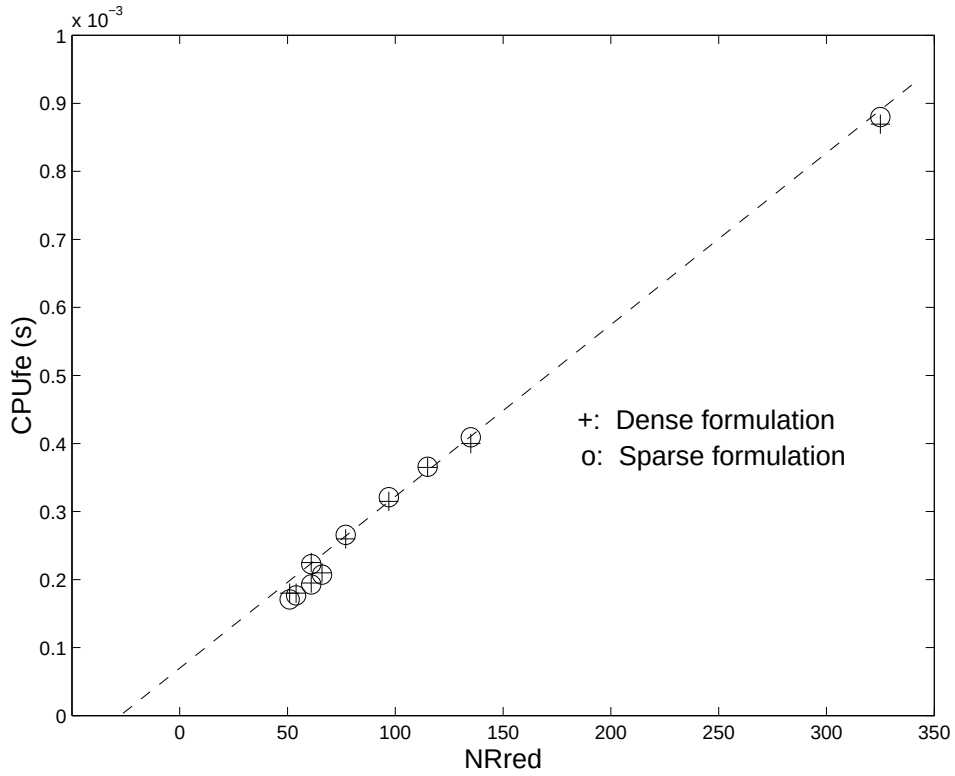

Figure 2-4: Function evaluation CPU times for sparse and dense formulations of the adiabatic, isobaric batch reactor. Function evaluation CPU time decreases linearly with the number of reactions $N_{R,red}$ for the GRImech-based reduced models.

such as ignition delay correctly [79, 4]. As shown in Fig. 2-3 the reduced model library approximates the full trajectory well over its entire time scale. However, it should be noted that the optimization procedure, based as it is on point-wise constraints, provides no guarantee of such close agreement. The approach taken here is based on the physical assumption that a reduced model that closely approximates the time derivatives of the true concentration profiles at several closely located points, should also closely approximate the concentration profile well over the range spanned by those points. Rigorous bounds on the error introduced by this approximation would be very helpful. A challenge for the future is to derive a rigorous quantitative relationship between the tolerances in the model reduction procedure and the resulting model truncation errors in computed concentration profiles.

The function and Jacobian evaluation costs for the reduced models were compared

$N_{R,red}$	β	$\langle CPU_{rr} \rangle \cdot 10^6$ (s)	$CPU_{te} \cdot 10^5(s)$	$\frac{CPU_{te}}{CPU_{fe}}$
325	0.0712	2.283	3.76	0.0428
61	0.0693	2.850	3.62	0.163
77	0.0696	2.720	3.69	0.140
97	0.0698	2.690	3.62	0.114
115	0.0700	2.571	3.89	0.107
135	0.0692	2.510	3.85	0.0944
66	0.0683	2.351	3.66	0.130
61	0.0693	2.370	3.91	0.201
54	0.0692	2.400	3.92	0.220
51	0.0688	2.432	3.90	0.228

Table 2.1: CPU time diagnostics for a single function evaluation of Γ in adiabatic, isobaric batch reactor simulations using reduced models based on GRImech. $N_{R,red}$ is the number of reactions in a given reduced model, β is the fraction of non-zero entries in the stoichiometric matrix. $< CPU_{rr} >$ is the average time required to calculate a reaction rate. CPU_{te} is the $N_{R,red}$-independent computational overhead associated with a single function evaluation; in all of the cases shown here it accounts for $\leq 23\%$ of the total function evaluation cost CPU_{fe}. These data explain why the overall function evaluation CPU time scales approximately linearly with $N_{R,red}$ (see section 3.2.1).

to those for the full GRImech 3.0. Fig. 2-4 shows that both dense ($CPU_{fe,den}$) and sparse ($CPU_{fe,sp}$) function evaluation costs decrease linearly with model size. This can be explained on the basis of the discussion in section 2.3.2. In this case function evaluation costs for the sparse and dense formulations are essentially indistinguishable. In the interest of clarity, the remaining part of this discussion addresses observations regarding $CPU_{fe,sp}$, but applies equally to $CPU_{fe,den}$. Table 2.1 shows that β [2] and $\langle CPU_{rr} \rangle$ are almost constant over the range of model sizes compared here. Furthermore, $CPU_{te} < 0.23 CPU_{fe,sp}$ for all of the reduced models. Since $CPU_{fe,sp}$ is dominated by terms linear in $N_{R,red}$ (i.e., CPU_{re} and CPU_{se} in Eq. (2.8)), overall $CPU_{fe,sp}$ decreases linearly with $N_{R,red}$ as shown in Fig. 2-4. The speedup factor $\frac{CPU_{fe,full,sp}}{CPU_{fe,red,sp}}$ increases from 1 to ~ 5 as $\frac{N_{R,full}}{N_{R,red}}$ varies between 1 and ~ 6 for the reduced models shown in Fig. 2-4. Fig. 2-5 compares the finite difference and sparse Jacobian evaluation costs for the GRImech-based models. Although sparse Jacobian

[2] β is the sparsity of the stoichiometric matrix, not to be confused with the sparsity of the Jacobian matrix which is usually increased by reaction elimination. Refer to Table 2.3 and discussion on n-heptane mechanism.

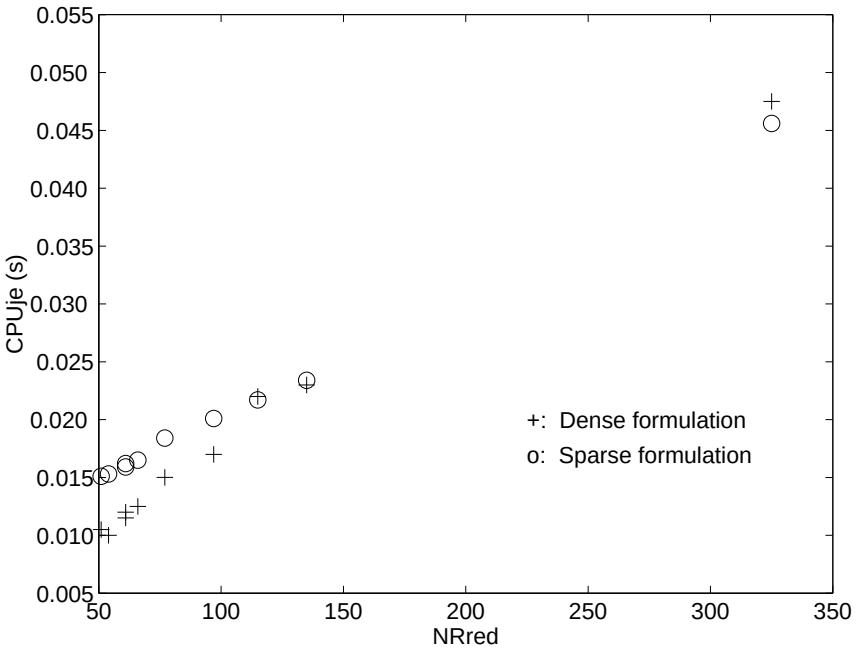

Figure 2-5: Jacobian evaluation CPU times for reduced models from GRImech 3.0. Finite differencing is more efficient than sparse Jacobian evaluation for the relatively small and dense reduced models.

evaluation is slightly cheaper for the full mechanism, finite difference evaluation becomes relatively more efficient as the size of the reduced model decreases. The finite difference Jacobian cost scales as $(N_S + 2) \cdot CPU_{fe,den}$, and $N_S = 53$ for all of the models. Since $CPU_{fe,den}$ increases linearly with $N_{R,red}$, the finite difference Jacobian evaluation CPU time, $CPU_{je,den}$, also increases linearly with $N_{R,red}$. Moreover, the finite difference speedup factor $\frac{CPU_{je,den,full}}{CPU_{je,den}}$ varies identically with $\frac{CPU_{fe,full,den}}{CPU_{fe,red,den}}$. Fig. 2-5 shows that although $CPU_{je,sp}$ also decreases linearly with $N_{R,red}$, sparse Jacobian evaluation is consistently less efficient than finite differencing for the relatively small and dense GRImech-based models.

2.3.4 Reduced Models from n-heptane Combustion Mechanism

In order to test the robustness of the method on large-scale mechanisms, it was applied to the Lawrence Livermore n-heptane combustion mechanism [16], consisting

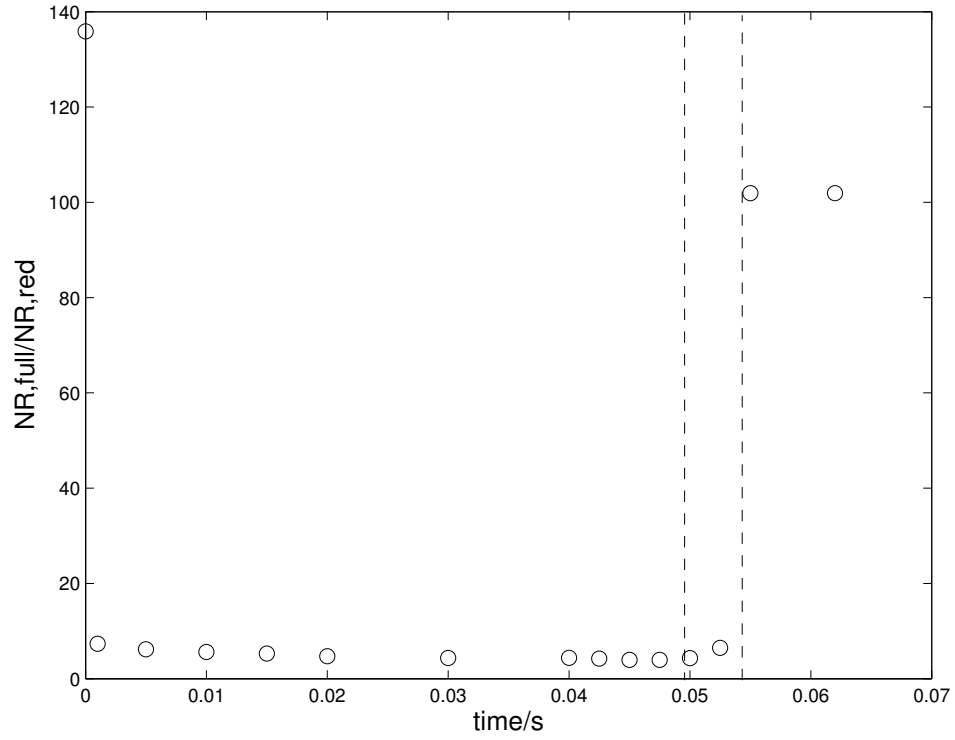

Figure 2-6: Optimal reduction factor at different points along the reaction trajectory for n-heptane combustion in adiabatic, isobaric batch reactor. The broken vertical lines show the time interval over which temperature rises from 1100K to 2500K. The full Lawrence Livermore mechanism has $N_R = 2446$ reactions. Reaction conditions: P= 1atm, $T_0 = 1000K$, $\phi = 0.82$.

of 2446 reactions and 544 species. The reduced models shown in Fig. 2-6 were obtained using ($\epsilon = 0.0001$, $rtol_j = 0.005$), with initial conditions $T = 1000$ K, $\phi = 0.82$ at atmospheric pressure, and by computing the trajectory for 0.2 s. Fig. 2-6 indicates that over 100-fold reduction of the n-heptane mechanism is obtained, compared to a maximum 6-fold reduction for GRImech 3.0 using identical tolerances. Fig. 2-6 emphasizes not only the dramatic reduction possible using simple reaction elimination, but also the possible variation in required model size with mechanism and reaction conditions. Again, the least reduced models are those constructed near ignition. Although the large reductions in model size yield significant speedup, Table 2.2 shows that neither $CPU_{fe,sp}$ nor $CPU_{fe,den}$ scales linearly with $N_{R,red}$ in the case of the n-heptane models. Nonetheless, these results are consistent with the discussion in section 2.3.2. Table 2.2 shows that β is almost constant (~ 0.006)

$N_{R,red}$	$CPU_{fe,sp} \cdot 10^3$ (s)	$CPU_{fe,den} \cdot 10^3$ (s)	$\beta \cdot 10^3$	$\langle CPU_{rr} \rangle \cdot 10^6$ (s)	$\frac{CPU_{te}}{CPU_{fe,sp}}$
2446	10.48	1.139	6.884	3.566	0.080
618	2.602	2.878	6.588	3.018	0.242
580	2.492	2.776	6.570	3.021	0.253
519	2.271	2.594	6.393	3.015	0.264
463	2.079	2.309	6.376	3.059	0.273
436	1.969	2.283	6.299	3.090	0.278
395	1.820	2.061	6.157	3.119	0.294
333	1.624	1.820	5.923	3.133	0.327
266	1.448	1.597	5.763	3.309	0.376
232	1.344	1.517	5.649	3.467	0.401
167	1.144	1.291	5.471	3.768	0.458
44	0.805	0.906	5.598	8.047	0.636
24	0.757	0.851	6.127	13.10	0.674
18	0.731	0.812	5.821	16.21	0.697

Table 2.2: Variation in CPU_{fe} with $N_{R,red}$ for reduced models based on the Lawrence Livermore n-heptane mechanism. β is approximately constant but $\langle CPU_{rr} \rangle$ varies by a factor of $\sim$5 for the range of reduced models shown here. As a result CPU_{re} does not scale linearly with $N_{R,red}$. Furthermore, the N_R-independent term, CPU_{te}, starts to dominate CPU_{fe} when $N_{R,red} << N_{R,full}$. The speedup factor $\frac{CPU_{fe,full}}{CPU_{fe,red}}$ is not linear with extent of reduction as shown in Fig. 2-7.

for the n-heptane mechanism-based models. However $\langle CPU_{rr} \rangle$ varies by a factor of $\sim$5 as $\frac{N_{R,full}}{N_{R,red}}$ varies between 1 and 136. Although the total cost of evaluating the Arrhenius rate expressions always scales with $N_{R,red}$, additional modifications such as third-body enhancement calculations contribute significantly here, and vary with the reaction set retained. As a result, the overall cost of evaluating the CPU_{re} term does not scale linearly with $N_{R,red}$. In addition, when the model is dramatically reduced (i.e., for $N_{R,red} << N_S$), CPU_{fe} begins to be dominated by the $N_{R,red}$-independent term. Table 2.2 shows that $\frac{CPU_{te}}{CPU_{fe}}$ is as high as 0.7 for the smallest n-heptane model. This corresponds to $\frac{N_{R,full}}{N_{R,red}} = 136$, compared to $\frac{N_{R,full}}{N_{R,red}} = 6.4$ for the most reduced GRImech-based model. At this point, further reaction elimination has little effect on the cost of the function evaluation; other approaches to model reduction, e.g., species elimination, must be investigated to achieve further computational benefit. The combined effects of nonconstant $\langle CPU_{rr} \rangle$ and high $\frac{CPU_{te}}{CPU_{fe}}$ ratio explain why CPU_{fe} does not decrease linearly with $\frac{N_{R,full}}{N_{R,red}}$ for the reduced models shown in Table 2.2. The nonlinear speedup factors $\frac{CPU_{fe,full,den}}{CPU_{fe,red,den}}$ and $\frac{CPU_{fe,full,sp}}{CPU_{fe,red,sp}}$ vary between 1 and $\sim$ 15 and are shown in Fig. 2-7. The finite difference and sparse Jacobian evaluation times for the n-heptane mechanism are shown in Table 2.3. Reflecting the nonlinear decrease in function evaluation time, the $CPU_{je,den}$ also decreases nonlinearly with $N_{R,red}$. However, because of the extremely high sparsity of the n-heptane models (95-99%), sparse Jacobian evaluation is 3-10 times more efficient than finite differencing for the reduced models obtained from the n-heptane mechanism. Fig. 2-8 shows the speedup factors for finite differencing and sparse Jacobian evaluation. Although finite differencing is actually less efficient, it reflects a higher speedup factor (saturation at $\frac{CPU_{je,den,full}}{CPU_{je,den,red}} \sim 15$ versus $\frac{CPU_{je,sp,full}}{CPU_{je,sp,red}} \sim 5$) because of the extremely high cost of a finite difference evaluation for the full mechanism.

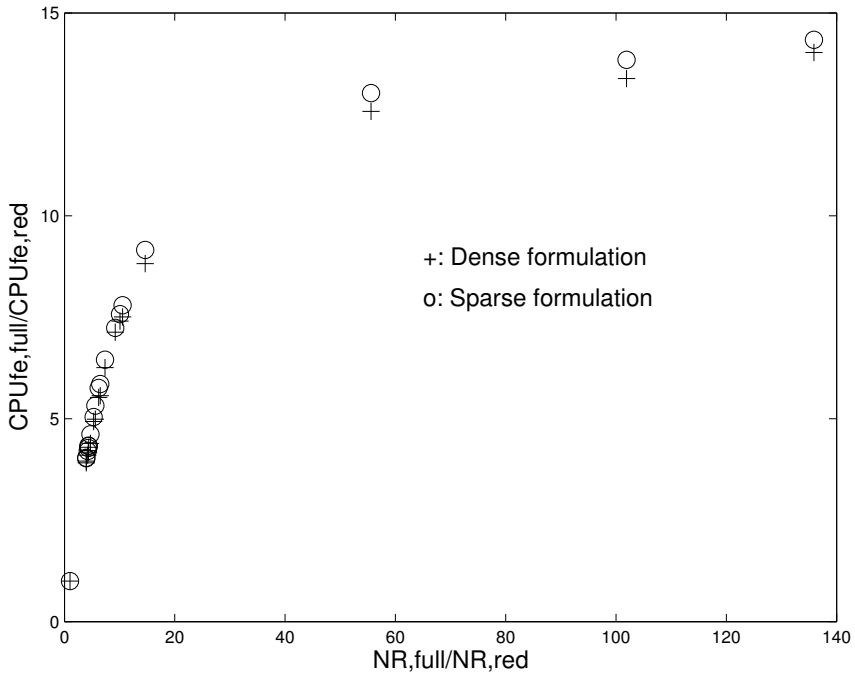

Figure 2-7: Speed-up in function evaluation for sparse and dense formulations of the n-heptane batch reactor model. ($CPU_{fe,sp}$ and $CPU_{fe,den}$ differ by $< 1\%$ for the full mechanism.) Speedup saturates when the reduced models become very small ($N_{R,red} \sim N_S$).

$N_{R,red}$	$CPU_{je,sp}$ **(s)**	$CPU_{je,den}$ **(s)**	**% sparse**
2446	0.663	6.246	95.2
618	0.272	1.569	98.3
580	0.264	1.527	98.3
519	0.251	1.417	98.4
463	0.248	1.257	98.5
436	0.229	1.246	98.6
395	0.233	1.129	98.6
333	0.223	0.984	98.8
266	0.212	0.864	98.9
232	0.206	0.809	99.0
167	0.193	0.700	99.1
44	0.169	0.477	99.3
24	0.165	0.445	99.4
18	0.164	0.424	99.4

Table 2.3: CPU times for Jacobian evaluation for reduced models based on the Lawrence Livermore n-heptane mechanism. The extremely high sparsity of the n-heptane mechanism makes sparse, analytical Jacobian evaluation significantly more efficient than conventional finite differencing.

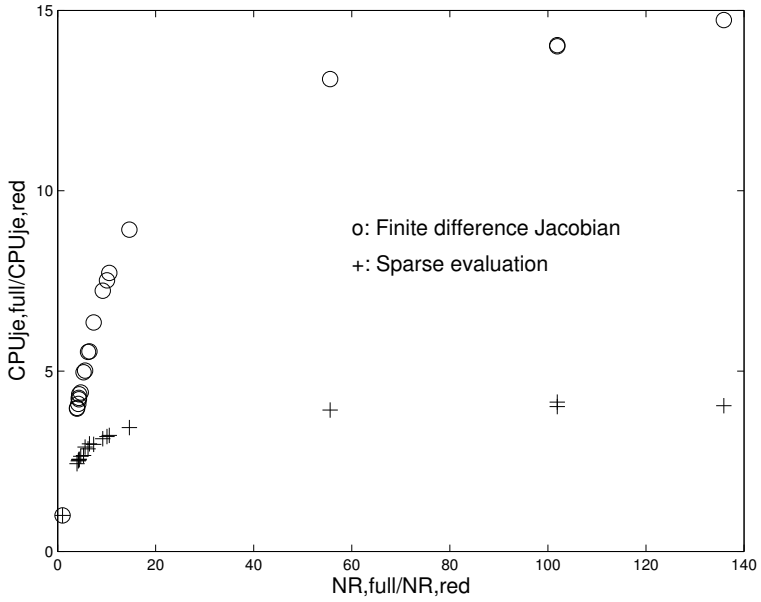

Figure 2-8: Speedup in Jacobian evaluation using finite differences (f.d.) and sparse analytical Jacobians. The (nonlinear) speedup in f.d. corresponds directly to the decrease in function evaluation cost (see Fig. 2-7). The sparse evaluation speedup factor $\frac{CPU_je,full,sp}{CPU_je,red,sp}$ is lower (although sparse, analytical Jacobian evaluation is actually more efficient than f.d., both for the full mechanism, and for each of the reduced models. See Table 2.3).

2.4 Other Methods for Locally-Constrained Reaction Elimination

Although it is considerably cheaper to solve than previous optimization formulations, the IP-based method is nonetheless more expensive than the more conventional reaction-elimination methods in the literature. A small test mechanism was used to investigate whether this extra cost is justified by the compactness of IP-derived models, when compared to reduced models of similar accuracy derived using other methods. The test mechanism (see Appendix A) was based on a hydrogen flame mechanism included in the Chemkin II distribution [39]. The MECHMOD [73] software was used to transform reversible reactions in the hydrogen mechanism into pairs of irreversible reactions, using the forward rate parameters and thermodynamic data provided. This was necessary to maintain compatibility with the software used to perform sensitivity-based reduction. The reference trajectory was calculated using the detailed kinetics

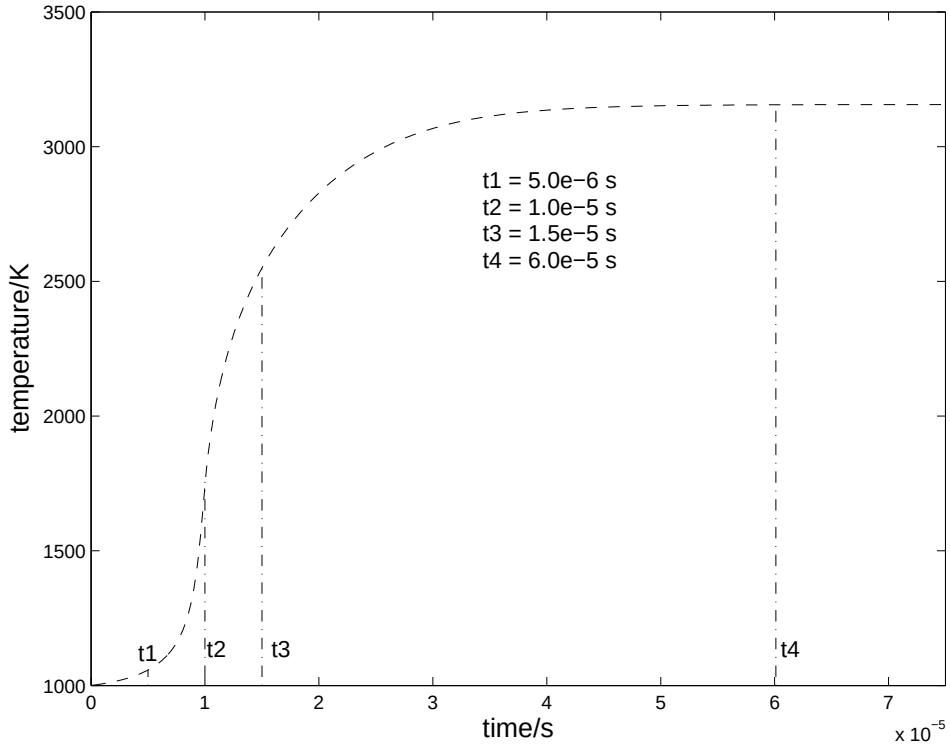

Figure 2-9: Reference points used for IP, DR, PCAS and PCAF reductions.

in Appendix A with the initial conditions shown in Table A.1. Fig. 2-9 shows the four points along the reaction trajectory at which IP-based reduction was performed using $\epsilon = 0.001$, $rtol_j = 0.05$, $N_t = 1$. Three other reaction-elimination methods were also applied in order to derive models satisfying:

$$|\Gamma_{j,ref} - \Gamma_j| \leq 0.001 \max_l |\dot{y}_{l,ref}| + 0.05|\Gamma_{j,ref}| \tag{2.11}$$

at each of these reduction points. Each of these methods, detailed reduction [24], principal component analysis of rate sensitivities [75], and principal component analysis of concentration sensitivities [76, 10] were summarized in Chapter 1.

For the DR implementation, $H + O_2 \longrightarrow OH + O$ was assumed to be the rate-limiting reaction following the recommended procedure in [24]. Since the algorithm does not allow $rtol_j$ or ϵ to be specified directly, an iterative procedure was used to derive reduced models satisfying Eq.(2.11) at each point t_i. The number of iterations

m and the threshold tolerance at the current iteration, κ_m, were both initialized to one. Reactions were eliminated using the criteria in Eq.(1.23), and relative errors for all species and temperature were evaluated using the resulting reaction set. Since the maximum relative error was found to be much above the desired tolerance of 0.05, the threshold value κ_m was updated using $\kappa_m = 0.9\kappa_{m-1}$ at the next iteration. The DR method was applied again, and the relative errors re-evaluated. This cyclic procedure was continued until a reduced model satisfying the error constraint in Eq. (2.11) was identified.

For the PCAS reduction the concentration sensitivities[1] were calculated using the SENKIN code [44]. At each reduction point, the KINALC postprocessor [72] was used to perform the eigenvector-eigenvalue decomposition of $\boldsymbol{S}^T\boldsymbol{S}$. A cyclic procedure similar to that used for DR was also applied to the PCAS analysis. The threshold values ζ_S, λ_S and the iteration count m were initialized to 0.2, 10^{-4} and 1 respectively. At each of the four reduction points, the decomposition yielded six major eigenvalues varying in size between 10^3 and 10^{-4}. The remaining (minor) eigenvalues were not considered in selecting reactions corresponding to high eigenvector elements. The maximum relative error associated with the PCAS-derived model was evaluated and found to be much higher than the desired tolerance of 0.05. At the next iteration, the threshold value was updated using $\zeta_{S,m} = 0.9\ \zeta_{S,(m-1)}$, the PCAS analysis was repeated and the relative error re-evaluated. This cyclic procedure was repeated until the derived reduced model satisfied Eqn. (2.11)

The PCAF analysis was also performed using the KINALC postprocessor. At each reduction point, the PCAF decomposition yielded six eigenvalues ranging between 0.1 and 10^{12} and two minor eigenvalues which were well below the threshold value of 10^{-3}. Thus only the 6 dominant eigenvalues were considered when retaining reactions corresponding to high eigenvector elements. Again the threshold value ξ_F was gradually lowered using $\xi_{F,m} = 0.9\ \xi_{F,(m-1)}$ at each iteration, until an acceptable

[1]For non-isothermal systems (such as the adiabatic model considered here), the reaction rate parameter k_i is not a constant. In such cases, the sensitivity coefficients used to construct $\boldsymbol{S}$ may be defined in a number of ways. In order to be consistent with the SENKIN code we consider $S_{j,i} = \frac{\partial \ln y_j}{\partial \ln D_i}$ where D_i is a multiplicative factor to the third-body enhanced Arrhenius rate expression for reaction i.

Method	t_1	t_3	t_3	t_4
IP	12	14	27	32
DR	14	20	32	34
PCAS	18	22	28	34
PCAF	17	21	29	33

Table 2.4: Sizes of hydrogen combustion models derived using various reaction-elimination algorithms. The entries show the number of reactions required to satisfy the error criteria defined by Eq. (16). Integer programming is guaranteed to generate the smallest model that satisfies these criteria at each point t_i

reduced model was found. The sizes of the reduced models obtained using IP, DR, PCAS and PCAF are shown in Table 2.4. Table A in the Appendix shows the final threshold values used to satisfy Eq. (2.11) using DR, PCAS and PCAF. It should be noted that the final threshold values depend on the update procedure (a geometric factor of 0.9 was selected arbitrarily here). Clearly, this has some influence on the exact size of the 'valid' reduced model reported in Table 2.4. Nonetheless, Table 2.4 show that the IP-derived reduced models are distinct from, and significantly more compact than reduced models of comparable accuracy identified using alternative approaches. This disparity in model size is expected to be even more pronounced in mechanisms of higher dimensionality. This has significant implications in adaptive chemistry implementations where the CPU cost scales linearly with $N_{R,red}$.

The importance of generating valid reduced models, i.e., reduced models which satisfy *some* error criterion, has already been emphasized; controlling model truncation error is essential to preserving accuracy and convergence in adaptive chemistry simulations. The IP method deterministically calculates an optimal reaction subset for any linear reformulation of the error definition in Eq. (2.3). Moreover, the parameters ϵ and $rtol_j$, can be specified relatively intuitively by the user. However reaction-elimination methods such as DR, PCAS and PCAF are not designed to yield reduced models which are known to be accurate to some user-defined tolerance, at least not in the sense of Eq. (2.3). This necessitates the iterative parameter reduction scheme, and consequently increases the overall cost of implementation.

2.5 Reduced Model Library for Partially-Premixed Methane Flame

The batch reactor studies described in the previous section have shown that reaction elimination can be performed robustly and efficiently for large-scale mechanisms, and presents significant speedup potential without over-compromising the accuracy of a reactor simulation. Of course, the real utility of model reduction lies in quantifiable computational savings in the simulation of complex reacting flows. In this section the reaction elimination method is used to construct a reduced-model library for 2-D laminar methane flame.

A steady-state flow field for a partially-premixed, laminar methane burner flame (case 2 in [60]) was first simulated using a 217-reaction mechanism derived by eliminating nitrogen chemistry from GRImech 3.0. The full mechanism, reacting flow conditions, solution procedure and convergence criteria are described in detail in [60, 61]. Optimally-reduced models were then assigned to each of the 12,740 interior points in the simulation grid. At every point previously-constructed reduced models were searched (in order of increasing size) to check if an existing model was valid for the point under consideration. A valid model was required to satisfy Eq. (2.3) at the current point. Whenever no suitable reduced model was found, the local temperature and species mass fractions from the converged full chemistry solution were used to calculate the ref quantities in Eq. (2.3), and the problem in Eq. (2.4) was re-solved using $\epsilon = 0.001,\ rtol_j = 0.01$. The derived model was assigned to the current point, and also added to the reduced model library. A total of 657 reduced models ranging in size from $N_{R,red} = 10$ to $N_{R,red} = 126$ were eventually included in the model library. The average size of these reduced models was found to be $N_{R,red} = 55$. Once the model library was complete, we compared the cost of integrating the converged solution one further iteration (CFL = 0.1) using the full chemistry in one case, and the optimally-reduced models in the other. The VODE solver was used to perform the stiff integration in both cases. Table 2.5 shows that even near convergence stiff chemistry integration is by far the most computationally intensive part of the full

Timing and ODE diagnostics	full chemistry	reduced models
stiff solver time (s)	275	91
overall solution time (s)	304	120
fn evals.	90,901	153,216
LU decompositions	43,474	56,819
convergence failures	135	258
error test failures	69	492
Newton iterations	52,681	114,939

Table 2.5: CPU times and VODE diagnostics for a single reacting-flow iteration of a 2-D laminar flame using full chemistry and IP-reduced models. The total stiff solver time decreases by a factor of ~ 3 when a library of reduced models is used in place of the full GRImech 3.0 chemistry.

chemistry simulation ($\sim 90\%$). By using the IP models the stiff solver CPU time decreases by a factor of ~ 3 from 275 s to 91 s. Since reaction elimination does not affect computations other than the ODE integrations, the overall solution time also decreases by exactly the same amount. This computational gain is due to cheaper function evaluations (including those used for finite difference Jacobian evaluation) when applying reduced models. As shown earlier, the cost of the kinetic source term evaluation scales roughly with the number of reactions in the reduced model (as long as $N_{R,red} > N_S = 35$). Here, using reduced models of average size $N_{R,red} = 55$, in place of the full 217-reaction mechanism, results in a $\sim$ 4-fold decrease in function evaluation time. However, this does not correspond exactly to the observed speedup factor of $\sim$3. As shown in Table 2.5, the cheaper function evaluations are compensated to some extent by other effects, including higher number of convergence and error test failures in the ODE solver. This is believed to be an artifact arising from the fact that a converged full chemistry solution was used as the starting point for the comparative iteration.

2.6 Conclusions

The linear reaction elimination formulation described in this chapter provides a way to generate optimally-reduced kinetic models cheaply and robustly from large-scale

mechanisms, under widely varying reaction conditions. When applied to constant-pressure, adiabatic batch reactors, dramatic reductions in mechanism size were obtained and the associated speedup was found to scale linearly with model size as long as $N_{R,red} > N_S$. The reduced models obtained were found to have good predictive value and allowed important physical effects, e.g., ignition delay, to be captured correctly.

The new formulation thus addresses a number of the issues barring the application of previous integer programming formulations to model library generation. The primary short-coming of the point-constrained approach is the absence of an associated range of validity. A rigorous range of validity is essential for preserving accuracy and possible convergence in reacting-flow simulations. Interval methods are being investigated as a possible solution to identifying conservative ranges of validity for point-constrained reduced models. As discussed in Chapter 3, a more distant goal is to identify maximum ranges of validity by solving generalized semi-infinite programs.

When applied to generate reduced models for a partially-premixed laminar flame, the reduced models yielded a significant decrease in stiff solver CPU time. However, a number of practical issues associated with generating, archiving and switching reduced models must be resolved before a full adaptive chemistry is possible. Some of these are discussed further in Chapter 3.

Chapter 3

Future Work for in Kinetic Model Reduction

The point-constrained reaction elimination problem described in Chapter 2 is only the simplest of a number of different formulations of the model reduction problem. In this chapter possible extensions to species reduction, reduction under uncertainty and reduction subject to postulated ranges of validity are suggested. A number of practical issues related to the implementation of reduced models in reacting-flow simulations are also considered here. Some of these challenges, e.g., efficient model archival, have already been fielded by other researchers in the area, and useful suggestions may be available in the literature [71, 54, 45].

3.1 Species Elimination

As shown in Chapter 2, the realizable computational gain from reaction elimination is limited to a linear decrease in CPU time with reduced model size. In order to achieve superlinear reductions in simulation time, species elimination/pseudo-steady state formulations must be investigated. Past IP formulations have all treated species reduction as a subset of reaction reduction by performing nested optimisations. However, species deletion and and reaction reduction can be applied simultaneously using

an additional binary vector, $\boldsymbol{w}$, as follows:

$$
\begin{aligned}
&\min_{\boldsymbol{z},\boldsymbol{w}} \textstyle\sum_{j=1}^{N_S} \alpha_j w_j + \sum_{i=1}^{N_R} \beta_i z_i \\
&\frac{M_j \sum_{i=1}^{N_R} \nu_{j,i} z_i k_i \Pi_{j=1}^{N_S} w_j c_{j,l,ref}}{\rho_{l,ref}} - \Gamma_{j,l,ref}(\boldsymbol{x}_{l,ref}) \leq atol_j + rtol_j |\Gamma_{j,l,ref}|, \quad j = 1, \ldots N_S,\ l = 1, \ldots, N_t \\
&\Gamma_{j,l,ref}(\boldsymbol{x}_{l,ref}) - \frac{M_j \sum_{i=1}^{N_R} \nu_{j,i} z_i k_i \Pi_{j=1}^{N_S} w_j c_{j,l,ref}}{\rho_{l,ref}} \leq atol_j + rtol_j |\Gamma_{j,l,ref}|, \quad j = 1, \ldots, N_S,\ l = 1, \ldots, N_t \\
&w_j \sum_{i=1}^{N_R} |\nu_{j,i}| \geq \sum_{i=1}^{N_R} |\nu_{j,i}| z_i, \qquad\qquad j = 1, \ldots N_S \\
&\boldsymbol{z} \in \{0,1\}^{N_R}, \quad \boldsymbol{w} \in \{0,1\}^{N_S}
\end{aligned}
\tag{3.1}
$$

where α_j and β_i are user-defined weighting factors. The first two sets of constraints in (3.1) correspond to the error bounds enforced in the reaction elimination formulation. The third set of constraints in (3.1) enforces mass conservation by ensuring that a species is deleted only if all of the reactions in which it appears, either as a product or a reactant, are simultaneously deleted, i.e., the net predicted flux for any species which does not appear in the reduced model is zero.

The combined species and reaction elimination problem is considerably more difficult than the simple reaction elimination problem because the error constraints in (3.1) are not only nonlinear, but also nonconvex in the binary variables $\boldsymbol{w}$ and $\boldsymbol{z}$. Although deterministic algorithms for such integer nonlinear programs (INLPs) do exist in the literature, they are much more computationally intensive than those available for linear integer programs. Thus, solving (3.1) for large-scale kinetic mechanisms presents a significant challenge.

Existing algorithms for INLPs fall into two main categories: linearization approaches in which the INLP is converted into an equivalent mixed-integer linear program (MILP) through the addition of further constraints and variables [26, 27, 1, 2], and direct solution strategies which rely on the construction of convex underestimators for nonconvex terms in (3.1). An initial examination suggests that direct methods are unlikely to outperform the linearization approach, since the convexification of bilinear and trilinear terms boils down to the procedure described in [27]. If the linear transformation is pursued, the main challenge lies in achieving a reasonable tradeoff

between the size and tightness of the linear programming relaxation. Alternatively, it may be worthwhile to explore other formulations for species reduction.

3.2 Ranges of Validity for Point-reduced Models

The constant-pressure batch reactor simulations described in Chapter 2 were performed using a simple heuristic to assign ranges of validity to the point-reduced models: a reduced model generated subject to constraints at a number of points in composition space was assumed to be valid at every point within the convex hull of those points.

For an adaptive chemistry application Δt is small enough for this to be a reasonable assumption. Split-operator reacting-flow codes solve the kinetic ODE, Eq. (2.1), repeatedly, but only over very short intervals (e.g., $\Delta t \sim 10^{-6} s$ for flames). The mesh size is set fine enough to resolve all the spatial gradients, and Δt is determined by the CFL condition. Regardless of the absolute time scale, the species change over Δt cannot be large or the spatial discretization will not resolve the gradients correctly. In order to rigorously control the error incorporated by model truncation, the reduced model should ideally satisfy:

$$\left| \int_{t_0}^{t_0+\Delta t} (\dot{x}_j - \dot{x}_j)\, dt \right| - \delta_2 \Delta t \leq 0, \qquad j = 1, \ldots, N_S + 1 \tag{3.2}$$

where δ_2 is the user-defined error tolerance. This will certainly be satisfied if:

$$|(\Gamma_j - \Gamma_{j,ref})| \; - \delta_2 \leq 0 \quad t_0 \leq t \leq t_0 + \Delta t, \qquad j = 1, \ldots, N_S + 1. \tag{3.3}$$

Reduced models generated by point-constrained reaction elimination will very likely satisfy Eq. (3.3) along the full mechanism trajectory from which they were derived. The difficulty lies in guaranteeing that Eq. (3.3) will also be satisfied in the neighborhood of the original trajectory. Once a reduced model has been generated using point constraints, such a range of validity may be identified using an iterative interval evaluation procedure. The reader is referred to [3, 48] for a comprehensive review

of interval notation and methods. The most relevant ideas are also summarized in Chapter 4 of this thesis.

Based on the observation that the set of dominant reactions is relatively constant over significant regions of the reaction space of interest [60], it is reasonable to assume that a point-reduced model nonetheless satisfies (3.3) over a connected set of points (X_v) in composition space. The goal of the following iterative procedure is to identify some interval $[\boldsymbol{x}^l,\ \boldsymbol{x}^u] \subset X_v$ having non-zero width in each dimension:

1. Guess an initial range of validity $X_0 = \left[\boldsymbol{x}_0^l,\ \boldsymbol{x}_0^u\right]$ such that $\{\boldsymbol{x}_{ref,l}\} \subset X_0$, where $\{\boldsymbol{x}_{l,ref}\}$ is the finite set of composition points at which constraints were imposed during model reduction,

2. Set k := 0

3. Evaluate an inclusion for $h_j(\boldsymbol{x}_{ref}) = |\Gamma_{j,ref}(\boldsymbol{x}_{ref}) - \Gamma_j(\boldsymbol{x}_{ref})|$ on X_k, i.e., $H_j(X_k) \supset \{h_j(\boldsymbol{x}_{ref}) : X_k^l \leq \boldsymbol{x}_{ref} \leq X_k^u\}, \quad j = 1, \ldots, N_S,$

4. If $H_j^u(X_k) \leq \boldsymbol{\delta}_2$, then stop. The reduced model is valid everywhere within the interval X_k. Else reduce the width of X, set $k := k + 1$ and return to Step 3.

As the estimated set X_k is reduced at each iteration, both, the range of truncation errors encountered, and the degree of overestimation incorporated by $H^u(X_k)$, are also reduced. The inclusion bound $H^u(X_k)$ may be obtained as the maximum of the upper inclusion bounds for $\Gamma_{ref}(\boldsymbol{x}_{ref}) - \Gamma(\boldsymbol{x}_{ref})$ and $\Gamma(\boldsymbol{x}_{ref}) - \Gamma_{ref}(\boldsymbol{x}_{ref})$. A convergent inclusion of order one or higher can be calculated for each of these two continuous functions. Consequently, the overestimation in the desired bound $H^u(X_k)$ also decreases linearly with the width of X_k.

The iterative process described above has been tested using natural interval extensions, and Taylor models of order two calculated by DAEPACK [70]. As expected, the Taylor models were found to converge much more quickly and produced correspondingly larger intervals at termination.

The width (in each dimension) of the identified interval $[\boldsymbol{x}^l,\ \boldsymbol{x}^u]$ depends not only on the actual region of validity, X_v, and the type of inclusion function applied,

but also on the heuristic used to decrease the set X_k at each iteration. The inclusion bound $H^u(X_k)$ is guaranteed to converge as long as the dimension of maximum width is decreased at each iteration. However, it is much more difficult to decide which dimensions should be reduced, and to what extent, in order to approximate the range of validity within a reasonable number of iterations without grossly underestimating X_v.

Although the iterative procedure suffices to identify *some* valid range for a point-reduced model, it is preferable to identify the largest possible interval of validity. Clearly, there is some tradeoff between the size of the generated model and its range of validity in composition space, e.g., a model generated by applying point constraints over a large region of composition space is likely to retain more reactions than one generated by applying the same number of constraints, centered around the same point, but distributed over a much smaller region of composition space. However, once a reduced model has been generated, identifying a larger (more optimal) interval within its region of validity allows the model to be reused at a greater number of times within the reacting flow simulation. This is advantageous for two reasons: it reduces the total number of models required to cover the composition space of interest, and it is also likely to reduce the computational costs associated with model switching.

The following generalized optimization problem may be solved in order to identify the maximum interval of validity associated with a reduced model:

$$\begin{gathered}\max_{\boldsymbol{x}^u,\boldsymbol{x}^l} \sum_{j=1}^{N_S+1} \alpha_j (x_j^u - x_j^l) \\ |\Gamma_{j,ref}(\boldsymbol{x}_{ref}) - \Gamma_j(\boldsymbol{x}_{ref})| \leq atol_j + rtol_j |\Gamma_{j,ref}(\boldsymbol{x}_{ref})| \quad \forall \boldsymbol{x}_{ref} \in [\boldsymbol{x}^l,\ \boldsymbol{x}^u], \quad j = 1, \ldots, N_S\end{gathered} \tag{3.4}$$

where α_j is some weighting factor for the individual species. The difficulty in solving (3.4) arises from the fact that an infinite number of constraints must be satisfied on $[\boldsymbol{x}^l, \boldsymbol{x}^u]$. Problems of this form are referred to as generalized semi-infinite programs and have been studied only relatively recently in the literature. The terms $\boldsymbol{\Gamma}_{ref}$ and $\boldsymbol{\Gamma}$ are now written explicitly in terms of the state variables $\boldsymbol{x}_{ref}$, and unless the reduced reaction set consists entirely of unimolecular reactions, the error constraints in (3.1)

are highly nonlinear and nonconvex. Numerical methods for solving such problems even to guaranteed feasibilty, let alone to guaranteed optimality, are not currently known. On the other hand, a special subset of GSIPs known as standard semi-infinite programs (SIPs) have been studied to a much greater extent in the literature. As a first step towards solving GSIPs, a global optimization for smooth, nonconvex SIPs is developed in Chapters 4 and 5 of this thesis. The range problem is then briefly revisited in Chapter 6.

As noted earlier, one alternative to a posteriori range identification is the reduction of a full mechanism subject to a postulated range of validity. This approach results in the following integer, nonlinear, semi-infinite program (ISIP):

$$\begin{aligned} &\min_{\boldsymbol{z}} \sum_{i=1}^{N_R} z_i \\ |\boldsymbol{\Gamma}_{j,ref}(\boldsymbol{x}_{ref}) - \boldsymbol{\Gamma}_j(\boldsymbol{x}_{ref}, \boldsymbol{z})| &\leq atol_j + rtol_j |\Gamma_{j,ref}(\boldsymbol{x}_{ref})|, \quad \forall \boldsymbol{x}_{ref} \in [\boldsymbol{x}^l, \boldsymbol{x}^u], \quad j = 1, \ldots, N_S \\ &\boldsymbol{z} \in \{0,1\}^{N_R} \end{aligned} \tag{3.5}$$

In certain cases, the formulation in (3.5) may be more relevant, e.g., applications for which a minimum range of model validity is known a priori. However, the use of (3.5) is currently limited by the lack of available numerical solution methods. As noted in Chapter 6, the branch-and-bound framework developed in this thesis for smooth, nonlinear SIPs may be extended to ISIPs in the future.

3.3 Reduction Under Parametric Uncertainty

In practice, the rate parameters used to construct detailed kinetic mechanisms are never known with absolute certainty. However, a reasonable bound on the degree of error reflected by the nominal value is often known. In such cases, it is possible to derive reduced models which are valid for the entire range of parameter values by solving a ISIP using the unknown parameters, instead of, or along with $\boldsymbol{x}_{ref}$, as the interval variables in (3.5).

3.4 Model Archival and Retrieval

In the single-step integration of the methane flame studied in the previous chapter, reduced models were assigned a priori to each of the cells in the reacting-flow grid. In a full adaptive chemistry simulation, the reaction conditions at each cell will change at each iteration and the reduced model in effect will have to updated accordingly, until the steady-state solution is approached. As noted in [71], searching the reduced model library efficiently for an appropriate reduced model presents a significant challenge. Over 600 reduced models were used to simulate the laminar methane flame described in Chapter 2. Even for this relatively simple system, it is not practical to linearly search the entire library at each one of the 10,000+ grid cells, at each iteration. Tonse [71] and Pope [54] used binary search trees to retrieve model parameters associated with the current index in the hyperspace. Sophisticated implementations of many data structures are now readily available, e.g., using the C++ Standard Template Library, and a careful choice will significantly decrease the CPU time associated with retrieving models and also storing those which are generated on-the-fly.

3.5 Mapping the Reaction Space of Interest

For a real adaptive chemistry application, no flow field solution using the full chemistry is available a priori to calculate the ref quantities. Indeed, the intractability of such (full chemistry) reacting flow simulations provides the main motivation for developing adaptive chemistry! The range of reaction conditions for which reduced models are required is thus unknown, i.e., the quantities subscripted ref in the optimization problem are not available. The iterative procedure applied in [24] may be used. In this case a partial library is constructed using an estimate of the converged flow field (e.g., using a skeletal mechanism). This library must then be supplemented 'on the fly' as the integration proceeds towards convergence, and new areas of reaction space are explored. Ranges of validity for the reduced models are essential to ensure that error introduced by model truncation is small compared to the error tolerance

for converging the reacting flow simulation.

Chapter 4

An Inclusion-constrained Reformulation Approach to Solving Semi-infinite Programs

4.1 Introduction

Optimization problems of the form

$$\begin{aligned} &\min_{\boldsymbol{x}} f(\boldsymbol{x}) \\ &\text{s.t. } g(\boldsymbol{x},\boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in P \subset \mathbb{R}^{n_p}, \quad |P| = \infty \\ &\boldsymbol{x} \in X \subset \mathbb{R}^{n_x} \end{aligned} \tag{4.1}$$

are commonly referred to as semi-infinite programs (SIPs) in the literature. A finite number of decision variables x_j are subjected to an infinite number of constraints from the compact set P. Additional finite or semi-infinite constraints may be applied to the variables x_j, but in the interest of clarity it is assumed here that a single semi-infinite constraint $g(\boldsymbol{x},\boldsymbol{p}) \leq 0$ is present. The set X is assumed to be compact and P is defined as a Cartesian product of intervals, although in general P may be defined by a set of inequalities. The SIP-feasible set is assumed to have a non-empty interior. In addition, when a global minimum of (4.1) is desired, the nonlinear functions f and

g are required to be continuously-differentiable in $\boldsymbol{x}$, $\forall \boldsymbol{p} \in P$, and both X and P are assumed to be Cartesian products of intervals.

As described earlier, the investigation of semi-infinite programming in this thesis is motivated by the occurrence of semi-infinite programs in various formulations of the kinetic model reduction problem in Chapter 1. However, problems of the form of (4.1) arise in a variety of engineering scenarios where hard constraints must be satisfied for a range of parameter values. In design applications, such ranges are often considered because one or more of the model parameters are subject to bounded uncertainties, e.g., the total design cost of a reactor which experiences fluctuations in cooling water flow may be minimized subject to a constraint on the maximum allowable temperature rise in the reactor. In this case, the temperature constraint must be satisfied for a range of values for the cooling water flowrate.

In the pollution control application described in [32], the cost of reducing emissions from n_x polluting sources is minimized such that the annual mean ground level pollution concentration at each point, p, in a region, P, satisfies some standard, $G(p)$:

$$\begin{aligned} &\min_{\boldsymbol{x}} \sum_{i=1}^{n_x} c_i x_i \\ &\sum_{i=1}^{n_x} \left(1 - x_i(p)\right) R_i(p) \leq G(p) \quad \forall p \in P \\ &0 \leq x_i \leq 1, \qquad\qquad i = 1, \ldots n_x, \end{aligned} \tag{4.2}$$

where $R_i(\boldsymbol{p})$ is the annual mean concentration at point p due to source i before control, x_i is the fractional reduction at source i, c_i is the linear cost coefficient associated with reduction at source i, and $R(p)$ is the maximum allowable pollutant concentration at location p. Here the constraint on maximum pollutant concentration must be satisfied for a range of values for p.

In the just-in-time production planning application described in [43], the delivery schedules for a number of orders are optimized by miminimzing a quadratic penalty function for earliness/tardiness subject to constraints on the total resources available

at any given time:

$$\begin{aligned} &\min_{\boldsymbol{x}} \sum_{i=1}^{n_x} (x_i - d_i)^2 \\ \text{s.t. } &\sum_{i=1}^{n_x} R_i(p, x_i) \leq G(p), \quad \forall p \in [0, P] \\ &0 \leq L_i \leq x_i \leq P, \quad i = 1, 2, \ldots n_x \end{aligned} \tag{4.3}$$

where P is the time horizon of the delivery schedule, d_i, L_i and $G_i(x_i, p)$ are the due date, production cycle and resource requirement for product i respectively, N is the total number of orders, and $R(p)$ is the total resource availability at time p. In this case, the infinite constraint arises from the need to satisfy resource availability at each point, p, in the delivery schedule.

Theoretical developments in semi-infinite programming have generally been focussed on extending optimality and duality results from finite nonlinear programming [32]. Defining $\{\boldsymbol{p}_v\}$ to be the set of points in P which result in active constraints $g(\bar{\boldsymbol{x}}, p) = 0$ at $\bar{\boldsymbol{x}}$, an analog of the first-order KKT conditions may be stated as follows:

$$\frac{\partial f}{\partial \boldsymbol{x}} + \sum_{v=1}^{r_v} \mu_v \frac{\partial g_v}{\partial \boldsymbol{x}}, \quad \mu_v \geq 0, \quad v = 1, \ldots r_v \tag{4.4}$$

where $g_v = g(\boldsymbol{x}, \boldsymbol{p}_v)$, i.e., (4.4) is a necessary condition for $\bar{\boldsymbol{x}}$ to be a local minimum of the SIP provided the following constraint qualification holds at $(\boldsymbol{x})$ for some $\boldsymbol{\xi}$:

$$\boldsymbol{\xi}^T \frac{\partial g_v}{\partial \boldsymbol{x}} < 0. \tag{4.5}$$

(4.4) simply states that $\frac{\partial f}{\partial \boldsymbol{x}}$ is required to lie within the convex cone generated by the gradients $\frac{\partial g_v}{\partial \boldsymbol{x}}$, and the constraint qualification in (4.5) guarantees that the convex cone of gradients is closed. A stronger sufficiency condition for $\bar{\boldsymbol{x}}$ to be a strongly unique local minimum is obtained by requiring $\frac{\partial f}{\partial \boldsymbol{x}}$ to lie in the interior of the convex cone generated by the gradients of the active constraints.

As discussed in the following section on numerical methods, when certain regularity conditions [32, 57] are known to hold, the feasible set of the SIP in the neighborhood of a given point can be represented by a finite number of constraints which are

parametric in $\boldsymbol{x}$. Second order optimality conditions for twice-continuously differentiable SIPs may be derived by applying known optimality conditions to the equivalent, finitely-constrained problem. A local minimum $\bar{\boldsymbol{z}}$, with associated Lagrange multipliers, $\boldsymbol{\mu}$, must satisfy the stationarity condition in (4.4) and the Hessian of the corresponding Lagrangian function must be positive definite. Similarly, if the Hessian of the Lagrangian function is known to be positive definite at a given stationary point $\bar{\boldsymbol{z}}$, then $\bar{\boldsymbol{z}}$ is known to be strict local minimum of the SIP.

4.2 Existing Numerical Methods for Nonlinear SIPs

Numerical methods for SIPs derive a finite analog of the SIP such that known methods and optimality conditions from nonlinear programming can be applied. As shown in [53], the following exact finite reformulation of (4.1) is inherently nonsmooth:

$$\begin{aligned} &\min_{\boldsymbol{x}\in X} f(\boldsymbol{x}) \\ \text{s.t. } &\max_{\boldsymbol{p}\in P} g(\boldsymbol{x},\boldsymbol{p}) \leq 0. \end{aligned} \tag{4.6}$$

Although nondifferentiable optimization techniques have been previously applied to (4.6), these methods are computationally expensive due to the large number of function evaluations required [53]. At best these approaches satisfy local optimality conditions; when the constraint g is not partially concave in the variables $\boldsymbol{p}$, neither the optimality nor the feasibility of the point identified by these methods is guaranteed.

For smooth, nonlinear SIPs which are partially convex in the interval variables p_m, local maximizers of the function $g(\boldsymbol{x},\boldsymbol{p})$ are known to occur at extreme points of P [31]. This property suggests that an equivalent, finite, smooth nonlinear program (NLP) may be derived by replacing the parametric constraint in (4.6) with the following set of 2^{n_p} constraints:

$$\begin{aligned} &\min_{\boldsymbol{x}\in X} f(\boldsymbol{x}) \\ \text{s.t. } &g(\boldsymbol{x},\boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in P_e \end{aligned} \tag{4.7}$$

where P_e is the set of extreme points of P. The NLP in (4.7) is then solved to global optimality for the SIP solution. A number of heuristics for alleviating the computational load associated with the exponential size of this reformulation are suggested in [31].

For smooth, nonconvex f and g, the emphasis in the literature has been on solving differentiable approximations of (4.1) using gradient-based techniques. Methods of this type fall into one of two main classes: discretization and reduction. These are briefly outlined below; the reader is directed to the review papers [32, 57] for more detailed discussions.

Discretization refers to a broad class of iterative methods in which the following finitely-constrained approximation of the SIP is solved at each iteration k,

$$\begin{aligned} &\min_{\boldsymbol{x}\in X} f(\boldsymbol{x}) \\ &\text{s.t. } g(\boldsymbol{x},\boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in S_k \end{aligned} \tag{4.8}$$

where the mesh S_k is a finite set of points in P, which may be defined either a priori, or adaptively. If the grid density of S_k (as defined in [57]) tends to zero, and the level set at each iteration is known to be compact, any accumulation point of the sequence of solutions $\boldsymbol{x}^k$ is known to be a 'solution' of the SIP [57]. The term 'solution' is used loosely to indicate that the algorithm converges to the same type of point for which each subproblem is solved, e.g., if the objective function $f(\boldsymbol{x})$ is minimized globally for each subproblem (as in (4.8)), then the overall procedure converges to a global minimum of the SIP. Similarly, if the subproblems are solved to local optimality then the overall method converges to a local minimum of the SIP.

A major drawback of discretization methods is that the size of the finite NLP solved in (4.8) grows with the increasing cardinality of S_k. For nonconvex problems, (4.8) must be solved using deterministic global optimization methods; this issue has been largely neglected in practical implementations and a KKT point of the SIP is generally considered an acceptable accumulation point for the discretization algorithm [57]. At each iteration k, some trade-off is made between the size of the grid used, and

the degree of infeasibility of the resulting iterate $\boldsymbol{x}^k$. A number of approaches aimed at reducing the cost of solving (4.8) have been suggested in the literature. Usually, $\boldsymbol{x}^k$ can be used as an initial guess for the solution to $\boldsymbol{x}^{k+1}$. Numerical algorithms have also explored the application of various NLP algorithms to (4.8) (superlinear successive quadratic programming (SQP) algorithms versus lower-order but cheaper cutting plane algorithms [42]), the feasibility of dropping constraints between iterations such that $S_k \not\subset S_{k+1}$, and the possibility of solving for stationary points, rather than local minima of (4.8). The other main disadvantage of discretization methods is that they generate only outer approximations to the SIP problem at finite termination. In general, the feasible set of a SIP cannot be represented using a finite number of constraints, e.g., in the case of Example 4.1 in [32]. Since $S_k \subset P$ at each iteration k, the feasible set of the discretized NLP and a corresponding global solution $\boldsymbol{x}^k$ satisfy the following relations:

$$\begin{gathered}\left\{\boldsymbol{x} \in X : g(\boldsymbol{x}, \boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in P\right\} \subset \left\{\boldsymbol{x} \in X : g(\boldsymbol{x}, \boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in S_k\right\} \\ f(\boldsymbol{x}^{SIP}) \geq f(\boldsymbol{x}^k)\end{gathered} \tag{4.9}$$

where $\boldsymbol{x}^{SIP}$ is a global minimum of the SIP. In summary, discretization methods converge from arbitrary starting points with relatively mild assumptions on the problem structure, but are generally expensive and guaranteed to yield only lower bounds to the true SIP solution. Moreover, the point $\boldsymbol{x}^k$ is not guaranteed to be feasible for the SIP. Furthermore, if the global solution is not found at each iteration k, no firm relationship can be established between $f(\boldsymbol{x}^k)$ and $f(\boldsymbol{x}^{SIP})$.

Local reduction methods are useful when an approximate solution in the proximity of a SIP KKT point is already known. Under suitable regularity assumptions (see [57] for a precise definition), the SIP-feasible region in a neighborhood of a KKT point, $U(\boldsymbol{x}^{SIP})$, may be defined by a finite number of implicitly-defined constraints:

$$g(\boldsymbol{x}, \boldsymbol{p}^v(\boldsymbol{x})) \leq 0, \quad v = 1, \ldots, r_v \tag{4.10}$$

where $\boldsymbol{p}^v(\boldsymbol{x})$ is a local maximizer of $g(\boldsymbol{x}, \boldsymbol{p})$ and the number of local maximizers r_v

remains constant for any $\boldsymbol{x} \in U(\boldsymbol{x}^{SIP})$. This implies that a local solution to the SIP may be found by solving:

$$\begin{aligned} &\min_{\boldsymbol{x}} f(\boldsymbol{x}) \\ \text{s.t. } &g(\boldsymbol{x}, \boldsymbol{p}^v(\boldsymbol{x})) \leq 0, \quad v = 1, \ldots, r_v \\ &\boldsymbol{x} \in U(\boldsymbol{x}^{SIP}) \cap X. \end{aligned} \tag{4.11}$$

The difficulty here is that neither the set of local maximizers $\{\boldsymbol{p}^v(\boldsymbol{x})\}$, nor the valid neighborhood $U(\boldsymbol{x}^{SIP})$, are known explicitly. The latter concern is simply ignored in most implementations. The former concern is addressed by solving a number of maximization subproblems locally to determine the unknown number of maximizers, r_v. The maximization subproblems are subsequently repeated and the set $\{\boldsymbol{p}^v(\boldsymbol{x})\}$ is updated at each iteration of a locally-convergent method. If $U(\boldsymbol{x}^{SIP})$ is within the convergence region of the NLP method, it can be shown that the method either terminates, or generates an infinite sequence, the accumulation points of which are KKT points of the SIP. Since the convergence of a local reduction approach relies on being in the vicinity of a local SIP solution, a suitable starting point must be generated using an alternate method, e.g., discretization. Alternatively, 'globalized' reduction algorithms may be used from arbitrary starting points by incorporating a penalty or merit function, and enforcing descent at each iteration of the NLP algorithm. Although globalized reduction methods have been implemented successfully for a number of problems in the literature [14, 55, 69, 30], the theoretical convergence of such methods makes strong assumptions on the properties of the iterates $\boldsymbol{x}^k$, e.g., reduction methods cannot be applied when $g(\boldsymbol{x}, \boldsymbol{p})$ is essentially flat over $\boldsymbol{p}$. From a practical standpoint, the accuracy and number of maximization subproblems that are solved has important consequences on the convergence rate and overall computational cost of the method [55]. The primary advantage of reduction-based methods is that they do not experience a blow-up in the number of finite constraints as the algorithm approaches the true SIP solution. Consequently, higher-order NLP methods can be applied to solve the finite approximations without the significant computational cost

incurred by discretization.

In this chapter a new class of methods for solving general nonlinear semi-infinite programs is introduced. A finite, inclusion-constrained reformulation of the SIP yields a finite NLP which is either equivalent or over-constrained relative to the original SIP, depending on the domain P and the functional form of the constraint $g(\boldsymbol{x}, \boldsymbol{p}) \leq 0$. When the finite reformulation is exact, it can be solved directly for the SIP global minimum; when the reformulation is inexact, it can be solved for a guaranteed SIP-feasible point. This feasible upper bound can then be refined using an iterative subdivision approach which converges to the true SIP solution.

The remainder of this chapter is organized as follows: a brief overview of interval analysis is provided to familiarize the reader with relevant notation and results; the inclusion-constrained reformulation (ICR) is presented and three different approaches for solving the derived NLP are proposed; finally, the application of the ICR method to literature problems is studied.

4.3 Review of Interval Techniques

The interval-constrained reformulation approach to be introduced in Section 4.4 draws heavily on well-known results from interval analysis. For the sake of completeness, relevant concepts and terminology from interval analysis are outlined below. The reader is directed to [3, 48] for a more thorough discussion.

An n_p-dimensional compact interval, P, bounded by the n_p-dimensional vectors $\boldsymbol{p}^l$ and $\boldsymbol{p}^u$, is defined by:

$$P = P_1 \times \ldots \times P_{n_p} = [\boldsymbol{p}^l, \boldsymbol{p}^u]. \tag{4.12}$$

The width of each dimension P_m and the overall width of P are defined to be:

$$\begin{aligned} w(P_m) &= p_m^u - p_m^l, \quad m = 1, \ldots, n_p \\ w(P) &= \max_m w(P_m). \end{aligned} \tag{4.13}$$

The range of values assumed by a real-valued, continuous function $g : P \rightarrow \mathbb{R}$ on the domain P is denoted by the scalar interval $\bar{g}(P)$:

$$\bar{g}(P) = [\bar{g}^l, \bar{g}^u] = \{g(\boldsymbol{p}) : \boldsymbol{p} \in P\}. \tag{4.14}$$

An interval function $G(P)$ which satisfies the following relation is referred to as an inclusion function for $g(\boldsymbol{p})$ on P:

$$\bar{g}(P) \subset G(P) = [G^l, G^u]. \tag{4.15}$$

The simplest inclusion function is derived by replacing each occurrence of $\boldsymbol{p}$ in $g(\boldsymbol{p})$ with the corresponding interval variable P, and applying the rules of interval arithmetic [48] to evaluate the resulting interval $G_{int}(P)$. $G_{int}(P)$ is referred to as the natural interval extension of g on P and can be expressed using only the bounds $\boldsymbol{p}^l,\ \boldsymbol{p}^u$ and selected constants (e.g., 1, -1, in the case of sine and cosine). For functions with special structure, e.g., rational functions in which each variable p_m appears only once, the natural interval extension yields an exact inclusion such that equivalence holds in (4.15). In general, the natural interval extension is inexact and overestimates the true range of a real-valued function. The tightness of any inclusion function may be quantified using the Hausdorff metric $H(\bar{g}, G)$, which is defined as follows for the scalar intervals $\bar{g}$ and G:

$$\begin{aligned} H(\bar{g}, G) = &\ \max(\left|\bar{g}^l - G^l\right|, |\bar{g}^u - G^u|) \\ = &\ \max(G^u - \bar{g}^u,\ \bar{g}^l - G^l). \end{aligned} \tag{4.16}$$

The width of the natural interval extension calculated for a given function $g(\boldsymbol{p})$ depends on the underlying expression used. In certain cases, tighter inclusions can be calculated by representing the function using a different underlying expression, i.e., by rearranging the expression before evaluating its natural interval extension. One such rearrangement, the small Horner scheme, is applied to univariate polynomial terms in Section 4.6. The polynomial terms in the constraint function are factored

as follows before the natural interval extension is evaluated:

$$\begin{gathered} g_s(p_m) = a_0 + a_1 p_m + \ldots + a_c p_m^c \\ g_{s,Horner}(p_m) = a_0 + p_m \left(a_1 + p_m \left(a_2 + p_m \left(a_3 + p_m \left(a_4 + \ldots + p_m \left(a_{c-1} + p_m a_c\right)\right)\right)\right)\right) \end{gathered} \tag{4.17}$$

where $g_s(p_m)$ is a subset of the terms in the constraint function $g(\boldsymbol{p})$. The natural interval extension of $g_{s,Horner}$ bounds $\bar{g}_s$ more tightly than $G_{s,int}$ such that:

$$\bar{g}_s(P_m) \subset G_{s,Horner}(P_m) \subset G_{s,int}(P_m). \tag{4.18}$$

It should be noted that when the interval P_m is symmetric, i.e., $p_m^l = -p_m^u$, it follows that $G_{s,Horner}(P_m) = G_{s,int}(P_m)$ and the Horner rearrangement has no effect on the width of the inclusion obtained. However, extended power evaluation may be helpful. Simple power evaluation on symmetric domains yields the following inclusion for an exponentiation operation:

$$\begin{aligned} g_s(p_m) &= p_m^c \\ G_{s,int}(P_m) &= [-(p_m^u)^c, (p_m^u)^c]. \end{aligned} \tag{4.19}$$

When c is a non-negative integer, extended power evaluation may be applied to calculate an exact inclusion $G_{s,E}$:

$$\begin{aligned} G_{s,E}(P_m) &= [0, (p_m^u)^c], \quad c \text{ even}, \\ G_{s,E}(P_m) &= [-(p_m^u)^c, (p_m^u)^c], \quad c \text{ odd}. \end{aligned} \tag{4.20}$$

A number of inclusion functions for functions with special structure have been presented in the literature [18, 40]. For general nonlinear functions, an exact inclusion over a nondegenerate domain ($\boldsymbol{p}^l \neq \boldsymbol{p}^u$) cannot be computed with finite computational effort. However, convergent inclusions of continuous functions [56, 3] can provide a

bound on the degree of overestimation incorporated by the inclusion:

$$H(\bar{g}, G) \leq \gamma w(P)^{\beta}$$
$$w(G(P)) \leq \delta w(P)^{\beta} \tag{4.21}$$

where $\beta \geq 1$ is the convergence order of the inclusion function, e.g., $\beta = 1$ for natural interval extensions, and $\delta, \gamma \geq 0$ are constants that depend on the form of the function g, and the interval P. This property suggests that progressively tighter inclusions may be calculated at the cost of more expensive function evaluations. Each dimension of the n_p-dimensional domain $P = [\boldsymbol{p}^l, \boldsymbol{p}^u]$ may be divided into n_k subintervals of equal width such that:

$$P_m^{\kappa} = [p_m^l + \frac{(\kappa - 1)w(P_m)}{n_k}, \quad p_m^l + \frac{\kappa \cdot w(P_m)}{n_k}], \quad \kappa = 1, \ldots, n_k. \tag{4.22}$$

Denoting $I_{n_k} = \{1, 2, \ldots, n_k\}^{n_p}$, it follows that

$$P = \bigcup_{\tau \in I_{n_k}} P_{\tau} \tag{4.23}$$

where $P_{\tau} = P_1^{\kappa_1} \times \ldots \times P_{n_p}^{\kappa_{n_p}}$ for $\tau = (\kappa_1, \kappa_2, \ldots, \kappa_{n_p}) \in I_{n_k}$. The range of the function over the interval P is then the union of the range of values assumed over each subinterval, i.e.:

$$\bar{g}(P) = \bigcup_{\tau \in I_{n_k}} \bar{g}(P_{\tau}). \tag{4.24}$$

Similarly, the union of the $n_k^{n_p}$ interval extensions $G(P_{\tau})$ yields a valid inclusion function for $g(\boldsymbol{p})$ on P, i.e.,

$$\bigcup_{\tau \in I_{n_k}} \bar{g}(P_{\tau}) \subset \bigcup_{\tau \in I_{n_k}} G(P_{\tau}) \subset G(P). \tag{4.25}$$

The inclusion $G_{n_k} = \bigcup_{\tau \in I_{n_k}} G(P_{\tau})$ is referred to as the n_k^{th} (uniform) refinement of G.

Applying (4.21), yields the following relations between G_{n_k} and $\bar{g}$:

$$\begin{aligned} H(\bar{g}, G_{n_k}) &\leq \gamma(w(P_\tau))^\beta &= \gamma\left(\frac{w(P)}{n_k}\right)^\beta \\ w(G_{n_k}) &\leq \delta(w(P_\tau))^\beta &= \delta\left(\frac{w(P)}{n_k}\right)^\beta. \end{aligned} \tag{4.26}$$

Thus the subdivision approach generates inclusion functions of arbitrary accuracy, at the cost of performing $n_k^{n_p}$ interval function evaluations. A number of modifications, e.g., Skelboe's principle, have been developed to alleviate the computational overhead associated with implementing uniform subdivisions [56]. Considerable effort has also been directed towards developing higher-order inclusions which converge more rapidly as the number of subdivisions increases.

In general, up to second-order convergence can be achieved for inclusions of real-valued functions [3] by using the centered form. Using the following underlying expression for a given function g:

$$g(p) = g(d) + g_s(p - d), \quad d \in P, \tag{4.27}$$

the centered form is calculated as the natural interval extension of the right-hand side of (4.27), and provides a quadratically-convergent inclusion for $g(\boldsymbol{p})$ on P. In particular, the n_T^{th} order Taylor expansion of a function of $n_T + 1^{th}$ order differentiability can be shown to satisfy the definition in (4.27) [56]. An inclusion for the n_T^{th} order Taylor expansion (polynomial + remainder term), henceforth referred to as a Taylor model, thus yields a quadratically-convergent inclusion for the function g on P, e.g., for a function of a scalar variable $g(p)$, the n_T^{th} order centered Taylor model calculated using natural interval extensions is given by:

$$\begin{aligned} G_{n_T}(P) &= g(p_d) + \frac{\partial g}{\partial p}\Big|_{p_d}(P - p_d) + \ldots + \frac{1}{n_T!}\frac{\partial^{n_T} g}{\partial p^{n_T}}\Big|_{p_d}(P - p_d)^{n_T} \\ &\quad + \frac{1}{(n_T+1)!}\frac{\partial^{n_T+1} g}{\partial p^{n_T+1}}\Big|_{P}(P - p_d)^{n_T+1} \end{aligned} \tag{4.28}$$

where $p_d = \frac{p^l + p^u}{2}$ is the midpoint of the interval P. Taylor models are particularly

useful because they can be generated automatically and efficiently for a fairly large class of functions [46, 49]. Assuming the n_T real-valued derivatives can be calculated correctly (e.g., using automatic differentiation [28]), the problem of obtaining an inclusion function for the original function is replaced with the problem of calculating an inclusion for the n_T^{th} order polynomial and the n_{T+1}^{th} order remainder term. The former may be evaluated using natural interval extensions or a specialized polynomial inclusion form, e.g., the Dussel-Schmitt method [18] (the Horner rearrangement is not helpful if the intervals $P - p_d$ are symmetric). The remainder inclusion may be calculated using interval arithmetic evaluations of the derivative and exponentiation terms. Alternatively, tighter remainder bounds may be calculated using remainder differential algebra [46].

It should be noted that the computational overhead of generating Taylor coefficients is justified only when subdivision is applied in calculating an inclusion. There is no general theory which dictates whether natural interval extensions or Taylor models will yield a tighter inclusion function for a particular function over a fixed domain (except, of course, when the natural interval extension is known to be exact). Clearly, tighter inclusions may be calculated using the Horner scheme or a specialized polynomial inclusion function to evaluate G_{n_T}. Alternatively, higher-order inclusion functions which do not require n_T^{th}-order differentiability of the function $g(\boldsymbol{p})$ may be used to calculate an inclusion. When implementing the ICR method, an appropriate inclusion function should be selected based on the form of the constraint $g(\boldsymbol{x}, \boldsymbol{p})$, the tightness of the resulting inclusion, and the cost associated with generating it. For example, when a feasible, rather than an optimal, solution to the SIP is required, a simple natural interval extension may suffice.

4.4 Interval-constrained Reformulation

In this section finite inner approximations of a general nonconvex SIP are derived using inclusion functions for the constraint $g(\boldsymbol{x}, \boldsymbol{p})$. As shown earlier, an exact reformulation of the SIP yields a difficult nonconvex, nonsmooth optimization problem.

Existing discretization and reduction-based methods generate smooth outer approximations which yield potentially infeasible lower bounds to the true SIP solution. In contrast, the interval-constrained reformulation introduced here generates a guaranteed feasible upper bound for (4.1). In certain cases, this feasible point is also the global minimum of the SIP. In other cases, a sequence of progressively larger (but finite) NLPs can be solved to approach the true SIP solution to within $\epsilon-$optimality.

4.4.1 Representation of SIP as a Finite NLP

As shown in Section 4.1, a lower bound to the SIP minimum may be derived by replacing the infinite set of constraints in (4.1) with a finite subset of those constraints, and solving the resulting problem to global optimality, i.e.,

$$\begin{aligned} &\min_{\boldsymbol{x}\in X} f(\boldsymbol{x}) \\ &\text{s.t. } g(\boldsymbol{x},\boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in T \end{aligned} \tag{4.29}$$

where $T \subset P,\ |T| < \infty$, e.g., $T = \{\boldsymbol{p}^v(\boldsymbol{x})\}$ for reduction-based methods, and $T = S_k$ for discretization methods. Inclusion functions may be used to generate a complementary upper-bounding problem for the SIP. Clearly, any upper-bounding problem for (4.6) also serves as an upper-bounding problem for (4.1). An upper-bounding problem may be derived by replacing the nondifferentiable constraint in (4.6) with one which has a smaller feasible set. Any valid inclusion function for $g(\boldsymbol{x},\boldsymbol{p})$ on P may be used to formulate such a constraint. Since $G^u(\boldsymbol{x},P) \geq \max\limits_{\boldsymbol{p}\in P} g(\boldsymbol{x},\boldsymbol{p}),\ \forall \boldsymbol{x} \in X$, it follows that

$$\left\{\boldsymbol{x} \in X : G^u(\boldsymbol{x},P) \leq 0\right\} \subset \left\{\boldsymbol{x} \in X : \max_{\boldsymbol{p}\in P} g(\boldsymbol{x},\boldsymbol{p}) \leq 0\right\} \tag{4.30}$$

where the inclusion bound G^u is some algebraic function of $\boldsymbol{x},\ \boldsymbol{p}^l$ and $\boldsymbol{p}^u$. This relation suggests the following finitely-constrained, upper-bounding problem for (4.6), which

is also a valid upper-bounding problem for the SIP:

$$\begin{aligned} &\min_{\boldsymbol{x}\in X} f(\boldsymbol{x}) \\ &G^u(\boldsymbol{x}, P) \leq 0. \end{aligned} \tag{4.31}$$

In the general case, solving (4.31) to local optimality yields a guaranteed feasible point for the SIP, provided a solution to (4.31) exists. Note that the global solution of (4.31) is not required to guarantee the feasibility property. When $G^u(\boldsymbol{x}, P) = \max_{\boldsymbol{p}\in P} g(\boldsymbol{x}, \boldsymbol{p}),\ \forall \boldsymbol{x} \in X$, problem (4.31) is an exact, finite reformulation of the SIP, such that solving (4.31) to global optimality yields the true solution of the SIP. In this case the inclusion bound yields an explicit expression for the parametric solution of $\max_{\boldsymbol{p}\in P} g(\boldsymbol{x}, \boldsymbol{p}),\ \forall \boldsymbol{x} \in X$. This case arises for, but is not limited to, the case where $G(\boldsymbol{x}, P)$ is known to be an exact inclusion for $g(\boldsymbol{x}, \boldsymbol{p})$ on P. More commonly, an exact reformulation is derived using an inexact inclusion, but $G^u(\boldsymbol{x}, P) = \max_{\boldsymbol{p}\in P} g(\boldsymbol{x}, \boldsymbol{p}),\ \forall \boldsymbol{x} \in X$ while $G^l(\boldsymbol{x}, P) < \min_{\boldsymbol{p}\in P} g(\boldsymbol{x}, \boldsymbol{p})$, for some or all $\boldsymbol{x} \in X$. In many respects, SIPs that are amenable to this exact, finite reformulation form a simpler class of problems than those for which explicit expressions for the parametric solution of the inner optimization problem cannot be derived.

The numerical results presented in Section 4.6 were obtained using natural interval extensions and Taylor models of second order ($n_T = 2$) as inclusion functions. The following procedure was used to define and reformulate natural interval extensions, both of the original constraint functions, and their Taylor expansions. The $p-$dependent and $x-$dependent terms are first isolated from the remaining mixed terms in g such that:

$$\begin{aligned} g(\boldsymbol{x}, \boldsymbol{p}) &= g_x(\boldsymbol{x}) + g_p(\boldsymbol{p}) + g_{xp}(\boldsymbol{p}, \boldsymbol{x}) \\ G^u(\boldsymbol{x}, P) &= g_x(\boldsymbol{x}) + G^u_p(P) + G^u_{xp}(\boldsymbol{x}, P). \end{aligned} \tag{4.32}$$

The term g_x does not contain any interval variables and thus does not need to be treated further. The inclusion bound G^u_p is calculated by applying the rules of interval arithmetic evaluation to g_p (after applying the Horner rearrangement to any polyno-

mial terms in g_p). The term g_{xp} is addressed similarly except that the optimization variables $\boldsymbol{x}$ are treated as degenerate interval variables and add an extra degree of complexity to evaluating the inclusion G^u_{xp}.

Any terms which are polynomial in p_m are first subjected to the Horner rearrangement. The resulting rearrangement of g_{xp} is assumed to be factorable using the McCormick scheme [47]. A factorable sequence $p_1, p_2, \ldots, p_N$ is defined for g_{xp} such that $p_N = g_{xp}$ and the first n_p factors correspond to the n_p elements of the vector $\boldsymbol{p}$. The remaining factors $p_n,\ n = n_p + 1, \ldots, N$ are expressed as products, sums, or univariate functions of previously-defined factors, i.e., each such factor is defined using one of the following compositions:

$$\begin{aligned} p_n &= p_{n_1} + p_{n_2} \\ p_n &= p_{n_1} \cdot p_{n_2} \\ p_n &= g_n\left(p_{n_1}\right) \end{aligned} \tag{4.33}$$

where $n_1,\ n_2 < n$. Note that the optimization variables $\boldsymbol{x}$ are simply treated as constants or coefficients in this assignment process. In the original reference [47], the third type of composition considered in (4.33) is any univariate function of previously-defined factors. Here a more restrictive condition is applied, and only elementary function compositions of previously-defined factors are admitted. The term 'elementary function' is used to refer to any function of p_{n_1} and $\boldsymbol{x}$ for which each inclusion bound can be defined using no more than one binary min or max function,[1] which includes exponential, exponentiation, logarithmic, trigonometric, affine and reciprocal functions. As noted in [47] the McCormick factorization scheme is not uniquely defined. Clearly, any Horner rearrangements of the underlying expression should be preserved while assigning compositions. Once the factors $p_{n_p+1}, \ldots, p_N$ have been defined, inclusion bounds $p^u_n,\ p^l_n$ are calculated for each of these factors using the rules of interval arithmetic evaluation. Factors which are defined using addition compositions are most easily dealt with. Following the notation in (4.33), the inclusion

[1] A more general definition is possible by admitting any univariate function for which an inclusion bound can be calculated as a function of a finite number of points. However, this generalized definition makes it more complicated to quantify the size of the reformulation.

bounds for such a factor are simply:

$$\begin{aligned} p_n^u &= p_{n_1}^u + p_{n_2}^u \\ p_n^l &= p_{n_1}^l + p_{n_2}^l. \end{aligned} \tag{4.34}$$

The inclusion bounds for a product composition are more complicated and entail explicit min/max functions:

$$\begin{aligned} p_n^l &= \min(p_{n_1}^l \cdot p_{n_2}^l,\ p_{n_1}^u \cdot p_{n_2}^u, p_{n_1}^u \cdot p_{n_2}^l,\ p_{n_1}^l \cdot p_{n_2}^u) \\ p_n^u &= \max(p_{n_1}^l \cdot p_{n_2}^l,\ p_{n_1}^u \cdot p_{n_2}^u,\ p_{n_1}^u \cdot p_{n_2}^l,\ p_{n_1}^l \cdot p_{n_2}^u). \end{aligned} \tag{4.35}$$

Finally, the inclusion bounds for an elementary function composition may be directly identifiable, or they may need to be defined using explicit min/max functions. In the general case, compositions of monotonic univariate functions, (e.g., exponential, affine, logarithmic and reciprocal functions over valid domains) fall into the latter category. Inclusion bounds for such compositions are defined as follows:

$$\begin{aligned} p_n^l &= \min\left(g_n(\boldsymbol{x},\ p_{n_1}^u),\ g_n(\boldsymbol{x},\ p_{n_1}^l)\right) \\ p_n^u &= \max\left(g_n(\boldsymbol{x},\ p_{n_1}^u),\ g_n(\boldsymbol{x},\ p_{n_1}^l)\right). \end{aligned} \tag{4.36}$$

For exponentiation functions of interval variables raised to even powers, extended power evaluation may be exploited to calculate tighter inclusion bounds on symmetric intervals, i.e.,

$$\begin{aligned} p_n^l &= \min\Big(g_n(\boldsymbol{x}, p_{n_1}^u),\ g_n(\boldsymbol{x}, 0)\Big) \\ p_n^u &= \max\Big(g_n(\boldsymbol{x}, p_{n_1}^u),\ g_n(\boldsymbol{x}, 0)\Big). \end{aligned} \tag{4.37}$$

In particular instances, the form of the elementary function g_n may obviate explicit min/max functions, e.g., when $g_n(\boldsymbol{x}, p_{n_1}) = x_1^2 \cdot e^p$, the inclusion bounds are simply:

$$\begin{aligned} p_n^l &= x_1^2 \cdot e^{p^l} \\ p_n^u &= x_1^2 \cdot e^{p^u}. \end{aligned} \tag{4.38}$$

Furthermore, either case may arise for the same elementary function composition depending on the domain over which the interval extension is to be evaluated. This

is easily illustrated using a composition involving a trigonometric function. Over a domain $w(P_m) \geq 2\pi$, a sine function assumes maximum and minimum values of 1 and -1 respectively. Thus an elementary function of the form $x_1^2 \cdot \sin(p)$ assumes values in the range $[-x_1^2, x_1^2]$ over such a domain. However, when P is further restricted, e.g., when $p_{n_1}^l \in [0, \frac{\pi}{2}],\ p_{n_1}^u \in [\frac{\pi}{2}, \pi]$, the inclusion bounds are given by:

$$\begin{aligned} p_n^u &= x_1^2 \\ p_n^l &= x_1^2 \cdot \min(\sin(p_{n_1}^l),\ \sin(p_{n_1}^u)). \end{aligned} \tag{4.39}$$

Note that in all of these cases no more than one min/max function is required to define each inclusion bound, consistent with the definition of an elementary function composition.

Once inclusion bounds have been defined for all of the factors $\{p_{n_p+1}, \ldots, p_N\}$, p_N^u is substituted for G_{xp}^u in (4.32) to arrive at a finitely-constrained representation of the SIP. This procedure is illustrated below using Example 2 from the Watson test set [78]:

$$\begin{aligned} &\min_{\boldsymbol{x} \in X} \tfrac{1}{3}x_1^2 + x_2^2 + \tfrac{1}{2}x_1 \\ &\left(1 - x_1^2 p^2\right)^2 - x_1 p^2 - x_2^2 + x_2 \leq 0 \quad \forall p \in [0, 1]. \end{aligned} \tag{4.40}$$

Following a Horner rearrangement the following expressions for g_p, g_x and g_{xp} are obtained:

$$\begin{aligned} g_p &= 0 \\ g_x &= 1 + x_2 - x_2^2 \\ g_{xp} &= p^2 \left(-x_1 - 2x_1^2 + x_1^4 \cdot p^2\right). \end{aligned} \tag{4.41}$$

The inclusion bound G_p^u is simply the degenerate interval 0 in this case. The McCormick factorization scheme is now applied to g_{xp}:

$$\begin{aligned} p_1 &= p \\ p_2 &= p_1^2 \\ p_3 &= -x_1 - 2x_1^2 + x_1^4 \cdot p_2 \\ p_4 &= p_2 \cdot p_3 \end{aligned} \tag{4.42}$$

where p_1 is simply the original interval variable, p_2 and p_3 are defined by elementary function compositions, and p_4 is defined by a product composition (clearly p_2 may also be treated as a product term but this results in a more complex expression for the inclusion bound). Upon defining inclusion bounds for each of the factors $p_2, \ldots, p_4$ and substituting $p_4^u = G_{xp}^u$ in (4.32) the following finite reformulation of (4.40) is obtained:

$$\begin{aligned}
\min_{\boldsymbol{x}\in X, \boldsymbol{r}^l\in R^l, \boldsymbol{r}^u\in R^u} \ & \tfrac{1}{3}x_1^2 + x_2^2 + \tfrac{1}{2}x_1 \\
p_2^l &= (p_1^l)^2 \\
p_2^u &= (p_1^u)^2 \\
p_3^l &= -x_1 - 2x_1^2 + x_1^4 \cdot p_2^l \\
p_3^u &= -x_1 - 2x_1^2 + x_1^4 \cdot p_2^u \\
p_4^u &= \max\left(p_2^u \cdot p_3^u,\ p_2^l \cdot p_3^u,\ p_2^l \cdot p_3^l,\ p_2^u \cdot p_3^l\right) \\
1 + x_2 - x_2^2 + p_4^u &\leq 0 \\
p_1^l = 0,\ p_1^u &= 1
\end{aligned} \tag{4.43}$$

where $\boldsymbol{r}$ is the vector of new variables introduced during the factorization process, i.e., $\boldsymbol{r} = (p_{n_p+1}, \ldots, p_N)$. In the original reference the set X was unrestricted and taken to be $X = \mathbb{R}^{n_x}$. However, global NLP and MINLP solvers require a compact feasible set to be specified for all of the optimization variables. For the numerical results presented here, $X = [-1000, 1000]^{n_x}$ was used as the default feasible set unless otherwise noted. Since the new variables $\boldsymbol{r}^u,\ \boldsymbol{r}^l$ are defined by explicit equations in our reformulation, interval analysis methods can also be applied to propagate the set X to suitable compact sets for $\boldsymbol{r}^u$ and $\boldsymbol{r}^l$. This procedure is performed automatically by the BARON solver, and R^u and R^l were not specified explicitly in our solution procedure.

The max function in (4.43) results in a nonsmooth problem (which is nonetheless much cheaper to solve than the min-max program in (4.6)), and can be solved for a guaranteed SIP-feasible point. The problem may be tackled without further treatment using a local nonsmooth optimizer. Multiple starting points may be selected heuristically in an attempt to ensure global optimality of the solution. Alternatively the nonsmooth problem may be reformulated either as a smooth NLP, or as a MINLP

with smooth relaxations. These equivalent reformulations allow the finite approximation to be solved to guaranteed global optimality using gradient-based algorithms. It should be noted that the differentiability assumptions made in Section 4.1 serve only to guarantee the applicability of gradient-based methods to solving a reformulation of (4.31). In the absence of such properties, the problem in (4.31) may still be solved locally using a nonsmooth optimizer.

The above discussion pertains to the calculation of natural interval extensions for general constraint functions. Clearly these rules also apply when Taylor models are generated using the natural interval extension of the Taylor expansion of g_{xp}. However, the form of the Taylor model requires that inclusions be calculated only for polynomial terms, and for derivatives of order n_T+1. Inclusion bounds for individual terms in the polynomial series are calculated using extended power evaluation, and subsequently summed together as in (4.34). Neither of these operations introduce explicit min/max functions into the reformulated problem. If the inclusion bounds for the $n_T + 1^{th}$ derivatives, and the products between the derivative inclusions and the $n_T + 1^{th}$ order polynomial terms can also be expressed without explicit min/max functions, the application of the Taylor model yields a smooth, inclusion-constrained reformulation which is directly solvable using a global NLP solver. Clearly, the overall degree of nonsmoothness in the Taylor model depends on the manner in which the polynomial and derivative inclusions are calculated, e.g., natural interval extensions versus dedicated polynomial inclusions and direct derivative interval evaluation versus remainder differential algebra. The preservation of smoothness in the overall formulation is just one consideration, and in many cases the cost and tightness of the interval evaluation may justify nonsmooth expressions for the inclusion bounds.

4.4.2 Reformulation of the Nonsmooth NLP

Reformulation as MINLP

Unlike the problem in (4.43), a mixed-integer nonlinear program (MINLP) with smooth relaxations can be solved to guaranteed global optimality in order to identify

a rigorous, and potentially tighter, upper bound to the SIP solution. An equivalent MINLP is derived by introducing binary variables to reformulate any max or min functions which appear in the expressions for the inclusion bounds $p_n^u,\ p_n^l$. Since inclusion bounds for addition compositions do not entail explicit min/max functions, they require no further reformulation. Inclusion bounds for those elementary function compositions which require explicit min/max operations may be reformulated as follows:

$$\begin{gathered} p_n^u = y_1 \cdot g_n(p_{n_1}^u, \boldsymbol{x}) + (1 - y_1) \cdot g_n(p_{n_1}^l, \boldsymbol{x}) \\ p_n^l = y_1 \cdot g_n(p_{n_1}^l, \boldsymbol{x}) + (1 - y_1) \cdot g_n(p_{n_1}^u, \boldsymbol{x}) \\ 0 \leq y_1 \cdot (g_n(p_{n_1}^u, \boldsymbol{x}) - g_n(p_{n_1}^l, \boldsymbol{x})) + (1 - y_1) \cdot (g_n(p_{n_1}^l, \boldsymbol{x}) - g_n(p_{n_1}^u, \boldsymbol{x})) \\ y_1 \in \{0, 1\}. \end{gathered} \tag{4.44}$$

The inequality in (4.44) forces y_1 to assume a value (0 or 1) such that the greater of the two values $g_n(\boldsymbol{x}, p_{n_1}^u)$ or $g_n(\boldsymbol{x}, p_{n_1}^l)$ is assigned to p_n^u in the first equation, and correspondingly the lesser value is assigned to p_n^l in the second equation. A product composition entails the introduction of four binary variables (or three if only p_n^u is needed). For a given factor $p_n = p_{n_1} \cdot p_{n_2}$, the inclusion bounds are defined as follows:

$$\begin{gathered} p_n^{u1} = y_1 \cdot p_{n_1}^l p_{n_2}^l + (1 - y_1) \cdot p_{n_1}^l p_{n_2}^u \\ p_n^{l1} = (1 - y_1) \cdot p_{n_1}^l p_{n_2} + y_1 \cdot p_{n_1}^l p_{n_2}^u \\ 0 \leq y_1 \cdot (p_{n_1}^l p_{n_2}^l - p_{n_1}^l p_{n_2}^u) + (1 - y_1) \cdot (p_{n_1}^l p_{n_2}^u - p_{n_1}^l p_{n_2}^l) \\ p_n^{u2} = y_2 \cdot p_{n_1}^u p_{n_2}^l + (1 - y_2) \cdot p_{n_1}^u p_{n_2}^u \\ p_n^{l2} = (1 - y_2) \cdot p_{n_1}^u p_{n_2}^l + y_2 \cdot p_{n_1}^u p_{n_2}^u \\ 0 \leq y_2 \cdot (p_{n_1}^u p_{n_2}^l - p_{n_1}^u p_{n_2}^u) + (1 - y_2) \cdot (p_{n_1}^u p_{n_2}^u - p_{n_1}^u p_{n_2}^l) \\ p_n^u = y_3 \cdot p_{n_1}^{u1} + (1 - y_3) \cdot p_{n_1}^{u2} \\ 0 \leq y_3 \cdot (p_{n_1}^{u1} - p_{n_1}^{u2}) + (1 - y_3) \cdot (p_{n_1}^{u2} - p_{n_1}^{u1}) \\ p_n^l = y_4 \cdot p_{n_1}^{l1} + (1 - y_4) \cdot p_{n_1}^{l2} \\ 0 \geq y_4 \cdot (p_{n_1}^{l1} - p_{n_1}^{l2}) + (1 - y_4) \cdot (p_{n_1}^{l2} - p_{n_1}^{l1}) \\ y_1,\ y_2,\ y_3,\ y_4 \in \{0, 1\}. \end{gathered} \tag{4.45}$$

Again, the inequality corresponding to each binary variable forces the larger of the two values being compared to be assigned to the upper inclusion bound, and the lesser value to be assigned to the lower bound.

Once all of the expressions $p_n^u,\ p_n^l$ have been reformulated to eliminate the min/max functions, p_N^u is substituted for G_{xp}^u in (4.31) to arrive at an MINLP reformulation of the SIP. When applied to g_{xp}, the reformulation described here generates a nonconvex MINLP with between $3n_c + n_e$ and $4n_c + n_e$ binary variables, where n_c is the number of product compositions, and n_e is the number of elementary function compositions whose inclusion bounds incorporate explicit min/max functions, i.e., the size of the reformulated problem is polynomial in the number of McCormick factors used to define g_{xp}.

The MINLP reformulation was applied to (4.43) to yield the following equivalent representation:

$$\begin{aligned}
&\min_{\boldsymbol{x}\in X, \boldsymbol{r}^u\in R^u, \boldsymbol{r}^l\in R^b, \boldsymbol{y}} \tfrac{1}{3}x_1^2 + \tfrac{1}{2}x_1 + x_2^2 \\
&1 + x_2 - x_2^2 + p_4^u \le 0 \\
&p_2^u = (p_1^u)^2 \\
&p_2^l = (p_1^l)^2 \\
&p_3^l = -x_1 - 2x_1^2 + x_1^4 \cdot p_2^l \\
&p_3^u = -x_1 - 2x_1^2 + x_1^4 \cdot p_2^u \\
&p_4^{u1} = y_1 \cdot p_2^l p_3^l + (1 - y_1) \cdot p_2^l p_3^u \\
&0 \le y_1 \cdot (p_2^l p_3^l - p_2^l p_3^u) + (1 - y_1) \cdot (p_2^l p_3^u - p_2^l p_3^l) \\
&p_4^{u2} = y_2 \cdot p_2^u p_3^l + (1 - y_2) \cdot p_2^u p_3^u \\
&0 \le y_2 \cdot (p_2^u p_3^l - p_2^u p_3^u) + (1 - y_2) \cdot (p_2^u p_3^u - p_2^u p_3^l) \\
&p_4^u = y_3 \cdot p_4^{u1} + (1 - y_3) \cdot p_4^{u2} \\
&0 \le y_3 \cdot (p_4^{u1} - p_4^{u2}) + (1 - y_3) \cdot (p_4^{u2} - p_4^{u1}) \\
&\boldsymbol{y} \in \{0,1\}^3,\ p_1^l = 0,\ p_1^u = 1.
\end{aligned} \tag{4.46}$$

The nonconvex problem in (4.46) has smooth relaxations and was solved to global optimality using the branch-and-reduce algorithm implemented by the GAMS BARON

solver [58, 59]. A solution of $f(\boldsymbol{x}^*) = 0.195,\ \boldsymbol{x}^* = (-0.75,\ -0.62)$ was obtained. This result agreed with the (lower-bounding) reduction-based solution reported in [78], and was thus confirmed to be a global minimum of the SIP.

Reformulation as Smooth NLP

The direction of the inequality constraint in (4.31) suggests that a smooth NLP formulation may be derived by explicitly enumerating the constraints implied by the max/min functions which define $p_n^u,\ p_n^l$. The NLP reformulation does not lend itself to the introduction of new variables $p_n^l,\ p_n^u$ since their exact reformulations using differentiable equations and inequalities violate constraint qualifications (see discussion below). The elementary function compositions are identified but not assigned to new variables. g_{xp} is then represented as recursive sums and products of the original variables $p_1, \ldots, p_{n_p}$ and any elementary functions present, e.g., for the problem in (4.40), g_{xp} may be expressed as a product of two elementary functions, p^2 and $(-x_1 - 2x_1^2 + x_1^4 \cdot p^2)$. A smooth reformulation of (4.43) is then derived by enumerating the inequalities implied by the arguments of the max/min functions which appear in the expression for p_4^u. Upon substituting expressions for $p_3^u,\ p_3^l,\ p_2^u$ and p_2^u directly into the inequality constraint, the following reformulation of (4.43) is obtained:

$$\begin{aligned} & \min_{\boldsymbol{x} \in X} \tfrac{1}{3}x_1^2 + x_2^2 + \tfrac{1}{2}x_1 \\ 1 + x_2 - x_2^2 + \quad & \max((p^u)^2 \cdot g_3(x_1, (p^u)^2), (p^l)^2 \cdot g_3(x_1, (p^u)^2), \\ & (p^l)^2 \cdot g_3(x_1, (p^l)^2),\ (p^u)^2 \cdot g_3(x_1, (p^l)^2)) \leq 0 \\ & p^l = 0,\ p^u = 1 \end{aligned} \tag{4.47}$$

where $g_3(x_1,\ p) = -x_1 - x_1^2 + x_1^4 \cdot p^2$. Applying the NLP reformulation described above yields the following equivalent representation:

$$\begin{aligned}
&\min_{\boldsymbol{x}\in X} \tfrac{1}{3}x_1^2 + x_2^2 + \tfrac{1}{2}x_1 \\
&1 + x_2 - x_2^2 + (p^u)^2 \cdot g_3(x_1,\ (p^u)^2) \leq 0 \\
&1 + x_2 - x_2^2 + (p^l)^2 \cdot g_3(x_1,\ (p^u)^2) \leq 0 \\
&1 + x_2 - x_2^2 + (p^l)^2 \cdot g_3(x_1,\ (p^l)^2) \leq 0 \\
&1 + x_2 - x_2^2 + (p^u)^2 \cdot g_3(x_1,\ (p^l)^2) \leq 0 \\
&p^l = 0,\ p^u = 1.
\end{aligned} \tag{4.48}$$

The nonconvex NLP in (4.48) was solved to global optimality using BARON. Since the MINLP and NLP formulations are equivalent, the MINLP solution was recovered. In this case the intermediate bounds $p_2^u,\ p_2^l$ can be calculated directly. In the general case, additional min/max operators are required to defined the inclusion bounds for elementary functions. In such cases, these operators are treated similarly to those arising from the multiplication operation in (4.43) and further increase the number of inequalities in the smooth reformulation, such that a total of $2^{2n_c+n_e}$ differentiable inequalities are necessary to reformulate an overall inclusion bound G_{xp}^u.

Note that it is possible to avoid this exponential growth in the number of constraints and still derive a smooth NLP formulation. This is illustrated for the term p_4^u in Example 2 above. A conventional bilevel programming formulation suggests that the term p_4^u may be redefined as follows:

$$\begin{aligned}
p_4^u = \arg\min_{z\in\mathbb{R}}\ & z \\
\text{s.t. } & z \geq p_3^u \cdot p_2^u \\
& z \geq p_3^u \cdot p_2^l \\
& z \geq p_3^l \cdot p_2^l \\
& z \geq p_3^l \cdot p_2^u.
\end{aligned} \tag{4.49}$$

The fifth constraint in (4.43) may then be replaced by the following set of constraints which arise from the application of the equivalent KKT optimality conditions to

(4.49):

$$
\begin{aligned}
1 + \lambda_1(-1) + \lambda_2(-1) + \lambda_3(-1) + \lambda_4(-1) &= 0 \\
\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4 &\geq 0 \\
\lambda_1(p_3^u \cdot p_2^u - p_4^u) &= 0 \\
\lambda_2(p_3^u \cdot p_2^l - p_4^u) &= 0 \\
\lambda_3(p_3^l \cdot p_2^l - p_4^u) &= 0 \\
\lambda_4(p_3^l \cdot p_2^l - p_4^u) &= 0 \\
p_4^u &\geq p_3^u \cdot p_2^u \\
p_4^u &\geq p_3^u \cdot p_2^l \\
p_4^u &\geq p_3^l \cdot p_2^l \\
p_4^u &\geq p_3^l \cdot p_2^u.
\end{aligned}
\tag{4.50}
$$

The resulting problem is a smooth, finitely-constrained NLP. As noted in [29], this reformulation is valid only when the KKT conditions for the inner problem are necessary and sufficient. Here, this requirement is always satisfied because of the linearity of the inner problem (4.49). However, the absence of an interior feasible point for (4.50) indicates that the outer problem does not in fact satisfy any constraint qualifications. As a result this reformulation cannot be solved using most local and global optimization algorithms.

When $G^u(\boldsymbol{x}, P) = \max_{\boldsymbol{p} \in P} g(\boldsymbol{x}, \boldsymbol{p}),\ \forall \boldsymbol{x} \in X$, e.g., when an exact inclusion function for the constraint g is known, the interval-constrained approach yields an equivalent, finite reformulation which may be solved for the global minimum of the SIP. When an exact inclusion for g is not known, the challenge lies in determining whether the resulting reformulation yields a globally optimal, or simply a guaranteed feasible solution to the SIP. We address this issue in detail in the following chapter, by generating lower bounds to complement the upper-bounding values generated by the ICR method.

An intermediate approach is to verify the tightness of the inclusion upper bound G^u at the solution $\boldsymbol{x}^{ICR}$. The following maximization problem is solved for $\bar{g}^u(\boldsymbol{x}^{ICR}, P)$:

$$
\max_{\boldsymbol{p} \in P} g(\boldsymbol{x}^{ICR}, \boldsymbol{p}) \tag{4.51}
$$

using a deterministic global optimization algorithm. A solution for which $\bar{g}^u(\boldsymbol{x}^{ICR}, P)$ is strictly less than $G^u(\boldsymbol{x}^{ICR}, P)$ indicates that the inclusion bound G^u is inexact, and that solving (4.31) globally is likely to yield only a rigorous upper bound to the true SIP solution. In such cases, the quality of this upper bound may be improved by using tighter inclusion functions to derive better approximations to the SIP constraints. If, on the other hand, $\bar{g}^u = G^u$, it cannot be concluded that the global minimum has been found.

4.4.3 Application of the Subdivision Method

Arbitrarily tight inclusions for continuous functions can be generated by applying the subdivision principle described in Section 4.3. Following the notation of Section 4.3, the domain of the SIP constraints, P, may be subdivided uniformly into n_k subintervals at each iteration, k. The convergence property of inclusion functions (4.21) suggests that a tighter upper bound for $g(\boldsymbol{x}, \boldsymbol{p})$ may be calculated by evaluating $n_k^{n_p}$ inclusions, i.e.,

$$\max_{\boldsymbol{p} \in P} g(\boldsymbol{x}, \boldsymbol{p}) \leq G^u_{n_k} = \max_{\tau \in I_{n_k}} G^u_\tau(\boldsymbol{x}, P_\tau) \leq G^u(\boldsymbol{x}, P). \tag{4.52}$$

This implies that the feasible set of the SIP is better approximated using $G^u_{n_k}$:

$$\begin{aligned} \Big\{\boldsymbol{x} \in X : G^u(\boldsymbol{x}, P) \leq 0\Big\} &\subset \Big\{\boldsymbol{x} \in X : G^u_{n_k}(\boldsymbol{x}, P) \leq 0\Big\} \\ &= \Big\{\boldsymbol{x} \in X : G^u_\tau(x, P_\tau) \leq 0, \quad \forall \tau \in I_{n_k}\Big\} \\ &\subset \Big\{\boldsymbol{x} \in X : \max_{\boldsymbol{p} \in P} g(\boldsymbol{x}, \boldsymbol{p}) \leq 0\Big\}. \end{aligned} \tag{4.53}$$

A tighter upper-bounding problem for the SIP is derived by replacing the single constraint in (4.31) with the $n_k^{n_p}$ constraints used to define the feasible set $\{\boldsymbol{x} \in X : G^u_{n_k}(\boldsymbol{x}, P) \leq 0\}$:

$$\begin{aligned} &\min_{\boldsymbol{x} \in X} f(\boldsymbol{x}) \\ &G^u_\tau(\boldsymbol{x}, P_\tau) \leq 0 \quad \forall \tau \in I_{n_k}. \end{aligned} \tag{4.54}$$

It should be noted that the constraints in the new problem are not necessarily a subset of those in the problem before subdivision. For a sequence of strictly increasing integers n_k, as $k \to \infty$, $\{\boldsymbol{x} \in X : G^u_{n_k} \leq 0\} \to \{\boldsymbol{x} \in X : \max_{\boldsymbol{p} \in P} g(\boldsymbol{x}, \boldsymbol{p}) \leq 0\}$ and provided a feasible interior point for the SIP exists, it follows that $\lim_{k\to\infty} f^{ICR}_{n_k} = f^{SIP}$ where $f^{ICR}_{n_k} = \min_{\boldsymbol{x} \in X} f(\boldsymbol{x})$ s.t. $G^u_{n_k} \leq 0$ and $f^{SIP} = \min_{\boldsymbol{x} \in X} f(\boldsymbol{x})$ s.t. $\bar{g}^u(\boldsymbol{x}, P) \leq 0$.

Theorem 1. *When the constraint domain P is subdivided uniformly at each iteration such that $w(P_\tau) = \frac{w(P)}{n_k}$, an inclusion function of convergence order $\beta \geq 1$ yields an exact reformulation of the SIP in the limit $k \to \infty$, such that $\lim_{k\to\infty} f^{ICR}_{n_k} = f^{SIP}$, provided a feasible interior point for the SIP exists, and $n_{k+1} > n_k$ for all $k \geq 1$.*

Proof. The convergence property of the inclusion function G guarantees that $\forall \boldsymbol{x} \in X, \quad \exists \gamma(\boldsymbol{x}) \geq 0$ and bounded such that $G^u_{n_k}(\boldsymbol{x}, P) - \bar{g}^u(\boldsymbol{x}, P) \leq \gamma(\boldsymbol{x}) w(P_\tau)^\beta$. The compactness of the set X then implies that there exists a bounded $\gamma^* \geq \gamma(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in X$ such that $G^u_{n_k}(\boldsymbol{x}, P) - \bar{g}^u(\boldsymbol{x}, P) \leq \gamma^* w(P_\tau)^\beta, \quad \forall \boldsymbol{x} \in X$. When subdivision is applied uniformly, such that $\lim_{k\to\infty} w(P_\tau) = \lim_{k\to\infty} \frac{w(P)}{n_k} = 0$, the elementary criterion for uniform convergence guarantees that the sequence of functions $G^u_{n_k}(\cdot, P)$ converges uniformly to $\bar{g}^u(\cdot, P)$ on X. Since $g^u(\boldsymbol{x}, P)$ is assumed to be strictly less than zero for at least one point in the SIP-feasible set, uniform convergence implies that there exists $\boldsymbol{x} \in X$ such that $G^u_{n_k}(\boldsymbol{x}, P) \leq 0, \quad \forall k \geq k^*$ for some finite $k^* \geq 1$, i.e., $\{\boldsymbol{x} \in X : G^u_{n_k}(\boldsymbol{x}, P) \leq 0\}$ is a non-empty, compact set for all $k \geq k^*$. It follows that the sequence of values obtained by solving (4.54) is a bounded, monotonic (non-increasing) sequence such that $f^{ICR}_{n_k} \geq f^{ICR}_{n_{k+1}} \geq f^{SIP}$ for all $k \geq k^*$. This establishes the existence of the limit $\lim_{k\to\infty} f^{ICR}_{n_k}$.

The application of subdivision generates optimization problems with increasingly larger constraint sets such that $\{\boldsymbol{x} \in X : G^u_{n_k}(\boldsymbol{x}, P) \leq 0\} \subset \{\boldsymbol{x} \in X : G^u_{n_{k+1}}(\boldsymbol{x}, P) \leq 0\} \subset \{\boldsymbol{x} \in X : g^u(\boldsymbol{x}, P) \leq 0\}, \forall k \geq 1$. By definition, $\{\boldsymbol{x} \in X : g^u(\boldsymbol{x}, P) \leq 0\}$ is the superior limit set of the increasing sequence, $\{\boldsymbol{x} \in X : G^u_{n_k}(\boldsymbol{x}, P) \leq 0\}$. Since an increasing sequence of sets is known to converge, the superior limit set is also the principal limit set to which the sequence $\{\boldsymbol{x} \in X : G^u_{n_k} \leq 0\}$ converges.

The sequence of minimizers $\{\boldsymbol{x}^{ICR}_{n_k}\}, \quad k \geq k^*$ identified by solving (4.54), is a

bounded sequence in the compact set X. Thus $\{\boldsymbol{x}_{n_k}^{ICR}\}$ has a convergent subsequence having an accumulation point in the limit set $\{\boldsymbol{x} : g^u(\boldsymbol{x}, P) \leq 0\}$, i.e., $\lim \boldsymbol{x}_{n'_k}^{ICR} = \bar{\boldsymbol{x}} \in \{\boldsymbol{x} : g^u(\boldsymbol{x}, P) \leq 0\}$ for some subset of indices $\{k'\} \subset \{k\}$. Since $\bar{\boldsymbol{x}}$ is a minimizer of $f(\boldsymbol{x})$ on the SIP-feasible set, we have $f(\bar{\boldsymbol{x}}) = f^{SIP}$. By the continuity of f, we also have $\lim\limits_{k' \to \infty} f(\boldsymbol{x}_{n_{k'}}^{ICR}) \equiv \lim\limits_{k' \to \infty} f_{n_{k'}}^{ICR} = f^{SIP}$. Since we have already shown that the sequence of minimum values is convergent, it follows that $\lim\limits_{k \to \infty} f_{n_k}^{ICR} = f^{SIP}$. $\square$

Note that the convergence of the sequence $f_{n_k}^{ICR}$ establishes the finite $\epsilon-$convergence of the subdivision approach. In other words, the true SIP solution can be approached to within ϵ-optimality in a finite number of iterations, k, for any SIP which satisfies the relatively mild differentiability and Slater point assumptions required by the ICR approach. It should be noted that the Slater point assumption applies only to boundaries defined by the infinite set of constraints expressed using the set P. The convergence of the ICR method is not affected if the assumed feasible points lie on boundaries defined by any ordinary (finite) constraints in the SIP.

An application of the subdivision method is illustrated here using Problem 3 from the Watson test set

$$\begin{aligned} &\min_{\boldsymbol{x} \in X} x_1^2 + x_2^2 + x_3^2 \\ x_1 + x_2 e^{x_3 p} + e^{2p} - 2\sin(4p) \leq 0 \quad &\forall p \in [0, 1]. \end{aligned} \tag{4.55}$$

Using natural interval extensions the following inclusion bounds were calculated for the $p-$dependent terms in g:

$$\begin{aligned} g_{xp}(\boldsymbol{x}, p) &= x_2 e^{x_3 p}, & G_{xp}^u &= \max(x_2 e^{x_3 p^l}, x_2 e^{x_3 p^u}) \\ g_p(p) &= e^{2p} - 2\sin(4p), & G_p^u &= e^2 - 2\sin(4) \end{aligned} \tag{4.56}$$

where $p^l = 0,\ p^u = 1$. The interval-constrained approximation was formulated using the bounds calculated in (4.56). An equivalent smooth NLP was then derived by applying the transformation described in Section 4.4.2. This NLP was solved to global optimality and the following result was obtained: $f^{ICR} = 39.6,\ \boldsymbol{x}^{ICR} = (-4.45, -4.45, 0),\ G^u = 0$. The SIP-optimality of this solution was checked by solv-

ing (4.51). A maximum of $\bar{g}^u(\boldsymbol{x}^{ICR}, P) = -0.123$, indicated that the inclusion bound G^u was inexact, and that $\boldsymbol{x}^{ICR}$ was not likely to be the global minimum of the SIP. In order to identify a better approximation to the SIP solution, the subdivision method was first applied using natural interval extensions:

$$\begin{aligned} g_{xp}(\boldsymbol{x}, p) &= x_2 e^{x_3 p}, & G^u_{xp,\kappa} &= \max(x_2 e^{x_3 p^l_\kappa},\ x_2 e^{x_3 p^u_\kappa}) \\ g_p(p) &= e^{2p} - 2\sin(4p), & G^u_{p,\kappa} &= e^{2p^u_\kappa} - 2sb_\kappa \end{aligned} \tag{4.57}$$

where $p^l_\kappa = \dfrac{(\kappa-1)}{n_k}$, $p^u_\kappa = \dfrac{\kappa}{n_k}$, $\quad \kappa = 1, \ldots, n_k$ and

$$\begin{aligned} \text{if } 4p^u_\kappa \leq \frac{\pi}{2} \quad sb_\kappa &= \sin(4p^l_\kappa) \\ \text{else if } 4p^l_\kappa \geq \frac{\pi}{2} \quad sb_\kappa &= \sin(4p^u_\kappa). \\ \text{else } sb_\kappa &= \min(\sin(4p^l_\kappa),\ \sin(4p^u_\kappa)). \end{aligned} \tag{4.58}$$

An updated upper-bounding problem for the SIP was formulated using the inclusions calculated in (4.57):

$$\begin{aligned} &\min_{\boldsymbol{x} \in X} x_1^2 + x_2^2 + x_3^2 \\ &x_1 + G^u_{p,\kappa} + G^u_{xp,\kappa} \leq 0 \quad \forall \kappa = 1, \ldots, n_k. \end{aligned} \tag{4.59}$$

The smooth NLP reformulation of (4.59) was solved for a number for different values of n_k. Using $n_k = 2^k$, at each iteration k, 9 iterations of the subdivision method were required to approach the (lower-bounding) literature solution to within two significant figures.

The subdivision method was also applied using Taylor models to calculate second order inclusions for the constraint function. As described earlier, an $n_T^{th}-$order Taylor model may be calculated by evaluating the natural interval extension of a centered $n_T^{th}-$order Taylor polynomial of g in $\boldsymbol{p}$. The second order Taylor coefficients for the

constraint function in (4.55) are:

$$\begin{aligned} g(\boldsymbol{x},\ p_{d,\kappa}) &= x_1 + x_2 e^{x_3 p_{d,\kappa}} + e^{2p_{d,\kappa}} - 2\sin(4p_{d,\kappa}) \\ \frac{\partial g(\boldsymbol{x},\ p_{d,\kappa})}{\partial p} &= x_2 x_3 e^{x_3 p_{d,\kappa}} + 2e^{2p_{d,\kappa}} - 8\cos(4p_{d,\kappa}) \\ \frac{1}{2!}\frac{\partial^2 g(\boldsymbol{x},\ p_{d,\kappa})}{\partial p^2} &= \frac{x_2 x_3^2 e^{x_3 p_{d,\kappa}}}{2} + 2e^{2p_{d,\kappa}} + 16\sin(4p_{d,\kappa}) \\ \frac{1}{3!}\frac{\partial^3 g(\boldsymbol{x},\ p_{r,\kappa})}{\partial p^3} &= \frac{x_2 x_3^3 e^{x_3 p_{r,\kappa}}}{6} + \frac{4e^{2p_{r,\kappa}}}{3} + \frac{64\cos(4p_{r,\kappa})}{3} \end{aligned} \tag{4.60}$$

where $p_{r,\kappa} \in P_\kappa$ and $p_{d,\kappa}$ is the midpoint of subinterval P_κ. The natural interval extension of this Taylor polynomial on P_κ was used to calculate each of the inclusion bounds $G^u_{n_T,\kappa}$. A new NLP was then formulated using these bounds in place of G^u_κ in (4.59). Proceeding as before, the smooth NLP reformulation was solved using $n_k = 2^k$ intervals at each iteration k. The higher order convergence of the Taylor method required only 4 iterations to converge to within two significant figures of the literature solution.

4.5 Application of the ICR method to Non-regular Problems

A unique contribution of the ICR approach is its ability to generate guaranteed SIP-feasible points within a finite number of iterations (just one iteration if the finite reformulation has a non-empty feasible set), without making strong assumptions on the problem structure. This property is illustrated using the following example from [32]:

$$\begin{aligned} &\min_{\boldsymbol{x}\in X} x_2 \\ -(x_1 - p)^2 - x_2 \le 0 \quad &\forall p \in [0,1] \\ 0 \le x_1 \le 1. & \end{aligned} \tag{4.61}$$

As noted in [32], the feasible set of (4.61) cannot be reproduced by enforcing constraints at a finite subset of points in P. As a result, discretization methods always yield infeasible lower bounds to the SIP solution at finite termination. Furthermore,

the local minimizer at $\boldsymbol{x} = (0, 0)$ is not a regular point, and thus local reduction methods should not be applied. At $\boldsymbol{x} = (0, 0)$, the Lagrange function for the inner maximization problem may be written as:

$$L(\boldsymbol{x}, p) = -(x_1 - p)^2 - x_2 + \nu_1(-p - 0) + \nu_2(p - 1). \tag{4.62}$$

The gradient of the Lagranage function is then $\nabla_p L = -2(x_1 - p) - \nu_1 + \nu_2$. For a critical point to occur, the following equalities must be satisfied:

$$\begin{aligned} \nabla_p L = -2(x_1 - p) - \nu_1 + \nu_2 = 0 \\ \nu_1(-p) + \nu_2(p - 1) = 0 \end{aligned} \tag{4.63}$$

At $\boldsymbol{x} = (0, 0)$ these conditions are satisfied by $p = 0,\ \nu_1 = 0,\ \nu_2 = 0$. However, since ν_1 is not strictly greater than zero while the constraint $p \geq 0$ is active, this is not a nondegenerate critical point at $\boldsymbol{x} = (0, 0)$ as defined in [57]. Thus the local minimum at $\boldsymbol{x} = (0, 0)$ does not satisfy the regularity assumption required by reduction methods.

However, the differentiability properties of (4.61) and its non-empty interior (e.g., $\boldsymbol{x} = (0.5,\ \ 0.5)$ is a feasible interior point) imply that a finitely-constrained reformulation of (4.61) may be constructed and solved for a SIP-feasible point using the interval-constrained approach. A Horner rearrangement was applied to the constraint function in (4.61), and the following smooth NLP representation of the interval-constrained reformulation was solved globally over the compact set $X = [-1000, 1000]^2$:

$$\begin{aligned} &\min_{\boldsymbol{x} \in X} x_2 \\ -x_1^2 - x_2 + p^l(2x_1 - p^l) \leq 0 \\ -x_1^2 - x_2 + p^l(2x_1 - p^u) \leq 0 \\ -x_1^2 - x_2 + p^u(2x_1 - p^l) \leq 0 \\ -x_1^2 - x_2 + p^u(2x_1 - p^u) \leq 0 \\ 0 \leq x_1 \leq 1 \\ p^l = 0,\ p^u = 1. \end{aligned} \tag{4.64}$$

Note that the ICR approach does not simply mimic discretization by applying the constraint $-(x_1 - p)^2 - x_2$ at finite points in the set $P = [0, 1]$; the second and third constraints in (4.64) cannot be generated by substituting points in $P = [0, 1]$ into the original SIP constraint. A solution $f^{ICR} = 0, \quad \boldsymbol{x} = (0, 0)$ was found. The feasibility of this point was verified by globally maximizing $g(\boldsymbol{x}^{ICR}, p)$ to obtain $\bar{g}^u(\boldsymbol{x}^{ICR}, P) = 0$. Moreover, the analysis in [32] confirms that the ICR solution is a global minimum of (4.61). This example clearly illustrates that the ICR approach can correctly identify a SIP-feasible point even when an infinite number of active constraints are encountered at certain feasible points within the compact set X. It should be noted that the feasible region of (4.64) is a subset of that of (4.61) but Theorem 1 guarantees convergence under subdivision.

4.6 Numerical Implementation and Results

The interval-constrained approach was applied to a number of nonconvex SIPs which have been studied previously in the literature (see Appendix A). Problems 1-9 from [78] are low-dimensional ($n_x \leq 8, n_p \leq 2$), twice-continuously differentiable examples which have been tackled using a number of different discretization and reduction-based methods [69, 55, 14, 42]; problems K, M, N are once-continuously differentiable examples which were solved in [55] using a globalized reduction method with exact penalty functions; problem L is from the same source, where it was presented as an example of a nonsmooth SIP. An equivalent continuously-differentiable reformulation of the original problem in [55] was solved here. Finally Problems S,[2] U are higher dimensional ($n_x \leq 6, n_p \leq 6$) examples which were also solved in [55].

All of the numerical algorithms presented in this work were implemented through the GAMS programming interface [9]. The smooth NLP/MINLP reformulations of the interval-constrained problems were derived by hand from the original SIP, and translated into the GAMS input language. Local NLP solutions were obtained by

[2]As noted in [77], the numerical results reported here correspond to $P = [0, 2]^{n_p}$ although the problem statement incorrectly defines $P = [0, 1]^{n_p}$ in the original source

calling CONOPT through GAMS. Global NLP and MINLP solutions were obtained using BARON. BARON implements a branch-and-reduce strategy to generate a sequence of converging upper and lower bounds for nonconvex optimization problems and is guaranteed to locate the global solution within $\epsilon-$optimality with a finite number of iterations. CPLEX is used to solve the linear relaxations and CONOPT is used to solve the nonlinear subproblems locally. The interested reader is referred to [59, 38, 17] for more details on these algorithms. The default optimization parameters for CPLEX, CONOPT and BARON are specified in [9]. Unless noted otherwise, these parameters were used throughout. The CPU times reported in this section correspond to the total execution time in seconds reported by GAMS on a 1.5GB, 1GHz Pentium IV PC running Linux.

4.6.1 Comparison of NLP and MINLP formulations

An initial study was done to compare the computational cost of solving the smooth NLP representation, to that of solving the MINLP representation of the interval-constrained problem in (4.31). Since both reformulations are exact, either approach yields the global solution to (4.31). (As discussed in Section 4.4, this solution is not necessarily the global minimum of the SIP.) Table 4.1 shows the resulting problem size using $n_k = 1$ in each case, i.e., without further domain subdivision. The column headings indicate the following: N_y is the number of binary variables used in the MINLP formulation. In parentheses is shown how these binary variables arise. A 'e' indicates an inclusion for an elementary function composition, and accounts for 1 out of N_y binary variables; a 'c' represents a product composition and accounts for 3 or 4 binary variables as shown in the previous section. NE is the number of equality constraints used to define G_p^u and the McCormick factors in the MINLP representation. Equality constraints are not used in the smooth NLP problems. NIE is the number of inequality constraints used in either reformulation. In the MINLP case it corresponds to the number of binary variables N_y plus one additional inequality which enforces the overall constraint $G^u \leq 0$. In the smooth NLP case, NIE grows exponentially with the number of min/max functions used to define G_{xp}^u, i.e., $NIE_{NLP} = 2^{2n_c+n_e}$,

Problem	N_y	NE	NIE_{MINLP}	NNL_{MINLP}	NIE_{NLP}	NNL_{NLP}
1	1(e)	3	2	9	2	6
2	3(c)	7	4	28	2	5
3	1(e)	2	2	9	2	5
4($n_x = 3$)	3(c)	6	4	12	4	0
5[3]	3(c)	6	4	15	4	3
6	1(e)	2	2	8	2	6
7	6(2c)	10	7	27	16	3
8	8(1e,2c)	15	9	36	32	0
9[3]	8(1e,2c)	15	9	48	32	0

Table 4.1: Comparison of sizes of MINLP and smooth NLP formulations

Problem	CPU_{MINLP}	CPU_{NLP}
1	0.09	0.03
2	0.16	0.42
3	0.28	0.06
4($n_x = 3$)	0.04	0.02
5[3]	0.22	0.03
6	0.48	0.09
7	0.23	0.02
8	0.63	0.01
9[3]	42.43	0.02

Table 4.2: Comparison of CPU times for global solution of SIP

as shown in the previous section. Finally, NNL is the number of Jacobian entries corresponding to nonlinear terms in the constraint set for each formulation. Table 4.2 shows the solution times for the MINLP and NLP problem formulations.

Table 4.2 shows that the computational costs of the NLP and MINLP reformulations are comparable for problems which have relatively simple functionality in the interval variable $\boldsymbol{p}$. For problems 7-9, the large number of binary variables required to reformulate the inclusion function adds significant nonlinearity to the MINLP reformulation, and the resulting problem is computationally expensive to solve. In

[3]This solution was obtained using $X = [-20, 20]^{n_x}$. The branch-and-reduce algorithm did not convergence in 900 seconds using the default specification for X.

the case of the NLP reformulation, although the number of inequality constraints increases exponentially with the number of min/max functions used to define the inclusion bounds, the degree of linearity/nonlinearity in the original SIP is preserved. As a result, the NLP formulation is more than an order of magnitude cheaper than the MINLP formulation for problems 8-9 which are linear in the optimization variables. Clearly, the MINLP solution times in Table 4.2 can be improved somewhat by applying standard heuristics, e.g., linear reformulation of bilinear binary/continuous terms. However, these refinements were not investigated further, and the smooth NLP reformulation was used to solve (4.31) to global optimality in the numerical studies reported in Tables 4.4 and 4.6.

4.6.2 Comparison of nonsmooth and smooth NLP formulations

In certain situations it is not essential to identify the global minimum of the SIP, and a guaranteed feasible or locally-optimal solution suffices. In such cases significant computational savings may be achieved by solving the interval-constrained reformulation locally, rather than globally. The GAMS programming interface does not provide access to special-purpose non-smooth optimizers. The nondifferentiable formulations were solved by specifying the discontinuous nonlinear program (DMINLP) option with the CONOPT NLP solver. This option simply enables CONOPT to ignore any discontinuities which arise when calculating derivatives of the constraints with respect to the optimization variables. Table 4.3 compares the cost of solving the smooth NLP reformulation to that of solving the original nondifferentiable formulation in (4.31). N_M is the number of max functions needed to define G^u_{xp} for the nondifferentiable formulation; NIE is the number of inequality constraints in the smooth NLP formulation. The non-differentiable formulation is subjected to a single constraint corresponding to $G^u_{xp} \leq 0$.

Table 4.3 indicates that the cost of solving the two formulations is comparable

[4]feasible, suboptimal solutions

Problem	n_x	N_M	CPU_{ND}	CPU_{NLP}	NIE
1	2	1(e)	0.003	0.003	2
3	3	1(e)	0.003	0.003	2
4^4	3	1(c)	0.003	0.002	4
5^4	3	1(c)	0.003	0.004	4
6	2	1(e)	0.003	0.005	2
7	3	2(2c)	0.003	0.003	16
8^4	6	3(1e,2c)	0.072	0.003	32
9^4	6	3(1e,2c)	0.293	0.004	32

Table 4.3: Comparison of CPU times for local solution of SIP

for relatively small problems ($n_x \leq 3$). Once again, the smooth NLP formulation is vastly cheaper for the more complex problems (8-9).

4.6.3 Comparison with reduction-based solutions

Tables 4.6.3 and 4.4 compare the global minima identified by the interval-constrained approach to SIP solutions reported previously using globalized reduction approaches. Natural interval extensions were used to calculate the inclusion functions, and the smooth NLP reformulation was solved to global optimality, without further subdivision of the domain P. The literature solutions (superscripted PCW) ([14] or [55]) and inclusion-constrained solutions (superscripted ICR) are shown in columns 2 and 4 of Table 4.4. The feasibility of both the literature and the obtained solutions were verified by finding the global maximum of $g(\boldsymbol{x}^{ICR}, \boldsymbol{p})$ and $g(\boldsymbol{x}^{SIP}, \boldsymbol{p})$ over P. Columns 3 and 5 show the maximum value assumed by $g(\boldsymbol{x}, \boldsymbol{p})$ at $\boldsymbol{x}^{PCW}$ and $\boldsymbol{x}^{ICR}$ respectively. An inclusion bound of $G^u(\boldsymbol{x}^{ICR}, P) = 0$ was obtained for each of the problems in Table 4.4. Column 6 shows the CPU time required to solve the inclusion-constrained NLP formulation. The solution points identified by the inclusion-constrained and reduction-based algorithms are shown in Table 4.6.3 Column 5 in Table 4.4 indicates that all of the inclusion-constrained solutions reported here are SIP-feasible as predicted by the properties of inclusion functions.[5] Although column 3 indicates that

[5]the absolute feasibility tolerance of the NLP solver cannot be controlled through the GAMS user interface and so cannot be reported here. All of the apparently positive constraint violations

Problem	f^{PCW}	$\max_{\boldsymbol{p}} g(\boldsymbol{x}^{PCW}, \boldsymbol{p})$	f^{ICR}	$\max_{\boldsymbol{p}} g(\boldsymbol{x}^{ICR}, \boldsymbol{p})$	**CPU**
1	-0.25	0	-0.25	0	0.03
2	0.1945	$-2.5 \cdot 10^{-8}$	0.1945	$-2.5 \cdot 10^{-8}$	0.42
3	5.3347	$5.3 \cdot 10^{-6}$	39.6287	-0.1233	0.06
4(n_x=3)	0.6490	$-2.7 \cdot 10^{-7}$	1.5574	-0.6505	0.02
4(n_x=6)	0.6161	0.	1.5574	0	0.03
4(n_x=8)	0.6156	0	1.5574	0	0.03
5	4.3012	$1.5 \cdot 10^{-8}$	4.7183	0	0.03
6	97.1588	$-5.9 \cdot 10^{-7}$	97.1588	$5.7 \cdot 10^{-6}$	0.09
7	1	0	1	0	0.02
8	2.4356	$9.9 \cdot 10^{-8}$	7.3891	$-3.9 \cdot 10^{-6}$	0.01
9	-12	0	-12	0	0.02
K	-3	0	-3	0	0.02
L	0.3431	$9.6 \cdot 10^{-6}$	1	-0.2929	0.03
M	1	0	1	0	0.01
N	0	0	0	0	0.02
S($n_p = 3$)	-3.6743	-1.1640	-1.2135	-1.2799	0.29
S($n_p = 4$)	-4.0871	-0.809	-0.8090	-1.2352	0.66
S($n_p = 5$)	-4.6986	-2.1733	-0.405	-1.3103	0.62
S($n_p = 6$)	-5.1351	-2.6513	0	-1.2634	0.02
U	-3.4831	$2.4 \cdot 10^{-8}$	-3.4822	-0.0002	0.03

Table 4.4: Comparison of reduction-based and inclusion-constrained solutions

Problem	$\boldsymbol{x}^{PCW}$	$\boldsymbol{x}^{ICR}$
1	(0, 0.5)	(0, 0.5)
2	(-0.75, -0.618)	(-0.75, -0.618)
3	(-0.213, -1.362, 1.853)	(-4.51, -4.51, -4.51)
4 ($n_x = 3$)	(0.0891, 0.423, 1.05)	(1.56, 0, 0)
4 ($n_x = 6$)	(0.00208, 0.963, 0.203, -0.0266, 0.0823, 0.333)	(1.56, 0, 0, 0, 0, 0)
4 ($n_x = 8$)	(0.00302, 0.954, 0.224, -0.0187, 0.0615, 0.306, -0.00690, 0.0348)	(1.56, 0, 0, 0, 0, 0, 0,0)
5	(1.01, -0.127, -0.380)	(1, 0, 0)
6	(0.720, -1.450)	(0.720, -1.450)
7	(-1, 0, 0)	(-1, 0, 0)
8	(2.58, -4.11, -4.11, 4.25, 4.53, 4.25)	(7.39, 0, 0, 0, 0, 0)
9	(3, 0, 0, 0, 0, 0)	(3, 0, 0, 0, 0, 0)
K	(0,1)	(0,1)
L	(0.707, 0.707)	(0.5, 0.5)
M	(1, 0)	(1, 0)
N	(0,0)	(0,0)
S(n_p=3)	(0.894, -1.29, 1.24, -0.749)	(0.911, -1.47, 1.47, -0.911)
S(n_p=4)	(0.948, -1.36, 1.30, -0.787)	(-0.831, 1.35, -1.35, 0.831)
S(n_p=5)	(0.914, -1.39, 1.52, -0.868)	(-0.872, 1.41, -1.41, 0.872)
S(n_p=6)	(0.961, -1.456, 1.581, -0.906)	(0.911, -1.47, 1.47, -0.911)
U	(1.17, 1.18, 1.14, 0.412)	(1.18, 1.18, 1.14, 0.413)

Table 4.5: Solutions identified by reduction and the ICR approach

Problem	$ndiv_{TM}$	CPU_{TM}	$ndiv_{IE}$	CPU_{IE}
3	16	172	512	291
$4(n_x = 3)$	4	0.1	256	0.42
5	2	0.40	16	0.16
L	16	0.68	512	60.48

Table 4.6: Comparison of subdivision implementation using natural interval extensions and second-order Taylor models.

all of the literature solutions reported in Table 4.4 are also SIP-feasible, reduction-based methods may, in theory, yield lower bounds which are not SIP-feasible at finite termination. For several cases shown in Table 4.4 the inclusion-constrained solution coincides with the reduction-based solution, i.e., problems 1, 2, 6, 7, 9, K, M, N. The upper-bounding property of the former, and the lower-bounding property of the latter, guarantee the SIP-optimality of such solutions. Column 5 shows that for these cases the inclusion bound is exact at the solution, i.e., $g^u(\boldsymbol{x}^{ICR}, \boldsymbol{p}) = G^u(\boldsymbol{x}^{ICR}, P) = 0$.

4.6.4 Application of subdivision

For other cases, e.g., example 3, Table 4.4 shows that $g^u(\boldsymbol{x}^{ICR}, P)$ is strictly less than $G^u(\boldsymbol{x}^{ICR}, P)$, thereby suggesting that the inclusion-constrained reformulation may not yield the true SIP minimum using natural extensions and $n_k = 1$. This is further confirmed by the discrepancy between the reduction and inclusion-constrained minima. The subdivision method was applied to a subset of these cases (those for which n_p=1) in order to derive better approximations to the SIP. Inclusion functions were calculated using natural interval extensions in one case (IE), and second-order Taylor models (TM) in the other. Table 4.6 compares the CPU time required to solve the final NLP using either approach, and the number of subdivisions ($ndiv$) required to approach the literature solution to within 2 significant figures using $n_k = 2^k$.

Table 4.6 shows that the lower-order convergence of the natural interval extensions require more iterations of the uniform subdivision approach to converge. However,

reported in Columns 3 and 5 are likely to be within this feasibility tolerance.

this does not necessarily correspond to a higher overall CPU solution time. Furthermore, the results in Table 4.6 do not reflect the computational effort required to generate the Taylor coefficients, which increases significantly with the dimensionality of $\boldsymbol{p}$.

4.7 Discussion

The ICR method introduced in this chapter offers a new approach to the numerical solution of smooth nonlinear SIPs. The method is applicable to the large class of SIPs which satisfy partial differentiability in $\boldsymbol{x}$, partial continuity in $\boldsymbol{p}$, possess an interior feasible point, and for which convergent inclusions of the constraint function can be constructed. The SIP is reformulated as a finite, nonlinear or mixed-integer nonlinear program, which is then treated using an appropriate global optimization algorithm. When applied to a limited class of nonlinear SIPs, this approach yields an exact finite reformulation which can be solved directly for the SIP minimum. In the general case the inclusion-constrained reformulation described here is over-constrained relative to the SIP, in contrast with existing methods for nonlinear SIPs. Thus the derived finite problem yields a guaranteed feasible upper bound to the SIP solution. This upper bound may be refined by successively solving NLPs/MINLPs of increasing size to global optimality. It has been shown that the sequence of solutions generated by this iterative subdivision procedure converges to the true SIP minimum, and $\epsilon-$convergence can be achieved within a finite number of steps. In other words, the method generates a convergent sequence of inner approximations to the SIP. In particular, the method has been shown to converge for problems which are not theoretically amenable to reduction, and for which discretization methods yield infeasible lower bounds at every level of grid refinement. When only a feasible solution to the SIP is required, such a point may be identified by solving the inclusion-constrained reformulation locally, without further iteration. In such cases, the differentiability requirement may be relaxed provided an appropriate nonsmooth optimizer is available to solve the derived finite NLP. In many applications, the ability to guarantee

SIP-feasibility may be more important than optimality.

The inclusion-constrained approach has been shown to perform robustly and efficiently on the relatively small-scale problems studied here. However, the method as it stands suffers from two main shortcomings. The first is the absence of a lower bound to which the feasible upper bound may be compared at finite termination. In other words, it is impossible to verify the 'goodness' of the current iterate without knowing the SIP minimum (or a valid lower bound) independently, e.g., using a discretization or reduction-based method. The second is the computational effort required to achieve convergence. When subdivision is applied, a sequence of nonconvex NLPs with an exponentially increasing number of nonlinear constraints must be solved. Clearly, this is impractical for all but the smallest problems. In the next chapter we describe a branch-and-bound framework which obviates the solution of multiple nonconvex NLPs by combining the upper and lower-bounding approaches to solve general, nonconvex SIPs to guaranteed $\epsilon-$optimality.

Chapter 5

A Branch-and-Bound Framework for Global Solution of SIPs

In the previous chapter, an upper-bounding approach to the the global solution of SIPs was introduced. Although the ICR method as it stands is useful for identifying SIP-feasible points, it is not practical for solving nonlinear SIPs to guaranteed global optimality. As discussed earlier, the implementation described so far is extremely computationally intensive, and provides no bound on the discrepancy between the minimum value and the value of the incumbent solution at finite termination. In this chapter these shortcomings are addressed by combining the upper-bounding, inclusion-constrained reformulation with a convex, lower-bounding discretized approximation. A Branch-and-Bound (B&B) framework [21] is developed as a practical approach to solving general, nonconvex SIPs to $\epsilon-$optimality.

In the next section relevant branch-and-bound terminology and concepts from finite nonlinear programing are reviewed. The upper and lower-bounding problems solved at each node of the SIP B&B tree are described in Section 5.2. The algorithm is stated in full, and subsequently proved to converge finitely to $\epsilon-$optimality. Numerical results from the application of the described algorithm are presented in Section 5.3. Finally, the chapter is concluded with a discussion on the overall utility and limitations of the SIP B&B approach.

5.1 Overview of Branch and Bound Methods

In this section basic terminology and concepts relevant to B&B procedures are reviewed. The reader is referred to [35] for a thorough discussion of B&B algorithms for global optimization of finite problems. To the extent possible, the presentation here follows the notation and terminology in [35].

The general approach in any B&B algorithm consists of generating convergent sequences $\{\alpha_k\}$ and $\{\beta_k\}$ of upper and lower bounds, respectively, on the optimal objective function value of the optimization problem being solved, (F). At least one solution of F is assumed to exist. An initial relaxation M_0 of the feasible set D is defined such that $D \subset M_0$, e.g., $M_0 = X$ for the SIP algorithm. At each iteration k, upper and lower-bounding problems are solved over a finite number of subsets, or nodes, of M_0. These nodes are denoted $M_{k_i} \in I_k$ where I_k is the set of active nodes at iteration k. A lower bound β_{k_i} is generated for each node $M_{k_i} \in I_k$ by solving a relaxation of F on M_{k_i}. An upper bound α_{k_i} on the solution of F is generated by solving F locally on the feasible subset $M_{k_i} \cap D$, if it exists. The 'overall' bounds are then taken to be $\beta_k = \min\limits_{M_{k_i} \in I_k} \beta_{k_i}$ and $\alpha_k = \min\limits_{M_{k_i} \in I_k} \alpha_{k_i}$. A B&B procedure is said to be 'finitely-convergent to $\epsilon-$optimality' if $\lim\limits_{k\to\infty} \alpha_k = \lim\limits_{k\to\infty} \beta_k = \min(F)$ such that $\forall \epsilon > 0, \quad \exists k^*$ such that $\alpha_k - \beta_k \leq \epsilon, \quad k > k^*$. Such a procedure is terminated when $\alpha_k - \beta_k \leq \epsilon$ is achieved, and $\boldsymbol{x}^k \in \{\boldsymbol{x} : f(\boldsymbol{x}) = \alpha_k,\ \boldsymbol{x} \in M_{k_i} \cap D,\ M_{k_i} \in I_k\}$ is taken to be an estimate of the global minimum solution point of F. Otherwise nodes with lower bounds which exceed α_k are excluded from further consideration (fathomed), since $\min(F)$ cannot occur within these sets. Every unfathomed node is required to be capable of further refinement. A finite subset of these surviving nodes are repartitioned, and a new iteration is begun. Sufficient conditions for the finite ϵ-convergence of a branch-and-bound procedure are stated in [35, 33, 34].

5.2 A Global Optimization Algorithm for Semi-infinite Programs

5.2.1 Upper-Bounding Problem

As shown in the previous chapter, a finitely-constrained upper-bounding problem for a SIP may be constructed using an inclusion for the constraint function $g(\boldsymbol{x}, \boldsymbol{p})$ on P. A convergent sequence of upper bounds may be generated using increasingly tighter inclusion functions derived using the subdivision notion. A partition of the interval P is used to formulate the upper-bounding problem solved at each node of the B&B tree. This partition, $P = \bigcup_{\tau \in T_q} P_\tau$ is determined solely by the level, q, at which the node occurs in the B&B tree, and is independent of the iteration number, k, and the node in question. For an infinite sequence of nested nodes $\{M_{k_{q_{i_q}}}\}$, the partition elements P_τ are required to be monotonically decreasing in width such that degeneracy is approached in the limit $q \to \infty$, i.e.,

$$\begin{gathered} \max_{\tau \in T_q} w(P_\tau) > \max_{\tau \in T_{q+1}} w(P_\tau) \\ \lim_{q \to \infty} \max_{\tau \in T_q} w(P_\tau) = 0, \end{gathered} \tag{5.1}$$

where the width of a multidimensional interval P_τ is defined to be $\max_m w(P_{\tau,m}),\ m = 1, \ldots, n_p$ [56]. The collection of sets $\{P_\tau\}_{\tau \in T_q}$ is used to define the feasible region for the upper-bounding problem. The objective function value at any ICR-feasible point, $f^{ICR}_{k_{q_{i_q}}}$, provides an upper bound on the minimum solution value of the SIP on $M_{k_{q_{i_q}}}$.

$$f^{ICR}_{k_{q_{i_q}}} \in \{f(\boldsymbol{x}) : \boldsymbol{x} \in M_{k_{q_{i_q}} \in I_k},\ G^u(\boldsymbol{x}, P_\tau) \leq 0,\ P_\tau in T_q\} \tag{5.2}$$

If no feasible point can be found for (5.2), $f^{ICR}_{k_{q_{i_q}}} = \infty$ is assigned. Once (5.2) has been determined for each node $M_{k_{q_{i_q}}} \in I_k$, the overall best available solution and upper

bound are updated by setting

$$\begin{gathered}\alpha_k = \min_{M_{kq_{iq}} \in I_k} f^{ICR}_{kq_{iq}} \\ \boldsymbol{x}^k \in \{\boldsymbol{x} : f(\boldsymbol{x}) = \alpha_k,\ \boldsymbol{x} \in M_{kq_{iq}} \in I_k,\ G^u(\boldsymbol{x}, P_\tau) \leq 0, \tau \in T_q\}.\end{gathered} \tag{5.3}$$

5.2.2 Lower-Bounding Problem

As discussed in Chapter 4, discretization methods can be used to generate convergent outer approximations for a semi-infinite program. In the context of the SIP B&B algorithm, a discretized approximation may be used to generate a valid relaxation for the SIP on a given node. The grid, or index set, $S_q \subset P$ is determined only by the level q at which the corresponding node occurs in the B&B tree relative to the root node. To preserve the convergence of the SIP B&B procedure, the grid sequence $\{S_q\}$ associated with an infinite sequence of nested nodes $\{M_{kq_{iq}}\}$ is required to satisfy the following properties:

$$\begin{gathered}S_q \subset S_{q+1} \subset P \\ \lim_{q \to \infty} dist(S_q, P) = 0,\end{gathered} \tag{5.4}$$

where the grid density $dist(S_q, P)$ is defined as [57]:

$$dist(S_q, P) = \sup_{\boldsymbol{p}_1 \in P} \inf_{\boldsymbol{p}_2 \in S_q} ||\boldsymbol{p}_1 - \boldsymbol{p}_2||_\infty. \tag{5.5}$$

If $f^{SIP}_{kq_{iq}}$ is defined to be the solution of the SIP on a (feasible) node $M_{k_i} \subset X$ such that

$$f^{SIP}_{kq_{iq}} = \min_{\boldsymbol{x} \in M_{kq_{iq}}} f(\boldsymbol{x}) \text{ s.t. } g(\boldsymbol{x}, \boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in P, \tag{5.6}$$

then solving the following finite relaxation yields a lower bound $f^D_{kq_{iq}} \leq f^{SIP}_{kq_{iq}}$:

$$f^D_{kq_{iq}} = \min_{\boldsymbol{x} \in M_{kq_{iq}}} f(\boldsymbol{x}) \text{ s.t. } g(\boldsymbol{x}, \boldsymbol{p}) \leq 0 \quad \forall \boldsymbol{p} \in S_q. \tag{5.7}$$

For notational convenience, the feasible sets defined by the constraints in (5.6) and(5.7) are referred to hereafter as $\{\boldsymbol{x} : g^s(\boldsymbol{x}) \leq 0\}$ and $\{\boldsymbol{x} : g^D_q(\boldsymbol{x}) \leq 0\}$ respectively.

Problems for which both the inclusion-constrained reformulation and the discretized approximation are convex in the optimization variables, $\boldsymbol{x}$, can be solved globally over any domain (without excessive computational effort). In such cases, it is not necessary to branch on the set X; a single upper and lower-bounding problem is solved at each iteration using $I_k = \{X\}$ such that the level, q, used to define S_q and T_q, is set by $q = k$. To solve the convex SIP to $\epsilon-$optimality, the upper and lower-bounding problems so defined are solved over a finite number of iterations k^* such that $f_k^{ICR} - f_k^D \leq \epsilon, \quad k \geq k^*$. When the SIP is convex in the optimization variables but the inclusion-constrained reformulation introduces nonconvexities in (5.2), the discretized problem in (5.7) is solved (globally) for a lower-bounding solution, and the inclusion-constrained reformulation is solved (locally) for an upper-bounding solution defined by (5.2), and in general it will be necessary to branch on X in order to converge the upper bound.

In the general case, the functions $f(\boldsymbol{x})$ and/or $g(\boldsymbol{x}, \boldsymbol{p})$ are nonconvex in $\boldsymbol{x}$, and (5.7) must be solved globally in order to identify a valid lower bound on $f_{k_{q_{i_q}}}^{SIP}$. Such an approach requires multiple nonconvex NLPs to be solved globally at each iteration, and quickly becomes computationally prohibitive with increasing q. Instead, a convex relaxation of (5.7) may be solved for a lower bound on $f_{k_{q_{i_q}}}^{SIP}$. This approach entails significantly lower computational cost per node, but yields a potentially looser lower bound than (5.7). Consequently, a larger number of iterations may be required for the lower-bounding sequence $\{\beta_k\}$ to converge. In order to derive a valid convex relaxation, the McCormick factorization scheme [47] is applied to (5.7) to generate an equivalent reformulation in the following form:

$$\begin{aligned} f_{k_{q_{i_q}}}^D &= \min_{\boldsymbol{y}} y_N(\boldsymbol{x}) \\ x_{n,k_{q_{i_q}}}^l \leq y_n(\boldsymbol{x}) &\leq x_{n,k_{q_{i_q}}}^u \quad \forall n = 1, \ldots, n_x \\ y_n(\boldsymbol{x}) &\leq 0 \qquad\quad \forall n = (N - |S_q|), \ldots, (N-1), \end{aligned} \tag{5.8}$$

where N is total number of McCormick factors needed to reformulate (5.7) exactly. The factors $y_n(\boldsymbol{x}),\ n = 1, \ldots, n_x$ correspond to the elements of the decision vector $\boldsymbol{x}$.

The remaining factors $y_n,\ n = (n_x + 1), \ldots, N$ are defined recursively as univariate compositions, sums or biproducts of previously-defined factors. Where ever necessary, the bounds for the intermediate factors $y_n(\boldsymbol{x}),\ n = (n_x + 1), \ldots, (N - |S_q| - 1)$ are estimated using interval analysis methods. The (unconstrained) terminal factor y_N is defined to evaluate to the objective function value of (5.7). The constrained factors $y_n,\ n = (N - |S_q|), \ldots, (N - 1)$ are defined to evaluate to the constraint values $g(\boldsymbol{x}, \boldsymbol{p}_m),\ m = 1, \ldots, |S_q|$.

The convex underestimating program derived from (5.8) is:

$$\begin{aligned} f^{DC}_{k_{q_{i_q}}} &= \min_{\boldsymbol{y}^c} y^c_N(\boldsymbol{x}) \\ x^l_{n,k_{q_{i_q}}} \leq y^c_n(\boldsymbol{x}) &\leq x^u_{n,k_{q_{i_q}}} \quad \forall n = 1, \ldots, n_x \\ y^c_n(\boldsymbol{x}) &\leq 0 \qquad\quad \forall n = (N - |S_q|), \ldots, (N - 1), \end{aligned} \tag{5.9}$$

where the factors $y^c_n(\boldsymbol{x}),\ n = n_x + 1, \ldots, N$ are defined recursively using convex and concave inequality and equality constraints. For notational convenience, the feasible set of (5.9) is denoted as $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g^{DC}_q(\boldsymbol{x}) \leq 0\}$. A lower-bounding solution for each node $M_{k_{q_{i_q}}} \in I_k$ is assigned by solving (5.9) on each of the active sets in the current partition. Each node for which $f^{DC}_{k_{q_{i_q}}}$ exceeds the best available upper bound α_k is fathomed. The overall lower bound is then updated by setting:

$$\beta_k = \min_{M_{k_{q_{i_q}}} \in I_k} f^{DC}_{k_{q_{i_q}}}. \tag{5.10}$$

The bound-improving property is established by selecting the node (or one of the nodes) at which a lower bound of β_k is attained, for further refinement. Two new nodes are subsequently generated by bisecting the selected node along the dimension (or one of the dimensions) which maximizes $x^u_{k_{q_{i_q}}} - x^l_{k_{q_{i_q}}}$. This refinement procedure results in an exhaustive subdivision scheme such that any infinite sequence of nested nodes generated by the SIP B&B procedure approaches degeneracy in the limit $q \to \infty$, i.e., $\lim_{q \to \infty} w(M_{k_{q_{i_q}}}) = 0$ [35].

5.2.3 Algorithm

The B&B procedure for solving general nonconvex SIPs to $\epsilon-$optimality is outlined below:

1. Define the grid sequence $\{S_q\}$ and the partition sequence $\{T_q\}$.

2. Set $k := 0, \quad M_{k_{q_{i_q}}} = X, \quad I_k = \{M_{0_{1_{1_1}}}\}$.

3. Set $\alpha_0 := \infty$.

4. Locate a feasible point for the inclusion-constrained reformulation on $M_{0_{1_{1_1}}}$ if possible. Solve the convexified discretized approximation (5.9) on $M_{0_{1_{1_1}}}$. Set α_0 and β_0 to the respective solution values.

5. If $\alpha_0 - \beta_0 \leq \epsilon$ then stop and assign $f^{SIP} = \alpha_0$ and $\boldsymbol{x}^{SIP} = \boldsymbol{x}^k \in \{\boldsymbol{x} : f(\boldsymbol{x}) = \alpha_0, \boldsymbol{x} \in M_{0_{1_{1_1}}}, \; G^u(\boldsymbol{x}, P_\tau) \leq 0, \tau \in T_1\}$.

6. Delete (fathom) from I_k all nodes $M_{k_{q_{i_q}}} \in I_k$ for which $f^{DC}_{k_{q_{i_q}}} \geq \alpha_k$.

7. Select a $M_{k_{q_{i_q^*}}} \in I_k$ such that $\beta_k = f^{DC}_{k_{q_{i_q^*}}}$.

8. Generate two new nodes by bisecting $M_{k_{q_{i_q^*}}}$ along the dimension (or one of the dimensions) which maximizes $x^u_{n,k_{q_{i_q^*}}} - x^l_{n,k_{q_{i_q^*}}}$. Delete $M_{k_{q_{i_q^*}}}$ from I_k.

9. Set $k := k + 1$.

10. Add the two newly-created nodes to the set I_k. Copy all surviving nodes from I_{k-1} to I_k.

11. Solve (5.9) for $f^{DC}_{k_{q_{i_q}}}$ on each of the newly-created nodes. Assign $f^{DC}_{k_{q_{i_q}}} = \infty$ for each node at which (5.9) is infeasible. For each node attempt to identify a feasible point $\boldsymbol{x}^{k_{q_{i_q}}}$ in the ICR-feasible set. If a feasible point is found, assign the corresponding objective function value to $f^{ICR}_{k_{q_{i_q}}}$. Otherwise assign $f^{ICR}_{k_{q_{i_q}}} = \infty$.

12. Set $\alpha_k = \min_{M_{k_{q_{i_q}}} \in I_k} f^{ICR}_{k_{q_{i_q}}}$.

13. Set $\beta_k = \min_{M_{k_{q_{i_q}}} \in I_k} f^{DC}_{k_{q_{i_q}}}$.

14. If $\alpha_k - \beta_k \leq \epsilon$ then stop and assign $f^{SIP} = \alpha_k$ and $\boldsymbol{x}^{SIP} \in \{\boldsymbol{x} : f(\boldsymbol{x}) = \alpha_k,\ \boldsymbol{x} \in M_{k_{q_{i_q}}} \in I_k,\ G^u(\boldsymbol{x}, P_\tau) \leq 0, \tau \in T_q\}$. Else repeat steps 5 - 12.

5.2.4 Finite ϵ-convergence of the SIP B&B algorithm

In this section the finite $\epsilon-$convergence of the SIP B&B algorithm is is proved by showing that the generated sequences of lower and upper bounds both converge to the true SIP solution in the limit $k \to \infty$. First, it is shown that the B&B scheme cannot generate an infinite sequence of nested nodes, $\{M_{k_{q_{i_q}}}\}$, which converges to an infeasible point in X, i.e., fathoming nodes for which the convex underestimating program is infeasible is a deletion-by-infeasibility rule [34] which is certain in the limit.

Lemma 1. *For an infinite sequence of nested nodes converging to a point $\bar{\boldsymbol{x}} \in X$, the sequence of feasible sets for (5.7), $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g^D_q(\boldsymbol{x}) \leq 0\}$, converges to the principal limit $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} \in X : g^s(\boldsymbol{x} :\leq 0\}$.*

Proof. The sequence of sets $\{\boldsymbol{x} \in X : g^D_q(\boldsymbol{x}) \leq 0\}$ defined by (5.7) is clearly a decreasing sequence such that $\{\boldsymbol{x} \in X : g^D_q(\boldsymbol{x}) \leq 0\} \supset \{\boldsymbol{x} \in X : g^D_{q+1}(\boldsymbol{x})\}, \quad q \geq 1$. The rectangular partitioning scheme of the branch-and-bound algorithm ensures that $M_{k_{q_{i_q}}} \supset M_{k_{q+1_{i_{q+1}}}}, \quad q \geq 1$ is satisfied for an infinite sequence of nested nodes. Thus the sequence of feasible sets for (5.7), $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g^D_q(\boldsymbol{x}) \leq 0\}$, is also a decreasing sequence. Since $\lim_{q\to\infty} dist(S_q, P) = 0$ by construction, the superior limit of this sequence is $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} \in X : g^s(\boldsymbol{x}) \leq 0\}$. It follows that the decreasing sequence, $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g^D_q(\boldsymbol{x}) \leq 0\}$, converges to the prinicipal limit $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} \in X : g^s(\boldsymbol{x}) \leq 0\}$. $\square$

Corollary 1. *For an infinite sequence of nested nodes converging to a point $\bar{\boldsymbol{x}} \in X$, $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} : g^s(\boldsymbol{x} \leq 0)\}$ is the inferior limit of the sequence $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g^D_q(\boldsymbol{x}) \leq 0\}$.*

Lemma 2. *For an infinite sequence of nested nodes converging to a point $\bar{\boldsymbol{x}} \in X$, the sequence of feasible sets for (5.9), $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g^{DC}_q(\boldsymbol{x}) \leq 0\}$, converges to the principal limit $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} \in X : g^s(\boldsymbol{x}) \leq 0\}$.*

Proof. The properties of the McCormick reformulation ensure that increasingly tighter approximations to (5.7) are generated on successive nodes which satisfy $M_{k_{q_{i_q}}} \supset M_{k_{q+1_{i_{q+1}}}}, \quad q \geq 1$. Consequently, the sequence of feasible sets for (5.8), $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g_q^{DC}(\boldsymbol{x}) \leq 0\}$, is a convergent decreasing sequence. In particular, the McCormick scheme generates underestimating and overestimating functions which evaluate to the original functions, $y_n(\boldsymbol{x})$, on any degenerate interval, e.g., $\bar{\boldsymbol{x}}$. Thus $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} \in X : g_{q=\infty}^{D}(\boldsymbol{x}) \leq 0\} = \bar{\boldsymbol{x}} \cap \{\boldsymbol{x} \in X : g^s(\boldsymbol{x}) \leq 0\}$ is the principal limit of the convergent sequence $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g_q^{DC}(\boldsymbol{x}) \leq 0\}$. □

Corollary 2. *For an infinite sequence of nested nodes converging to a point, $\bar{\boldsymbol{x}}$, $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} \in X : g^s(\boldsymbol{x}) \leq 0\}$ is the inferior limit of the sequence $\{\boldsymbol{x} \in M_{\cdot} g_q^{DC}(\boldsymbol{x}) \leq 0\}$.*

Lemma 3. *The fathoming of nodes which are infeasible for the convex lower-bounding problem (5.9) is a deletion-by-infeasibility rule which is certain in the limit.*

Proof. As noted in the previous section, the exhaustiveness of the SIP subdivision procedure guarantees that any infinite sequence of nested nodes converges to a point $\bar{\boldsymbol{x}}$. Assume that $\bar{\boldsymbol{x}}$ is infeasible. By continuity of $g(\boldsymbol{x}, \boldsymbol{p})$ it follows that there exists some finite q^0 such that $g^s(\bar{\boldsymbol{x}}) \geq \varepsilon > 0, \quad \forall \boldsymbol{x} \in M_{k_{q_{i_q}}}, \forall q \geq q^0$. Since $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g^s(\boldsymbol{x}) \leq 0\} = \emptyset, \quad q \geq q^0$, it follows from Corollary 2 that there exists some finite $q^1 \geq q^0$ such that $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : g_q^{DC}(\boldsymbol{x}) \leq 0\} = \emptyset, \quad q \geq q^1$. Since (5.9) is clearly infeasible for $q \geq q^1$, it follows that the sequence of nested nodes will be fathomed at $q = q^1$. Thus the branch-and-bound procedure cannot generate an infinite sequence of nested nodes which converges to an infeasible point.

□

Lemma 4. *The bounding operation is consistent such that* $\lim_{q \to \infty} f_{k_{q_{i_q}}}^{ICR} - f_{k_{q_{i_q}}}^{DC} = 0$

Proof. The difference between the calculated upper and lower bounds on a node $M_{k_{q_{i_q}}}$ may be expressed as follows:

$$f_{k_{q_{i_q}}}^{ICR} - f_{k_{q_{i_q}}}^{DC} = \left(f_{k_{q_{i_q}}}^{ICR} - f_{k_{q_{i_q}}}^{SIP}\right) + \left(f_{k_{q_{i_q}}}^{SIP} - f_{k_{q_{i_q}}}^{DC}\right). \tag{5.11}$$

As shown in Lemma 3, each node in an infinite sequence of nested nodes is known to be SIP-feasible, and correspondingly, is also feasible for the discretized problem and its convex relaxation. Thus it is known that $f^{DC}_{k_{q_{i_q}}} \leq f^{SIP}_{k_{q_{i_q}}} < \infty, \quad q \geq 1$ in an infinite sequence of nested nodes. The decrease in the size of the respective feasible sets with q ensure that $\{f^{SIP}_{k_{q_{i_q}}}\}$ and $\{f^{DC}_{k_{q_{i_q}}}\}$ are both nondecreasing sequences. The continuity of f and the compactness of X further require that $\{f^{SIP}_{k_{q_{i_q}}}\}$ and $\{f^{DC}_{k_{q_{i_q}}}\}$ are bounded from above by $f^m = \max_{\boldsymbol{x} \in X} f(\boldsymbol{x})$. Therefore $\{f^{SIP}_{k_{q_{i_q}}}\}$ and $\{f^{DC}_{k_{q_{i_q}}}\}$ are known to converge in the limit $q \to \infty$.

The sequence of minimizers, $\{\boldsymbol{x}^{DC}_{k_{q_{i_q}}}\}$, identified by solving (5.9) at each node, is a bounded sequence in the compact set X. Thus $\{\boldsymbol{x}^{DC}_{k_{q_{i_q}}}\}$ has a convergent subsequence having an accumulation point in the limit set $\{\bar{\boldsymbol{x}}\} \cap \{\boldsymbol{x} \in X : g^s(\boldsymbol{x}) \leq 0\}$, i.e. $\lim_{q' \to \infty} \boldsymbol{x}^{DC}_{k_{q' i_{q'}}} = \bar{\boldsymbol{x}}$ for some subset of indices $\{q'\} \subset \{q\}$, since $\bar{\boldsymbol{x}}$ is known to be SIP-feasible. From the continuity of the function $y^c_N(\boldsymbol{x})$, we have $\lim_{q' \to \infty} y^c_N(\boldsymbol{x}_{k_{q' i_{q'}}}) = y^c_N(\bar{\boldsymbol{x}})$. From the properties of the McCormick underestimators, and the degeneracy of the limit set, $\bar{\boldsymbol{x}}$, it follows that $y^c_{N,k_{q' i_{q'}}}(\bar{\boldsymbol{x}}) = f(\bar{\boldsymbol{x}})$.

Similarly, the sequence of minimizers identified by solving (5.6) at each node has a convergent subsequence such that $\lim_{q' \to \infty} \boldsymbol{x}^{SIP}_{k_{q' i_{q'}}} = \bar{\boldsymbol{x}}$ and $\lim_{q' \to \infty} f^{SIP}_{k_{q' i_{q'}}} = f(\bar{\boldsymbol{x}})$. Since both $\{f^{SIP}_{k_{q_{i_q}}}\}$ and $\{f^{DC}_{k_{q_{i_q}}}\}$ are known to be convergent sequences, it follows that $\lim_{q \to \infty} f^{SIP}_{k_{q_{i_q}}} = \lim_{q \to \infty} f^{DC}_{k_{q_{i_q}}} = f(\bar{\boldsymbol{x}})$.

Theorem 1 states that the sequence $\{\boldsymbol{x} \in X : G^u_q(\boldsymbol{x}, P) \leq 0\}$ converges to $\{\boldsymbol{x} \in X : g^s(\boldsymbol{x}) \leq 0\}$ in the limit $q \to \infty$. Any infinite sequence of nested nodes generated by the SIP B&B procedure has already been shown to converge to an SIP-feasible point, $\bar{\boldsymbol{x}}$, such that $g^s(\bar{\boldsymbol{x}}) \leq 0$, i.e., $\bar{\boldsymbol{x}}$ is in the inferior limit set of the sequence $\{\boldsymbol{x} \in X : G^u_q(\boldsymbol{x}, P) \leq 0\}$. Since $\bar{\boldsymbol{x}} \in M_{k_{q_{i_q}}}, \ q \geq 1$, it follows that $\{\boldsymbol{x} \in M_{k_{q_{i_q}}} : G^u_{k_{q_{i_q}}}(\boldsymbol{x}) \leq 0\} \neq \emptyset, \quad q \geq q^2$ and consequently $f^{ICR}_{k_{q_{i_q}}} < \infty, \quad q \geq q^2$ for some finite $q^2 \geq 1$. By the properties of inclusion functions, and the potential suboptimality of the identified feasible point we have $f^{ICR}_{k_{q_{i_q}}} - f^{SIP}_{k_{q_{i_q}}} \geq 0, \quad q \geq 1$. By the compactness of $M_{k_{q_{i_q}}}$ we have $f^{ICR}_{k_{q_{i_q}}} - f^{SIP}_{k_{q_{i_q}}} \leq \epsilon_q, \quad q \geq q^2$. Since exhaustive partitioning ensures that $\lim_{q \to \infty} w(M_{k_{q_{i_q}}}) = 0$, the continuity of f yields $\lim_{q \to \infty} \epsilon_q = 0$. Combining these results using the squeeze theorem, we have $\lim_{q \to \infty} \lim f^{ICR}_{k_{q_{i_q}}} - f^{SIP}_{k_{q_{i_q}}} = 0$.

It has been shown that each of the differences in (5.11) approaches 0 in the limit $q \to \infty$. The bounding property is now evident by combining these results and noting the linearity of (5.11). □

Theorem 2. *The SIP branch-and-bound procedure is finitely-convergent to $\epsilon-$optimality, such that $\alpha_k - \beta_k \leq \epsilon$ at termination and $\boldsymbol{x}^k$ approximately solves the SIP.*

Proof. The selection procedure of the SIP B&B procedure is bound-improving by construction as noted in Section 5.2.2. The consistency of the bounding operation was proved in Lemma 4. Theorem 2 now follows directly from Theorem IV.3 and Corollary IV.2 in [35]. □

5.3 Numerical Implementation and Results

The SIP B&B algorithm was implemented using the global NLP Branch-and-Bound code described in [63]. The upper and lower-bounding problems were formulated by hand, and solved at each node of the B&B tree using the SQP solver, NPSOL 5.0 [25]. Natural interval extensions were used to calculate the inclusion bounds $G^u_{n_k}(\boldsymbol{x}, P)$. The set of partition elements used to define the upper-bounding problem at a node occurring at level q was defined to be $\{P_\tau\},\ \tau \in T_q$, where $T_q = \{1, 2, \ldots, 2^q\}^{n_p}$. The index set S_q associated with each node occurring at level q was defined to be the set of upper right endpoints of the subintervals P_τ, i.e., $S_q = \{P^u_\tau\}, \tau \in T_q$. These sets can be shown to satisfy (5.1) and (5.4) respectively.

The SIP B&B procedure was applied to a number of nonconvex problems in the SIP literature. The relative and absolute tolerances for retaining a node in LibBandB were set to 0.01, and the NPSOL optimality tolerance was set to 10^{-7}. Default settings were used for all remaining parameters parameters in LibBandB and NPSOL. Examples 1,2 and 3 are from the test set in [14]. Example H (Example 4.1 from [32]) is a SIP for which the feasible set cannot be reproduced by applying constraints at a finite number of points in P. Columns 2, 3 and 4 in Table 1 show the number of nodes visited by the B&B procedure, the maximum depth of any node visited,

Problem	Total Nodes	Max. Depth	CPU	f	f^{RED}	f^{ICR}
1	27	13	2.66	-0.250	-0.250	-0.250
2	25	8	0.11	0.195	0.195	0.195
3	29	8	3.08	5.335	5.334	39.629
H	27	11	22.3	0	0	0

Table 5.1: Results from Global Solution of SIPs

and the CPU solution time in seconds on a 1 GB, 1.5 GHz, PIV PC running Linux. Column 5 shows the minimum value identified by the SIP B&B algorithm. Column 6 shows the solution values reported in [14] for Problems 1-3, and the known solution value for H. Column 7 shows the value of the initial feasible point identified by a single iteration of the ICR method (from Table 4 in [8]).

For the test problems studied here, the feasibility of the lower-bounding solutions in Column 6 was verified in [8]. Thus, the results in Column 6 are known to be the SIP minimum values. Column 5 in Table 1 procedure shows that the global SIP algorithm converges finitely to the known minimum value, even when the inclusion-constrained reformulation method significantly underestimates the feasible set of the SIP at the root node, as in Problem 3. For Problem H, Column 7 indicates that the SIP minimum value is first attained at the root node However, because of the weakness of the discretized relaxation, a significant number of additional nodes are visited before the lower-bounding solution converges to within the requested tolerance of the upper bound.

5.4 Conclusions

A branch-and-bound procedure for the global solution of smooth, nonconvex SIPs has been developed. Inclusion functions are used to construct a finite upper-bounding approximation, and a convexified discretization of the SIP is solved as the lower-bounding problem at each node in the tree. The method is finitely convergent to $\epsilon-$optimality provided the feasible set of the SIP has a non-empty interior. The

algorithm has been tested on a number of literature examples. The main drawback of the method, as it stands, is the rapid increase in the size of the lower and upper-bounding problems which must be solved as nodes of increasing depths are visited by the B&B procedure. A number of heuristics and alternate upper and lower-bounding problems for alleviating this problem are considered in Chapter 6.

Chapter 6

Future Work in Semi-infinite Programming

The purpose of this final chapter is to summarize a number of ideas pertaining to the future investigation of semi-infinite programming algorithms, and the development of a fully automated code for the global solution of nonconvex SIPs. Some of these points have already been touched upon in previous chapters, and are repeated here only for the sake of completeness.

6.1 Heuristics for the SIP B&B Procedure

A number of variants on the SIP B&B procedure described in Chapter 5 are worth investigating. It is not immediately obvious that any of these alternatives will improve the performance of the SIP B&B procedure uniformly for all applications. However, in certain cases, e.g., those for which the SIP-feasible set cannot be approximated reasonably well by a finite set of constraints, or those for which the natural interval extension is grossly inaccurate in predicting the inclusion bound G^u, one or more of these heuristics may significantly improve the rate of convergence of the SIP B&B, and/or the overall CPU time required to identify and verify a global miniumum.

6.1.1 Selection of Inclusion Function for ICR

The numerical studies presented in Chapter 5 were all performed using natural interval extensions to construct the ICR. However, it was shown in Chapter 4 that the exact form of the inclusion function used to reformulate the SIP significantly affects the rate of the convergence of the ICR method. Clearly, this has direct relevance to the rate of convergence of the SIP branch-and-bound procedure (in particular, the rate of convergence of the overall upper bound, α_k, to the SIP minimum value, f^{SIP}.) A number of modifications/alternatives to natural interval extensions were considered briefly in Chapter 4, e.g., Horner rearrangements, extended interval arithmetic and Taylor models [56, 48]. In addition, an optimal centered form may also be of interest in the calculation of first-order Taylor models. When evaluating Taylor models (or other centered forms) the midpoint of a given interval is typically used as the centering point. Such an approach has the advantage of requiring interval operations to be performed only on the symmetric intervals $P - p_d$. However, the midpoint of an interval P does not necessarily yield the tightest possible upper bound for a centered form inclusion. In [7] it is shown how an optimal centering point can be calculated for first order Taylor models and other centered forms which can be represented as $G(P) = L(P)(p - p_d)$. It is not clear whether optimal centering points can be calculated for higher-order Taylor models for which $L = L(P, p_d)$. In certain cases, the choice of known inclusions is limited/dictated by the form of g, e.g., polynomial/rational forms, etc. The choice of inclusion function warrants more detailed study in light of its impact on the overall rate of convergence of the B&B procedure. In particular, it is necessary to thoroughly understand the implications of 'mixing and matching' various inclusion heurstics, e.g., Horner rearrangements are not useful for symmetric intervals, and higher-order Taylor models do not yield inclusions of guaranteed increasing tightness when extended interval arithmetic is used [56].

From the standpoint of overall CPU time, it is not necessarily advantageous to use the tightest known inclusion for a particular constraint g; the preliminary results in Table 4.6 suggest that in some instances the additional cost of more complex

function evalutions overcompensates for the faster rate of convergence of a higher-order inclusion function. Furthermore, in a fully automated procedure, it may be necessary to take the cost of generating inclusions (e.g., automatic differentiation for Taylor models) into account.

6.1.2 MINLP and Nonsmooth Approximations of the SIP

The MINLP form of the inclusion-constrained reformulation was provided in Chapter 4, and its computational performance was compared to that of the smooth NLP form in Section 4.6.1. The MINLP approximation was not considered further because it did not appear to provide any computational advantage for the small scale problems studied in Chapter 4. However, as discussed in Section 4.4.2 the MINLP form has an important theoretical advantage over the smooth NLP form: the number of constraints in the finite reformulation scales polynomially rather than exponentially with n_c and n_e, the number of product and elementary function compositions used to factorize g respectively. For the upper-bounding step of the SIP B&B procedure, any SIP-feasible solution suffices as an acceptable outcome. Thus, when solved for a SIP-feasible solution rather than global minimimum, the MINLP form may prove useful for applications with complicated constraint forms, or those which require the B&B tree to be solved to significant depth ($q \gg 1$).

A similar reasoning justifies further investigation of the nonsmooth representation of the inclusion-constrained reformulation. Since nonsmoothness only arises here in the form of binary/finitely-indexed min/max functions, a gradient-based NLP solve may yield SIP-feasible points quite satisfactorily in practice.

6.1.3 MPEC Solution of ICR

As discussed in Section 4.4.2, a smooth, finitely-constrained NLP approximation of the SIP may derived by applying a bilevel programming approach to eliminate any min/max terms occuring in the ICR approximation. Since the reformulation is exact, the resulting mathematical program is clearly a valid upper-bounding problem for the

SIP B&B procedure. Moreover, unlike the smooth NLP representation, the number of nonlinear constraints does not increase exponentially with the complexity of the constraint function g in this case.

The primary drawback of this respresentation is that constraint qualifications are not satisfied because of the complementarity conditions which are imposed in (4.50) Consequently, this formulation cannot be solved using most conventional NLP methods which assume the KKT conditions to be necessary for optimality. However, (4.50) belongs to a class of problems known as mathematical programs with equilibrium constraints (MPECs). Over the past decade or so, a number of approaches including disjunctive programming, gap functions and penalty functions have been used to derive optimality conditions and numerical solution techniques for MPECS [50]. If a feasible point for (4.50) can be identified relatively cheaply using one of these methods, the bilevel programming approach may be useful for solving the ICR approximation for SIPs with a large number of constraints, or those which must be solved to significant depths in the B&B tree in order to achieve $\epsilon-$convergence.

6.1.4 Reduction-based Lower-Bounding Problem

As discussed in Chapter 4, both discretization and reduction-based methods generate (convergent) sequences of outer approximations to a SIP. In theory, either method may be used to generate a lower-bounding problem for the SIP B&B procedure. Discretized approximations were favored in the initial implementation of B&B code based simply on their wider (theoretical) applicability, and on ease of development. In contrast, reduction methods, and globalized reduction methods in particular, make relatively strong regularity assumptions on the SIP problem structure. However, it has been demonstrated repeatedly in literature studies [14, 55, 78] that reduction methods work well on many problems for which the theoretical requirements of the algorithms cannot be verified. In particular, reduction can be applied successfully to a pathological example in [32] for which discretization methods always yield an infeasible lower bounding solution at finite termination.

A reduction-based lower-bounding approach also offers the distinct advantage of

maintaining a relatively small fixed number of constraints in the lower-bounding problem. This advantage becomes correspondingly more important when the SIP B&B tree must be solved to significant depths such that $q \gg 1$, or when problems of larger scale are attempted. However, the computational benefit arising from a potentially more compact lower-bounding problem is partially offset by the cost of solving for the points in the constraint set T (for discretization methods, this set is either postulated a priori (as in Chapter 5), or else updated adaptively using relatively cheap function evaluations).

Strong results can be obtained for the application of reduction-based methods to convex problems [32]. This suggests that it may be useful to first convexify the SIP, and then generate the finitely-constrained reduction-based approximiation.

6.1.5 Bilevel Programming-based Reformulations

For SIPs where the number of constraints in the discretized approximation becomes prohibitively large, the upper-bounding ICR approximation will also suffer from an explosion in the number of constraints. This scenario motivates the investigation of an alternate upper-bounding problem for the SIP B&B scheme. One possibility is to solve the nonsmooth representation in (4.6) as a bilevel program. Smooth, convex bilevel programs have previously been solved in the literature by replacing the inner level problem with the corresponding KKT conditions [13]. This reformulation is exact (subject to the satisfaction of constraint qualifications for the inner problem), and has previously been solved using MPEC methods [50], and via conversion to a mixed-integer nonlinear program (MINLP) [13].

The general nonconvex bilevel problem has been addressed previously in [29] using a branch-and-bound framework. The authors claim that a valid upper-bounding problem for the bilevel program is obtained by replacing the inner problem with the corresponding KKT conditions. Since the KKT conditions are necessary, but not sufficient, for optimality in the nonconvex case, the suggested reformulation is actually a relaxation of the bilevel program. In fact, a valid lower-bounding problem for the SIP can be derived by replacing the inner maximization problem with the

corresponding KKT conditions, and then convexifying the resulting single level formulation. Likewise, for a bilevel program, the single inner problem may be replaced by the corresponding KKT conditions and then convexified to yield a lower bound. Both of these proposals deserve further study.

It remains to be clarified whether a rigorous upper-bounding problem can be identified for nonconvex bilevel programs, and if so, whether a feasible point can identified cheaply enough to make this approach viable in the context of an upper bounding problem for the SIP B&B scheme.

6.2 Towards Fully-automated Global Solution of SIPs

The subdivision results presented in Chapter 4 were obtained by formulating and solving a sequence of ICR approximations within the GAMS modeling environment [9]. The GAMS interfaces provides simple programming constructs and is extremely useful for testing new algorithms without investing significant development effort. Nonetheless, using the GAMS environment severely limits both the developer and the user in terms of speed, input/output compatibility and algorithmic development. For research purposes, it is more practical in the long run to develop a SIP application code based on (numerical/ I/O/data structure) libraries compiled from (largely) open sources.

The computational results for the B&B procedure presented in Chaper 5 were obtained using an in-house global NLP code [63]. For the structurally simple problems attempted in this thesis, it was possible to generate the appropriate finitely-constrained upper and lower-bounding approximations, and to derive the analytical derivative expressions used by the gradient-based NLP solver, by hand. However, for more realistic engineering applications, e.g., for calculating ranges of validity for point-reduced kinetic models, the global SIP code will have to be extended to perform most of these preliminary steps automatically.

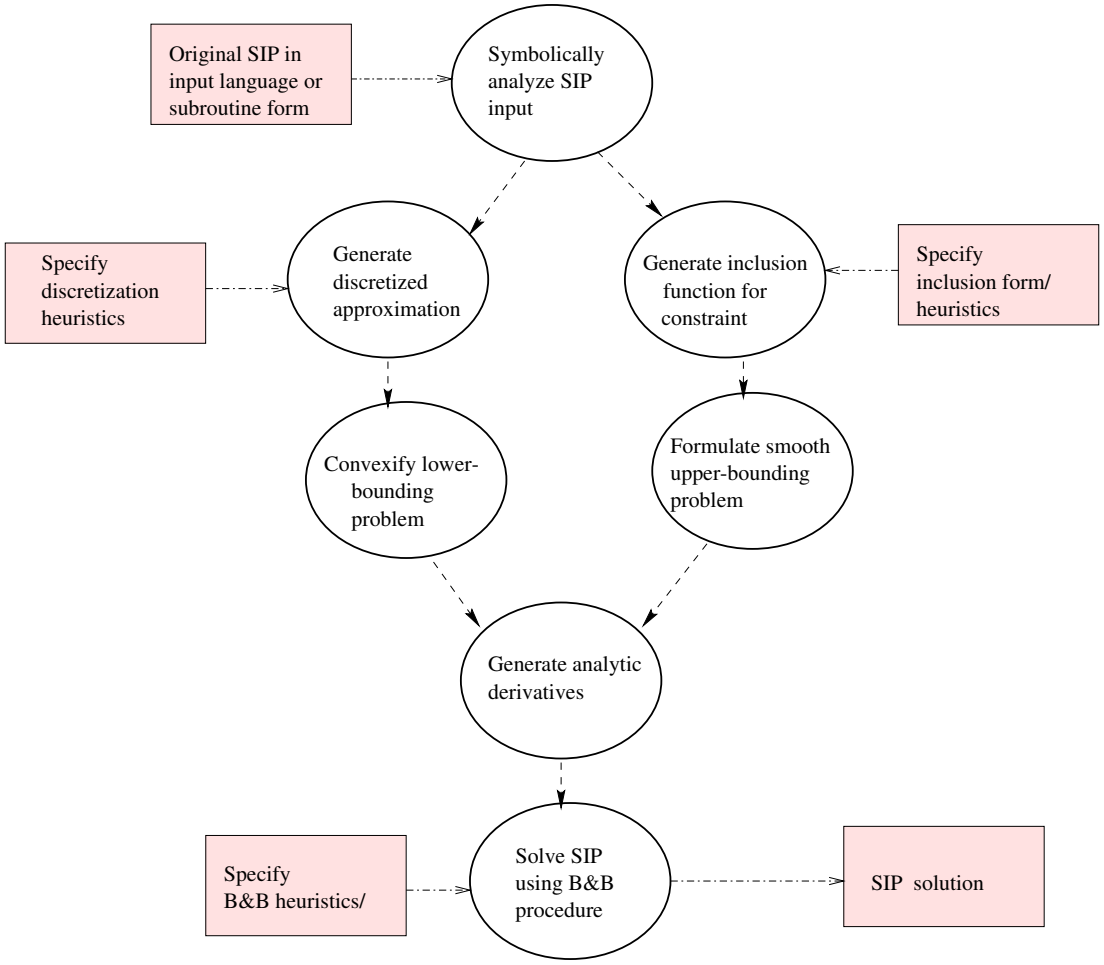

Figure 6-1: Schematic of Global SIP Solver

Figure 6-1 shows the various steps involved in automatically reformulating and solving a general, nonconvex SIP to $\epsilon-$optimality. However, it should be noted this is a somewhat optimistic view of the code structure. Unlike conventional B&B algorithms, the SIP B&B procedure described in Chapter 5 requires constraints to be added at each level of the B&B tree, i.e., when branching on a given node, the same upper and lower-bounding problems are not solved at a child node over a reduced domain. Rather, upper-and lower-bounding problems must be formulated by adding constraints to the ICR and discretized approximations. However, the *form* of these new constraints is identical to those used in the upper and lower-bounding problems solved at the parent node. The code structure in Figure 6-1 assumes that once ap-

propriate ICR and discretization approximations have been constructed for the root node in the B&B tree, the constraint set and the associated Jacobian matrix can be supplemented correctly at each node without reconstructing the smooth inclusion and reconvexifying the discretized approximation from scratch, at each child node. It remains to be seen whether this is a realistic assumption in practice; it is possible that some of the steps shown in Figure 6-1 as occuring prior to the branch-and-bound procedure may actually have to be peformed at each node within the branch-and-bound tree. Clearly, this has serious implications on the overall performance of the SIP solution procedure, and such limitations may in turn influence the choice of upper and lower-bounding problems used in the B&B procedure.

A comprehensive list of semi-infinite programs to which numerical methods have been applied is provided in [77]. In particular, a large-scale problem from robotic trajectory planning is described in [30]. Once the SIP B&B code has been developed further it should be tested on a more extensive test set than that studied in Chapter 5.

6.3 Extensions to Generalized Semi-infinite Programs and Integer Nonlinear Semi-infinite Programs

The original motivation for studying semi-infinite programming in this thesis was to construct optimal ranges of validity for point-reduced kinetic models. As described in Chapter 3, once a point-constrained problem has been solved to identify a valid reduced model, an optimal (maximum) range of validity may be identified by solving a generalized semi-infinite program (GSIP). Using the notation of Chapter 3, this

GSIP may be represented as:

$$\begin{aligned} &\max_{\boldsymbol{x}^l,\boldsymbol{x}^u} \|\boldsymbol{x}^u - \boldsymbol{x}^l\| \\ &\|\Gamma_{ref}(\boldsymbol{x}_{ref}) - \Gamma(\boldsymbol{x}_{ref})\| \leq 0, \quad \boldsymbol{x}^l \leq \boldsymbol{x}_{ref} \leq \boldsymbol{x}^u, \\ &\boldsymbol{x}^l,\ \boldsymbol{x}^u \in \mathbb{R}^{N_S} \end{aligned} \tag{6.1}$$

where the vectors $\boldsymbol{x}^u$ and $\boldsymbol{x}^l$ denote the upper and lower limits of the range of the validity for the reduced model. The GSIP in (6.1) has special structure since the limits of the interval on which the constraints are defined, are identified explicitly. It appears that (6.1) can be solved simply by treating the problem as a standard semi-infinite program in which the interval limits $\boldsymbol{x}^l,\ \boldsymbol{x}^u$ are treated as variables instead of parameters. In other words, the ICR method can be applied directly to (6.1) and yields the following finitely-constrained, upper-bounding approximation:

$$\begin{aligned} &\max_{\boldsymbol{x}^l,\boldsymbol{x}^u} \|\boldsymbol{x}^u - \boldsymbol{x}^l\| \\ &G^u_{ref}(X) - G^b(X) \leq 0 \\ &G^u(X) - G^b_{ref}(X) \leq 0 \\ &X = [\boldsymbol{x}^l,\ \boldsymbol{x}^u], \end{aligned} \tag{6.2}$$

where $G(X)$ and $G_{ref}(X)$ are inclusions for $\boldsymbol{\Gamma}$ and $\boldsymbol{\Gamma_{ref}}$ respectively on $[\boldsymbol{x}^l,\ \boldsymbol{x}^u]$. A branch-and-bound solution of (6.1) is, however, more problematic, since discretization cannot be applied to generate a valid lower bound (see discussion on general GSIPs).

An alternate model reduction formulation is possible; a detailed mechanism may be reduced subject to a postulated range of validity. When the scope of model reduction is limited to the reaction elimination procedure described in Chapter 2, this problem may be formulated as the following linear integer semi-infinite program (ISIP) :

$$\begin{aligned} &\min_{\boldsymbol{z}} \textstyle\sum_{i=1}^{N_R} z_i \\ &\|\Gamma_{ref}(\boldsymbol{x}) - \Gamma(\boldsymbol{z},\ \boldsymbol{x})\| \leq 0 \quad \forall \boldsymbol{x}^l \leq \boldsymbol{x} \leq \boldsymbol{x}^u \\ &\boldsymbol{z} = \{0,1\}^{N_R} \end{aligned} \tag{6.3}$$

A preliminary examination suggests that the SIP B&B procedure described in Chap-

ter 5 may be extended quite easily to ISIPs but extensions to GSIPs without special structure may require significant theoretical developments.

Semi-infinite programs with integer decision variables do not appear to have been studied previously in the literature. The general problem may be represented as:

$$\begin{aligned} &\min_{\boldsymbol{z}} f(\boldsymbol{z}) \\ &g(\boldsymbol{p}, \boldsymbol{z}) \leq 0 \quad \forall \boldsymbol{p} \in P \subset \mathbb{R}^{n_p}, |P| = \infty \\ &\boldsymbol{z} \in \{0,1\}^{n_z}. \end{aligned} \tag{6.4}$$

As before, inclusion functions may be used to define a subset of the feasible set of (6.4), and a discretization over a finite set $S_q \subset P$ yields a relaxation of the feasible set of (6.4). It appears that the ISIP in (6.4) can be solved to global optimality by simply replacing the NLP B&B framework described in Chapter 5 with an integer branch-and-bound framework. In particular, (6.3) can be reformulated so it is linear in $\boldsymbol{z}$. Consequently, the discretized relaxations solved at each node of the B&B tree are simply linear programs which are easily solved to global optimality. From a mathematical standpoint, it appears that reaction elimination subject to known ranges of validity is relatively easy problem to solve given the state of the art in semi-infinite programming.

The solution of GSIPs has been studied to a limited extent in the literature [67, 65], and may be represented as

$$\begin{aligned} &\min_{\boldsymbol{x}} f(\boldsymbol{x}) \\ &g(\boldsymbol{x}, \boldsymbol{p}) \leq 0 \qquad \forall \boldsymbol{p} \in P(\boldsymbol{x}) \\ &P(\boldsymbol{x}) = \{\boldsymbol{p} : \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{p}) \leq 0\}. \end{aligned} \tag{6.5}$$

Two theoretical challenges to the extension of the SIP B&B method are immediately obvious: since the infinite set P is now defined as a function of $\boldsymbol{x}$ rather than as a fixed n_p-dimensional interval, it is not clear that inclusion functions can be used to define a feasible subset of (6.5). In this case a bilevel programming-based or alternate upper-bounding problem may need be to investigated further; in fact,

connections between bilevel programming and GSIPs have already been studied in [66]. Furthermore, as shown in [67], discretization methods can no longer to used to generate convergent lower-bounding approximations to (6.5), and a reduction-based lower-bounding approach will probably be required. Conditions under which GSIPs may be reformulated as standard SIPs [67] have also been studied in the literature. If these reformulations can be made exact and implemented numerically, they may offer a more direct approach to solving GSIPs to global optimality.

Appendix A

Hydrogen Test Mechanism

(k = A T**b exp(-E/T))

	REACTIONS CONSIDERED			A	b	E
1.	H2+O2=>2OH			1.70E+13	0.0	24046.3
2.	2OH=>H2+O2			3.37E+10	0.3	14446.8
3.	OH+H2=>H2O+H			1.17E+09	1.3	1824.9
4.	H2O+H=>OH+H2			1.08E+11	0.9	9947.1
5.	O+OH=>O2+H			4.00E+14	-0.5	0.0
6.	O2+H=>O+OH			1.50E+17	-0.9	8684.8
7.	O+H2=>OH+H			5.06E+04	2.7	3165.6
8.	OH+H=>O+H2			3.76E+04	2.6	2250.8
9.	H+O2+M=>HO2+M			3.61E+17	-0.7	0.0
	H2O	Enhanced by	1.860E+01			
	H2	Enhanced by	2.850E+00			
10.	HO2+M=>H+O2+M			2.52E+19	-1.2	25214.6
	H2O	Enhanced by	1.860E+01			
	H2	Enhanced by	2.850E+00			
11.	OH+HO2=>H2O+O2			7.50E+12	0.0	0.0
12.	H2O+O2=>OH+HO2			3.29E+13	0.1	35410.0
13.	H+HO2=>2OH			1.40E+14	0.0	540.0

14.	2OH=>H+HO2			1.32E+10	0.9	18228.3
15.	O+HO2=>O2+OH			1.40E+13	0.0	540.0
16.	O2+OH=>O+HO2			4.97E+11	0.4	26913.1
17.	2OH=>O+H2O			6.00E+08	1.3	0.0
18.	O+H2O=>2OH			7.41E+10	1.0	9037.0
19.	2H+M=>H2+M			1.00E+18	-1.0	0.0
	H2O	Enhanced by	0.000E+00			
	H2	Enhanced by	0.000E+00			
20.	H2+M=>2H+M			3.34E+18	-1.0	52502.4
	H2O	Enhanced by	0.000E+00			
	H2	Enhanced by	0.000E+00			
21.	2H+H2=>2H2			9.20E+16	-0.6	0.0
22.	2H2=>2H+H2			3.07E+17	-0.6	52502.4
23.	2H+H2O=>H2+H2O			6.00E+19	-1.2	0.0
24.	H2+H2O=>2H+H2O			2.00E+20	-1.2	52502.4
25.	H+OH+M=>H2O+M			1.60E+22	-2.0	0.0
	H2O	Enhanced by	5.000E+00			
26.	H2O+M=>H+OH+M			4.91E+24	-2.3	60624.7
	H2O	Enhanced by	5.000E+00			
27.	H+O+M=>OH+M			6.20E+16	-6.0	0.0
	H2O	Enhanced by	5.000E+00			
28.	OH+M=>H+O+M			1.54E+17	-6.0	51587.7
	H2O	Enhanced by	5.000E+00			
29.	2O+M=>O2+M			1.89E+13	0.0	-899.9
30.	O2+M=>2O+M			1.76E+16	-0.4	59372.6
31.	H+HO2=>H2+O2			1.25E+13	0.0	0.0
32.	H2+O2=>H+HO2			5.97E+11	0.5	27287.8
33.	2HO2=>H2O2+O2			2.00E+12	0.0	0.0
34.	H2O2+O2=>2HO2			3.24E+13	-0.1	19317.0
35.	H2O2+M=>2OH+M			1.30E+17	0.0	22898.8

36. 2OH+M=>H2O2+M	1.08E+10	1.4	-3944.5
37. H2O2+H=>HO2+H2	1.60E+12	0.0	1912.4
38. HO2+H2=>H2O2+H	4.71E+09	0.6	9883.2
39. H2O2+OH=>H2O+HO2	1.00E+13	0.0	905.9
40. H2O+HO2=>H2O2+OH	2.71E+12	0.3	16998.9

NOTE: A units mole-cm-sec-, E units Kelvins

Variable	**Initial Value**
P_0	1 atm
T_0	1000 K
N_{H_2}	0.4 mol
N_{O_2}	0.2 mol
N_H, N_O, N_{OH}	
N_{HO_2}, $N_{H_2O_2}$, N_{H_2O}	0.0002 mol

Table A.1: Initial conditions for hydrogen combustion reference trajectory

	t_1	t_2	t_3	t_4
DR	0.0038	0.000077	$9.26 \cdot 10^{-6}$	$6.17 \cdot 10^{-5}$
PCAS	0.0031	0.0062	0.000078	0.000039
PCAF	0.0045	0.0085	0.0016	0.0069

Table A.2: Final threshold values for κ (DR), ζ (PCAS), and ξ (PCAF), at each point t_i

Appendix B

SIP Test Problems

1.

$$P = [0, 2]$$
$$f = \tfrac{1}{3}x_1^2 + \tfrac{1}{2}x_1 + x_2^2 - x_2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1^2 + 2x_1x_2p^2 - \sin(p).$$

2.

$$P = [0, 1]$$
$$f = \tfrac{1}{3}x_1^2 + x_2^2 + \tfrac{1}{2}x_1$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = (1 - x_1^2p^2)^2 - x_1p^2 - x_2^2 + x_2.$$

3.

$$P = [0, 1]$$
$$f = x_1^2 + x_2^2 + x_3^2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1 + x_2e^{x_3p} + e^{2p} - 2\sin(4p).$$

4.

$$P = [0, 1]$$
$$f = \sum_{i=1}^{n_x} \tfrac{x_i}{i}$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = \tan(p) - \sum_{i=1}^{n_x} x_ip^{i-1}.$$

5.

$$P = [0, 1]$$
$$f = \sum_{i=1}^{3} e^{x_i}$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = \tfrac{1}{1+p^2} - x_1 - x_2 p - x_3 p^2.$$

6.

$$P = [0, 1]$$
$$f = (x_1 - 2x_2 + 5x_2^2 - x_2^3 - 13) + (x_1 - 14x_2 + x_2^2 + x_2^3 - 29)^2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1^2 + 2x_2 p^2 + e^{x_1+x_2} - e^p.$$

7.

$$P = [0, 1]^2$$
$$f = x_1^2 + x_2^2 + x_3^2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1(p_1 + p_2^2 + 1) + x_2(p_1 p_2 - p_2^2) + x_3(p_1 \cdot p_2 + p_2^2 + p_2) + 1.$$

8.

$$P = [0, 1]^2$$
$$f = x_1 + \tfrac{1}{2}x_2 + \tfrac{1}{2}x_3 + \tfrac{1}{2}x_4 + \tfrac{1}{4}x_5 + \tfrac{1}{3}x_6$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = e^{p_1^2+p_2^2} - (x_1 + x_2 p_1 + x_3 p_2 + x_4 p_1^2 + x_5 p_1 p_2 + x_6 p_2^2).$$

9.

$$P = [-1, 1]^2$$
$$f = -4x_1 - \tfrac{2}{3}(x_4 + x_6)$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1 + x_2 p_1 + x_3 p_2 + x_4 p_1^2 + x_5 p_1 p_2 + x_6 p_2^2 - 3 - (p_1 - p_2)^2(p_1 + p_2)^2.$$

K.

$$P = [0, \pi]$$
$$f = x_2^2 - 4x_2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1 \cos(p) + x_2 \sin(p) - 1$$

L.

$$P = [0, \pi]$$
$$f = (x_1 + x_2 - 2)^2 + (x_1 - x_2)^2 + 30 \min(0, (x_1 - x_2))^2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1 \cos(p) + x_2 \sin(p) - 1.$$

M.

$$P = [0, \pi]$$
$$f = (x_1 - 2)^2 + x_2^2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = x_1 \cos(p) + x_2 \sin(p) - 1.$$

N.

$$P = [-1, 1]$$
$$f = x_2$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = 2x_1^2 p^2 - p^4 + x_1^2 - x_2.$$

S.

$$P = [0, 1]^{n_p}$$
$$f = x_1 x_2 + x_2 x_3 + x_3 x_4$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = 2(x_1^2 + x_2^2 + x_3^2 + x_4^2) - 6 - 2n_p + \sin(p_1 - x_1 - x_4) + \sin(p_2 - x_2 - x_3)$$
$$+ \sin(p_3 - x_1) + \sin(2p_4 - x_2) + \sin(p_5 - x_3) + \sin(2p_6 - x_4).$$

This problem was solved for $n_p = 3, 4, 5, 6$. When $p < 6$, any terms in g containing $p_j, j \geq n_p$ were omitted.

U.

$$P = [-1, 1]^6$$
$$f = \sum_{i=1}^{4} \tfrac{1}{10} x_i^2 - x_i$$
$$g(\boldsymbol{x}, \boldsymbol{p}) = \tfrac{x_4}{5} \sin(30 p_1 \sin(x_1) + 30 p_2 \cos(x_2)) + \tfrac{x_3}{10} \sin(\tfrac{p_1 p_2}{10} + p_3 x_1 + p_4 x_2 + p_5 x_3 + p_6 x_4 - 4)$$

Bibliography

[1] Warren P. Adams and Hanif D. Sherali. Linearization strategies for a class of zero-one mixed integer programming problems. *Operations Research*, 38:217–226, 1990.

[2] Warren P. Adams and Hanif D. Sherali. Mixed-integer bilinear programming problems. *Mathematical Programming*, 59:279–305, 1993.

[3] G. Alefeld and G. Mayer. Interval analysis: theory and applications. *Journal of Computational and Applied Mathematics*, 121:421–464, 2000.

[4] Ioannis P. Androulakis. Mechanism reduction based on an integer programming approach. *AIChE Journal*, 46:361–371, 2000.

[5] Egon Balas, Sebastian Ceria, and Gerard Cornuejols. Mixed 0-1 programming by lift and project in a branch and bound framework. *Management Science*, 42:1229–1246, 1996.

[6] Egon Balas, Sebastian Ceria, Gerard Cornuejols, and N. Natraj. Gomory cuts revisited. *Operations Research Letters*, 19:1–9, 1996.

[7] Eckart Baumann. Optimal centered forms. *BIT*, 28:80–87, 1988.

[8] Binita Bhattacharjee, William H. Green, and Paul I. Barton. Interval methods for semi-infinite programs. Submitted to Computational Optimization and Applications, July 2003.

[9] Anthony Brooke, David Kendrick, Alexander Meeraus, and Ramesh Raman. *GAMS: A User's Guide*. GAMS Development Corporation, Washington D.C., December 1998. http://www.gams.com/solvers/solvers.htm.

[10] Nancy J. Brown, Guoping Li, and Michael L. Koszykowski. Mechanism reduction via principal component analysis. *International Journal Chemical Kinetics*, pages 393–414, 1997.

[11] P. N. Brown, G. D. Byrne, and A. C. Hindmarsh. VODE: A variable coefficient ODE solver. *SIAM Journal of Scientific and Statistical Computing*, 10:1038–1051, 1989.

[12] M. Caracotsios and W.E. Stewart. Sensitivity analysis of initial value problems with mixed ODEs and algebraic constraints. *Computers and Chemical Engineering*, 9:359–365, 1985.

[13] P.A. Clark and A.W. Westerberg. Bilevel programming for steady-state chemical process design-I. fundamentals and algorithms. *Computers and Chemical Engineering*, 14:87–97, 1990.

[14] I.D. Coope and G.A. Watson. Projected Lagrangian algorithm for semi-infinite programming. *Mathematical Programming*, 32:337–356, 1985.

[15] R.J. Cukier, H.B. Levine, and K.E. Shuler. Nonlinear sensitivity analysis of multi-parameter model systems. *Journal of Computational Physics*, 26:1–42, 1978.

[16] H. J. Curran, P. Gaffuri, William J. Pitz, and Charles K. Westbrook. A comprehensive modeling study of n-heptane oxidation. *Combustion and Flame*, 114:149–177, 1998. http://www-cms.llnl.gov/combustion/combustion2.html.

[17] Arne Drud. *CONOPT*. ARKI Consulting and Development, Bagsvaerdvej 246-A, DK-2880, Bagsvaerd, Denmark. http://www.gams.com/solvers/solvers.htm.

[18] R. Dussel and B. Schmitt. Die berechnung von schranken fur den wertebereich eines polynoms in einem intervall. *Computing*, 6:35–60, 1970.

[19] Keith Edwards, T. F. Edgar, and V. I. Manousiouthakis. Kinetic model reduction using genetic algorithms. *Computers and Chemical Engineering*, 22:239–246, 1998.

[20] Keith Edwards, T. F. Edgar, and V. I. Manousiouthakis. Reaction mechanism simplification using mixed-integer nonlinear programming. *Computers and Chemical Engineering*, 24:67–79, 2000.

[21] James E. Falk and Richard M. Soland. An algorithm for separable nonconvex programming problems. *Managment Science*, 15:550–569.

[22] William F. Feehery, John E. Tolsma, and Paul I. Barton. Efficient sensitivity analysis of large-scale differential-algebraic systems. *Applied Numerical Mathematics*, 25:41–54, 1997.

[23] Christodoulos A. Floudas. *Nonlinear and Mixed-Integer Programming*. Oxford University Press, 1995.

[24] Michael Frenklach and Hai Wang. Detailed reduction of reaction mechanisms for flame modeling. *Combustion and Flame*, 87:365–370, 1991.

[25] Philip E. Gill, Walter Murray, Michael A. Saunders, and Margaret H. Wright. *User's Guide for NPSOL 5.0:A FORTRAN Package for Nonlinear Programming*, 1998.

[26] Fred Glover. Converting the 0-1 polynomial programming problem to a 0-1 linear problem. *Operations Research*, 22:180–182, 1974.

[27] Fred Glover and Eugene Woolsey. Imporved linear integer programming formulations of integer nonlinear programs. *Management Science*, 22:455–460, 1975.

[28] Andreas Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, chapter 10. Frontiers in Applied Mathematics. SIAM, Philadelphia, PA, 2000.

[29] Zeynep H. Gümüş and Cristodoulos A. Floudas. Global optimization of nonlinear bilevel programming problems. *Journal of Global Optimization*, 20:1–31, 2001.

[30] E. Haaren-Retagne. *A Semi-infinite Programming Algorithm for Robot Trajectory Planning*. PhD thesis, University of Trier, 1992.

[31] K.P. Haleman and I.E. Grossmann. Optimal process design under uncertainty. *AIChE Journal*, 29:425–433, 1983.

[32] R. Hettich and K.O. Kortanek. Semi-infinite programming: Theory, methods and applications. *SIAM Review*, 35:380–429, 1993.

[33] R. Horst. A general class of branch-and-bound methods in global optimization with some new approaches for concave minimization. *Journal of Optimization Theory and Applications*, 51:271–291, 1986.

[34] R. Horst. Deterministic global optimization with partition sets whose feasibility is not known: Application to concave minimization, reverse convex constraints, DC-programming, and Lipschitzian optimization. *Journal of Optimization Theory and Applications*, 58:11–37, 1988.

[35] Reiner Horst and Hoang Tuy. *Global Optimization: Deterministic Approaches*. Springer-Verlag, 1996.

[36] J.T. Hwang. Sensitivity analysis in chemical kinetics by the method of polynomial approximations. *International Journal of Chemical Kinetics*, page 959, 1983.

[37] J.T. Hwang, E.P. Dougherty, S. Rabitz, and H. Rabitz. Green's function method of sensitivity analysis in chemical kinetics. *Journal of Chemical Physics*, 69:5180–5191, 1978.

[38] ILOG. *GAMS/CPLEX 7.5 User Notes*. http://www.gams.com/solvers/solvers.

[39] R. J. Kee, F. M. Rupley, and J. A. Miller. CHEMKIN II: A FORTRAN chemical kinetics package for the analysis of gas-phase chemical kinetics. Technical Report SAND89-8009, Sandia National Laboratories, 1990.

[40] R. Krawczyk and K. Nickel. Die zentrische form in der intervallarithmetik, ihre quadratische konvergenz und ihre inkhlusionsiotonie. *Computing*, 28:117–132, 1982.

[41] S.H. Lam and D.A. Goussis. Understanding complex chemical kinetics with computational singular perturbation. In *Twenty-Second Symposium on Combustion*, page 931, 1988.

[42] C.T. Lawrence and A.L. Tits. Feasible sequential quadratic programming for finely discretized problems from SIP. In R. Reemsten and J. Ruckmann, editors, *Semi-infinite Programming*, pages 159–193. Kluwer Academic Publishers, Dordrecht, Netherlands, 1998.

[43] Y. Li and D. Wang. A semi-infinite programming model for earliness/tardiness production planning with simulated annealing. *Mathematical and Computer Modelling*, 26:35–42, 1997.

[44] A. E. Lutz, R. J. Kee, and J. A. Miller. SENKIN: A FORTRAN program for predicting homogenous gas phase chemical kinetics with sensitivity analysis. Technical Report SAND87-8248, Sandia National Laboratories, 1988.

[45] Uni Maas and Stephen B. Pope. Simplifying chemical kinetics: Intrinsic low-dimensional manifolds in chemical composition space. *Combustion and Flame*, 88:239–264, 1992.

[46] K. Makino and M. Berz. *Computational Differentiation: Techniques, Applications, and Tools*, chapter Remainder Differential Algebras and Their Applications. SIAM, Philadelphia, PA, 1996.

[47] Garth P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I: Convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[48] R.E. Moore. *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, PA, 1979.

[49] National Superconducting Cyclotron Laboratory, Michigan State University, East Lansing, MI 48824. *COSY INFINITY Version 8 Reference Manual*, 1997. http://www.cosy.nscl.msu.edu.

[50] Jiri Outrata, Michal Kocvara, and Jochem Zowe. *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints*. Kluwer Academic Publishers, 1998.

[51] Linda Petzold and Timothy Maly. Numerical methods and software for sensitivity analysis of differential-algebraic systems. *Applied and Numerical Mathematics*, 20:57–59, 1996.

[52] T.H. Pierce and R.J. Cukier. Global nonlinear sensisitivity analysis using walsh functions. *Journal of Computational Physics*, 41:427–443, 1981.

[53] E. Polak. On the mathematical foundations of nondifferentiable optimization in engineering design. *SIAM Review*, 29:21–89, 1987.

[54] Stephen B. Pope and B. Yang. Treating chemistry in combustion with detailed mechanisms - in situ adaptive tabulation in principal directions. *Combustion and Flame*, 112:85–112, 1998.

[55] C.J. Price and J.D. Coope. Numerical experiments in semi-infinite programming. *Computational Optimization and Applications*, 6:169–189, 1996.

[56] H. Ratschek and J. Rokne. *Computer Methods for the Range of Functions*. Ellis Horwood Limited, Chichester, England, 1984.

[57] R. Reemsten and S. Gorner. Numerical methods for semi-infinite programming: A survey. In R. Reemsten and J. Ruckmann, editors, *Semi-infinite Programming*, pages 195–275. Kluwer Academic Publishers, Dordrecht, Netherlands, 1998.

[58] N. V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8:201–205, 1996.

[59] N.V. Sahinidis and M. Tawarmalani. *GAMS/BARON 5.0: Global Optimization of Mixed-Integer Nonlinear Programs*, August 2002. http://www.gams.com/solvers/solvers.htm.

[60] Douglas A. Schwer, Pisi Lu, and William H. Green. Adaptive chemistry approach to modeling complex kinetics in reacting flows. *Combustion and Flame*, 133:451–465, 2003.

[61] Douglas A. Schwer, Pisi Lu, William H. Green, and Viriato Semiao. A consistent-splitting approach to computing stiff steady-state reacting flows with adaptive chemistry. *Combustion Theory and Modelling*, 7:383–399.

[62] Douglas A. Schwer, John E. Tolsma, William H. Green, and Paul I. Barton. On upgrading the numerics in combustion chemistry codes. *Combustion and Flame*, 128:270–291, 2002.

[63] Adam B. Singer. *LibBandB.a Version 3.1 Manual*. Massachusetts Institute of Technology, Cambridge, MA, 2003.

[64] Gregory P. Smith, David M. Golden, Michael Frenklach, Nigel W. Moriarty, Boris Eiteneer, Mikhail Goldenberg, C. Thomas Bowman, Ronald K. Hanson, Soonho Song, William C. Gardiner, Jr., Vitali V. Lissianski, and Zhiwei Qin. http://www.me.berkeley.edu/gri-mech/.

[65] Oliver Stein and Georg Still. On optimality conditions for generalized semi-infinite programming problems. *Optimization Theory and Applications*, 104:443–458, 2000.

[66] Oliver Stein and Georg Still. On generalized semi-infinite optimization and bilevel optimization. *European Journal of Operations Research*, 142:444–462, 2002.

[67] G. Still. Generalized semi-infinite programming: Theory and methods. *European Journal of Operations Research*, 119:303–313, 1999.

[68] R.G. Susnow, A.M. Dean, W.H. Green, and P. Peczak. Rate-based construction of kinetic models for complex systems. *Journal of Physical Chemistry*, 20:3731, 1997.

[69] Y. Tanaka, M. Fuksima, and T. Ibaraki. A globally convergent SQP method for semi-infinite nonlinear optimization. *Journal of Computational and Applied Mathematics*, 23:141–153, 1988.

[70] John E. Tolsma and Paul I. Barton. DAEPACK: An open modeling environment for legacy codes. *Ind. Eng. Chem. Res.*, 39:1826–1839, 2000.

[71] Shaheen R. Tonse, Nigel W. Moriarty, Nancy J. Brown, and Michael Frenklach. PRISM: Piecewise reusable implementation of solution mapping. An economical strategy for chemical kinetics. *Israeli Journal of Chemistry*, 39:97–106, 1999.

[72] Tamas Turanyi. *KINALC: Kinetic Analysis of Reaction Systems*. http://www.chem.leeds.ac.uk/Combustion/Combustion.html.

[73] Tamas Turanyi. *MECHMOD: Software Aid to the Development of Reaction Mechanisms*. http://www.chem.leeds.ac.uk/Combustion/Combustion.html.

[74] Tamas Turanyi. Parameterisation of reaction mechanisms using orthonormal polynomials. *Computers and Chemical Engineering*, 18:45, 1994.

[75] Tamas Turanyi, T. Berces, and S. Vajda. Reaction rate analysis of complex kinetics systems. *International Journal of Chemical Kinetics*, 21:83–99, 1989.

[76] S. Vajda, P. Valko, and Tamas Turanyi. Principal component analysis of kinetic models. *International Journal of Chemical Kinetics*, 17:55–81, 1985.

[77] A.I.F. Vas, E.M.G.P. Fernandes, and M.P.S.F. Gomes. SIPAMPL 2.0: Semi-infinite programming with AMPL. Technical report, Universidade do Minho, Braga, Portugal, 2002. http://www.norg.uminho.pt/aivaz.

[78] G.A. Watson. Numerical experiments with globally convergent methods for semi-infinite programming problems. In A.V. Fiacco and K.O. Kortanek, editors, *Semi-Infinite Programming and Applications, Proceedings of an International Symposium*, pages 193–205. Springer-Verlag, Heidelberg, Germany, 1983.

[79] W. Zhu and Linda R. Petzold. Model reduction for chemical kinetics: An optimization approach. *AIChE Journal*, 45:869–886, 1999.