Reduced distillation models via stage aggregation

Andreas Linhart[*], Sigurd Skogestad
Department of Chemical Engineering,
Norwegian University of Science and Technology,
N-7491 Trondheim, Norway

March 24, 2009

Abstract

A method for deriving computationally efficient reduced nonlinear distillation models is proposed, which extends the aggregated modeling method of Lévine and Rouchon (1991) to complex models. The column dynamics are approximated by a low number of slow dynamic aggregation stages connected by blocks of steady-state stages. This is achieved by simple manipulation of the left-hand sides of the differential equations. In order to reduce the computational complexity of the resulting DAE system, the algebraic equations resulting from the reduction procedure are replaced by interpolation in tables or polynomial approximations. The resulting reduced model approximates the original dynamic model very accurately, and increases the simulation speed by about an order of magnitude. This makes the reduced models attractive for real-time optimizing control applications. The numerical properties of the models and possible improvements are discussed.

Keywords

Model reduction; Distillation; Aggregated modeling; Real-time optimizing control; Mathematical modeling; Dynamic simulation

[*] Corresponding author: andreas.linhart@chemeng.ntnu.no; phone +47 73550346

1 Introduction

(shortened) This study describes and analyzes a model reduction method for staged distillation column models. Reduced models are used for system analysis and controller design, and for speeding up simulations. The latter is much desired for model predictive control (Allgöwer and Zheng 2000, Qin and Badgwell, 2003) and dynamic real-time optimization (Schlegel, 2005) applications. Numerous model reduction methods for linear (Antoulas, 2005) and nonlinear systems (Marquardt, 2001, van den Berg, 2005) have been described in literature.

The model reduction method presented in this work extends the aggregated modeling method of Lévine and Rouchon (1991). The original method was developed as an improvement of the compartmental modeling method of Benallou et al. (1986), and used a very simple distillation model assuming constant stage holdups, constant molar flows and constant relative volatility. The column is partitioned into compartments, each of which comprises a number of consecutive stages. The dynamics of a compartment is approximated by one dynamic stage, which is assigned the total compartment holdup, while the remaining stages of the compartment are treated as in steady-state. This is achieved by a state transformation for the compartment variables, and a subsequent quasi-steady-state approximation of the fast equations of the transformed system. This approach has been used recently by Khowinij et al. (2004, 2005), and Bian et al. (2005) to derive reduced models of a distillation column with variable stage holdups, with the objective of obtaining reduced models that increase the simulation speed. They conclude that a tailor-made DAE solver is necessary to significantly speed up the simulations.

Likewise, it was shown by Linhart and Skogestad (2009) that applying the original method in combination with an ordinary DAE solver does not increase the simulation speed. The reason for this is that the method converts the majority of the dynamic equations of the full model into algebraic equations, which does not change the overall size of the system. Since in a DAE solver, dynamic and algebraic equations are treated very similarly (Hairer and Wanner, 2002, Ascher and Petzold, 1998), no gain in computation speed can be expected. Alternatively, Linhart and Skogestad (2009) show that the algebraic equations resulting from the reduction procedure can be eliminated from the model due to the banded Jacobian structure of the reduced model, which yields a much smaller ODE or DAE system. This model now yields a significant improvement in computational performance. In addition, it was shown that the method of Lévine and Rouchon can be interpreted to be basically compartment-free. This means that only the dynamic aggregation stages have to be specified, but no partition of the column into compartments is necessary.

The model reduction method presented in this study extends the original method of Lévine and Rouchon (1991) and the extension of Khowinij et al. (2005) and Bian et al. (2005) in the following aspects:

1. The notion of "compartments" is abandoned; only the specification of "aggregation stages" is necessary. The aggregated holdup factor multiplying the left-hand sides of their dynamic equations can be chosen freely to obtain optimal dynamic approximation of the original dynamics.

2. The method can be applied to all kinds of staged processes with mass and energy balances, and complex hydraulic and thermodynamic relationships. No simplification of the original system prior to the model reduction procedure is needed.

3. The algebraic equations resulting from the reduction procedure are eliminated from the reduced model and replaced by table interpolation or polynomial approximations. This is the crucial step to obtain computationally superior reduced models. Elimination is, however, only practical if there is a relatively low number of dynamic variables on each stage, which generally means a low number of components in the process.

4. The physical interpretation of the reduction principle is different. Instead of comparing the different time-scales of the slow compartment dynamics and fast stage dynamics, the method can be interpreted by making signal transport through the steady-state stages infinitely fast, and slowing down the column dynamics by assigning large aggregated holdup factors to the aggregation stages. Although related, the method is no true singular perturbation method (Kokotovic et al., 1986, Linhart and Skogestad, 2009).

This paper is organized as follows. In section 2, the full distillation model is introduced. Important structural and implementation issues of the model are discussed. Section 3 describes the derivation of the reduced models from the full model. In a first step, by a simple manipulation of the left-hand sides of the differential equations of the full column, a reduced model of the same size as the original model is obtained. In a second step, the resulting algebraic equations are eliminated from the model and replaced by more efficient approximations such as table interpolations. As the second step is crucial for the performance of the reduced model, this part is described in more detail. In section 4, the approximation quality and computational performance of the reduced model is investigated. For this purpose, simulations with fast changes in the input variables of the models over a range of simulation tolerances are performed. The accuracy of the reduced models is compared to the original model, and is set into relation with the simulation speed. An analysis of the numerical behavior and of the distribution of computational complexity in the models and the solver is given. Finally, the advantages and disadvantages of the stage aggregation method, possible improvements and applications, and a brief comparison to other model reduction methods for distillation models are discussed in section 5.

2 Full model

2.1 System and modeling assumptions

The distillation column investigated in this study is a high-purity distillation column with 92 stages, a reflux drum with a total condenser, and a reboiler. The case-study model in this study uses a binary mixture, but the model description and the model reduction procedure is for a multi-component mixture. Ideal stages with perfect mixing and vapor-liquid equilibrium on each stage are assumed.

2.2 Mathematical description

For notational convenience, the reboiler and reflux drum are written as stages 1 and N, respectively.

For a mixture with N_c components, the state of each stage is described by N_c+1 dynamic variables: M_i^{tot} (total mole number on stage i), $\mathbf{M}_i$ (vector of $N_c - 1$ component moles on stage i), and U_i^{tot} (total internal energy on stage i). Since the sum of the N_c components gives the total holdup M_i^{tot}, this formulation is equivalent to including all N_c components in the $\mathbf{M}_i$ vector. The dynamic evolution of each state is governed by a differential balance equation. In addition, there is a large number of algebraic equations, including thermodynamic relationships for the vapor-liquid equilibrium.

2.2.1 Dynamic balance equations

On each stage, $N_c + 1$ balance equations can be formulated. The balance equations for the stages except the reboiler, condenser and feed stage ($2 \leq i \leq N-1, i \neq i_F$) read

$$\begin{aligned} \dot{M}_i^{tot} &= L_{i-1} + V_{i+1} - L_i - V_i, && (1) \\ \dot{\mathbf{M}}_i &= L_{i-1}\mathbf{x}_{i-1} + V_{i+1}\mathbf{y}_{i+1} - L_i\mathbf{x}_i - V_i\mathbf{y}_i, && (2) \\ \dot{U}_i^{tot} &= L_{i-1}h_{i-1}^L + V_{i+1}h_{i+1}^V - L_ih_i^L - V_ih_i^V - Q_i^{hl}. && (3) \end{aligned}$$

The balance equations for the feed stage i_F read

$$\begin{aligned}
\dot{M}^{tot}_{i_F} &= L_{i_F-1} + V_{i_F+1} - L_{i_F} - V_{i_F} + F, & (4)\\
\dot{\mathbf{M}}_{i_F} &= L_{i_F-1}\mathbf{x}_{i_F-1} + V_{i_F+1}\mathbf{y}_{i_F+1} - L_{i_F}\mathbf{x}_{i_F} - V_{i_F}\mathbf{y}_{i_F} + F\mathbf{z}_F, & (5)\\
\dot{U}^{tot}_{i_F} &= L_{i_F-1}h^L_{i_F-1} + V_{i_F+1}h^V_{i_F+1} - L_{i_F}h^L_{i_F} - V_{i_F}h^V_{i_F} - Q^{hl}_{i_F} & (6)\\
&\quad +Fh_F.
\end{aligned}$$

The balance equations for the reflux drum with total condenser ($i = 1$) read

$$\begin{aligned}
\dot{M}^{tot}_1 &= V_{top} - (R + D), & (7)\\
\dot{\mathbf{M}}_1 &= V_{top}\mathbf{y}_2 - (R + D)\mathbf{x}_1, & (8)\\
\dot{U}^{tot}_1 &= V_{top}h^V_2 - (R + D)h^L_1 + Q_{condenser}. & (9)
\end{aligned}$$

The balance equations for the reboiler ($i = N$) read

$$\begin{aligned}
\dot{M}^{tot}_N &= L_{N-1} - B - V_N, & (10)\\
\dot{\mathbf{M}}_N &= L_{N-1}\mathbf{x}_{N-1} - B\mathbf{x}_N - V_N\mathbf{y}_N, & (11)\\
\dot{U}^{tot}_N &= L_{N-1}h^L_{N-1} - Bh^L_N - V_Nh^V_N + Q_{reboiler}. & (12)
\end{aligned}$$

The variables used in the above equations are explained in table 3. Note that $\mathbf{M}_i$, $\mathbf{x}_i$, $\mathbf{y}_i$ and $\mathbf{z}_F$ are vectors of length $N_c - 1$, except in the binary case, where they are scalars.

2.2.2 Algebraic relations for sum of phases

(shortened) The intensive variables for the individual phases $\mathbf{x}_i$, $\mathbf{y}_i$, h^L_i, h^V_i must satisfy some algebraic relations, since the sum of the phases make up the total holdup. The sum of the total mass, component masses, energy and volume of the phases on stage i can be written as

$$\begin{aligned}
M^{tot}_i &= M^L_i + M^V_i, & (13)\\
\mathbf{M}_i &= M^L_i\mathbf{x}_i + M^V_i\mathbf{y}_i, & (14)
\end{aligned}$$

$$
\begin{aligned}
U_i^{tot} &= M_i^L h_i^L + M_i^V h_i^V - p_i V_i, && (15)\\
V_i &= M_i^L v_i^L + M_i^V v_i^V, && (16)
\end{aligned}
$$

where V_i is the total volume of stage i (which is assumed constant), and M_i^L and M_i^V the stage liquid and vapor masses, respectively. To reduce the number of algebraic equations that need to be solved by the DAE solver, note that equations (13) and one of equations (14)-(16) may be solved to obtain explicit expressions for M_i^L and M_i^V. The simplest choice is to combine eqs. (13) and (16), to get

$$
\begin{aligned}
M_i^L &= (V_i - M_i^{tot} v_i^V)/(v_i^L - v_i^V), && (17)\\
M_i^V &= M_i^{tot} - M_i^L, && (18)
\end{aligned}
$$

where v_i^L and v_i^V are the specific volumes of liquid and vapor phase, respectively. The remaining N_c algebraic equations that need to be solved are then (14) and (15).

2.2.3 Algebraic thermodynamic relationships

The vapor-liquid equilibrium on stage i gives N_c algebraic relations (one for each component):

$$
f_i^L(\mathbf{x}_i, p_i, T_i) = f_i^V(\mathbf{y}_i, p_i, T_i). \qquad (19)
$$

In this study, the thermodynamic quantities f_i^L, f_i^V, $\mathbf{x}_i$, $\mathbf{y}_i$, h_i^L, h_i^V, v_i^L, v_i^V, ρ_i^L and ρ_i^V are obtained by means of the Soave-Redlich-Kwong equations of state (Reid et al., 1997).

2.2.4 Algebraic hydraulic relationships

(shortened) The liquid flows L_i are calculated by means of a modified Francis weir equation (Green and Perry, 2007)

$$
L_i = \gamma \rho_i^L |l_i/\beta - h_{W,i}|^{1.5}, \qquad (20)
$$

where β and γ are geometry-dependent factors, l_i the liquid level, and $h_{W,i}$ is the weir height of stage i, respectively.

The vapor flows V_i are calculated by

$$V_i \quad = \gamma\sqrt{|p_i - p_{i-1} - \rho^L_{i-1} g l_{i-1}| \rho^V_i}, \tag{21}$$

where g is the standard gravity.

2.2.5 Algebraic equations for heat loss

The heat loss of a tray to the environment is modeled by a linear heat transfer equation

$$Q^{hl}_i = \alpha_i (T_i - T_{environment}), \tag{22}$$

where α_i is the heat transduction coefficient through the outer wall of stage i. The heat loss is frequently neglected ($\alpha_i = 0$) in distillation modeling.

2.2.6 Algebraic equations for condenser cooling

The cooling of the condenser is modeled as

$$Q_{condenser} = -V_{top}(h^V_2 + \beta), \tag{23}$$

where β an adjustable parameter.

2.2.7 Dynamic equations for controllers

(shortened) The column is stabilized by four base-layer PI-controllers. The controllers with their controlled variables (CV) and manipulated variables (MV) are listed in table 2.

2.3 Alternative strategies for solution of the algebraic equations

As mentioned, there are $(N_c + 1) \cdot N$ dynamic balance equations, where N_c is the number of components and N is the number of stages in the column. The associated dynamic state variables on stage i are M_i^{tot}, $\mathbf{M}_i$ (vector of length $N_c - 1$) and U_i^{tot}. In addition, there is a large number of algebraic equations which are generally not explicit in the dynamic state variables (M_i^{tot}, $\mathbf{M}_i$, U_i^{tot}), and therefore need to be solved. Several approaches for solving the algebraic equations are possible:

- **Approach 1.** (shortened) The algebraic equations are solved separately at each evaluation of the derivatives of the dynamic state variables (right hand side of dynamic balance equations). This is in general not numerically efficient.

- **Approach 2.** (shortened) The differential and algebraic equations (DAEs) are solved simultaneously using a DAE solver. Here, one generally tries to minimize the number of algebraic equations and associated algebraic state variables used in the DAE solver by finding the lowest number of algebraic equations that must be solved to make the rest of algebraic equation set explicit. In most cases, the algebraic vapor-liquid equilibrium (VLE) relations (19) are explicit in the variables pressure p, temperature T, liquid composition $\mathbf{x}$ (vector with $N_c - 1$ independent variables) and vapor composition $\mathbf{y}$ (vector with $N_c - 1$ independent variables). In total, this gives $2N_c$ algebraic state variables. Thus, on each stage the $2N_c$ algebraic equations given in (14), (15), and (19) (for p, T, $\mathbf{x}$, $\mathbf{y}$) need to be included in addition to the $N_c + 1$ differential equations (for M_i^{tot}, $\mathbf{M}_i$, U_i^{tot}).

- **Approach 3.** (shortened) Solve the algebraic equations off-line and represent the solution in terms of functions of suitable independent variables. Note that one can use the same functions for all stages in the column. There are many possibilities for the choice of the independent variables:

- **3.1 Use the dynamic state variables also for functions:**

  A straightforward choice are the N_c+1 dynamic state variables (M_i^{tot}, $\mathbf{M}_i$, U_i^{tot}). Then, the algebraic equations can be solved offline to generate the required functions in terms of these $N_c + 1$ variables. However, this set of variables may not be the best, since high-dimensional functions with independent variables which can assume values on a large domain are difficult to implement.

- **3.2 Introduce special algebraic variables for functions:**

  It is desirable to minimize the number of variables to reduce the dimension of the functions. From the Gibbs phase rule, it is actually sufficient to specify N_c (rather than $N_c + 1$) intensive variables for a system in vapor-liquid equilibrium. For the binary mixture in this study ($N_c = 2$), pressure p and temperature T are chosen as independent variables.

2.4 Final DAE equation set for full model

In this study, a binary mixture is considered and approach 3.2 with T and p (on each stage) as algebraic state variables is used. On each of the N stages, the DAE set includes three differential equations (see equations (1)-(12)), plus one algebraic equation for the sum of phases holdup of component 1 (14) and one algebraic equation for the sum of phases internal energy (15). The 5 associated state variables X_i on each stage are

$$X_i = \{M_i^{tot}, M_i, U_i^{tot}, p_i, T_i\}. \tag{24}$$

Note that for a binary mixture, M_i is the scalar holdup of component 1. In addition to these $3N$ dynamic and $2N$ algebraic equations, the full DAE model has one dynamic equation for each controller with integral action. The resulting full set of equations solved by the DAE solver can be written in the form

$$M\frac{d\mathbf{X}}{dt} = \mathbf{F}(\mathbf{X}, \mathbf{u}), \tag{25}$$

where $\mathbf{X}$ are the $5N + 4$ state variables used by the DAE solver, $\mathbf{u}$ is a input vector, and M is the diagonal mass matrix with a 1 on the diagonal for a differential equation and a 0 for an algebraic equation.

The remaining algebraic equations, including eqs. (18) and (17), and also the flash equations which are represented by tables, are explicit in $\mathbf{X}$ and are solved at each evaluation of the right hand side $\mathbf{F}(\mathbf{X}, \mathbf{u})$.

2.5 Jacobian structure

(shortened) The DAE set for the full model described by (25) in section 2.4 is highly structured as can be seen from figure 1 a, which shows the Jacobian structure ($d\mathbf{F}/d\mathbf{X}$) of the full model. The Jacobian is basically a banded matrix. However, the temperature controller in the bottom section, which has influence on the temperature of stage 76 in the bottom section, introduces elements into the Jacobian, which correspond to the proportional and integral action of the controller and which lie outside the narrow band. By this, the width of the Jacobian band is increased several times. The level and pressure controllers at the top of the column also increase the width of the Jacobian band, but to a much less extent, as the manipulated and controlled variables are positioned spatially close to each other. The system including all controllers except the temperature controller in the bottom section has a Jacobian non-zero entry band of width 19 (see figure 1 b). If the temperature controller is included in the system with the temperature measurement located at stage 76 (19 stages from the bottom), the width is increased to 94. In order to avoid a loss in computational performance, the special structure of the Jacobian has to be taken into account when decomposing the Jacobian for the solution of the linear equations arising during the integration of the model. This is described in section 2.6.4.

2.6 Implementation of full model

2.6.1 System size

In the present case, with $N = 94$ stages in the column, the full differential-algebraic model contains $94 \cdot 3$ dynamic and $94 \cdot 2$ algebraic variables for the stages, and 4 dynamic variables for the states of the PI-controllers, adding up to a total of 474 variables in the state vector $\mathbf{X}$.

2.6.2 Numerical solution

For simulation, the DAE solver DASPK 3.0 (Li and Petzold, 2000) is used. This solver is implemented in FORTRAN 77. The residual $\mathbf{F}$ and the analytic Jacobian $d\mathbf{F}/d\mathbf{X}$ of the model are programmed in C-code. This guarantees a simulation speed of the model which is close to the optimum.

2.6.3 Tabulation of thermodynamic properties

The thermodynamic VLE relations and property relations were programmed as two-dimensional look-up tables. From these 8 tables, the thermodynamic quantities x, y, h^V, h^L, v^V, v^L, ρ^V and ρ^L are obtained as functions of T and p by cubic spline interpolation (Press et al., 2007) of the table entries. Each table has 1 000 x 1 000 entries, where 273 K $< T <$ 350 K and 1 bar $< P <$ 8 bar.

2.6.4 LU-factorization of Jacobian

The Jacobian is evaluated analytically. Each time the Jacobian is recomputed, it is factorized into a lower and an upper triangular matrix. For this, a modified banded Gaussian LU-factorization is used. The LINPACK (LINPACK, 1978) routine DGBFA as used in DASPK to LU-factorize a banded matrix was modified to work with the narrow banded Jacobian matrix as described in section 2.5. An efficient special treatment of the off-band elements was introduced, where the rows containing the off-band elements are included in the elimination steps of the in-band rows above them. Correspondingly, the LINPACK routine DG-

BSL was modified to solve the linear equation system arising at each integration step using the previously generated LU-factorization.

3 Reduced model

3.1 Summary of reduction method

(shortened) The model reduction method used in this study is based on the stage aggregation method of Lévine and Rouchon (1991). Their original method was for a simple column model with only one mass balance per stage. The column is partitioned into a number of "compartments" of consecutive stages. In each compartment, an "aggregation stage" is selected, which is assigned the average concentration of the total compartment. Using a singular perturbation argument (Kokotovic at al., 1986), the holdups on the aggregation stages are increased to equal the total holdups of the respective compartments, whereas all remaining stages are rendered quasi-steady-state with zero holdup. The resulting model is therefore of the same size as the original model, but the majority of the dynamic stage equations are converted into algebraic equations. It assumes the same steady-state as the original full-order model.

It was shown in a previous study by Linhart and Skogestad (2009) that the method can actually be derived without the notion of compartments. This is due to an undocumented simplification step in the original derivation of the method. Although this simplification step deviates from the standard procedure for deriving singular perturbation models (Kokotovic et al., 1986), it greatly simplifies the derivation of the reduced model and its structure. For the application of the method to a given full column model, it is sufficient to select some stages as aggregation stages, and assign them large aggregated holdup factors, while all remaining stages are modeled as "steady-state stages" by setting their left-hand sides to zero. This way, the original method can easily be generalized to more complex models including mass and energy balances; see Linhart and Skogestad (2009) for details.

It was found previously by Linhart and Skogestad (2009) that only applying the reduction procedure as described above does not necessarily improve the computational performance of the reduced model compared to the original model, since the number of equations in the reduced model is the same as in the original model. Therefore, in a second step, the algebraic equations resulting from the reduction procedure are eliminated from the reduced model. For this, the block-wise structure of the reduced model is exploited.

The complete reduction procedure can therefore be described as a two-step procedure:

Step 1. Select a number of aggregation stages and slow down their dynamics by assigning them large aggregated holdup factors. Represent all remaining stages using steady-state equations. The resulting model is a DAE model of the same size as the original model. It has, however, reduced dynamics.

Step 2. Eliminate the algebraic equations of each block of steady-state stages by use of precomputed functions or tables. This yields a model with a reduced number of variables and equations, which can be simulated faster than the original model. This step can be divided into two sub-steps:

- **a)** Replace all variables of the steady-state stages that appear in the aggregation stage equations by functions obtained from the solutions of the steady-state equations in dependence of the variables of the aggregation stages;
- **b)** Eliminate some of the functions and independent variables to obtain a final reduced model that is as compact as possible.

Step 1 can be applied immediately to the full model by simple manipulation of the left-hand sides of the differential equations. This procedure will be described in section 3.2. Step 2a is necessary to produce a reduced-order model that increases the simulation speed. The basic procedure is described in section 3.3.1. The key idea is to replace the algebraic equations resulting from the reduction procedure by pre-computed functions. Due to the complexity of the model, these functions can become very complex themselves. To obtain efficient reduced models, in step 2b the number of functions needed and the number of independent variables these functions depend on is reduced to a minimum. This is described in section 3.3.2.

Note on notation: In order to stay consistent with the notation used in Linhart and Skogestad (2009), the variables in the reduced model of step 1, where the states are partially dynamic and partially algebraic, are marked by the bar notation $\bar{M}_i^{tot}$, $\bar{\mathbf{M}}_i$ and $\bar{U}_i^{tot}$. This is to distinguish them from the variables in the full model (1)-(12), where the states M_i^{tot}, $\mathbf{M}_i$ and U_i^{tot} are purely dynamic. In order to simplify notation, the variables of the reduced model of step 2 are not marked in a special way. The numbering is now different from the full model, since the final form of the reduced model consists only of aggregation stages. The functions that replace the algebraic equations of the steady-state stages are marked by $^{()}$. The final form of the reduced model is shown in table 3.

3.2 Reduction step 1: Introducing aggregation stages and steady-state stages

Figure 2 illustrates the reduction method: A number n stages on positions with the indices s_j, $j = 1...n$, in the column are selected dynamic aggregation stages. s_j is the stage index of aggregation stage j in the reduced model containing both aggregation and steady-state stages; see figure 3 a. For example, $s_3 = 10$ means that aggregation stage 3 corresponds to stage 10 in the original model. The dynamics of the aggregation stages are slowed down by multiplying the left-hand sides of the corresponding dynamic equations of each aggregation stage j by the aggregated holdup factor $H_j >> 1$:

$$H_j\dot{\bar{M}}_{s_j}^{tot} = \bar{L}_{s_j-1} + \bar{V}_{s_j+1} - \bar{L}_{s_j} - \bar{V}_{s_j}, \tag{26}$$

$$H_j\dot{\bar{\mathbf{M}}}_{s_j} = \bar{L}_{s_j-1}\bar{\mathbf{x}}_{s_j-1} + \bar{V}_{s_j+1}\bar{\mathbf{y}}_{s_j+1} - \bar{L}_{s_j}\bar{\mathbf{x}}_{s_j} - \bar{V}_{s_j}\bar{\mathbf{y}}_{s_j}, \tag{27}$$

$$H_j\dot{\bar{U}}_{s_j}^{tot} = \bar{L}_{s_j-1}\bar{h}_{s_j-1}^L + \bar{V}_{s_j+1}\bar{h}_{s_j+1}^V - \bar{L}_{s_j}\bar{h}_{s_j}^L - \bar{V}_{s_j}\bar{h}_{s_j}^V - \bar{Q}_{s_j}^{hl}. \tag{28}$$

The equations for the feed stage are treated correspondingly. The equations for the reflux drum and the reboiler are left unchanged, since their dynamics are slow due to their large holdups.

The remaining stages $i = 1...N, i \neq s_j$ $(j = 1...n)$, are converted into steady-state stages by setting the left hand sides of the respective dynamic equations to 0:

$$0 = \bar{L}_{i-1} + \bar{V}_{i+1} - \bar{L}_i - \bar{V}_i, \tag{29}$$

$$0 = \bar{L}_{i-1}\bar{\mathbf{x}}_{i-1} + \bar{V}_{i+1}\bar{\mathbf{y}}_{i+1} - \bar{L}_i\bar{\mathbf{x}}_i - \bar{V}_i\bar{\mathbf{y}}_i, \tag{30}$$

$$0 = \bar{L}_{i-1}\bar{h}^L_{i-1} + \bar{V}_{i+1}\bar{h}^V_{i+1} - \bar{L}_i\bar{h}^L_i - \bar{V}_i\bar{h}^V_i - \bar{Q}^{hl}_i. \tag{31}$$

3.3 Reduction step 2: Elimination of steady-state stages

(shortened) In the second step of the reduction procedure, the algebraic equations of the steady-state stages are eliminated from the model. Despite the large number of algebraic equations, this is possible because of the structure of the reduced model, where the steady-state stages are grouped in blocks between the dynamic aggregation stages.

3.3.1 Step 2a: Replacement of steady-state equations by functions

Figure 3 illustrates the principle. To avoid complicated notation, aggregation stages 2 and 3 are used for demonstration. Table 3 can be used as a reference for the general form of the equations. A block of steady-state stages is located between aggregation stages 2 and 3 (figure 3 a). It is referred to in the following as steady-state block 3. It constitutes a system of algebraic equations, consisting of a set of equations (29)-(31) for each of the steady-state stages with the indices $i = s_2 + 1$ to $i = s_3 - 1$. It can be solved in dependence on a certain set $\mathbf{z}_3$ of variables of aggregation stages 2 and 3.

In order to eliminate the equations of steady state block 3, the variables $\bar{\mathbf{y}}_{s_2+1}, \bar{h}^V_{s_2+1}$ and $\bar{V}_{s_2+1}$

in the dynamic equations of aggregation stage 2 are replaced by the functions

$$\bar{\mathbf{y}}_{s_2+1} = \mathbf{y}_3^{()}(\mathbf{z}_3), \tag{32}$$

$$\bar{h}^V_{s_2+1} = h_3^{V()}(\mathbf{z}_3), \tag{33}$$

$$\bar{V}_{s_2+1} = V_3^{()}(\mathbf{z}_3), \tag{34}$$

and the variables $\bar{\mathbf{x}}_{s_3-1}, \bar{h}^L_{s_3-1}, \bar{L}_{s_3-1}$ and $\bar{V}_{s_3}$ in the dynamic equations of aggregation stage 3 are replaced by the functions

$$\bar{\mathbf{x}}_{s_3-1} = \mathbf{x}_3^{()}(\mathbf{z}_3), \tag{35}$$

$$\bar{h}^L_{s_3-1} = h_3^{L()}(\mathbf{z}_3), \tag{36}$$

$$\bar{L}_{s_3-1} = L_3^{()}(\mathbf{z}_3), \tag{37}$$

$$\bar{V}_{s_3} = V_3^{b()}(\mathbf{z}_3). \tag{38}$$

The $\mathbf{y}_3^{()}$ notation signifies that the respective variable is a function of the variables of the neighboring aggregation stages 2 and 3; see figure 3 b.

The variables above correspond to the flow rates and intensive properties of the flows from the steady-state block into the aggregation stages. In addition, the vapor flow rate $\bar{V}_{s_3}$ from aggregation stage 3 depends on the variables of the bottom stage of the steady-state block. As a consequence, it is replaced by the function $V_3^{b()}$. The ^{b} indicates that this vapor flow is located at the bottom of the steady-state block 3, in contrast to the vapor flow $V_3^{()}$, which is located at the top.

It is assumed here that the liquid flows only depend on the variables of the departing stage, otherwise the liquid flow $\bar{L}_{s_2}$ departing from aggregation stage 2 would have to be replaced by a function as well.

Aggregation stage 3 is used for illustration of the the dynamic equations of the reduced model after the substitution:

$$H_3\dot{M}_3^{tot} = L_3^{()} - V_3^{b()} + V_4^{()} - L_3, \tag{39}$$

$$H_3\dot{\mathbf{M}}_3 = L_3^{()}\mathbf{x}_3^{()} - V_3^{b()}\mathbf{y}_3 + V_4^{()}\mathbf{y}_4^{()} - L_3\mathbf{x}_3, \tag{40}$$

$$H_3\dot{U}_3^{tot} \quad = \quad L_3^{()}h_3^{L()} - V_3^{b()}h_3^V + V_4^{()}h_4^{V()} - L_3h_3^L - Q_3^{hl}. \tag{41}$$

Here, the notation is simplified, and M_3, L_3 etc. signify that the reduced model after step 2 consists only of equations and variables corresponding to aggregation stages.

As independent variables for the functions (32)-(34) and (35)-(38), in principle the variable set

$$\mathbf{z}_3 = \{M_2^{tot}, \mathbf{M}_2, U_2^{tot}, M_3^{tot}, \mathbf{M}_3, U_3^{tot}\} \tag{42}$$

consisting of $2N_c + 2$ variables is valid. However, as mentioned before, it is crucial to keep the complexity of these functions to a minimum to obtain an efficient reduced model. A suitable minimal selection of functions and independent variables is therefore discussed in the next section.

3.3.2 Step 2b: Minimal selection of functions and independent variables

(shortened) The functions (32)-(34) and (35)-(38) are $2N_c + 3$ functions, while the variable set (42) contains $2N_c + 2$ variables. However, the functions are not completely independent of each other. Furthermore, not all state variables of both aggregation stages are needed as independent variables. In the following, it will therefore be shown that

1. The number of independent variables needed is $2N_c{+}1$ (instead of $2N_c{+}2$),
2. The number of functions needed is $N_c + 1$ (instead of $2N_c + 3$) .

Minimal number of independent variables:

The following variables that occur in the system of algebraic equations of steady-state block 3, consisting of the set of equations (29)-(31) for each of the steady-state stages, depend on the variables of the aggregation stages 2 and 3:

$$\bar{L}_{s_2}, \bar{\mathbf{x}}_{s_2}, \bar{h}_{s_2}^L \text{ and } \bar{V}_{s_2+1} \tag{43}$$

depend on variables of aggregation stage 2, and

$$\bar{V}_{s_3}, \bar{\mathbf{y}}_{s_3} \text{ and } \bar{h}^V_{s_3} \tag{44}$$

depend on variables of aggregation stage 3 (compare figure 3 a). The vapor flow $\bar{V}_{s_2+1}$ is a variable of steady-state stage $s_2 + 1$, but appears here because of its dependence on the variables of aggregation stage 2. Except for the liquid flow $\bar{L}_{s_2}$, all variables depend only on N_c intensive variables on the respective aggregation stage. $\bar{L}_{s_2}$ depends on an additional extensive variable due to its dependence on the liquid level on aggregation stage 2.

A suitable set of $2N_c + 1$ independent variables is therefore, for example,

$$\mathbf{z}_3 = \{\mathbf{x}_2, T_2, L_2, \mathbf{y}_3, T_3\}. \tag{45}$$

Here, the liquid flow from aggregation stage 2, L_2, is directly used as an independent variable for the functions of steady-state block 3.

For the case-study model with a binary mixture in the present work, the it is convenient to use set of independent variables

$$\mathbf{z}_{j+1} = \{T_j, p_j, L_j, T_{j+1}, p_{j+1}\}. \tag{46}$$

Minimal number of functions:

Using the fact that mass is conserved in the steady-state blocks and considering the total and $N_c - 1$ component mass balances around steady-state block 3 (compare figure 3 b),

$$\begin{aligned} 0 &= L_2 - V_3^{()} - L_3^{()} + V_3^{b()}, & (47) \\ 0 &= L_2\mathbf{x}_2 - V_3^{()}\mathbf{y}_3^{()} - L_3^{()}\mathbf{x}_3^{()} + V_3^{b()}\mathbf{y}_3, & (48) \end{aligned}$$

N_c additional equations are obtained to reduce the number of functions needed to be substituted in the dynamic equations of the aggregation stages (39)-(41). Energy is, however, only conserved if the heat loss occurring at each stage is neglected:

$$0 = L_2 h_2^L - V_3^{()} h_3^{V()} - L_3^{()} h_3^{L()} + V_3^{b()} h_3^V - Q_3^{hl()}. \tag{49}$$

Here, $Q_3^{hl()}$ is the accumulated heat loss of steady-state block 3.
Equations (47)-(49) can be rearranged to

$$\begin{aligned}
L_3^{()} - V_3^{b()} &= L_2 - V_3^{()}, && (50)\\
L_3^{()} \mathbf{x}_3^{()} - V_3^{b()} \mathbf{y}_3 &= L_2 \mathbf{x}_2 - V_3^{()} \mathbf{y}_3^{()}, && (51)\\
L_3^{()} h_3^{L()} - V_3^{b()} h_3^V &= L_2 h_2^L - V_3^{()} h_3^{V()} - Q_3^{hl()}, && (52)
\end{aligned}$$

and can then be used to eliminate the corresponding terms in the dynamic equations of aggregation stages. The equations for aggregation stage 3 (39)-(41) then read

$$\begin{aligned}
H_3 \dot{M}_3^{tot} &= L_2 - L_3 + V_4^{()} - V_3^{()}, && (53)\\
H_3 \dot{\mathbf{M}}_3 &= L_2 \mathbf{x}_2 - L_3 \mathbf{x}_3 + V_4^{()} \mathbf{y}_4^{()} - V_3^{()} \mathbf{y}_3^{()}, && (54)\\
H_3 \dot{U}_3^{tot} &= L_2 h_2^L - L_3 h_3^L + V_4^{()} h_4^{V()} - V_3^{()} h_3^{V()} - Q_3^{hl()} - Q_3^{hl}, && (55)
\end{aligned}$$

where only the vapor flow variables $\mathbf{y}_3^{()}$, $h_3^{V()}$, $V_3^{()}$, and the accumulated heat loss $Q_3^{hl()}$ remain as functions of steady-state block 3 (compare figure 3 c). Note that equation (55) also includes the heat loss term Q_3^{hl} for aggregation stage 3.

A further reduction of the number of functions can be achieved by using the fact that the vapor flow rate $V_3^{()}$ depends only on intensive variables of the topmost steady-state stage $s_2 + 1$ (compare figure 3 a and b). It is therefore sufficient to know N_c intensive variables on this stage, for example $\mathbf{y}_3^{()}(\mathbf{z}_3)$ and $p_3^{()}(\mathbf{z}_3)$, to calculate all other intensive variables of the vapor flow (i.e. $h_3^{V()}$), and the vapor flow rate $V_3^{()}$.
If the heat loss on each tray is not neglected, an additional function

$$Q_3^{hl()} = Q_3^{hl()}(\mathbf{z}_3) \tag{56}$$

has to be included in the set of functions.

In the case of a binary mixture, it is practical to use the set of functions

$$T_3^{()}(\mathbf{z}_3),\ p_3^{()}(\mathbf{z}_3),\ Q_3^{hl()}(\mathbf{z}_3), \tag{57}$$

because then $y_3^{()}$ (which is scalar in this case) and $h_3^{V()}$ can be conveniently calculated from the tabulated thermodynamics as described in section 2.6.3.

3.4 Jacobian structure

The Jacobian structure of the reduced model as given in table 3 is exactly the same as the structure of the full model, but the reduced model has fewer stages (see figure 1 b). Since the temperature controller in the bottom now only spans over two stages, the width of the Jacobian of the model including the controller does not differ much from that of the reduced model without controller.

3.5 Reduced model structure and parameters

(shortened) The reduced model in this study consists of nine dynamic aggregation stages and $94 - 9 = 85$ steady-state stages. There is a considerable freedom in choosing the locations and aggregated holdup factors of the aggregation stages. However, the choice of some stages as aggregation stages is straightforward (compare figure 2):

- Reboiler and reflux drum: This is a natural choice because their dynamics are usually slow due to their large capacities. Their aggregated holdup factors H_1 and H_n are set to 1.

- Feed stage: This makes inclusion of the feed into the reduced model straightforward.

- Stage below the reflux drum: This makes the application of the top pressure controller to the reduced model straightforward.

- Last stage above the reboiler: It was found that the dynamic behavior of the temperature control loop is improved considerably by introducing an additional aggregation stage between the measurement stage and the reboiler, increasing the dynamic order of the control loop by one.

The positions of all other aggregation stages (three in the present case) are degrees of freedom that can be used to adapt the reduced model to the dynamic requirements of the application it is intended for. The same applies to the aggregated holdup factors.

In this study, two sets of parameters for the reduced model with nine aggregation stages are compared:

1. An "equally-distributed" choice of parameters, where the free aggregation stages are distributed between the fixed aggregation stages at equal distances. The aggregated holdup factor of each aggregation stage corresponds to half of the number of steady-state stages between the aggregation stage and the adjacent aggregation stages on both sides plus one for the aggregation stage.

2. An "optimized" choice of parameters, where the free parameters are determined to give the best least-squares fit of the top concentration trajectories of the reduced model and a reference trajectory generated with the input signal described in section 4.1. The parameter optimization can be performed conveniently using the reduced model in DAE form that is obtained after reduction step 1 as described above.

Table 4 shows the position and aggregated holdup factors of the reduced models used in this study.

3.6 Functional approximation of steady-state stages by table interpolation

(shortened) For a numerical approximation of the functions (57), a five-dimensional look-up table is used. The function values are calculated numerically on a grid of a certain resolution spanning the input domain. Function values at arbitrary points on the input domain can then be retrieved by interpolating between the adjacent table entries.

The following issues are important when generating and using the table:

1. The simplest way to obtain continuous function values is multi-dimensional linear interpolation (Press et al., 2007) of the discrete table entries. For a five-dimensional interpolation, $2^5 = 32$ table look-up operations and proportionally many calculations are needed. This is computationally relatively expensive, compared to other calculations in the column model. Possible simplifications are discussed later in section 5.

2. The table needs a certain resolution to achieve a sufficient approximation accuracy using linear interpolation. It is therefore advisable to restrict the domain of the independent variables. This can be done by determining the extremal values of these variables during a suitable simulation, and by adding some safety margin.

3. There are many possibilities for choosing the set of independent variables. A good choice may yield a significant decrease in table size for a given accuracy. This is illustrated in figure 4. Depicted are trajectories of the temperatures T and pressures p of two neighboring dynamic aggregation stages. While the temperatures assume values in large parts of the domain, the pressures are tighter correlated and move only in a narrow band of the whole domain. This can be explained by the fast nature of the pressure dynamics, which is due to the immediate dependence of the vapor flow on the pressure difference between two stages. It is therefore advisable to choose the pressure p_j of one dynamic stage j, and the pressure difference $\Delta p = p_{j+1} - p_j$ as independent variables, instead of the two pressures p_j and p_{j+1}. This reduces the domain of the independent variables and thereby the size of the table several times.

4. The choice of independent variables does not only affect the size of their

domain, but also the shape of the functions in general. The functions might depend more linearly on some variables than on others. In that case, a lower resolution for the corresponding table dimension is required.

5. In order to make optimal use of the available memory, the table resolution along each dimension and thereby the total table size can be adapted to the accuracy requirements. This can be done in two steps:

   (a) The interpolation error for a given table resolution is estimated. For this, the function value at a test point is calculated numerically. Symmetrically around this point, 2^5 grid points with a distance in each dimension corresponding to the table resolution are calculated numerically, and the interpolated function value at the test point is determined. This can be repeated for a number of test points to scan the domain of independent variables systematically, because the degree of curvature of the functions might vary over the domain. Either the average or the maximum of the absolute differences between exact and interpolated function values can be taken as a measure for the interpolation error.

   (b) The effect of the interpolation error on the outputs of interest in steady-state is estimated. The two outputs of primary interest of the model are the top and bottom product concentrations of component 1. The sensitivity of these concentrations to the error in one function can be calculated by perturbing the corresponding function value and calculating the finite-difference quotient. It was found that the sensitivities do not change significantly when different steady-states (corresponding to different constant inputs) are used to calculate the difference quotient.

The interpolation error of a function multiplied by the corresponding sensitivity gives an estimate for the effect of the interpolation error on the outputs. Appropriate table dimensions can now be found by minimizing a certain norm of the vector of the interpolation error effects for a given total storage space. Table 5 shows the sensitivities of the top concentration to errors in functions (57) of the steady-state blocks in the reduced model. The resulting table dimensions are shown in table 6.

4 Reduced model performance

(shortened) In this section, the performance of the reduced model is compared with the performance of the original model. The performance of a model always depends on the application the model is intended for. The objective of the performance assessment in this study is to give general insight into the approximation quality and the numerical performance of the reduced model in comparison to the original model. For this, simulations with fast continuous changes in the different inputs are performed. After each input change, the inputs are kept constant for a long time, allowing the system to approach steady state again.

4.1 Test input trajectories

Figure 5 shows the input trajectories used for the performance assessment. The inputs F, z_F and h_F describe the feed into the column, and can be seen from a control perspective as disturbance variables. The inputs p^s, T^s and R are the pressure controller setpoint, the temperature controller setpoint, and the reflux rate, respectively. The can be used as manipulated variables for higher-level control of the column. The input changes are implemented as continuous cubic-spline functions with a transition time of 10 s. After each change, the inputs are kept constant for 15 000 s, allowing the system to approach steady state.

4.2 Accuracy of reduced model

(shortened) Figures 6-11 show sections of the responses of the top and bottom concentrations of the full and the reduced models to changes in the various inputs. In relative terms, the deviations of the bottom concentrations of the reduced models are larger than the deviations of the top concentrations from the original model. In absolute terms, however, the bottom concentration deviations are minimal compared to the top concentration deviations, due to the action of the temperature controller in the bottom section. The parameters of the optimized reduced model have been determined by fitting the top concentration trajectories only. This explains that the approximation of the bottom concentration is not more accurate for the optimized reduced model than for the equally-distributed model.

Generally, in terms of top concentration approximation accuracy, the optimized reduced model is superior to the equally-distributed reduced model. This is not the case for input changes in the feed concentration (figure 7), where both models are approximating the original dynamics very accurately, but the equally-distributed model is slightly more accurate. This is because the optimized reduced model has been optimized to approximate the original model over the whole simulation domain, which lowers the approximation quality at some points to gain a larger improvement at others.

4.3 Computational performance of reduced model

In order to compare the original and the reduced model, both were simulated at different simulation tolerances, and their accuracies were compared to the corresponding simulation times. The accuracy of the original model can be expressed as

$$\varepsilon^{full}(\theta) = f(\theta), \tag{58}$$

where ε^{full} is the simulation error of the full model compared to the exact solution, and θ is the simulation tolerance. The accuracy of the reduced model can be expressed as

$$\varepsilon^{reduced}(\theta) = g(\theta) + \varepsilon^{reduction}, \tag{59}$$

where $\varepsilon^{reduction}$ is the invariant reduction error introduced by the model reduction procedure. This reduction error limits the maximal achievable accuracy of the reduced model. The total error $\varepsilon^{reduced}$ is therefore the sum of the reduction error and the simulation error described by the term $g(\theta)$.

The simulation tolerance can usually be specified as relative and absolute tolerances with individual values for every state of the model, or with one value which is used for all states. For simplicity, the relative tolerance is set to the same value as the absolute tolerance, and the same value is used for all states in this study. Both models are compared at the tolerances

$$\theta^{abs} = \theta^{rel} = 10^{i/2},\ i = 2, ..., 8, \tag{60}$$

except in figure 13, where simulation results of the reduced model with tolerances up to $\theta = 10^{-6}$ are shown.

4.3.1 Simulation time versus tolerance

(shortened) Figure 12 shows a logarithmic plot of the simulation times for a simulation of 370 000 s (real time) using the input trajectory described in section 4.1 of the full and the reduced model in dependence of the simulation tolerance θ. Using the same simulation tolerance, the reduced model can be simulated about 5 times faster for crude tolerances, and about 8 times faster for tight tolerances ($\theta \leq 10^{-2}$). At the average tolerance of $\theta = 10^{-2.5}$, the simulation time is about 0.056 s for the reduced model, and 0.43 s for the full model.

4.3.2 Simulation time versus error

(shortened) To quantify the model accuracy, the average deviation of the top concentrations from the exact trajectory are used as accuracy measures in this study:

$$\varepsilon^{full}_{average} = \frac{1}{t_{end}} \int\limits_{0}^{t_{end}} \left| x_1^{exact}(t) - x_1^{full}(t) \right| dt, \tag{61}$$

$$\varepsilon_{average}^{red} = \frac{1}{t_{end}} \int_{0}^{t_{end}} \left| x_1^{exact}(t) - x_1^{red}(t) \right| dt. \tag{62}$$

In practice, the integral is replaced by average of sample points at intervals of 50 s. Since the bottom concentration is varying little compared to the top concentration due to the temperature controller action, it is not included in the accuracy measure.

Figure 13 shows the simulation time of the full and the reduced model versus the average error. It can be seen that the simulation times of both the full and reduced model increase with increasing simulation accuracy (decreasing simulation error). For the reduced model, the reduction error is visible for the simulation with tight tolerances, where increasing the simulation accuracy does not lead to an increase of the overall accuracy. The reduction error starts to dominate the overall accuracy from tolerances of around $\theta = 10^{-2.5}$ and on. At the maximal achievable accuracy, the overall error is around $4.7 \cdot 10^{-4}$. This is the average error in the top concentration of the reduced model. Below this accuracy, the simulation time of the reduced model is considerably lower than that of the full model, with a factor of approximately 6.5 at $\theta = 10^{-2.5}$.

4.3.3 Computational complexity of model and solver

Figures 14 and 15 show the call graphs of the simulations of the full and the reduced model, respectively. They visualize how the different numerical functions of the solver and the model code call each other, and show the percentage of the total execution time that is spent in each function.In the call graphs, each block represents a function, with the following information:

- name of the function (see appendix A for details);
- percentage of total execution time spent in this function including execution times of sub-functions;
- (in brackets) percentage of total execution time spent only in this function (excluding execution times of sub-functions);

- total number of function executions. To obtain better statistics, each simulation was run 100 times.

Each arrow represents calls of a sub-function, with the following information:

- percentage of total execution time spent in the called function when called from the calling function, including execution times of sub-functions;
- total number of function calls from the calling function.

Only functions and calls that account for more than 1% of the total execution time are shown in the graphs. The graphs in figures 14 and 15 Both graphs were generated from simulations with the simulation tolerance $\theta = 10^{-2.5}$. At this tolerance, the reduced model shows the best performance (see figure 13).

From figures 14 and 15 it can be seen that the largest part of the execution time in both full and reduced model simulation is spent in the functions **res_** ($\sim$ 16% and $\sim$ 24%, respectively), **jac_** ($\sim$ 49% and $\sim$ 40%), and **psol_** ($\sim$ 14% and $\sim$ 13%). For the full model, residual and Jacobian evaluation of the DAE ($\sim$ 16% and $\sim$ 19%) are less computationally intensive than the LU-decomposition **dgbfamod** and LU-solution **dgbslmod** functions ($\sim$ 28% and $\sim$ 14%). The execution time of the residual evaluations is dominated by the look-up table time for obtaining the thermodynamic quantities on every stage (**srktlu_**), whereas in the Jacobian calculations, the execution times of the functions for computing the hydraulic quantities and their derivatives are slightly more costly than the thermodynamic calculations.

No function uses much more of the execution time than the other functions. This means that no significant increase in simulation speed can be achieved by reducing the execution time of a single function by some means. The most expensive functions are the linear algebra functions. A doubling of execution speed here would lead to a 22% decrease in total simulation time.

For the reduced model, the percentaged execution time of the residual evaluations **res_** ($\sim$ 24%) is significantly higher than for the full model. This is due to the computationally expensive table look-ups of the steady-state block variables

SSBtableLU. They require with $\sim 11\%$ almost half of the execution time of the function. The situation is similar for the Jacobian evaluations, where the derivative calculation of the tabulated functions account for $\sim 6\%$ of $\sim 17\%$. With $\sim 22\%$, the LU-factorization is not the most expensive operation anymore.

5 Discussion

5.1 Model reduction method

The main theoretical aspects of the model reduction method used in this study compared to the original method of Lévine and Rouchon (1991) have been discussed previously by Linhart and Skogestad (2009). As shown in section 4, the reduced model is capable of reproducing the dynamic behavior with good accuracy, and almost perfectly reproduces the steady-states, except for some interpolation error. The computational complexity is several times lower than that of the original full model.

The simplified derivation using aggregation stages instead of compartments makes the method applicable in a straightforward fashion to all kinds of staged processes. Since in step 1 of the reduction procedure only simple manipulations of the left-hand sides of the dynamic equations of the original model are needed, it is easy to quickly derive a model with reduced dynamics to test the suitability of the reduction method for the given case. The same way, a suitable parametrization and dynamic analysis of the reduced model can be done.

Step 2 of the reduction procedure is conceptually straightforward, but requires more implementation effort. Due to the high dimensionality of the functions that are substituted into the dynamic equations, the method is restricted to a low number of state variables on each stage. This is the main bottleneck of the method. However, the functional approximation by tabulation used in this study is a relatively simple and straightforward approach, which works very well for the discussed example system. Possible improvements are discussed in the next section.

The physical argument behind the original method by Lévine and Rouchon (1991) was to neglect the fast t

5.2 Functional approximation of steady-state stage variables

(shortened) The functional approximation of the dependent variables of the steady-state stages as described in section 3.3 is difficult because of the large number of independent variables. In the example distillation column in this study, the number of independent variables is five. This is one less than the total number of dynamic states of the aggregation stages on both sides of each block of steady-state stages. This is due to the unsymmetrical flows in the column, where the vapor flow only depends on the intensive quantities on each stage.

In this study, the functions values were obtained by multi-linear interpolation of five-dimensional look-up tables. The interpolation is computationally expensive, because each interpolation dimension increases the computational complexity by factor two. Thus, $2^5 = 32$ operations are needed to obtain one interpolated value from a table. However, as shown in section 4.3.3, the table look-up and interpolation takes only about 17% of the total simulation time. This means that the reduced model is only insignificantly slowed down by the additional complexity resulting from the elimination of the algebraic equations. If the reduction method is applied to a ternary system, then two more independent variables corresponding to one additional state on each side of each steady-state block have to be included in the table and the interpolation. The interpolation complexity will then increase by factor four. However, since the computational complexity of the other parts of the model will also increase due to the additional component, the interpolation complexity will still not dominate the rest of the model. This means that for extensions to systems with three or four components, the reduced model will still achieve a considerable gain in computation speed compared to the original model.

To reduced the complexity of the tables, the following ideas can be considered:

- The function to be approximated can be partially linearized in the following way:

$$f(x_1, x_2, x_3) \approx f_1(x_1, x_2) + f_2(x_1, x_2)x_3. \tag{63}$$

  This can be done when, for example, the function depends on the concentration of a component that has a very low concentration compared to the other components. Then, the nonlinear function that has to be tabulated is of lower dimension.

- Cubic spline interpolation can be used instead of linear interpolation along dimensions which require a high resolution. For example, table 6 shows that the table dimension corresponding to the independent variable δP requires a high resolution. This is due to a more nonlinear dependence of the function values on this variable. Cubic spline interpolation is easy to implement, but requires four look-up operations per dimension. If one table dimension is interpolated with cubic splines, the computational complexity of the interpolation will therefore double. However, since the interpolation error is of higher order, the complexity of the tables can be reduced several times.

- The table resolution can be adapted locally to the curvature of the tabulated function. A simple way to do this is to use non-uniform table grids. A more sophisticated method is the use of sparse grids, where the table resolution is adapted locally (Barthelmann et al., 2000).

As an alternative to tables, functional approximation using polynomials or other suitable basis functions can be used. Their coefficients can be determined by, for example, least-square fits to sample data on a certain domain of the independent variables. However, also here the resulting expressions can be rather complex due to the high number of independent variables, and the approximation accuracy can be unsatisfactory due to the global nature of the approximation. If, for example, polynomials up to third order are used, the resulting expression will consist of 56 terms. Such an approximation has been tested in a reduced model, but yielded rather large steady-state deviations of about 2%. However, by carefully selecting suitable basis functions and sample points for the determination of the coefficients, the approximation quality can possibly be improved.

For all function approximation methods, it is advisable to chose the domain of the independent variables on which the function is approximated as small as possible. The domain has to be, however, large enough to cover the operating domain the reduced model is intended for.

5.3 Application of reduced model in real-time optimizing control

It was shown in section 4.3 that the reduced models can increase the simulation speed by a factor of about 7.5 when the same tolerance $\theta = 10^{-2.5}$ is used. This makes the models attractive for model predictive control and dynamic real-time optimization applications. However, the performance of the reduced models were assessed only in open-loop simulations with long intervals between changes in the inputs. In real-time optimizing control applications, input changes are made at much higher frequencies. Therefore, the approximation quality in the initial period after a change in the inputs is of greater importance.

The reduced model reproduces the outputs of the full model with good accuracy. Because the reduced models are "slow models", that means that they asymptotically approximate the dynamics of the full model, the deviation is largest in the initial period after a rapid change in one or more input signal. The suitability of a reduced model of this kind for MPC and other real-time optimization applications will depend largely on how well the time-scales of the application and the model are matched, that means if the reduced model is capable to follow the changes in control and disturbance inputs at the frequency and speed they occur in the closed loop application.

As shown in section 4, the approximation quality of the reduced models is very good for changes in the disturbance variables F, z_F and h_F, and in the reflux rate R. The approximation of the dynamics of fast changes in the controller setpoints is less accurate. This can be compensated by changing the controller setpoints less rapidly and possibly less frequent to allow for accurate predictions of the reduced model.

5.4 Comparison to other reduction methods for distillation models

A comparison of the reduction method to other reduction methods for distillation columns has been given in Linhart and Skogestad (2008). The conclusions basically also hold for the extension of the method to more complex distillation models as discussed in this study.

Collocation methods (Cho and Joseph, 1983, Dalaouti and Seferlis, 2006) are probably the most similar methods in terms approximation accuracy and gain in simulation speed. While they are not restricted to a low number of components as the method described in the present study, they possibly loose some approximation accuracy by approximating staged columns by continuous equations and applying collocation methods to the resulting partial differential equations.

Wave propagation methods (Hankins, 2007, Kienle, 2000, Marquardt, 1990) are so far restricted to distillation models with rather strict assumptions such as constant molar flows, since they make use of analytic solution of wave profile equations. The resulting models can therefore be expected to have limited approximation accuracy when used as reduced models for complex distillation models. However, they result in models of very low order, which promise very fast simulations.

Other methods are more suitable for nonlinear controller design than for fast simulations (Kumar and Daoutidis, 2003). An overview of further reduction and simplification methods for distillation column is given by Skogestad (1997).

6 Conclusions

An extension and simplification of the aggregated modeling method of Lévine and Rouchon (1991) to complex distillation models is presented. The method is applicable in a straightforward fashion by manipulating the left-hand sides of the differential equations. It was shown that if the resulting algebraic equations are eliminated from the reduced model, the reduced model yields gains in computational speed of a factor of around 7.5. The elimination of the algebraic equations is conceptually straightforward, but requires the approximation of functions of five independent variables. In this study, look-up tables combined with multi-linear interpolation was for the function approximation. The approximation quality of the reduced model was shown by simulations to be very accurate. In this study, a binary distillation model was investigated. The extension of the method to systems with a larger number of components is possible, but limited by the increasing complexity of the function approximation, which is the main bottleneck of the method. If this bottleneck is treated carefully, the resulting fast and accurate reduced models are attractive for real-time optimizing control applications.

Acknowledgments

The authors thank Stefan de Graaf and Pål Kittilsen of Cybernetica AS for fruitful discussions. Furthermore, the authors thank Uwe Weimann of Zuse Institut Berlin for providing help using the non-linear equations solver NLEQ (Novak and Weimann, 1990) that was used to calculate the look-up tables. This work has been supported by the European Union within the Marie-Curie Training Network PROMATCH under the grant number MRTN-CT-2004-512441.

Appendix A: Full and reduced model main functions

(shortened) The following are the main functions of the full and reduced model implementations:

- Model functions:
  - **rhs_**: computes the right-hand side (residual) of the DAE systems. It calls sub-functions for calculating the thermodynamic and hydraulic quantities in the column.
  - **jacobian_**: computes the analytic Jacobian of the DAE systems.
  - **stageThermodynamics**: computes the thermodynamic quantities on each stage.
  - **srktlu**: obtains the thermodynamic properties from tabulated SRK correlations using cubic spline interpolation.
  - **liquidFlow**: computes the liquid flow between two stages.
  - **vaporFlow**: computes the vapor flow between two stages.
  - **SSB_tableLU**: computes the tabulated variables of the steady-state blocks by multi-linear interpolation.
  - **d_...**: derivatives of the above functions.
- Solver functions:
  - **ddaspk_**: is the main function of the DASPK solver [18]. It is called iteratively by the main function to integrate the system starting from a certain start time for a certain interval.
  - **res_**: provides the residual of the DAE systems by directly calling the function **rhs_**.
  - **jac_**: provides the Jacobian of the DAE systems by calling the function **jacobian_**. In addition, it calls the function **dgbfamod** to decompose the Jacobian into lower and upper triangular matrices.

  - **psol_**: solves the nonlinear system of equations arising from the integration algorithm.
  - **ddwnrm**: computes the weighted root-mean-square norm of a vector.
  - **ddatrp**: performs interpolation to get an output solution.

- Linear algebra functions:
  - **dgbfamod_**: is a modified version of the LINPACK [20] LU-decomposition routine *dgbfa_* for banded matrices. The modification allows for the inclusion of off-band elements arising from decentralized control loops, as described in section 2.6.
  - **dgbslmod_**: is a modified version of the LINPACK LU-solution routine *dgbsl_* for banded matrices.
  - **daxpy_**: is a LINPACK routine for adding scaled matrix rows. It is the most intensively used subroutine of **dgbfamod** and **dgbslmod_**.

The remaining functions are mostly internal functions of the DASPK solver. They are, for example, used for step-size control of the integration time step, and are documented in the DASPK code (Li and Petzold, 2000).

References

[1] Allgöwer, F., Zheng, A., 2000. Nonlinear Model Predictive Control. Progress in Systems Theory 26, Birkhäuser, Basel.

[2] Antoulas, A.C., 2005. Approximation of Large-Scale Dynamical Systems. Cambridge University Press, Cambridge.

[3] Ascher, U.M., Petzold, L.R., 1998. Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations. SIAM, Philadelphia.

[4] Barthelmann, V., Novak, E., Ritter, K. 2000. High dimensional polynomial interpolation on sparse grids. Advanced in Computational Mathematics 12, 273-288.

[5] Benallou, A., Seborg, D.E., Mellichamp, D.A., 1986. Dynamic Compartmental Models for Separation Processes. AICHE Journal 32, 1067-1078.

[6] Bian, S., Khowinij, S., Henson, M.A., Belanger, P., Megan, L., 2005. Compartmental modeling of high purity air separation columns. Computers & Chemical Engineering 29, 2096-2109.

[7] Cho, Y.S., Joseph, B., 1983. Reduced-Order Steady-State and Dynamic Models for Separation Processes. Part I. Development of the Model Reduction Procedure. AIChE Journal 29, 261-269.

[8] Dalaouti, N., Seferlis, P., 2006. A unified modeling framework for the optimal design and dynamic simulation of staged reactive separation processes. Computers & Chemical Engineering 30, 1264-1277.

[9] Green, D.W., Perry, R.H., 2007. Perry's Chemical Engineers' Handbook, Eighth Edition. McGraw-Hill, New York.

[10] Hairer, E., Wanner, G., 2002. Solving Ordinary Differential Equations II - Stiff and Differential-Algebraic Problems. Springer, Berlin.

[11] Hankins, N.P., 2007. A non-linear wave model with variable molar flows for dynamic behavior and disturbance propagation in distillation columns. Chemical Engineering Research & Design 85, 65-73.

[12] Khowinij, S., Henson, M.A., Belanger, P., Megan, L., 2005. Dynamic compartmental modeling of nitrogen purification columns. Separation and Purification Technology 46, 95-109.

[13] Khowinij, S., Bian, S., Henson, M.A., Belanger, P., Megan, L., 2004. Reduced Order Modeling of High Purity Distillation Columns for Nonlinear

Model Predictive Control, Proceedings of the 2004 American Control Conference 5, 4237-4242.

[14] Kienle, A., 2000. Low-order dynamic models for ideal multicomponent distillation processes using nonlinear wave propagation theory. Chemical Engineering Science 55, 1817-1828.

[15] Kokotovic, P., Khalil, H.K., O'Reilly, J., 1986. Singular Perturbation Methods in Control: Analysis and Design. SIAM classics in applied mathematics 25, SIAM, London.

[16] Kumar, A., Daoutidis, P., 2003. Nonlinear model reduction and control for high-purity distillation columns. Industrial and Chemistry Research 42, 4495-4505.

[17] Lévine, J., Rouchon, P., 1991. Quality Control of Binary Distillation Columns via Nonlinear Aggregated Models. Automatica 27, 463-480.

[18] Li, S., Petzold, L.R., 2000. Software and Algorithms for Sensitivity Analysis of Large-Scale Differential Algebraic Systems. Journal of Computational and Applied Mathematics 125, 131-145.

[19] Linhart, A., Skogestad, S., 2009. Computational performance of aggregated distillation models. Computers & Chemical Engineering 33, 296-308.

[20] LINPACK, 1978. http://www.netlib.org/linpack/

[21] Marquardt, W., 2001. Nonlinear model reduction for optimization based control of transient chemical processes. Proceedings Chemical Process Control VI, 30-60.

[22] Marquardt, W., 1990. Traveling waves in chemical processes. International Chemical Engineering 30, 585-606.

[23] Nowak, U., Weimann, L., 1990. A Family of Newton Codes for Systems of Highly Nonlinear Equations - Algorithm, Implementation, Application. ZIB, Technical Report TR 90-10.

[24] Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 2007. Numerical Recipes: The Art of Scientific Computing, Third Edition. Cambridge University Press, Cambridge.

[25] Qin, S.J., Badgwell, T.A., 2003. A survey of industrial model predictive control technology. Control Engineering Practice 11, 733-764.

[26] Reid, R.C., Prausnitz, J.M., Poling, B.E., 1997. The properties of gases & liquids (4th edition). McGraw-Hill, New York.

[27] Schlegel, M., 2005. Adaptive discretization methods for the efficient solution of dynamic optimization problems. VDI-Verlag, Düsseldorf.

[28] Skogestad, S., 1997. Dynamics and Control of Distillation Columns: A Critical Survey. Modeling, Identification and Control 18, 177-217.

[29] van den Berg, J., 2005. Model reduction for dynamic real-time optimization for chemical processes. PhD Thesis, TU Delft.

List of figures

Figure 1: Jacobian structures of the full (plot a) and reduced (plot b) models. Shown are the dependency of the right-hand sides **F** on the states **X**. The solid lines mark the width of the non-zero Jacobian elements, when the elements corresponding to the temperature controller are excluded. The dashed lines mark the width when including these elements.

Figure 2: Schematic diagram of reduced column model.

Figure 3: Schematic illustration of a block of consecutive steady-state stages between aggregation stages 2 and 3. Part a) shows the structure after reduction step 1. Part b) shows the structure after elimination of the steady-state stages by substitution of functions (32)-(34) and (35)-(38). Part c) shows the structure after elimination of the flows on the bottom of the steady-state block by mass conservation.

Figure 4: Temperature and pressure correlations of aggregation stages 2 and 3.

Figure 5: Input trajectories used for model performance assessment.

Figure 6: Top and bottom concentration trajectories of the full and the reduced models. The feed flow rate F is changed from 155 to 140 (left part) and back (right part).

Figure 7: Top and bottom concentration trajectories of the full and the reduced models. The feed concentration z_F is changed from 0.34 to 0.19 (left part) and back (right part).

Figure 8: Top and bottom concentration trajectories of the full and the reduced models. The feed enthalpy z_h is changed from 0.2098 to 0.2598 (left part) and back (right part).

Figure 9: Top and bottom concentration trajectories of the full and the reduced models. The pressure setpoint p_1^s is changed from 4.8 to 4.75 (left part) and back (right part).

Figure 10: Top and bottom concentration trajectories of the full and the reduced models. The temperature setpoint T_{75}^s is changed from 322.35 to 321.35 (left part) and back (right part).

Figure 11: Top and bottom concentration trajectories of the full and the reduced models. The reflux rate R is changed from 370 to 340 (left part) and back (right part).

Figure 12: Simulation times of full and reduced models at different tolerances.

Figure 13: Simulation time versus average error. The numbers along the data points are the simulation tolerances used during the corresponding simulations.

Figure 14: Call graph of a simulation of the full model with simulation tolerance $\theta = 10^{-2.5}$.

Figure 15: Call graph of a simulation of the reduced model with simulation tolerance $\theta = 10^{-2.5}$.

List of tables

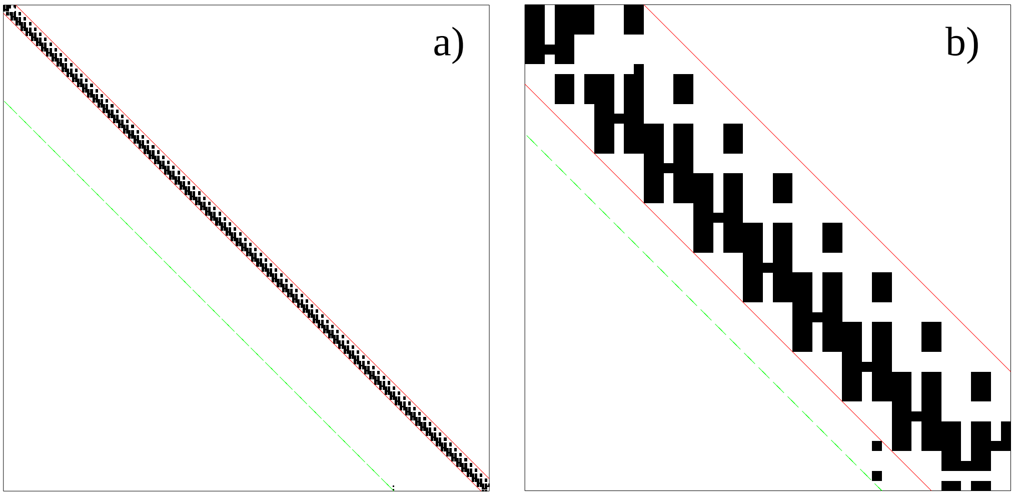
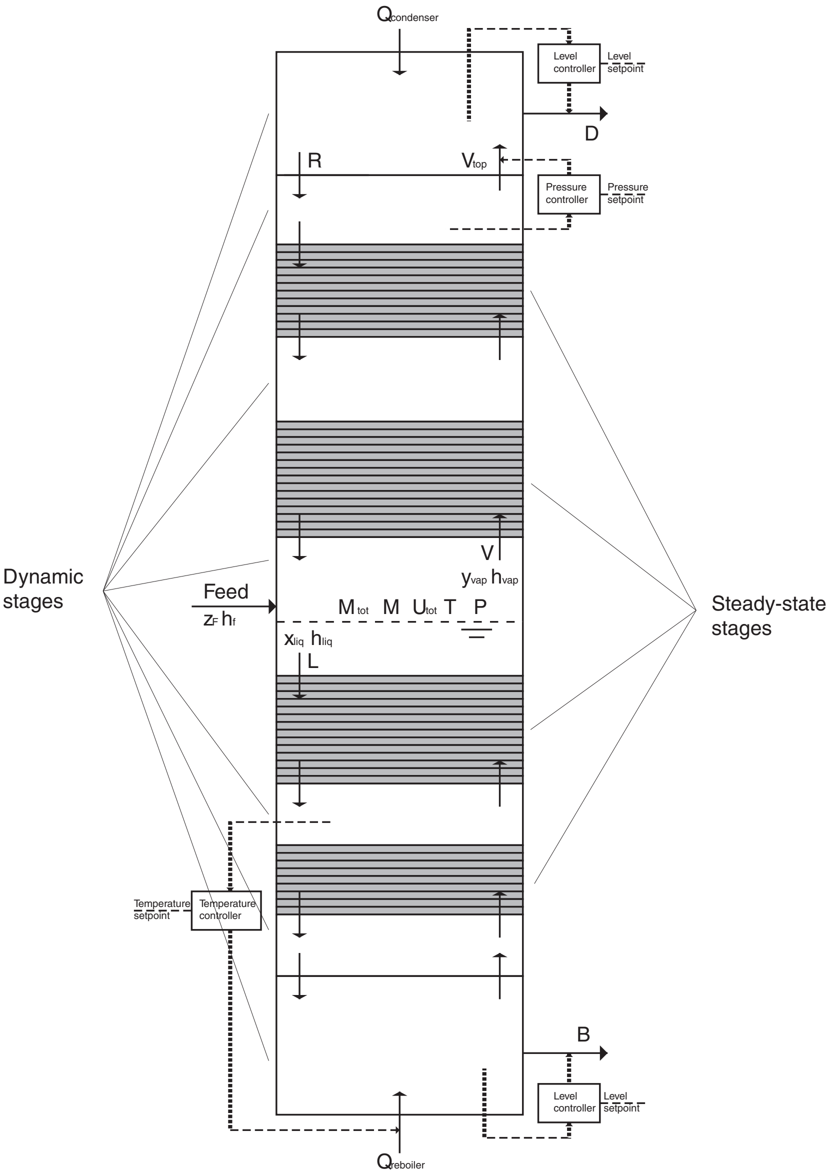


Figure 1:



$Q_{condenser}$
Level controller
Level setpoint
D
R
V_{top}
Pressure controller
Pressure setpoint
Dynamic stages
Steady-state stages
V
y_{vap} h_{vap}
Feed
z_F h_f
M_{tot} M U_{tot} T P
x_{liq} h_{liq}
L
Temperature setpoint
Temperature controller
B
Level controller
Level setpoint
$Q_{reboiler}$

Figure 2:

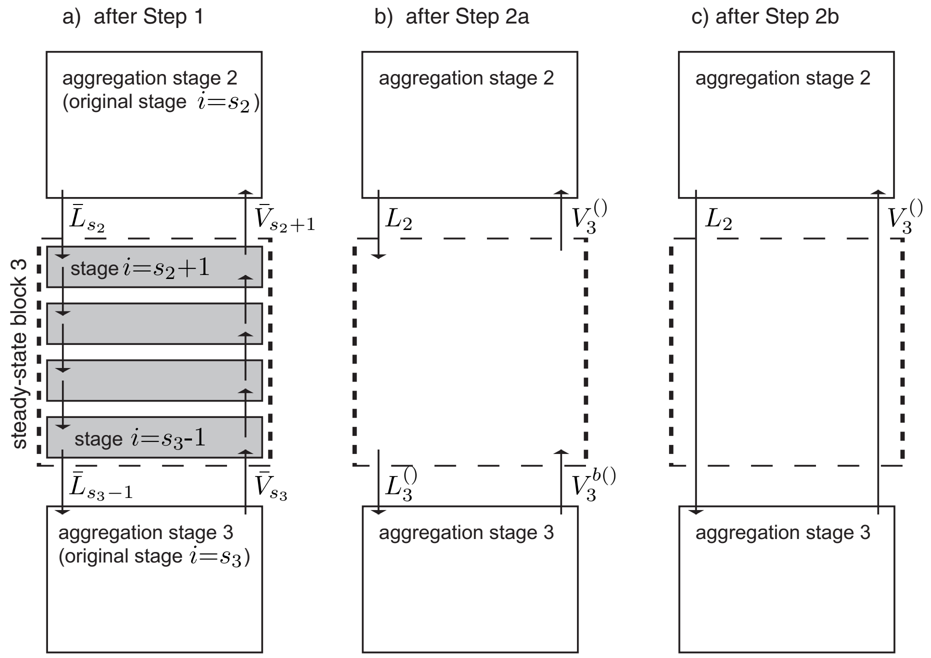


Figure 3:

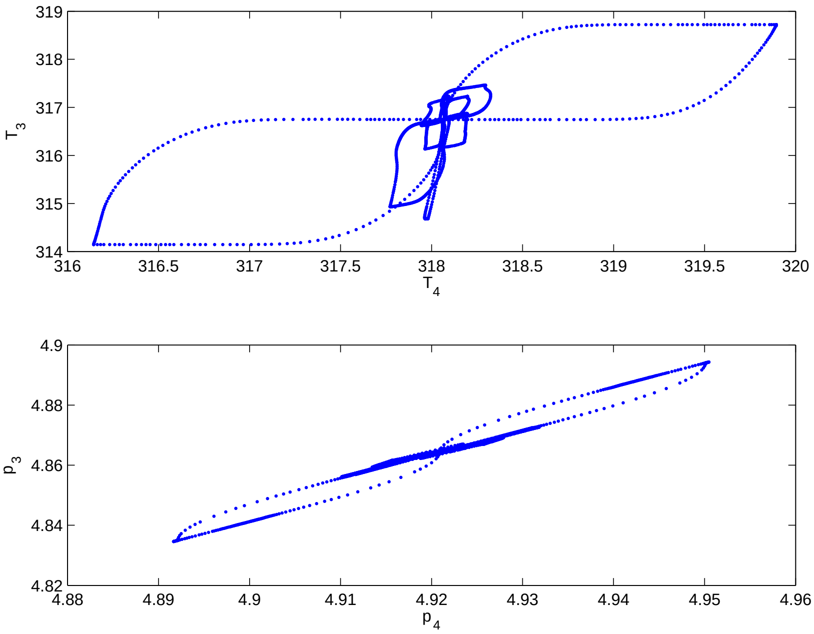


Figure 4:

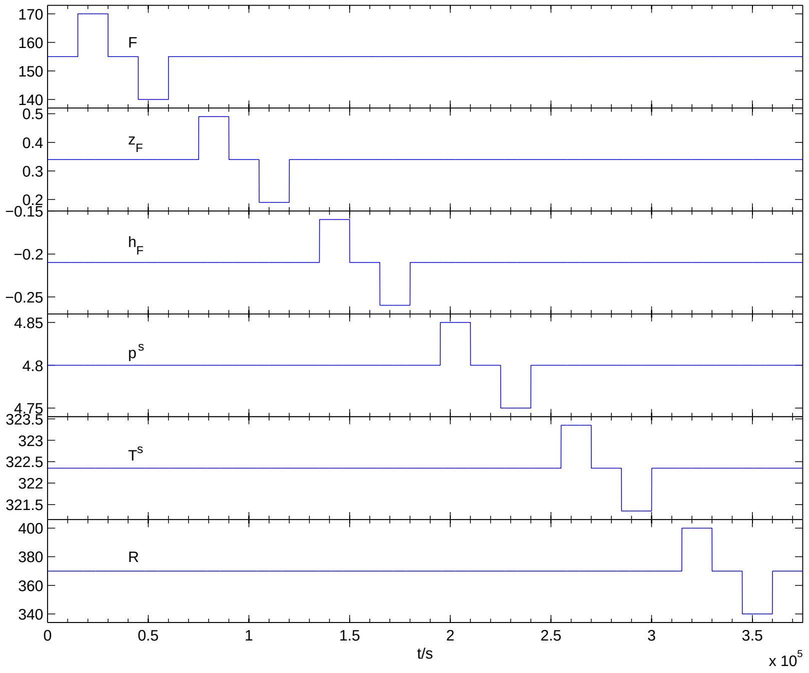


Figure 5:

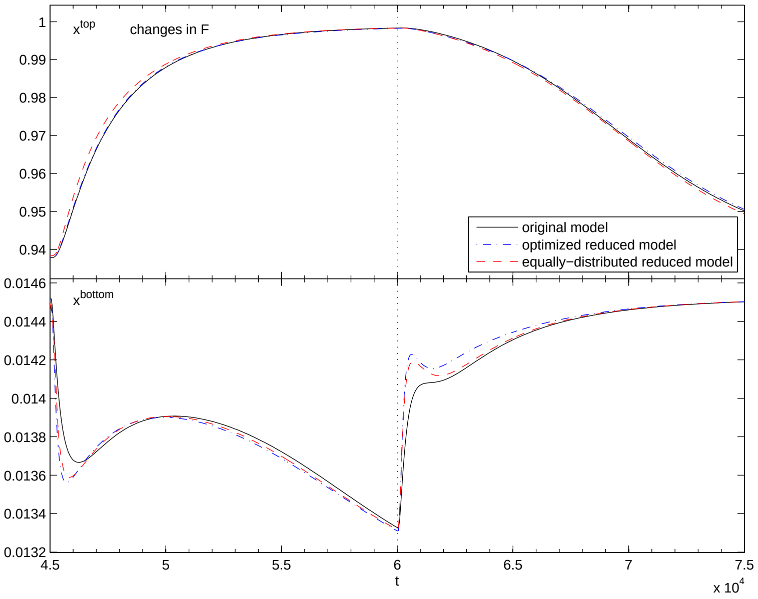


Figure 6:

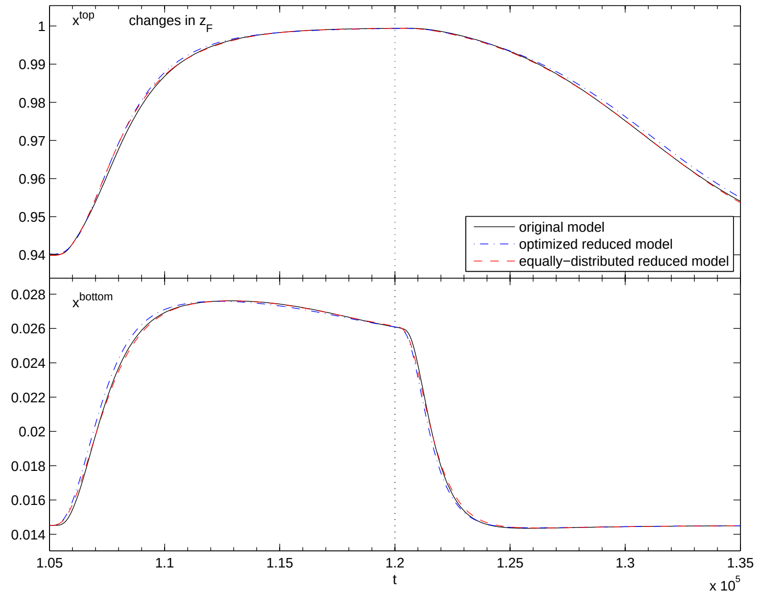


Figure 7:

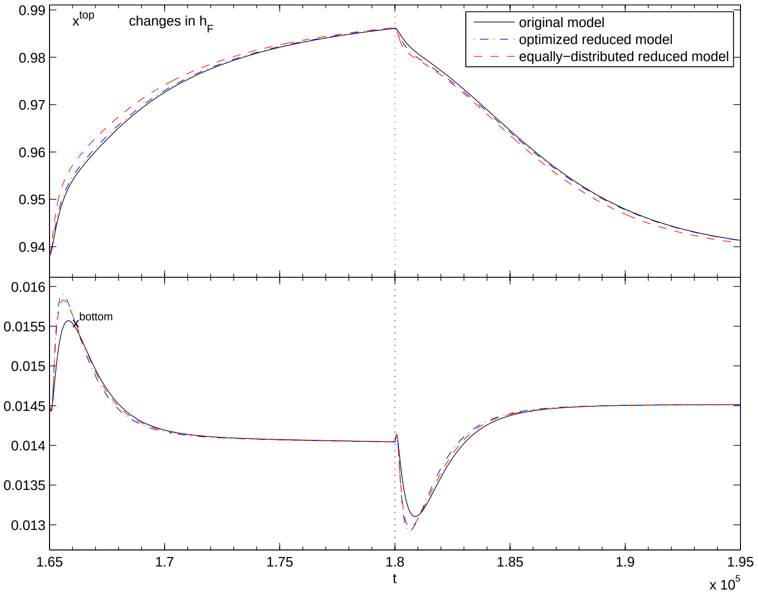


Figure 8:

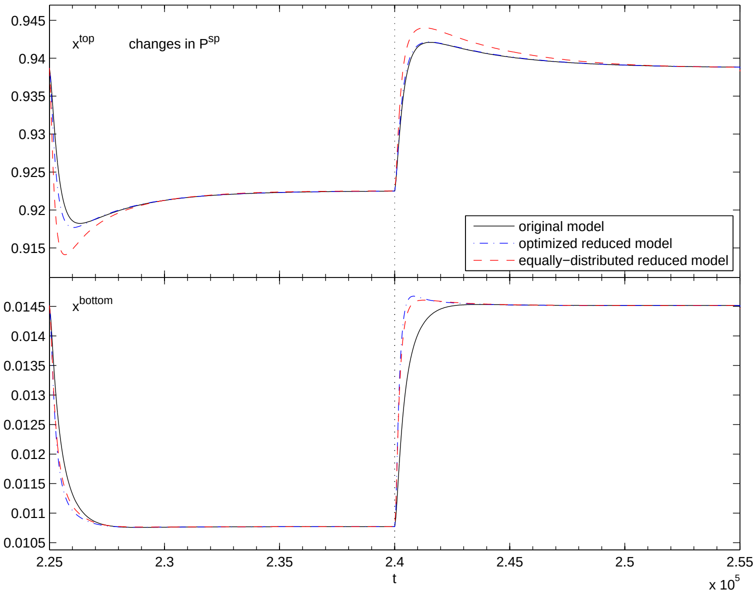


Figure 9:

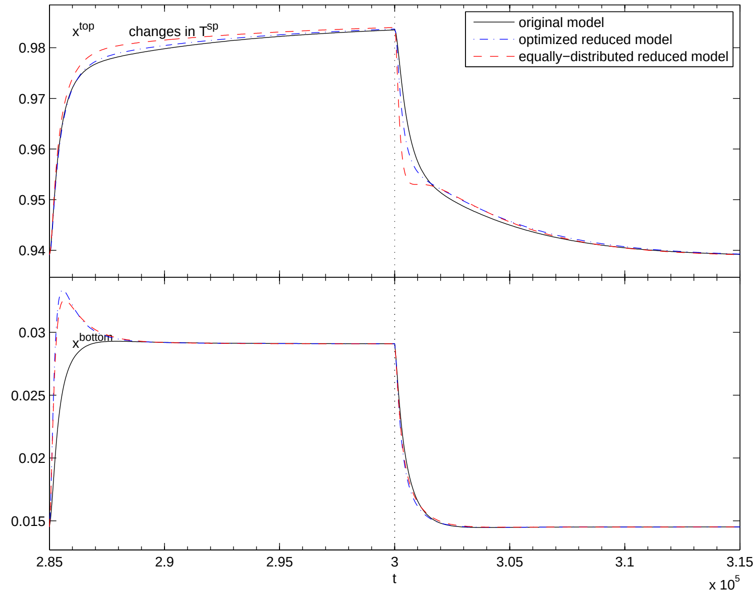


Figure 10:

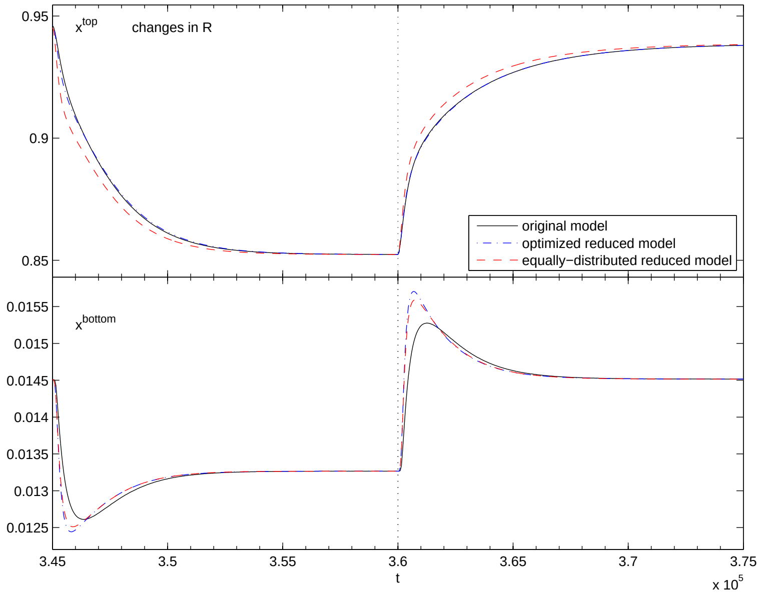


Figure 11:

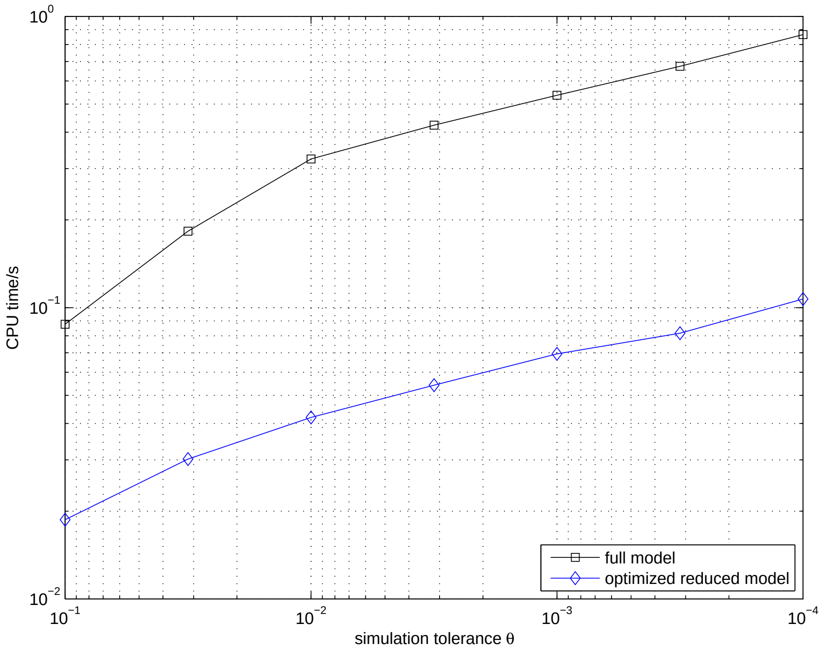


Figure 12:

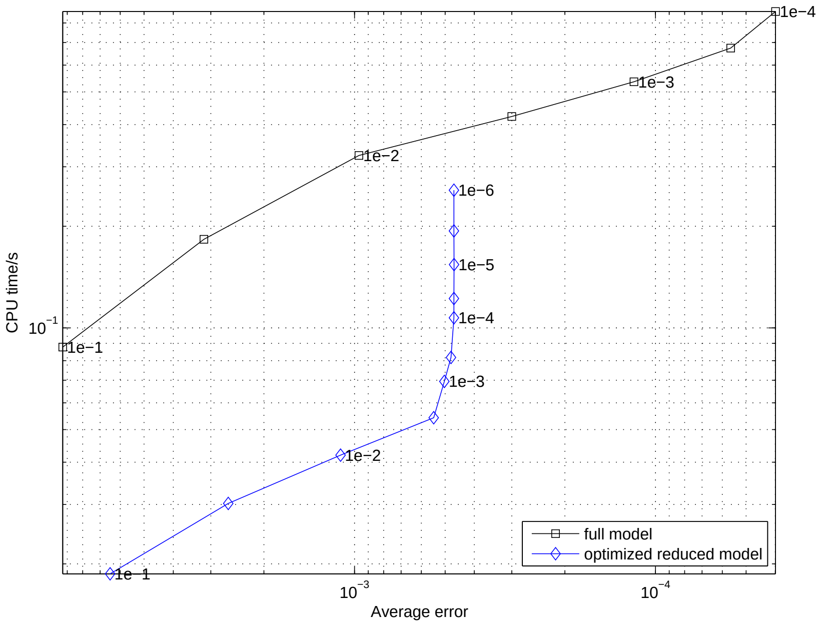

1e−4
1e−3
1e−2
1e−6
1e−5
1e−4
1e−1
1e−3
1e−2
1e−1
CPU time/s
10^{-1}
10^{-3}
10^{-4}
Average error
full model
optimized reduced model

Figure 13:

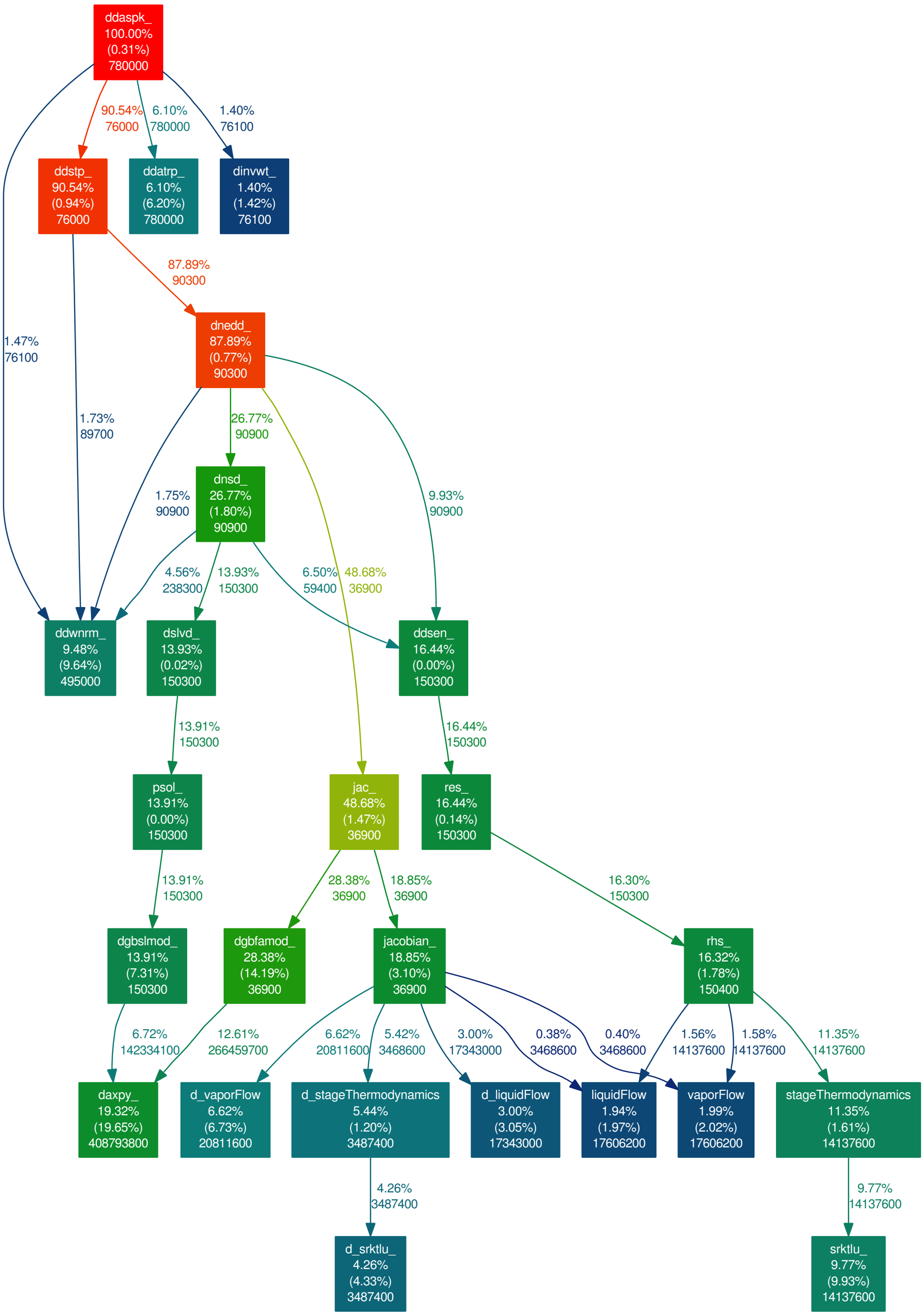
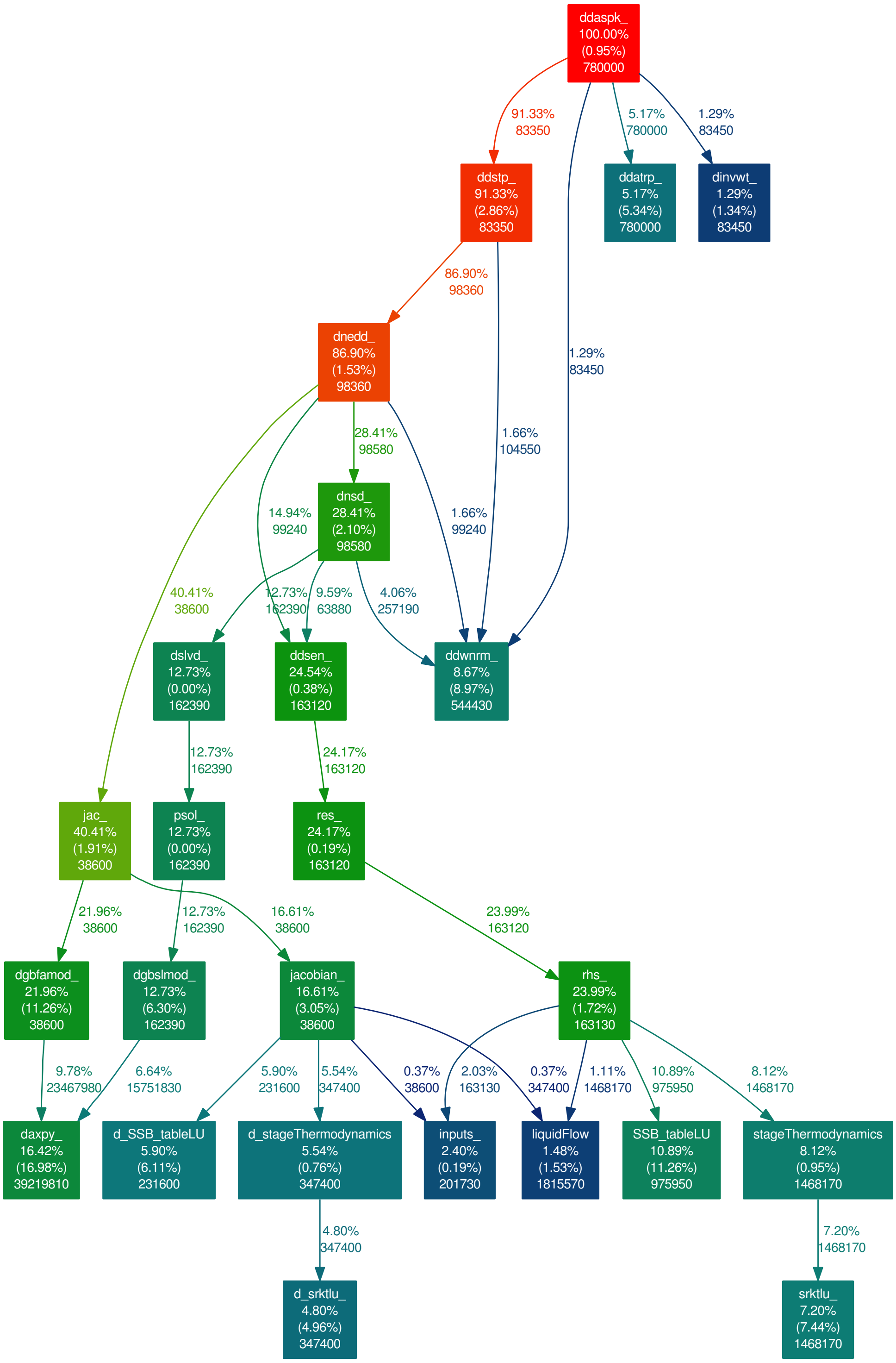

ddaspk_ 100.00% (0.31%) 780000
1.47% 76100
90.54% 76000
6.10% 780000
1.40% 76100
ddstp_ 90.54% (0.94%) 76000
ddatrp_ 6.10% (6.20%) 780000
dinvwt_ 1.40% (1.42%) 76100
87.89% 90300
1.73% 89700
dnedd_ 87.89% (0.77%) 90300
26.77% 90900
1.75% 90900
9.93% 90900
dnsd_ 26.77% (1.80%) 90900
4.56% 238300
13.93% 150300
6.50% 59400
48.68% 36900
ddwnrm_ 9.48% (9.64%) 495000
dslvd_ 13.93% (0.02%) 150300
ddsen_ 16.44% (0.00%) 150300
13.91% 150300
16.44% 150300
psol_ 13.91% (0.00%) 150300
jac_ 48.68% (1.47%) 36900
res_ 16.44% (0.14%) 150300
13.91% 150300
28.38% 36900
18.85% 36900
16.30% 150300
dgbslmod_ 13.91% (7.31%) 150300
dgbfamod_ 28.38% (14.19%) 36900
jacobian_ 18.85% (3.10%) 36900
rhs_ 16.32% (1.78%) 150400
6.72% 142334100
12.61% 266459700
6.62% 20811600
5.42% 3468600
3.00% 17343000
0.38% 3468600
0.40% 3468600
1.56% 14137600
1.58% 14137600
11.35% 14137600
daxpy_ 19.32% (19.65%) 408793800
d_vaporFlow 6.62% (6.73%) 20811600
d_stageThermodynamics 5.44% (1.20%) 3487400
d_liquidFlow 3.00% (3.05%) 17343000
liquidFlow 1.94% (1.97%) 17606200
vaporFlow 1.99% (2.02%) 17606200
stageThermodynamics 11.35% (1.61%) 14137600
4.26% 3487400
9.77% 14137600
d_srktlu_ 4.26% (4.33%) 3487400
srktlu_ 9.77% (9.93%) 14137600

Figure 14:



ddaspk_
100.00%
(0.95%)
780000
91.33%
83350
5.17%
780000
1.29%
83450
ddstp_
91.33%
(2.86%)
83350
ddatrp_
5.17%
(5.34%)
780000
dinvwt_
1.29%
(1.34%)
83450
86.90%
98360
1.29%
83450
dnedd_
86.90%
(1.53%)
98360
28.41%
98580
1.66%
104550
14.94%
99240
dnsd_
28.41%
(2.10%)
98580
1.66%
99240
40.41%
38600
12.73%
162390
9.59%
63880
4.06%
257190
dslvd_
12.73%
(0.00%)
162390
ddsen_
24.54%
(0.38%)
163120
ddwnrm_
8.67%
(8.97%)
544430
12.73%
162390
24.17%
163120
jac_
40.41%
(1.91%)
38600
psol_
12.73%
(0.00%)
162390
res_
24.17%
(0.19%)
163120
21.96%
38600
12.73%
162390
16.61%
38600
23.99%
163120
dgbfamod_
21.96%
(11.26%)
38600
dgbslmod_
12.73%
(6.30%)
162390
jacobian_
16.61%
(3.05%)
38600
rhs_
23.99%
(1.72%)
163130
9.78%
23467980
6.64%
15751830
5.90%
231600
5.54%
347400
0.37%
38600
2.03%
163130
0.37%
347400
1.11%
1468170
10.89%
975950
8.12%
1468170
daxpy_
16.42%
(16.98%)
39219810
d_SSB_tableLU
5.90%
(6.11%)
231600
d_stageThermodynamics
5.54%
(0.76%)
347400
inputs_
2.40%
(0.19%)
201730
liquidFlow
1.48%
(1.53%)
1815570
SSB_tableLU
10.89%
(11.26%)
975950
stageThermodynamics
8.12%
(0.95%)
1468170
4.80%
347400
7.20%
1468170
d_srktlu_
4.80%
(4.96%)
347400
srktlu_
7.20%
(7.44%)
1468170

Figure 15:

Table 1:

Variable	Description	SI-Unit	typical values		
			stage	reflux drum	reboiler
i	stage index				
i_F	index of feed stage				
M_i^{tot}	total mole number	mol	3500	49000	100000
M_i	total mole number of component 1	mol	3300	45500	1500
U_i^{tot}	total internal energy	J	-570	-10000	-17000
x_i	liquid concentration of component 1		0.34	0.94	0.015
y_i	vapor concentration of component 1		0.40	0.96	0.019
h_i^L	liquid enthalpy	J/mol	-0.18	-0.22	-0.18
h_i^V	vapor enthalpy	J/mol	0.012	0.016	0.018
L_i	liquid outflow	mol/s	500		
V_i	vapor outflow	mol/s	500		500
V_{top}	vapor flow from top stage into reflux drum	mol/s		425	
D	liquid distillate outflow of reflux drum	mol/s		55	
B	liquid bottom product outflow of reboiler	mol/s			100
R	reflux flow out of reflux drum	mol/s		370	
F	feed flow into feed stage	mol/s	155		
z_F	concentration of component 1 in feed		0.34		
h_F	feed enthalpy	J/mol	-0.22		
$Q_{condenser}$	heat flow into condenser	W		-95	
$Q_{reboiler}$	heat flow into reboiler	W			100
Q_i^{hl}	stage heat loss from to environment	W	0.024		

Table 2:

Controller	CV	MV
Level controller reflux drum	l_1	D
Pressure controller top stage	p_2	V_{top}
Temperature controller stage 76	T_{76}	$Q_{reboiler}$
Level controller reboiler	l_N	B

Table 3:

reflux drum (aggregation stage 1):

$$\begin{aligned}
\dot{M}_1^{tot} &= V_{top} - (R+D) \\
\dot{\mathbf{M}}_1 &= V_{top}\mathbf{y}_2 - (R+D)\mathbf{x}_1 \\
\dot{U}_1^{tot} &= V_{top}h_2^V - (R+D)h_1^L + Q_{condenser}
\end{aligned}$$

aggregation stage 2 (below reflux drum):

$$\begin{aligned}
H_2\dot{M}_2^{tot} &= R - L_2 + V_3^{()} - V_{top} \\
H_2\dot{\mathbf{M}}_2 &= R\mathbf{x}_1 - L_2\mathbf{x}_2 + V_3^{()}\mathbf{y}_3^{()} - V_{top}\mathbf{y}_2 \\
H_2\dot{U}_2^{tot} &= Rh_1^L - L_2h_2^L + V_3^{()}h_3^{V()} - V_{top}h_2^V - Q_2^{hl}
\end{aligned}$$

aggregation stage j:

$$\begin{aligned}
H_j\dot{M}_j^{tot} &= L_{j-1} - L_j + V_{j+1}^{()} - V_j^{()} \\
H_j\dot{\mathbf{M}}_j &= L_{j-1}\mathbf{x}_{j-1} - L_j\mathbf{x}_j + V_{j+1}^{()}\mathbf{y}_{j+1}^{()} - V_j^{()}\mathbf{y}_j^{()} \\
H_j\dot{U}_j^{tot} &= L_{j-1}h_{j-1}^L - L_jh_j^L + V_{j+1}^{()}h_{j+1}^{V()} - V_j^{()}h_j^{V()} - Q_j^{hl()} - Q_j^{hl}
\end{aligned}$$

feed stage j_F:

$$\begin{aligned}
H_{j_F}\dot{M}_{j_F}^{tot} &= L_{j_F-1} - L_{j_F} + V_{j_F+1}^{()} - V_{j_F}^{()} + F \\
H_{j_F}\dot{\mathbf{M}}_{j_F} &= L_{j_F-1}\mathbf{x}_{j_F-1} - L_{j_F}\mathbf{x}_{j_F} + V_{j_F+1}^{()}\mathbf{y}_{j_F+1}^{()} - V_{j_F}^{()}\mathbf{y}_{j_F}^{()} + F\mathbf{z}_F \\
H_{j_F}\dot{U}_{j_F}^{tot} &= L_{j_F-1}h_{j_F-1}^L - L_{j_F}h_{j_F}^L + V_{j_F+1}^{()}h_{j_F+1}^{V()} - V_{j_F}^{()}h_{j_F}^{V()} - Q_{j_F}^{hl()} - Q_{j_F}^{hl} + Fh_F
\end{aligned}$$

aggregation stage $n-1$ (before reboiler):

$$\begin{aligned}
H_{n-1}\dot{M}_{n-1}^{tot} &= L_{n-2} - L_{n-1} + V_n - V_{n-1}^{()} \\
H_{n-1}\dot{\mathbf{M}}_{n-1} &= L_{n-2}\mathbf{x}_{n-2} - L_{n-1}\mathbf{x}_{n-1} + V_n\mathbf{y}_n - V_{n-1}^{()}\mathbf{y}_{n-1}^{()} \\
H_{n-1}\dot{U}_{n-1}^{tot} &= L_{n-2}h_{n-2}^L - L_{n-1}h_{n-1}^L + V_nh_n^V - V_{n-1}^{()}h_{n-1}^{V()} - Q_{n-1}^{hl()} - Q_{n-1}^{hl}
\end{aligned}$$

reboiler (aggregation stage n):

$$\begin{aligned}
\dot{M}_n^{tot} &= L_{n-1} - B - V_n \\
\dot{\mathbf{M}}_n &= L_{n-1}\mathbf{x}_{n-1} - B\mathbf{x}_n - V_n\mathbf{y}_n \\
\dot{U}_n^{tot} &= L_{n-1}h_{n-1}^L - Bh_n^L - V_nh_n^V + Q_{reboiler}
\end{aligned}$$

Table 4:

	Equally distributed:		Optimized:	
aggregation stage	s_j	H_j	s_j	H_j
1 (Reflux drum)	1	1	1	1
2	2	8	2	8.42
3	17	14.5	13	10.93
4	31	14.5	26	16.22
5 (Feed)	46	15	46	19.51
6	61	15	65	12.34
7 (Temp. controlled)	76	16	76	14.46
8	93	9	93	6.99
9 (Reboiler)	94	1	94	1

Table 5:

Steady-state block j	$\partial x^{top}/\partial T_j^{()}$	$\partial x^{top}/\partial p_j^{()}$	$\partial x^{top}/\partial Q_j^{hl()}$
2	-0.0048	-0.66	-0.014
3	-0.014	5.17	-0.013
4	-0.64	60.8	-0.0077
5	-0.96	91.3	-0.00030
6	-0.20	40.6	0.00078
7	0.68	-259.7	7.8e-6

Table 6:

Steady-state block j	T_j	p_j	L_j	T_{j+1}	Δp_j
2	10	11	14	20	55
3	15	14	17	15	44
4	23	17	25	15	92
5	22	15	29	15	70
6	12	13	16	12	78
7	12	13	17	10	200