DETCHEM™
User Manual

Version 2.9.1

http://www.detchem.com

Olaf Deutschmann *, Steffen Tischer, Stefan Kleditzsch,
Vinod Janardhanan, Chrys Correa, Daniel Chatterjee,
Nikolay Mladenov, Hoang Duc Minh, Hüseyin Karadeniz,
Matthias Hettel, Vikram Menon, Aayan Banerjee, Hendrik Gossler,
Akash Shirsath, Eric Daymo

omegadot software & consulting GmbH
Mühlweg 40
67117 Limburgerhof
Germany

December 15, 2022

*Tel.: +49 7247 9542387, E-Mail: mail@detchem.com

Contents

Chapter 1

Getting Started

This chapter provides an introduction to DETCHEM, an explanation of its structure and capabilites. The detailed explanation of the models are presented in the subsequent chapters.

1.1 Introduction

Chemical reaction engineering and combustion processes are very often characterized by complex interactions between transport and chemical kinetics. The chemistry may include gas phase as well as surface reactions.

The simulation of fluid flows, which includes detailed schemes for surface and gas phase chemistry, has recently received considerable attention due to the availability of faster computers, the development of new numerical algorithms and the establishment of elementary reaction mechanisms. A key problem still remaining is the stiffness of the governing equations because of the different time scales introduced by chemical reactions including adsorption and desorption. Therefore simulations often use a simplified model, either for the flow field or for the chemistry. This simplification can be risky if there are strong interactions between flow and chemistry. In this case, extrapolations of the achieved results to conditions, which are different from those used for the model validation, are not reliable.

The DETCHEM software package consequently applies detailed models for the description of the chemical reactions and transport processes. It has been designed for a better understanding of the interactions between transport and chemistry and can assist in reactor and process development and optimization.

The basic models of DETCHEM are

- DETCHEM$^{\mathbf{GASPROBE}}$
- DETCHEM$^{\mathbf{SURFPROBE}}$
- DETCHEM$^{\mathbf{EQUIL}}$
- DETCHEM$^{\mathbf{BATCH}}$
- DETCHEM$^{\mathbf{CSTR}}$
- DETCHEM$^{\mathbf{STAG}}$
- DETCHEM$^{\mathbf{PLUG}}$
- DETCHEM$^{\mathbf{PBR}}$
- DETCHEM$^{\mathbf{PFR}}$
- DETCHEM$^{\mathbf{CHANNEL}}$
- DETCHEM$^{\mathbf{GRIDGEN3D}}$
- DETCHEM$^{\mathbf{MONOLITH}}$
- DETCHEM$^{\mathbf{RESERVOIR}}$
- DETCHEM$^{\mathbf{SOFC}}$

- DC4OpenFOAM
- DUO

1.2 Program structure

All the DETCHEM models are built up on the library modules ***lib_detchem***, ***lib_utils*** and ***lib_input***. ***lib_detchem*** comprises all library routines for calculating the chemical reaction source terms, while ***lib_utils*** comprises the auxiliary routines for the standard solvers and ***lib_input*** consists of all the routines for reading the newly designed input format of version 2.0.

1.3 Program Capabilities

DETCHEM has the following modeling capabilities

DETCHEM$^{\text{GASPROBE}}$ is a computational tool that calculates homogeneous chemical reaction rates and species thermodynamic and transport data at given conditions of temperature, pressure and composition.

DETCHEM$^{\text{SURFPROBE}}$ is a computational tool that calculates the surface coverages and fluxes at catalytic surfaces at given conditions of temperature, pressure and composition.

DETCHEM$^{\text{EQUIL}}$ is a computational tool to calculate equilibrium gas-phase compositions on the basis of thermodynamical data.

DETCHEM$^{\text{BATCH}}$ is a tool that simulates the temporal variations of temperature and concentrations of gas-phase species as a result of homogeneous gas-phase and/or heterogeneous surface chemical reactions.

DETCHEM$^{\text{CSTR}}$ is a tool that simulates the time dependent variations of temperature and concentrations of gas-phase species as a result of homogeneous gas-phase and/or heterogeneous surface chemical reactions in a continuous stirred tank reactor.

DETCHEM$^{\text{STAG}}$ is a tool that simulates the one dimensional profile of a reacting flow in a stagnation point flow configuration.

DETCHEM$^{\text{PLUG}}$ is a computational tool that simulates one dimensional reacting flows, with and without mass and heat transfer coefficients.

DETCHEM$^{\text{PBR}}$ is a computational tool that simulates one dimensional reacting flows in packed or fixed bed reactors by taking into account the mass and heat transfer effects.

DETCHEM$^{\text{PFR}}$ is a computational tool that simulates non-discursive one dimensional flow of chemically reacting ideal gas mixture in plug flow reactor considering heat and mass transfer effects.

DETCHEM$^{\text{CHANNEL}}$ is a computational tool that simulates parabolic two-dimensional reacting flows in straight channels and annulus using boundary-layer approach.

DETCHEM$^{\text{MONOLITH}}$ is a computational tool that simulates the two dimensional as well as the three-dimensional heat field in monoliths. The reactive flows inside the monolith channels are simulated by the tools DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$.

DETCHEM$^{\text{GRIDGEN3D}}$ is a pre-processor that can be used to set up the computational grid for three-dimensional monolith simulations with DETCHEM$^{\text{MONOLITH}}$.

DETCHEM$^{\text{RESERVOIR}}$ is a computational tool that simulates the transient concentrations within a storage catalyst. It serves as a transient one-dimensional wrapper model for single channel simulations with DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$.

DETCHEM^{SOFC} is a computational tool that simulates the two-dimensional transient fields within a Solid Oxide Fuel Cell (SOFC).

DC4OpenFOAM is a library which provides transport properties, gas-phase and surface chemistry to OpenFOAM to calculate an arbitrary number of coupled fluid and solid regions in three dimensions.

DUO is an interface to OpenFOAM to calculate monoliths coupled with 3D flow regions upstream. The flow, the gas-phase and surface chemistry in the tubes are handled in 1D or 2D optionally.

Chapter 2

Fundamentals of DETCHEM

The core of all DETCHEM applications is a library of routines that is called to calculate species data and chemical source terms. The species properties include thermodynamic and transport parameters. Whereas for the chemistry reaction schemes based on elementary and global reaction mechanism are used.

2.1 Species

2.1.1 General

The smallest model level is an entity called species. A species represents identical oder similar objects (molecules, adsobats, etc.) of a system in the thermodynamic limit. Microscopic properties of single particles are considered only indirectly in the derivation of transport coefficients.

The fundamental property of a species is its concentration c. Furthermore, if the species is formed by real particles, a molar mass M can also be defined.

The set of all Species is called a thermodynamic system $\mathbf{S}$. Depending on the kind of interactions among the species, it is convenient to consider various subsets of the system.

2.1.2 Gas-phase species

The gas phase $\mathbf{S}_\mathrm{g}$ is formed by the gas-phase species (atomic gases, molecules, radicals, etc.). The particles consist of one or more atoms. Their number, composition, electronic state etc. are characteristics for each species.

For the composition of the gas, we shall consider the following physical properties: total concentration

$$c_\mathrm{total} = \sum_{i \in \mathbf{S}_\mathrm{g}} c_i \tag{2.1}$$

and mean molar mass

$$\bar{M} = \frac{\sum_{i \in \mathbf{S}_\mathrm{g}} c_i M_i}{c_\mathrm{total}} \quad . \tag{2.2}$$

Instead of looking at the composition of a mixture with respect to the total volume, the mixture is quatifyed with respect to moles or mass. Then we can define the mole fraction

$$X_i = \frac{c_i}{c_{total}} \tag{2.3}$$

and the mass fraction

$$Y_i = \frac{M_i}{\bar{M}} X_i \quad . \tag{2.4}$$

Under the assumption of the ideal gas law, we can write the density ρ for given pressure p and temperature T as

$$\rho = \frac{p\bar{M}}{RT} \quad . \tag{2.5}$$

R is the gas constant.

Thermodynamic parameters

For most systems we assume an almost constant static pressure. Therefore, the thermodynamic potential of choice is the enthalpy H. The enthalpy density per mole h is a specific property of each species. When we also include the enthalpy of formation $\Delta_R \overline{H}_f^0$ in the definition, then the total enthalpy is conserved in a reacting system under isobaric conditions.

The heat capacity C gives the relation between heat δQ and a temperature change

$$C = \frac{\delta Q}{\mathrm{d}T} \quad . \tag{2.6}$$

The specific heat capacity at constant pressure c_p is then connected to the enthalpy density by:

$$c_p = \frac{\mathrm{d}h}{\mathrm{d}T} \quad . \tag{2.7}$$

Under the assumption of the ideal gas law, the heat capacity at constant volume is given by

$$c_V = c_p - R \quad . \tag{2.8}$$

Translations, rotations and vibrations contribute to c_V, which can be written as follows:

$$c_V = c_{V,tr} + c_{V,rot} + c_{V,vib} \quad , \tag{2.9}$$

$$c_{V,tr} = \frac{3}{2} R \quad , \tag{2.10}$$

$$c_{V,rot} = \frac{f}{2} R \quad . \tag{2.11}$$

f is the number of excited rotational degrees of freedom (0 for atoms, 2 for linear molecules, 3 for others).

Another variable of interest is the entropy S. According to the second law of thermodymanics it can be written as:

$$\mathrm{d}S = \frac{\delta Q}{T} \quad . \tag{2.12}$$

The thermodynamic properties of species i is described by a polynomial fit of fourth order to the specific heat at constant pressure:

$$C_{pt}^o = R \sum_{n=1}^{5} a_{ni} T^{(n-1)} = R(a_{1i} + a_{2i}T + a_{3i}T^2 + a_{4i}T^3 + a_{5i}T^4) \tag{2.13}$$

Once the specific heat is known, the other thermodynamic properties can be evaluated in terms of the specific heats. The standard state enthalpy is given as

$$H_i^o = \int_{T^o}^{T} C_{pt}^o(T')dT' \tag{2.14}$$

and the standard state entropy

$$S_i(T) = \int_{T^o}^{T} \frac{C_{pt}^o(T')}{T'} dT' \tag{2.15}$$

and with $T^o = 0\text{K}$, ${S_i}^o$ and ${H_i}^o$ can be written as polynomial fits

$$H_i^o = R(a_{1i}T + \frac{a_{2i}}{2}T^2 + \frac{a_{3i}}{3}T^3 + \frac{a_{4i}}{4}T^4 + \frac{a_{5i}}{5}T^5 + a_{6i}) \tag{2.16}$$

The constant of integration a_{6i} is the standard heat of formation at 0K. This constant can be evaluated from the knowledge of standard heat of formation at 298 K

$$S_i^o = R(a_{1i}lnT + a_{2i}T + \frac{a_{3i}}{2}T^2 + \frac{a_{4i}}{3}T^3 + \frac{a_{5i}}{4}T^4 + a_{7i}) \tag{2.17}$$

The constant of integration a_{7i} can be evaluated from the standard state entropy at 298 K.

Diffusion coefficients

The description of the interaction between the particles is based on the model of the Lennard-Jones-(6-12)-potential:

$$E_{\text{L.-J.}}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad , \tag{2.18}$$

where σ is the diameter of the molecule and ε the depth of the potential. The deviations from the model of solid spheres are expressed in terms of the reduced collision integral $\Omega^{(l,s)*}(T^*)$, which is only a function of the reduced temperature

$$T^* = \frac{k_\text{B}T}{\varepsilon} \tag{2.19}$$

(k_B- Boltzmann constant, T - temperature).

The diffusive transport due to a gradient in concentration can be described by the kinetic theory of diluted gases by *Chapman* and *Enskog*. We define the following reduced quantities:

- reduced molar mass

$$M_{ij} = \frac{M_i M_j}{M_i + M_j} \quad , \tag{2.20}$$

- reduced diameter

$$\sigma_{ij} = \frac{1}{2}(\sigma_i + \sigma_j)\xi^{-\frac{1}{6}} \tag{2.21}$$

- reduced potential depth

$$\varepsilon_{ij} = \sqrt{\varepsilon_i \varepsilon_j}\,\xi^2 \quad . \tag{2.22}$$

ξ is a correction factor to account for the interactions between polar (index p) and non-polar particles (index n)

$$\xi_{\text{np}} = 1 + \frac{1}{4}\left(\frac{\alpha_\text{n}}{\sigma_\text{n}^3}\right)\left(\frac{\mu_\text{p}^2}{4\pi\varepsilon_0\varepsilon_\text{p}\sigma_\text{p}^3}\right)\sqrt{\frac{\varepsilon_\text{p}}{\varepsilon_\text{n}}} \quad , \tag{2.23}$$

with α polarizability, μ dipole moment, and ε_0 permeability constant. In all other cases, ξ equal unity.

With these definitions, we can express the binary diffusion coefficient as a function of temperature T and pressure p:

$$D_{ij} = \frac{3}{16}\frac{\sqrt{2\pi N_\text{A} k_\text{B}^3 T^3 / M_{ij}}}{p\pi\sigma_{ij}^2\Omega_{ij}^{(1,1)*}(T_{ij}^*)} \quad , \tag{2.24}$$

(N_A - Avogadro constant).

For simplicity, most calculations use and averaged diffusion coefficient, which can be approximated by:

$$D_{i,\text{M}} = \frac{1 - Y_i}{\sum_{j\in\mathbf{S}_\text{g}\setminus\{i\}}\frac{X_i}{D_{ij}}} \quad . \tag{2.25}$$

Viscosity

The kinetic theory yields a result for the viscosity of a pure species:

$$\eta_i = \frac{5}{16}\frac{\sqrt{\pi M_i k_B T/N_A}}{\pi\sigma_i^2\Omega^{(2,2)*}(T_i^*)} \quad . \tag{2.26}$$

Here, an averaged coefficient of viscosity shall be used, which can be approximated by

$$\eta = \frac{1}{2}\left[\sum_{i\in\mathbf{S}_g} X_i\eta_i + \left(\sum_{i\in\mathbf{S}_g}\frac{X_i}{\eta_i}\right)^{-1}\right] \quad . \tag{2.27}$$

Heat conductivity

Heat conduction and viscosity in gases are caused by transfer of energy and momentum, respectively. Therefore, they are related to each other. However, translational, rotational, and vibrational energy give different contributions :

$$\lambda_i = \frac{\eta_i}{M_i}\left(f_{tr}c_{v,tr} + f_{rot}c_{v,rot} + f_{vib}c_{v,vib}\right) \quad . \tag{2.28}$$

Using the following relations

$$f_{tr} = \frac{5}{2}\left(1 - \frac{2}{\pi}\frac{c_{v,rot}}{c_{v,tr}}\frac{A}{B}\right) \quad , \tag{2.29}$$

$$f_{vib} = \frac{\rho D_{ii}}{\eta_i} \quad , \tag{2.30}$$

$$f_{rot} = f_{vib}\left(1 + \frac{2}{\pi}\frac{A}{B}\right) \quad , \tag{2.31}$$

with

$$A = \frac{5}{2} - f_{vib} \tag{2.32}$$

and

$$B = Z_{rot} + \frac{2}{\pi}\left(\frac{5}{3}\frac{c_{v,rot}}{R} + f_{vib}\right) \quad . \tag{2.33}$$

Z_{rot} is also a characteristic parameter. Once know for a given temperature (298 K), it can be calculated by the proportionality

$$\frac{1}{Z_{rot}(T^*)} \propto 1 + \frac{\pi^{\frac{3}{2}}}{2}T^{*-\frac{1}{2}} + \left(\frac{\pi^2}{4} + 2\right)T^{*-1} + \pi^{\frac{3}{2}}T^{*-\frac{3}{2}} \quad . \tag{2.34}$$

Averaging is handled in analogy to the viscosity by

$$\lambda = \frac{1}{2}\left[\sum_{i\in\mathbf{S}_g} X_i\lambda_i + \left(\sum_{i\in\mathbf{S}_g}\frac{X_i}{\lambda_i}\right)^{-1}\right] \quad . \tag{2.35}$$

2.1.3 Third-body species

In most recombination and decomposition reactions, a third particle is involved in order to guarantee conservation of energy and momentum. Any species of the mixture can take this part, which therefore is denoted by the Symbol M:

$$\mathrm{A} + \mathrm{B} + \mathrm{M} \rightleftharpoons \mathrm{AB} + \mathrm{M} \quad . \tag{2.36}$$

The efficiency of collisions with diffent species is different. In order to summarize all reactions of the same kind, we can define an effective concentration of species M by:

$$c_{\mathrm{M}} \quad = \quad \sum_{i \in \mathbf{S}_{\mathrm{g}}} \alpha_{\mathrm{M},i} c_i \quad , \tag{2.37}$$

where $\alpha_{\mathrm{M},i}$ are the collision efficiencies of the gas-phase species i with respect to a reference species (e.g. Argon).

The set of all third-body species shall be denoted by $\mathbf{S}_{\mathrm{M}}$.

2.1.4 Surface species

In analogy to the gas-phase, the properties of the surface are described in terms of a set of surface species $\mathbf{S}_{\mathrm{s}}$. Every catalytic material and in fact every surface structure has different catalytic properties. Therefore, it might be convenient to divide the set $\mathbf{S}_{\mathrm{s}}$ into subsets $\mathbf{S}_{\mathrm{s}}^j$ ($j = 1, 2, \ldots$); e. g. $\mathbf{S}_{\mathrm{s}} = \mathbf{S}_{\mathrm{Pt}} \cup \mathbf{S}_{\mathrm{Rh}}$.

On each surface type, there is only a limited number of adsorption sites available. An adsorption can be viewed as a reaction of a gas-phase species with an empty site. Thus, empty surface-sites shall be considered as a species.

The total number of surface sites is a conserved quantity. However, some species may occupy more than one site. For convenience we can define the surface coverage fraction

$$\theta_i \quad = \quad \frac{c_i \, \sigma_i}{\Gamma_{\mathrm{s}}^j} \quad . \tag{2.38}$$

Here, σ_i is the number of sites occupied by one particle of the species i and Γ_{s}^j is the surface site density of the specific surface type. For each subset $\mathbf{S}_{\mathrm{s}}^j$

$$\sum_{i \in \mathbf{S}_{\mathrm{s}}^j} \theta_i \quad = \quad 1 \tag{2.39}$$

must be fulfilled.

2.2 Chemical Reactions

2.2.1 General

General reaction equation

In general, a chemical reaction can be written as

$$\sum_{i \in \mathbf{S}} \nu_i' \mathrm{A}_i \quad \rightarrow \quad \sum_{i \in \mathbf{S}} \nu_i'' \mathrm{A}_i \quad , \tag{2.40}$$

where A_i is a symbol of the i-th species. ν_i' and ν_i'' are the stoichiometric coefficients of the reactants and products, respectively. There always exist integer coefficients. Their difference shall be denoted by

$$\nu_i \quad = \quad \nu_i' - \nu_i'' \quad . \tag{2.41}$$

For the set of all reactions we use the symbol $\mathbf{R}$.

Chemical equilibrium

For each reaction, there is always a reverse reaction, which leads to an equilibrium between the reactants and products:

$$\sum_{i \in \mathbf{S}} \nu_i' \mathrm{A}_i \quad \rightleftharpoons \quad \sum_{i \in \mathbf{S}} \nu_i'' \mathrm{A}_i \quad . \tag{2.42}$$

In case of equilibrium, the Gibbs Free Energy

$$G \quad = \quad H - TS \tag{2.43}$$

is a minimum for a given pressure and temperature. The specific Gibbs Energy can be defined using the specific enthalpy and entropy for a given pressure p^0 as

$$g_i = h_i - T s_i \quad . \tag{2.44}$$

The change of Gibbs Free Energy in a reaction that converts ν_i mol of each species is

$$\Delta_\mathrm{R}\overline{G}^0 = \sum_{i\in\mathbf{S}} \nu_i g_i \,\mathrm{mol} \quad , \tag{2.45}$$

This quantity is related to the equilibrium constant

$$K_\mathrm{p} = \prod_{i\in\mathbf{S}} \left(\frac{p_i}{p^0}\right)^{\nu_i} \tag{2.46}$$

by

$$K_\mathrm{p} = \exp\left(-\frac{\Delta_\mathrm{R}\overline{G}^0}{RT}\right) \quad . \tag{2.47}$$

In analogy, a equilibrium constant with respect to concentrations can be defined as

$$K_\mathrm{c} = \prod_{i\in\mathbf{S}} c_i^{\nu_i} \quad . \tag{2.48}$$

Using the reference concentration c^0 at pressure p^0 we can write

$$K_\mathrm{c} = \exp\left(-\frac{\Delta_\mathrm{R}\overline{G}^0}{RT}\right) {c^0}^{\Delta_\mathrm{R}\nu} \tag{2.49}$$

with

$$\Delta_\mathrm{R}\nu = \sum_{i\in\mathbf{S}} \nu_i \quad . \tag{2.50}$$

For a gas mixture the ideal gas law yields

$$c^0 = \frac{p^0}{RT} \quad . \tag{2.51}$$

Reaction rate

The reaction rate in general depends on the concentrations of the reactants (probability of a collision) and on temperature (collision energy). Therefore, the dynamics of reaction can be written as:

$$\frac{\mathrm{d}c_i}{\mathrm{d}t} = \nu_i\, k_\mathrm{f} \prod_{j\in\mathbf{S}} c_j^{\widetilde{\nu_j}\prime} \quad , \tag{2.52}$$

where k_f is a temperature-dependent rate coefficient (of the forward reaction) and $\widetilde{\nu_j}\prime$ are the reaction orders of each species. In analogy, we can write for the backward reaction:

$$\frac{\mathrm{d}c_i}{\mathrm{d}t} = -\nu_i\, k_\mathrm{r} \prod_{j\in\mathbf{S}} c_j^{\widetilde{\nu_j}\prime\prime} \quad . \tag{2.53}$$

The probability of a reaction due to a collision depends on the kinetic energy of the colliding particles. Therefore, *Arrhenius* chose the following ansatz for the rate coefficient:

$$k = A \cdot \exp\left(-\frac{E_\mathrm{a}}{RT}\right) \quad . \tag{2.54}$$

The constant A is called pre exponential factor. However, sometimes there is an additional temperature dependence in this factor. Thus, we use the following expression:

$$k = A\, T^\beta \cdot \exp\left(-\frac{E_\mathrm{a}}{RT}\right) \tag{2.55}$$

with an additional fit parameter β.

Elementary reactions

An elementary reaction is a reaction that can be observed on a molecular scale between single particles. Therefore, every complex chemical mechanism can (at least in theory) be broken down into elementary reactions. The advantage of such a scheme is that the coefficients in the Arrhenius equation get a physical meaning, which can be measured. The reaction orders equal the stoichiometric coefficients:

$$\widetilde{\nu}_j\prime = \nu_j\prime \qquad \text{bzw.} \qquad \widetilde{\nu}_j\prime\prime = \nu_j\prime\prime \quad . \tag{2.56}$$

In practical matters, collisions between three particles a relevant only.

The reaction is in equilibrium when the forward reaction (eq. 2.52) is as fast as the reverse reaction (eq. 2.53). Rearranging the terms, we get:

$$\frac{k_\mathrm{f}}{k_\mathrm{r}} = \prod_{j\in\mathbf{S}} c_j^{\nu_j} \quad . \tag{2.57}$$

Comparing with eq. 2.48 yields

$$K_\mathrm{c} = \frac{k_\mathrm{f}}{k_\mathrm{r}} \quad . \tag{2.58}$$

2.2.2 Gas-phase reactions

The set of all gas-phase reactions in an mechanism shall be denoted by $\mathbf{R}_\mathrm{g}$. These reactions only involve gas-phase species in $\mathbf{S}_\mathrm{g}$. If only the collision efficiency of a species matters, we can formally take it as a representant of the set $\mathbf{S}_\mathrm{M}$. In this case ν_i must be zero ($i \in \mathbf{S}_\mathrm{M}$).

The reaction rate of each gas-phase species is then computed as a sum over the reaction rates of each reaction:

$$\dot{\omega}_i = \frac{\mathrm{d}c_i}{\mathrm{d}t} = \sum_{k\in\mathbf{R}_\mathrm{g}} \nu_{ik}\, k_k \prod_{j\in\mathbf{S}_\mathrm{g}\cup\mathbf{S}_\mathrm{M}} c_j^{\widetilde{\nu}_{jk}\prime} \quad . \tag{2.59}$$

2.2.3 Pressure-dependent gas-phase reactions

Various dissociation and recombination reactions are treated like elementary reactions for simplicity. They always involve a third body in the collision process (symbol M). Due to complex interactions between the reactants, the reaction coefficient k depends on the concentration of M. According to the Lindemann theory, one can observe a direct proportionality in the low-pressure limit, where as in the high pressure limit we get saturation. Therefore we can define the two rate-limiting constants:

$$k_0 = \lim_{[\mathrm{M}]\to 0} \frac{k}{[M]} \quad . \tag{2.60}$$

and

$$k_\infty = \lim_{[\mathrm{M}]\to \infty} k \quad . \tag{2.61}$$

These two constants can be modeled by Arrhenius equations (eq. 2.55)

The Lindemann theory further yields for the rate coefficient with given concentrations:

$$k = k_\infty \left(\frac{p_\mathrm{r}}{1+p_\mathrm{r}}\right) F \quad , \tag{2.62}$$

where p_r is the reduced pressure

$$p_\mathrm{r} = \frac{k_0\,[\mathrm{M}]}{k_\infty} \tag{2.63}$$

and F the so-called *pressure fall-off blending function*. In the most simple case, the so-called Lindemann equation, the latter one equals unity. Two more formalisms have benn implemented in DETCHEM.

Troe reactions

$$\log_{10} F = \left[1 + \left(\frac{\log_{10} p_\mathrm{r} + c}{n - d\,(\log_{10} p_\mathrm{r} + c)}\right)^2\right]^{-1} \log_{10} F_\mathrm{cent} \quad . \tag{2.64}$$

With

$$c = -0,4 - 0,67 \log_{10} F_\mathrm{cent} \quad , \tag{2.65}$$
$$n = 0,75 - 1,27 \log_{10} F_\mathrm{cent} \quad , \tag{2.66}$$
$$d = 0,14 \tag{2.67}$$

and

$$F_\mathrm{cent} = (1-\alpha)\mathrm{e}^{-T/T^{***}} + \alpha\,\mathrm{e}^{-T/T^{*}} + \mathrm{e}^{-T^{**}/T} \quad . \tag{2.68}$$

The parameters α, T^{***}, T^{*} und T^{**} are the so-called Troe parameters and they are used to fit experimental data.

SRI reactions

$$F = d\left[a \exp\left(\frac{-b}{T}\right) + \exp\left(\frac{-T}{c}\right)\right]^X T^e \tag{2.69}$$

where

$$X = \frac{1}{1 + (\log_{10} p_\mathrm{r})^2} \tag{2.70}$$

2.2.4 Logarithmic interpolation of pressure dependency

For compatibility with published gas-phase mechanisms, empirical formulations of pressure dependency can be used. Suppose, rate constants k_i for various pressures p_i have been determined. The rate constant k is then interpolated for pressures $p_i < p < p_{i+1}$ using a logarithmic scale:

$$\ln k = \ln k_i + (\ln k_{i+1} - \ln k_i)\,\frac{\ln p - \ln p_i}{\ln p_{i+1} - \ln p_i} \tag{2.71}$$

2.2.5 Surface Reactions

The description of surface reactions has been chosen in analogy to the gas-phase reactions. However, since there exist almost innumerable numbers of different surface structures, only little is know about the exact reaction paths. Instead of looking at interactions between single adsorbed species, we will use a mean-field approximation for the description of the surface. Thus, the adsorbate on the surface are described in terms of locally varying coverage fractions that represent concentrations of randomly distributed particles.

The set of surface reactions shall be denoted by $\mathbf{R}_\mathrm{s}$. In the case of the existence of more than on surface type, it can consist of several disjoint subsets $\mathbf{R}_\mathrm{s}^j \quad (j = 1, 2, \ldots)$.

The surface reaction rate is then given in analogy to eq. 2.59:

$$\dot{s}_i = \sum_{k \in \mathbf{R}_\mathrm{s}} \nu_{ik}\, k_k \prod_{j \in \mathbf{S}} c_j^{\widetilde{\nu}_{jk}'} \quad . \tag{2.72}$$

Using an Arrhenius expression (eq. 2.55), local interactions between adsorbate and their influence on the rate coefficients can be modeled in the mean-field approximation by an additional factor:

$$k_k = A_k\, T^{\beta_k} \cdot \exp\left(-\frac{E_{\mathrm{a},k}}{RT}\right) \cdot f_k\left(\{\theta_i\}\right) \quad . \tag{2.73}$$

Some adsorbate may change the energy potentials of the surface, which lead to a change of activation energies for some reactions. Furthermore, the probability of adsorption can be changed, which is associated with the pre-exponential factor. Therfore, the function f_k shall be chosen as follows:

$$f_k = \prod_{i \in \mathbf{S}_\mathrm{s}} \theta_i^{\mu_{ik}} \cdot \exp\left(\frac{\epsilon_{ik}\theta_i}{RT}\right) \quad . \tag{2.74}$$

Here μ_{ik} and ϵ_{ik} are additional model parameters.

A more convenient way to describe adsorption reactions is given in terms of sticking coefficients S_i^0. They basically quantify a probability ($0 \leq S_i^0 \leq 1$) that a particle hitting the surface is adsorbed. First of all, the probability depends on the existence of suitable adsorption sites. Furthermore, lateral interactions with other adsorbed species may have an influence. Therefore, one can define a local adsorption probability by

$$S_i^{\text{eff}} = S_i^0 \cdot \prod_{j \in \mathbf{S}_s} \theta_j^{\nu_{jk}' + \mu_{jk}} \quad . \tag{2.75}$$

The reaction rate $\dot{s}_i$ can be computed using the kinetic theory of gases by

$$\dot{s}_i = S_i^{\text{eff}} \sqrt{\frac{RT}{2\pi M_i}}\, c_i \quad . \tag{2.76}$$

The latter equation assumes a Boltzmann distribution of molecular velocities near the surface. However, Motz and Wise showed that this assumption is not fulfilled for sticking probabilities close to unity. In this case the effiective sticking coefficient should be corrected by

$$S_i^{\text{MW}} = \frac{S_i^{\text{eff}}}{1 - \frac{1}{2} S_i^{\text{eff}}} \quad . \tag{2.77}$$

2.3 Modeling of transport and surface reactions in washcoats

This section describes the washcoat models of DETCHEM to take into account the pore diffusion within a porous washcaot. DETCHEM provides two washcoat models. A simple model, which is based on the concept of effectiveness factors, and a detailed approach, which is based on solving reaction-diffusion equations within the washcoat.

2.3.1 Diffusion and reaction in porous catalysts

In a porous catalyst, the reaction occurs inside the pores: therefore the reactants must diffuse into the catalyst. As the reactants diffuse into the catalyst they encounter active sites and some of the reactants react. Therefore the reactant concentration declines as the distance into the catalyst increases, giving rise to a concentration gradient inside the catalyst. If the diffusion velocity is much lower than the intrinsic rate, the concentration gradients can be quite large. Also, the intrinsic rate depends on the gas phase concentration at the active sites. Hence, it is necessary to have a mean of calculation the effective rate of reaction at the gas-wall boundary, for a catalytic wall with variable concentration profiles inside.

2.3.2 Detailed washcoat model

The detailed washcoat model solves the reaction-diffusion equations in radial direction for every species i within the washcoat.

$$\frac{\partial j_i}{\partial r} - \gamma \dot{s}_i = 0 \tag{2.78}$$

The radial diffusion flux of species i is calculated by

$$j_i = -D_{\text{eff},i} \frac{\partial c_i}{\partial r} \tag{2.79}$$

where c_i = concentration of species i within the washcoat, γ = ratio of catalytic active area to washcoat volume and $D_{\text{eff},i}$= effective diffusion coefficient.

Implementation: In order to solve the reaction-diffusion equations, the washcoat is discretized according to the method of finite differences. The reaction rate at the gas-wall boundary is determined from the diffusion flux at the gas-wall boundary.

Application: The detailed washcoat model is a time consuming model. Depending on the reaction system, the calculation times can rise by one or two orders of magnitude. The benefit of this model is that it can be applied to every reaction system. Also, the concentrations and coverages within the washcoat can be calculated.

2.3.3 Effectiveness factor approach

Consider the reaction-diffusion equation (2.78) for a species with the folowing conditions:

- The species is consumed by a reaction with a first-order rate law, i.e. $\dot{s}_i = -kc_i$.
- The diffusion coefficient is constant.
- The washcoat is thick enough to assume zero concentration gradients at deepest point of the washcoat at position L.

In this case, there is an analytical solution of equation 2.78

$$c(r) = \frac{e^{-\phi}}{e^{\phi} + e^{-\phi}} c_{i,0} \cdot e^{\phi r/L} - \frac{e^{\phi}}{e^{\phi} + e^{-\phi}} c_{i,0} \cdot e^{-\phi r/L} \tag{2.80}$$

where the so-called Thiele modulus

$$\phi = L \sqrt{\frac{k}{D_{\text{eff},i}}} \tag{2.81}$$

has been used for convenience.

The diffusion flux at the boundary between the gas phase and the washcoat can be seen as an effective reaction rate $\bar{\dot{s}}_i$.

$$\bar{\dot{s}}_i = j_{i,0} = -\tanh(\phi) \cdot \sqrt{kD} c_{i,0} \tag{2.82}$$

This expression can be compared to the surface reaction rate $\dot{s}_i$ of the same amount of catalyst without diffusion limitations.

$$\dot{s}_i = -kLc_{i,0} \tag{2.83}$$

The ratio of the two rate expressions is called effectiveness factor η_i, which can be calculated analytically.

$$\eta_i = \frac{\bar{\dot{s}}_i}{\dot{s}_i} = \frac{\tanh(\phi)}{\phi} \tag{2.84}$$

Implementation: If the effectiveness factor model is used, first the reaction rate $\dot{s}_i$ is calculated. From this, the effective rate constant k, the Thiele modulus ϕ and the effectiveness factor η_i are calculated for a user-defined species. It is important that this species is representative for the consumption of all reactants. Choosing a species with vanishing concentration may yield non-realistic effectiveness factors. Since conservation of mass is required, all reaction rates of a surface are multiplied by the same effectiveness factor.

Application The effectiveness factor model is a very fast model. The calculation times are nearly the same with or without this model. Because of the application of the same effectiveness factor for the mean reaction rates of all species, the model can only be applied for reaction systems which are determined by the consumption of a major species.

2.3.4 Effective diffusion coefficients

In both washcoat models effective diffusion coefficients are needed. Depending on the pore size distribution of the washcoat different models can be used.

1. Knudsen diffusion coefficient

$$D_{\text{knud},i} = \frac{\epsilon_p}{\tau} \frac{d_p}{3} \sqrt{\frac{8RT}{\pi M_i}} \tag{2.85}$$

2. Molecular diffusion coefficient

$$D_{\text{eff},i} = \frac{\epsilon_p}{\tau} D_{\text{mol},i} \tag{2.86}$$

3 Mixed diffusion coefficient

$$D_{\text{eff},i} = \frac{\epsilon_p}{\tau} \bar{D}_i \tag{2.87}$$

$\bar{D}_i$ is defined as

$$\frac{1}{\bar{D}_i} = \frac{1}{D_{\text{mol},i}} + \frac{1}{D_{\text{knud},i}} \tag{2.88}$$

4. Random pore model

$$D_{\text{eff},i} = \bar{D}_{M,i}\epsilon_M^2 + \frac{\epsilon_\mu^2(1+3\epsilon_M)}{1-\epsilon_M}\bar{D}_{\mu,i} \tag{2.89}$$

where $\bar{D}_{M,i}$ and $\bar{D}_{\mu,i}$ is defined as

$$\frac{1}{\bar{D}_{M,i}} = \frac{1}{D_{\text{mol},i}} + \frac{1}{(D_{\text{knud},i})_M} \tag{2.90}$$

and

$$\frac{1}{\bar{D}_{\mu,i}} = \frac{1}{D_{\text{mol},i}} + \frac{1}{(D_{\text{knud},i})_\mu} \tag{2.91}$$

In all the above models ϵ_p, τ, d_p, M_i, $D_{\text{mol},i}$, $D_{\text{knud},i}$, $D_{\text{eff},i}$, ϵ_M, ϵ_μ represent the porosity, tortuosity, pore diameter, molecular weight of species i, Binary diffusion coefficient of species i, Knudsen diffusion coefficient of species i, Molecular diffusion coefficient of species i, porosity for macro pores and porosity for micro pores respectively.

Chapter 3

Input files

This chapter briefs about the input files used in DETCHEM models.
Other than the files described in this chapter there are model specific input files and the **CHEMINP** generated ***detchem.inp*** file. Model specific input files are explained in the specific chapters for each model and ***detchem.inp*** is explained in chapter 4.

3.1 General structure of inp-files

One of the major changes comparing version 2.0 with previous versions of DETCHEM is the syntax of input files. Except for the databases (***thermdata***, ***moldata*** and the mechanism files) all application specific data is now specified using the same format. These files should be assigned an application specific name with the extention ***.inp***.

The generic structure of such input files can be described as follows:

```
{command}
<tag1>
  option1 = value     # comment1
  option2   value     # comment2
  option3 = "string"  # comment3
  option4             # comment4
  <tag2>
    option5 = value
  </tag2>
</tag1>
```

There are several items:

- tags,
- options,
- values (numerical, strings or logical),
- commands, and
- comments.

All items must be separated by spaces, line breaks, commas (,), or equal signs (=). The number of delimiters are not significant. A line break is also required behind the last tag in any input file. Tags, options and commands do not distinguish small from capital letters, however in strings and names (e.g. species names) they are treated differently. The length of a line is limited to 999 characters, each option name can be up to 32 characters.

3.1.1 Tags

Tags are used to group data in the input file. They always consist of a pair in sharp brackets: `<tag> ... </tag>`. They define a hierarchy in the input file. Therefore do not forget the closing tag at the appropriate place. An error message will be generated in case of an unknown option or insufficient information given within the tag-environment.

3.1.2 Options and Values

The user defined data is specified using options. There are several types of options, most of them require a value, but some do not. Depending on the type of information, the values can be integers, floating point numbers, strings or logicals. Integers are numbers that may contain a negative sign, but no decimal point or exponent, whereas all FORTRAN type formats are accepted for floating point numbers (e.g. 0.0001, 1e-4, 1.d-04). Logical values are `yes` and `no` (or shorter `y` and `n`). All character combinations that do not contain any delimiting character can also be interpreted as strings. To avoid confusion or to include delimiting characters (especially blanks) or other special characters (#,!,=, etc.), use quotation marks.

In the later chapters we use a notation like in the following table to list the members of a tag.

<**TAG_NAME**>	tag
<*tag_name*>	tag with variable name
option1=*d*	floating point argument
option2=*i*	integer argument
option3=*b*	logical argument (yes or no)
option4=*s*	string argument
option5	option without argument
value	value without option name

3.1.3 Commands

The input file parser supports a few basic commands that help increasing the reusability of input files and debugging. Commands are always included in curly brackets as shown in the following example.

```
{verbose y}
{include filename1}
{include filename2}
{define temperature 523}
....
....
{ifdef temperature label1}
   T = 273               # use default temperature
{goto label2}
{label label1}
   T = {get temperature} # use defined temperature
{label label2}
....
....
```

include *filename* is used to include other files which are present in the working directory. This command can be used to increase readability and reusability of the input file. For instance, data commmon to several simulations like species input data can be saved in a separate file. By including the external file, it can be easily established that all simulations use the same data. An included file can also contain an include command. A maximum of five recursions is allowed.

path *pathname* can be used to define a default path name where DETCHEM looks for include files as well as for ***thermdata*** and ***moldata*** files. If a path name is given, DETCHEM will first search any subsequent file in this path directory. Only in case it is not found in the path directory, DETCHEM will open a file in the local working directory. An error message is displayed if an include file is present neither in the path directory nor in the local directory. The default path in any case is the local directory **"."**.

verbose *yes_or_no* switch screen output on or off. When activated, the parser repeats the lines of the input files while the program reads them. This command is especially useful for debugging purposes. The user can find out which data was read and where the program has terminated in case an error occurred.

The parser also provides the possibility to define user variables in the input file. Variables can be decalred anywhere in the input file using the command **define** *variable_name value*. The *variable_name* can be any combination of letters (case sensitive), numbers and some special characters (e.g., _,+,−). *value* can also be any expression whout blank characters. If the parser later finds the commands **get** *variable_name*, it will be replaced by the content of *value*. Depending on the context where the variable is inserted, *value* can be interpreted as string or numeric value. Each variable can only be defined once. Attempting to redefine a variable will create an error message.

For numeric variables the **get** command can also perform simple arithmetic calculations. In this case, use it in the form

{ **get** *v1 operator v2* }

Here, *v1* and *v2* are either constant number expressions (e. g. 5, 1e10, -0.003) or variable names containing numerical values. The operators can be chosen from `+`, `-`, `*` and `/` . Please note to separate arguments and operator by at least one blank character.

A series of numeric values (e.g. for temperature inputs) can be generated using the **do** command. It can be used in the forms
{ **do** *value_from value_to* }
or
{ **do** *value_from value_to step_size* }
Here, *value_from*, *value_to*, and *step_size* are either constant number expressions or variable names containing numerical values. If *step_size* is omitted, a step size of 1 is used.

Example:

```
{define Tstep 25}        # define a variable
...
<TPROFILE>
  {do 300 400 Tstep}     # is replaced by: 300 325 350 375 400
</TPROFILE>
....
```

The pair commands **goto** *label_name* and **label** *label_name* allows skipping of sections in the input file. On encounter of a **goto** command all following lines will be ignored until the matching **label** command is found. That implies that only forward jumps are possible.

A simple branching is possible by the command **ifdef** *variable_name label_name*. This command checks if a variable *variable_name* is defined without looking at its value. If yes, **goto** *label_name* is executed. Otherwise the parser continues at the next available statement.

3.1.4 Comments

Text in one line preceeded by a # or ! is treated as comment statements in the input file. We recommend the use of # over the other one.

3.1.5 Units

In DETCHEM version 2.3 a new feature was added to DETCHEM input files. Most physical properties, which are usually treated to be given in SI-units, can automatically be converted by the input file parser from arbitrary units into the required standard unit. Where possible, a check for consistency of the unit is made. Thus, by stating a unit explicitly in the input file, you can avoid confusion of physical properties. For instance, a step size required in units of time (i. e. seconds) will be refused if the value is given in meters.

For instance, an input file could consist of the following lines:

```
<myproperties>
  p=1[bar]        # pressure
  V=5[L]          # volume
  T=473[K]        # temperature
  Vdot=2[L/min]   # volume flux
  Cp=1.5[kJ/kg/K] # heat capacity
</myproperties>
```

Thus, a unit follows a numerical value using square brackets [ ]. You can leave blanks between the numerical value and the brackets, but the unit must be on the same line of the input file.

The unit symbol is case sensitive. While **[mm]** stands for 10^{-3} meters, **[Mm]** means 10^{6} meters. Each single unit symbol can be followed by an integer exponent, e. g. a cubic centimeter is written as **[cm3]**. To multiply two unit symbols, simply separate them by a blank character or use the symbol *****. That is, both **[N m]** and **[N*m]** symbolize the unit Newton-meter. The slash charcater **/** is used to indicate that the next unit appears in the denominator of a fraction. However, notice that it only applies to one following unit symbol. In order to specify the unit J/(kgK) write **[J/kg/K]** or **[J kg-1 K-1]**.

SI-units usually can be prefixed to abbreviate powers of ten. If a unit is prefixable, you can use the prefixes **h** for 10^{2}, **k** for 10^{3}, **M** for 10^{6}, ... (**G**,**T**,**P**,**E**,**Z**,**Y**); and **d** for 10^{-1}, **c** for 10^{-2}, **m** for 10^{-3}, **u** for 10^{-6}, ... (**n**,**p**,**f**,**a**,**z**,**y**). Note that micrometers are written as **[um]** instead of **[μm]**.

Usually DETCHEM expects temperatures in Kelvin **[K]**. Now you can also use Celsius **[°C]** or Fahrenheit **[°F]**. Note however that the parser does not check if the use of these temperature units is allowed in a specific context. When the program expects a temperature difference, where you enter **10[°C]**, the program will use a step size of 283.15 K! Also notice that the use of °C and °F is not recommended in compound units, e. g. always use **[J/K]** instead of **[J/°C]**.

The folowing table lists the known unit symbols:

Quantity	Unit	Prefixable
None	1	
	1 % = 10^{-2}	
	1 ppm = 10^{-6}	
	1 ppb = 10^{-9}	
Length	1 m	yes
	1 Angstrom = 10^{-10} m	
	1 in = 25.4 mm	
	1 mil = 1/1000 in	
	1 ft = 12 in	
	1 yd = 3 ft	
	1 mile = 1760 yd	
Volume	1 l = 1 L = 1 dm^3	yes
Time	1 s	yes
	1 min = 60 s	
	1 h = 60 min	
	1 d = 24 h	
Mass	1 kg	yes
	1 lb = 453.59237 g	
	1 oz = 1/16 lb	
Temperature	1 K	
	0 °C = 273.15 K	
	0 degC = 0 °C	
	32 °F = 0 °C ; 212 °F = 100 °C	
	32 degF = 32 °F	
Quantity	1 mol	yes
Frequency	1 Hz = 1 s^{-1}	yes
Force	1 N = 1 kg m s^{-2}	yes
	1 lbf = 4.44822 N	
Pressure	1 Pa = 1 N/m^2	yes
	1 bar = 10^5 Pa	yes
	1 mmHg = 101325/760 Pa	
	1 psi = 1 lbf/in^2	
Energy	1 J = 1 N m	yes
	1 eV = $1.602176487 \cdot 10^{-19}$ J	yes
	1 cal(IT) = 4.1868 J	yes
	1 cal = 1 cal(th) = 4.184 J !!! changed in version 2.7 !!!	yes
Power	1 W = 1 J/s	yes
Electrical Current	1 A	yes
Voltage	1 V = 1 W/A	yes
Electrical Resistance	1 Ohm = 1 V/A	yes
Electrical Charge	1 C = 1 A s	yes
Electrical Dipole Moment	1 Debye = $3.33564 \cdot 10^{-30}$ C m	

In DETCHEM version 2.7 you can also define more units. The volumetric quantities galon, quart, pint and liquid ounce have different definitions in Britain and the US. If you want to use any of these units, you need to add them by a command like

```
{use_imperial_units}
volume = 1 [pt] # one glass of cider
```

or

```
{use_US_units}
volume = 10 [gal] # gasoline for your car
```

The exact definitions are as follows:

{use_imperial_units}	{use_US_units}
1 gal = 277.42 in^3	1 gal = 231 in^3
1 qt = 1/4 gal	1 qt = 1/4 gal
1 pt = 1/2 qt	1 pt = 1/2 qt
1 floz = 1/20 pt	1 floz = 1/16 pt

Furthermore, you can define your own user-defined units in the following form

```
{unit cP = 1e-3 [Pa s]} # new unit: centiPoise
viscosity = 2.5 [cP] # dynamic visosity
```

That is, a new unit is defined by the command **unit** (in curly brackets) followed by a symbol for the new unit (without brackets), an optional equal sign (=), a numerical value and an existing unit (in square brackets). After this definition, the new unit can be used like all other units throughout the whole input file.

You may use user defined units, for instance, to do conversions of input data automatically. As an example, consider a plug-flow reactor. The volume flux of your inlet gases is given in terms of gas-hourly space-velocity. However, in the input file, you need to enter a velocity. You can do the calculation in the input file:

```
{unit Length = 10 [cm]} # length of reactor
{unit Radius = 0.5 [cm]} # radius of reactor
{unit Area = 3.1416 [Radius^2]} # cross-sectional area = pi*r^2
{unit Volume = 1 [Area*Length]} # volume of the reactor
{unit GHSV = 1 [Volume/h]} #  volume flux
...
<SECTION>
   length = 1 [Length]
   rin = 1 [Radius]
   ...
</SECTION>
<INLET>
   u0 = 40000 [GHSV/Area]  # convert GHSV into velocity
   ...
</INLET>
```

3.2 thermdata

The thermodynamic database contains enthalpy, entropy and heat capacity data for each species in the NASA Equilibrium code [10]. The calculation of thermodynamic data from the coefficients $a_1 \ldots a_7$ is described by equation 2.13, 2.16 and 2.17. There are two temperature intervals with different sets of coefficients to achieve better accuracy. The first seven entries are the values $a_1 \ldots a_7$ for the temperature interval $T_{\text{jump}} < T < T_{\text{high}}$, the next seven entries are the values $a_1 \ldots a_7$ for the temperature interval $T_{\text{low}} < T < T_{\text{jump}}$.

```
----------------------------------------
Example 3.1.1: thermdata (selection)
----------------------------------------------------------------------
H2               J 3/77 H   2    0    0    0G   300.000  5000.000
 0.30667095E+01 0.57473755E-03 0.13938319E-07-0.25483518E-10 0.29098574E-14
-0.86547412E+03-0.17798424E+01 0.33553514E+01 0.50136144E-03-0.23006908E-06
-0.47905324E-09 0.48522585E-12-0.10191626E+04-0.35477228E+01
----------------------------------------------------------------------
```

The following format is used for the section above, selected from the ***thermdata*** file.

line 1: The first 8 characters in this line are used to define the chemical species with the format specification(A8),which is followed by comments with (A16) format. The next 20 columns are reserved to specify the atoms present in the defined chemical species with the format 4(A5). In the above example H represents the atom and 2 stands for the number of atoms present. The next 3 columns are reserved for the species type, but not evaluated in the present version. The rest of the columns are used to define the temperature intervals as follows. Low temperature T_{low} (E10.0) and high temperature T_{high} (E10.0), jump temperature T_{jump} (1000K unless specified)(E8.0).

line 2-4: Coefficients $a_1 \ldots a_7$ for upper (first seven rules) and lower (next seven rules) temperature interval (5(E15.8) per line).

In the case of reversible reactions this data is used to calculate the kinetic data for the reverse reaction. And also the thermodynamic data for the gas-phase species are used to calculate the reaction enthalpies. If the thermodynamic data of the adsorbed species are unknown and if these species are not involved in reversible reactions, dummy values (e.g 0.0) can be given. A section of ***thermdata*** file with dummy values is shown below.

```
------------------------------------------------
 Example 3.1.2: thermdata (selection)
------------------------------------------------------------------------------
H2O(PD)          92491  O   1H   2PD  1     I    300.00   3000.00 1000.00
0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00
0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00
0.00000000E+00 0.00000000E+00 0.00000000E+00 0.00000000E+00
------------------------------------------------------------------------------
```

3.3 moldata

The ***moldata*** file contains kinetic theory parameters of gas-phase chemical species.

```
-------------------------------------------
Example 3.2.1: moldata (selection)
-----------------------------------------------------------------------
H2           1   38.000    2.920   0.000   0.790   280.000  Wa.(82)
-----------------------------------------------------------------------
```

A section of ***moldata*** is shown above, where the entries have the following meaning

H2	Species Symbol
1	Index indicating whether the molecule has a monoatomic, linear or non-linear geometrical configuration (0= single atom, 1= linear, 2=nonlinear)
38.000	Lennard-Jones potential well depth ε/k_B in K
2.920	Lennard-Jones collision diameter σ in Angstroms
0.000	Dipole moment μ in Debye
0.790	Polarizability α in cubic Angstrom
280.000	Rotational relaxation collision number Z_{rot} at 298 K
Wa.(82)	Comment

The entries are not required to be listed in fixed columns, but they must be separated by one or more blank characters. Any information given beyond column 80 is ignored. Due to compatibility with older DETCHEM versions, an additional value can be given between the second and third column, which used to contain the molar mass of a species. However, this value is ignored when specified, i.e. the third column is ignored when six instead of five numerical values are listed.

For codes that evaluate transport coefficients, e.g. diffusion coefficients, all gas-phase species must have an entry in this database, otherwise the program will stop execution with an error message. If transport coefficients are not required (as in pure equilibrium calculations), only a warning is displayed for missing data.

3.4 Gas-phase chemistry mechanism file

The gas-phase chemistry mechanism file lists the elementary reactions occurring in the gas-phase.This file can be give any name, and the same should be mentioned in the ***mech.inp*** file (description follows). For convenience, the user is advised to copy and modify the lines, while keeping the same format. An example of gas-phase reaction mechanism file is shown below.

```
-------------------------------------
Example 3.3.1: noo2m
-----------------------------------------------------
MECHANISM OF THE NO-O2 REACTION (P. KLAUS 1997)
 ****
O      +O     +M(1)   =O2    +M(1)    2.900E+17    -1.000   0.0
NO2    +M(1)  =NO     +O     +M(1)    1.100E+16     0.0   275.88
NO2    +O     =NO     +O2             1.000E+13     0.0     2.51
NO2    +NO2   =NO     +NO    +O2      1.600E+12     0.00  109.19
END
000 COMPLEX REACTIONS
END
COLLISION EFFICIENCIES:
M(1)    =O2    +N2
         0.40   0.40
END
-----------------------------------------------------
```

In the above example each reaction is given in one line. Irreversible reactions (denoted by >) as well as reversible ones (denoted by =) can be specified. The reaction equations can have a maximum of 3 species on the

left and three species on the right side of the equation, however the total number of species in an equation is not allowed to exceed five. A species can contain a maximum of 8 characters at maximum followed by a separator (+,>,=) unless the end of the equation is reached. Then the kinetics data follows. The format of the whole line is: 5(A8,A1), E10.3, F7.1, F10.1. Unless familiar with FORTRAN nomenclature, just copy and modify wherever appropriate, keeping the given format.

The reaction rate is calculated by using concentrations in the units mol/cm^3. The rate coefficients are given in terms of modified Arrhenius expression as stated in equation 2.55. The input order is assigned as follows.

A_k value in the first column in cm, s, mol (according to the reaction order.
β_k value in second column.
E_a value in third column in kJ/mol For reversible reactions, the reverse reaction rates are derived from the equilibrium constants using Equations 2.48. All of the reaction orders are according to the stoichiometry of the reaction.

The symbol M(1) in the mechanism denotes third bodies as described in section 2.1.3. The last part of the *gas-phase mechanism* file denotes the collision efficiencies for these third bodies, if reaction with third bodies are specified. The collision are by default unity for the species not given in the list. In example 3.3.1, the collision efficiencies of the species O_2 and N_2 are modified to be 0.4 instead of unity. Information on proper collision efficiencies will be provided in connection with reaction mechanisms. If the collision efficiencies for all species in a reaction are unity, them plain symbol M can be used.

```
-------------------------------------
Example 3.3.2:
-------------------------------------------------------------------
H     +CH2O          +M(2) =CH2OH    +M(2)   0.540E+12   0.5   15.070
                                     LOW    1.270E+32 -4.820 27.32
                                     TROE .7187 103.0 1291.0 4160.0
-------------------------------------------------------------------
```

Aside from the conventional way to model the reaction rate by an *Arrhenius* expression, a *Troe* expression can be used, which is very useful to include the additional pressure dependence of recombination and dissociation reactions. The rate law here becomes more complicated; the 3.3.2 illustrates the format used to define *Troe reactions* by ten parameters. The data in the first line are A_∞, β_∞, $E_{a\infty}$, the data in the line behind the LOW keywords are A_0, β_0, E_{a0} and the data behind the TROE keyword are a, T^{***}, T^*, T^{**}. If a = 0.5 and T^{***}= T^* = 1E30 then a Lindemann expression is used where the broadening factor F is unity.

3.5 Surface chemistry mechanism file

This file lists the surface elementary reaction mechanism. This file can be given any name and the same must be mentioned in the ***mech.inp*** file (described later). An example of the surface mechanism file is shown below. For convenience, the user is advised to copy and modify the lines, while keeping the same format. An example of surface chemistry mechanism file is shown below.

```
-------------------------------
Example 3.4.1: noo2sm
-----------------------------------------------------------------------
SURFACE MECHANISM OF THE NO-O2 REACTION
****
************************************************************************
****                                                                   *
**** NO-O2 SIMPLE TEST SURFACE MECHANISM ON PT                         *
****                                                                   *
************************************************************************
**** 1. ADSORPTION
STICK
O2       +PT(s)       +PT(s)     >O(s)    +O(s)    7.000E-02  0.0    0.0
STICK
NO       +PT(s)       >NO(s)                       8.500E-01  0.0    0.0
STICK
NO2      +PT(s)       >NO2(s)                      9.000E-01  0.0    0.0
STICK
O        +PT(s)       >O(s)                        1.000E-00  0.0    0.0
***** 2 DESORPTION
O(s)     +O(s)        >PT(s)     +PT(s)   +O2      3.700E+21  0.0  213.0
```

```
$O(s)                                                 0.0         0.0   70.0
NO(s)    >NO          +PT(s)                          1.000E+16   0.0   90.0
NO2(s)   >NO2         +PT(s)                          1.000E+13   0.0   60.0
***** 3 SURFACE REACTION
NO(s)    +O(s)        >NO2(s)   +PT(s)                3.700E+21   0.0   96.3
$O(s)                                                 0.0         0.0   70.0
$NO(s)                                                0.0         1.0   70.0
NO2(s)   +PT(s)       >NO(s)    +O(s)                 3.700E+21   0.0   79.57
END
-----------------------------------------------------------------------
```

The first line of the mechanism must begin with SURF.

The mechanism is given in terms of elementary reactions. Each reaction is given in one line with possible extension lines, one above and/or one below this line. Irreversible reactions (denoted by >) as well as reversible ones (denoted by =) can be specified. The reaction equations can have a maximum of three species on the left and three species on the right side of the equation, however the total number of species in one equation is not allowed to exceed five. A species can contain a maximum of 8 characters followed by a separator (+, >, =) unless the end of the equation is reached. Then, the kinetic data follow. The format of the whole line is: 5(A8,A1), E10.3, F7.1, F10.1. Unless familiar with FORTRAN nomenclature, just copy and modify wherever appropriate, keeping the given format.

The reaction rate is calculated by using concentrations with the units mol/cm^3 in the gas phase and mol/cm2 on the surface. The rate coefficients are given in terms of a modified Arrhenius expression as stated in Equation 2.73. The input order is assigned as follows:

A_k value in the first column (cm,s or mol according to order of the reaction).
β_k value in the second column.
E_a value in the third column kJ/mol.

For reversible reactions, the reverse reaction rates are derived from the equilibrium constants using Equation 2.48

STICK means that a sticking coefficient is given (first column in the subsequent line) for the reaction in the next line. The given coefficient is the initial sticking coefficient S_0. It can be modified by additional parameters β and E_a.

By default, all of the reaction orders are according to the stoichiometry of the reaction, for instance the reaction

```
STICK
O2        +PT(s)       +PT(s)    >O(s)    +O(s)   7.000E-02   0.0   0.0
```

is second order in vacancies (PT(s)) and first order in O2. Here, the value 7.000E-02 is the initial sticking coefficient (= 0.07).

The order of the reaction can be modified, and the activation energy can be made coverage dependent by using the parameters μ_{ik} and ϵ_{ik} in Equation 2.73. The modified data can be given in a line below the reaction equation; this line must start with a symbol followed by the species that the modification refers to (species i in Equation 2.73). Several modifications can be used as shown in Example 3.4.1. The value in the second column modifies the order of the reaction (μ_{ik}), the change to the stoichiometric value is specified here. The value in the third column (ϵ_{ik}) modifies the activation energy. In the following reaction,

```
STICK
H2     +PT(s)    +PT(s)    >H(s)    +H(s)    0.046E-00   0.0   0.0
$PT(S)                                                 -1.0   0.0
```

STICK in the line above the reaction means that 0.046 is the initial sticking coefficient, PT(s) below the reaction line means that the rate coefficients/order of the reaction have an additional coverage dependence in the following way: PT(s) gives the related species (here: PT(s) = vacancies), and a nonzero value in second column (-1.0) modifies the order by -1 for this species, i.e. the reaction order in PT(s) is now first order and not second order. In the following reaction,

```
O(s)   +O(s)     >PT(s)     +PT(s)  +O2      3.700E+21   0.0 213.0
$O(s)                                                    0.0  70.0
```

A is 3.7x10^{21} cm^2/(mol*s), β is 0, and E$_a$ is 213 kJ/mol. O(s) below the reaction line means the rate coefficients/order of the reaction have an additional coverage dependence in the following way (Equation 2.73): The nonzero value in the third column (70.0) reduces the activation energy by 70 kJ/mol if the surface is completely covered with the given species, here O(s).

3.6 Species input

For any DETCHEM application the user needs to define the species that are to be considered. In many cases it is advisible to put the declaration in a separate file, e. g. ***species.inp***, and bind it to the application's input file by an an include statement

```
{include species.inp}
```

The **<SPECIES>** section contains gas-phase species as well as surface species and species from condensed phases. Always there must be exactly one gas-phase ensemble, where as you can have any number of surface ensembles (for reasons of memory allocation the number is usually limited to 5) and any number of different condensed phases. In addition, third-body species can be defined in this section. An example is shown below.

```
<SPECIES>
  <GASPHASE>
    NO    O2    NO2    O    N2
  </GASPHASE>

  <SURFACE name="Pt" mol/cm2 = 2.72e-9>
    PT(s)    O(s)    NO2(s)    NO(s)

    <INITIAL>
      PT(s)    *
      O(s)     0.01
      NO(s)    0.08
    </INITIAL>
  </SURFACE>

  <PHASE liquid>            # liquid is the user-defined name of this phase
      H2O(l)    998         # species and corresponding density are required
  </PHASE>

  <THIRD-BODY>
    M = 0.4 O2 + 0.4 N2
  </THIRD-BODY>
</SPECIES>
```

In order to distinguish different surface types (e.g. Platinum, Rhodium etc.) you can assign an optional string parameter **name** inside the **<SURFACE>** tag (up to 8 characters). The same applies for the **<PHASE>** tag. Further, for each surface a surface site density must be defined. It can be given in moles per square centimeter or moles per square meter. The initial coverage of surface species must be specified under element tag **<INITIAL>**. While specifying the initial coverage, one species can be given * value. In this case the value assigned to the particular species would be 1-(sum of initial coverage of rest of the species). In the example given above Pt(s) would be assigned a value of 0.91. Surface species having zero initial coverage need not be specified.
Within the **<PHASE>** tag, the species must be listed space-separated from their densities in kg/m3.

The tag **<THIRD-BODY>**, which is optional can be used to define reactions involving third bodies. This option is especially useful while defining global reactions with third bodies. In the above example M stands for the third body (It can be any alphabet) and 0.4 is the collision efficiency. If the third body is already defined in the mechanism file, then the definitions of the mechanism will be taken. In short the third body definitions in the mechanism file over-rides the definitions in the **<SPECIES>** section.

Species occupying more than one site

There are instances where an adsorbed (surface) species can occupy more than one site on the given surface. For example some hydrocarbon species have size which needs more than one site to be adsorbed on the surface. In such cases, the information should be specified in the **<SURFACE>** section as shown below.

```
<SURFACE mol/cm2 = 2.7E-09>
  PT(s)
  O(s)
  NO2(s) * 2
  NO(s)
  <INITIAL>
    PT(s)    *
    O(s)     0.01
    NO(s)    0.08
  </INITIAL>
</SURFACE>
```

In the above example NO2(s) occupies 2 surface sites.

Options in <SPECIES> section

<**GASPHASE**>	gas-phase species list
<**PHASE**>	condensed phase species list
<**SURFACE**>	surface species list
<**THIRD-BODY**>	third-body species list

Options in <SURFACE> subsection

name=*s*	surface name, e.g. 'Pt', 'Rh', etc.
mol/cm2=*d*	surface site density [mol/cm^2]
mol/m2=*d*	surface site density [mol/m^2]
***=*i*	number of surface sites occupied by one molecule of species (default: 1)
species name	species name
<**INITIAL**>	initial coverages

The SOC model requires surface species that are linked intrinsically in the model. These species are tagged as **<INTRINSIC_SURFACE>**. The options are the same as for normal **<SURFACE>** species. Since these species do not appear in user defined reaction mechanisms, no ***thermdata*** or ***moldata*** information is required for them.

3.7 Mechanism input

Most DETCHEM applications deal with chemical reaction mechanisms. Therefore the reaction system needs to be declared in the application input file. This is done in the **<MECHANISM>** section of the input file. An example using the classic style of mechanism input files is shown below:

```
<MECHANISM>
  <GASPHASE>
    file="gasphase"
  </GASPHASE>

  <SURFACE name="Pt">
    file="surface"
  </SURFACE>
</MECHANISM>
```

In this case the **<MECHANISM>** tag only acts as a name holder for the external mechanism files. Due to compatibility with older DETCHEM versions, such mechanism files do not follow the new input format and therefore they can not be included directly into the input file. These files must be refered to by using the **file** option.

If several apllications shall use the same sets of reactions, it is advisible to put the mechanism declaration into a separate file, e. g. ***mech.inp***, and inlcude it in the application input file by

```
{include mech.inp}
```

DETCHEM can be used with zero or one gas-phase mechanism and any number (memory allocation defaults to five) of surface mechanisms. In order to distinguish different surface mechanisms, the **name** member can be assigned a string of up to 8 characters. A mechanism is enclosed within a pair of **<GASPHASE>** or **<SURFACE>** tags. However, each mechanism can can constist of an arbitrary number of mechanism files, e. g.

```
<MECHANISM>
  <GASPHASE>
    file="gasphase"
  </GASPHASE>
```

```
  <SURFACE name="Pt">
    file="surface1"
    file="surface2"
  </SURFACE>

  <SURFACE name="Rh">
    file="surface3"
  </SURFACE>

  <SURFACE name="Ni">
    file="surface4"
  </SURFACE>
</MECHANISM>
```

Note: Now, the use of external mechanism files is not mandatory anymore. Each reaction can also be defined directly in the input file using the format free input file syntax.

Gasphase mechanism

The **<GASPHASE>** tag can hold the following types of input:

file=*s*	gas-phase mechanism file
<**UDF**>	code for user-defined functions
<**REACTION**>	Arrhenius reaction
<**LINDEMANN**>	Lindemann reaction
<**TROE**>	Troe reaction
<**SRI**>	SRI reaction
<**PLOG-Reaction**>	reaction with logarithmic pressure interpolation
<**UDF-Reaction**>	reaction with user-defined rate
<**GLOBAL**>	global reaction (obsolete feature)

An external gas-phase mechanism file as described in chapter 3.4 is specified by the **file** option. Alternatively or in addition, the user can define single reactions or whole reaction mechanisms consisting of reactions with various rate laws. The syntax will be described later in this chapter.

Condensed phase mechanism

The **<HOMOGENEOUS>** tag is equivalent to the **<LIQUID>** tag. The mechanisms with reactions in any phase can be listed here. In a reaction all reactants must be from the same phase. The input types are equal to those in the **<GASPHASE>** tag.

Reactions at phase boundaries

Reactions at the interface between two phases with reactants from both phases can be defined in the **<INTERPHASE>** tag. Available options are:

file=*s*	interface mechanism file
<**UDF**>	code for user-defined functions
<**REACTION**>	elementary-step reaction
<**STICK**>	stick reaction
<**UDF-Reaction**>	reaction with user-defined rate
<**GLOBAL**>	global reaction (obsolete feature)

Surface mechanisms

The **<SURFACE>** subsection may contain the following options:

name=*s*	surface mechanism name
file=*s*	surface mechanism file name
<**UDF**>	code for user-defined functions
<**REACTION**>	elementary-step reaction
<**STICK**>	stick reaction
<**UDF-Reaction**>	reaction with user-defined rate
<**GLOBAL**>	global reaction (obsolete feature)

One or more external surface mechanism files (see chapter 3.5) can specified by the **file** option. Alternatively or in addition, the user can define stick reactions, elementary-step reactions with Arrhenius rate laws or complex (global) reactions with user-defined rate laws. The syntax will described later in this chapter.

If the logical option **MotzWise** is set, then the Motz-Wise correction for stick reactions will be applied.

Reaction strings

When defining a reaction in this input format, the user can write the reaction equation in a natural way. It is only required to separate names, stoichiometic coefficients and operators by at least one blank character. Thus the following definitions are valid reaction strings

```
O2 + H = O + OH
NO2 + NO2 = NO + NO + O2
CH4 + 2 O2 > CO2 + 2 H2O
C3H6 +  4.5 O2 > 3 CO2 + 3 H2O
```

Use a + sign to separate two reactants or two products. The > and = symbols are used for a non-reversible and reversible reactions respectively. For gas-phase reactions also non-integer stoichiometric coefficients are allowed. However, be careful when using non-iteger coefficents. The balance is only checked up to a precission of 10^{-6}. Thus rounding errors may occur when writing 0.3333333 for the value $1/3$.

In the current DETCHEM version, elementary surface reactions are limited to three reactants and three products without additional stoichiometric conditions. These limitations can be avoided by defining global surface reactions.

<REACTION>

Both, gas-phase and surface reactions can be declared in the input file by using the **<REACTION>** tag. Each tag holds one reaction and its Arrhenius parameters as shown in the example below:

```
<REACTION>
   NO2 + NO2 = NO + NO + O2
   A/cm_units = 1.6e12
   Ea         = 109.19 [kJ/mol]
</REACTION>
```

The Arrhenius rate coefficients need to be declared, i. e. with a pre-exponential factor A, a temperature exponent β and an activation energy E_a. The pre-exponential factor and the activation energy can be given in a variety of units. It is recommended to use an explicit unit for the activation energy, e. g. [**kJ/mol**] or [**kcal/mol**] The possible options are summarized below:

A/SIunits=d	pre-exponential factor [mol, m, s, K]
A/cm_units=d	pre-exponential factor [mol, cm, s, K]
beta=d	temperature exponent (default: 0)
Ea=p	activation energy with unit (default: J/mol)
Ea/J_mol=d	activation energy [J/mol]
Ea/kJ_mol=d	activation energy [kJ/mol]
Ea/R=d	activation energy / R [K]
cflag=s	character flag
<**COV**>	coverage dependency (surface reactions only)

The rate expression for surface reactions can further be modified by coverage dependent factors. Such additional factors are defined using the **<COV>** tag, the details of which are given later in this chapter.

The character flag is only used for special applications like sensitivity analysis. It will be described in detail where necessary.

Pressure dependent reactions

As described in chapter 2.2.3, gas-phase reactions can introduce a pressure dependency based on Lindemann theory. Common to all these reactions is a third-body species that need to appear on both sides of the reaction equation. After the reaction equation, two sets of Arrhenius parameters need to be declared for k_0 and k_∞. The declaration is similar to standard Arrhenius reaction parameters, e. g.

```
<LINDEMANN>
   OH + OH + M1 =  H2O2 + M1
   A_inf/cm_units = 7.2e+13
   beta_inf       = -0.37
```

```
   Ea_inf         = 0 [kJ/mol]
   A_0/cm_units   = 1.45e+18
   beta_0         = 0
   Ea_0           = 0 [kJ/mol]
</LINDEMANN>
```

<TROE>

The pressure fall-off curve according to the Troe formulation (see chapter 2.2.3) needs additional parameters in the mechanism input file, e. g.

```
<TROE>
   H2O2 + M1 = OH + OH + M1
   A_inf/cm_units = 4e11
   Ea_inf         = 155 [kJ/mol]
   A_0/cm_units   = 4e16
   Ea_0/          = 183 [kJ/mol]
   alpha          = 0.62
   T***           = 58
   T*             = 2700
   T**            = 21700
</TROE>
```

The reaction equation is written in the same way as explained before. A third-body species must appear in the reaction equation on both sides. Then, the two rates coefficients k_∞ and k_0 are defined by their corresponding Arrhenius parameters. Finally, the four Troe parameters need to be defined. The available parameters are summarized below:

alpha=*d*	Troe parameter alpha
T*=*d*	Troe parameter T*** [K]
T*=*d*	Troe parameter T* [K]
T**=*d*	Troe parameter T** [K]
cflag=*s*	character flag

<SRI>

Likewise, reactions with a pressure fall-off curve according to the SRI model (see chapter 2.2.3) can be defined

```
<SRI>
      C2H6 + M2   =  C2H5 + H + M2
      A_inf/cm_units = 8.85e+20
      beta_inf       = -1.228
      Ea_inf         = 102210 [cal/mol]
      A_0/cm_units   = 6.9e+42
      beta_0         = -6.431
      Ea_0           = 107175 [cal/mol]
      a = 47.61
      b = 16182
      c = 3371
</SRI>
```

The available additional parameters are summarized below:

a=*d*	SRI parameter *a*
b=*d*	SRI parameter *b* [K]
c=*d*	SRI parameter *c* [K]
d=*d*	SRI parameter *d* (default: 1)
e=*d*	SRI parameter *e* (default: 0)

<PLOG-Reaction> – Logarithmic interpolation of pressure dependency

An alternative way to treat pressure dependent gas-phase reactions is by logarithmic interpolation of the rate constants (see chapter 2.2.4). The rates are fitted for given pressures by Arrhenius expressions. The rate at the current pressure is given by logarithmic interpolation (Eq. 2.71). A generic example is shown below:

```
<PLOG-Reaction>
spec1 + spec2 > spec3 + spec4
p_unit = 1 [atm]        # default
c_unit = 1 [mol/cm3]    # default
Ea_unit = 1 [kJ/mol]    # default
  <PLOG>  0.1    1e10   0    100   </PLOG>    # pressure, A, beta, Ea : without units
  <PLOG>  1      1e12   0    110   </PLOG>    # sort by ascending pressure
  <PLOG> 10      1e13   0    120   </PLOG>
  <PLOG> 10     -1e4    0     20   </PLOG>    # duplicate a pressure for approximation by sum of two Arrhenius terms
</PLOG-Reaction>
```

In contrast to definitions of other reaction types, the individual Arrhenius terms are given in a tabulated way without units. The default units for this table can be specified before the first **<PLOG>** entry. If the current pressure of the gas-phase falls between two listed values (in ascending order), equation 2.71 is applied to calculate the interpolated rate of reaction. No exrapolation is applied if the pressure is smaller than the first value or larger than the last value. In these cases the rate is calualated according to the rate law of the lowest or highest pressure, respectively. Thus, it is also allowed to enter just a single pressure in the list. Then this rate expression is always applied at any pressure (which is the same as writing this reaction as a standard Arrhenius reaction).

Furthermore, for compatibility with published mechanisms, it is also allowed to approximate a reaction rate by the sum of two Arrhenius (but no more than two) terms. If a pressure p_i is listed twice, then the rate k_i at this pressure is the sum of two Arrhenius terms.

$$k_i = A_{i,1} * T^{\beta_{i,1}} * \exp\left(\frac{E_{\mathrm{ai},1}}{RT}\right) + A_{i,2} * T^{\beta_{i,2}} * \exp\left(\frac{E_{\mathrm{ai},2}}{RT}\right) \tag{3.1}$$

The summed term is then used in the interpolation. Therefore, one of the Arrhenius terms may even yield a negative result. However, the user must take care that the summed rate is positive, otherwise erroneous results (NaN - not a number) will be produced because of undefined logarithm.

<STICK> – surface adsorption reactions

Adsorption reactions are often described in terms of a sticking probability. Such reactions are declared as in the following example:

```
<STICK>
   CO + (s) = CO(s)
   S0 = 0.9
</STICK>
```

The reaction equation is written in a natural way. It is only required to separate species names and operators by at least one blank character. Each reaction can contain up to 3 reactants and up to 3 products. However, stoichiometric coefficients cannot be applied (see also input for user-defined reactions further below). Use a + sign to separate two reactants or two products. The > and = symbols are used for a non-reversible and reversible reactions respectively. A stick reaction must contain exactly one gas-phase species on the left-hand side of the reaction equation.

As a reaction parameter, only the sticking coefficient S_0 is required. Additional parameters are summarized below:

S0=*d*	sticking coefficient
beta=*d*	temperature exponent (default: 0)
Ea=*p*	activation energy with explicit unit (default: J/mol)
Ea/J_mol=*d*	activation energy [J/mol]
Ea/kJ_mol=*d*	activation energy [kJ/mol]
Ea/R=*d*	activation energy / R [K]
cflag=*s*	character flag
<**COV**>	coverage dependency

The rate expression can further be modified by coverage dependent factors. Such additional factors are defined using the **<COV>** tag, the details of which are given later in this chapter.

The character flag is only used for special applications like sensitivity analysis. It will be described in detail where necessary.

<COV> – coverage dependend factors

The rate expression of surface reactions can be modified by coverage dependent factors of the form

$$f(\theta_i) = \theta_i^{\mu_i} \mathrm{e}\dot{\exp}\left(\frac{\epsilon_i}{RT}\right) \tag{3.2}$$

The parameters μ_i and ϵ_i for a surface species i can be defined using the **<COV>** tag within a definition of a surface reaction.

```
<REACTION>
   O(s) + O(s) > (s) + (s) + O2
   A/cm_units = 3.7e21
   Ea/kJ_mol  = 213
   <COV> O(s) epsilon=70 [kJ/mol] </COV>
</REACTION>
```

The **<COV>** tag first lists the name of the surface species that appears as the argument of the modifying function. Then the parameters μ_i and ϵ_i can be defined. If one of them is omitted, its value defaults to zero. For the change of activation energy varaious units may be used. All possible options are summarized in the table below.

(species name)	name of surface species
epsilon=p	change of activation energy with explicit unit (default: J/mol)
epsilon/J_mol=d	change of activation energy [J/mol]
epsilon/kJ_mol=d	change of activation energy [kJ/mol]
epsilon/R=d	change of activation energy / R [K]
mu=d	change of reaction order

<UDF> – User-defined rate laws

Sometimes, published reaction rates do not fit into the scheme of predefined rate laws. Especially in case of lumped global reactions, complex algebraic expressions for reaction rates shall be applied. in DETCHEM version 2.8 you can define user-defined formulae. This is done in two steps.

First, every reaction mechanism can contain a calculation worksheet, where algebraic expression are defined, e. g.

```
<MECHANISM>
  <GASPHASE>
   <UDF>
        k1 = 1e12*exp(-25000/R/T)
        r1 = k1*c_CO*sqrt(max(c_O2,0))
...
     </UDF>
 ...
  </GASPHASE>
</MECHANISM>
...
```

The formula interpreter is invoked by the tag name <**UDF**> (user defined functions). Each line must contain an assignment to a new variable name (case sensitive, i. e. **a** and **A** are different variables). Predefined variables are

- **T** for temperature (in K),
- **p** for pressure (in Pa, note: ideal gas law is assumed),
- **R** for the gas constant,
- **c_**+*species name* for concentrations (in SI units),
- **X_**+*species name* for mole fractions (gas-phase species),
- **Y_**+*species name* for mass fractions (gas-phase species), and
- **theta_**+*species name* for coverages (surface species).

That means, the variable **c_CO** stands for the gas-phase concentration of species CO. Note that species names to be used here must contain only characters (a–z, A–Z), numbers (0–9) and the underscore character (_). Therefore a better name for a surface species for an adsorbed CO is **CO_s** instead of **CO(s)**. Constant numbers can be written like in most programming languages (e. g. 3, 0.1234, 1.e-12, etc.).

The right hand side of the assingment must be an algebraic expression combining variables and constants with operators **+** (addition), **-** (subtraction), ***** (multiplication), **/** (division) or **^** (power). Use round brackets to indicate operator precedence. You can also use the comparison operators **>, <, <=, >=** that evaluate to 0 for false or 1 for true. Using these and the multiplication operator, you can implement conditional expressions. The second factor is only evaluated if the first one is not zero, e. g.

```
logF = (F>0)*log(F)
```

The following functions can be used in arithmetic expressions:

- **abs(x)** = $|x|$
- **sqrt(x)** = $\sqrt{x}$
- **exp(x)** = $\exp(x)$

- **log(x)** = $\ln(x)$ logarithm with base e
- **log10(x)** = $\log_{10}(x)$
- **sin(x)** = $\sin(x)$
- **cos(x)** = $\cos(x)$
- **tan(x)** = $\tan(x)$
- **asin(x)** = $\sin^{-1}(x)$
- **acos(x)** = $\cos^{-1}(x)$
- **atan(x)** = $\tan^{-1}(x)$
- **min(x,y)** minimum of two values
- **max(x,y)** maximum of two values
- **print(x)** print value on screen and return x

All user-defined term of a mechanism need to be declared in one **<UDF>** block. However, each mechanism (gas phase, surface, etc.) can contain its own user-defined function block. The variable names are only valid for the current mechanism. That means that both, a gas-phase and a surface mechanism, can contain for instance a variable **k1** with different values.

In the second step, the individual reactions in the mechanism are declared as **<UDF-Reaction>**.

```
<MECHANISM>
  <GASPHASE>
   <UDF>
        k1 = 1e12*exp(-25000/R/T)
        r1 = k1*c_CO*sqrt(max(c_O2,0))
...
     </UDF>
 ...
     <UDF-Reaction>
       2 CO + O2 > 2 CO2
       variable = r1
     </UDF-Reaction>
  </GASPHASE>
</MECHANISM>
...
```

As for Arrhenius type reactions, write the reaction equation in a natural way. Stoichiometric coefficients, species names and operators must be separated by at least one blank character. Currently only integer stoichiometric coefficients are supported. The reaction rate is given by the keyword **variable** and a variable name from the user-defined function block. This is the rate of reaction in SI units (mol/(m^3 s) for homogeneous reactions or mol/(m^2 s) for surface reactions) with respect to the given stoichiometric coefficients. You get the rate of each species by multiplication of the rate of reaction with their respective stoichiometric coefficient. Reverse rates are not computed automatically for user-defined reactions. However, you can declare a reversible reaction and compute a rate that includes the reverse reaction, i. e. if the resulting rate is negative then the reaction is running in reverse direction.

As an example, an implementation of the catalytic converter mechanism from Oh and Cavendish (AIChE Journal, Vol. 26, No. 6, November 1980, 935) is given.

```
<MECHANISM>
  <SURFACE name=Pt>
    <UDF>
      cCO = c_CO*1e-6      # convert to mol/cm3
      cC3H6 = c_C3H6*1e-6
      cO2 = c_O2*1e-6
      kCO = 6.802e16 * exp(-13108/T)
      kC3H6 = 1.416e18 * exp(-15109/T)
      KCO = 8.099e6 * exp(409/T)
      Kc3H6 = 2.579e8 * exp(-191/T)
    inhibit = (1 + KCO*cCO + KC3H6*cC3H6)^2
    RCO = kCO*cCO*cO2 / inhibit        # mol(CO)/cm^2(Pt)/s
    RC3H6 = kC3H6*cC3H6*cO2 / inhibit  # mol(C3H6)/cm^2(Pt)/s
    r1 = RCO/2*1e4      # convert to integer stoichiometric coeff. and to mol/m^2/s
    r2 = RC3H6/2*1e4
    </UDF>

    <UDF-Reaction>
       2 CO + O2 > 2 CO2
```

```
      variable = r1
    </UDF-Reaction>
    <UDF-Reaction>
      2 C3H6 + 9 O2 > 6 CO2 + 6 H2O
      variable = r2
    </UDF-Reaction>
  </SURFACE>
</MECHANISM>
```

obsolete: <GLOBAL> – complex (global) reactions

This reaction type can now be completely replaced by the standard reaction types or by user-defined reactions. They are still supported for compatibility with older DETCHEM versions.

Sometimes, more complex reactions that are no elementary reactions need to be defined. Due to the strict format limitations of the gas-phase and surface mechanism files, they cannot be declared using the external mechanism files. Global reactions were introduced in DETCHEM version 2.0 and are to be declared in the the input file directly. An example of a global gas-phase reaction is shown below:

```
<MECHANISM>
  <GASPHASE>
    <GLOBAL>

      2 CO + O2 > 2 CO2

      <ARRHENIUS>
        CO  1.1
        A/cm_units = 1E10
        beta = 0
        Ea/kJ_mol = 275
      </ARRHENIUS>

    </GLOBAL>
  </GASPHASE>
</MECHANISM>
```

This input defines a reaction

$$2\,\mathrm{CO} \;+\; \mathrm{O_2} \;\longrightarrow\; 2\,\mathrm{CO_2} \tag{3.3}$$

with the rate law

$$\frac{\mathrm{d[O_2]}}{\mathrm{d}t} = -10^{10}\mathrm{cm^3/mol\,s}\,\exp\left(-\frac{275\mathrm{kJ/mol}}{RT}\right)\,[\mathrm{CO}]^{1.1}\,[\mathrm{O_2}] \tag{3.4}$$

In the global section, the user can specify any chemical reaction. Just write the reaction equation in a natural way:

component1 + component2 + ...> component3 + component4 + ...

for forward reactions or

component1 + component2 + ...= component3 + component4 + ...

for reversible reactions. Each component consits of an optional stoichiometric coefficient and a species name. The stoichiometric coefficient must be integer, if omitted it defaults to 1. Stoichiometric coefficients, species names and **+** signs must be separated by space characters.

The rate law can be defined in two ways. First, the user can specify the coefficients of an Arrhenius rate law. The possible parameters in the **<ARRHENIUS>** subsection are

A/SIunits=*d*	pre-exponential factor [mol, m, s, K]
A/cm_units=*d*	pre-exponential factor [mol, cm, s, K]
A/molefracs=*d*	pre-expontential factor w.r.t. mole fractions [mol/m3 s, K]
beta=*d*	temperature exponent (default: 0)
Ea/J_mol=*d*	activation energy [J/mol]
Ea/kJ_mol=*d*	activation energy [kJ/mol]
Ea/R=*d*	activation energy / R [K]
species symbol=*d*	reaction order of species

So, the user can specify the preexponential factor A, the temperature exponent β, and/or the activation energy E_A. The reaction order of a species defaults to the reactant's stoichiometric coefficient. If stoichiometric coefficient and reaction order of a species shall be different, then specify the species name and its reaction order within the **<ARRHENIUS>** subsection.

Another way to define a rate law of a global reaction is a user defined rate law. This can be declared in the following way:

```
<MECHANISM>
  <GASPHASE>
    <GLOBAL>

      2 NO + O2 > 2 NO2
      userID = 1

    </GLOBAL>
  </GASPHASE>
</MECHANISM>
```

Using **userID**, an integer number is assinged to the global reaction, which identifies the corresponding rate in the source code. Then the user needs to provide FORTRAN code for the rate law. The interface routines are contained in the DETCHEM distribution in file ***lib_detchem/DCRGU_UserSub.f*** for gas-phase mechanisms and ***lib_detchem/DCRSU_UserSub.f*** for surface mechanisms. Before running the application, the code must be recompiled with your user program code (obsolete).

obsolete: Inhibition factors

Instead of separate definition of inihibition factors, it is recommended to use user-defined rate laws instead. The description here is for compatibility with older DETCHEM versions.

The rate laws of global net reactions can be quite complex. However for technical systems like in the automotive exhaust-gas aftertreatment, a detailed reaction mechanism is either not available or not feasible for a fast solution. In such cases, experimental data is used to fit parameters that cover the desired range of conditions. Often, a simple Arrhenius law is not satisfying enough. Especially in cases where washcoat effects are directly taken into account in the rate expression, so-called inhibition factors are frequently used. The type of inhibition factors that can be evaluated within a global reaction definition is a follows:

$$I = \frac{1}{T^{\beta} \cdot \left(1 + K_1 X_1^{a_1} X_2^{a_2} + K_2 X_3^{a_3} X_4^{a_4} + \ldots\right)^{b_1} \cdot \left(1 + K_3 X_5^{a_5} X_6^{a_6} + \ldots\right)^{b_2} \cdot \ldots} \tag{3.5}$$

with

$$K_i = A_i \cdot \exp\left(-\frac{E_i}{RT}\right) \tag{3.6}$$

The inhibition factors contain a temperature dependent factor and a product of composition dependencies. Each of those factors consists of a sum of terms that depend on the mole fractions of one or two species. The factors themselves can be raised to an arbitrary power. Furthermore, the K_i are Arrhenius type of rate factors, with and pre-expontial factor A_i and an activation energy E_i. Therefore, several parameters can be defined: an temperature exponent β, mole fraction exponents a_i, factor exponents b_i, and Arrhenius parameters A_i and E_i

The definition of an inhibition factor will be illustrated by an example. Suppose an inhibition factor

$$I_1 = \frac{1}{T^{1.05}\left(1 + 0.3 \cdot X_{\mathrm{CO}}^{0.75} X_{\mathrm{C_3H_6}}^{0.5}\right)\left(1 + 5\exp\left(-\frac{5000\mathrm{K}}{T}\right) \cdot X_{\mathrm{NO}}\right)^2} \tag{3.7}$$

To define this inhibition factor insert the following text in the **<MECHANISM>** section of the input file (it must be directly in the hierarchy level of the **<MECHANISM>** tag, **not** inside the **<GASPHASE>** or **<SURFACE>** tags)

```
<INHIBITION-FACTOR>
  ID=1
  T-Exponent=1.05
  <FACTOR>
    <SUMMAND>
      A=0.3
      CO ** 0.75
      C3H6 ** 0.5
    </SUMMAND>
  </FACTOR>
  <FACTOR>
    <SUMMAND>
      A=5
      Ea/R = 5000
      NO
    </SUMMAND>
    exponent=2
  </FACTOR>
</INHIBITION-FACTOR>
```

The complete list of options of the **<INHIBITION-FACTOR>** tag is given in the following table:

ID=*i*	ID of inhibition factor
T-Exponent=*d*	temperature exponent (default: 1)
<**Factor**>	factor of the form (1+Ki*Xi+...)

with following options in the **<FACTOR>** tag

exponent=*d*	exponent of the factor (default: 1)
<**Summand**>	summand of the form K * X1**n1 * X2**n2

and in the **<SUMMAND>** tag

A=*d*	preexponential factor (default: 1)
Ea/J_mol=*d*	activation energy [J/mol] (default: 0)
Ea/kJ_mol=*d*	activation energy [kJ/mol] (default: 0)
Ea/R=*d*	activation energy / R [K] (default: 0)
(species name)	species name
****=*d*	species exponent

Each inhibition factor is identified by an ID number, which must be an integer number (1,2,...). The inhibition factor can then be used to modify the Arrhenius law of a global reaction. This is done by the optional parameter **INHIBITION-FACTOR** within the **<GLOBAL>** tags as shown in the example below.

```
<GLOBAL>

  2 CO + O2 > 2 CO2

  <ARRHENIUS>
     A/cm_units = 1E10
     Ea/kJ_mol = 275
  </ARRHENIUS>
  INHIBITION-FACTOR=1

</GLOBAL>
```

3.8 Surface models

Note for DETCHEM Version 2.0 users: Because of some ambiguities between washcoat input and **<CHEMSURF>** input the data has been rearranged. For any simulation involving surface species the new section **<SURFACE-MODEL>** has to be defined. The information must be given between the **<MECHANISM>** section and the beginning of the application specific section. Now, as a consequence, all DETCHEM applications can apply at least the washcoat effectivness factor model.

The **<SURFACE-MODEL>** section contains information for the calculation of the net surface fluxes due to chemical reactions. Three properties influence the solution method:

- The steady-state solver **<CHEMSURF>**.
- The ratio of catalytic to geometric surface area.
- The washcoat diffusion model.

A simple example is shown below:

```
<SURFACE-MODEL>

  <CHEMSURF>
     time=1.         # integration time for surface coverages
     analytical=no   # use analytical Jacobian matrix
  </CHEMSURF>

  Fcatgeo=1.         #ratio of catalytic to geometric area

  # mo washcoat diffusion model defined
</SURFACE-MODEL>
```

The possible options inside the **<SURFACE-MODEL>** tags are listed in the following table:

<**CHEMSURF**>	CHEMSURF parameters
Fcatgeo=*d*	ratio catalytic / geometric surface (default: 1)
<**Fcatgeo**>	ratio catalytic /geometric surface (individual values for each surface type)
Ageo=*d*	geometric surface area (without washcoat)
<**active_moles**>	moles of active catalyst
Eff_Model=*s*	species for effictiveness factor model (skip this option for detailed washcoat model)
<**EFF_MODEL**>	list of species for effictiveness factor model (skip this option for detailed washcoat model)
<**MOLECULAR_DIFF**>	molecular diffusion model
<**KNUDSEN_DIFF**>	Knudsen diffusion model
<**MIXED_DIFF**>	molecular diffusion model
<**RANDOM_PORE**>	molecular diffusion model
<**OUTS**>	modified species list in OUTS file

3.8.1 <CHEMSURF> input

The DETCHEM library provides routines to calculate surface reaction source terms. For most DETCHEM applications steady-state conditions of the surface conditions are of interest. For this purpose, the routine **CHEMSURF** is called. **CHEMSURF** integrates the surface coverages under the constraints of a given gas-phase composition and temperature for a given time. It is assumed that this integration time is sufficient to reach steady state.

In many cases it is sufficient to specify an integration time only. To do so, use the **time** member. The integration time should be long enough in order to equilibrate the system. Use shorter times only in cases when transient effects of surface coverages are desired in your simulation (e. g. long term storage effects). On the other hand, some ill-conditioned surface reaction mechanisms may result in poor convergence behavior. In this case, long integration times may slow down the simulation.

Using the member **analytical** the user can choose if the Jacobian of the ODE system shall be determined analytically or numerically. In general, the analytical solution should be faster, but in some cases the numerical solution is more stable.

In case the ChemSurf integration process shall be monitored, the option **write_file** can be activated. Then, a file *chemsurf.plt* will be written upon the first call of ChemSurf during execution of any DETCHEM application, e. g. DETCHEM$^{\text{SURFPROBE}}$. This file contains the surface concentrations of all non-inert surface species vs. time. Note: When using the detailed reaction-diffusion washcoat model, this option will be ignored, because a different solver is called.

All options in the **<CHEMSURF>** section are summarized below.

time=*d*	integration time [s] (default: 1)
hini=*d*	initial integration time step [s] (default: 1e-8)
rTol=*d*	relative tolerance of surface coverages (default: 1e-4)
aTol=*d*	absoulute tolerance of surface coverages (default: 1e-20)
maxIter=*i*	maximum number of iterations (default: unlimited)
analytical=*b*	analytical (yes) or numerical (no) calculation of Jacobian (default: no)
write_file=*b*	write file *chemsurf.plt* during first ChemSurf call (default: no)

3.8.2 Catalytic vs. geometric surface area

The rates of reaction (in moles per area) are always defined for an ideal flat surface. Unless the surface is covered with a thin foil of the catalytic material, the boundary between the reactor volume and the surface is rough. Often, washcoats are used to increase the roughness and therefore to increase the surface area. To account for this effect DETCHEM uses a factor $F_{\text{cat/geo}}$ that gives the ratio of the catalytic and the geometric surface area.

The catalytic area of the reactor can be determined experimentally, e. g. by temperature programmed desoprtion experiments. If the number of active moles of catalyst is known, then this can be converted into an area using the surface site density Γ defined in the **<SPECIES>** section. The geometric area is the surface area of the reactor with a flat surface, i. e. in case of a cylindrical channel its equals its circumference times the channel length.

There are three ways defining the ratio:

- In case there is only one type of surface, it is sufficient to define a constant value.

```
Fcatgeo = 5.    # increase catalytic surface area by factor 5
```

- If you have specified more than one surface type in the **<SPECIES>** section of your input file, you can assign an individual ratio of catalytic area to geometric area for each of these surfce types.

```
<Fcatgeo>       # surface type dependend Fcat/geo
   Pt  0.75
   Rh  0.25
</Fcatgeo>
```

- Alternatively, the conversion of active moles to $F_{\mathrm{cat/geo}}$ can be done automatically.

```
<active_moles>
   Pt  0.002
   Rh  0.0005
</active_moles>
Ageo = 3.14e-4   # geometric surface area [m2]
```

3.8.3 Washcoat diffusion models

Washcoats are used to enlarge the catalytic surface of a reactor. However, because of their porous structure, diffusion limitations can become significant. Therefore, additional information about the geometric properties of the washcoat is required. An example of washcoat input is shown below.

```
<RANDOM_PORE>
 <ZONE>
  thickness=1.d-04    # thickness of the washcoat layer
  ngrid=10            # number of grid cells
  aspect=1            # aspect ratio of neighboring grid cells
 </ZONE>
 tau=3                # tortuosity

 <MACRO>
   porosity=0.15      # porosity
   diameter=1d-6      # pore diameter
 </MACRO>

 <MICRO>
   porosity=0.35
   diameter=1d-9
 </MICRO>

 <SOLVER>
   time=10            # solver integration time
   hini=1.d-4         # initial integration step size
 </SOLVER>

</RANDOM_PORE>
```

The fours diffusion models described in chapter 2.3.4 can be accessed by the following tags:

- **<MOLECULAR_DIFF>**
- **<KNUDSEN_DIFF>**
- **<MIXED_DIFF>**
- **<RANDOM_PORE>**

Including one of the fours tags in the **<SURFACE-MODEL>** section activates the (detailed) washcoat model. Note: In case the washcoat model shall be easily activated and deactivated, it is reccommended to put the washcoat definition into a separate file, e.ġ. ***washcoat.inp***, and activate or deactivate the include statement.

```
<SURFACE-MODEL>
  ...
  {include washcoat1.inp}    # activated washcoat diffusion model
# {include washcoat2.inp}    # deactivated washcoat diffusion model
</SURFACE-MODEL>
```

The options for each model are summarized below

<**ZONE**>	1-dimensional washcoat grid zone
porosity=*d*	porosity of washcoat
tau=*d*	tortuosity of washcoat
diameter=*d*	mean pore diameter [m]
<**MACRO**>	macro pore parameters (random model)
<**MICRO**>	micro pore parameters (random model)
relative_load=*d*	relative load of this zone
zone2:	start a second zone with different parameters
<**SOLVER**>	DAE solver parameters

The washcoat model basically solves a reaction diffusion equation. The tags **<MACRO>** and **<MICRO>** stand for macro pores and micro pores which are required for the random pore model. Otherwise, specify **porosity** and **diameter** for all cells once without these tags.

The information under the **<ZONE>** tag is used for the discretization of the governing equation 2.78. There can be any number of **<ZONE>** tags, however these zones will all have the same properties (except for the grid spacing). If the washcoat properties (porosity, pore diameter or loading) should vary along the grid, the zone definition can be devided into two layers by using the keyword **zone2:** as shown in the following example

```
<SURFACE-MODEL>
  <MIXED_DIFF>
    <ZONE>
      ngrid=10
      thickness=1.d-4
    </ZONE>
    porosity=.5
    tau=3
    diameter=1.-6
    relative_load= 1
  zone2:
    <ZONE>
      ngrid=10
      thickness=1.d-4
    </ZONE>
    porosity=.5
    tau=3
    diameter=1.-7
    relative_load= 3

    <SOLVER> time=10 </SOLVER>
  </MIXED_DIFF>
  ...
</SURFACE-MODEL>
```

In this example, the two zones are both .1 mm thick, but they differ in their pore diameters and their catalyst loadings. First, the fluid encouters a layer with large pores. Below there is a layer with smaller pores, but the specific catalytic surface area is three times larger. Use the parameter **relative_load** in order to define a ratio of the catalytic surface areas of the two layers (only the ratio of the two values is important). The actual amount of catalyst in the layers also depends on their thickness and the total catalytic surface area defined in **<Fcatgeo>**. Unfortunately it is not possible to define layers with different $F_{\mathrm{cat/geo}}$, i.e. the composition of the washcoat must not change. Otherwise the effectiveness factor model would not be applicable. Therefore it is not possible so far to define a system with two layers of different coatings, e.g. one layer platinum and one layer rhodium. However, by assigning **relative_load**=0 in one layer, an inactive layer can be created.

Options in **<ZONE>**:

ngrid=*i*	number of grid points
aspect=*d*	aspect ratio of adjacent grid cells (default: 1 ,i.e. equal spacing)
thickness=*d*	thickness of washcoat layer [m]

Options in **<MACRO>** and **<MICRO>**:

porosity=*d*	porosity of washcoat
diameter=*d*	pore diameter [m]

Options in **<SOLVER>**:

time=*d*	integration time [s] (default: 1)
hini=*d*	initial integration step size (default: 1e-8)
rTol=*d*	relative tolerances of concentrations and coverages (default: 1e-4)
aTol=*d*	absolute tolerances of concentrations and coverages (default: 1e-20)
maxiter=*i*	maximum number of iterations (default: no limit)

3.8.4 Effectiveness factor model

The detailed washcoat diffusion model is very time consuming and may not be available in some application (e. g. MONOLITH with transient coverages of a storage medium). In order to account for diffusion limitations of washcoats, an effectiveness factor approch can be applied.

In this model, the diffusion of one species is assumed to account for the diffusion limitions. Assign the name of one species to the **Eff_Model** or list some under **<EFF_MODEL>** in the **<SURFACE-MODEL>** section in addition to the washcaot definition.

Then no reaction-diffusion equation will be solved. Instead, the surface reaction rate will be calclulated the same way as without the washcoat diffusion model. Using the total rate for the chosen species, an effictiveness factor is calculated. Finally, all rates are multiplied by this effectiveness factor.

In case more than one species is listed, the effectiveness factor of the species that gives the strongest limitation will be selected automatically.

An example definition of an effectiveness factor model is shown below:

```
<SURFACE-MODEL>
  ...
  <MIXED_DIFF>
     <ZONE>
       thickness=1.d-04    # thickness of the washcoat layer
       ngrid=10            # dummy value (no grid generated)
     </ZONE>
     tau=3                 # tortuosity
     porosity=0.5          # porosity
     diameter=1d-9         # pore diameter

     <SOLVER>
       time=1              # dummy value (no separate integration)
     </SOLVER>

   </MIXED_DIFF>

   EFF_MODEL = CO          # CO is the diffusion limiting species

#  alternatively:
#  <EFF_MODEL>
#    CO   C3H6             # effectiveness factor for CO or C3H6 is applied
                           # (whichever is smaller)
#    threshold = 1e-10     # minimum mole fraction for Thiele modulus
#  </EFF_MODEL>

</SURFACE-MODEL>
```

Chapter 4

DETCHEM$^{\text{CHEMINP}}$

4.1 Introduction

DETCHEM$^{\text{CHEMINP}}$ is a pre-processor and translator for DETCHEM species and mechanism information files. It creates a compact species and mechanism input file ***chem.new*** as well as input files in CHEMKIN format.

4.2 User Input

DETCHEM$^{\text{CHEMINP}}$ requires one input file named ***cheminp.inp***. This file acts as a container for the species and mechanism information. If this information is stored in external files, e. g. ***species.inp*** and ***mech.inp***, then ***cheminp.inp*** may simply look like the following example.

```
<CHEMINP>
  {include species.inp}
  {include mech.inp}
</CHEMINP>
```

Like in all other DETCHEM applications, the files ***thermdata*** and ***moldata*** need to be present. For more information on these files please refer to chapter 3.

4.3 Output

DETCHEM$^{\text{CHEMINP}}$ produces several output files.

The file ***chem.new*** contains the species and the mechanism information in a compact form. It can be used to replace the species and the mechanism information in any detchem apllication. Just write

```
{include chem.new}
```

at the beginning of the application specific input file.

The files ***thermdata.new*** and ***moldata.new*** contain the thermodynamic data and the molecular properties (respectively) of the required species. That is, only the information of the relavant species is extracted from ***thermdata*** and ***moldata***. If you want to use the new files with your application, please remove ***.new*** from the file names.

Furthermore, two files named ***gasmech.chemkin*** and ***surfmech.chemkin*** are created. These files contain the gas-phase and surface mechanisms in CHEMKIN format. Thus, DETCHEM$^{\text{CHEMINP}}$ can be used as a DETCHEM to CHEMKIN translator for input files.

4.4 Isotope analysis

An experimental method to examine reaction paths is the isotope analysis. Reactants could be marked using different isotopes in order to find out which product is formed from which reactant. This set-up can be simulated in any DETCHEM application. However, one needs to create new input files, because isotopes need to be treated as different species. Such new input files can be created using DETCHEM$^{\text{CHEMINP}}$.

This is done by adding a line to ***cheminp.inp*** using the option **`isotopes`** followed by the symbol of a chemical element. If you want to distinguish two oxigen isotopes, then use an input like the following:

```
<CHEMINP>
  {include species.inp}
  {include mech.inp}
  isotopes O
</CHEMINP>
```

After running DETCHEM^{CHEMINP} the files ***chem.new***, ***thermdata.new*** and ***moldata.new*** will contain species new species and reactions where the number of atoms that have been replaced by generic isotopes go through all possible combinations. (Note: It is recommended to use only species that contain no more than two atoms of the desired isotopes. The rate laws of complex reactions can only be translated if the desired atom does not appear more than once.) A new atom type, e. g. **0***, will be created. For each species containing a replaced atom, a generic species name will be created starting with an integer number that indicates the number of replace atoms. Therefore, the species **02** will be replaced by three species **02**, **102**, **202**.

The files ***gasmech.chemkin*** and ***surfmech.chemkin*** are not influenced by the isotope variation.

In order to use the isotope analysis in any application, copy the files ***chem.new***, ***thermdata.new*** and ***moldata.new*** to your working directory. Remove ***.new*** from ***thermdata.new*** and ***moldata.new***. Include ***chem.new*** at top of your application specific input file. Then, for example, you want to find out whether the oxygen in your product H_2O originates from O_2 or CO_2. In this case you may use the species **02** (unmarked) and **2C02** (marked) in your list of reactants. By looking at the product composition (**H20** and/or **1H20**), the reaction path can be determined.

4.5 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***./go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 5

DETCHEM$^{\text{GASPROBE}}$

5.1 Introduction

DETCHEM$^{\text{GASPROBE}}$ is basically a tool to probe the gas-phase chemistry, thermodynamic and transport properties. Whenever a new gas-phase mechanism is to be used with the DETCHEM reactive flow models, it is recommended to test the mechanism using DETCHEM$^{\text{GASPROBE}}$

5.2 User Input

Before running DETCHEM$^{\text{GASPROBE}}$ the user must prepare the input file ***gasprobe.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<GASPROBE>
  T/K = 1200
  p/Pa= 100000
  <MOLEFRAC>
     NO2    0.1
     N2     *
  </MOLEFRAC>
</GASPROBE>
```

The file ***gasprobe.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

The options within the **<GASPROBE>** tag are as follows.

T/K=*d*	temperature [K]
p/Pa=*d*	pressure [Pa]
<**MOLEFRAC**>	species mole fractions
<**MASSFRAC**>	species mass fractions

In the **<MASSFRAC>** or **<MOLEFRAC>** section, one species can be assigned a ***** instead of a numerical value. The mass or mole fraction of this species will be balanced to unity, respectively.

5.3 Output

DETCHEM$^{\text{GASPROBE}}$ produces only screen output. The output contains information on operating conditions, mass and mole fractions, chemical source terms, heat capacity, enthalpy and entropy.

```
*************************************************************************
****                        DETCHEM GASPROBE                         ****
****                   S. Tischer, O. Deutschmann                    ****
****                  VERSION 2.9.1 ** 2022-12-15                    ****
*************************************************************************

 initial values
 Temperature [K]:    1200.0000000000000
 Pressure   [Pa]:    100000.00000000000
 species      mass fraction       mole fraction
```

```
NO              0.00000000          0.00000000
O2              0.00000000          0.00000000
NO2             0.15431558          0.10000000
O               0.00000000          0.00000000
N2              0.84568442          0.90000000

 species      chemical source term [mol/(m^3 s)]
NO                5.684270E+01
O2                2.839643E+01
NO2              -5.684270E+01
O                 4.984329E-02
N2                0.000000E+00

 species     heat capacity [J/kg K]   enthalpy [J/kg]   entropy [J/kg K]
NO             1.16144502E+03         3.98033058E+06         8.48984838E+03
O2             1.11232123E+03         9.30481633E+05         7.81081865E+03
NO2            1.16563315E+03         1.65345571E+06         6.59683859E+03
O              1.30515344E+03         1.67655879E+07         1.19061303E+04
N2             1.20326289E+03         1.00380352E+06         8.35843383E+03
 average       1.19745604E+03         1.10405497E+06         8.17725487E+03

 species    mix diff coeff [m^2/s] viscosity [kg/m s]   heat cond [J/m K s]
NO             2.15500398E-04         4.83687258E-05         7.94899428E-02
O2             2.20299385E-04         5.39859100E-05         8.48080255E-02
NO2            1.82631959E-04         5.71341364E-05         9.00908335E-02
O              3.39077003E-04         6.30648684E-05         1.22899213E-01
N2             2.99930563E-04         4.67350570E-05         7.97204189E-02
 average                              4.76882120E-05         8.07031163E-02


 DETCHEM -- copyright by O. Deutschmann
 For further information visit www.detchem.com
```

5.4 Examples

There is an example supplied with the installation. The example directory contains the following files ***gasprobe.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable DETCHEM_DIR that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

5.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

5.6 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 6

DETCHEM$^{\text{SURFPROBE}}$

6.1 Introduction

DETCHEM$^{\text{SURFPROBE}}$ is basically a tool to probe the surface chemistry. It calculates the surface coverages and fluxes at catalytic surfaces at constant ambient conditions. It is recommended to test a new mechanism using DETCHEM$^{\text{SURFPROBE}}$ before using it with DETCHEM reactive flow models. Most surface reactions are supposed to reach a steady state in few seconds or even less. DETCHEM$^{\text{SURFPROBE}}$ integrates the surface coverages until it reaches steady state.

6.2 User Input

Before running DETCHEM$^{\text{SURFPROBE}}$ the user must prepare the input file ***surfprobe.inp***.An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
    time=100
    analytical=no
    write_file=yes
  </CHEMSURF>

  Fcatgeo = 1

</SURFACE-MODEL>

<SURFPROBE>
  T/K    = 1200
  p/Pa    = 1.0d05

  <GASPHASE>
    <MOLEFRAC>
      NO2  0.1
      N2    0.9
    </MOLEFRAC>
  </GASPHASE>

</SURFPROBE>
```

The file ***surfprobe.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

The options within the **<SURFPROBE>** tag are as follows.

Fcat/geo=*d*	ratio catalytic / geometric surface area
T/K=*d*	temperature [K]
p/Pa=*d*	pressure [Pa]
<**GASPHASE**>	gas-phase composition
<**SURFACE**>	surface coverages (default: initial coverages from species definition)

In the **<MASSFRAC>** or **<MOLEFRAC>** subsection, one species can be assigned a ***** instead of a numerical value. The mass or mole fraction of this species will be balanced to unity, respectively.

6.3 Output

DETCHEM^{SURFPROBE} produces only screen output. The output contains information on operating conditions, mass and mole fractions, concentrations, chemical source terms, coverages and surface fluxes. It can easily be inferred from the output, whether the surface mechanism reaches a steady state in the specified integration time.

```
************************************************************************
****                      DETCHEM SURFPROBE                         ****
****    Vinod M Janardhanan, Steffen Tischer & Olaf Deutschmann     ****
****                  VERSION 2.9.1 ** 2022-12-15                   ****
************************************************************************

 Initial Values
 Time of integration (s) :    0.0000000000000000
 Temperature (K)         :    1200.0000000000000
 Pressure (Pa)           :    100000.00000000000
 Gas Species     Mass fraction       Mole fraction     concentration (mol/m**3)
 NO              0.00000000          0.00000000          0.00000000
 O2              0.00000000          0.00000000          0.00000000
 NO2             0.15431558          0.10000000          1.00226996
 O               0.00000000          0.00000000          0.00000000
 N2              0.84568442          0.90000000          9.02042962
 Initial Coverage
 PT(s)      :     1.0000000000000000
 O(s)       :     0.0000000000000000
 NO2(s)     :     0.0000000000000000
 NO(s)      :     0.0000000000000000

************************************************************************
****                            LIMEX                               ****
**** (C) Konrad-Zuse-Zentrum fuer Informationstechnik Berlin (ZIB)  ****
****                  version 4.2A1 of 2000/03/17                   ****
************************************************************************

 Coverage after    100.00000000000000      s:
 PT(s)     :    0.44510821735842770
 O(s)      :    0.55478603279999639
 NO2(s)    :     1.0550724460620868E-004
 NO(s)     :     1.3443609410418216E-007
 Fluxes at surface:    mole/(m**2 s)           kg/(m**2 s)
 NO                     0.44214221E+01          0.13266963E+00
 O2                     0.22107111E+01          0.70740102E-01
 NO2                   -0.44214221E+01         -0.20340974E+00
 O                      0.00000000E+00          0.00000000E+00
 N2                     0.00000000E+00          0.00000000E+00

 Net surface mass flux :    3.0531133177191805E-016   kg/(m**2 s)

 d(Coverage)/dt after    100.00000000000000      s:
 PT(s)                  -0.14367592E-08
 O(s)                    0.15347201E-08
 NO2(s)                 -0.91430131E-09
 NO(s)                   0.19592171E-09

 DETCHEM -- copyright by O. Deutschmann
 For further information visit www.detchem.com
```

6.4 Examples

There is one example supplied with the installation. The example directory contains the following files ***surfprobe.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, a surface mechanism ***noo2sm***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

6.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory which becomes your working directory and must contain all

the input files mentioned in the above section. The user can copy the files from the example directory to the working directory and make the necessary changes.

6.6 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 7

DETCHEM$^{\text{EQUIL}}$

7.1 Introduction

This code calculates the equilibrium composition of a given gas-phase mixture, which is defined by its thermodynamic potentials.

7.1.1 Theoretical background

The state of a closed system can be described by pressure p, temperature T, and the number of moles of each species n_i. This state can be expressed in terms of the Gibbs Free Enthapy as potential function $G(T, p, n_i)$. The system is in equilibrium, when G is a minimum with respect to all possible combinations of n_i.

In chemical systems, the mass is conserved, and hence the number of atoms must not change in a chemical reaction. If we denote the number of atoms of type j in species i by a_{ji}, then the changes δn_i of amount of species i must fulfil

$$\sum_i a_{ji}\,\delta n_i = 0 \quad \forall j. \tag{7.1}$$

If the system is described in terms of chemical reactions, we denote the stoichiometric coefficient of species i in reaction k by ν_{ik} and the reaction coordinate by ξ_k. Applying this, we can write

$$\delta n_i = \sum_k \nu_{ik}\xi_k \quad . \tag{7.2}$$

Inserting eq. 7.2 into eq. 7.1, we get

$$\sum_k \left(\sum_i a_{ji}\nu_{ik} \right) \xi_k = 0 \tag{7.3}$$

This must be fulfilled for arbitrary ξ_k. Therefore the inner brackets must vanish. The matrices $(a_{ji})^{\mathrm{T}}$ and (ν_{ik}) are othogonal with respect to each other. Hence, for any (ν_{ik}) with maximum rank any possible reaction can be written as linear combination of its columns. The columns of matrix (ν_{ik}) are the stoichiometric coefficients of hypothetical reactions.

Instead of solving the minimization problem with constraints (eq. 7.1), we can solve the minimization problem for the hypothetical reactions.

For a mixture of ideal gases, the Gibbs Free Enthalpy is given by the expression

$$G(T, p, n_i) = \sum_i n_i \left(g_i^0(T) + RT \ln \frac{p_i}{p^0} \right) \tag{7.4}$$

where g_i^0 denotes the standard specific molar free enthalpy at pressure p^0. p_i is the partial pressure according to the ideal gas law

$$p_i = \frac{n_i RT}{V} \quad . \tag{7.5}$$

Equation 7.4 can be generalized to other mixtures than ideal gases by introduction of an activity a_i that replaces the term $\frac{p_i}{p^0}$. Here, we also consider mixtures of ideal liquids, where a_i is the mole fraction of species i in the condensed phase.

Reaction k is in equilibrium when

$$\left(\frac{\partial G(T,p,n_i+\nu_{ik}\xi_k)}{\partial \xi_k}\right)_{T,p} = 0 \quad . \tag{7.6}$$

This is equivalent to the formualtion involving the equilibrium constant

$$K_p = \exp\left(\frac{-\Delta_R G^0}{RT}\right) = \prod_i (a_i)^{\nu_{ik}} \tag{7.7}$$

with

$$\Delta_R G^0 = \sum_i \nu_{ik} g_i^0 \quad . \tag{7.8}$$

7.1.2 Algorithm

Applying the mathematical model, we can outline the following algorithm:

1. Construct hypothetical reactions preferring those involving species with large amounts,
2. Equilibrate all reactions sequentially,
3. If amounts of all species have changed less than a given tolerance, then finish, else go to 1.

7.2 User input

DETCHEM^{EQUIL} requires an input file ***equil.inp***, thermodynamic data in file ***thermdata***, and the database for molecular properties ***moldata***. For convenience, parts of the input file ***equil.inp***, e. g. the species list, may be saved in a seperate file and included using an **`include`** statement.

7.2.1 *equil.inp*

For running DETCHEM^{EQUIL} once use an input file like this:

```
<EQUIL>
  <SPECIES>
    H2  O2  OH  H2O H O H2O2 CH4 CO2 CO N2
  </SPECIES>
  <CONDENSED>          # optional list of species in a condensed phase
    C  2260            # species name and its density in kg/m3
  </CONDENSED>

  T=1000               # temperature in K
  const p T            # constraints: constant pressure & temperature

  <MOLEFRAC>           # initial composition
    CH4 0.15
    O2  0.15
    N2  *
  </MOLEFRAC>

  atol=1.d-20          # absolute tolerances
  rtol=1.d-6           # relative tolerances
  Ttol=0.1             # temperature tolerance
</EQUIL>
```

Instead of a single temperature, the user can also specify a temperature range by replacing the **`T`** option with a definition in the following format:

```
  <T> from 500 to 1200 step 50 </T>
```

The table below lists all available options. Where applicable, the default values are given.

<**SPECIES**>	species list
<**CONDENSED**>	list of species in a condensed phase (optional)
T=*d*	temperature [K] (default: 298)
<**T**>	list of temperatures [K]
p=*d*	pressure [Pa] (default: 101325)
V=*d*	volume [m^3] (default: 1e-3)
<**MOLEFRAC**>	species mole fractions
<**MASSFRAC**>	species mass fractions
atol=*d*	absolute tolerance of concentrations [mol/m^3] (default: 1e-30)
rtol=*d*	realtive tolerance of concentrations (default: 1e-6)
Ttol=*d*	absolute tolerance of temperature [K] (default: 0.1)
const v_1 v_2	operation condition: choose 2 constant properties from p, V, T, S, H

Two constraints can be chosen from constant pressure **p**, volume **V**, temperature **T**, enthalpy **H** or entropy **S**. However, since enthalpy is a function of temperature only for ideal mixtures, constants **T** and **H** cannot be selected at the same time. The classical equilibrium problem is solved for **const T p**. If you want the adiabatic case choose **const H p**.

7.2.2 *thermdata*

For a detailed description of the ***thermdata*** file see chapter 3.2.

Here it shall only be mentioned that besides the thermodynamic coefficients the file ***thermdata*** also contains information about the atomic composition of each species. This information plays an important role when running DETCHEM$^{\text{EQUIL}}$. Make sure that the first line of each species contains the atomic composition in the correct format, e.g.

```
CH3O                  C   1H   3O   1    0G   300.000  5000.000             1
                      ^    ^    ^    ^
                      C    H3   O    none
```

The symbols are expected in colums 25, 30, 35, and 40. Each atom symbol can be up to two characters followed by an integer value in the next 3 columns.

7.3 Output

7.3.1 Screen output

After successfully executing DETCHEM$^{\text{EQUIL}}$ the screen will look like this:

```
*************************************************************************
****                         DETCHEM EQUIL                           ****
****                           S. Tischer                            ****
****                  VERSION 2.9.1 ** 2022-12-15                    ****
*************************************************************************
 species:
 H2        H   2
 O2        O   2
 OH        H   1 O   1
 H2O       H   2 O   1
 H         H   1
 O         O   1
 H2O2      H   2 O   2
 CH4       H   4 C   1
 CO2       O   2 C   1
 CO        O   1 C   1
 N2        N   2
 C         C   1
 initial conditions:
 p =   101325.00000000000       Pa
 V =   1.0000000000000000E-003  m^3
 Vg=   1.0000000000000000E-003  m^3
 Vc=   0.0000000000000000       m^3
 T =   1000.0000000000000       K
 S =   2.9259917950880610       J/K
 H =   157.58217438080524       J
 G =  -2768.4096207072557       J
gas-phase:
 species          moles      mole frac
H2         0.000000E+00   0.000000E+00
O2         1.827990E-03   1.500000E-01
OH         0.000000E+00   0.000000E+00
H2O        0.000000E+00   0.000000E+00
H          0.000000E+00   0.000000E+00
```

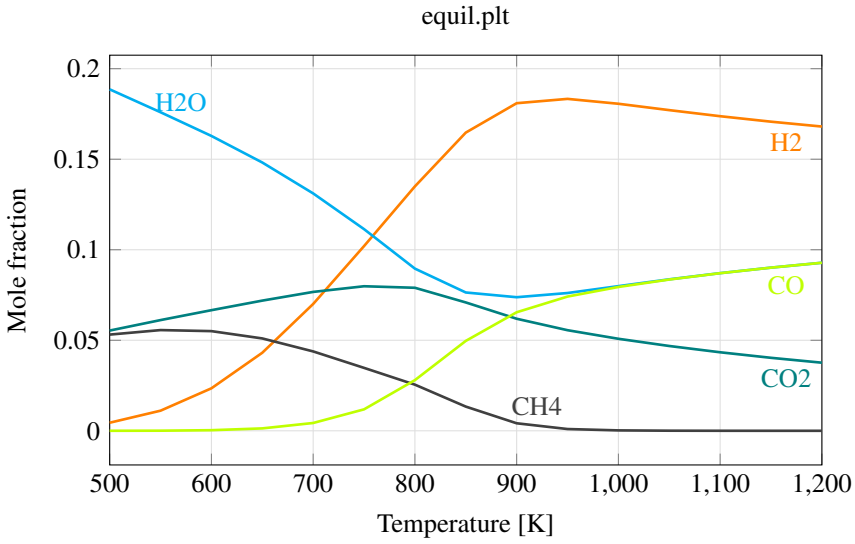
```
O          0.000000E+00   0.000000E+00
H2O2       0.000000E+00   0.000000E+00
CH4        1.827990E-03   1.500000E-01
CO2        0.000000E+00   0.000000E+00
CO         0.000000E+00   0.000000E+00
N2         8.530620E-03   7.000000E-01
condensed phase:
C          0.000000E+00            NaN
 hypothetical reactions:
           2  H2O      >            2  H2               1  O2
           2  H2O      >            1  H2               2  OH
           1  H2       >            2  H
           1  H2O      >            1  H2               1  O
           2  H2O      >            1  H2               1  H2O2
           3  H2               1  CO       >            1  H2O              1  CH4
           1  H2O              1  CO       >            1  H2               1  CO2
           1  H2               1  CO       >            1  H2O              1  C
 Thermodynamic equilibrium:
 p =   101325.000000000000       Pa
 V =   1.1494736214777213E-003  m^3
 Vg=   1.1494736214777213E-003  m^3
 Vc=   6.6277004694695433E-029  m^3
 T =   1000.0000000000000       K
 S =   3.2156224455464173       J/K
 H =  -361.49569784847614       J
 G =  -3577.1181433948932       J
gas-phase:
 species          moles      mole frac
H2         2.530325E-03   1.806320E-01
O2         7.895447E-23   5.636314E-21
OH         1.368507E-14   9.769342E-13
H2O        1.119240E-03   7.989907E-02
H          1.345763E-11   9.606986E-10
O          4.290334E-23   3.062736E-21
H2O2       2.812345E-22   2.007645E-20
CH4        3.207386E-06   2.289653E-04
CO2        7.119573E-04   5.082441E-02
CO         1.112825E-03   7.944114E-02
N2         8.530620E-03   6.089744E-01
condensed phase:
C          1.247074E-23   1.000000E+00

 DETCHEM -- copyright by O. Deutschmann
 For further information visit www.detchem.com
```

First, the atomic compostion of all species is printed, in order to allow checking that the data is read correctly. Then the initial conditions are shown (thermodynamic variables and composition). As an additional information, one set of hypothetical reactions is printed. Finally, the thermodynamic variables and the compostion of the mixture in equilibrium are shown.

7.3.2 *equil.plt*

DETCHEM^{EQUIL} creates a file ***equil.plt*** that contains the equilibrium composition(s) and their thermodynamic variables. This file may be useful for graphical data evaluation in case of multi temperature runs. However, a new file is created upon each call of DETCHEM^{EQUIL}, so data will not be appended.

The output file is in TECPLOT format. TECPLOT will create a line graph upon loading the file ***equil.plt***. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the file are: initial temperature, equilibrium values of pressure, volume, temperature, entropy, enthalpy, free enthalpy, amount of species, and species mole fractions (separately for gas and condensed phase).



7.4 Example

The distribution contains an example for DETCHEM^{EQUIL} in the directory ***($DETCHEM_DIR)/EQUIL/example***. It contains three input files ***equil.inp***, ***thermdata***, and ***moldata***. The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

7.5 Setting up a problem

It is recommended to create new problems in separate directories. The user must supply the files ***equil.inp***, ***thermdata***, and ***moldata***. In most cases it might be convenient to copy an existing directory and modify the files.

If the same species data is used by more than one DETCHEM application, it might also be advisible to save the species information together with ***thermdata*** and ***moldata*** in a separate directory and create symbolic links to your working directory, e.g.

```
ln -s ~/detchem/data/species/thermdata .
```

Make sure that all species in your species list are also contained in ***thermdata*** and ***moldata***. Define the initial conditions and tolerances and start the program.

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***equil***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/equil*** and call ***./equil***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 8

DETCHEM$^{\mathrm{BATCH}}$

8.1 Introduction

The batch code simulates homogeneous gas-phase and heterogeneous surface reactions in a batch reactor. The user can specify his own time-dependent pressure, volume and temperature profiles. In case of missing temperature profile, isenthalpic or adiabatic analysis can be performed.

8.1.1 Governing equations

$$\frac{dn_k}{dt} = V\dot{\omega}_k + A\dot{s}_k \quad \text{gas-phase species} \tag{8.1}$$

$$\frac{dn_k}{dt} = \alpha A\dot{s}_k \quad \text{surface species} \tag{8.2}$$

$$pV = nRT \quad \text{ideal gas equation} \tag{8.3}$$

In the above equations $\dot{\omega}_k$, $\dot{s}_k$, A, V, α represent the gas-phase reaction rate, surface reaction rate, catalytic surface area, reactor volume, and surface relaxation factor respectively. The species concentrations n_k are expressed in terms of mole numbers.

If the reactor temperature as a function of time is not known the user can define an isenthalpic or adiabatic reactor:

$$\frac{dH}{dt} = 0 \quad \text{in the isenthalpic case} \tag{8.4}$$

$$\frac{dQ}{dt} = 0 \quad \text{in the adiabatic case} \tag{8.5}$$

The expressions (8.4) and (8.5) provide an additional equation for the reactor temperature.

DETCHEM$^{\mathrm{BATCH}}$ can perform sensitivity analysis for all species with respect to all or reduced number of kinetic parameters. Pre-exponential factors A_j ($A_{0,j}$ and $A_{\infty,j}$ in case of Troe reactions) and initial sticking coefficients S_j^0 are considered parameters in DETCHEM$^{\mathrm{BATCH}}$. Absolute and relative sensitivity values are calculated using the following expressions:

$$E_{i,j} = \frac{dn_i}{dp_j} \quad \text{absolute sensitivities} \tag{8.6}$$

$$E_{i,j}^{rel} = \frac{p_j}{n_i}\frac{\partial n_i}{\partial p_j} = \frac{\partial \ln n_i}{\partial \ln p_j} \quad \text{relative sensitivities} \tag{8.7}$$

In order to improve the accuracy of the computed relative sensitivities, an option for sensitivity analysis with respect to $\ln p_j$ is also introduced:

$$E_{i,j}^{\ln p_j} = p_j\frac{\partial n_i}{\partial p_j} = \frac{\partial n_i}{\partial \ln p_j} \tag{8.8}$$

8.1.2 Solution

In DETCHEM^{BATCH} the standard implicit solvers LIMEX or DAESOL can be used to solve the coupled governing equations. The temperature solution in batch code offers user-defined time-dependent thermal conditions and non-isothermal isenthalpic or adiabatic conditions in a batch reactor.

8.2 User Input

Before running DETCHEM^{BATCH} the user must prepare an input file ***batch.inp***. The program considers four batch reactor types:

1. with user-defined volume and temperature profiles
2. with user-defined pressure and temperature profiles
3. with user-defined volume profile (isenthalpic or adiabatic reactor)
4. with user-defined pressure profile (isenthalpic or adiabatic reactor)

Examples for these four cases are shown in the table below.

The file ***batch.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition, the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

example 1

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 0   # 0 LIMEX, 1 DAESOL
  p/Pa = 8000.d0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-02
  atol = 1.0e-09

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <V_PROFILE>
    0.0   1.5
   30.0   1.0
   70.0   1.5
  100.0   1.0
  </V_PROFILE>

  <T_PROFILE>
    0.0   1173
   50.0   1273
  100.0   1173
  </T_PROFILE>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
    monitor = 1
  </OUTPUT>

</BATCH>
```

example 2

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 0   # 0 LIMEX, 1 DAESOL
  V/m3 = 1.0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-02
  atol = 1.0e-09

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <P_PROFILE>
    0.0    8000
   30.0   10000
   70.0    8000
  100.0   10000
  </P_PROFILE>

  <T_PROFILE>
    0.0   1173
   50.0   1273
  100.0   1173
  </T_PROFILE>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
    monitor = 1
  </OUTPUT>

</BATCH>
```

example 3

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 1   # 0 LIMEX, 1 DAESOL
  p/Pa = 8000.d0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-03
  atol = 1.0e-09

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <V_PROFILE>
    0.0   1.5
   30.0   1.0
   70.0   1.5
  100.0   1.0
  </V_PROFILE>

  <CONST_QUANTITY>
    T/K            = 1173
    const_quantity = Q      # Q or H
    surf_thermdata = n
  </CONST_QUANTITY>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
    monitor = 1
  </OUTPUT>

</BATCH>
```

example 4

```
{include species.inp}
{include mech.inp}

<BATCH>

  solver_id = 1   # 0 LIMEX, 1 DAESOL
  V/m3 = 1.0
  A/m2 = 1.0
  time = 100

  h    = 1.0e-08
  hmax = 1.d0
  rtol = 1.0e-02
  atol = 1.0e-09
  monitor = 1

  <MOLEFRAC>
     C3H6 0.996
     CH4  0.003
     C3H8 0.001
  </MOLEFRAC>

  <P_PROFILE>
    0.0    8000
   30.0   10000
   70.0    8000
  100.0   10000
  </P_PROFILE>

  <CONST_QUANTITY>
    T/K            = 1173
    const_quantity = Q      # Q or H
    surf_thermdata = n
  </CONST_QUANTITY>

  <OUTPUT>
    mole          = y
    concentration = y
    mole_fraction = y
    mass_fraction = y
    fileNr = 1
    dt_out = 0.01
  </OUTPUT>

</BATCH>
```

The input file must always contain either user-specified pressure (exapmles 1 and 3) or volume (examples 2 and 4) profile, but not both profiles simultaneously. The user can also define a time-dependent temperature profile (examples 1 and 2) or specify adiabatic (examples 3 and 4) or isenthapic conditions. Initial values for pressure, volume and/or temperature must be specified, provided the corresponding parameter has no user-specified profile. This means that in case of user-defined volume profile, initial pressure is to be given (P/Pa in examples 1 and 3), in case of user-defined pressure profile, initial volume is to be given (V/m3 examples 2 and 4), and in the adiabatic (or isenthalpic) examples initial temperatures (T/K in the **<CONST_QUANTITY>** section in examples 3 and 4) have to be specified. The simultaneous presence of user-defined profile and initial value for the same parameter (p, V or T) is ambiguous and causes an error message.

Profiles are defined by pairs of values. The first value is the time, and the second one is the value of the corresponding parameter (p, V or T). Time must always increase, i.e., each time value must be greater than the preceding times. For instance:

```
<V_Profile>
   0.0   0.001
  20.0   0.002
  40.0   0.0025
  60.0   0.0028
 100.0   0.003
</V_Profile>
```

The user can specify an arbitrary number of pairs. If the profile consists only of one pair of values, the specified parameter (p, V or T) remains constant during the whole simulation. Intermediate states are determined by means of linear interpolation. For example: according to the volume profile shown above, the reactor volume after 30 s will be 0.00225 m^3.

Adiabatic and isenthalpic conditions are specified in the <**CONST_QUANTITY**> section. It contains the initial reactor temperature T/K and the options const_quantity and surf_thermdata. The const_quantity option can be set to Q for adiabatic and to H for isenthalpic simulation. For adiabatic-isobaric problems either of both values can be used. However, the user is recommended to use const_quantity=H in this case because the solution converges faster and provides more accurate results. The surf_thermdata determines whether thermodynamic data for the surface species have to be taken into account (y) or not (n). This option ist set to n by default because in most cases thermodynamic data for the surface species are not available.

The options within the **<BATCH>** tag are as follows:

solver_id=*i*	swtich between the LIMEX (0) and DAESOL (1) solvers (default: 0)
p/Pa=*d*	initial pressure [Pa]
V/m3=*d*	initial reactor volume [m^3]
A/m2=*d*	catalytic surface area [m^2]
time=*d*	integration time [s]
ini_surf=*b*	calculate equilibrium surface coverages for initialization (default: no). Implemented only with the LIMEX solver
cov_relax=*d*	under relaxation factor for surface coverages (default: 1)
atol=*d*	absolute tolerances for concentrations [mol/m^3] (default: 1e-12)
rtol=*d*	relative tolerances for concentrations (default: 1e-6)
h=*d*	initial integration step size [s] (default: 1e-10)
hmax=*d*	maximal integration step size [s] (default: not defined, the solver determines the step size)
daesol_ESW=*d*	tolerance needed by the DAESOL solver (default: 1.0d-02)
daesol_ETA=*d*	tolerance needed by the DAESOL solver (default: 1.0d-11)
<**MOLEFRAC**>	initial gas-phase mole fractions
<**MASSFRAC**>	initial gas-phase mass fractions
<**COVERAGE**>	initial surface coverages (default: coverages from <SPECIES> section)
<**V_Profile**>	user-specified time dependent volume profile
<**p_Profile**>	user-specified time dependent pressure profile
<**T_Profile**>	user-specified time dependent temperature profile
<**CONST_QUANTITY**>	section to define isenthalpic or adiabatic conditions
<**OUTPUT**>	output options
<**SENSITIVITY**>	sensitivity analysis (only with the DAESOL solver)

The tolerances daesol_ESW and daesol_ETA are needed by the the DAESOL solver. For simple gas-phase problems, default values work fine, but by stiff gas-phase and surface systems convergence difficulties may occur. In case of divergence or slow convergence, following values are recommended: daesol_ESW in the range 1E-2 → 1E-7 and daesol_ETA in the range 1E-10 → 1E-14.

The options within the **<OUTPUT>** tag are:

mole=*b*	output in terms of moles; output file: ***batch_n_***.plt*** (default: y)
concentration=*b*	output in terms of concentrations; output file: ***batch_c_***.plt*** (default: n)
mole_fraction=*b*	output in terms of mole fractions for gas-phase species and of coverages for surface species; output file: ***batch_x_***.plt*** (default: n)
mass_fraction=*b*	output in terms of mass fractions for gas-phase species and of coverages for surface species; output file: ***batch_y_***.plt*** (default: n)
fileNr=*i*	output file number (default: 1)
dt_out=*d*	minimum step size for output [s] (default: none, solution is written at every step)
monitor=*i*	0 - no time output on the screen, 1 - time output on the screen (default: 0)

The options within the **<CONST_QUANTITY>** tag are:

T/K=*d*	initial temperature [K]
const_quantity	H for isenthalpic problems, Q for adiabatic problems (default: H)
surf_thermdata=*b*	thermodynamic data for the surface species (default: n)

The options within the **<SENSITIVITY>** tag are:

complete=*b*	Sensitivity analysis with respect to all kinetic parameters (default: n)
absolute_sensitivity=*b*	output of absolute sensitivities; output file: ***batch_sens_abs_***.plt*** (default: n)
relative_sensitivity=*b*	output of relative sensitivities; output file: ***batch_sens_rel_***.plt*** (default: n)
d(n_i)/d(lnp_j)=*b*	computation of sensitivity values with respect to $\ln p_j$ (eq. 8.8); (default: y)
threshold_lnp=*d*	computation of sensitivity values with respect to $\ln p_j$ (eq. 8.8) only for $p_j >$ **threshold_lnp**; (in force only when **d(n_i)/d(lnp_j)**=y; default: 1000)

The option complete in the **<SENSITIVITY>** tag determines with respect to which kinetic parameters sensitivity coefficients have to be computed. If it is set to yes sensitivities with respect to all reactions will be calculated. If complete=n, only flagged reactions in the mechanism file will be taken into account. The flag is the symbol (*) and it has to be put before the first kinetic parameter (column 45). In the example below, only the pre-exponential factor of the first reaction will be included in the sensitivity analysis.

```
CH      +H      =C      +H2             * 1.100E+14  0.0      0.000
3CH2    +C      =C2H    +H                0.500E+14  0.0      0.000
```

The way DETCHEM$^{\text{BATCH}}$ internally computes the sensitivity values is controlled by the option **d(n_i)/d(lnp_j)**. If it is set to (y), sensitivities $E_{i,j}^{\ln p_j}$ will be calculated, otherwise absolute sensitivities $E_{i,j}$ will be calculated. Relative sensitivities $E_{i,j}^{rel}$ are derived from $E_{i,j}^{\ln p_j}$ by multiplying by n_i and from $E_{i,j}^{p_j}$ by multiplying by n_i and dividing by p_j. The first way is the more accurate one from numerical point of view. Relative sensitivities are more useful than absolute sensitivities during mechnism development. For that reason, users are recommended to use **d(n_i)/d(lnp_j)**=y.

8.3 Output

8.3.1 Screen output

On screen the integration progress is monitored. The values in the column displayed stand for time (monitor = 1).

```
*************************************************************************
****                          LIMEX                                  ****
**** (C) Konrad-Zuse-Zentrum fuer Informationstechnik Berlin (ZIB)  ****
****                 version 4.2A1 of 2000/03/17                     ****
*************************************************************************

    time,s
    0.0000000E+00
    1.0101010E-02
    4.7034414E-01
    1.4703441E+00
    ...
    1.0000000E+02
BATCH finished!

 DETCHEM -- copyright by O. Deutschmann
 For further information visit www.detchem.com
```

8.3.2 *batch_***.plt*

DETCHEM$^{\text{BATCH}}$ produces four output files: ***batch_n_***.plt***, ***batch_c_***.plt***, ***batch_x_***.plt*** and ***batch_y_***.plt***. The output files are in TECPLOT format. TECPLOT will create a line graph upon loading a plt file. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the files are: time, pressure, volume, temperature and species data. Species data can be written as:

1. mole numbers - prefixed with n and written in ***batch_n_***.plt***
2. concentrations - prefixed with c and written in ***batch_c_***.plt***
3. mole fractions - prefixed with x and written in ***batch_x_***.plt*** (only gas-phase species)

4. mass fractions - prefixed with y and written in ***batch_y_***.plt*** (only gas-phase species)
5. coverages - prefixed with cov and written in ***batch_x_***.plt*** and ***batch_y_***.plt*** (only surface species)

The output of the four plt files is controlled via the options mole, concetration, mole_fraction and mass_fraction in the <**OUTPUT**> section. By default, only the ***batch_n_***.plt*** is written.

8.3.3 *batch_sens_***.txt*

Absolute sensitivity coefficients (eq. 8.6) are written to the file ***batch_sens_abs***.txt***, while relative sensitivities (eq. 8.7) are written to ***batch_sens_rel***.txt***. Both files have a similar structure. The first line contains one of the following three strings identifying the kind of the written sensitivity values:

- SENS_ij=d(N_i)/d(P_j) - for absolute sensitivities $E_{i,j}$ (in case **d(n_i)/d(lnp_j)**=n, only in ***batch_sens_abs***.txt***)
- SENS_ij=d(N_i)/d(lnP_j) - for sensitivities $E_{i,j}^{\ln p_j}$ (in case **d(n_i)/d(lnp_j)**=y, only in ***batch_sens_abs***.txt***)
- SENS_ij=d(lnN_i)/d(lnP_j) - for relative sensitivities $E_{i,j}^{rel}$ (only in ***batch_sens_rel***.txt***)

The variables' names for which sensitivity analysis has been done, are listed in a **<VARIABLES>** tag. Such variables could be not only the species' concentrations n_i but pressure P, volume V and temperature T of the reactor as well. It is followed by a **<REACTIONS>** tag, where the reactions and their corresponing parameters (in case **d(n_i)/d(lnp_j)**=n) or the natural logarithm of the parameters (in case **d(n_i)/d(lnp_j)**=y) used for the sensitivity analysis are listed. The tag also provides information about the kind of each reaction: gas-phase or surface; Arrhenius, Troe, global or reaction with given sticking coefficient. The **<SENSITIVITY_VALUES>** tag contains the sensitivity values ($E_{i,j}$, $E_{i,j}^{\ln p_j}$ or $E_{i,j}^{rel}$) computed during the integration time. Files ***batch_sens_abs***.txt*** additionally contain the tag **<SOLUTION_MAX_VALUES>** where maximal values for all variables for the whole integration time are stored.

8.4 Examples

There is one example supplied along with the installation. The example directory contains the following files: ***batch.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

8.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

8.6 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***batch***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/batch*** and call ***./batch***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 9

DETCHEM$^{\mathrm{MPTR}}$

9.1 Introduction

DETCHEM$^{\mathrm{MPTR}}$ simulates homogeneous reactions in multiple homogeneous phases and heterogeneous reactions between phases in a zero-dimensional multi-phase tank reactor (MPTR). However, a two-dimensional interface area needs to be defined, where interphase reactions (reactions between different phases) and phase transitions take place. The user can specify an external time-dependent temperature profile with or without heat-transfer effects or else choose isenthalpic analysis.

9.1.1 Governing equations

The program solves equations for reaction rates and energy. The reaction rate over all phases is:

$$\frac{dn_i}{dt} = \sum_{phases} V_{phase} \cdot \dot{\omega}_i + A_{interface} \cdot \dot{s}_i \tag{9.1}$$

Another equations deals with the heat transfer via an user-defined interface area:

$$\frac{dH}{dt} = k_W \cdot \left(T^{extern} - T\right) + Q_{const} \qquad \text{or:} \quad T = T^{extern} \tag{9.2}$$

The species concentration in one phase is:

$$c_i = n_i \cdot \left(\sum_{i \in phase} n_i \cdot v_i \right)^{-1} \qquad v_i = \begin{cases} \frac{M_i}{\rho_i} & \text{condensed} \\ \frac{RT}{p} & \text{gas phase} \end{cases} \tag{9.3}$$

In equation (9.1) $\dot{\omega}_k$ and $\dot{s}_k$ represent the reaction rate in a homogeneous phase and the reaction rate at a phase boundary, respectively. $A_{interface}$ is the user-defined constant area of the interface and V_{phase} the phase volume. Equation (9.2) determines if and how the external temperature profile affects the internal change of enthalpy. The heat transfer coefficient k_W considers the interface area and the external temperature profile, whereas Q_{const} is a constant heating rate independent from both the temperature profile and the interface area. Moreover, it is possible to force the external exactly on the internal temperature. Concentrations within one phase are calculated with equation (9.3).

9.1.2 Solution

In DETCHEM$^{\mathrm{MPTR}}$ the standard semi-implicit solver LIMEX is used to solve the coupled governing equations.

9.2 User Input

In order to run DETCHEM$^{\mathrm{MPTR}}$ the input file ***mptr.inp*** must be provided, which is exemplary depicted below.

```
<SPECIES>
    <GASPHASE>                          # space-separated list of all gas phase species
        Sp1(g) Sp2(g) H2O N2
    </GASPHASE>

    <PHASE liquid>
        DCSLMAX = 20                    # optional list of liquid species with corresponding densities in kg/m3
        Sp1(l) 1320                     # set maximum number of species in the PHASE tag
        # ...
    </PHASE>

    <PHASE aq>
        H2O(l)  998                     # optional list of species in an aqueous phase and their solvent liquid water
        Sp1(aq) 1100                    # liquid water is listed here, if the aqueous phase is applied
        # ...
    </PHASE>
# ...
</SPECIES>
<MECHANISM>
    <HOMOGENEOUS>                       # define homogeneous reactions with reactants of one phase and products of
                                        # any phase, the volume-specific rate is multiplied with the phase volume

        <GLOBAL>
            2 Sp1(l) > Sp2(g)
            <ARRHENIUS>                 # define kinetic parameters of the reaction
                Sp1(l) 2                # define reaction order
                A/cm_units = 1e9
                beta = 0                # kinetic information not necessary if equal zero
                Ea/kJ_mol = 150         # activation energy in kJ mol-1
            </ARRHENIUS>
        </GLOBAL>
        # ...
    </HOMOGENEOUS>

    <INTERPHASE>                        # define phase transitions and interfacial reactions between two phases
                                        # reactants may originate from two different phases
        <REACTION>                      # the area-specific rate is multiplied with the interface_area
            H2O = H2O(l)
            A/SIunits = 0.086
            beta = 0.5
        </REACTION>
        # ...
    </INTERPHASE>
</MECHANISM>
<MPTR>                                  # define reaction conditions
    <T-PROFILE>                         # define temperature ramp
        0[s]       311[K]               # starting at 0 seconds and 311 Kelvin
        281[min]   873[K]               # linear temperature increase to 873 Kelvin after 281 minutes
        300[min]   873[K]               # constant temperature for 19 minutes (from 281 minutes to 300 minutes)
    </T-PROFILE>
    p = 1E5[Pa]                         # specify constant pressure
    V_gas = 6E-5[m3]                    # specify constant gas phase volume

    <GASPHASE>                          # initial gas phase composition in MOLAR fraction
        N2    *
    </GASPHASE>
    # ...

    <PHASE aq>
        mass = 19.99[mg]
        Sp1(aq) 0.325
        H2O(l)  *
    </PHASE>

    interface_area = 30.0[cm2]          # constant parameter for surface area, that is the interface
                                        # (phase boundaries)and the contact area for heat transfer via kW
    kW = 10[W/m2/K]                     # optioal heat transfer coefficient
    Qconst = 0.4[W]                     # optional absolute heating rate
    Textern = no                        # if yes, the internal temperature follows exactly the external
                                        # temperature profile (default: no)
    atol = 1e-18                        # LIMEX variable absolute tolerances
    rtol = 1e-6                         # LIMEX variable relative tolerances
    h_ini = 1e-17                       # LIMEX variable initial step size
    h_max = 2.0                         # LIMEX variable maximal step size

    <INLET>                             # Specify an optional inlet flow
        <GASPHASE>
            T=Textern                   # Inlet temperature is the given temperature profile
            volume_flux=1.0e-7          # volume_flux (in m^3/s) or mass_flux (in kg/s) of gas inlet
            N2  *                       # Molar composition of gas inlet
        </GASPHASE>
    </INLET>
    <OUTLET>                            # Specify an optional outlet flow
        <GASPHASE>
            volume_flux=1.0e-7          # volume_flux (in m^3/s) or mass_flux (in kg/s) of gas outlet;
                                        # same value as inlet is suggested
                                        # The current composition in the simulation is used as outlet
        </GASPHASE>
    </OUTLET>
```

</MPTR>

The input file is structured in three main parts, that are **<SPECIES>**, **<MECHANISM>** and **<MPTR>**. As usual, each part is delimited by a beginning and an ending tag. Within these main parts several subtags are defined:

1. <SPECIES> ... </SPECIES> → with multiple user-defined phases declared via subtags as follows:
   - <GASPHASE> ... </GASPHASE> → a space-separated list of all relevant gas phase species
   - <PHASE *NAME*> ... </PHASE> → replace *NAME* with any user-defined phase name, then provide a list comprising all species of this phase, each space-separated from its corresponding density, examples:
     - <PHASE liquid> ... </PHASE> → with all liquid species
     - <PHASE aq> ... </PHASE> → with all aqueous species and liquid water
     - <PHASE sSOLID> ... </PHASE> → with a solid phase where SOLID is the species name
   - <SURFACE> ... </SURFACE> → Optional: list of surface species and INITIAL coverage
2. <MECHANISM> ... </MECHANISM> → reaction rates are defined per volume or per interface area
   - <HOMOGENEOUS> ... </HOMOGENEOUS> → define homogeneous reactions with reactants from one phase; this tag is equivalent to <LIQUID> ... </LIQUID>
   - <INTERPHASE> ... </INTERPHASE> → reactions or phase transitions at any phase boundary
   - <SURFACE> ... </SURFACE> → reactions at a specified catalytic surface
3. <SURFACE-MODEL> ... </SURFACE-MODEL> → If surface species and reactions are defined
4. <MPTR> ... </MPTR> → specifications regarding the reaction conditions
   - <T-PROFILE> ... </T-PROFILE> → define a temperature ramp (like in the example above)
   - p and V_gas → define initial pressure **p**=*d* in Pa and gas volume **V_gas**=*d* in m3
   - phase tags:
     - <GASPHASE> ... </GASPHASE → a list of initial gas phase species with molar fraction
     - other phases → like above but either specify total mass and corresponding mass fraction or total volume and molar fraction
   - <SURFACE> ... </SURFACE> → Define gaschem, wall tag, relaxation factors (cov_relax, sdot_relax), diffusion constant (Diff)
   - interphase_area → define the interface area where the phases and the heat source make contact
   - kW or Qconst → define a heat transfer coefficient **kW**=*d* and or a constant heating rate **Qconst**=*d*
   - LIMEX solver specifications → **atol**=*d*, **rtol**=*d*, **h_ini**=*d* and **h_max**=*d*
   - <INLET> ... </INLET> → define inlet flows (see example)
   - <OUTLET> ... </OUTLET> → define outlet flows (see example)

The file ***mptr.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition, the file ***thermdata*** must be located in the executing directory. The file ***moldata*** is not mandatory.

Initial values for pressure and volume are mandatory. Futhermore, the program offers different heat transfer modes:

1. with a heat transfer coefficient kW connected with to the temperature profile and user-defined interface area
2. with a constant heating rate Qconst, the temperature profile has no effect
3. with kW and Qconst together
4. with the internal temperature following directly an external temperature profile (set Textern=yes)

The options within the **<MPTR>** tag are as follows:

<**T-Profile**>	user-specified time dependent temperature profile
p=*d*	constant pressure, unit in brackets, default is [Pa]
V_gas=*d*	initial gas phase volume [m^3]
<**GASPHASE**>	initial gas-phase mole fractions
user-defined phase tags	initial phase mass and mass fractions or initial phase volume and molar fractions
<**SURFACE**>	Surface options gaschem, wall tag, relaxation factors, diffusion constant
interface_area=*d*	interface area for heat transfer, interphase reactions and phase transitions, default is [cm^2]
kW=*d*	heat transfer coefficient [W/m^2/K]
Qconst=*d*	constant heating rate [W]
Textern=*b*	if yes, the internal follows exactly the external temperature
atol=*d*	absolute tolerances for amount of substance [mol]
rtol=*d*	relative tolerances for amount of substance
h_ini=*d*	initial integration step size [s]
h_max=*d*	maximal integration step size [s]

When handling numerical problems, try altering the LIMEX parameters, particularly **h_ini**.

9.3 Output

9.3.1 Screen output

On screen the integration progress is monitored. The three arising columns are the time, the external temperature and the internal temperature.

```
*************************************************************************
****               DETCHEM * Multiphase Tank Reactor                 ****
****                         S. Tischer                              ****
****                   VERSION 2.9.1 ** 2022-12-15                   ****
*************************************************************************
      0.00    311.00    311.00
      0.00    311.00    311.00
      0.00    311.00    311.00
      ...
  18000.00    873.00    886.33
```

9.3.2 *mptr.plt*

DETCHEM^{MPTR} produces an output file ***mptr.plt*** which is in TECPLOT format. TECPLOT will create a line graph upon loading a .plt file. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The columns in the file are time, Textern, T and the amount of substance in mole per species. The species names in the column head are in quotation marks.

```
TITLE = "DETCHEM_MPTR"
VARIABLES =
          time         Textern              T          Sp1(g)          Sp2(g)    ...            m_aq      Film_thick       heat_flow
  0.000000E+000   3.110000E+002   3.110000E+002   0.000000E+000   0.000000E+000    ...   1.999000E-005   6.475476E-006   0.000000E+000
  1.000000E-017   3.110000E+002   3.110000E+002  -4.522575E-059  -1.370314E-052    ...   1.999000E-005   6.475476E-006  -4.000000E-001
  1.010000E-015   3.110000E+002   3.110000E+002  -4.522579E-053  -3.788911E-047    ...   1.999000E-005   6.475476E-006  -4.000000E-001
```

9.4 Examples

There is one example supplied along with the installation. The example directory contains the following files: ***mptr.inp***, ***thermdata***, the output file ***mptr.plt*** and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

9.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

9.6 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***mptr***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/mptr*** and call ***./mptr***.

The latter option does not require setting of system variables and is therefore less system specific.

An example of a simulation result is given in figure 9.1.

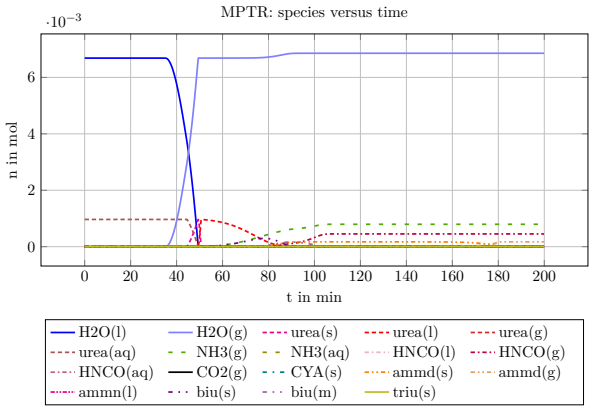


Figure 9.1: Example: Simulation of the decomposition of urea-water solution.

Chapter 10

DETCHEM^{CSTR}

10.1 Introduction

The program DETCHEM^{CSTR} simulates the behavior of the continuous stirred tank reactor (CSTR) under different temperature conditions for applications involving gasses. The code is designed to simulate homogeneous gas-phase and heterogeneous surface reactions in a CSTR.

10.1.1 Governing equations

$$\frac{dc_k}{dt} = \dot{\omega}_k + \frac{A}{V}\dot{s}_k + \frac{\dot{V}_{\text{in}}}{V}c_{k,\text{in}} - \frac{\dot{V}_{\text{out}}}{V}c_k \quad \text{gas-phase species} \tag{10.1}$$

$$\frac{dc_k}{dt} = \alpha\dot{s}_k \quad \text{surface species} \tag{10.2}$$

$$\frac{dH}{dt} = \sum_k \left(h_k(T_{\text{in}})c_{k,\text{in}}\dot{V}_{\text{in}} - h_k(T)c_k\dot{V}_{\text{out}}\right) \tag{10.3}$$

$$\dot{V}_{\text{out}} = \frac{T_{\text{out}}\bar{M}_{\text{in}}}{T_{\text{in}}\bar{M}_{\text{out}}}\dot{V}_{\text{in}} \tag{10.4}$$

In the above equation $\dot{\omega}_k$, $\dot{s}_k$, A, V, $\dot{V}_{\text{in}}$, $\dot{V}_{\text{out}}$, α, T_{out}, T_{in}, $\bar{M}_{\text{in}}$, $\bar{M}_{\text{out}}$, h_{in}, and h represent the gas-phase reaction rate, surface reaction rate, surface area, reactor volume, inlet flow, outlet flow, coverage relaxation factor, outlet temperature, inlet temperature, mean inlet molar mass, mean outlet molar mass, enthalpy at inlet and enthalpy at outlet respectively.

10.1.2 Solution

In DETCHEM^{CSTR} the standard implicit code LIMEX is used to solve the coupled governing equations. The temperature solution in cstr code offers isothermal and non-isothermal adiabatic conditions in a continuous stirred tank reactor.

10.2 User Input

Before running DETCHEM^{CSTR} the user must prepare an input file ***cstr.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
    time=1.          # surface integration time for steady state
  </CHEMSURF>
</SURFACE-MODEL>

<CSTR>
  <INITIAL>
    T = 1200         # reactor temperature in K
```

```
    <MOLEFRAC>
       NO2    0.1     # species name and mole fraction
       N2     *       # balance mole fraction
    </MOLEFRAC>
  </INITIAL>

  <INLET>
    T = 1200          # inlet temperature in K
    <MOLEFRAC>
       NO2   0.1      # inlet mole fraction
       N2    *
    </MOLEFRAC>
    Vflux = 0.01      # inlet flux in m^3/s
  </INLET>

  p/Pa= 100000        # pressure in Pa
  time=100            # integration time

  volume = 1          # reactor volume in m^3
  surface = 1         # catalytic surface in m^2

  isothermal=n        # isothermal=y: T=const,
                      # isothermal=n: solve enthalpy equation
  fileNr=1            # number of output file
  dt_out=0.05         # minimum output step size

  ini_surf=n          # find steady-state surface coverages  before
                      # integration starts
</CSTR>
```

The file ***cstr.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

The options within the **<CSTR>** tag are as follows.

<INITIAL>	initial conditions
<INLET>	inlet conditions
p/Pa=*d*	pressure [Pa] (default: 100000)
p/bar=*d*	pressure [bar] (default: 1)
time=*d*	integration time [s]
volume=*d*	reactor volume [m^3] (default: 1)
surface=*d*	catalytic surface area [m^2]
isothermal=*b*	isothermal (yes) or adiabatic (no) conditions (default: no)
dT/s=*d*	linear temperature gradient (per second) for TPD simulations
dT/min=*d*	linear temperature gradient (per minute) for TPD simulations
cov_relax=*d*	under relaxation factor for surface coverages (default: 1)
ini_surf=*b*	call CHEMSURF for initial coverages (yes) or use default (no) (default: no)
thermal_coverage_contribution=*b*	consider surface species in heat balance equation – requires reliable thermodynamic data (default: no)
fileNr=*i*	file number for output, e.g. cstr001.plt (default: 1)
dt_out=*d*	minimum step size for output [s] (default: none)
atol=*d*	absolute tolerances for concentrations [mol/m^3] (default: 1e-12)
rtol=*d*	relative tolerances for concentrations (default: 1e-6)
h=*d*	initial integration step size [s] (default: 1e-10)

The subsection **<INITIAL>** may contain:

T=*d*	temperature [K]
<**MOLEFRAC**>	gas-phase mole fractions
<**MASSFRAC**>	gas-phase mass fractions
<**COVERAGE**>	surface coverages (default: coverages from <SPECIES> section)

and the subsection **<INLET>**:

T=*d*	temperature [K]
<**MOLEFRAC**>	gas-phase mole fractions
<**MASSFRAC**>	gas-phase mass fractions
Vflux=*d*	inlet volume flux (at current T,p) [m^3/s]

10.2.1 User-defined volume function

Instead of declaring a constant volume of the reactor, the user may also provide a user-defined function to calculate the volume. The procedure is similar to the declaration of user-defined rate laws as described in

section 3.7. Replace the definition of a constant volume by a formula declaration section **<UDF>** and then refer to the user-defined variable name by **UDF:volume**. The calculated value must be in SI units, i. e. in m^3. For example

```
<CSTR>
   ...
   <UDF>
      V = 0.001 + 0.0002 * sin(6.2831853 * t / 60)
   </UDF>
   UDF:volume = V
   ...
</CSTR>
```

This will change the volume of the reactor between 0.8 and 1.2 liters periodically with a period of one minute. The calculation of the user-defined volume can constist of more than one line. Thus it is possible to do more complex calculations using intermediate variables. The default variables are **t** for time, **T** for temperature, **p** for pressure, and **theta**_+*surface species name* for coverages.

10.3 Output

10.3.1 Screen output

On screen the integration progress is monitored. The values in the two columns displayed stand for time and temperature.

```
**************************************************************************
****                          DETCHEM CSTR                              ****
****                    S. Tischer, O. Deutschmann                      ****
****                   VERSION 2.9.1 ** 2022-12-15                      ****
**************************************************************************

    0.00E+00    1200.0
    5.02E-02    1102.1
    1.02E-01    1083.1
    ...
      100.00    1040.5
 CSTR finished successfully
```

10.3.2 *cstr***.plt*

DETCHEM^{CSTR} produces one output file ***cstr***.plt***. The output file is in TECPLOT format. TECPLOT will create a line graph upon loading the file ***cstr***.plt***. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the file are: time, temperature, species mole fractions (prefixed with X), and species mass fractions (prefixed with Y).

10.3.3 Examples

There is one example supplied along with the installation. The example directory contains the following files: ***cstr.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***noo2m***, ***thermdata***, ***moldata***, and a script file ***go***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable DETCHEM_DIR that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

10.4 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

10.5 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***batch***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/batch*** and call ***./batch***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 11

DETCHEM$^{\text{STAG}}$

11.1 Introduction

The DETCHEM$^{\text{STAG}}$ code simulates the behaviour of a catalytically active stagnation point flow reactor. Model equations couple flow equations together with the energy, species continuity and surface equations.

11.2 Governing Equations

In the stagnation point flow reactor, catalytically active surface interacts with the surrounding flow. Therefore, model equations consist of three parts, i.e., gas-phase equations, surface equations and boundary conditions.

11.2.1 Gas-phase equations

Temperature equation

$$\frac{\partial T}{\partial t} = -\left[\frac{\rho v_x}{\rho} + \frac{1}{\rho c_p}\sum_{i=1}^{N_g} c_{p,i}\, j_i\right]\frac{\partial T}{\partial x} - \frac{1}{\rho c_p}\sum_{i=1}^{N_g}\dot{\omega}_i M_i h_i + \frac{1}{\rho c_p}\frac{\partial}{\partial x}\left(\lambda\frac{\partial T}{\partial x}\right) \tag{11.1}$$

Species equation

$$\frac{\partial Y_i}{\partial t} = -\frac{\rho v_x}{\rho}\frac{\partial Y_i}{\partial x} + \frac{1}{\rho}\dot{\omega}_i M_i - \frac{1}{\rho}\frac{\partial j_i}{\partial x} \tag{11.2}$$

Continuity equation

$$0 = \frac{p}{R}\frac{\bar{M}^2}{T^2}\left[T\sum_{i=1}^{N_g}\frac{\partial_t Y_i}{M_i} + \frac{\partial_t T}{\bar{M}}\right] - 2\rho V - \frac{\partial(\rho v_x)}{\partial x} \tag{11.3}$$

Radial momentum equation

$$0 = -\frac{\rho v_x}{\rho}\frac{\partial V}{\partial x} - V^2 - \frac{\Lambda}{\rho} + \frac{1}{\rho}\frac{\partial}{\partial x}\left(\mu\frac{\partial V}{\partial x}\right) \tag{11.4}$$

The equation for the eigenvalue of the momentum

$$0 = \frac{\partial \Lambda}{\partial x} \tag{11.5}$$

Ideal gas law

$$\rho = \frac{p\bar{M}}{RT} \tag{11.6}$$

Axial velocity v_x appears in the equations. However ρv_x is considered to be an independent variable, not only the axial velocity. Heat capacity c_p, thermal conductivity λ and viscosity μ of the mixture, heat capacity $c_{p,i}$, molar mass M_i and enthalpy h_i of each species and gas-phase reaction rates $\dot{\omega}_i$ are calculated in the DETCHEM library. Diffusion velocity of each species is also introduced in the temperature and species equations. It is calculated at the mixture average level by the following equation.

$$\hat{j}_i = -\left(\rho D_{i,M}\frac{Y_i}{X_i}\frac{\partial X_i}{\partial x} + \frac{{D_i}^T}{T}\frac{\partial T}{\partial x}\right) \tag{11.7}$$

Diffusion velocities are corrected with the Eq. 11.8

$$j_i = \hat{j}_i - Y_i \sum_{k=1}^{N_g} \hat{j}_i \tag{11.8}$$

where Y_i is the mass fraction of each species. Molar mass of the mixture $\bar{M}$, averaged diffusion $D_{i,M}$ and thermal diffusion ${D_i}^T$ coefficient of each species are also calculated in the DETCHEM library. Please note that N_g represents the gas-phase species in the equations.

11.2.2 Surface equations

Chemical processes at the surface can be calculated based on three different approaches: Instantaneous diffusion at the gas/surface interface, reaction-diffusion equations and effectiveness factor approach. Instantaneous diffusion considers a flat surface where the catalyst is virtually distributed. Therefore, it does not account the internal mass transfer limitations. Reaction-diffusion equations and the effectiveness factor approach account for internal mass transfer limitations that are due to the porous washcoat layer. More detailed information can be found in section 2.3.

11.2.3 Boundary conditions

In order to completely describe the system, boundary conditions must also be included.

Gas-Surface/Washcoat Interface

In order to couple the catalytic surface and the surrounding flow, interaction between them must be accounted. Energy balance and species conservation equations are established at the interface. In case of a flat surface, energy balance is calculated as follows

$$\left(\rho_s c_{p,s} d + \rho c_p \Delta x^+\right) \frac{\partial T}{\partial t} = \lambda \frac{\partial T}{\partial x} - \sum_{i=1}^{N_g} \left(j_i + \rho Y_i u\right) h_i - \sigma \varepsilon \left(T^4 - {T_{rad}}^4\right) - \sum_{i=1}^{N_g+N_s} \dot{s}_i M_i h_i + \dot{P} - \frac{\lambda_c}{d} \left(T - T_b\right) \tag{11.9}$$

where d, ρ_s and $c_{p,s}$ are the thickness, density and heat capacity of the catalytic foil, respectively. If there is washcoat on the surface, and the effectiveness factor approach is used, energy balance is calculated as

$$\left(\rho_{w,c} c_{p,wc} t_w + \rho_s c_{p,s} d + \rho c_p \Delta x^+\right) \frac{\partial T}{\partial t} = \lambda \frac{\partial T}{\partial x} - \sum_{i=1}^{N_g} \left(j_i + \rho Y_i u\right) h_i - \sigma \varepsilon \left(T^4 - {T_{rad}}^4\right) - \sum_{i=1}^{N_g+N_s} \dot{s}_i M_i h_i + \dot{P} - \frac{\lambda_c}{d} \left(T - T_b\right) \tag{11.10}$$

where t_w, ρ_{wc} and $c_{p,wc}$ are the thickness, density and heat capacity of the washcoat, respectively. If there is washcoat on the surface, and the reaction-diffusion equations are used, energy balance is calculated as

$$\begin{aligned}\left(\rho_{w,c} c_{p,wc} t_w + \rho_s c_{p,s} d + \rho c_p \Delta x^+\right) \frac{\partial T}{\partial t} &= \lambda \frac{\partial T}{\partial x} - \sum_{i=1}^{N_g} \left(j_i + \rho Y_i u\right) h_i \\ &- \sigma \varepsilon \left(T^4 - {T_{rad}}^4\right) - \sum_{i=1}^{N_g+N_s} \frac{1}{t_w} \left(\int_{y=0}^{\delta} \dot{s}_i M_i h_i dy\right) + \dot{P} - \frac{\lambda_c}{d} \left(T - T_b\right)\end{aligned} \tag{11.11}$$

Species governing equation accounts diffusion and convection processes. Due to the fact that species can be created or destroyed by the surface reactions, the source term is also included. The source term appears different based on the considered surface model. In case of instantaneous diffusion, the species governing equation at the gas/surface interface becomes

$$\rho \frac{\partial Y_i}{\partial t} \Delta x^+ = -j_i - \rho u Y_i + F_{cat/geo} \dot{s}_i M_i \tag{11.12}$$

Since there is mass flux to the surface, velocity cannot be zero. Therefore, Stefan velocity is considered in the species governing equation.

$$u = \frac{1}{\rho} \sum_{i=1}^{N_g} \dot{s}_i M_i \tag{11.13}$$

If the surface model is based on reaction-diffusion equations, the species governing equation at the gas-washcoat interface becomes

$$\rho \frac{\partial Y_i}{\partial t} \Delta x^+ = -j_i - \rho u Y_i + \dot{s}_i, eff M_i \tag{11.14}$$

where the diffusion flux at the gas/washcoat interface is treated as an effective surface reaction rate.

If the surface model is based on the effectiveness factor approach, the species governing equation at the gas-washcoat interface becomes

$$\rho \frac{\partial Y_i}{\partial t} \Delta x^+ = -j_i - \rho u Y_i + \eta F_{cat/geo} \dot{s}_i, eff M_i \tag{11.15}$$

Finally, continuity and radial velocity are also included at the interface.

$$\rho v_x := \rho u \qquad V := 0 \tag{11.16}$$

Inlet boundary

Inlet boundary conditions follow.

$$T = T^0 \qquad Y_i = Y_i^0 \tag{11.17}$$

$$\rho v_x = \rho^0 {v_x}^0 \qquad V = 0 \tag{11.18}$$

In order to close the equation system, a zero gradient condition is used for the mass flow at the inlet

$$\frac{\partial (\rho v_x)}{\partial x} = 0 \tag{11.19}$$

11.3 Solution

In DETCHEM^{STAG} the standard implicit code LIMEX is used to solve the coupled governing equations. PCHIPD package is used for the interpolations needed for the adaptive grid refinement procedure.

11.4 User Input

Before running DETCHEM^{STAG} the user must prepare an input file ***stag.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
   time=10.
  </CHEMSURF>

  Fcatgeo = 30

  <MIXED_DIFF>
    <ZONE>
      ngrid=20
      aspect=1.06
      thickness=1.0d-4
    </ZONE>
    porosity=0.8
    tau=4
    diameter=1.d-8
    <SOLVER>
      time=10.
      hini=1.d-12
      rtol=1.d-3
      atol=1.d-6
    </SOLVER>
  </MIXED_DIFF>

  EFF_MODEL=CH4

</SURFACE-MODEL>

<STAGNATION>
  TIN    =  3.00E+02
  VIN    =     -0.50
  XLEN   =     0.030
  p      =     50000
```

```
  NGRID  =        60
  TIMES  =  0.00E+00
  TIMEF  =  1.00E+02
  HSTEP  =  1.00E-12
  ATOL   =  1.00E-06
  RTOL   =  1.00E-03
  T_ATOL =  1.E-6
  T_RTOL=   1.E-3
  V_ATOL =  1.E-3
  V_RTOL=   1.E-2
  ROVAU_ATOL =  1.E-3
  ROVAU_RTOL=   1.E-2
  EVPGF_ATOL =  1.E-3
  EVPGF_RTOL=   1.E-2
  WASHCOAT2=y
  ASPECT =  1.05
  TRANSIENTOUT=n
  TCONSTANT=n

  <MOLEFRAC>
      H2  5.00E-02
      AR  *
      O2  2.00E-02
  </MOLEFRAC>

  <SURFACE>
    TSURF   =  6.00E+02 [K]
    ALPHA   =  1.44E+00 [W/m/K]
    PINIT   =  2.50E+04 [W/m2]
    PLAST   =  3.00E+04
    PINCR   =  1.00E+03
    RFOI    =  9.00E-03
    TRAD    =  3.00E+02
    EPS     =  2.30E-01
    EPSWC   =  2.50E-01
    HCFOIL  =  3.00E+03
    DFOIL   =  4.00E+02
    HCWC    =  3.00E+03
    DENWC   =  4.00E+02
    TBACK   =  3.00E+02
    PTDM    =  1.00E-07
  </SURFACE>
</STAGNATION>
```

The file ***stag.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition, the files ***thermdata*** and ***moldata*** must be located in the executing directory.

Next, the **<SURFACE-MODEL>** tag must be included in the input file. Please note that <MIXED_DIFF> tag and EFF_MODEL=CH4 are used if there is washcoat over the catalyst support. If there is only catalytic foil without washcoat, or infinitely fast mass transport in the washcoat (instantaneous diffusion model), <MIXED_DIFF> tag and EFF_MODEL=CH4 are not needed in the **<SURFACE-MODEL>** tag, only Fcat/geo is needed for the simulations. Please see the surface models section for more detailed information.

The <STAGNATION> tag This is the opening tag for the stagnation input parameters. The following parameters must be included under this tag. If the default values are not given in the parenthesis for the parameters they must be supplied by the user. Default values for some of the parameters can be used or they can also be supplied by the user.

```
TIN=d               Inlet gas temperature [K]
VIN=d               Axial inlet velocity [m/s]
XLEN=d              Surface-inlet distance [m]
p=d                 Pressure [Pa] (default:1.E5)
NGRID=i             Number of the grid points (default: 60)
TIMES=d             Initial integration time [s] (Default: 0.0)
TIMEF=d             Maximum integration time [s] (Default: 100)
HSTEP=d             Initial time integration step size [s] (Default: 1.E-12)
ATOL=d              Absolute tolerance [mol/m^3] (Default:1.E-6)
RTOL=d              Relative tolerance (Default:1.E-3)
T_ATOL=d            Absolute tolerance for the temperature calculation (optional)
R_TOL=d             Relative tolerance for the temperature calculation (optional)
ROVAU_ATOL=d        Absolute tolerance for the continuity eq. calculation (optional)
ROVAU_RTOL=d        Relative tolerance for the continuity eq. calculation (optional)
EVPGF_ATOL=d        Absolute tolerance for the eigenvalue pressure gradient calculation (optional)
EVPGF_RTOL=d        Relative tolerance for the eigenvalue pressure gradient calculation (optional)
ADAPTIVE=b          Adaptive gridding (Default=n)
GRAD=d              Adaptive grid parameter to insert new grid points in the gas-phase in high gradient regions (needed only if ADAPTIVE=y)
CURV=d              Adaptive grid parameter to insert new grid points in the gas-phase in high curvature regions (needed only if ADAPTIVE=y)
WCGRAD=d            Adaptive grid parameter to insert new grid points in thewashcoat in high gradient regions
                    (needed only if ADAPTIVE=y and WASHCOAT2=y)
WCCURV=d            Adaptive grid parameter to insert new grid points in the washcoat in high curvature regions
                    (needed only if ADAPTIVE=y and WASHCOAT2=y)
ASPECT=d            Aspect ratio for the non-equidistant grid generation (Default: 1.0)
                    (An equidistant grid is generated if aspect=1.0)
```

```
THERMALDIFF=b       Thermal diffusion effect (Default: n)
                    (Thermal diffusion is important especially for light species, e.g. H2)
DUSTYGAS=b          Dusty-gas model. The model can be used for the case WASHCOAT2=y (Default=n)
                    (The model can be used with adaptive gridding,  but the solution is less time consuming
                     and more stable when the aspect ratio option is used)
WASHCOAT=b          The effect of the washcoat is taken into account based on the reaction-
                    diffusion equations. A direct coupling is considered between the washcoat and the
                    surrounding gas-phase (Default:n)
                    (A normal simulation is performed (with a catalytic foil without washcoat), if this option is not
                    invoked)
WASHCOAT2=b         The effect of the washcoat is taken into account based on the reaction-
                    diffusion equations. An indirect coupling is considered between the washcoat and the
                    surrounding gas-phase. Indirect coupling implies that the washcoat is solved first.
                    After it reaches steady-state, it is coupled with the gas-phase at the interface. This
                    is repeated through the whole time integration of the system (Default:n)
                    (A normal simulation is performed (for a catalytic foil without washcoat), if this option is not
                    invoked)
WASHCOAT3=b         The effect of the washcoat is calculated based on the effectiveness factor approach
                    (Default:n)
                    (A normal simulation is performed (for a catalytic foil without washcoat), if this option is not
                    invoked)
FASTER=b            Decrease computational expenses by alternative memory allocating for WASHCOAT2 case (with or without DUSTYGAS option).
                    The option decreases the computational expenses enormously, but it can also rise numerical instabilities
                    in some cases (Default:n)
TRANSIENTOUT=b      Enable transient solution output
                    (Default:n, in this case just the steady-state solution output will be given)
TCONSTANT=b         Activate the option to keep the surface temperature constant through whole solution
                    (Default:n)
```

The <MOLEFRAC> tag Mole fraction of the species entering from the inlet must be included within this tag.

The <SURFACE> tag The following surface parameters must/can also be supplied by the user within this tag.

```
TSURF=d         Surface temperature [K]
ALPHA=d         Thermal conductivity of the foil or the washcoat support [W/m/K] (Default:1.44)
                (Optional, needed just for the case, TCONSTANT=n)
CURRENT=b       If CURRENT=y, current input is given to the foil or washcoat support for resistive heating. IF CURRENT=n,
                a heat flux will be supplied to the volume element, in which the foil or washcoat support are heated.
PINIT=d         (Initial) Power or current supplied to heat the catalytic surface [W/m^2 if CURRENT=n, A if CURRENT=y] (Default: 0.0)
                (Optional, needed just for the case, TCONSTANT=n)
PLAST=d         (Last) Power supplied to heat the catalytic surface [W/m^2 if CURRENT=n, A if CURRENT=y] (Default: 0.0)
                (Optional, needed just for the case, TCONSTANT=n)
PINCR=d         (Step) Power supplied to heat the catalytic surface [W/m^2 if CURRENT=n, A if CURRENT=y] (Default: 0.0)
                (Optional, needed just for the case, TCONSTANT=n)
RHOEL=d         Resistivity of the catalytic foil or washcoat support
                (Optional, optional can be supplied for the case, TCONSTANT=n and CURRENT=y)
FOILDIAM=d      Diameter of the catalytic foil or the washcoat support [m]  (Default: 0.05)
                (Optional, needed just for the case, TCONSTANT=n and CURRENT=y)
RFOI=d          Thickness of the catalytic foil or washcoat support [m] (Default: 0.003)
                (Optional, needed just for the case, TCONSTANT=n)
TRAD=d          Radiation temperature to the wall [K] (Default: 300)
                (Optional, needed just for the case, TCONSTANT=n)
EPS=d           Emissivity of the catalytic foil or the washcoat support (Default: 0.77)
                (Optional, needed just for the case, TCONSTANT=n)
EPSWC=d         Emissivity of the washcoat (Default: 0.77)
                (Optional, needed if reaction-diffusion equations or effectiveness factor approach is used, and TCONSTANT=n)
HCFOIL=d        Heat capacity of the catalytic foil or washcoat support  [J/kg/K]
                (Optional, needed just for the case, TCONSTANT=n)
DFOIL=d         Density of the catalytic foil or washcoat support [kg/m^3]
                (Optional, needed just for the case, TCONSTANT=n)
HCWC=d          Heat capacity of the washcoat  [J/kg/K]
                (Optional, needed if reaction-diffusion equations or effectiveness factor approach is used, and TCONSTANT=n)
DENWC=d         Density of the washcoat [kg/m^3]
                (Optional, needed if reaction-diffusion equations or effectiveness factor approach is used, and TCONSTANT=n)
TBACK=d         Backside temperature of the catalytic foil or washcoat support [K] (Default: 300)
                (Optional, can be given for the case; TCONSTANT=n)
```

As seen in the **<SURFACE>** tag, the user can give just one exact power input with the initial power *PINIT* parameter (in this case *PLAST* and *PINCR* are not required). It is also possible to give a linear power increase to the program by supplying *PINIT*, *PLAST*, *PINCR* values.

Estimated coverage of the surface species and surface site density must also be supplied to the code. An example is given below in the ***species.inp*** input file.

```
<SPECIES>
  <GASPHASE>
    AR
    H2
    H2O
    O
    OH
    H
  </GASPHASE>
```

```
  <SURFACE mol/cm2=  2.72E-09>
    PT(s)
    H2O(s)
    H(s)
    OH(s)
    <INITIAL>
      H(s)    4.0E-01
      PT(s)   *
      H2O(s)  2.0E-01
    </INITIAL>
  </SURFACE>
</SPECIES>
```

As seen on the above example, estimated coverage of the surface species are given within the **<INITIAL>** tag. Sum of the surface coverages must add upto 1. Surface site density is given with the unit of mol/cm^2 in the **<SURFACE>** tag declaration.

11.5 Output

DETCHEM^{STAG} creates five different output files and a screen output.

11.5.1 Screen output

A short information of the current solution will be given on the screen as given below.

```
**************************************************************************
****                        DETCHEM STAGNATION                        ****
****                           H.KARADENIZ                            ****
****                     VERSION 2.3.21 ** 2012/01/30                 ****
**************************************************************************
 Calculation for:     0.00 W/m2  ,  Time:  0.00s
 Calculation for:     0.00 W/m2  ,  Time:  0.00s
 Calculation for:     0.00 W/m2  ,  Time:  0.01s
 Calculation for:     0.00 W/m2  ,  Time:  0.02s
 Calculation for:     0.00 W/m2  ,  Time:  0.04s
 Calculation for:     0.00 W/m2  ,  Time:  0.07s
      ...  ...      ...
      ...  ...            ...
      Limex finished successfully
```

In the above lines following information is given: for which power the calculation is performed (for instance, 300.0 W/m2) and current integration time of the solution). The last line gives information whether the solution is finished successfully.

11.5.2 gas_***.dat

This file is generated according to the power input. For instance, if the user gives zero power input the output file will look like **gas_ 0.00dat**. If a power increase is supplied in the ***stag.inp***, the program will also give an output for each power increase (for instance **gas_ 300.00.dat**, **gas_ 350.00.dat**). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the surface), **TEMP** (temperature at the spatial distances from the surface), **V** (scaled radial velocity at the spatial distances from the surface), **RHOVAU** (the variable ρv at the spatial distances from the surface, please note that 100 is added as a default value to the variable, the actual value is normally -100 of it), **LAMBDA** (eigenvalue of the momentum equation which will remain constant at the spatial distances from the surface) and **mole fractions** (mole fraction of each gas-phase species at the spatial distances from the surface).

11.5.3 surf_***.dat

This file is generated for the normal simulation (without washcoat effect) and the effectiveness factor approach. It is named according to the power input. For instance, if the user gives zero power input the output file will look like **surf_ 0.00dat**. If a power increase is supplied in the ***stag.inp***, the program will also give an output for each power increase (for instance **surf_ 300.00.dat**, **surf_ 350.00.dat**). The output files will consist of the following information: **TIME** (integration time of the solution), **TEMP** (temperature on the catalytic surface),**surface coverages** (coverages of the surface species)

11.5.4 wcoat_***.dat

This file is generated for the 'washcoat' and 'washcoat2' simulations. It is named according to the power input (as explained in the gas_***.dat file generation). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the washcoat support side to the gas-washcoat interface; the interface is not included), **mole fractions** (mole fraction of each gas-phase species at the spatial distances)

11.5.5 wsurf_***.dat

This file is generated for the 'washcoat' and 'washcoat2' simulations. It is named according to the power input (as explained in the gas_***.dat file generation). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the washcoat support side to the gas-washcoat interface; the interface is not included), **surface coverages** (coverages of the surface species at spatial distances)

11.5.6 wcstag_***.dat

This file is generated for the 'washcoat' simulation. It is named according to the power input (as explained in the gas_***.dat file generation). The output files will consist of the following information: **TIME** (integration time of the solution), **R** (distance from the washcoat support side to the inlet at the gas-phase; the gas-washcoat interface and the gas phase is consequently included), **mole fractions** (mole fraction of each gas-phase species at the spatial distances). The spatial distances are printed in 'mm' because of the low thickness of the washcoat. An illustration of the output follows.

```
TITLE = "STAG_WCOAT_AND_GASPHASE"
VARIABLES=
"                TIME"  "                   R"  "            CO      "  "            CO2     "
 0.100000000000000E+03   0.000000000000000E+00   0.486946019131708E-03   0.722022776070702E-01
 0.100000000000000E+03   0.219975998217398E-01   0.486946019131708E-03   0.722022776070702E-01
 0.100000000000000E+03   0.429476948900635E-01   0.486947118172136E-03   0.722022762284908E-01
 0.100000000000000E+03   0.629001663837050E-01   0.487295820737408E-03   0.722018319160494E-01
 0.100000000000000E+03   0.819025201871732E-01   0.587980625221570E-03   0.720735682409933E-01
 0.100000000000000E+03   0.100000000000000E+00   0.257785466728995E-01   0.399837172511616E-01
 0.100000000000000E+03   0.914846147153505E+01   0.558739720682844E-01   0.835443519329686E-03
 0.100000000000000E+03   0.186493460166468E+02   0.566910726668075E-01   0.676735726010959E-05
 0.100000000000000E+03   0.286252747890142E+02   0.566999270155862E-01   0.413898503389330E-07
 0.100000000000000E+03   0.391000000000000E+02   0.566999997202967E-01   0.118393880838135E-09
```

The washcoat thickness is taken 100μ m for the above example. 0 mm indicates there the washcoat support side, 0.1 mm indicates the gas-washcoat interface and 39.1 mm indicates the gas-inlet side.

11.6 Examples

There is one example supplied along with the installation. The example directory contains the following files: ***stag.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism ***gas_test***, a surface mechanism ***surf_test***,***mech.inp***, ***thermdata***, ***moldata*** and a script file ***go***.

11.7 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

11.8 Running the tool

The example directory is supplied with a ***go*** script which can be used to run the tool just by typing ***go*** in the command prompt, or by any of the appropriate methods explained in the installation guide.

Chapter 12

DETCHEM$^{\text{PLUG}}$

12.1 Introduction

The plug code simulates the behavior of plug flow chemical reactors. The code is designed for the non-dispersive one dimensional flow of chemically reacting ideal gas mixture. The following sections describe the DETCHEM$^{\text{PLUG}}$ code that can be used to analyze the plug flow reactor for applications involving gases.

12.1.1 Governing Equations

For setting up the above equations it is assumed that (a) there is no variation in the transverse direction, and (b) axial diffusion of any quantity is negligible relative to the corresponding convective term.

$$\frac{d(\rho u A_c)}{dz} = A_s \sum_{k=1}^{kg} \dot{s_k}\, M_k \tag{12.1}$$

$$\rho u A_c \frac{dY_k}{dz} + Y_k A_s \sum_{k=1}^{kg} \dot{s_k}\, M_k = M_k (A_s\, \dot{s_k} + A_c\, \dot{\omega_k}) \tag{12.2}$$

$$\rho u A_c \frac{d(C_p T)}{dz} + \sum_{k=1}^{kg} \dot{\omega_k}\, h_k M_k A_c + \sum_{k=1}^{kg} \dot{s_k}\, h_k M_k A_s = U A_s (T_w - T) \tag{12.3}$$

$$PM = \rho R T \tag{12.4}$$

The above listed represents the total continuity, species continuity, energy, and the equation of state respectively. In-addition to these equation, since the residence time of gas is often a quantity of interest, a differential equation which computes it automatically is also included. This is simply

$$\frac{d\tau}{dz} = \frac{1}{u} \tag{12.5}$$

To improve the accuracy mass-transfer coefficients can be used additionally. They approximate the "resistance" to species mass transport between the mean composition and the composition at the reacting channel surface. The mass-transfer coefficients are defined by the following relationship

$$\dot{s_k}\, M_k = h_k (\rho_s Y_{k,s} - \rho_{k,m}) \tag{12.6}$$

where h_k is the mass-transfer coefficient, which can be represented in non-dimensional form using the species *Sherwood* numbers

$$Sh = \frac{h_k d}{D_{im}} \tag{12.7}$$

The *Sherwood* numbers are calculated locally using standard correlations. Since the heat and mass transfer phenomena in reacting flow is still an area of active research, the correlation used here represents the best efforts to-date.

12.1.2 Mass and Heat transfer coefficients

For the case of constant wall temperature the Nussult number is defined as

$$Nu_T = 3.657 + 8.827\left(\frac{1000}{Gz}\right)^{-0.545} exp\left(-\frac{48.2}{Gz}\right) \tag{12.8}$$

for the case of constant heat flux

$$Nu_H = 4.364 + 13.18\left(\frac{1000}{Gz}\right)^{-0.524} exp\left(-\frac{60.2}{Gz}\right) \tag{12.9}$$

Where Gz is the Graetz number for heat transfer. In the case of mass transfer the Sherwood numbers are expressed by the same equation with the Gz replaced by the Greatz number for mass transfer. Since the Greatz number is calculated as a function of reactor position, the mass and heat transfer coefficient calculated in DETCHEM^{PLUG}are the local values.

12.1.3 Solution

In DETCHEM^{PLUG} the standard implicit code LIMEX is used to solve the coupled governing equations. (Instead of LIMEX, DASPK3 can also be used, but for problems involving mass transfer coefficient, use of DASPK3 is not advised) DETCHEM^{PLUG} offers various options to solve the problem. The temperature solution offers **Isothermal**,**Non-isothermal, Adiabatic** and **Non-adiabatic** conditions. In addition to these, the axial temperature profile can be specified by a subroutine. In **Isothermal** as well as in the **Axial temperature** profile case the energy equation is not solved. The species transport can be modeled for **Gas-phase** and the **Surface** reactions. As a special case, the surface reactions can be specified to take place on **Washcoat**.
Though DETCHEM^{PLUG} is essentially a one dimensional model, it supports a number of geometrical configuration (The mass and heat transfer coefficients are specific to the flow geometry). The configurations supported currently are **cylindrical, cylindrical with gradually increasing cross-sectional area, triangular, square, hexagonal** and a **sinusoidal** geometry.

12.2 User Input

Before running DETCHEM^{PLUG}the user must prepare an input file ***plug.inp***. An example is shown below.

```
{include species.inp}      # Species input file
{include mech.inp}         # Mechanism input file
{include detchem.inp}      # optional
{include washcoat.inp}     # if washcoat reaction is desired

<SURFACE-MODEL>
  <CHEMSURF>
    hini=1.d-10
    time=1.d-10
  </CHEMSURF>

  Fcatgeo=1

</SURFACE-MODEL>


<PFR>                     # opening tag for plug specific data
  <GEOMETRY>               # opening tag for geometry specific data
   reactor=cylindrical     #                    factor stands for the type of geometry
   <SECTION>               # opening tag for cascading reactors
      rin=0.025             # radius at reactor inlet
      rout=0.025
      length=1.0           # length of the zone
      tw=1200              # wall temperature of the reactor section
   </SECTION>              # closing tag for cascade
  </GEOMETRY>              # closing tag for geometry
  <SOLVER>
    ini_step=1e-5         # minimum stepsize allowed for integration
    max-h=0.1              # maximum stepsize allowed for integration
    <TOLERANCE>            # opening tag for convergence tolerance
        s_aTol=1e-20
     s_rTol=1e-4
     t_aTol=1e-20
        t_rTol=1e-4
    </TOLERANCE>            # closing tag for tolerances
```

```
  </SOLVER>               # closing tag for solver
  <OUTPUT>                # opening tag for output options
    file_num=6               # integer number to be appended at the end of output file names
    title=noo2            # required title in the output file
    monitor=yes             # option to monitor the solution progress
    species=y             # mass or mole output option for the
  </OUTPUT>
</PFR>
```

The file ***plug.inp*** holds all the parameters (which are grouped into different sections) required to setup the problem. The file must start with the include statements for ***species.inp***,***mech.inp***, and ***washcoat.inp*** (if washcoat reaction is desired). Alternatively the pre-processed input file ***detchem.inp*** can be included.
The following table specifies the different geometries that can be used with mass and heat transfer coefficients.

```
Factor   Shape
1        regular cylinder
2        circular cylinder with steadily varying c/s area
3        square
4        equilateral triangle
5        hexagonal
6        sinusoidal
```

In the input file when factor is given the value 1, then plug assumes a regular cylinder and *rout* will be set equal to *rin*. The rout option in the cylindrical tab is effective only when factor is given the value 2. For any value of factor greater than 2, the dimension given in the OTHERS tab will be used to calculate the surface area and cross sectional area.

If *htceff* is supplied in process parameters, then the user supplied value will be used while solving the energy equation. On the other hand the user can comment the tag *htceff* then the heat transfer coefficient will be calculated using correlations.

If axial temperature profile is opted in operating conditions then a subroutine named *PFR_- TempProfile_Usr* must be supplied. A pseudo code for subroutine TempProfile_Usr is supplied along with the installation. Note that the point temperatures must be stored in the variable *tp*. *z* is the position along the length of the reactor. The user can modify the code and recompile the files using Makefile.

The convergence rate can vary from case to case. In any case if the code fails during the run, then the user is advised to change the tolerance parameters. There are no strict guidelines for setting the tolerance.

12.2.1 The <GEOMETRY> tag

The important point to be noticed in the **<GEOMETRY>** tag is the option **factor**. The values assigned for the factor decides the reactor shape under consideration. The values that can Ga assigned to **factor** is explained above.

12.2.2 The <SECTION> tag

By this feature the reactor can be divided into a number of sections and each section can be assigned with different process/boundary conditions. A total of 20 sections are supported. If there is only one **<SECTION>** tag, then the reactor is assumed to be of single section with the length of the reactor equal to the length of the section. If more than one **<SECTION>** is specified, then the length of reactor is equal to the total length of the sections. And the simulation of each reactor section will be carried out according to the conditions specified under each **<SECTION>** tag.

12.2.3 The <SOLVER> tag

This tag holds the option for choosing the solver and the tolerance levels that can be assigned for different solution variables. In any case if the solver is unable to finish the calculation, it is advisable to relax the tolerance criteria as well as the **maxh**, especially when calculations are performed with mass transfer option.

12.2.4 The <SUMMARY> and <SUM> tag

In the plug input file, there is an option to produce summary of the simulation, which is nothing but the exit conditions of the chemical species, In case if there are large number of species involved in the problem, and the user wants to know the exit conditions of certain specific species, then the name of those species can be listed under the **<SUMMARY>** tag and hence the exit conditions will be written in the file ***summary.dat*** .

If the user wants to find the sum of mass/mole fraction of certain species up on exit, then the **<SUM>** tag can be used, which will produce the sum of the mass/mole fraction of the species in the output file ***summary.dat***.

12.3 Output

Depending on the options set in the input file DETCHEM^{PLUG} produces several file outputs. All the output files are in Tecplot format.

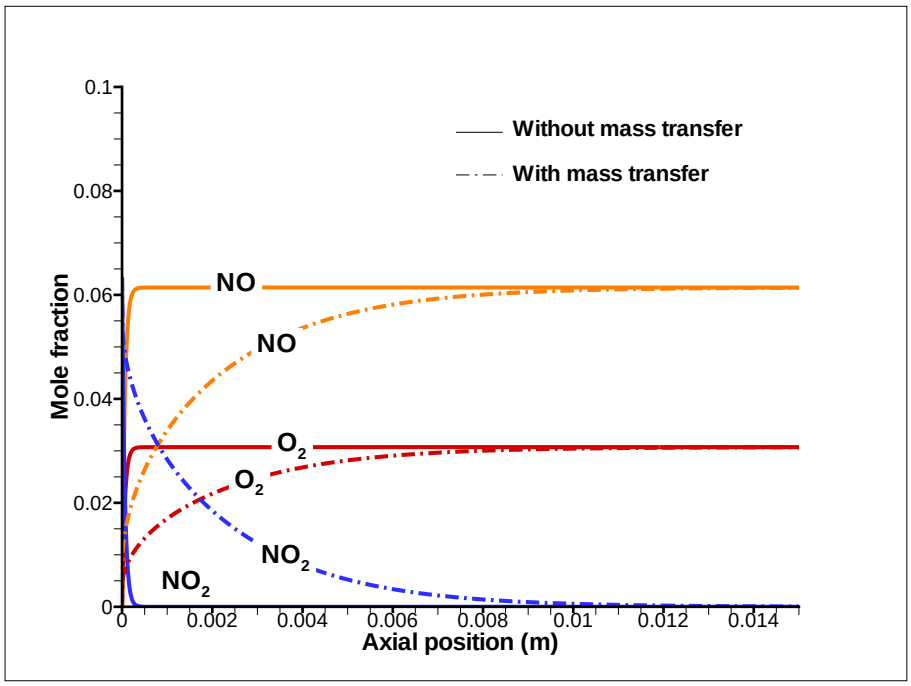
12.3.1 Screen output

Other than the file output, DETCHEM^{PLUG} produces screen output depending on the monitor option specified in the input file. With *monitor=1*, the screen output reports the iteration count and the temperature. With *monitor=2*, it reports the iteration count, temperature and the species mass fractions. An example of screen output is shown below.

```
**********************************************************************
****                          DETCHEM PLUG                          ****
****    Vinod M Janardhanan, Steffen Tischer & Olaf Deutschmann     ****
****                    VERSION 2.0 ** 01,01,2004                   ****
****                      ---------------                           ****
**********************************************************************
 Iterations
       1    0.00000  700.00000
       2    0.00000  698.27944
       3    0.00000  700.85369
 .    .......  .........
 .    .......  .........
      56    0.04867 1200.01246
      57    0.04967 1200.01246
      58    0.05000 1200.01199
 Elapsed Execution time  :   0.180971995
 User time  :   0.175972998
 System time  :   0.00499900011
```

12.3.2 *outg***.plt*

The output file ***outg***.plt*** lists velocity, residence times, temperature density and the gas-species mass or mole fractions, depending on whether the species tag in **<OUTPUT>** element is supplied with *mass* or *mole* option.



12.3.3 *outs***.plt*

If *coverage* option is chosen in then the file ***outs***.plt*** will be produced,which lists the surface coverages along the axial position of the reactor.

12.3.4 *trans***.plt*

If *transfer-coeff* option is chosen then the file ***trans***.plt*** will be produced which will list the mass and heat transfer coefficient calculated using the correlations.

12.3.5 *summary.dat*

The file ***summary.dat*** contains the exit temperature and composition of the species listed in the **<SUMMARY>** tag. If the **<SUM>** tag is also used then the file ***summary.dat*** also contains the sum of the mass/mole fraction of the species listed in the **<SUM>** tag.

12.4 Examples

There are a few examples supplied along with the installation. Each example directory contains the following files: ***plug.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism file, a surface mechanism file, ***thermdata***, and ***moldata***. The optional washcoat input has been saved in a separate file ***washcoat.inp***.

12.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.In case of problems, adjusting the solver and tolerance parameters may improve the performance.

12.6 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***./plug***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/plug*** and call ***./plug***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 13

DETCHEM^{PBR}

13.1 Introduction

Today, since the finding of solid catalysts, most commercial, at the industry level, catalytic processes are being carried out in a fixed bed reactor. The catalytic fixed bed reactor plays an important role in the synthesis of large scale basic chemicals and intermediates. A fixed bed reactor is a vertical vessel packed with compact, transfixed catalyst pellets. This catalyst bed act as a porous media at the macroscopic level. Compared to other reactors types or designs utilizing heterogeneous catalysts, the fixed packed bed reactors are preferred because of simpler technology and ease of operation.
There are several ways by which one can represent a packed bed reactor mathematically and hence to solve it numerically. The most commonly found models are 1) 1D-homogeneous 2) 2D-homogeneous model 3) 1D-heterogeneous model 4)2D-heterogeneous model.DETCHEM^{PBR} is basically a 1D-heterogeneous model. The 1D-heterogeneous model assumes that there is no radial variations in the flow properties. Most of the industrial reactors runs under adiabatic conditions. The model presented here can be used for any operating conditions. Furthermore, the integrated rate can be computed with present model. The model do not takes care of axial diffusion.

13.1.1 Governing equations

For setting up the above equations it is assumed that (a) there is no variation in the transverse direction, and (b) axial diffusion of any quantity is negligible relative to the corresponding convective term.

The equation 13.1 is the continuity equation,

$$\frac{d(\rho u)}{dz} = a_v \sum_{i \in S_g} \dot{s_i}\, M_i \tag{13.1}$$

Where,
ρ - fluid phase density, z - axial coordinate, u - superficial velocity, a_v - area to volume ratio, $\dot{s_i}$ - surface phase reaction rate, M_i - molar mass of species i

The equation 13.2 stands for the gas phase species balance,

$$\rho u \frac{dY_i}{dz} + Y_i a_v \sum_{i \in S_g} \dot{s_i}\, M_i = M_i (a_v\, \dot{s_i} + \dot{\omega_i}\, \epsilon) \tag{13.2}$$

Where,
Y - mass fraction of species i, ω - fluid phase reaction rate, ϵ - bed porosity

The equation 13.3 stand for the gas phase temperature balance taking solid surface into the account,

$$\rho u \frac{d(C_p T_f)}{dz} + h_{fs} a_v (T_s - T_f) = \frac{4}{d_t} U (T_w - T_f) \tag{13.3}$$

Where,
C_p - heat capacity, h_{fs} - fluid solid heat transfer coefficient, T_s - solid phase temperature, T_f - fluid phase temperature, d_t - reactor tube diameter, U - overall heat transfer coefficient, T_w - reactor wall temperature

For pressure drop calculation in packed bed reactor, equation 13.4 is used,

$$\frac{dp}{dz} = f\frac{(\rho u^2)}{d_p} \tag{13.4}$$

Where,
p - pressure, d_p - diameter of particle (packing material), f - friction factor for packed bed

And equation 13.5 is used as equation of states,

$$pM = \rho R T_f \tag{13.5}$$

Where,
R - universal gas constant

The above listed equations are solved using LIMEX solver. In-addition to these equations, since the residence time of gas is often a quantity of interest, a differential equation which computes it automatically is also included. This is simply equation 13.6,

$$\frac{d\tau}{dz} = \frac{1}{u} \tag{13.6}$$

Where,
τ - residence time

To improve the accuracy mass-transfer coefficients are used to approximate the "resistance" to species mass transport between the mean composition and the composition at the reacting catalyst surface. The mass-transfer coefficients are defined by the following relationship equation 13.7,

$$\dot{s_i}\ M_i = k_{fsi}(\rho_{i,s}Y_{i,s} - \rho_{i,f}) \tag{13.7}$$

Where,
k_{fs_i} - fluid solid mass transfer coefficient for species i, $\rho_{i,s}$ - solid phase density for species i, ρ_{if} - fluid phase density for species i, $Y_{i,s}$ - mass fraction of solid phase species i

The mass transfer coefficients k_{fs} from equation 13.7 can be represented in non-dimensional form using the species Sherwood numbers from equation 13.8,

$$\mathrm{Sh} = \frac{k_{fs}d_p}{D_i} \tag{13.8}$$

Where,
D_i - effective diffusion for species i, Sh - Sherwood number

The Sherwood numbers are calculated using different standard correlations. Since the heat and mass transfer phenomena in reacting flow is still an area of active research, the correlation used here represents the best efforts to-date. For more details check section 13.9.
Inter-phase heat transfer between the solid and the fluid phase inside the reactor leads to difference in temperature between the solid and fluid phase. Therefore the surface reaction rate has to be evaluated at the prevailing surface temperature of the catalyst. The heat transfer inside the reactor reads as equation 13.9.

$$h_{fs}a_v(T_s - T_f) = \epsilon \sum_{i \in S_g} \dot{\omega_i}\ h_i M_i + a_v \sum_{i \in S_g} \dot{s_i}\ h_i M_i \tag{13.9}$$

Where,
h_{fs} - fluid solid heat transfer coefficient,

The heat transfer coefficients from equation 13.9 can be represented in non-dimensional form using the species Nusselt numbers and calculated using different standard correlation. For more details check section 13.9.

$$\mathrm{Nu} = \frac{h_{fs}d_p}{k_f} \tag{13.10}$$

Where,
k_f - fluid phase thermal conductivity, Nu - Nusselt number

13.1.2 List of variables

Latin symbols

a_v	area to volume ratio	m^2/m^3
c	molar concentration	mol/m^3
c_p	heat capacity at constant pressure	$J/(kg.k)$
d_p	particle diameter	m
d_t	reactor tube diameter	m
D	effective diffusion coefficient	m^2/s
f	friction factor	$-$
h	enthalpy	J/kg
h_{fs}	fluid solid heat transfer coefficient	$W/m^2/K$
k_{fs}	fluid solid mass transfer coefficient	m/s
M	molar mass	$kg/kmol$
Nu	Nusselt number	$-$
p	Pressure	Pa
R	universal gas constant	$J/mol/K$
$\dot{s}$	rate of surface phase reaction	$mol/m^2/s$
Sh	Sherwood number	$-$
t	time	s
T_f	fluid phase temperature	K
T_w	reactor wall temperature	K
T_s	solid phase temperature	K
u	superficial velocity	m/s
U	overall heat transfer coefficient	$W/m^2/K$
X	mole fraction	$-$
Y	mass fraction	$-$
z	reactor length	m

Greek symbols

λ	thermal conductivity	$W/(m.K)$
μ	dynamic viscosity	$Pa.s$
ν	kinematic viscosity	m/s^2
ρ	density	kg/m^3
ϵ	porosity of bed	$-$
τ	residence time	s
$\dot{\omega}$	rate of fluid phase reaction	$kg/(m^3.s)$

13.2 Solution

In DETCHEM^{PBR} the standard implicit code LIMEX is used to solve the coupled governing equations. DETCHEM^{PBR} offers various options to solve the problem. The temperature solution offers **Isothermal**, **Non-isothermal**, and **Adiabatic** along with option of **Mass transfer**. The species transport can be modeled for **Gas-phase** and the **Surface-phase** reactions.

13.3 User input

Before running DETCHEM^{PBR} the user must prepare an input file ***pbr.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
    hini=1.d-10
    time=10.
  </CHEMSURF>
</SURFACE-MODEL>
```

```
<PBR>
  <GEOMETRY>
reactor = cylindrical
radius = 0.025
<SECTION>
    length = 0.05
    wall_temp = 1200
    particle_geometry = sphere
        particle_diameter = 3.79d-03
        porosity = 0.37
        particle_therm_conductivity = 0.3
        area_to_volume_ratio = 400
        gas_chem = n
        surf_chem = y
        Fcat/geo = 1.0
</SECTION>
<WALL>
    wall_thickness = 0.01
    thermal_conductivity = 45
    heat_trans_coeff_outside = 0
</WALL>
  </GEOMETRY>
  <INITIAL>
    T = 500
    P = 1.0E5
    velocity = 0.1
    <MASSFRAC>
       NO2    0.1
       N2     *
    </MASSFRAC>
  </INITIAL>
  <PROCESS>
    isothermal = n
    adiabatic = n
    mass_transfer = n
  </PROCESS>
  <SOLVER>
    h=1e-20
    atol = 1e-8
    rtol = 1e-8
  </SOLVER>
  <OUTPUT>
    fileNr=1
    write_molefractions = y
    write_massfractions = y
    write_coverages = y
  </OUTPUT>
</PBR>
```

The file pbr.inp starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files thermdata and moldata must be located in the executing directory. Next, an optional washcoat definition may appear in the input file. If washcoat models shall be applied, the user must supply this information. However, by including the washcoat definition the model is not activated automatically. Use the washcoat member in the <PROCESS> tag to activate the model. The options within the <PBR> tag are as follows.

<PBR>	opening tag for PBR
<GEOMETRY>	reactor geometry parameters
<INITIAL>	initial reactor operating parameters
<PROCESS>	process conditions for reactor
<SOLVER>	DAE solver parameters
<OUTPUT>	output options
<REACTION-FLOW>	integrated reaction rate calculation
< /PBR>	closing tag for PBR

13.3.1 The <GEOMETRY> tag

The **<GEOMETRY>** tag can be used to define the shape and size of the reactor, basically diameter only. The total length can be defined withing **<SECTION>** tag (details in 13.3.2). Also the **<GEOMETRY>** is extended to consider **<WALL>**(details in 13.3.3). The options withing **<GEOMETRY>** tag are :

<**GEOMETRY**>	opening tag for geometry
reactor=*s*	Geometry of the reactor(available options are cylindrical, square_duct, equi_triangle, hexagonal. default: cylindrical)
radius=*d*	radius of the defined reactor geometry
<**SECTION**>	Opening tag for section
< /**SECTION**>	closing tag for section
<**WALL**>	Opening tag for section
< /**WALL**>	closing tag for section
< /**GEOMETRY**>	closing tag for geometry

13.3.2 The <SECTION> tag

The user can divide the packed bed in to several (default: up to 25) sections. This is desired when you have different catalyst properties along the length of the packed bed, or when the wall surface reactions are happening only at certain specific part of the reactor.

<**SECTION**>	opening tag for section
length=*d*	length of the reactor section (m)
wall_temp=*d*	wall temperature of the reactor section (K)
ignition_temp=*d*	ignition temperature to initiate the reactions (K) (default: 0 K)
gas_chem=*b*	gas phase chemistry(default: yes)
surf_chem=*b*	surface phase chemistry(default: no)
particle_geometry=*s*	packing material geometry in reactor section (available options are spherical and cylindrical. default: spherical)
particle_diameter=*d*	diameter of the packing particles (m)
particle_length=*d*	length of the packing particles(in case of cylindrical) (m)
porosity=*d*	void fraction of the packing (default: using internal subroutine)
particle_therm_conductivity=*d*	thermal conductivity of packing particle (W/m/K)
radial_thermal_conductivity_solid=*d*	user defined radial thermal conductivity of packing particle (W/m/K) (default: using internal subroutine)
nusselt_number_coefficient=*d*	nusselt number coefficient (applicable only for if wall_nusselt_number_method 4 is selected in WALL section)
emissivity=*d*	emissivity (applicable only for if radial_solid_cal_method 3 or 5 is selected in WALL section)
wall_nusselt_number=*d*	wall nusselt number (default: using internal subroutine)
area_to_volume_ratio=*d*	porous area to volume ratio (/m) (default: using internal subroutine)
< /**SECTION**>	closing tag for section

13.3.3 The <WALL> tag

When it is desired to take care of the heat transfer from the outside wall, this tag must be used and the wall thickness must be given. In case if the thickness is assigned a value 0 then the conduction within the wall will not be taken care in the simulation.

<**WALL**>	opening tag for wall
wall_thickness=*d*	thickness of the reactor wall (m)
thermal_conductivity_wall=*d*	thermal conductivity of the reactor wall (W/m/K)
heat_trans_coeff_outside=*d*	outside heat transfer coefficient (W/m2/K)
overall_heat_trans_coeff=*d*	overall heat transfer coefficient (W/m2/K) (default: using internal subroutine)
wall_nusselt_number_method=*i*	method for wall nusselt number calculations (available options are 1, 2, 3 4 default: 3)
fluid_solid_transfer_cal_method=*i*	method for fluid solid heat transfer coefficient calculations (available options are 1, 2, 3 default: 2)
radial_solid_cal_method=*i*	method for radial solid heat transfer coefficient (available options are 1, 2, 3, 4, 5 default: 1)
radial_fluid_cal_method=*i*	method for radial fluid heat transfer coefficient (available options are 1, 2, 3 default: 2)
Biot_number_cal_method=*i*	method for radial solid heat transfer coefficient (available options are 1, 2, 3, 4 default: 2)
Overall_U_cal_method=*i*	method for calculation of overall heat transfer coefficient (available options are 1, 2 default: 1)
plug_wall_heat_trans_method=*i*	method for wall heat transfer coefficient under plug condition (available options are 1, 2, 3, 4 default: 4)
heat_flux=*d*	heat flux (W/m2)
< /**WALL**>	closing tag for wall

13.3.4 The <INITIAL> tag

In this section the user specifies the gas-phase properties at the inlet. Following options can be specified:

<**INITIAL**>	opening tag for initial
T=*d*	inlet temperature of the fluid stream (K)
P=*d*	inlet pressure of the fluid stream (Pa) (default: 1e5 Pa)
velocity=*d*	inlet superficial velocity of the fluid stream (m/s)
<**MOLEFRAC**>	opening tag for inlet mole fractions
< /**MOLEFRAC**>	closing inlet for mole fractions
<**MASSFRAC**>	opening tag for inlet mass fractions
< /**MOLEFRAC**>	closing tag for inlet mole fractions
< /**INITIAL**>	closing tag for initial

Note that the **velocity** is the actual inlet velocity at the given temperature **T** and pressure **P**.

13.3.5 The <PROCESS> tag

Within this section the user need to specify the operating conditions for the packed bed reactor. The is allowed to select between **isothermal**, **adiabatic** and **mass transfer**. For **non-isothermal** conditions user need to say no (**n**) to **isothermal** and **adiabatic** condition.

<**PROCESS**>	opening tag for process
isothermal=*b*	isothermal conditions(default: yes)
adiabatic=*b*	adiabatic conditions(default: no)
mass_transfer=*b*	mass_transfer conditions(default: no)
cov_relax=*d*	under relaxation factor for surface coverages (default: 1.d0)
pseudo_homogenous=*b*	pseudo homogenous reaction, (If user don't want species and temperature balance to take porosity into account) (default: no)
T_solid_balance=*b*	solving solid phase temperture balance (default: no)
continuity_eq=*b*	solving continuity equation (default: yes)
axial_T_profile=*b*	use temperature profile for calculation (default: no)
< /**PROCESS**>	closing tag for process

13.3.6 The <T-PROFILE> tag

In order to define a temperature profile along the reactor wall for a single run, use the tag **<T-PROFILE>** and is used to define a piecewise linear temperature profile. Specify at least two pairs of values in that case. The tag requires an arbitrary number of pairs of values in the form (position, temperature), e. g.

```
<T-PROFILE>
    0.000    1000    # z-position [m], temperature [K]
    0.001    1200
    0.002    1300
    0.003    1400
</T-PROFILE>
```

13.3.7 The <SOLVER> tag

In this section the user can define solver specific options:

<**SOLVER**>	opening tag for solver
atol=*d*	absolute tolerance (default: 1e-6)
rtol=*d*	relative tolerance (default: 1e-6)
h=*d*	initial integration step size (m) (default: 1e-10)
hmax=*d*	maximum step size (m) (default: none)
max_iter=*d*	maximum number of iterations (default: unlimited)
IPos=*i*	values of the corresponding solution component will be checked to be nonnegative if IPos = 1 (available options are 0, 1 default: 1)
< /**SOLVER**>	closing tag for solver

The members h, and hmax define the step size of the integration process. If the integration process fails, adjusting these values along with atol and rtol values may help.The parameter max_iter defines a maximum number of integration steps. By limiting the number of iterations, the user can avoid packedbed simulations that need too much time to be solved. There are no strict guidelines for setting the tolerance.

13.3.8 The OUTPUT tag

The particular tag contains many options for writing the output file. Depending on the options set within the tag, different variables will be wrote in the output files.

< /**OUTPUT**>	opening tag for output
fileNr=*d*	file number for output, e.g. PBR001.plt, PBR-molefracs001.plt, PBR-massfracs001.plt, PBR-coverages001.plt, PBR-concentrations001.plt (default: 1)
dz_out=*d*	minimum step size for output (m) (default: none)
write_molefractions=*b*	write mole fractions, e.g. PBR-molefracs***.plt (default: yes)
write_massfractions=*b*	write mass fractions, e.g. PBR-massfracs***.plt (default: no)
write_coverages=*b*	write coverages, e.g. PBR-coverages***.plt (default: no)
write_concentrations=*b*	write gas phase concentrations, e.g. PBR-concentrations***.plt (default: no)
write_transfer_coeff=*b*	write transfer coefficient (default: no)
< /**OUTPUT**>	closing tag for output

13.3.9 The REACTION-FLOW tag

This tag provide access to the integrated reaction rate calculation, which can be further use for reaction path analysis. If the option is used, two additional output files will be generate, PBR-ROP-xxxx.plt and PBR-integrated-rflow-xxxx.csv. NOTE: 1. PBR-ROP: This file contains the local net rate of production for species, for gas phase species unit is mol/m^3/s and for surface phase species unit is mol/m^2/s. 2. PBR-integrated-rflow : This file contains the integrated reaction rate calculated at the end of reactor. On can adjust reactor length to achieve desired result against residence time, unit of gas phase reaction is mol/m^3 and unit of surface reaction is mol/m^2.

<**REACTION-FLOW**>	opening tag for reaction flow
gas_flow_analysis=*b*	gas phase integrated reaction rate (default: no)
surf_flow_analysis=*b*	surface phase integrated reaction rate (default: no)
< /**REACTION-FLOW**>	closing tag for reaction flow

13.4 Output

Depending on the setting in the ***pbr.inp*** file, DETCHEM^{PBR} produces screen output and file output.

13.4.1 Screen output

On screen the integration progress is monitored. The values in the two columns displayed stand for time and temperature.

```
**************************************************************************
****                          DETCHEM PBR                             ****
****            A. Shirsath, S. Tischer, O. Deutschmann               ****
****                 VERSION 2.7.1 ** 2018/08/01                      ****
**************************************************************************

    0.00E+00     500.0
    1.82E-04     653.9
    7.50E-03     999.7
    ...
    5.00E-02    1199.6
 PBR finish successfully
```

13.4.2 *PBR***.plt*

DETCHEM^{PBR} produces few different output files like, ***PBR-molefracs***.plt***, ***PBR-massfracs***.plt***, ***PBR-coverages***.plt***, and ***PBR-concentrations***.plt***. The output file is in TECPLOT format. TECPLOT will create a line graph upon loading the file ***PBR***.plt***. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the file are: axial length(z, 'm'), gas phase temperature(T_gas, 'K'), surface phase temperature(T_surface), pressure(P, 'Pa'), residence time(Tau, 's'), velocity('u', 'm/s'), average gas density(rho, 'kg/m^3'), mass flow rate(mdot, 'kg/s'), volumetric flow rate(vdot, 'm^3/s'), molar flow rate(fdot, 'mol/s'), average molecular weight(Meff, 'kg/mol'), species concentration(prefixed with c, 'mol/m^3'), species mole fractions (prefixed with X), and species mass fractions (prefixed with Y). Under write heat transfer coefficient option, additional variable in file are: overall heat transfer coefficient(overall_U, 'W/m^2/K'), fluid solid heat transfer coefficient(h_fs, 'W/m^2/K'), fluid solid mass transfer coefficient(prefixed with k_fs, 'm/s'), radial thermal conductivity of solid(k_rs, 'W/m/K'), radial thermal conductivity of gas(k_rf, 'W/m/K'), effective radial thermal conductivity (k_r, 'W/m/K'), and wall heat transfer coefficient (h_w, 'W/m^2/K').

13.5 Examples

There are a few examples supplied along with the installation. Each example directory contains the following files: ***pbr.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism file, a surface mechanism file, ***thermdata***, and ***moldata***.
The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

13.6 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.
When setting up a new case, we also suggest to start as simple as possible and to become more complex gradually. Check if the geometry settings and wall temperature profile does not contain errors by running simulations without chemistry first. Then add surface and gas-phase mechanisms one after another. Always check for error messages and watch the solution process to detect convergence problems. In case of problems, adjusting the solver and tolerance parameters may improve the performance.

13.7 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable DETCHEM_DIR.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***pbr***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/pbr*** and call ***./pbr***.

The latter option does not require setting of system variables and is therefore less system specific.

13.8 PBR Transient

The DETCHEM$^{\text{PBR}}$ code has been further account for to simulate in transient mode. The main purpose of the simulations with DETCHEM$^{\text{PBR_TRANSIENT}}$ is the storage process. For this simulation, user must provide one surface type for which transient concentration should get calculated. For the storage medium we must assume that the rate of reaction is slow enough to be decoupled from the flow field. That is, the time scale for storage reactions must be larger than the residence time of the gases inside the reactor. For example, storage of C (solid carbon) on catalyst surface. During such a case, a sub-set of surface reactions can no longer be considered in quasi steady state.The following equation is solved for transient storage species

$$\frac{\mathrm{d}c_i^{\text{storage}}}{\mathrm{d}t} = \dot{s}_i(c^{\text{gas}}, c^{\text{surf}}, c^{\text{storage}}) \tag{13.11}$$

An example is provide with package to show how to setup storage system. Lets say, a system with solid carbon (Cde) on catalytic surface. You can define surface type in species list followed by use of storage tag in surface of mechanism tag as follows.

```
<MECHANISM>
  ...
  <SURFACE name="storage">
    <GLOBAL>
  CH4 + (C) > Cde + 2 H2 + (C)
  <ARRHENIUS>
    A/SIunits = 5e-4
    (C)         0
  </ARRHENIUS>
</GLOBAL>
  </SURFACE>
</MECHANISM>
```

Further more, following options can be used for the transient simulation. The options can be used right after species and mechanism option.

```
{define time_step 30.}   # time step used for the transient simulation, also files will be printed at this time step
{define end_time 180.}   # end time for the transient simulation
{define n_sect 1000}     # no of points to discretized the length of reactor for given time step
```

13.8.1 The <INLET> tag

For inlet conditions it is necessary to prepare another input file. It can have any name (e. g. inlet.inp). Use this name as reference for the file member in the **<INLET>** section of the input file.

<**INLET**>	opening tag for initial
file=*s*	name of the inlet file
< /**INLET**>	closing tag for initial

The inlet file consists of two sections: **<INLETINFO>** and **<INLETDATA>**.There are two types of inlet conditions: constant and variable. An example is shown below.

```
<INLETINFO>
  T = list      # temperatures appear in list
  u = 0.8       # constant velocity
  p = 1.d5      # constant pressure
  <MOLEFRAC list>
     CH4
     O2
     N2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
```

```
#time  Temp   CH4    O2    N2
    0   900   0.0   0.2   0.8
    5   900   0.1   0.2   0.7
   60   800   0.1   0.1   0.8
  120   700   0.1   0.1   0.8
</INLETDATA>
```

In **<INLETINFO>** the user needs to specify which data is read from a list or file and which is constant. Assign one of the following options to the members **T**, **u**, and **p** for temperature, velocity and pressure, respectively:

- a *numerical value* for a constant,
- **list** for data that is listed in **<INLETDATA>**, or
- a *TECPLOT variable name* for spatially varying data in separate files.

The species composition can be given in mole (**<MOLEFRAC>**) or mass fractions (**<MASSFRAC>**). Examples of the syntax for each case are shown below.

- constant mass or mole fractions

```
<MOLEFRAC const>    # define composition as usual
   CH4   0.1
   O2    0.1
   N2    *          # * may be used for balance
</MOLEFRAC>
```

- data listed in **<INLETDATA>**

```
<MOLEFRAC list>     # give the species in the same order
   CH4              # as they appear in the list
   O2
   N2
</MOLEFRAC>
```

13.8.2 Running the tool

The user can call executable in same way as DETCHEM^PBR only change is instead of using ***pbr***, one should use ***pbr_transient***.

13.8.3 Output(*global.plt*)

The file ***global.plt*** summarizes the global inlet and global outlet composition at each time step (not only the times listed). The file is in TECPLOT format and can be used to draw line graphs of gas-phase temperature and mass fractions. However, this format also allows for simple import into other spreadsheet programs (e. g. Microsoft Excel).

The variables in this file are: time, temperature, mole fractions(prefixed with X), mass fractions(prefixed with Y), and concentration(prefixed with c, 'mol/m^3'). You can use this file for simple calculation of conversions and selectivities.

A new file ***global.plt*** is written on each execution of DETCHEM^PBR (except at postprocessing of a single file). If you want to restart an earlier calculation, save ***global.plt*** with a different name before calling the executable again.

13.9 Correlation used for heat and mass transfer coefficient

This section will cover the type of equation and available options for heat and mass transfer method in PBR.

13.9.1 Heat transfer

Fluid solid heat transfer coefficient

Method 1: Gnielinski, V. in G esellschaft, V. D. I., ed. VDI Heat Atlas. 2nd ed. 2010 edition. Berlin; New York: Springer, 2010
Method 2: Wakao, Noriaki, and SeiichirÅ Kagei. Heat and Mass Transfer in Packed Beds. Taylor and Francis, 1982 (NOTE: default method)
Method 3: Reactor Core Design of High-Temperature Gas-Cooled Reactors Part 2: Heat Transfer in Spherical Fuel Elements (June 1983)

Radial solid thermal conductivity

Method 1: Zehner, P. and Schlunder, E. U., Chemie. Ingr. Tech., 42, pp933, (1970) (NOTE: default method)
Method 2: Sphecchia. V.. Baldi. G. and Sicardi. S.. 1980. Heat transfer in-packed bdd reactors with one phase flow. Chem. Engng Commun. 4, 361-380.
Method 3: Bauer and E. U. Schlunder, Effective radial thermal conductivity of packings in gas flow. Part II Thermal conductivity of the packing fraction without gas flow, Int. Chem. Eng 18, 189-204 (1978). (NOTE: Includes radiation terms)
Method 4: D. Kunii and J. M. Smith, Heat transfer characteristics of porous rocks, AlChE J. 6,71-78 (1960).
Method 5: D. Kunii and J. M. Smith, Heat transfer characteristics of porous rocks, AlChE J. 6,71-78 (1960). (NOTE: Includes radiation terms)

Radial fluid thermal conductivity

Method 1: Yagi, S. and Wakao, N. (1959), Heat and mass transfer from wall to fluid in packed beds. AIChE J., 5: 79-85
Method 2: Sphecchia. V.. Baldi. G. and Sicardi. S.. 1980. Heat transfer in-packed bdd reactors with one phase flow. Chem. Engng Commun. 4, 361-380. (NOTE: default method)
Method 3: R. Bauer, and E. U. Schlnder, Effective radial thermal conductivity of packings in gas flow. Part II. Thermal conductivity of the packing fraction without gas flow, International Chemical Engineering 18(2), (1978), p. 189204
Method 4: M. Winterberg, E. Tsotsas, A. Krischke, D. Vortmeyer, A simple and coherent set of coefficients for modelling of heat and mass transport with and without chemical reaction in tubes filled with spheres, Chemical Engineering Science, Volume 55, Issue 5, 2000

Reactor Biot number

Method 1: Melanson, M.M., and Dixon, A.G., "Solid Conduction in Low d,/d Beds of Spheres, Pellets and Rings", Int. J. Heat Mass Transfer 28, 383 (1985)
Method 2: Dixon, A.G. (1985), Thermal resistance models of packedbed effective heat transfer parameters. AIChE J., 31: 826-834 (NOTE: default method)
Method 3: Cresswell, D. L., and A. G. Dixon, Reply to Comments by Vortmeyer and Berninger on the Paper Theoretical Prediction of Effective Heat Transfer Parameters in Packed Beds (AIChE J. 25,663,1979): AIChE J., 28,511, 1982
Method 4: Cresswell, D. L., "Heat Transfer in Packed Bed Reactors", NATO ASI Series E-No. 110, 687 (1986)

13.9.2 Nusselt number

Method 1: Dixon, Anthony G., Wall and particle-shape effects on heat transfer in packed beds, Chemical Engineering Communications, 71:1, (1988) 217-237
Method 2: Dixon, A.G. (2012), Fixed bed catalytic reactor modellingthe radial heat transfer problem. Can. J. Chem. Eng., 90: 507-527
Method 3: Jurtz N, Srivastava U, Moghaddam AA, Kraume M. Particle-Resolved Computational Fluid Dynamics as the Basis for Thermal Process Intensification of Fixed-Bed Reactors on Multiple Scales. Energies. 2021; 14(10):2913 (NOTE: default method)
Method 4: Dixon, Anthony G. "Local Structure Effects on Heat Transfer in Very Low Tube-to-Particle Diameter Ratio Fixed Beds of Spheres." Industrial and Engineering Chemistry Research 60.27 (2021): 9777-9786

Overall heat transfer coefficient

Method 1: Dixon, Anthony G., An improved equation for the overall heat transfer coefficient in packed beds., Chemical Engineering and Processing: Process Intensification 35, no. 5 (1996): 323-331 (NOTE: default method)
Method 2: Dixon, Anthony G., An improved equation for the overall heat transfer coefficient in packed beds., Chemical Engineering and Processing: Process Intensification 35, no. 5 (1996): 323-331 (U' method)

13.9.3 Wall heat transfer under plug conditions

Method 1: Tronconi, E., and Forzatti, P. (1992). Adequacy of lumped parameter models for SCR reactors with monolith structure. AIChE Journal, 38(2), 201210
Method 2: Gnielinski, V. in G esellschaft, V. D. I., ed. VDI Heat Atlas. 2nd ed. 2010 edition. Berlin; New York: Springer, 2010
Method 3: Stephan, K., Druckabfall bei nicht ausgebildeter Laminarstromung in Rohren und in ebenen Spahen. (1959). 12, 773778
Method 4: Churchill, S. W., and Ozoe, H., 1973, Correlations for Laminar Forced Convection With Uniform Heating in Flow Over a Plate and in Developing and Folly Developed Flow in a Tube, ASME J. Heat Transfer, 95(1), pp. 7884 (NOTE: default method)

13.9.4 Mass transfer

Correlations are the same as for solid-fluid heat transfer coefficient. Only Pr is replaced by Sc and Nu is replaced by Sh.

Chapter 14

DETCHEM^{PFR}

14.1 Introduction

The plug code simulates the behavior of plug flow chemical reactors. The code is designed for the non-dispersive one dimensional flow of chemically reacting ideal gas mixture. The following sections describe the DETCHEM^{PFR} code that can be used to analyze the plug flow reactor for applications involving gases.
Basically, a plug flow reactor demonstrates a physically non changing system with area of crossection A. The gas is completely homogenous in the direction perpendicular to the flow. All the states of the gas are allowed to change in axial direction z. Furthermore, the integrated rate can be computed with present model. However, it is assumed that all diffusion processes are negligible hence can be neglected.

14.1.1 Governing equations

For setting up the above equations it is assumed that (a) there is no variation in the transverse direction, and (b) axial diffusion of any quantity is negligible relative to the corresponding convective term.

The equation 14.1 is the continuity equation,

$$\frac{d(\rho u A)}{dz} = P' \sum_{i \in S_g} \dot{s}_i \, M_i \tag{14.1}$$

Where,
ρ - fluid phase density, z - axial coordinate, u - superficial velocity, P' - chemically active channel perimeter, A - cross-sectional area, $\dot{s}_i$ - surface phase reaction rate, M_i - molar mass of species i

The equation 14.2 stands for the gas phase species balance,

$$\rho u \frac{dY_i}{dz} + Y_i P' \sum_{i \in S_g} \dot{s}_i \, M_i = M_i (P' \, \dot{s}_i + \dot{\omega}_i) \tag{14.2}$$

Where,
Y - mass fraction of species i, ω - fluid phase reaction rate

The equation 14.3 stand for the gas phase temperature balance taking solid surface into the account,

$$\rho u A \frac{d(C_p T_f)}{dz} + A \sum_{i \in S_g} \dot{\omega}_i \, h_i M_i + P' \sum_{i \in S_g} \dot{s}_i \, h_i M_i = PU(T_w - T_f) \tag{14.3}$$

Where,
C_p - heat capacity, T_w - reactor wall temperature, T_f - fluid phase temperature, U - overall heat transfer coefficient, T_w - reactor wall temperature, P - perimeter of duct

For pressure drop calculation in plug flow reactor, equation 14.4 is used,

$$\frac{dp}{dz} = f \frac{(\rho u^2)}{4r} \tag{14.4}$$

Where,
p - pressure, r - radius of reactor tube, f - friction factor for plug flow reactor

And equation 14.5 is used as equation of states,

$$pM = \rho R T_f \tag{14.5}$$

Where,
R - universal gas constant

The above listed equations are solved using LIMEX solver. In-addition to these equations, since the residence time of gas is often a quantity of interest, a differential equation which computes it automatically is also included. This is simply equation 14.6,

$$\frac{d\tau}{dz} = \frac{1}{u} \tag{14.6}$$

Where,
τ - residence time

To improve the accuracy mass-transfer coefficients are used to approximate the "resistance" to species mass transport between the mean composition and the composition at the reacting catalyst surface. The mass-transfer coefficients are defined by the following relationship equation 14.7,

$$\dot{s}_i \, M_i = k_i(\rho_{i,s} Y_{i,s} - \rho_{i,f}) \tag{14.7}$$

Where,
k_i - fluid solid mass transfer coefficient for species i, $\rho_{i,s}$ - density for species on solid surface i, ρ_{if} - density for species in fluid phase i, $Y_{i,s}$ - mass fraction of solid phase species i

The mass transfer coefficients k_i from equation 14.7 can be represented in non-dimensional form using the species Sherwood numbers from equation 14.8,

$$\mathrm{Sh} = \frac{k_i d}{D_i} \tag{14.8}$$

Where,
D_i - effective diffusion for species i, Sh - Sherwood number

The heat transfer coefficients h for overall heat transfer coefficient U from equation 14.3 can be represented in non-dimensional form using the species Nusselt numbers and calculated using different standard correlation. For more details check section 14.9.

$$\mathrm{Nu} = \frac{h_f L}{k_f} \tag{14.9}$$

Where,
k_f - fluid phase thermal conductivity, Nu - Nusselt number, L - characteristics length

14.1.2 List of variables

Latin symbols

c	molar concentration	mol/m^3
c_p	heat capacity at constant pressure	$J/(kg.k)$
d_t	reactor tube diameter	m
D	effective diffusion coefficient	m^2/s
f	friction factor	$-$
h	enthalpy	J/kg
k_i	mass transfer coefficient of species i	m/s
M	molar mass	$kg/kmol$
p	Pressure	Pa

P	perimeter of duct	m
P'	chemically active channel perimeter	m
R	universal gas constant	$J/mol/K$
$\dot{s}$	rate of surface phase reaction	$mol/m^2/s$
Sh	Sherwood number	$-$
t	time	s
T_f	fluid phase temperature	K
T_w	reactor wall temperature	K
u	velocity	m/s
U	overall heat transfer coefficient	$W/m^2/K$
X	mole fraction	$-$
Y	mass fraction	$-$
z	reactor length	m

Greek symbols

ρ	density	kg/m^3
τ	residence time	s
$\dot{\omega}$	rate of fluid phase reaction	$kg/(m^3.s)$

14.2 Solution

In DETCHEM^{PFR} the standard implicit code LIMEX is used to solve the coupled governing equations. DETCHEM^{PFR} offers various options to solve the problem. The temperature solution offers **Isothermal**, **Non-isothermal**, and **Adiabatic** along with option of **Mass transfer**. The species transport can be modeled for **Gas-phase** and the **Surface-phase** reactions.

14.3 User input

Before running DETCHEM^{PFR} the user must prepare an input file ***pfr.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
    hini=1.d-10
    time=10.
  </CHEMSURF>
</SURFACE-MODEL>

<PFR>
  <GEOMETRY>
reactor = cylindrical
radius = 0.025
<SECTION>
    length = 0.05
    wall_temp = 1200
        gas_chem = y
        surf_chem = y
        Fcat/geo = 1.0
</SECTION>
<WALL>
    wall_thickness = 0.01
    thermal_conductivity_wall = 45
    heat_trans_coeff_outside = 0
</WALL>
  </GEOMETRY>
  <INITIAL>
    T = 500
    P = 1.0E5
    velocity = 0.1
    <MASSFRAC>
       NO2    0.1
       N2     *
    </MASSFRAC>
  </INITIAL>
  <PROCESS>
    isothermal = n
    adiabatic = n
    mass_transfer = n
  </PROCESS>
  <SOLVER>
```

```
    h=1e-20
    atol = 1e-8
    rtol = 1e-8
  </SOLVER>
  <OUTPUT>
    fileNr=1
    write_molefractions = y
    write_massfractions = n
    write_coverages = n
  </OUTPUT>
</PFR>
```

The file pfr.inp starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files thermdata and moldata must be located in the executing directory. Next, an optional washcoat definition may appear in the input file. If washcoat models shall be applied, the user must supply this information. However, by including the washcoat definition the model is not activated automatically. Use the washcoat member in the <PROCESS> tag to activate the model. The options within the <PFR> tag are as follows.

<**PFR**>	opening tag for PFR
<**GEOMETRY**>	reactor geometry parameters
<**INITIAL**>	initial reactor operating parameters
<**PROCESS**>	process conditions for reactor
<**SOLVER**>	DAE solver parameters
<**OUTPUT**>	output options
<**REACTION-FLOW**>	integrated reaction rate calculation
< /**PFR**>	closing tag for PFR

14.3.1 The <GEOMETRY> tag

The **<GEOMETRY>** tag can be used to define the shape and size of the reactor, basically diameter only. The total length can be defined withing **<SECTION>** tag (details in 14.3.2). Also the **<GEOMETRY>** is extended to consider **<WALL>**(details in 14.3.3). The options withing **<GEOMETRY>** tag are :

<**GEOMETRY**>	opening tag for geometry
reactor=*s*	Geometry of the reactor(available options are cylindrical, square_duct, equi_triangle, hexagonal. default: cylindrical)
radius=*d*	radius of the defined reactor geometry
<**SECTION**>	Opening tag for section
< /**SECTION**>	closing tag for section
<**WALL**>	Opening tag for section
< /**WALL**>	closing tag for section
< /**GEOMETRY**>	closing tag for geometry

14.3.2 The <SECTION> tag

The user can divide the packed bed in to several (default: up to 5) sections. This is desired when you have different catalyst properties along the length of the packed bed, or when the wall surface reactions are happening only at certain specific part of the reactor.

<**SECTION**>	opening tag for section
length=*d*	length of the reactor section (m)
wall_temp=*d*	wall temperature of the reactor section (K)
ignition_temp=*d*	ignition temperature to initiate the reactions (K) (default: 0 K)
gas_chem=*b*	gas phase chemistry(default: yes)
surf_chem=*b*	surface phase chemistry(default: no)
< /**SECTION**>	closing tag for section

14.3.3 The <WALL> tag

When it is desired to take care of the heat transfer from the outside wall, this tag must be used and the wall thickness must be given. In case if the thickness is assigned a value 0 then the conduction within the wall will not be taken care in the simulation.

<**WALL**>	opening tag for wall
wall_thickness=*d*	thickness of the reactor wall (m)
wall_roughness=*d*	surface roughness of the reactor wall (m)
thermal_conductivity_wall=*d*	thermal conductivity of the reactor wall (W/m/K)
heat_trans_coeff_outside=*d*	outside heat transfer coefficient (W/m2/K)
overall_heat_trans_coeff=*d*	overall heat transfer coefficient (W/m2/K) (default: using internal subroutine)
plug_wall_heat_trans_method=*i*	method for calculation of heat transfer coefficient under plug conditions(available options are 1, 2, 3, 4 default: 4)
heat_flux=*d*	heat flux (W/m2)
< /**WALL**>	closing tag for wall

14.3.4 The <INITIAL> tag

In this section the user specifies the gas-phase properties at the inlet. Following options can be specified:

<**INITIAL**>	opening tag for initial
T=*d*	inlet temperature of the fluid stream (K)
P=*d*	inlet pressure of the fluid stream (Pa) (default: 1e5 Pa)
velocity=*d*	inlet velocity of the fluid stream (m/s)
<**MOLEFRAC**>	opening tag for inlet mole fractions
< /**MOLEFRAC**>	closing inlet for mole fractions
<**MASSFRAC**>	opening tag for inlet mass fractions
< /**MOLEFRAC**>	closing tag for inlet mole fractions
< /**INITIAL**>	closing tag for initial

Note that the **velocity** is the actual inlet velocity at the given temperature **T** and pressure **P**.

14.3.5 The <PROCESS> tag

Within this section the user need to specify the operating conditions for the packed bed reactor. The is allowed to select between **isothermal**, **adiabatic** and **mass transfer**. For **non-isothermal** conditions user need to say no (**n**) to **isothermal** and **adiabatic** condition.

<**PROCESS**>	opening tag for process
isothermal=*b*	isothermal conditions(default: yes)
adiabatic=*b*	adiabatic conditions(default: no)
mass_transfer=*b*	mass_transfer conditions(default: no)
cov_relax=*d*	under relaxation factor for surface coverages (default: 1.d0)
continuity_eq=*b*	solving continuity equation (default: yes)
axial_T_profile=*b*	use temperature profile for calculation (default: no)
< /**PROCESS**>	closing tag for process

14.3.6 The <T-PROFILE> tag

In order to define a temperature profile along the reactor wall for a single run, use the tag **<T-PROFILE>** and is used to define a piecewise linear temperature profile. Specify at least two pairs of values in that case. The tag requires an arbitrary number of pairs of values in the form (position, temperature), e. g.

```
<T-PROFILE>
    0.000    1000    # z-position [m], temperature [K]
    0.001    1200
    0.002    1300
    0.003    1400
</T-PROFILE>
```

14.3.7 The <SOLVER> tag

In this section the user can define solver specific options:

<**SOLVER**>	opening tag for solver
atol=*d*	absolute tolerance (default: 1e-6)
rtol=*d*	relative tolerance (default: 1e-6)
h=*d*	initial integration step size (m) (default: 1e-10)
hmax=*d*	maximum step size (m) (default: none)
max_iter=*d*	maximum number of iterations (default: unlimited)
IPos=*i*	values of the corresponding solution component will be checked to be nonnegative if IPos = 1 (available options are 0, 1 default: 1)
< /**SOLVER**>	closing tag for solver

The members h, and hmax define the step size of the integration process. If the integration process fails, adjusting these values along with atol and rtol values may help.The parameter max_iter defines a maximum number of integration steps. By limiting the number of iterations, the user can avoid plug flow simulations that need too much time to be solved. There are no strict guidelines for setting the tolerance.

14.3.8 The `OUTPUT` tag

The particular tag contains many options for writing the output file. Depending on the options set within the tag, different variables will be wrote in the output files.

< /**OUTPUT**>	opening tag for output
fileNr=*d*	file number for output, e.g. PFR001.plt, PFR-molefracs001.plt, PFR-massfracs001.plt, PFR-coverages001.plt, PFR-concentrations001.plt (default: 1)
dz_out=*d*	minimum step size for output (m) (default: none)
write_molefractions=*b*	write mole fractions, e.g. PFR-molefracs***.plt (default: yes)
write_massfractions=*b*	write mass fractions, e.g. PFR-massfracs***.plt (default: no)
write_coverages=*b*	write coverages, PFR-coverages***.plt (default: no)
write_concentrations=*b*	write gas phase concentrations, e.g. PFR-concentrations***.plt (default: no)
write_transfer_coeff=*b*	write transfer coefficient (default: no)
< /**OUTPUT**>	closing tag for output

14.3.9 The `REACTION-FLOW` tag

This tag provide access to the integrated reaction rate calculation, which can be further use for reaction path analysis. If the option is used, two additional output files will be generate, PFR-ROP-xxxx.plt and PFR-integrated-rflow-xxxx.csv. NOTE: 1. PFR-ROP: This file contains the local net rate of production for species, for gas phase species unit is mol/m^3/s and for surface phase species unit is mol/m^2/s. 2. PFR-integrated-rflow : This file contains the integrated reaction rate calculated at the end of reactor. On can adjust reactor length to achieve desired result against residence time, unit of gas phase reaction is mol/m^3 and unit of surface reaction is mol/m^2.

<**REACTION-FLOW**>	opening tag for reaction flow
gas_flow_analysis=*b*	gas phase integrated reaction rate (default: no)
surf_flow_analysis=*b*	surface phase integrated reaction rate (default: no)
< /**REACTION-FLOW**>	closing tag for reaction flow

14.4 Output

Depending on the setting in the ***pfr.inp*** file, DETCHEM^{PFR} produces screen output and file output.

14.4.1 Screen output

On screen the integration progress is monitored. The values in the two columns displayed stand for time and temperature.

```
*************************************************************************
****                        DETCHEM PFR                              ****
****            A. Shirsath, S. Tischer, O. Deutschmann              ****
****                 VERSION 2.7.1 ** 2018/08/01                     ****
*************************************************************************

   0.00E+00     500.0
   1.82E-04     653.9
```

```
   7.50E-03     999.7
   ...
   5.00E-02    1199.6
 PFR finish successfully
```

14.4.2 *PFR***.plt*

DETCHEM^{PBR} produces few different output files like, ***PFR-molefracs***.plt***, ***PFR-massfracs***.plt***, ***PFR-coverages***.plt***, and ***PFR-concentrations***.plt***. The output file is in TECPLOT format. TECPLOT will create a line graph upon loading the file ***PFR***.plt***. Nevertheless, this format also allows for easy import into spreadsheet programs like Microsoft Excel. The variables in the file are: axial length(z, 'm'), gas phase temperature(T_gas, 'K'), pressure(P, 'Pa'), residence time(Tau, 's'), velocity('u', 'm/s'), average gas density(rho, 'kg/m^3'), mass flow rate(mdot, 'kg/s'), volumetric flow rate(vdot, 'm^3/s'), molar flow rate(fdot, 'mol/s'), average molecular weight(Meff, 'kg/mol'), species concentration(prefixed with C, 'mol/m^3'), species mole fractions (prefixed with X), and species mass fractions (prefixed with Y). Under write heat transfer coefficient option, additional variable in file are: overall heat transfer coefficient(overall_U, 'W/m^2/K'), wall heat transfer coefficient (h_w, 'W/m^2/K'), and mass transfer coefficient(prefixed with k, 'm/s').

14.5 Examples

There are a few examples supplied along with the installation. Each example directory contains the following files: ***pfr.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism file, a surface mechanism file, ***thermdata***, and ***moldata***.
The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable DETCHEM_DIR that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

14.6 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.
When setting up a new case, we also suggest to start as simple as possible and to become more complex gradually. Check if the geometry settings and wall temperature profile does not contain errors by running simulations without chemistry first. Then add surface and gas-phase mechanisms one after another. Always check for error messages and watch the solution process to detect convergence problems. In case of problems, adjusting the solver and tolerance parameters may improve the performance.

14.7 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable DETCHEM_DIR.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***pfr***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/pfr*** and call ***./pfr***.

The latter option does not require setting of system variables and is therefore less system specific.

14.8 PFR Transient

The DETCHEM^{PFR} code has been further account for to simulate in transient mode. The main purpose of the simulations with DETCHEM^{PFR_TRANSIENT} is the storage process. For this simulation, user must provide one surface type for which transient concentration should get calculated. For the storage medium we must assume that the rate of reaction is slow enough to be decoupled from the flow field. That is, the time scale for storage reactions must be larger than the residence time of the gases inside the reactor. For example, storage of C (solid carbon) on catalyst surface. During such a case, a sub-set of surface reactions can no longer be considered in quasi steady state.The following equation is solved for transient storage species

$$\frac{\mathrm{d}c_i^{\mathrm{storage}}}{\mathrm{d}t} = \dot{s}_i(c^{\mathrm{gas}}, c^{\mathrm{surf}}, c^{\mathrm{storage}}) \tag{14.10}$$

An example is provide with package to show how to setup storage system. Lets say, a system with solid carbon (Cde) on catalytic surface. You can define surface type in species list followed by use of storage tag in surface of mechanism tag as follows.

```
<MECHANISM>
  ...
  <SURFACE name="storage">
    <GLOBAL>
  CH4 + (C) > Cde + 2 H2 + (C)
  <ARRHENIUS>
    A/SIunits = 5e-4
    (C)         0
  </ARRHENIUS>
</GLOBAL>
  </SURFACE>
</MECHANISM>
```

Further more, following options can be used for the transient simulation. The options can be used right after species and mechanism option.

```
{define time_step 30.}   # time step used for the transient simulation, also files will be printed at this time step
{define end_time 180.}   # end time for the transient simulation
{define n_sect 1000}     # no of points to discretized the length of reactor for given time step
```

14.8.1 The <INLET> tag

For inlet conditions it is necessary to prepare another input file. It can have any name (e. g. inlet.inp). Use this name as reference for the file member in the **<INLET>** section of the input file.

<**INLET**>	opening tag for initial
file=*s*	name of the inlet file
< /**INLET**>	closing tag for initial

The inlet file consists of two sections: **<INLETINFO>** and **<INLETDATA>**.There are two types of inlet conditions: constant and variable. An example is shown below.

```
<INLETINFO>
  T = list      # temperatures appear in list
  u = 0.8       # constant velocity
  p = 1.d5      # constant pressure
  <MOLEFRAC list>
     CH4
     O2
     N2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time  Temp   CH4    O2    N2
    0   900   0.0   0.2   0.8
    5   900   0.1   0.2   0.7
   60   800   0.1   0.1   0.8
  120   700   0.1   0.1   0.8
</INLETDATA>
```

In **<INLETINFO>** the user needs to specify which data is read from a list or file and which is constant. Assign one of the following options to the members **T**, **u**, and **p** for temperature, velocity and pressure, respectively:

- a *numerical value* for a constant,
- **list** for data that is listed in **<INLETDATA>**, or

- a *TECPLOT variable name* for spatially varying data in separate files.

The species composition can be given in mole (**<MOLEFRAC>**) or mass fractions (**<MASSFRAC>**). Examples of the syntax for each case are shown below.

- constant mass or mole fractions

```
<MOLEFRAC const>    # define composition as usual
   CH4   0.1
   O2    0.1
   N2    *          # * may be used for balance
</MOLEFRAC>
```

- data listed in **<INLETDATA>**

```
<MOLEFRAC list>     # give the species in the same order
   CH4              # as they appear in the list
   O2
   N2
</MOLEFRAC>
```

14.8.2 Running the tool

The user can call executable in same way as DETCHEM^{PFR} only change is instead of using ***pfr***, one should use ***pfr_transient***.

14.8.3 Output(*global.plt*)

The file ***global.plt*** summarizes the global inlet and global outlet composition at each time step (not only the times listed). The file is in TECPLOT format and can be used to draw line graphs of gas-phase temperature and mass fractions. However, this format also allows for simple import into other spreadsheet programs (e. g. Microsoft Excel).

The variables in this file are: time, temperature, mole fractions(prefixed with X), mass fractions(prefixed with Y), and concentration(prefixed with c, 'mol/m^3'). You can use this file for simple calculation of conversions and selectivities.

A new file ***global.plt*** is written on each execution of DETCHEM^{PFR} (except at postprocessing of a single file). If you want to restart an earlier calculation, save ***global.plt*** with a different name before calling the executable again.

14.9 Correlation used for heat and mass transfer coefficient

This section will cover the type of equation and available options for heat and mass transfer method in pbr.

14.9.1 Heat transfer

Method 1: Tronconi, E., and Forzatti, P. (1992). Adequacy of lumped parameter models for SCR reactors with monolith structure. AIChE Journal, 38(2), 201210
Method 2: Gnielinski, V. in G esellschaft, V. D. I., ed. VDI Heat Atlas. 2nd ed. 2010 edition. Berlin; New York: Springer, 2010
Method 3: Stephan, K., Druckabfall bei nicht ausgebildeter Laminarstromung in Rohren und in ebenen Spahen. (1959). 12, 773778
Method 4: Churchill, S. W., and Ozoe, H., 1973, Correlations for Laminar Forced Convection With Uniform Heating in Flow Over a Plate and in Developing and Folly Developed Flow in a Tube, ASME J. Heat Transfer, 95(1), pp. 7884 (NOTE: default method)

14.9.2 Mass transfer

Correlations are the same as for solid-fluid heat transfer coefficient. Only Pr is replaced by Sc and Nu is replaced by Sh.

Chapter 15

DETCHEM$^{\text{CHANNEL}}$

15.1 Introduction

The DETCHEM$^{\text{CHANNEL}}$ code simulates the steady state chemically reacting gas flow through a cylindrical channel using the boundary layer approximation. Detailed mechanisms for surface and gas phase reactions are considered.

15.1.1 The boundary layer approach

In the most general case of simulating the flow field of a fluid, the governing equations are of Navier-Stokes type. However, due to their mathematical structure, finding a solution can become very time and resource consuming. In order to get a faster algorithm, one possible simplification is given by the boundary layer approximation. It is suitable for systems with a main direction of the convective flow, in which the diffusive transport along this direction is negligible compared to the convection. This assumption becomes valid for a channel or any cylindrical reactor with sufficiently high velocity of the fluid or sufficiently small diameter of the channel. All other transport effects within the fluid, e.g. diffusion limitations of surface reaction rates, are considered in this approximation.

15.1.2 Governing Equations

$$\frac{\partial(r\rho u)}{\partial z} + \frac{\partial(r\rho v)}{\partial r} = 0 \tag{15.1}$$

$$\frac{\partial(r\rho u^2)}{\partial z} + \frac{\partial(r\rho uv)}{\partial r} = -r\frac{\partial p}{\partial z} + \frac{\partial}{\partial r}\left(\mu r\frac{\partial u}{\partial r}\right) \tag{15.2}$$

$$\frac{\partial p}{\partial r} = 0 \tag{15.3}$$

$$\frac{\partial(r\rho uh)}{\partial z} + \frac{\partial(r\rho vh)}{\partial r} = u\frac{\partial p}{\partial z} + \frac{\partial}{\partial r}\left(\lambda r\frac{\partial T}{\partial r}\right) - \frac{\partial}{\partial r}\left(\sum_s r j_s h_s\right) \tag{15.4}$$

$$\frac{\partial(r\rho uY_s)}{\partial z} + \frac{\partial(r\rho vY_s)}{\partial r} = -\frac{\partial}{\partial r}(r j_s) + r\dot{\omega}_s \tag{15.5}$$

The above listed represent the total continuity, axial momentum, radial momentum, the energy and the species continuity respectively.

The flux j_s is calculated by

$$j_s = \begin{cases} \dot{r_s} & \text{if } r = r_{\min} \\ -\rho D_s \frac{M_s}{M}\frac{\partial X_s}{\partial r} & \text{if } r_{\min} < r < r_{\max} \\ -\dot{r_s} & \text{if } r = r_{\max} \end{cases} \tag{15.6}$$

where
z - axial coordinate, r - radial coordinate, u - axial component of velocity, v - radial component of velocity, p - pressure, h - enthalpy density, Y_s - mass fraction of species s, X_s - mole fraction of species s, μ - viscosity, λ -

thermal conductivity, h_s - species enthalpy, $\dot{\omega}_s$ - gas-phase reaction rate, and j_s - the radial diffusion flux.

15.1.3 Solution

After the radial discretization of the above equations into n cells with boundary coordinates $r_0 \dots r_n$ one gets a system of first order differential equations in terms of z coordinate. The aim is to solve the system by method of lines. However, the mathematical structure of equation system is not suitable for the used stiff-stable integrator LIMEX. Therefore some transformation of the system is done.

A finite volume method is used to solve the system. The dependent variables are defined as (the index i denotes the number of the cell):

mass flux

$$\dot{m}_i = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u dr' \tag{15.7}$$

flux of axial momentum

$$\dot{p}_i = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u^2 dr' \tag{15.8}$$

enthalpy flux

$$\dot{H}_i = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u h dr' \tag{15.9}$$

and species mass flux

$$\dot{m}_{s,i} = 2\pi \int_{r_{i-1}}^{r_i} r' \rho u Y_s dr' \tag{15.10}$$

The system of equations are then transformed by using the mass flux instead of radial coordinate as an independent variable. This will satisfy equation (1) and the terms containing the radial velocity v vanish. Here, the boundary layer equations are solved as follows:

- axial momentum

$$\frac{\partial \dot{p}_i}{\partial z} = \left[-\pi r^2 p_z + 2\pi r \mu \frac{\partial u}{\partial r} \right]_{r_{i-1}}^{r_i} \tag{15.11}$$

- energy

$$\frac{\partial \dot{H}_i}{\partial z} = \left[\pi r^2 u p_z + 2\pi r \lambda \frac{\partial T}{\partial r} \right]_{r_{i-1}}^{r_i} \tag{15.12}$$

- species mass

$$\frac{\partial \dot{m}_{s,i}}{\partial z} = \left[\pi r^2 \dot{\omega}_s - 2\pi r j_s \right]_{r_{i-1}}^{r_i} \tag{15.13}$$

where $p_z = \frac{\partial p}{\partial z}$ is used as abbreviation.

The radial coordinate as a dependent variable is calculated by an algebraic equation derived from equation(7). The temperature T is determined implicitly through the dependence of the enthalpy on temperature and species composition. The coefficients μ, λ, D_s are also functions with both dependencies.

The pressure gradient p_z must be determined consistently in order to satisfy the boundary conditions that the radius of the outermost cell is fixed to $r_{\max}$. For reasons of numerical stability of the solver LIMEX, a differential equation is used, which provides a relaxation mechanism to find the correct pressure gradient.

$$\alpha_p \frac{\partial p_z}{\partial z} = p_{z,\mathrm{ref}} \left(1 - \frac{r_n}{r_{\max}} \right) \tag{15.14}$$

The factor α_p can be thought of as the relaxation length of the pressure gradient.

The coupled solution of governing equations in DETCHEM$^{\mathrm{CHANNEL}}$ is attained by the standard implicit code LIMEX.

15.2 User Input

Before running DETCHEM$^{\text{CHANNEL}}$ the user must prepare an input file ***channel.inp***. An example is shown below.

```
{include species.inp}
{include mech.inp}

<SURFACE-MODEL>
  <CHEMSURF>
    hini=1.d-10
    time=1.d0
  </CHEMSURF>
  <Fcatgeo>
    Pt   1
  </Fcatgeo>
  {include washcoat.inp}
</SURFACE-MODEL>

<CHANNEL Version=2.0>
<BASICS>
  title="EXAMPLE DETCHEM V2.0"  # title used in the output file
  ngrid=15                      # number of grids in the radial direction
  zmax=0.1                      # length of the channel [m]
  rmax=0.001                    # radius of the channel [m]
</BASICS>

<SECTION>
  gaschem=y                     # gas-phase chemistry (yes/no)
  <WALL>
    surfchem=y                  # surface chemistry (yes/no)
    mechanism="Pt"              # name of the surface mechanism
  </WALL>
</SECTION>

<SOLVER>
  hini=1.d-10                   # initial step size [m]
  hmax=0.01                     # maximum step size [m]
</SOLVER>

<OUTPUT>
  outg=y                        # gas-phase output file (y/n)
  outs=y                        # surface output file (y/n)
  summary=y                     # add a line to summary_out.dat (y/n)
  file=1                        # file no. appended to output file names
  copy=y                        # save a copy of this input file (y/n)
  monitor=1                     # screen output (0=none, 1=normal, 2=extended)
</OUTPUT>

<INLET>
  T0=300                        # inlet temperature [K]
  u0=0.1                        # inlet velocity [m/s]
  p=1.00e5                      # static pressure [Pa]
  <MOLEFRAC>
    NO2  0.1                    # inlet mole fraction
    N2   *                      # balance mole fraction
  </MOLEFRAC>
</INLET>

<TPROFILE>
  <DISCRETE>
    0.000    1123               # z-position[m], temperature [K]
  </DISCRETE>
</TPROFILE>

</CHANNEL>
```

The file ***channel.inp*** starts with the species and mechanism definition section, that can appear directly in the input file or can be included from external input files. In addition the files ***thermdata*** and ***moldata*** must be located in the executing directory. Alternatively, the pre-processed input file ***detchem.inp*** can be included.

Next, an optional washcoat definition may appear in the input file. If washcoat models shall be applied, the user must supply this information. However, by including the washcoat definition the model is not activated automatically. Use the **washcoat** member in the **<SECTION>** tag to activate the model.

The options within the **<CHANNEL>** tag are as follows.

Version=*s*	input file version (not evaluated)
name=*s*	channel name (used in multi-channel simulations) (default: none)
<**BASICS**>	basic options (must appear as first tag)
<**SECTION**>	channel properties (channel may be devided into sections)
<**SOLVER**>	DAE solver parameters
<**TOLERANCE**>	tolerance parameters
<**OUTPUT**>	output options
<**INLET**>	inlet specifications
<**TPROFILE**>	wall temperature profile

The optional member **name** is only required for setting up multi-channel simulation with the tool DETCHEM^{MONOLITH}.

15.2.1 The <BASICS> tag

The options within the **<BASICS>** tag are:

title=*s*	title (used in TECPLOT output) (default: 'DETCHEM-CHANNEL')
planar=*b*	use planar geometry model (default: no)
ngrid=*i*	number of radial grid points
zmin=*d*	starting position of the channel (default: 0)
zmax=*d*	end point of the channel (or length)
rmin=*d*	inner radius of an annular channel (default: 0)
rmax=*d*	radius of the channel

DETCHEM^{CHANNEL} was desinged to simulate cylindrical channels. Nevertheless, the same approach is also valid for annular reactors, i. e. cylinder geometries with an inner and an outer wall. Use the **rmin** member to specify the radius of an inner wall.

15.2.2 The <SECTION> tag

The user can divide the channel in to several (default: up to 5) sections. This is desired when you have different catalyst properties along the length of the channel, or when the wall surface reactions are happening only at certain specific part of the reactor.

In each of the sections, you can specify the following options:

to=*d*	end point of section (default: zmax)
gaschem=*b*	enable gas-phase chemistry (default: no)
<**INWALL**>	inner wall boundary conditions for an annular channel (default: none)
<**WALL**>	wall boundary conditions

whereas the **<WALL>** and **<INWALL>** subsections contain

boundary=*b*	enable flow boundary condition (default: yes)
surfchem=*b*	enable catalytic surface reactions (default: no)
adiabatic=*b*	adiabatic wall conditions (default: no)
Fcatgeo=*d*	ratio catalytic / geometric surface area (default: 1)
mechanism=*s*	surface mechanism name (default: first surface mechanism defined in <MECHANISM>)

The member **Fcatgeo** defines an additional surface factor for this section. However, this factor cannot distinguish between different surface types. So it applies to all surface reactions in the same way. If the user wants individual catalytic surface areas, declare them in the **<SURFACE-MODEL>** section. If $F_{\mathrm{cat/geo}}$ is defined in both **<SECTION>** and **<SURFACE-MODEL>**, then both apply to the calculation of the reaction rates, i. e. the rates are multiplied by the product of the two values.

WARNING !!!: Declaration of **Fcatgeo** in this section may give wrong results when CHANNEL is used as part of other applications involving the calculation of transient coverages. This value is not passed to higher-order applications!

15.2.3 The <SOLVER> tag

In this section the user can define solver specific options:

hini=*d*	initial integration step size [m] (default: 1e-10)
hmin=*d*	minimum step size [m] (default: 1e-20)
hmax=*d*	maximum step size [m] (default: zmax)
chemmax=*i*	maximum number of iterations with chemical source terms (default: unlimited)
p_relax=*d*	pressure equation relaxation length [m] (default: 1e-10)
bulk=*s*	bulk species, i.e. species for that no conservation equation is solved (default: none)

The members **hini**, **hmin**, and **hmax** define the step size of the integration process. If the integration process fails, lowering these values may help. **p_relax** defines the relaxation length in the pressure correction equation. It should be as small as possible. However by decreasing this parameter, the differential equation system becomes more stiff.

The parameter **chemmax** defines a maximum number of integration steps, for that the chemical source terms are considered. This parameter is only useful in applications where DETCHEM$^{\text{CHANNEL}}$ is called automatically multiple times, e. g. in DETCHEM$^{\text{MONOLITH}}$. By limiting the number of iterations, the user can avoid channel simulations that need too much time to be solved. Although the result is incorrect and an error flag is passed, the simulation may continue.

With the **bulk** member, the user can specify a diluting species for that no transport equation will be solved. The mass fraction of this species always balances to unity. It is recommended that this species is always present (e.ġ. nitrogen or argon) and that this species is the last species in the gas-phase list.

15.2.4 The <TOLERANCE> tag

In this optional section the user may define tolerances for the system variables:

<**MDOT**>	mass flux [kg/s]
<**PDOT**>	momentum flux [kg m/sˆ2]
<**HDOT**>	enthalpy flux [J/s]
<**RADIUS**>	radial grid [m]
<**TEMPERATURE**>	temperature [K]

Each of the subtags may contain the following to parameters:

atol=*d*	absoulte tolerance
rtol=*d*	relative tolerance

In most cases, the default values work fine. If the simulation is not successful, changing these parameters may help.

15.2.5 The <OUTPUT> tag

The options in this section define the screen and file output of DETCHEM$^{\text{CHANNEL}}$

outg=*b*	enable gas-phase output file, e.g. outg001.plt (default: no)
outs=*b*	enable surface output file, e.g. outs001.plt (default: no)
outflux=*b*	enable output file for axial species fluxes (default: no)
Sherwood=*b*	enable additional columns in *outflux*-file for mass-transfer coefficient and Sherwood number (default: no)
summary=*b*	enable output in file summary_out.dat (default: no)
file=*i*	file number used as file name extension (default: 1)
copy=*b*	save copy of input file, e.g. channel001.inp (default: no)
minstep=*d*	minimum step size for writing output [m] (default: none)
monitor=*i*	monitoring options : 0= no monotoring, 1= monitor integration progress, 2= additional screen output (default: 0)
molefrac=*b*	write mole fractions instead of mass fractions (default: no)

The output files will be described later in this chapter. Each of the output file filenames (except ***summary_out.dat***) contains a three-digit extention with the filenumber specified by the **file** member. For parameter studies it may be useful to log the input for a specific cas. If the **copy** option is activated, a copy of the input file ***channel.inp*** will be saved with the same file extention as the other output files.

The **monitor** option determines the screen output during the integration process. There are three levels of output: none (0), normal (1), extended (2). In a single run of DETCHEM$^{\text{CHANNEL}}$ it is useful to monitor the integration process. By doing this, the user may get useful information about the speed of calculation and its convergence. In case of unsuccessful simulations, the extended output can write additional information. For

multiple channel calculations, e. g. using DETCHEM$^{\text{MONOLITH}}$, we recommend to switch off monitoring of the channel calculations.

15.2.6 The <INLET> tag

In this section the user specifies the gas-phase properties at the inlet. Following options can be specified:

T0=*d*	inlet temperature [K]
u0=*d*	inlet velocity [m/s]
p=*d*	pressure [Pa] (default: 101325)
p/bar=*d*	pressure [bar]
<**MOLEFRAC**>	inlet mole fractions
<**MASSFRAC**>	inlet mass fractions

Note that the velocity **u0** is the actual inlet velocity at the given temperature **T0** and pressure **p**. It is up to the user to convert volume fluxes (e. g. in slpm) in to velocities (in m/s)

15.2.7 The <TPROFILE> tag

Specify the wall temperature in this section. The following options apply:

ignite_at=*d*	ignition temperature [K] – acts as a minimum temperature in adiabatic simulation (default: 0)
ignore_below=*d*	minimum temperature [K] for considering chemical reactions (default: 0)
adiabatic=*b*	use adiabatic wall (yes) or temperature profile (no) (default: no)
<**DISCRETE**>	piecewise constant temperature profile
<**LINEAR**>	piecewise linear temperature profile
<**ISOTHERM**>	multiple simulation runs with isothermal conditions

In order to define a temperature profile along the channel wall for a single run, use the tags **<DISCRETE>** or **<LINEAR>**. Both tags require an arbitrary number of pairs of values in the form (position, temperature), e. g.

```
<TPROFILE>
  <DISCRETE>
    0.000    300    # z-position [m], temperature [K]
    0.001    400
    0.002    450
    0.003    500
  </DISCRETE>
</TPROFILE>
```

Using **<DISCRETE>** a piecewise constant temperature profile is defined. The position indicates the begin of the interval with the specified temperature. Define only one pair of values for a constant wall temperature.

<LINEAR> is used to define a piecewise linear temperature profile. Specify at least two pairs of values in that case.

The tag **<ISOTHERM>** is used for multiple runs of DETCHEM$^{\text{CHANNEL}}$ under isothermal conditions. In that case the wall temperature and the inlet temperature are set to a constant value. Also the inlet velocity is being adjusted to the new inlet temperature automatically. Specify a temperature for each run, e. g.

```
<TPROFILE>
  <ISOTHERM>
    300    # temperature [K]
    400
    500
    600
  </ISOTHERM>
</TPROFILE>
```

In this example DETCHEM$^{\text{CHANNEL}}$ will perform four simulations at temperatures 300, 400, 500, and 600 K, respectively. The number of the output file is incremented automatically.

No temperature profile needs to be defined in case of adiabatic conditions. Activate the **adiabatic** member instead. Note: **adiabatic** suppresses heat exchange through the channel wall and therefore should not be used in conjunction with a superordinate simulation like DETCHEM$^{\text{MONOLITH}}$.

Especially in the case of adiabatic calculations, the reaction may not ignite under the given conditions. The option **ignite_at** can be used to simulate a spark to initiate the chemical reaction. The temperature specified in this option is used as a minimum temperature for the calculation of the chemical source terms. All properties of the fluid are calculated at the actual temperature. The **ignite_at** temperature shall be chosen high enough to initiate the chemical reaction, but it should be lower than the expected reaction temperature.

In case a reaction mechanism is negligible for low temperatures, the option **ignore_below** can be used to speed up calculations. For temperatures lower than the specified value, the chemical source terms will not be calculated and are set zero.

15.3 Output

Depending on the settings in the **<OUTPUT>** section of the input file, DETCHEM$^{\text{CHANNEL}}$ produces screen output and file output.

15.3.1 Screen output

If monitoring is activated, the screen output will look like below

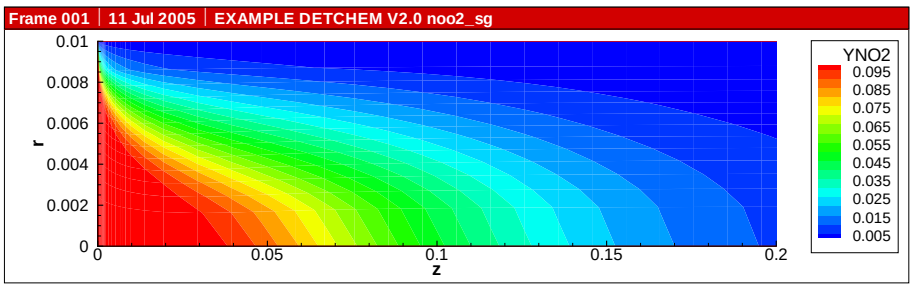
```
************************************************************************
****                        DETCHEM CHANNEL                         ****
****                        STEFFEN TISCHER                         ****
****                  VERSION 2.0.001 ** 2003/07/24                 ****
************************************************************************
     0     0.00000000     0.01000000   1123.00   1123.00
     1     0.00000000     0.01000000    300.00   1123.00
     2     0.00000000     0.01000002    300.00   1123.00
     3     0.00000000     0.01000014    300.00   1123.00
...
   161     1.00000000     0.01000000   1123.03   1123.00
 Iterations: 161    Errors: 0
```

The five colums show the number of integration step, the z-position along the channel, covergence monitoring for the radial coordinate (should always be close to **rmax**), temperature at the channel center, and temperature of the wall.

15.3.2 *outg***.plt*

The file ***outg***.plt*** contains two-dimensional data of the gas-phase of the channel. The file is in TECPLOT format and can be used to draw two-dimensional contour plots. Since the data is stored in ASCII format, it can easily be imported in to other programs, e. g. Microsoft Excel (but Excel cannot be used to draw 2d-contour plots).

The variables in ***outg***.plt*** are: axial coordinate (z), radial coordinate (r), temperature (T), axial velocity (u), radial velocity (v), density (rho), dynamic pressure (pdyn), either mole or mass fractions (prefix X or Y respectively – depends on settings in **<OUTLET>** section).



15.3.3 *outs***.plt*

In case of surface chemistry, the file ***outs***.plt*** is written. The file is in TECPLOT format and can be used to draw one-dimensional line plots of the variables at the wall. The data can easily be imported in to and displayed with other programs, e. g. Microsoft Excel.

The variables in ***outs***.plt*** are: axial coordiante (z), radial coordinate (r – shows convergence of the solver), wall temperature (T), surface coverages (species name without prefix), gas-phase mole or mass fractions (prefix X or Y respectively – depends on settings in **<OUTLET>** section).

15.3.4 *outflux***.plt*

If the user wants to get information about the averaged conversion and selectivity at each position along the channel axis, the file ***outflux***.plt*** can be used. The file contains the radially avaraged mole or mass fractions vs. the axial coordinate. The data is stored in TECPLOT format, but it can easily be imported in to and be displayed with other programs, e. g. Microsoft Excel.

An ***outflux***.plt*** file will also be written if the output option **`Sherwood=y`** has been selected. Additional columns for the effective Sherwood numbers and mass transfer coefficients of all gas-phase species will be added to the output.

$$\beta_s = -\frac{j_s^{\text{surf}}}{\rho^{\text{surf}} \cdot Y_s^{\text{surf}} - \bar{\rho} \cdot \bar{Y}_s} \tag{15.15}$$

$$\text{Sh}_s = \frac{\beta_s \cdot d_{\text{channel}}}{D_s(\bar{T})} \tag{15.16}$$

15.3.5 *summary_out.dat*

Each run of DETCHEM$^{\text{CHANNEL}}$ appends one line in the file ***summary_out.dat***. If the file does not exist, it will be created and a header line is written first. In ***summary_out.dat*** the inlet and the outlet compostions of each channel simulation are summarized. Hence it can be used to compare conversion and selectivity of multiple runs of DETCHEM$^{\text{CHANNEL}}$.

The file ***summary_out.dat*** is a simple text file that has been designed to be opened with any spreadsheet program, e. g. Microsoft Excel. It contains the file numbers, velocities, temperatures and eiter mole or mass fractions of all gas-phase species at the inlet (suffix i) and outlet (suffix o). The header is only valid as long as the species list does not change. So remember to delete or move ***summary_out.dat*** manually when the species list has changed.

15.4 Examples

There are a few examples supplied along with the installation. Each example directory contains the following files: ***channel.inp***, ***species.inp***, ***mech.inp***, a gas-phase mechanism file, a surface mechanism file, ***thermdata***, and ***moldata***. The optional washcoat input has been saved in a separate file ***washcoat.inp***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

15.5 Setting up a problem

It is recommended to create new problems in separate directories. The directories can be created anywhere the user wishes them to be. The created directory, which becomes your working directory must contain all the input files mentioned in the above section. The user can copy these files from the example directory to the working directory and make necessary changes.

When setting up a new case, we also suggest to start as simple as possible and to become more complex gradually. Check if the geometry settings and wall temperature profile does not contain errors by running simulations without chemistry first. Then add surface and gas-phase mechanisms one after another. Always check for error messages and watch the solution process to detect convergence problems. In case of problems, adjusting the solver and tolerance parameters may improve the performance.

15.6 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.

- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***channel***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/channel*** and call ***./channel***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 16

DETCHEM$^{\text{GRIDGEN3D}}$

16.1 Introduction

DETCHEM$^{\text{GRIDGEN3D}}$ is a preprocessor for generating grid information used by DETCHEM$^{\text{MONOLITH}}$ in case of 3-dimensional simulations. Run DETCHEM$^{\text{GRIDGEN3D}}$ prior to setting up your 3d-problem with DETCHEM$^{\text{MONOLITH}}$. The preprocessor can be used to create simple grids for an arbitrary cross-section or to import grids in TECPLOT format that have been created with other programs. If the user wishes to run a 3d-simulation with spatially varying inlet conditions - the grid of the inlet data must be imported using DETCHEM$^{\text{GRIDGEN3D}}$.

In any DETCHEM$^{\text{MONOLITH}}$ simulation the cross-section of the domain must not change along the channel axis. Therefore it is sufficient to specify the geometry of the cross-section. This arbitrary two-dimensional shape will be discretized by a triangular grid. DETCHEM$^{\text{GRIDGEN3D}}$ tries to make the grid as regular as possible, so that neighboring cells have similair size.

On execution, DETCHEM$^{\text{GRIDGEN3D}}$ generates an output file ***r_grid.inp*** that contains the radial grid specification to be included in DETCHEM$^{\text{MONOLITH}}$'s input file ***monolith.inp***.

16.2 User Input

DETCHEM$^{\text{GRIDGEN3D}}$ requires only one input file named ***gridgen.inp***. An example is shown below.

```
<GRIDGEN>

  <POLYGON>
    name="poly"
    n=20          # number of points
    r=0.10        # radius
    dmax=0.03     # maximum distance between 2 grid points
  </POLYGON>

  <LAYER>
    name="insu"
    cells=2
    thickness=0.005
  </LAYER>

  <BORDER>
    name="bound"
  </BORDER>

</GRIDGEN>
```

Since DETCHEM$^{\text{GRIDGEN3D}}$ is a preprocessor and does not do any simulation, there is no species or mechanism information in the input file. ***gridgen.inp*** immediately starts with the **<GRIDGEN>** tag that may contain the following subsections:

<**POLYGON**>	define a regular polygon
<**CIRCUMPOINTS**>	define an arbitrary shape
<**TECPLOT**>	use a grid in TECPLOT format
<**LAYER**>	define additional layers
<**BORDER**>	define border

Depending on the type of grid definition, one of the tags **<POLYGON>**, **<CIRCUMPOINTS>**, or **<TECPLOT>** is required. For all types, additional layers may be declared in the **<LAYER>** subsection. Finally, a border name declaration is required.

16.2.1 The <POLYGON> tag

Use this tag, if the cross-section of the monolith has a shape that can be fitted by a regular polygon with an ellipsoidal circumference. Hence, regular triangles, squares, hexagons and other polygons can be modeled. The **<POLYGON>** sections expects the following parameters.

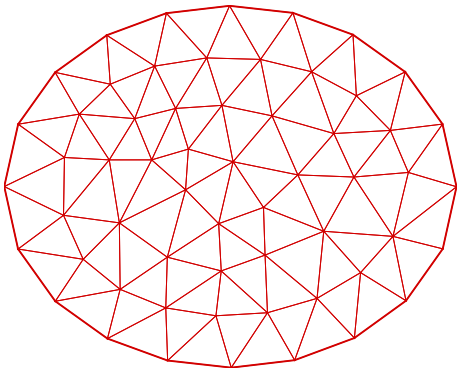
n=*i*	number of edges
r=*d*	radius of polygon [m]
r2=*d*	second radius of an ellipsoidal polygon [m] (default: r)
dmax=*d*	maximum distance of grid points [m] (default: circumference / n)
name=*s*	layer name (used as ID)

Define the shape, maximum grid size and a name of the polygomial domain. The **name** member (use up to 8 characters) will be used in the grid definition of the DETCHEM^MONOLITH input file to refer to this domain.

The definition

```
<POLYGON>
    name="poly"
    n=20            # number of points
    r=0.10          # radius
    r2=0.08         # 2nd radius (if elliptical) - can be omitted dmax=0.03 # maximum distance between 2 grid points
</POLYGON>
```

results in a grid like shown below.



16.2.2 The <CIRCUMPOINTS> tag

In this section, an arbitrary shape of the monolithic domain can be declared by giving the coordinates of its vertices. Besides the coordinates two more parameters are required:

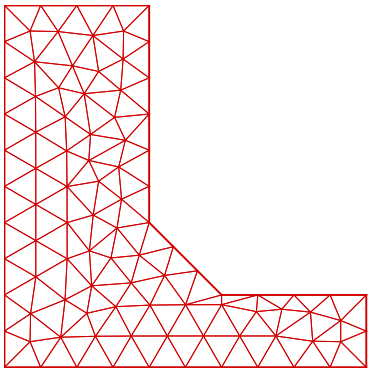
dmax=*d*	maximum distance of grid points [m]
name=*s*	layer name (used as ID)
x-coordinate y-coordinate	coordinates (x,y) of circumferential points [m]

The **name** member (use up to 8 characters) will be used in the grid definition of the DETCHEM^MONOLITH input file to refer to this domain.

The definition

```
<CIRCUMPOINTS>
    name="poly"
    dmax=0.01       #maximum distance between two grid points
    0       0      #y- and z-coordinates of circumference
    .10     0
    .10     .02
    .06     .02
    .04     .04
    .04     .10
    0       .10
</CIRCUMPOINTS>
```

results in a grid like shown below.



16.2.3 The <TECPLOT> tag

If the user wants to use a grid created by other applications, then the **<TECPLOT>** tag shall be used. Especially in case of spatially varying inlet conditions the grid of the inlet data must be the same as the grid of the monolith's cross-section. Thus derive the monolith grid from the inlet data grid using the **<TECPLOT>** tag.

An example is shown below

```
<TECPLOT>
  name="poly"
  file="velin001.dat"   # name of tecplot file
  3d-slice=y            # transformation into y-z-plane necessary
</TECPLOT>
```

where the possible options are

file=*s*	TECPLOT file name
3d-slice=*b*	co-ordinates are given in 3d (yes) or 2d (no) – if yes transformation to 2d is done (default: no)
name=*s*	layer name (used as ID)

The **name** member (use up to 8 characters) will be used in the grid definition of the DETCHEM$^{\text{MONOLITH}}$ input file to refer to this domain.

If the **3d-slice** option is not set, the first two variables of the TECPLOT file will be interpreted as the coordinates of the grid points. Otherwise, The first three variables are considered and the coordinates are transformed into the y-z-plane automatically.

16.2.4 The <LAYER> tag

This tag may be used to define additional layers around the monolithic domain, e.ġ. to represent insulation layers. These layers must have equidistant thickness around the whole domain.

Use a new **<LAYER>** tag for each layer with different material properties. The options within this section are

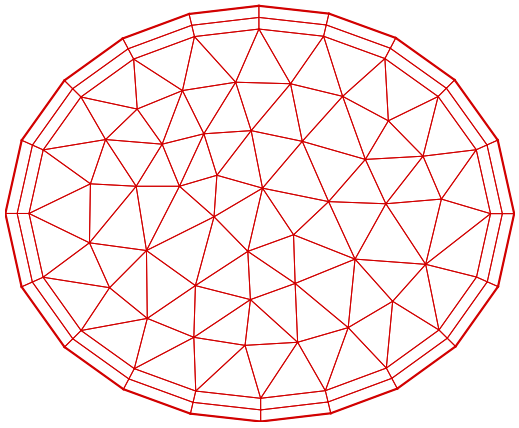
thickness=*d*	layer thickness [m]
cells=*i*	number of cells (default: 1)
name=*s*	layer name (used as ID)

The domain and the layers must have different names (8 significant characters), because these names are used in the grid declaration of DETCHEM$^{\text{MONOLITH}}$.

If the following definition is added to the example shown in section 16.2.1

```
<LAYER>
  name="insu"
  cells=2
  thickness=0.01
</LAYER>
```

the result is



16.2.5 The <BORDER> tag

In order to assign a boundary condition to the whole domain (including additional layers), its border needs an ID that is used in the input file of DETCHEM$^{\text{MONOLITH}}$. This ID is declared as shown in the example below.

```
<BORDER>
  name="bound"
</BORDER>
```

Just assign a string of up to 8 characters to the **name** member.

16.3 Output

DETCHEM$^{\text{GRIDGEN3D}}$ produces two output files ***r_grid.inp*** and ***r_grid.plt***. The former contains the grid information to be read by DETCHEM$^{\text{MONOLITH}}$, and the latter is a TECPLOT file for visualization of the generated grid.

16.3.1 Screen output

There is only few screen output when running DETCHEM$^{\text{GRIDGEN3D}}$. The program informs the user about the current step and eventual errors. After successful execution of the program the screen looks like follows

```
Initialize Polygon
Triangulate Polygon
Initialize Cells
Add layer
Calculate nodes weight
Calculate flux parameters
Calculate cell areas
Write grid
```

16.3.2 *r_grid.inp*

The file ***r_grid.inp*** will be necessary to set up a 3d simultion with DETCHEM$^{\text{MONOLITH}}$. This file should not be edited or changed by the user. Just copy it to your DETCHEM$^{\text{MONOLITH}}$ case directory and include the information in the input file ***monolith.inp***. Details will be explained in chapter 17.

16.3.3 *r_grid.plt*

Before using the grid with DETCHEM$^{\text{MONOLITH}}$ it can be visualized using the file ***r_grid.plt***. This file contains the grid information in TECPLOT format. Upon loading the file as 2d plot with TECPLOT the grid is displayed. The uniformity of the grid can be visulized by activating a contour plot of the cell volumes.

Since the file is saved in text format, it can also be imported into other viewing programs. There are three variables: the two coordintes and the cell volume. They are stored in block format, i.e. all values of one variable sequentially. Below, the cell definitions follow. Four integers make one cell. If the first and the last point are identical, the cell is triangular, otherwise it is quadrilateral.

16.4 Examples

There is an example for each of the three types of grid definition in the distribution. In each subdirectory you can find a file ***gridgen.inp***. In the case of the TECPLOT import example there is also another file ***velin001.dat***.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable DETCHEM_DIR that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

16.5 Running the tool

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable DETCHEM_DIR.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***gridgen3d***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/gridgen3d*** and call ***./gridgen3d***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 17

DETCHEM$^{\text{MONOLITH}}$

17.1 Introduction

DETCHEM$^{\text{MONOLITH}}$ is a simulation code that is designed to simulate transient problems of monolithic reactors. The model can be 2d or 3d. A major characteristic of monolithic reactors is that they consist of a large number of parallel channels. It is assumed that there is no gas exchange between the channels and that the residence time of the gas inside the channels is small compared with the response time scale of the monolith. Further we assume that the cross-section of the monolith does not change along the channel axis. This shall also remain valid when taking into account additional layers with no channels.

Under these circumstances DETCHEM$^{\text{MONOLITH}}$ simulates the transient temperature field facilitating detailed simulations of representative channels with DETCHEM$^{\text{CHANNEL}}$ (or DETCHEM$^{\text{PLUG}}$).

17.2 Physical and Chemical Fundamentals

17.2.1 Governing Equation

DETCHEM$^{\text{MONOLITH}}$ solves the unsteady two or three dimensional heat conduction equation

$$\rho c_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial x_i}\left(\lambda_{ij}\frac{\partial T}{\partial x_j}\right) + q \tag{17.1}$$

where t is the time, T temperature, ρ density, c_p heat capacity, λ_{ij} tensor of heat conductivity, q the heat source term from the interaction with the channels.

17.2.2 Boundary conditions

Solving the partial differential equation requires the specification of boundary conditions at the borders of the simulation domain. DETCHEM$^{\text{MONOLITH}}$ offers two default models: Dirichlet boundary conditions

$$T|_{\text{wall}} = \text{const} \tag{17.2}$$

and Neumann conditions

$$-\lambda \left.\frac{\partial T}{\partial n}\right|_{\text{wall}} = \Phi \tag{17.3}$$

Where the heat flux Φ in the normal direction of the border n may consist of a linear heat condition term, a biquadratic radiation term and a constant term:

$$\Phi = c(T - T_{\text{surr}}) + \epsilon\sigma(T^4 - T_{\text{surr}}^4) + \Phi_{\text{const}} \tag{17.4}$$

σ is the Stefan-Boltzmann constant. The heat transfer coefficient c, the emissivity ϵ and the temperature of the surroundings T_{surr} have to be specified by the user. Instead of specifying a heat transfer coefficient c, the user can also choose to use the free convection in air model using the following empirical correlations for $c = \frac{Nu}{l_{char}} * \lambda_{air}$:

Vertical surface:

$$f_1 = (1 + 0.671Pr^{-\frac{9}{16}})^{-\frac{8}{27}}$$
$$Nu = (0.825 + 0.387Ra^{\frac{1}{6}} f_1)^2$$

Upper surface:

$$f_2 = (1 + 0.536Pr^{-\frac{11}{20}})^{-\frac{20}{11}}$$
$$Nu = \begin{cases} 0.766(Ra \cdot f2)^{\frac{1}{5}}, & \text{if } Ra \cdot f_2 \leq 70000 \\ 0.15(Ra \cdot f2)^{\frac{1}{3}}, & \text{otherwise} \end{cases}$$

Lower surface:

$$f_2 = (1 + 0.536Pr^{-\frac{11}{20}})^{-\frac{20}{11}}$$
$$Nu = 0.27(Ra \cdot f_2)^{\frac{1}{4}}$$

Furthermore, DETCHEM^{MONOLITH} also offers the opportunity of user defined boundary conditions.

17.2.3 Material properties

The simulation domain can be dived into zones with different material properties. These are the density ρ, the heat capacity c_p and the heat conductivity λ_{ij}. For the latter two components may be defined independently: the heat conductivity along the channel axis λ_{ax} and perpendicular to this axis λ_{rad}. All these properties are properties of the porous medium, taking the porosity into account. (Note that the denisity is the average density including the channels.) Their temperature dependence is modeled by polynoms, e. g.

$$c_p(T) = c_0 + c_1 T + c_2 T^2 + c_3 T^3 + c_4 T^4 + \ldots \tag{17.5}$$

17.2.4 Heat source term

The heat source term is derived from the simulation of individual channels. For systems with nearly constant pressure, the energy conservation can be expressed in term of enthalpy. Then the enthalpy flux in the channel $\dot{H}_{\text{channel}}$ can only be change alonge the channel axis by heat exchange with the monolith. If the channel density is σ (channels per unit area of the cross-section), the source term can be expressed as

$$q = -\sigma \cdot \frac{\partial \dot{H}_{\text{channel}}}{\partial x} \tag{17.6}$$

17.2.5 Storage effects

The version 2.0 of DETCHEM^{MONOLITH} has been extended to the simulation of storage catalysts. In this kind of monolithic reactors components have been added to the surface that take place in slower long-term reactions (e. g. storage of NOx on BaO). In this case, a sub-set of surface reactions can no longer be considerd in quasi steady state. For surface species involved in these reactions, DETCHEM^{MONOLITH} has to solve a transient species equation

$$\frac{\mathrm{d}c_i^{\text{storage}}}{\mathrm{d}t} = \dot{s}_i(c^{\text{gas}}, c^{\text{surf}}, c^{\text{storage}}) \tag{17.7}$$

17.3 Solution methods

17.3.1 Discretization of the PDE

The patial differential equation (PDE) needs to be discretized. Therefore we use a finite volume method. The grid along the channel axis is always chosen perpendicular to the grid in radial direction. In case of 3d simulations the radial grid does not need to be regular.

For a PDE of the form

$$\frac{\partial \psi}{\partial t} = \frac{\partial}{\partial x_i}\left(k \frac{\partial \psi}{\partial x_j}\right) \tag{17.8}$$

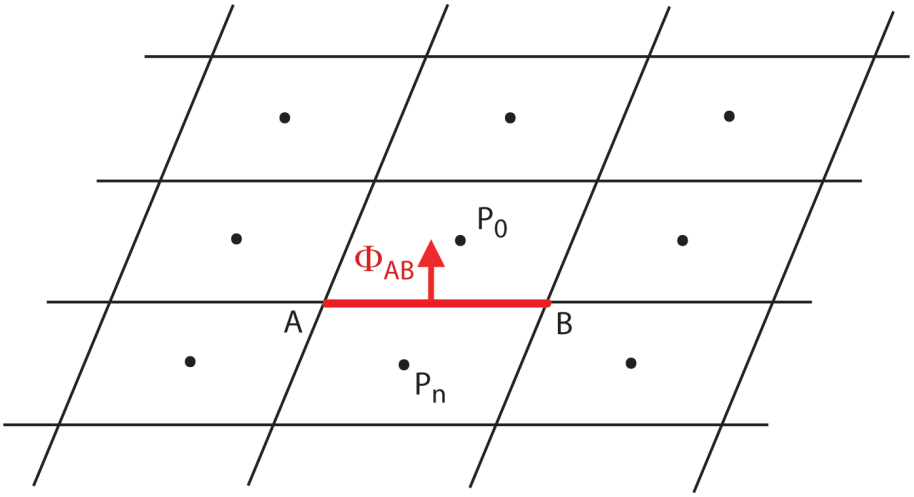
we integrate over a cell and apply Stokes' integral theorem to get

$$V \cdot \frac{\partial \psi}{\partial t} = \sum_{\text{faces}} \Phi_{\text{face}} \tag{17.9}$$

where Φ_{face} is the flux through a face of the cell. The cell itself shall be denoted by index 0 and its neighbor by index n. In axial direction we can directly use a finite difference approximation of the gradient because of the orthogonality of the grid

$$\Phi_{\text{ax}} = -kA_{\text{face}} \frac{\psi_0 - \psi_n}{x_0 - x_n} \tag{17.10}$$

In radial direction we need to consider a situation like shown below



The gradient can be approximated by the four values $\psi_0, \psi_n, \psi_A, \psi_B$. Then the flux becomes

$$\Phi_{\text{rad}} = \Phi_{AB} \cdot \Delta x \tag{17.11}$$

with

$$\Phi_{AB} = k \cdot \begin{pmatrix} \frac{\partial \psi}{\partial y} \\ \frac{\partial \psi}{\partial z} \end{pmatrix} \cdot \begin{pmatrix} -(z_B - z_A) \\ y_B - y_A \end{pmatrix} \tag{17.12}$$

This expression can be simplified into

$$\Phi_{AB} = -k \left(q(\psi_0 - \psi_n) + p(\psi_B - \psi_A) \right) \tag{17.13}$$

with

$$q = \frac{\vec{AB}^2}{\left| \vec{AB} \times \vec{P_n P_0} \right|} \tag{17.14}$$

$$p = \frac{\vec{AB} \cdot \vec{P_n P_0}}{\left| \vec{AB} \times \vec{P_n P_0} \right|} \tag{17.15}$$

The parameters q and p only have to be calculated once for the grid. With them the right hand side of the PDE is transormed into a system of algebraic equations.

17.3.2 Choosing representative channels

Calculating the channel flow field is the most time consuming step in a DETCHEM$^{\text{MONOLITH}}$ simulation. Therefore it is worth to reduce to amount of work. The concept is that only representative channels are chosen for a detailed numerical simulation.

Due to the representation of the monolith by a discrete grid, there is a unique channel wall temperature profile for each axial row of grid points. Furthermore, when applying spatially vaying inlet conditions, the

channels may differ in inlet temperature T, velocity u, and species mass fractions $Y_1, \ldots, Y_s$. However, this finite number of variables completely defines the parameters for a channel simulation, i. e. the calculation of source terms can be viewed as a mapping from the input vector

$$x^k = \left(T^k_{\text{wall},1}, \ldots, T^k_{\text{wall},n}, T^k_{\text{in}}, u^k_{\text{in}}, Y^k_{\text{in},1}, \ldots, Y^k_{\text{in},s}\right) \tag{17.16}$$

of the kth discretized channel to an output vector containing the heat source terms q of the respective grid points. This mapping can be expected to be continuous – channels with similair input vectors shall be similair in source terms. For the identification of similair channels an agglomerative cluster algorithm is applied.

A weight w^k can be assigned to each vector x^k that accounts for the absoulute number of channels repesented by x^k, which is proportional to the size of the corresponding monolith grid cell. In addition, a distance function $d(x^i, x^j)$ is necessary for which a normalized maximum norm was chosen:

$$\begin{aligned} d(x^i, x^j) \quad = \quad & (d_T(T^i_{\text{wall},1}, T^j_{\text{wall},1}), \ldots, d_T(T^i_{\text{wall},n}, T^j_{\text{wall},n}), \\ & d_T(T^i_{\text{in}}, T^j_{\text{in}}), d_u(u^i_{\text{in}}, u^j_{\text{in}}), \\ & d_Y(Y^i_{\text{in},1}, Y^j_{\text{in},1}), \ldots, d_Y(Y^i_{\text{in},s}, Y^j_{\text{in},s})) \end{aligned} \tag{17.17}$$

with distance functions

$$d_T(T^i, T^j) = \frac{\left|T^i - T^j\right|}{\delta T} \quad , \tag{17.18}$$

$$d_u(u^i, u^j) = \frac{\left|u^i - u^j\right|}{\xi_u \max(u^i, u^j)} \quad , \tag{17.19}$$

$$d_Y(Y^i, Y^j) = \frac{\left|Y^i - Y^j\right|}{\xi_Y \max(Y^i, Y^j)} \quad . \tag{17.20}$$

Temperatures are normalized with respect to a temperature difference δT, whereas inlet velocities and mass fractions are normalized by their relative deviations ξ_u and ξ_Y, respectively.

The agglomerative cluster algorithm, illustrated in Fig. 17.1, can be sketched as follows:

1. Find the minimum $d^* = d\left(x^{i^*}, x^{j^*}\right) = \min_{i \neq j}\left(d(x^i, x^j)\right)$.
2. If $d^* > 1$ then stop.
3. Let $x^* = \left(w^{i^*} x^{i^*} + w^{j^*} x^{j^*}\right) / \left(w^{i^*} + w^{j^*}\right)$ and $w^* = w^{i^*} + w^{j^*}$.
4. Eliminate $\left(x^{i^*}, w^{i^*}\right)$, $\left(x^{j^*}, w^{j^*}\right)$ and replace by (x^*, w^*).
5. Go to 1.

The remaining x^k are the input vectors for representative channels. All input vectors in one cluster are associated with the same output source terms. Since the channel equations are not explicitly time dependent, the number of channel calculations can be further reduced by including the x^k of previously calculated channels into the clustering and reusing their results.

In case of the storage model, the transient coverages of the storage medium have to be included in the input vector x^k. During clustering, a distance function with normalization with respect to an absolute coverage difference $\delta\Theta$ is used:

$$d_\Theta(\Theta^i, \Theta^j) = \frac{\left|\Theta^i - \Theta^j\right|}{\delta\Theta} \tag{17.21}$$

17.3.3 Co-flow and Counter-flow configurations

For DETCHEM^{MONOLITH} the single channel simulation is a black-box process. Therefore, the same input vector could be used to call more than one channel simulation in order to calculate the heat source terms. The channels can, for instance, differ in size and surface properties. However, the different channel types must be arranged regularly, so that the continuum model is still applicable. By this approach, co-flow and counter-flow configurations can be simulated.

17.4 User input

DETCHEM$^{\text{MONOLITH}}$ requires an input file named ***monolith.inp***. Furthermore, since DETCHEM$^{\text{CHANNEL}}$ is called for the single channel simulations, all input information for one or more channel types must also be provided in the executing directory (see chapter 15). An example of the file ***monolith.inp*** is shown below.

```
{include channel.inp}

<MONOLITH2D Version=2.0>           # use 2d model

 <GRID>
   cylindrical= yes                # cylindrical geometry
   <XDOMAIN>
     from=      0.                 # axial coordinate
     to=        0.01
     cells=    10                  # number of cells
     aspratio=  1.                 # aspect ratio of neighboring cells
   </XDOMAIN>

   <YBORDER>
     <NAMES> wall </NAMES>         # boundary name
   </YBORDER>

   <YDOMAIN>
     from=      0                  # radial coordinate
     to=        0.01
     cells=    10                  # number of cells
     aspratio=  1                  # aspect ratio of neighboring cells
     monolithic=yes                # domain contains channels
     <NAMES> wall monolith wall </NAMES>
                                   # names of boundaries and materials
   </YDOMAIN>

   <YBORDER>
     <NAMES> wall </NAMES>         # boundary name
   </YBORDER>
 </GRID>

 <MATERIALS>                       # material properties
   <TLIMITS>                       # temperature limits for polynoms
     Tmin=   300.
     Tmax=  1200.
   </TLIMITS>

   <monolith>                      # name of the material
                                   # polynomial coefficients for:
     <k dir=axial>   0.5 -0.00015  6.e-08  2.5e-10 </k>
                                   # axial heat conductivity
     <k dir=radial>  0.3 -0.00025  7.e-07 -3.4e-10 </k>
                                   # radial heat conductivity
     <rho>   500. </rho>           # density
     <cp>   90  3.2 -0.0033  1.2e-6 </cp>
                                   # heat capacity
   </monolith>
 </MATERIALS>

 <BOUNDARIES>                      # boundary conditions
   <wall type=Neumann>             # name of the boundary, Neumann b.c.
     htc=  0                       # heat transfer coefficient
     Tsurr=  298.15                # surrounding temperature
     cflux=  0.                    # constant flux
     emiss=  0.3                   # emissivity
   </wall>
 </BOUNDARIES>

 <INITIAL>                         # initial conditions
    T0=  298.15                    # initial temperature
 </INITIAL>

 <SOLUTION>                        # solution parameters
   tend=    100                    # end time of simulation
   dt_ini=    0.03                 # initial time step
   dt_max=   10                    # maximum time step
   delta_T=  15                    # maximum temperature step
 </SOLUTION>

 <OUTPUT>                          # output parameters
   write_restart=yes               # write file restart.plt
   <DO> from=10 to=100 step=10 </DO>
                                   # write monolith***.plt every 10 seconds
 </OUTPUT>

 <CHANNELS>                        # channel parameters
   <channel1>
     ch/cm2=  62                   # channel density
     reverse= no                   # reverse flow
```

```
  </channel1>
</CHANNELS>

<CHOOSE>                          # clustering parameters
  delta_T=  20                    # temperature difference
  pvar_v=  10                     # velocity variance (percent)
  pvar_Y=  10                     # mass fraction variance (percent)
  memorize=yes                    # memorize results between time steps
</CHOOSE>

<INLET type=CONSTANT>             # constant inlet conditions
  u0=  2.                         # inlet velocity
  T0=  1200.                      # inlet temperature
  <molefrac>                      # inlet mole fractions
    NO2       0.1
    N2        *                   # balance
  </molefrac>
</INLET>

</MONOLITH2D>
```

The file ***monolith.inp*** must start with the definitions necessary for the channel calculations. This implies that first, the species definiton, the mechanism definition, the optional washcoat deinition, and the channel definitions appear first in the input file. It is probably most convenient to use the same input file as for the channel simulations and include the channel input file by an **include** statement. Alternatively, it might be convenient to provide channel input file(s) without the species and mechanism information and include each definition seperately by an include statement:

```
{include species.inp}
{include mech.inp}
{include washcoat.inp}
{include channel1.inp}
{include channel2.inp}
```

The monolith definition has to be enclosed by the tags **<MONOLITH2D>** or **<MONOLITH3D>** depending on which model the user wants to apply – two- or three-dimensional. In either case the definition consists of the following options and sections:

Version=*s*	input file version (not evaluated)
<**GRID**>	define grid (must be the first tag)
<**MATERIALS**>	define material properties
<**BOUNDARIES**>	define boundary conditions
<**INITIAL**>	define initial conditions
<**SOLUTION**>	define solver specific options
<**OUTPUT**>	define output options
<**CHANNELS**>	define channel properties
<**CHOOSE**>	define options for choosing representative channels
<**POSTPROCESS**>	postprocessing
<**STORAGE**>	storage model
<**INLET**>	define inlet conditions

A detailed description of each of the sections follows.

17.4.1 The <GRID> tag (2d simulations)

An example for a definition of a simple two-dimensional grid has been given in the example above. In this case the **<GRID>** section requires the following members:

cylindrical=*b*	cylindrical (yes) or flat (no) geometry (default: no)
<**XDOMAIN**>	define a domain in axial direction
<**YDOMAIN**>	define a domain in radial direction
<**YBORDER**>	define a radial border

Two-dimensional grids can either represent a flat geometry or a geometry with cylinder symmetry. In the latter case the y-direction is the radial one. Use the **cylindrical** member in the input file to define the geometry type.

The simulation area may be divided into domains. Both, the x- and the y-axis can consist of one or more sections. A domain is then given by the area covered by one x-domain and one y-domain. Each domain can have its own grid and material properties. The borders of each domain can also be assigned with individual boundary conditions.

Suppose you want to divide the simulation area into domains as shown in Fig. 17.2. You have two shields of the same type and a monolith, all together they are covered by an insulation layer. The boundary conditions

at the inlet and the outlet shall be the same for shield and insulation, whereas there are three different boundary conditions for the outer wall. Follow the steps below to define such a grid.

<XDOMAIN>

First, the user needs to define the sections of the domains along the x-axis. Write a **<XDOMAIN>** tag for each section. In the example case define three sections as shown below.

```
<XDOMAIN>
  from    =  0
  to      =  0.02
  cells   = 10
</XDOMAIN>
<XDOMAIN>
  to      =  0.05
  cells   = 20
  aspratio=  1.05    # non-uniform grid
</XDOMAIN>
<XDOMAIN>
  to      =  0.07
  cells   = 10
</XDOMAIN>
```

The members in each **<XDOMAIN>** tag are:

from=*d*	axial coordinate of start of the domain (default: end of the previous domain)
to=*d*	axial coordinate of end of the domain
cells=*i*	number of grid cells
aspratio=*d*	aspect ratio, i.e. ratio of the lengths of two adjacent cells (default: 1)

<YDOMAIN> and <YBORDER>

Next, define borders and domains along the y-direction in ascending order. In the example case there is the symmetry border, two domains, and the wall border. Therefore a it is necessary to declare a **<YBORDER>**, followed by two **<YDOMAIN>**'s and another **<YBORDER>**.

Along with the structure in y-direction the names of the domains are declared. Since a domain is formed by one sections in x-direction and one in y-direction, there must be as many domain names in each y-section as there are x-sections. In the example case the x-axis has been divided into three and the y-axis into two sections. Therefore there are a total of six domains. Each of these domains may have different properties, but they do not have to. Here we use three different material types and only need to declare three material names. In addition, the boundaries are also identified by names. A **<YBORDER>** needs to declare the boundary names for each section in x-direction. On the other hand, a **<YDOMAIN>** has two boundaries – one at inlet and one at outlet side. Therefore two boundary names are to be declared. For the example shown in Fig. 17.2 this part of the declaration is shown below

```
<YBORDER>
  at 0
  <NAMES> symmetry symmetry symmetry </NAMES>   # same b.c. for all sections
</YBORDER>
<YDOMAIN>
  to        =  0.04
  cells     = 20
  asparatio =  0.95
  monolithic= yes
  <NAMES> inlet shield monolith shield outlet </NAMES>   # two boundaries
</YDOMAIN>                                               # and three materials
<YDOMAIN>
  to        =  0.05
  cells     =  5
  monolithic= no
  <NAMES> inlet insu insu insu outlet </NAMES>   # boundaries and material
</YDOMAIN>
<YBORDER>
  <NAMES> wall1 wall2 wall3 </NAMES>             # different b.c.'s
</YBORDER>
```

Note that names only need to be different when the properties of the domains are different. DETCHEM$^{\text{MONOLITH}}$ uses eight characters to distinguish material properties and boundary conditions. Therefore for the insulation the abbreviated name **insu** has been used in the example. Thus, three material names (**monolith**, **shield**, and **insu**) and six boundary names (**symmetry**, **inlet**, **outlet**, **wall1**, **wall2**, **wall3**) have been declared.

In case of a cylindrical geometry negative coordinates are not allowed. The border at position 0 has to be declared for syntax reasons, but because this border has no area, the boundary condition has no influence on the simulation result.

If DETCHEM^MONOLITH is not only used to calculate a heat balance without internal heat sources, then at least one **<YDOMAIN>** must be declared **monolithic**. This member is used to specify where the channels are located. In the example, the channels cross the **monolith** and **shield** domains, but not the insulation.

The members of the **<YDOMAIN>** section are summarized below:

from=*d*	radial coordinate of start of the domain (default: end of the previous domain)
to=*d*	radial coordinate of end of the domain
cells=*i*	number of grid cells
aspratio=*d*	aspect ratio, i.e. ratio of the lengths of two adjacent cells (default: 1)
monolithic=*b*	Does this domain contain channels? (default: no)
<**NAMES**>	define names (IDs) for boundary conditions (left and right) and materials (one name for each XDOMAIN)

and for **<YBORDER>**:

at=*d*	radial coordinate of the boundary (default: coordinate of the next YDOMAIN)
<**NAMES**>	define names (IDs) for boundary conditions (one name for each XDOMAIN)

17.4.2 The <GRID> tag (3d simulations)

The declaration of a three-dimensional grid differs from the two-dimensional case. As before, the user also needs to declare domains in x-direction. However, the grid in the y-z-plane cannot be declared directly in ***monolith.inp***. Use the grid generator DETCHEM^GRIDGEN3D (see chapter 16) for its creation. Its output file (default name ***r_grid.inp***) is then used as input file for DETCHEM^MONOLITH.

A 3d grid definition will look like this:

```
<GRID>
  <XDOMAIN>
     from    =  0
     to      =  0.05
     cells   = 20
  <XDOMAIN>

  {include r_grid.inp}

  # suppose that the following names were defined by GRIDGEN3D:
  # layers: ellipse, layer1
  # border: border1

  <ellipse>
    monolithic=yes
    <NAMES> inlet monolith outlet </NAMES>
  </ellipse>

  <layer1>
    monolithic=no
    <NAMES> inlet insu outlet </NAMES>
  <layer1>

  <border1>
    <NAMES> wall </NAMES>
  <border1>
</GRID>
```

The members of the **<GRID>** tag are

<**XDOMAIN**>	define a domain in axial direction
<**RGRID**>	tag used by ***r_grid.inp*** created with GRIDGEN3D
<***rlayer_name***>	define layer properties (layers defined in GRIDGEN3D)
<***rborder_name***>	define border properties (borders defined in GRIDGEN3D)

The definition of the **<XDOMAIN>** is identical to the 2d case. The section must start with one or more **<XDOMAIN>** tags. Next define the radial grid. This is usually done by including the grid information file (default: ***r_grid.inp***).

In order to assign names to the domains, it is important to know the names of the layers and borders defined in DETCHEM^GRIDGEN3D. Each layer of the radial grid can be assigned to two boundary conditions (inlet and outlet side of the monolith) and as many domains as there are **<XDOMAIN>** tags. Therefore, the **NAME** tag of a

radial grid layer contains a boundary name, one or more material names and another boundary name. At least one layer shall be have the attribute **monolithic**. It defines the domains where the channels are located.

monolithic=*b* Does this domain contain channels? (default: no)
<**NAMES**> define names (IDs) for boundary conditions (left and right) and materials (one name for each XDOMAIN)

In the current version of DETCHEM$^{\text{GRIDGEN3D}}$ the whole border of the radial grid is assigned one name. However, the format of ***r_grid.inp*** allows to define more than one boundary name. Each radial boundary is also a boundary of each domain of the x-axis. Therefore the **NAME** tag of the radial boundary section contains as many boundary names as there are **<XDOMAIN>** tags.

<**NAMES**> define names (IDs) for boundary conditions (one name for each XDOMAIN)

17.4.3 The <MATERIALS> tag

This section contains the definitions of the material properties of the solid structure. Besides the tag **<TLIMITS>** it must contain one tag for each of the material names defined in the **<GRID>** section of the input file.

```
<MATERIALS>                       # material properties
  <TLIMITS>                       # temperature limits for polynoms
    Tmin=   300.
    Tmax=  1200.
  </TLIMITS>

  <monolith>                      # name of the material
    <k dir=axial>   0.5 -0.00015  6.e-08  2.5e-10 </k>
                                  # axial heat conductivity
    <k dir=radial>  0.3 -0.00025  7.e-07 -3.4e-10 </k>
                                  # radial heat conductivity
    <rho>   500. </rho>           # density
    <cp>   90  3.2 -0.0033  1.2e-6 </cp>
                                  # heat capacity
  </monolith>
</MATERIALS>
```

Since the material properties are defined as polynoms, it is required to give the minimum and maximum temprertures for that the polynomial approximation is valid. Use the tag **<TLIMITS>** and its members **Tmin** and **Tmax** to define these limits.

Use the material names defined in the **<GRID>** section as tag names to define the properties: heat conductivity, density, and heat capacity. Each of the name tags contains the following members.

<**k**> polynomial coefficients for heat conductivity [W/m K]
dir=*s* direction : axial or radial
<**rho**> polynomial coefficients for effective density [kg/m^3]
<**cp**> polynomial coefficients for heat capacity [J/kg K]

The heat conductivity tag **<k>** requires an additional parameter to specify the direction. The effective heat conductivity may be different for the axial and the radial direction.

Between the property tags any number of coefficients (memory allocation defaults to maximum 5) can be given to define the polynomial dependence of the property with respect to temperature (in K), i. e. in the example above, the heat capacity is given by

$$c_p = 90 + 3.2T - 0.0033T^2 + 1.2 \cdot 10^{-6}\, T^3 \quad \text{in J/kg K}$$

Note that all properties need to be assigned effective values, i. e. global values of the (porous) solid structure. Especially the density of the monolith needs to take into account the volume of the channels, too.

17.4.4 The <BOUNDARIES> tag

Define the boundary conditions in this section. Create a tag for each boundary name defined in the **<GRID>** section of the input file. An example with two boundary conditions (names: **wall** and **inlet**) is shown below.

```
<BOUNDARIES>
  <wall type=Neumann>
    htc  =  0
    Tsurr=  300
    cflux=  0
    emiss=  0.8
  </wall>

  <inlet type=Dirichlet>
    T = 1000
  </inlet>
</BOUNDARIES>
```

The first member in each tag must be the **type** specification, therefore it is recommended to use a syntax that includes this option in the tag brackets. The options for Dirichlet (given temperature) and Neumann (given heat flux) boundary conditions are summarized in the following table.

type=*s*	type of boundary condition: Dirichlet, Neumann or Hotbox
T=*d*	constant temperature [K]
htc=*d*	heat transfer coefficient [W/m^2 K] (default: 0)
free_conv=*b*	Use free convection in air model to compute htc? (default: no)
Tsurr=*d*	temperature of surroundings [K] (default: 298)
cflux=*d*	constant heat flux [W/m^2] (default: 0)
emiss=*d*	emissivity (0<= emiss <=1) (default: 0)
<**T-Profile**>	time-dependent temperature profile (replacing **T** or **Tsurr**)

Define a constant temperature or a Dirichlet boundary condition according to eq. 17.2 using the **T** member.

The four parameters of a Neumann boundary condition (eq. 17.4) are defined by the members **htc** (c), **Tsurr** (T_{surr}), **cflux** (Φ_{const}), and **emiss** (σ). The flag **free_conv** can alternatively be used to compute **htc** using the free convection model described in section 17.2.2.

The boundary temperature parameter may also be befined time dependendent using the tag **<T-Profile>**. This section contains a list with two columns, for time and temperature, respectively. Between the time intervals, the temperature is interpolated linearly.

```
<T-Profile>
   0[s]    300[K]
  60[s]    400[K]
   2[min]  160[degC]
# if units are omitted, they default to [s] and [K]
</T-Profile>
```

17.4.5 HotBox boundary model

Aside from the boundary conditions described above, the placement of a monolithic stack in a HotBox can be simulated. This boundary condition contains convective heat exchange with the hot cavity gas (i.e. within the box), radiative heat exchange with the HotBox inner wall, conduction across the HotBox thickness, as well as convective and radiative heat loss to the box surroundings. Thereby, component-to-HotBox interactions of additional system components like a catalytic afterburner can be considered. The normal heat flux Φ at the monolith borders are evaluated by solving thermal balances of cavity gas temperature T_{gas} (eq. 17.23), HotBox inner wall temperature T_{HMi} (eq. 17.24) and HotBox outer wall temperature T_{HMo} (eq. 17.25).

$$\Phi = h_{\text{rad}}(T - T_{\text{HMi}}) + h_{\text{conv}}(T - T_{\text{gas}}) \tag{17.22}$$

$$h_{\text{conv,B}}A_{\text{B}}(T_{\text{B}} - T_{\text{gas}}) + n_{\text{stack}}h_{\text{conv}}A_{\text{B}}(T - T_{\text{gas}}) + h_{\text{conv,HMi}}A_{\text{HMi}}(T_{\text{HMi}} - T_{\text{gas}}) + \dot{n}_{\text{p,gas}}c_{\text{p,gas}}(T_{\text{gas,in}} - T_{\text{gas}}) = 0 \tag{17.23}$$

$$-h_{\text{conv,HMi}}A_{\text{HMi}}(T_{\text{HMi}} - T_{\text{gas}}) + n_{\text{stack}}h_{\text{rad}}A(T - T_{\text{HMi}}) + h_{\text{rad,B}}A_{\text{B}}(T_{\text{B}} - T_{\text{HMi}}) = \lambda_{\text{HM}}S(T_{\text{HMi}} - T_{\text{HMo}}) \tag{17.24}$$

$$h_{\text{conv,HMo}}A_{\text{HMo}}(T_{\text{HMo}} - T_{\text{amb}}) + h_{\text{rad,HMo}}A_{\text{HMo}}(T_{\text{HMo}} - T_{\text{amb}}) = \lambda_{\text{HM}}S(T_{\text{HMi}} - T_{\text{HMo}}) \tag{17.25}$$

An example input file defining HotBox boundary conditions is given below.

```
<BOUNDARIES>
  <wall type=HotBox>
    ndot_gas= 2.0 [mol/s]
    T_gas_initial= 800 [K]
    T_gas_in= 600 [K]
    T_amb= 298.15 [K]
    T_B= 823.15 [K]
    h_conv_ins= 15 [W/m2/K]
    epsilon_ins= 0.5
```

```
    h_conv_B= 15 [W/m2/K]
    epsilon_B= 0.5
    h_conv_HMo= 4.8 [W/m2/K]
    h_conv_HMi= 15 [W/m2/K]
    epsilon_HMo= 0.5
    A_ins= 0.316 [m2]
    A_B= 0.0292 [m2]
    L_HM= 0.02 [m]
    lambda_HM= 0.0295 [W/m/K]
    V= 0.05 [m3]
    n_stack= 1
  </wall>
</BOUNDARIES>
```

The following table contains the parameter definitions, as necessary to evaluate eqs. 17.22- 17.25.

ndot_gas=*d*	molar flux of cavity gas [mol/s]
T_gas_initial=*d*	initial guess for cavity gas temperature [K] (default: *T_gas_in*)
T_gas_in=*d*	cavity gas inlet temperature [K]
T_amb=*d*	temperature of HotBox surroundings [K]
T_B=*d*	catalytic afterburner outer surface temperature [K]
h_conv_ins=*d*	convective htc of monolith insulation [W/m^2 K]
h_conv_B=*d*	convective htc of catalytic afterburner [W/m^2 K]
h_conv_HMo=*d*	convective htc of HotBox outer surface [W/m^2 K]
h_conv_HMi=*d*	convective htc of HotBox inner surface [W/m^2 K]
epsilon_ins=*d*	emissivity of monolith insulation (0<= emiss <=1)
epsilon_B=*d*	emissivity of catalytic afterburner (0<= emiss <=1)
epsilon_HMo=*d*	emissivity of HotBox outer surface (0<= emiss <=1)
A_ins=*d*	monolith insulation outer surface area [m^2]
A_B=*d*	catalytic afterburner outer surface area [m^2]
L_HM=*d*	HotBox thickness [m]
lambda_HM=*d*	HotBox thermal conductivity [W/m K]
V=*d*	HotBox volume [m^3]
n_stack=*d*	number of stacks in the HotBox

17.4.6 The <INITIAL> tag

This section defines the initial conditions of the temperature field. There are two possibilities: either specify a uniform temperature (**T**) or load a monolith output file of a previous run (**file**).

T0=*d*	initial temperature
FILE=*s*	name of a restart file (e.g. restart.plt or monolith001.plt)

If using a monolith (or restart) file from a previous run, ensure that the grid definition has not changed. The grid defined in the **<GRID>** section and the grid of the output file must be the same. The file also contains information about the output time. The initial time will be set automatically to the value read in the file. Hence, the user can restart an aborted calculation and continue the calculation without any further changes by

```
<INITIAL>
  file= restart.plt
</INITIAL>
```

If the calculation shall start at time 0 using the temperature field, the user must change the header information of the file prior to running the simulation.

17.4.7 The <SOLUTION> tag

This section defines solution parameters that control the time step of the simulation. The most important and required parameter in this section is the end time (**tend**) of the simulation. All other parameters are summarized below:

tini=*d*	starting time [s] (default: 0)
tend=*d*	end time [s]
dt_ini=*d*	initial time step [s] (default: 0.01)
dt_max=*d*	maximum time step [s] (default: 1)
delta_T=*d*	maximum temperature change in each time step [K]
method=*i*	method parameter (LSODE only) (default: 0)

17.4.8 The <OUTPUT> tag

Define the times at that an output file is generated in the **<OUTPUT>** section of the input file. Basically, specify the time(s) by writing values without an option name. The definition

```
<OUTPUT>
  5   10   20
</OUTPUT>
```

generates three output files ***monolith001.plt***, ***monolith002.plt***, ***monolith003.plt*** written at time 5, 10, and 20 seconds, respectively.

If output is desired in constant intervals, you can also use a do-loop instead of a series of numbers.

```
<OUTPUT>
  <DO> from=10 to=100 step=10 </DO>
</OUTPUT>
```

The two forms of stating output times can be mixed as long as the times appear in ascending order. The output files receive a number as an extension according to their position in this section. Hence, in case of a restart numbering continues at the appropriate position; file numbers of times less than the restart time will be skipped. An output at the end time of the simulation is added to the list automatically.

Furthermore, a binary option **write_restart** can be specified. If activated, the file ***restart.plt*** is (re-)written at each time step of the simulation. It is of the same format as the ***monolith***.plt*** files. It may be useful for time consuming simulation. This files enables an in situ monitoring of the simulation and – in case of an interrupt – a restart at the latest point.

The members of the **<OUTPUT>** tag are summarized in the following table

<**DO**>	generate equistant output times
output time	time for that output shall be written [s]
WRITE_RESTART=*b*	write a file restart.plt after each iteration (default: no)

whereas the **<DO>** loop contains

from=*d*	start time [s]
to=*d*	end time [s]
step=*d*	time between two outputs [s]

17.4.9 The <CHANNELS> tag

This section acts as an interface between DETCHEM^{MONOLITH} and the channel simulation. It basically determines how the input data for the channels is generated and how the returned heat source terms are converted. Furthermore, this section also allows to define single or multiple channel models.

The **<CHANNELS>** section expects a tag for each channel type. Use as tag name the name of the channel as it is defined your channel input (see chapter 15), e. g. if the channel definition starts with

```
<CHANNEL Version=2.0 name=channel1>
```

then your **<CHANNELS>** section may read

```
<CHANNELS>
  <channel1>
    ch/cm2 = 62
  </channel1>
</CHANNELS>
```

If only one channel type is defined, you can use any tag name. (Note: in this case you may continue using version 1.5.4's tag **<TYPE>**.)

Within each channel type's tag you may specify the following options

ch/cm2=*d*	channel density - number of channels per cm^2
ch/m2=*d*	channel density - number of channels per m^2
cpsi=*d*	channel density - number of channels per square inch
reverse=*b*	reverse flow direction (default: no)
inlet=*s*	name a channel, whose outlet is used as inlet for this channel (default: none)
V_factor=*d*	ratio of new inlet velocity / old outlet velocity (default: 1)

It is mandatory to define the channel density σ by using one of the three options **ch/cm2**, **ch/m2**, or **cpsi**. The channel density is used as conversion factor for the heat source term of the channel (energy per unit length) into the heat source term of the monolith (energy per unit volume). However, this is only applied to domains of the monolith that have been marked **monolithic** in the **<GRID>** definition.

The flow direction of the channel type can be reversed using the **reverse** option. By this means, you can simulate counter-flow configurations as in a heat exchanger, e. g.

```
<CHANNELS>
  <channel1>
    ch/cm2 = 31
  </channel1>
  <channel2>
    ch/cm2 = 31
    reverse=y
  </channel2>
</CHANNELS>
```

It is also possible to define an outlet of one channel as inlet of another. Use the **inlet** member with a name of another channel. It is only important that the outlet channel appears before the inlet channel in the list of channels at the beginning of the input file ***monolith.inp***. Hence, the heat exchanger model with exhaust gas reuse would read

```
<CHANNELS>
  <channel1>
    ch/cm2 = 31
  </channel1>
  <channel2>
    ch/cm2 = 31
    reverse=y
    inlet=channel1
    v_factor=1
  </channel2>
</CHANNELS>
```

An additional parameter **v_factor** can be used to account for velocity changes due to different size of the channels. The outlet velocity will be multiplied by this factor to give the new inlet velocity. Note: the inlet data for all channels of one type is the same if they use an outlet as their inlet. Thus, first all channels of the outlet channel type are simulated. Then their outlets are averaged and used as inlets for the inlet channel type. Therefore you can think of the outlet-inlet coupeling realized by one big pipe instead of a local loop of each channel.

17.4.10 The <CHOOSE> tag

This section controls the choosing of representative channels as it is described in chapter 17.3.2. The members are summarized below

delta_T=*d*	absolute variation of temperatures [K] (default: 20)
delta_cap=*d*	absolute variation of capacities (default: 0.05)
pvar_v=*d*	relative variation of velocities [%] (default: 10%)
pvar_Y=*d*	relative variation of mass fractions [%] (default: 10%)
memorize=*b*	memorize channel vector for comparision with later time steps (default: no)

For a simulation with constant inlet conditions, this section may look as follows

```
<CHOOSE>
  delta_T = 5
  memorize= y
</CHOOSE>
```

The larger the values are chosen the less channels will be similuated in detail. So the user has to decide between speed of calculation and accuracy of the result. However, in most cases it is sufficient if the parameters are chosen in such a way that a number of two to ten representative channels is selected (depending on the gradients inside the monolith). The **memorize** option can speed up the simulation dramatically – especially if the system is approaching equilibrium. However, since the output may become more "stepped", it is recommended to decrease the other parameters. The **memorize** option can not be used in case of simulations with more than one channel type.

17.4.11 The <INLET> tag

There are two types of inlet conditions: **constant** and **variable**. Their specification requires a different syntax. Therefore, the user needs to define the **type** member first. This can be done directly within the **<INLET>** tag.

Constant inlet conditions

In case of constant inlet conditions, the definition is made directly in the input file. The user needs to specify inlet velocity, temperature and species composition. An example is shown below

```
<INLET type=constant>
  u0=  2.            # velocity in m/s
  T0=  1200.         # temperature in K
  <MOLEFRAC>
    NO2       0.1    # mole fraction
    N2        *      # balance
  </MOLEFRAC>
</INLET>
```

<MASSFRAC>	species mass fractions
<MOLEFRAC>	species mole fractions
u0=*d*	velocity at channel inlet [m/s]
T0=*d*	inlet temperature [K]
user=*b*	use user defined inlet conditions (default: no)

Note that the velocity is the inlet velocity at the channel entrance at the given temperature. The user needs to convert units like standard liters per minute (slpm) manually. When assigning mole or mass fractions, one of the species can be marked with an asterisk (*****) to balance the sum to unity.

Use the **user** option to activate user defined inlet conditions (see chapter 17.5.1).

Variable inlet conditions

Inlet conditions may vary in space and time. In either case, variable inlet conditions need to be defined in an external input file that will be read sequentially during the simulation. Therefore the definition in ***monolith.inp*** is rather short.

```
<INLET type=variable>
  file     = inlet.inp   # name of the external file
  velfactor= 2.0         # velocity conversion factor
</INLET>
```

FILE=*s*	name of inlet input file
velfactor=*d*	conversion factor for inlet velocities (default: 1)
user=*b*	use user defined inlet conditions (default: no)

The format of the external file, e. g. ***inlet.inp***, will be described later in this chapter (see 17.4.14). Quite frequently, the inlet velocity needs to be converted due to diameter change of the channels. Therefore the member **velfactor** can be assigned a numerical value. The velocities in the external file are multiplied with this factor.

Negative velocities reverse the channel flow. This may be used for simulations with alternating flow direction or even couter-flow configurations.

All inlet data can be altered by user defined functions. If the user wants to provide own code (see chapter 17.5.1), activate the **user** option.

17.4.12 The <POSTPROCESS> tag

The output files for specified channels (i. e. ***outg***.plt*** and ***outs***.plt***) are written during postprocessing. Usually, as the name implies, the user will do postprocessing after a simulation. If you want to take a detailed look at single channels at a given time, you just need an output file, e. g. ***monolith001.plt***, at that time and add the following section to your input file ***monolith.inp***

```
<POSTPROCESS>
  file     = monolith001.plt
  0.0                        # position of the channel (2d case)
</POSTPROCESS>
```

Running the executable of DETCHEM^{MONOLITH} again will generate the files ***outg001001.plt*** and ***outs001001.plt*** and then the execution stops.

Thus, give a file name using the **file** member that contains the monolith data at a given time. Note: you can also use the file ***restart.plt***. Also specify the positions of all channels for that you want to generate output. In case of a 2d simulation one coordinate is required per channel, for a 3d simualtion there must be two coordinates. For each location and each channel type, there will be an output file ***outg###***.plt*** and in case of a catalytic channel ***outs###***.plt***. If the **file** is named ***monolith###.plt*** then the extension ***###*** is automatically added to the channel output files. The extension *** is incremented for each channel.

In case of a large number of monolith output files it might become quite tedious to do postprocessing for each channel individually. So if you want to create a movie with hundreds of monolith files and their corresponding channel output files, you may choose a different approach. You can force to postprocess the channel calculation at each time step of the simulation. This is done by adding a **<POSTPROCESS>** section without a **file** member.

```
<POSTPROCESS>
  0.0           # postprocess this channel in each time step
</POSTPROCESS>
```

However, this procedure is only recommended if a large number of channel output files has to be generated at the same time as the monolith output, because it can not take advantage of the **memorize** option.

17.4.13 The <STORAGE> tag

DETCHEM$^{\text{MONOLITH}}$ can also be used to simulate monolithic catalyst with a storage compound. The storage medium must be represented by a set of surface species that slowly change their coverages.

An example will illustrate how the simulation of storage systems is set up. Suppose you have a system with a platinum (for the fast catalytic reactions) and barium (for the slow storage reactions) compounds on the surface. Define two surface types in the species list.

```
<SPECIES>
  ...
  <SURFACE name="Pt" mol/cm2=2.72e-9>
    (Pt)  O(Pt)  NO(Pt)  NO2(Pt)  N(Pt)  ...
    <INITIAL>
      (Pt)   *
    </INITIAL>
  </SURFACE>

  <SURFACE name="Ba" mol/cm2=1e-8>
    BaO  BaNO3
    <INITIAL>
      BaO    *
    </INITIAL>
  </SURFACE>
</SPECIES>
```

Set up a reaction mechanism that contains reactions on both platinum and barium.

```
<MECHANISM>
  ...
  <SURFACE name="storage">
    file = ...
  </SURFACE>
</MECHANISM>
```

Then add a new **<STORAGE>** tag in ***monolith.inp***

```
<STORAGE>
  surface  ="Ba"
  mechanism="storage"
</STORAGE>
```

Give the names of the surface type and of the mechanism as they were defined in the species and mechanism section before, respectively. The coverages of the surface type named by the **surface** member will be added to the list of variables calculated by DETCHEM$^{\text{MONOLITH}}$. The time integration will be performed by using the mechanism defined with the **mechanism** member.

If the catalytic reactions do not interact with the storage reactions, it is also possible to define two mechanisms, one for the channel simulation that contains all reactions and one for the time integration that contains only the storage reactions.

```
<MECHANISM>
  ...
  <SURFACE name="Pt-Ba">
    file = "Pt-mech"
    file = "Ba-mech"
  </SURFACE>

  <SURFACE name="Ba_only">
    file = "Ba-mech"
  </SURFACE>
</MECHANISM>
```

17.4.14 External inlet file

For variable inlet conditions it is necessary to prepare another input file. It can have any name (e. g. ***inlet.inp***). Use this name as reference for the **file** member in the **<INLET>** section of the input file.

The inlet file consists of two sections: **<INLETINFO>** and **<INLETDATA>**. An example is shown below

```
<INLETINFO>
  u = 0.8       # constant velocity
  T = list      # temperatures appear in list
  <MOLEFRAC list>
     CH4
     O2
     N2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time  Temp   CH4    O2    N2
    0   900   0.0   0.2   0.8
    5   900   0.1   0.2   0.7
   60   800   0.1   0.1   0.8
  120   700   0.1   0.1   0.8
</INLETDATA>
```

In **<INLETINFO>** the user needs to specify which data is read from a list or file and which is constant. Assign one of the following options to the members **u** and **T** for velocity and temperature, respectively:

- a *numerical value* for a constant,
- **list** for data that is listed in **<INLETDATA>**, or
- a *TECPLOT variable name* for spatially varying data in separate files.

The species composition can be given in mole (**<MOLEFRAC>**) or mass fractions (**<MASSFRAC>**). Examples of the syntax for each case are shown below.

- constant mass or mole fractions

```
<MOLEFRAC const>    # define composition as usual
   CH4   0.1
   O2    0.1
   N2    *           # * may be used for balance
</MOLEFRAC>
```

- data listed in **<INLETDATA>**

```
<MOLEFRAC list>     # give the species in the same order
   CH4              # as they appear in the list
   O2
   N2
</MOLEFRAC>
```

- spatially varying data in TECPLOT files.

```
<MASSFRAC file>     # give the name of the species
   CH4   ych4       # and their corresponding name
   O2    yo2        # in the TECPLOT files
   N2    yn2
</MASSFRAC>
```

Note that the options list and TECPLOT file cannot be used at the same time. Therefore, provide all data that is not constant either in the list or in separate TECPLOT files.

The section **<INLETDATA>** contains the values of all data that has been assigned the **list** option or the file names of all TECPLOT files. Although it is not required, it is strongly recommended to use a column oriented format for better readability. The first column contains the start time of the following conditions. These conditions are valid until the next time in the list or until the simulation ends. The following columns contain the velocity, temperature, and species composition in that order. If they are constant, the column is skipped.

An example for listed inlet data is shown at the beginning of this chapter. An interesting application is a simulation of a monolith with alternating flow direction.

```
<INLETINFO>
  u = list      # alternating velocity
  T = 900
  <MOLEFRAC list>
     CH4   0.1
     O2    0.2
     N2    *
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time  velocity
    0    2
   60   -2      # switch flow direction
  120    2
  180   -2
</INLETDATA>
```

An example with spatially varying inlet profiles is shown below

```
<INLETINFO>
  u = vel            # assign the name of a TECPLOT variable
  T = temp
  <MOLEFRAC file>
     CH4   ych4
     O2    yo2
     N2    yn2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time  TECPLOT file name
    0  inlet001.dat
   10  inlet002.dat
   20  inlet003.dat
</INLETDATA>
```

The TECPLOT files must be of the same format (number and order of the variables). Only the data of the first zone is read. In case of a DETCHEM$^{\text{MONOLITH}}$ 2d simulation, the first variable in the TECPLOT files must be a coordinate. According to this coordinate the data is interpolated to the monolith grid.

For 3d simulations, the 2d inlet grid must be identical to the radial grid of the monolith. Use therefore DETCHEM$^{\text{GRIDGEN3D}}$ to generate the monolith grid from the TECPLOT grid (see chapter 16).

In **<INLETDATA>** the user also may specify two options between two data lines (not before the first data line)

dtmax=*d*	maximum step size (use in case step size shall vary with inlet data) (default: difference to next time in list)
skip_failed=*b*	skip inlet data in case it produces erroneous results (default: no)

The parameter **dtmax** may be useful in case of time dependent step size, for instance if there is a cycle with a long and slow storage phase and a short and fast regeneration phase. The binary option **skip_failed** may be activated in case the simulation is not successful for a few sets of inlet data. If activated, the inlet data of unsuccessful simulation time steps is skipped. The simulation continues with the next time step. Although the result is erroneous at this particular time step, the simulation may continue instead of abortion due to an error.

17.5 User defined routines

17.5.1 User defined inlet conditions

Provide your own Fortran code for user defined inlet conditions. A dummy subroutine ***MIN_UserIN*** can be found within the file

($DETCHEM_DIR)/MONOLITH/code/MIN_UserIN.f

The user may alter the inlet data read from the input section or the external input file. Change the values for velocity, temperature, and/or species composition (make sure that fractions add to unity) depending on time and position at the inlet face. An example of a definition of a counter current flow (adjacent channels do not alternate) is shown in the example code.

```
      subroutine MIN_UserIN(time,y,z,channelID,isMoleFrac,u,T,mfrac)
      implicit none
c     in:
c       time      : current time [s]
c       y,z       : coordiantes of the inlet face [m]
c       channelID : ID of this channel (multi-channel model)
c       isMoleFrac: .true. if mfrac contains mole fractions
c                   .false. if mfrac contains mass fractions
```

```
c     in/out:
c       u         : inlet velocity [m/s]
c       T         : inlet temperature [K]
c       mfrac(*)  : mass or mole fraction of gas-phase species

c     variables default to values set in <INLET> section or file

      double precision time,y,z,u,T,mfrac(*)
      integer channelID
      logical isMoleFrac

c     example:
c     counter current flow
c     change flow direction for channels in the lower half space

      if (y .lt. 0) u=-u

c     end subroutine MIN_UserIN
      end
```

After supplying program code, recompile DETCHEM^{MONOLITH} by calling ***make*** in the MONOLITH directory or DETCHEM root directory.

17.6 Output

DETCHEM^{MONOLITH} creates a number of output files and screen output. By postprocessing (see chapter 17.4.12) you can also generate channel output for selected channels.

17.6.1 Screen output

The screen output lets you monitor the integration progress. If channel monitoring is deactivated, the output will look like follows

```
*************************************************************************
****                        DETCHEM MONOLITH                         ****
****             S. TISCHER, C. CORREA, O. DEUTSCHMANN               ****
****                 VERSION 2.0.002 ** 2004/08/11                   ****
*************************************************************************
 Number of channels :  1
 CHANNEL  1 (# 10)
    at position   0.0095
    u0=  2. T0=  1200.
 Iterations: 61    Errors: 0
 Memorize channel # 10 as # 11
 -----------------------
 Time =   0. s
   Maximum temperature in the monolith =  298.15
   Minimum temperature in the monolith =  298.15
 Next timestep:  0.03
 Number of channels :  0
 -----------------------
 Time =   0.03 s
   Maximum temperature in the monolith =  302.761427
   Minimum temperature in the monolith =  298.150001
 Next timestep:  0.0975836823
 Number of channels :  0
 -----------------------
 Time =   0.127583682 s
   Maximum temperature in the monolith =  316.949506
   Minimum temperature in the monolith =  298.150004
 Next timestep:  0.103167959
 Number of channels :  1
 CHANNEL  1 (# 10)
    at position   0.0095
    u0=  2. T0=  1200.
 Iterations: 83    Errors: 0
 Memorize channel # 10 as # 11
 -----------------------
...
```

It gives information about the current time and the minimum and maximum temperature in the monolith. By observing the temperatures one may get hints about the conversion of the solution (you can abort the calculation in case the temperature jumps to sensless values). Furthermore, the number of representative channels a monitored. For each channel, the position, its inlet velocity and inlet temperature are shown. After the channel simulation is finished, the number of iterations and the error flag is displayed. DETCHEM^{MONOLITH} has got some strategies to deal with erroneous channels, but if the error flag keeps to be non-zero for a longer time, the simulation may be aborted. Finally, if the **`memorize`** option is activated, the memorized channels are shown.

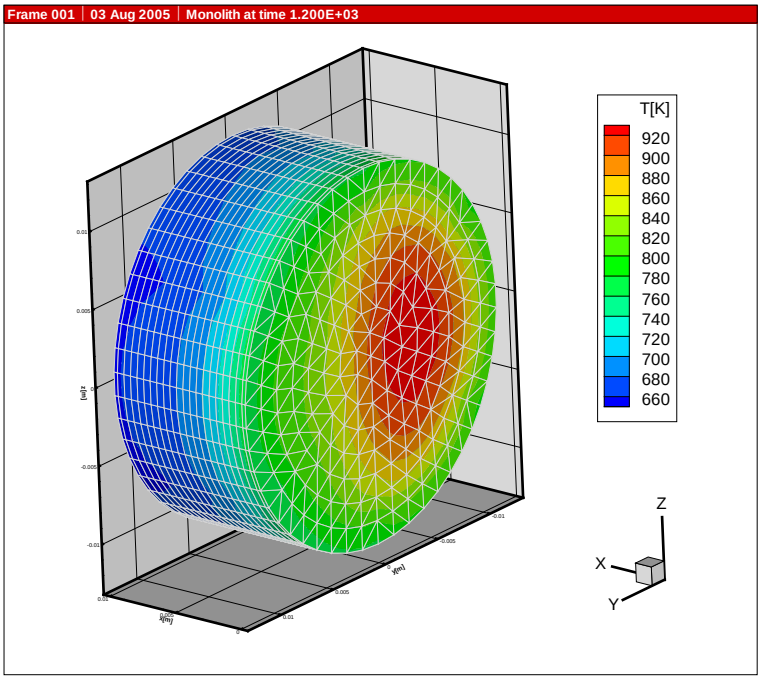
On successful execution, the simulation stops when the end time is reached. Otherwise, an error message will be displayed.

17.6.2 *monolith***.plt*

At each time defined in the **<OUTPUT>** section a file ***monolith***.plt*** is written. The file is in TECPLOT format and can be used to draw two- or three-dimensional contour plots of the temperature field of the monolith.

The file header contains the time when the file was written. The user may change this time when this file shall be used as a restart file of a new simulation. The variables of this file are the two or three coordinates and the temperature. In case of the storage model, the coverages of the storage species follow.

The data is organized in two zones. The first zone contains the values at the cell centers without a grid definition. This data is read for restart. The second zone contains the interpolated values at the grid nodes and the grid definition. This data is plotted. An example of a 3d TECPLOT illustration is shown below.



17.6.3 *restart.plt*

The file ***restart.plt*** contains the same data as a ***monolith***.plt*** file. If the **write_restart** option is activated, the file will be written in each time step. Therefore it always contains the state of the monolith at the latest time step.

17.6.4 *grid.plt*

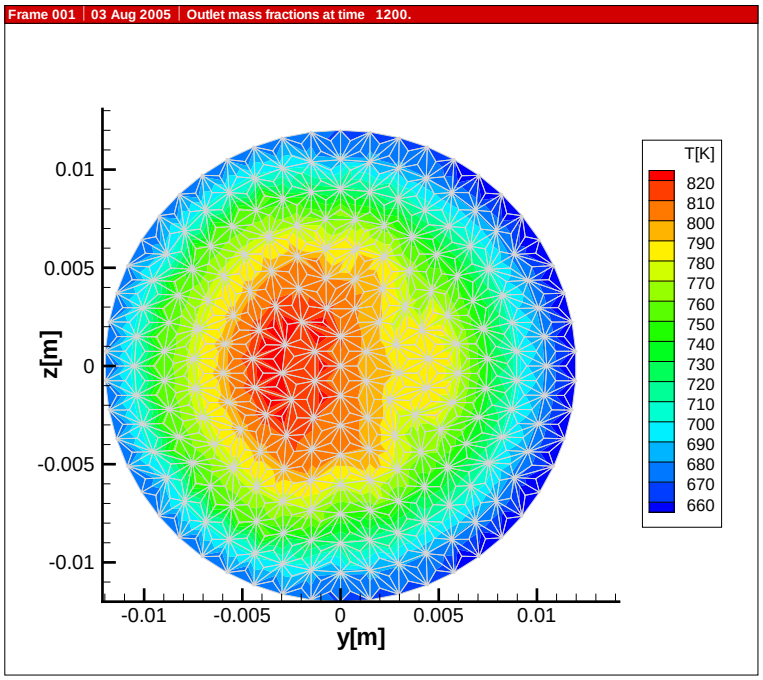
The file ***grid.plt*** is writting at the beginning of program execution. It contains the same grid information as the second zone of the file ***monolith***.plt***, but without temperatures or coverages. It can be used to look at the grid properties before the simulation starts. By this means, you may discover errors or irregularities in the grid definition without waiting for the result of the simulation.

17.6.5 *outlet***.plt*

Especially in the case of large gradients or inhomogeneous inlet conditions, you may want to examine the outlet distribution of temperatures and species. Every time a file ***monolith***.plt*** is written, also file named ***outlet***.plt*** is generated. It is a one- or two-dimensional TECPLOT file, which can be used to draw line or contour plots of the outlet face.

The variables contained in the file are: the coordinates, the outlet gas temperature, and the outlet mass fractions of the gas-phase species. Furthermore, there is a variable `SIMFLAG` that indicates by which representative channel this channel was represented. Channels with the same `SIMFLAG` default to the same outlet data.

An example of an TECPLOT visualization of a two-dimensional outlet face is shown below.



17.6.6 *global.plt*

The file ***global.plt*** summarizes the global inlet and global outlet composition at each time step (not only the times listed). The file is in TECPLOT format and can be used to draw line graphs of gas-phase temperature and mass fractions. However, this format also allows for simple import into other spreadsheet programs (e. g. Microsoft Excel).

The variables in this file are: time, inlet temperature, inlet mass fractions, outlet temperature, outlet mass fractions, and an error flag. You can use this file for simple calculation of conversions and selectivities.

It is recommended to pay attention the the error flag. If the flag is not zero, an error in one or more channel simulations at this time step has occurred. The outlet composition at this time step is most likely wrong. If this only happens sparsely, you may delete the wrong line. However, if the error flag keeps to be set, the simulation was not successful.

A new file ***global.plt*** is written on each execution of DETCHEM^{MONOLITH} (except at postprocessing of a single file). If you want to restart an earlier calculation, save ***global.plt*** with a different name before calling the executable again.

17.7 Examples

There are several examples in the distribution – two- and three-dimensional simulations with and without varying channels. Each directory contains several input files (****.inp***), species and mechanism information as well as the databases ***thermdata*** and ***moldata***. The ***.plt*** files are generated on execution.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

17.8 Running the tool

There are several executables for DETCHEM^{MONOLITH} in the ***($DETCHEM_DIR)/bin*** directory. They are named in the form ***monolith_C_S_P***

where

- ***C*** – channel model (currently: ***channel*** or ***plug***)
- ***S*** – solver (***limex***, ***lsode***)
- ***P*** – parallelization (***serial***, ***parallel***)

The parallel version of DETCHEM$^{\text{MONOLITH}}$ can be used on multi-processor machines to split the single channel calculations between the processors. The channel calculations themselves are not parallelized. That is, if you have two processors and four channels have to be simulated in a time step, each processor will simulate two channels. If there are more processors than channels to be simulated in any time step, then some processors will remain idle.

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***monolith_C_S_P***.
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/monolith_C_S_P*** and call ***./monolith_C_S_P***.

The latter option does not require setting of system variables and is therefore less system specific.

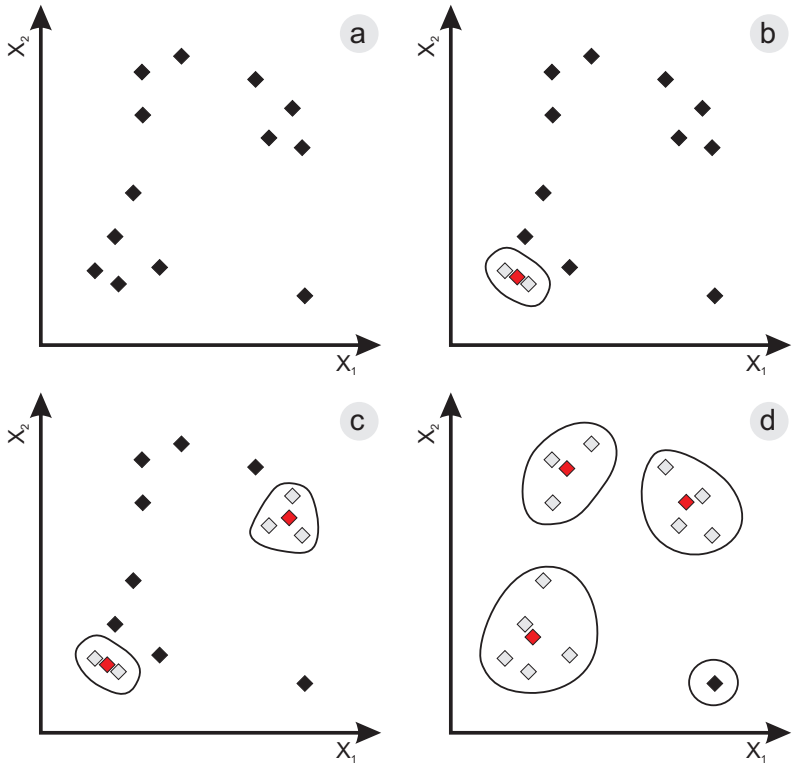


Figure 17.1: Agglomerative cluster algorithm. (a) Given set of data points. (b) Join two data points into one cluster. (c) After the third agglomeration step two clusters have been formed. (d) Final representation of the data by four clusters.

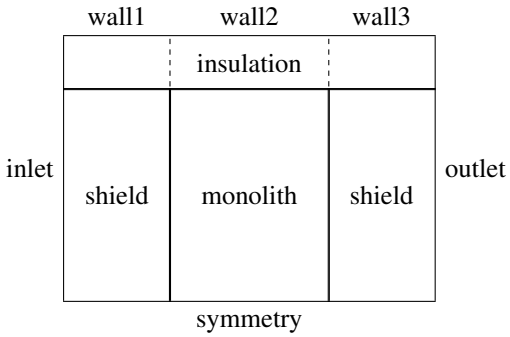


Figure 17.2: Example for a 2d domain definion

Chapter 18

DETCHEM$^{\text{RESERVOIR}}$

18.1 Introduction

DETCHEM$^{\text{RESERVOIR}}$ serves as a transient wrapper for single channel simulations with storage effects. It can be used in conjunction with DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$. Unlike in DETCHEM$^{\text{MONOLITH}}$, the wall temperature profile of the channel does not change in time. Only the concentrations of selected surface species (those of the storage medium) are transient. For the storage medium we must assume that the rate of reaction is slow enough to be decoupled from the flow field. That is, the time scale for storage reactions must be larger than the residence time of the gases inside the channel. For surface species involved in these reactions, DETCHEM$^{\text{RESERVOIR}}$ solves the following transient species equation

$$\frac{\mathrm{d}c_i^{\text{storage}}}{\mathrm{d}t} = \dot{s}_i(c^{\text{gas}}, c^{\text{surf}}, c^{\text{storage}}) \tag{18.1}$$

The concentrations of the gas-phase species c^{gas} are calculated by a quasi steady-state channel simulation using DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$, whereas for the non-storage surface species c^{surf} steady-state is also assumed. Thus, for each time step DETCHEM$^{\text{RESERVOIR}}$ calls DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$ with fixed storage concentrations c^{storage}. After the single channel simulation, the locally resolved gas-phase concentrations c^{gas} are passed back to the reservoir model. Then equation 18.1 is integrated for a given time step and the cycle starts from the beginning again. The principle of the decoupling of the two codes is illustrated in figure 18.1.

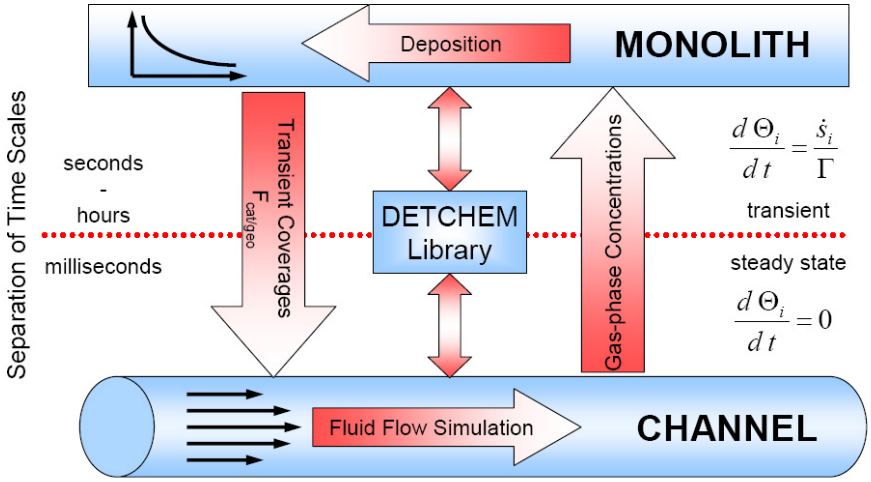


Figure 18.1: Schematic communication between DETCHEM$^{\text{RESERVOIR}}$ and single channel simulation

18.2 User input

The input required for DETCHEM$^{\text{RESERVOIR}}$ is similar to DETCHEM$^{\text{MONOLITH}}$, however a few sections can be dropped due to the simplifications of the model. DETCHEM$^{\text{RESERVOIR}}$ requires one input file named ***reservoir.inp***. An example is shown below.

```
{include channel.inp}

 <RESERVOIR>

 <GRID>
   <XDOMAIN>
     from=  0.
     to=  200.d-3
     cells= 40
     aspratio=  1.
   </XDOMAIN>
 </GRID>

 <SOLUTION>
   time =  130
   h = 2.6
   T/K = 623
 </SOLUTION>

 <OUTPUT>
  <do> from 5 to 55 step 5 </do>
  59.9
  60.1
  64.9
  <do> from 65 to 125 step 5 </do>
  write_restart=yes
 </OUTPUT>

 <STORAGE>
   surface="BaCO3"
   mechanism="Speicher"
 </STORAGE>

 <INLET type=variable>
   file=inlet.inp
 </INLET>

 </RESERVOIR>
```

The file ***reservoirinp*** must start with the definitions necessary for the channel calculation. This implies that first, the species definiton, the mechanism definition, the optional washcoat deinition, and the channel definition appear first in the input file. It is probably most convenient to use the same input file as for the channel simulations and include the channel input file by an **include** statement. The DETCHEM$^{\text{RESERVOIR}}$ specific input data follows enclosed in **<RESERVOIR>** tags. The options inside this section are listed in the following table.

Version=*s*	input file version (not evaluated)
<**GRID**>	define grid (must be the first tag)
<**SOLUTION**>	define solver options
<**OUTPUT**>	define output options
<**POSTPROCESS**>	postprocessing
<**STORAGE**>	storage model
<**INLET**>	define inlet conditions

<GRID> – grid definition

For the calculation of the transient concentraition profiles, the channel needs to be discretized in axial direction. This is done by the (one-dimensional) grid definition. The **<GRID>** tag must contain one or more **<XDOMAIN>** sections. In most cases, one such section should be enough. However, if the grid should be non-uniform, the use of more than one **<XDOMAIN>** definitions may be useful. The **<XDOMAIN>** section can also be used to override the default temperature and catalyst distribution ($F_{\text{cat/geo}}$) in this section. e. g.

```
<GRID>
  <XDOMAIN>
    from    =  0
    to      =  0.02
    cells   = 10
    aspratio=  1.05    # non-uniform grid at entrance
  </XDOMAIN>
  <XDOMAIN>
```

```
    to      =  0.1
    cells   = 20
    T/K     = 500      # override temperature from <SOLUTION>
    <Fcatgeo>          # override Fcat/geo from <SURFACE-MODEL>
       BaCO3   20
    </Fcatgeo>
  </XDOMAIN>
</GRID>
```

Each domain can be divided into several cells. In each cell represents one set of transient concentrations. Therefore, the number of cells should be chosen accordingly to the expected concentration gradients. The user must further make sure that the total length of the discretized grid (the **to** parameter in the last domain) equals the length of the channel. The members in each **<XDOMAIN>** tag are:

from=*d*	axial coordinate of start of the domain (default: end of the previous domain)
to=*d*	axial coordinate of end of the domain
cells=*i*	number of grid cells
aspratio=*d*	aspect ratio, i.e. ratio of the lengths of two adjacent cells (default: 1)
T/K=*d*	wall temperature (overriding definition from **<SOLUTION>**)
<Fcatgeo>	catalyst composition (overriding definition from **<SURFACE-MODEL>**)

<SOLUTION> – operation parameters

The operation conditions of the DETCHEM$^{\text{RESERVOIR}}$ simulation are defined within the **<SOLUTION>** tag. These parameters are the total time of simulation, the maximum step size and the (isothermal) wall temperature. Alternatively to a constant wall temperature, a linear temperature profile can be defined. Moreover, if a previous simulation is to be continued, you can use an exisiting output file for initialization.

All available options are summarized below:

time=*d*	end time [s]
h=*d*	integration time step [s]
T/K=*d*	temperature [K]
T_in=*d*	temperature at inlet [K]
T_out=*d*	temperature at outlet [K]
dT/s=*d*	temperature ramp for TPD simulations [K/s]
dT/min=*d*	temperature ramp for TPD simulations [K/min]
restart=*s*	restart file name

<OUTPUT> – output options

Several output files are written during a DETCHEM$^{\text{RESERVOIR}}$ simulation. Use the **<OUTPUT>** tag to define the time steps for wich individual output files are desired. The time steps can be listed individually (just give the numeric value of the time in seconds) or generated with regular frequency (use a **<DO>** tag). The times listed must be in ascending order.

Furthermore, there is an option to generate an output file ***restart.plt*** that contains the latest status of the simulation. This file might be useful in case a simulation needs to be restarted after an interruption.

The option within the **<OUTPUT>** tag can be summarized as

<**DO**>	generate equistant output times
output time	time for that output shall be written [s]
WRITE_RESTART=*b*	write a file restart.plt after each iteration (default: no)
reservoir_all=*b*	generate output in reservoir.plt for all time steps

and the **<DO>** parameters are

from=*d*	start time [s]
to=*d*	end time [s]
step=*d*	time between two outputs [s]

<STORAGE> – transient species and mechanism

The main purpose of simulations with DETCHEM$^{\text{RESERVOIR}}$ is the storage process. The user must choose one surface type for which the transient concentrations are calculated. For instance, when a catalytic converter contains platinum and barium as catalytic active components, the platinum surface may be considered in quasi steady state, whereas the slow NOx storage processes occur on the barium sites. In this case, choose the barium surface as storage surface (use the name of the surface as it is defined in the **<SPECIES>** section at the

beginning of the input file). The reaction mechanism containing the relavant reactions for the storage must also be referenced. In most cases this mechanism can be identical with the complete surface reaction mechanism. Refer to it by the name as it is defined in the **<MECHANISM>** section at the beginning of the input file.

surface=*s*	name of the storing surface
mechanism=*s*	name of the storage mechanism

<INLET> – inlet information

The transient nature of DETCHEM^{RESERVOIR} will usually require transient input data. However, it is also possible to define constant input data directly inside this input file.

In the latter case, just specify the inlet velocity and the species composition, e. g.:

```
<INLET type=constant>
  u0=  2.            # velocity in m/s
  <MOLEFRAC>
    NO2       0.1    # mole fraction
    N2        *      # balance
  </MOLEFRAC>
</INLET>
```

As an alternative to mole fractions, mass fractions can be defined using the tag **<MASSFRAC>**.

In most cases, however, the inlet conditions will vary in time. This is declared like in the following example:

```
<INLET type=variable>
  file     = inlet.inp   # name of the external file
  velfactor= 2.0         # velocity conversion factor
</INLET>
```

Instead of declaring all information inside this input file, the transient inlet information is stored in an external file, e. g. ***inlet.inp***. The format of this external file will be described later in this chapter. Sometimes, the inlet velocity needs to be converted due to changes in the diameter of the channel or in the temperature. Therefore the member **velfactor** can be assigned a numerical value. The velocities in the external file are multiplied with this factor. Note: Use for convenience this conversion factor in case your external file contains mass or volume fluxes instead of velocities.

All inlet data can also be altered by user defined functions. If the user wants to provide own code (see chapter 17.5.1 for DETCHEM^{MONOLITH}), activate the **user** option.

Thus, the inlet options can be summarize as follows:

type=*s*	type of inlet condition: constant or variable
FILE=*s*	name of inlet input file
velfactor=*d*	conversion factor for inlet velocities (default: 1)
user=*b*	use user defined inlet conditions (default: no)

External inlet file

For variable inlet conditions it is necessary to prepare another input file. It can have any name (e. g. ***inlet.inp***). Use this name as reference for the **file** member in the **<INLET>** section of the input file.

The inlet file consists of two sections: **<INLETINFO>** and **<INLETDATA>**. An example is shown below

```
<INLETINFO>
  u = 0.8       # constant velocity
  <MOLEFRAC list>
     CH4
     O2
     N2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time   CH4    O2    N2
    0   0.0   0.2   0.8
    5   0.1   0.2   0.7
   60   0.1   0.1   0.8
  120   0.1   0.1   0.8
</INLETDATA>
```

In **<INLETINFO>** the user needs to specify which data is read from a list or file and which is constant. Assign a constant value or the keyword **list** to the parameter **u** for velocity and to **<MASSFRAC>** or **<MOLEFRAC>** for the species composition.

The section **<INLETDATA>** contains the values of all data that has been assigned the **list** option. Although it is not required, it is strongly recommended to use a column oriented format for better readability. The first

column contains the start time of the following conditions. These conditions are valid until the next time in the list or until the simulation ends. The following columns contain the velocity and species composition in that order. If they are constant, the column is skipped.

In **<INLETDATA>** the user also may specify further options between two data lines (not before the first data line)

velocity	velocity [m/s]
mass or mole fraction	mass or mole fractions
dtmax=d	maximum step size (use in case step size shall vary with inlet data) (default: difference to next time in list)
skip_failed=b	skip inlet data in case it produces erroneous results (default: no)
<**SURFACE-MODEL**>	change <SURFACE-MODEL> options

The parameter **dtmax** may be useful in case of time dependent step size, for instance if there is a cycle with a long and slow storage phase and a short and fast regeneration phase. The binary option **skip_failed** may be activated in case the simulation is not successful for a few sets of inlet data. If activated, the inlet data of unsuccessful simulation time steps is skipped. The simulation continues with the next time step. Although the result is erroneous at this particular time step, the simulation may continue instead of abortion due to an error.

In case of the use of the washcoat effectiveness factor model, it may be required to change the effectiveness factor species according to the current inlet data. For this purpose the settings of the **<SURFACE-MODEL>** section (see chapter 3.8) may be altered by the transient inlet file. This can be done by including a **<SURFACE-MODEL>** definition after the data of the according time step. Most conveniently this can be done by saving the different **<SURFACE-MODEL>** sections in a separate files and including them in the input file, e.ġ.

```
<INLETINFO>
  u = 0.8       # constant velocity
  <MOLEFRAC list>
     CO  NO  O2  N2
  </MOLEFRAC>
</INLETINFO>

<INLETDATA>
#time   CO    NO    O2     N2
    0   0.01  0     0.09   0.9
 {include surface-model1.inp}  # CO is limiting species
   60   0.09  0.01  0      0.9
 {include surface-model2.inp}  # NO is limiting species
</INLETDATA>
```

18.3 Output

*reservoir***.plt*

At each time defined in the **<OUTPUT>** section a file ***reservoir***.plt*** is written. The file is in TECPLOT format and can be used to draw one-dimesional contour plots of the storage concentrations. It can also be imported into other spreadsheet programs like Microsoft Excel.

The file header contains the time when the file was written. The user may change this time when this file shall be used as a restart file of a new simulation.

In contrast to ***outs***.plt*** files, the storage concetrations are now written with respect to $F_{\mathrm{cat/geo}}$. Since the capacity may change along the channel, the written coverages are already multiplied with $F_{\mathrm{cat/geo}}$ of the storage component.

reservoir.plt

This file contains the data of all ***reservoir***.plt*** files. The file is in TECPLOT format and can be used to draw two-dimesional contour plots of the storage concentrations depending on time and position.

Note: In case of a restart, the previosly written data may be lost. Therefore it is advisible to copy ***reservoir.plt*** to a different file and concatenate the data manually after successful simulation.

restart.plt

The file ***restart.plt*** contains the same data as a ***reservoir***.plt*** file. If the **write_restart** option is activated, the file will be written in each time step. Therefore it always contains the state of the simulation at the latest time step.

global.plt

The file ***global.plt*** summarizes the global inlet and global outlet composition at each time step (not only the times listed). The file is in TECPLOT format and can be used to draw line graphs of gas-phase temperature and mass fractions. However, this format also allows for simple import into other spreadsheet programs (e. g. Microsoft Excel).

The variables in this file are: time, inlet temperature, inlet mass fractions, outlet temperature, outlet mass fractions, and an error flag. You can use this file for simple calculation of conversions and selectivities.

It is recommended to pay attention the the error flag. If the flag is not zero, an error in one or more channel simulations at this time step has occurred. The outlet composition at this time step is most likely wrong. If this only happens sparsely, you may delete the wrong line. However, if the error flag keeps to be set, the simulation was not successful.

A new file ***global.plt*** is written on each execution of DETCHEM$^{\text{RESERVOIR}}$. If you want to restart an earlier calculation, save ***global.plt*** with a different name before calling the executable again.

Channel output

For DETCHEM$^{\text{RESERVOIR}}$ it is not required to do post processing in order to generate channel output. All channel output files are written automatically for each time step listed in the **<OUTPUT>** section of the input file.

18.4 Examples

There are several examples in the distribution. Each directory contains several input files (****.inp***), species and mechanism information as well as the databases ***thermdata*** and ***moldata***. The ***.plt*** files are generated on execution.

The script file ***go*** may be used for convenience to call the executable and run the program. It requires a system variable `DETCHEM_DIR` that contains the path to the DETCHEM root directory. Depending on your system you can set this variable by either of these commands

```
export DETCHEM_DIR=~/myDirectory/DETCHEM
```

or

```
setenv DETCHEM_DIR ~/myDirectory/DETCHEM
```

18.5 Running the tool

There are two executables for DETCHEM$^{\text{RESERVOIR}}$ in the ***($DETCHEM_DIR)/bin*** directory. They are named ***reservoir_channel*** and ***reservoir_plug*** indicating the channel model used for the single channel simulation, i. e. DETCHEM$^{\text{CHANNEL}}$ or DETCHEM$^{\text{PLUG}}$, respectively.

There are several ways to call the executable:

- Using the ***go*** script from the example directory. The user needs to define the system variable `DETCHEM_DIR`.
- Add the ***($DETCHEM_DIR)/bin*** directory to your system path and call ***reservoir_channel*** or***reservoir_plug*** .
- Create a symbolic link to the executable ***($DETCHEM_DIR)/bin/reservoir_channel*** and call ***./reservoir_channel***.

The latter option does not require setting of system variables and is therefore less system specific.

Chapter 19

DETCHEM^{SOC}

19.1 Introduction

DETCHEM^{SOC} is a fully transient simulation tool designed to model solid oxide cells (SOCs). It is capable of modelling both "fuel cell" and "electrolysis" modes of operation and can generate both steady-state current-voltage (i-V) plots and frequency domain AC electro-impedance spectra (EIS) alongside concentration, surface coverage, current density and temperature profiles throughout the cell.

When combined with DETCHEM^{MONOLITH}, the tool can also model commercial-scale cell stacks with minimal loss of information in the μm-scale physics governing the individual repeating units that make up the stack. The salient feature of this hierarchical approach is the decoupling of the much slower solid phase stack temperature from other cell level dependent variables in the solution vector. Together with a cluster algorithm as previously described in Chapter 15, the resultant iterative problem becomes tractable and computationally efficient. The hierarchical approach is illustrated in Fig. 19.1. For a more detailed discussion, the user is kindly requested to refer to [1].

19.2 Model Framework

The complete set of equations used to develop the hierarchical mathematical framework is summarized below.

The following cell geometries are supported by the model:

1. half cells; corresponding to a setup focusing on a single working electrode since the opposite electrode has been replaced by an ideal counter electrode
2. symmetrical button cells (cf. Fig. 19.1, left)
3. (full) button cells
4. (full) planar single cells w/o a peripheral ceramic cell housing
5. (full) single planar repeating units of a stack including metallic interconnects (cf. Fig. 19.1, center)
6. stacks (cf. Fig. 19.1, right)

The model can be applied to state-of-the-art cell configurations used in academia and industry, for example

- Anode-supported configurations (ASCs), which e.g. comprise of a Ni mesh for contacting Ni-based anodes with the interconnectors, bi-layered electrode materials with a more porous "diffusion" layer near the current collector interface and a less porous "functional" layer near the electrolyte interface, and additional "barrier" layers between the electrodes and electrolyte (see Fig. 19.1)
- Metal-supported cell configurations (MSCs), which contain a thick porous metallic layer attached to the functional electrode layer, and which are attractive in the field of mobile/portable applications
- Electrolyte-supported cell configurations (ESCs), where the central electrolyte is the thickest constituent of the membran-electrode assembly (MEA), and which are frequently used for electrolysis applications.

Prior to using DETCHEM^{SOC}, the following characterisitics with respect to the model's sub-units should be respected by the user:

19.2.1 Microstructural model

To describe the porous nature of the MEA, percolation theory is used to depcit the electrodes' properties statistically in terms of volume-averaged properties. Assuming spherical particles and isotropic properties, the model is able to simulate either a conventional cermet microstructure (single phase or composite), as well as an infiltrated electrode morphology comprising of nanoparticles deposited onto a backbone scaffold. By default, the microstructural properties of the electrodes, $A^V_{gas/ede}$, $A^V_{el/io}$ and λ^V_{TPB}, are calculated as a function of the percolation probability, γ, and the coordination number, Z, based on the percolation model developed by Bertei et al. [2, 3] for cermet electrodes (Eqs. 19.38-19.40), or based on the analytical framework developed by Vijay et al. [4] for impregnated electrodes (Eqs. 19.41-19.45). However, these properties can be overridden and values for these parameters can be entered directly into the input file, which is a good option when these values have been evaluated using X-Ray Computer Tomography. The same is true for $\tau_{fac,m}$ and d_{pore}. If $\tau_{fac,m}$ is also provided for the electronic and ionic phases of a porous composite electrode, then σ^e_m is computed according to Eq. 19.46, and d_{pore} is evaluated as a function of ϵ according to Eq. 19.38 or Eq. 19.42.

19.2.2 Thermo-catalytic heterogeneous chemistry

Depeding on the fuel composition and the present electrode materials, thermo-catalytic heterogeneous chemistry can happen on active metallic centres in the fuel electrode in addition to the charge-transfer chemistry. The approach adopted here is to include detailed elementary kinetic mechanisms that need to be supplied via mechanism files ***mech.inp***, which are read in by DETCHEM^{SOC} upon execution. A 52-step elementary mechanism describing catalytic oxidation and steam/dry reforming of CH_4 over Ni [5], and a 12-step mechanism describing NH_3 decomposition/formation kinetics on Ba/Ru-YSZ [6] are currently in the distribution.

19.2.3 Electrochemical model

For depicting charge-transfer chemistry within the cells due to electric potential differences between different phases, the model uses the so-called distributed model for charge transport based on Ohm's law for both electronic and ionic conduction, spatially resolving the phases' electric potential and current density. While the transient term due to double-layer charging (Eq. 19.23) is responsible for the observed capacitive behavior at the small time-scales concerned during impedance spectroscopy, this term is set to zero in the charge balances (Eqs. 19.21-19.22) when simulating steady-state operation points or i-V curves, since it is usually negligibly small. The Faradaic source term due to electrochemical reactions (Eq. 19.28), which enters the charge balance equations is evaluated from a global Butler-Volmer framework for the anodic or cathodic half-cell reactions. To obtain the Faradaic current density i^V_F in volumetric form, a multiplication with a specific geometrical factor needs to occur. The code selects this geometrical factor based on the electrode material applied (Eq. 19.29): In case of a pure, single-phase mixed ion-electron conducting electrode (MIEC), the electrode's specific surface area is $A^V_{gas/ede}$ is used by the code, while in case of a Ni-CGO composite electrode (which can be directly specified by the user in the simulation input file), the specific pore/CGO interfacial area $A^V_{gas/io}$ acts as multiplier. In all other cases, the volumetric triple phase boundary length λ^V_{TPB} enters the equation. In common Butler-Volmer form, the user can provide the parameters pre-exponential factor, activation energy, species pressure exponents and symmetry factors as appropriate to define the kinetic set of a half-cell reaction. On the fuel electrode side, the electrochemical model supports (i) the H_2/H_2O and (ii) CO/CO_2 half-cell reactions individually, as well as (iii) the mixed pathway, the latter being e.g. relevant in the context of co-electrolysis. In this case, the net faradaic current density is evaluated from the two individual pathways via weighting factors, relating the current density to the availability of the reactive species in the electrochmically active region close to the fuel electrode-electrolyte interface, Eq. 19.37. With minimal changes to the computational framework, it is also possible to run DETCHEM^{SOC} to simulate a proton-conducting ceramic cell (PCC). When this option is selected by the user in the input file, the central electrolyte acts as a protonic instead of an oxygen ion conductor, and H_2O is produced/consumed at the air instead of the fuel electrode side. Thus, all parameters with respect to the H_2/H_2O pathway read in from the input file when modelling a SOC are simply used for the H_2 oxidation/reduction reaction at the fuel electrode side in a PCC. Consequently, no CO/CO_2 or mixed pathways are supported when the PCC-option is enabled, though thermo-catalytic heterogeneous chemistry certainly can be active within the fuel electrode. Aside from electrically isolating behavior, DETCHEM^{SOC} also supports to model an electronic leakage current across the central electrolyte, thereby modifying the electric potential and current density profiles. This behavior can simply be triggered by the user by defining the electrolyte to have an electronic conductivity in the corresponding **<Electro_Props>** tag.

19.2.4 Thermal model

To study heat development and evaluate spatially-resolved temperature profiles, non-isothermal simulations can be run. In this case, the thermal properties of the solid phase of the single planar RU that need to be specified in the input file are NOT factoring in the porosity, However, if the electrode is a composite, they need to account for the mean of materials 1 and 2. For planar stack simulations, the effective thermal properties directly enter the input file, so they need to be calculated prior to the simulation. The calculation of these parameters can be quite involved, and the user can either adopt an approach we have previously described in [1] or any other that is appropriate.

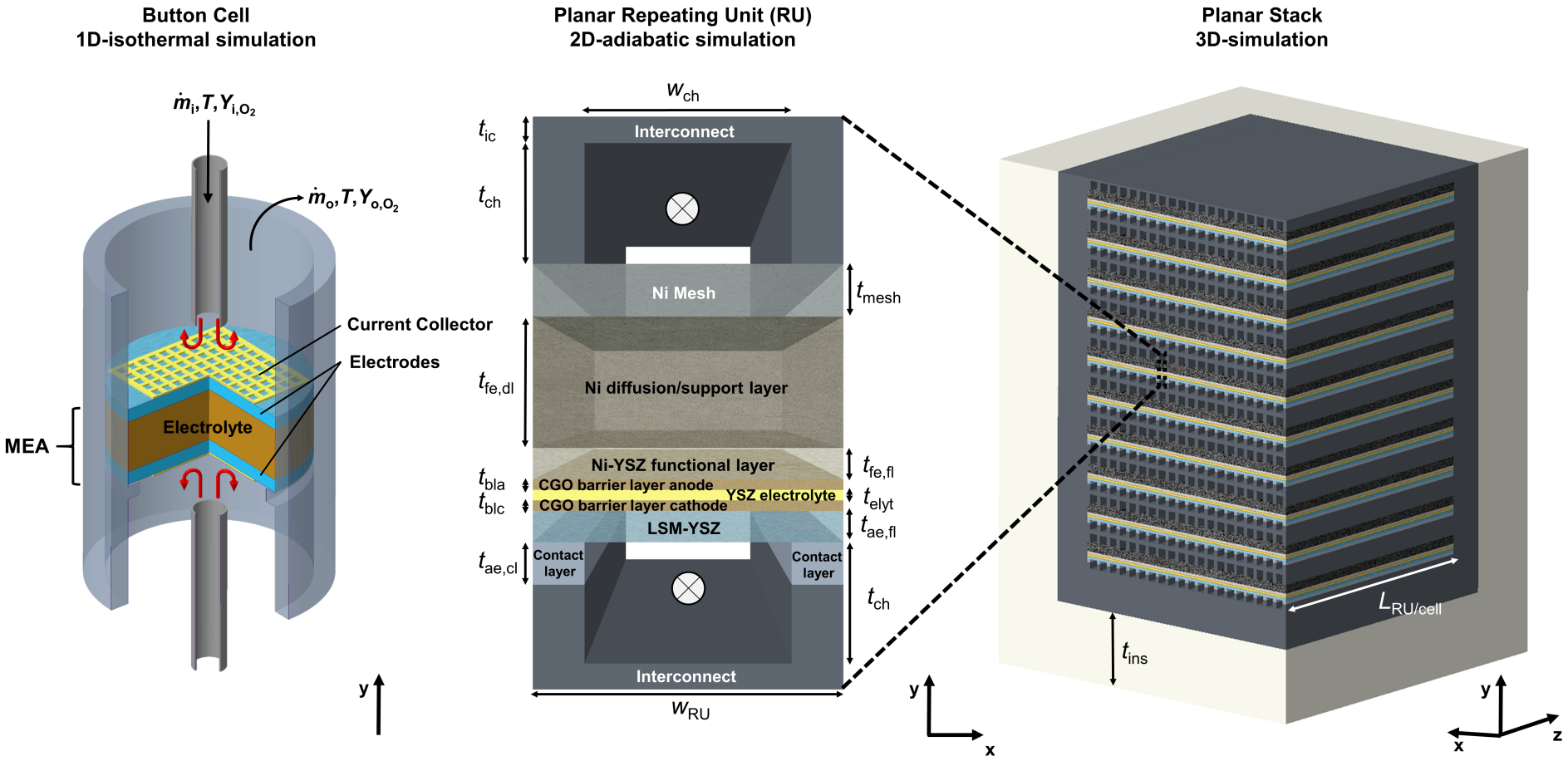


Figure 19.1: Schematic representation of the hierarchical approach utilized to model commercial scale stacks with detailed resolution of chemical kinetics and micro-scale transport phenomena. The crosses indicate flow into the page and the abbreviated material names are the default materials simulated by the model.

Summary of the model equations

1. Mass and momentum conservation:

- Bulk gas phase:
  - Button cell:

$$\begin{aligned}\rho_g V\frac{\partial\left(Y_k\right)}{\partial t} =&\dot{m}_i\left(Y_{i,k}-Y_k\right)+\\ &\dot{m}_{ede/elyt,k}-Y_k\sum_{k=1}^{K_g}\dot{m}_{ede/elyt,k},\quad k=1,\ldots,K_g\end{aligned} \tag{19.1}$$

$$\frac{\partial\rho_g}{\partial t}=-\rho_g\bar{W}\sum_{k=1}^{K_g}\frac{1}{W_k}\frac{\partial Y_k}{\partial t} \tag{19.2}$$

  - Planar RU:

$$\begin{aligned}\rho_g\frac{\partial Y_k}{\partial t} =&-\rho_g u\frac{\partial Y_k}{\partial z}\\ &+\frac{1}{t_{\text{ch}}}J_{k,ede}\Big|_{gas/CC}W_k-\frac{Y_k}{t_{ch}}\sum_{k=1}^{K_g}J_{k,ede}\Big|_{gas/CC}W_k\quad k=1,\ldots,K_g\end{aligned} \tag{19.3}$$

$$\rho_g\frac{\partial u}{\partial t}=-\rho_g u\frac{\partial u}{\partial z}-u\left(\sum_{k=1}^{K_g}\frac{1}{t_{ch}}J_{k,ede}\Big|_{gas/CC}W_k\right) \tag{19.4}$$

$$\rho_g=\frac{p}{RT}\frac{1}{\sum_{k=1}^{K_g}Y_k/W_k} \tag{19.5}$$

$$J_{k,ede}\big|_{gas/CC}=k_k\left(C_{k,ch}-C_{k,ede}\big|_{gas/CC}\right) \tag{19.6}$$

$$Sh=\frac{k_k d_h}{D_{k,mix}}=Sh_\infty+8.9336\left(\frac{1000}{Gz}\right)^{-0.5386}\exp\left(-\frac{6.7275}{Gz}\right) \tag{19.7}$$

$$d_h=\frac{2w_{ch}t_{ch}}{(w_{ch}+t_{ch})} \tag{19.8}$$

$$Gz=\frac{d_h}{z}ReSc \tag{19.9}$$

$$\begin{aligned}Sh_\infty =&7.541\left(1-2.6\frac{t_{ch}}{w_{ch}}+4.97\left(\frac{t_{ch}}{w_{ch}}\right)^2\right.\\ &\left.-5.11\left(\frac{t_{ch}}{w_{ch}}\right)^3+2.702\left(\frac{t_{ch}}{w_{ch}}\right)^4-0.548\left(\frac{t_{ch}}{w_{ch}}\right)^5\right)\end{aligned} \tag{19.10}$$

- Porous electrodes:

$$\frac{\partial\left(\epsilon\rho_k\right)}{\partial t}=-\frac{\partial\left(J_kW_k\right)}{\partial y}+\frac{\nu_k i_F W_k}{n_{e,r}F}+\hat{s}_k W_k A^v_{gas/ede},\quad k=1,\ldots,K_g \tag{19.11}$$

$$\frac{\partial\left(\epsilon\rho_g\right)}{\partial t}=-\sum_{k=1}^{K_g}\frac{\partial\left(J_kW_k\right)}{\partial y}+\sum_{k=1}^{K_g}\hat{s}_k W_k A^V_{gas/ede}+\sum_{k=1}^{K_g}\frac{\nu_k i_F W_k}{n_{e,r}F} \tag{19.12}$$

– Dusty Gas Model:

$$J_k = -\left[\sum_{i=1}^{K_a} D_{kl}^{DGM} \nabla [X_l] + \left(\sum_{l=1}^{K_s} \frac{D_{kl}^{DGM} [X_l]}{D_{l,Kn}^e}\right) \frac{B_g}{\mu} \nabla p\right]. \tag{19.13}$$

$$B_g = \frac{\epsilon^3 d_p^2}{72 \tau_{fac,pore} (1-\epsilon)^2} \tag{19.14}$$

$$D_{kl}^{DGM} = \left[\frac{1}{D_{k,Kn}^e} + \sum_{j \neq k} \frac{X_j}{D_{kj}^e}\right] \delta_{kl} + (\delta_{kl} - 1) \frac{X_k}{D_{kl}^e} \tag{19.15}$$

$$D_{k,Kn}^e = \frac{\epsilon}{\tau_{fac,pore}} \frac{d_{pore}}{3} \sqrt{\frac{8RT}{\pi W_k}} \tag{19.16}$$

$$D_{kl}^e = \frac{\epsilon}{\tau_{fac,pore}} D_{kl} \tag{19.17}$$

$$\tau_{fac,pore} = \left(1.23 \frac{(1-\epsilon)^{4/3}}{\epsilon}\right)^2 \tag{19.18}$$

– At the electrode-electrolyte interface:

$$J_k = 0 \tag{19.19}$$

2. Charge conservation:

- Reversible voltage:

$$E_{rev} = \frac{\Delta G_r^0}{n_{e,r} F} + \frac{RT}{n_{e,r} F} \ln\left(\frac{\prod_{k \in R_s} p_k^{\nu_k''}}{\prod_{k \in R_f} p_k^{\nu_k'}}\right) \tag{19.20}$$

- Distributed charge transfer model:

$$i_{el} = \frac{\partial}{\partial y}\left(\sigma_{el}^e \frac{\partial \phi_{el}}{\partial y}\right) = i_F^V + i_{DL}^V \tag{19.21}$$

$$i_{io} = \frac{\partial}{\partial y}\left(\sigma_{io}^e \frac{\partial \phi_{io}}{\partial y}\right) = -\left(i_F^V + i_{DL}^V\right) \tag{19.22}$$

$$i_{DL}^V = \frac{\partial}{\partial t}\left(C_{DL}^V \Delta\phi\right) \cdot \begin{cases} A_{gas/io}^V, & \text{if Ni-CGO composite electrode} \\ A_{gas/ede}^V, & \text{if pure MIEC electrode} \\ A_{el/io}^V, & \text{else} \end{cases} \tag{19.23}$$

$$C_{DL}^{el/io} = A_{DL} T_{ede} - B_{DL} \tag{19.24}$$

$$i = i_{el} + i_{io} \tag{19.25}$$

– At electrode-current collector interface:

$$\frac{\partial \phi_{io}}{\partial y} = 0 \tag{19.26}$$

– At electrode-electrolyte or electrode-barrier layer interface:

$$\frac{\partial \phi_{el}}{\partial y} = 0 \tag{19.27}$$

- Faradaic current density:

$$i_F = i_0 \left[\exp\left(\frac{n_{e,r} \beta_a F \eta_{act,fe/ae}}{RT} \right) - \exp\left(\frac{-n_e \beta_c F \eta_{act,fe/ae}}{RT} \right) \right] \tag{19.28}$$

$$i_F^V = i_F \cdot \begin{cases} A_{gas/io}^V, \text{ if Ni-CGO composite electrode} \\ A_{gas/ede}^V, \text{ if pure MIEC electrode} \\ \lambda_{TPB}^V, \text{ else} \end{cases} \tag{19.29}$$

  - Electrochemical oxidation of H_2 or reduction of H_2O:

$$\begin{aligned} i_F = i_{H_2/H_2O} = i_{0,H_2/H_2O} \Bigg[& \exp\left(\frac{\beta_{a,H_2} F \eta_{act,H_2/H_2O,fe}}{RT} \right) \\ & - \exp\left(- \frac{\beta_{c,H_2} F \eta_{act,H_2/H_2O,fe}}{RT} \right) \Bigg] \end{aligned} \tag{19.30}$$

$$i_{0,H_2/H_2O} = A_{H_2/H_2O} \exp\left(- \frac{E_{H_2/H_2O}}{RT} \right) f\left(p_{H_2}, p_{H_2O}\right) \tag{19.31}$$

  - Electrochemical oxidation of CO or reduction of CO_2:

$$\begin{aligned} i_F = i_{CO/CO_2} = i_{0,CO/CO_2} \Bigg[& \exp\left(\frac{\beta_{a,CO} F \eta_{act,CO/CO_2,fe}}{RT} \right) \\ & - \exp\left(- \frac{\beta_{c,CO} F \eta_{act,CO/CO_2,fe}}{RT} \right) \Bigg] \end{aligned} \tag{19.32}$$

$$i_{0,CO/CO_2} = A_{CO/CO_2} \exp\left(- \frac{E_{CO/CO_2}}{RT} \right) f\left(p_{CO}, p_{CO_2}\right) \tag{19.33}$$

  - Electrochemical consumption or production of O_2:

$$i_F = i_{O_2} = i_{0,0_2} \left[\exp\left(\frac{\beta_{a,O_2} F \eta_{act,ae}}{RT} \right) - \exp\left(\frac{-\beta_{c,O_2} F \eta_{act,ae}}{RT} \right) \right] \tag{19.34}$$

$$i_{0,O_2} = A_{O_2} \exp\left(- \frac{E_{O_2}}{RT} \right) f\left(p_{\mathrm{O}_2}\right) \tag{19.35}$$

  - Net faradaic current density (only for mixed pathway):

$$i_F = i_{H_2/H_2O,net} + i_{CO/CO_2,net} = w_{fac} i_{H_2/H_2O} + \left(1 - w_{fac}\right) i_{CO/CO_2} \tag{19.36}$$

$$w_{fac} = \left. \frac{X_{H_2/H_2O,fe/elyt}}{\left(X_{H_2/H_2O,fe/elyt} + X_{CO/CO_2,fe/elyt}\right)} \right|_{TPB} \tag{19.37}$$

3. Microstructural model:

- Cermet electrode:

$$d_{pore} = \frac{2}{3} * \frac{\varepsilon}{1 - \varepsilon_{bs}} \frac{d_m d_n}{v_{f,m,bs} d_n + v_{f,n,bs} d_m} \tag{19.38}$$

$$\gamma_m = 1 - \left(\frac{4.236 - Z_{m,m}}{2.472} \right)^{3.7} \tag{19.39}$$

$$\lambda_{TPB}^V = 6 \left(1 - \varepsilon_{bs}\right) \min\left(d_m, d_n\right) \sin \frac{\pi}{12} n_m^V Z_{m,n} \gamma_m \gamma_n \tag{19.40}$$

- Infiltrated electrode:

$$\varepsilon = \varepsilon_B - \frac{4\pi n_{np}^V r_{np}^3}{3} \tag{19.41}$$

$$d_{pore} = \frac{2}{3} * \frac{\varepsilon}{1-\varepsilon} d_B \tag{19.42}$$

$$\gamma_{np} = \left(1 - \left(\frac{0.75 - S_{np}}{0.25}\right)^{0.52}\right)^{0.097} \tag{19.43}$$

$$\gamma_{pore} = \left(1 - \left(\frac{0.2 - \varepsilon}{0.15}\right)^{2.68}\right)^{0.25} \tag{19.44}$$

$$\lambda_{TPB}^V = 2\pi r_{np} n_{np}^V Z_{np,B} \gamma_{np} \gamma_{pore} \tag{19.45}$$

- Effective electronic/ionic conductivity:

$$\sigma_m^e = \left\{ \begin{array}{c} \sigma_{m,b} \frac{(1-\epsilon) v_{f,m}}{\tau_{fac,m}}, \text{ if } \tau_{fac,m} \text{ is defined in .inp} \\ \sigma_{m,b} \left[(1-\epsilon) v_{f,m} \gamma_m\right]^\gamma, \text{ else} \end{array} \right\}, m \in el, io \tag{19.46}$$

4. Energy conservation:

- Bulk gas phase:

$$h_{conv} = \frac{Nu \lambda_g}{d_h} \tag{19.47}$$

$$Nu = 3.095 + 8.933 \left(\frac{1000}{Gz}\right)^{-0.5386} \exp\left(-\frac{6.7275}{Gz}\right) \tag{19.48}$$

$$Gz = \frac{d_h}{z} RePr \tag{19.49}$$

  - Single planar RU:

$$\begin{aligned} \rho_g C_{p,g} \frac{\partial T_s}{\partial t} = & -u \rho_g C_{p,g} \frac{\partial \left(T_g\right)}{\partial z} + h_{conv} \frac{1}{t_{ch}} \left(T_{ede} - T_g\right) \\ & + h_{conv} \frac{2t_{ch} + w_{ch}}{w_{ch} t_{ch}} \left(T_{ic} - T_g\right) \end{aligned} \tag{19.50}$$

  - Stack:

$$\rho_g C_{p,g} \frac{\partial \left(T_g\right)}{\partial t} = -u \rho_g C_{p,g} \frac{\partial \left(T_g\right)}{\partial z} + \frac{4}{d_h} h_{conv} \left(T_{stack,s} - T_g\right) \tag{19.51}$$

- Solid phase of Single Planar RU:

  - Porous Electrodes:

$$\begin{aligned} \rho_{ede} C_{p,ede}^e \frac{\partial T_{ede}}{\partial t} = & \frac{\partial}{\partial y} \left(\lambda_{ede}^e \frac{\partial T_{ede}}{\partial y}\right) + \frac{\partial}{\partial z} \left(\lambda_{ede}^e \frac{\partial T_{ede}}{\partial z}\right) \\ & + \frac{i^2}{\sigma_{ede}^e} + \dot{Q}_{act} + \dot{Q}_{echem} \\ & + \left\{ \begin{array}{cc} \dot{Q}_{hchem}, & \text{within the fuel electrode} \\ 0, & \text{within the air electrode} \end{array} \right. \end{aligned} \tag{19.52}$$

$$\dot{Q}_{\text{echem}} = \begin{cases} -\frac{i_F^V}{n_{e,r}F} T\Delta S_r, \text{ within fuel electrode} \\ 0, \text{ within the air electrode} \end{cases} \tag{19.53}$$

$$\dot{Q}_{act} = i_F^V \eta_{act,ede} \tag{19.54}$$

$$\dot{Q}_{hchem} = -\sum_{k=1}^{K_g} H_k \dot{s}_k A_{gas/ede}^V \tag{19.55}$$

* Additional heat source terms at the electrode-electrolyte (elyt) or electrode-barrier layer (bl) interface:

$$\frac{\dot{Q}_{contact}}{\partial y} = \frac{\frac{i^2 R_{contact,ede\rightarrow elyt(bl)}}{A_{ede/elyt(bl)}}}{\partial y} \tag{19.56}$$

* Additional heat source terms at the electrode-bulk gas interface:

$$\frac{\dot{Q}_{conv}}{\partial y}\left(\frac{w_{ch}}{w_{RU}}\right) = \frac{h_{conv}\left(T_g - T_{ede}\right)}{\partial y}\left(\frac{w_{ch}}{w_{RU}}\right) \tag{19.57}$$

* Radiative heat exchange:
  · Interconnects:

$$-\frac{\dot{Q}_{rad}}{\partial y}\left(\frac{1}{w_{RU}}\right) = -\frac{\frac{\sigma_{SB}\left(T_{ede}^4 - T_{ic}^4\right)}{R_{rad,ede\rightarrow ic}}}{\partial y}\left(\frac{1}{w_{RU}}\right) \tag{19.58}$$

$$R_{rad,ede\rightarrow ic} = \frac{1-\epsilon_{rad,ede}}{\epsilon_{rad,ede} w_{ch}} + \frac{1}{F_{ede\rightarrow ic} w_{ch}} + \frac{1-\epsilon_{rad,ic}}{\epsilon_{rad,ic}\left(w_{\text{ch}} + 2t_{ch}\right)} \tag{19.59}$$

$$F_{ede\rightarrow ic} = F_{ede\rightarrow ic,plate} + 2F_{ede\rightarrow ic,rib} \tag{19.60}$$

  · Cell housing:

$$-\frac{\dot{Q}_{rad}}{\partial y} = -\frac{\frac{\sigma_{SB}\left(T_{ede}^4 - T_{hg}^4\right)}{\left(\frac{1}{\epsilon_{rad,ede}} + \frac{1}{\epsilon_{rad,ic}} - 1\right)}}{\partial y} \tag{19.61}$$

* Additional heat source term at the electrode-interconnect rib interface:

$$\frac{\dot{Q}_{contact}}{\partial y} = \frac{i^2 R_{contact,ede\rightarrow ic,rib}}{\partial y} \tag{19.62}$$

– Dense electrolyte (elyt) and barrier layers (bl):

$$\rho_{elyt} C_{p,elyt(bl)} \frac{\partial T_{elyt(bl)}}{\partial t} = \frac{\partial}{\partial y}\left(\lambda_{elyt(bl)} \frac{\partial T_{elyt(bl)}}{\partial y}\right) + \frac{\partial}{\partial z}\left(\lambda_{elyt(bl)} \frac{\partial T_{elyt(bl)}}{\partial z}\right) + \frac{i^2}{\lambda_{elyt(bl)}} \tag{19.63}$$

– At the electrode-electrolyte interface:

$$\lambda_{ede}^e \frac{\partial T_{ede}}{\partial y} = \lambda_{elyt,b} \frac{\partial T_{elyt}}{\partial y} \tag{19.64}$$

– Interconnects (ic) or cell housing (hg):

$$\rho_{ic/hg} C_{p,ic/hg} \frac{\partial T_{ic/hg}}{\partial t} = \frac{\partial}{\partial y}\left(\lambda_{ic/hg} \frac{\partial T_{ic/hg}}{\partial y}\right) + \frac{\partial}{\partial z}\left(\lambda_{ic/hg} \frac{\partial T_{ic/hg}}{\partial z}\right) \tag{19.65}$$

* Additional heat source terms at the interconnect-bulk gas interface:

$$\dot{Q}_{conv}\left(\frac{2t_{ch}+w_{ch}}{vol_{ic}}\right)=h_{conv}\left(T_g-T_{ic}\right)\left(\frac{2t_{ch}+w_{ch}}{vol_{ic}}\right) \tag{19.66}$$

$$\frac{\dot{Q}_{rad}}{vol_{ic}}=\frac{\frac{\sigma_{SB}\left(T_{ede}^4-T_{ic}^4\right)}{R_{rad,ede\rightarrow ic}}}{vol_{ic}} \tag{19.67}$$

$$vol_{ic}=\left(t_{ic}+t_{ch}\right)w_{RU}-w_{ch}t_{ch} \tag{19.68}$$

* Additional heat source terms at the cell housing-bulk gas interface:

$$\frac{\dot{Q}_{rad}}{t_{hg}}=\frac{\frac{\sigma_{SB}\left(T_{ede}^4-T_{hg}^4\right)}{\left(\frac{1}{\epsilon_{rad,ede}}+\frac{1}{\epsilon_{rad,ic}}-1\right)}}{t_{hg}} \tag{19.69}$$

$$\frac{\dot{Q}_{loss}}{t_{hg}}=\frac{\sigma_{SB}\epsilon_{rad,hg}\left(T_{surr}^4-T_{hg}^4\right)+h_{conv,hg}\left(T_{surr}-T_{hg}\right)}{t_{hg}} \tag{19.70}$$

– At the electrode-interconnect rib interface:

$$\lambda_{ede}^{e}w_{RU}\frac{\partial T_{ede}}{\partial y}=\left(\frac{1}{\lambda_{rib,b}w_{rib}}+\frac{1}{\lambda_{ic,b}w_{RU}}\right)^{-1}\frac{\partial T_{ic}}{\partial y} \tag{19.71}$$

- Solid phase of Planar Stack:

$$\begin{aligned}\rho_{stack,s}C_{p,stack}^{e}\frac{\partial T_{stack,s}}{\partial t}&=\nabla\cdot\left(\lambda_{stack}^{e}\nabla T_{stack,s}\right)\\&+\sum_{ch\in air,fuel}h_{conv}\left(T_g-T_{stack,s}\right)\left(\frac{2\left(w_{ch}+t_{ch}\right)}{w_{RU}t_{RU}}\right)\\&+\left(\dot{Q}_i+\dot{Q}_{hchem}\right)\frac{t_{fe}}{t_{RU}}\end{aligned} \tag{19.72}$$

$$t_{RU}=2\left(t_{ic}+t_{ch}\right)+t_{ae}+t_{elyt}+t_{bla}+t_{blc}+t_{fe}+t_{mesh} \tag{19.73}$$

$$\dot{Q}_i=-i_F^V\left(E_{cell}+\frac{\Delta H_r}{n_{e,r}F}\right) \tag{19.74}$$

19.3 User Input

19.3.1 DETCHEM^{SOC}

DETCHEM^{SOC}requires a ***.inp*** file as input. The name of the input file is user-defined. This enables the simultaneous simulation of multiple input files using either a job scheduler or using virtual screens. The contents of the input files need to vary depending on the type of simulation. Conveniently, the type of a simulation can be defined according to the following factors: (i) the applied cell geometry (button cell/single planar repeating unit (RU)/single planar cell), (ii) the electrochemical state or characterization technique mimicked (steady-state operation point/i-V curve/transient EIS) and (iii) the thermal boundary conditions (isothermal/adiabatic/heat loss). In principle, the code supports to vary these simulation parameter definitions independently from each other, aside from the definition of (i) a non-isothermal button cell simulation (choose isothermal conditions instead), (ii) a sequence of simulations at different cell voltages to model an entire i-V-curve under non-isothermal conditions (perform mutliple individual simulations or choose isothermal conditions instead), (iii) a single planar RU simulation with heat loss enabled (choose isothermal or adiabatic conditions instead). Such definitions will immediately cause an error message to be displayed. To illustrate all the features, several often-used simulation types are discussed in detail below based on example input files.

19.3.2 Isothermal steady-state i-V button cell simulation

An example of the input file used to run an isothermal steady-state i-V button cell simulation is shown below. This input file is used to introduce and explain all features common to all simulation types.

```
# Button cell isothermal - steady-state i-V simulation.
{verbose y}
{include species.inp}      # Include the list of all species involved.
{include mech.inp}         # Include a thermo-catalytic heterogeneous elementary mechanism
                           # in mech.inp, choose between
                           # (i) CH4/CO/CO2 chemistry on Ni and (ii) NH3 chemistry on Ru.
<SURFACE-MODEL>            # DETCHEM inputs to solve heterogeneous thermochemistry.
 <CHEMSURF>
    hini = 1.e-10 [s]      # Initial time step for integration of thermo-catalytic elementary mechanism.
    time = 10 [s]          # End time for integration of the surface mechansim.
 </CHEMSURF>
</SURFACE-MODEL>

<SOFC>                           # SOC inputs begin
  op_pressure = 101325.0 [Pa]    # Initial guess of pressure in electrodes
  reference_anode = n            # The fuel electrode is a working electrode,
                                 # i.e. it is not a half cell comprising of the cathode and electrolyte only.
  reference_cathode = n          # The air electrode is a working electrode,
                                 # i.e. it is not a half cell comprising of  the anode and electrolyte only.
  <GEOMETRY model=button>  # Defines the cell geometry. Choose between 'button', 'planar' and 'planar_stack'.
  electrochemistry = y           # Electrochemical reactions are activated.
  surface_chemistry = y          # Thermo-catalytic heterogeneous chemistry in the fuel electrode is active.
    <SECTION>
      name = air                 # Air channel section.
      height = 1.0e-03 [m]
      ny = 1                     # No. of radial nodes along air channel height (or thickness).
                                 # It must always be 1 for air and fuel channels.
      <INLET>
        u = 1 [m/s]              # Inlet gas velocity.
        temperature = 1073.15 [K] # Inlet gas temperature.
        pressure = 101325.0 [Pa]  # Inlet gas pressure.
        <SPECIES>
          <MOLEFRAC>             # Inlet gas composition (can either be 'MOLEFRAC' or 'MASSFRAC')
            O2 0.211
            N2 *
          </MOLEFRAC>
        </SPECIES>
      </INLET>
    </SECTION>
      <SECTION>
        name = cathode             # Air electrode section. Even in electrolysis mode 'cathode' represents the air electrode.
        <Electro_Props>            # Define cathode electrical properties.
          # Cathode-electrolyte interfacial contact resistance = k*exp(-Ea/(R*T))*pO2^pO2m or set temperature-independent to R
            Rcon_ede_elyt = 0.2e-5 [Ohm m2]
          # k_con_ede_elyt = 0.015 [Ohm m2]
          # Ea_con_ede_elyt = 80e6 [J/mol]
          # A_fac_ede_elyt = 1.0   # Cathode-electrolyte interface area per unit of cathode area. Default is 1.
          <Functional_Layer>       # Define functional layer electrical properties.
               # Define Arrhenius-type expressions for electric (sigma_el) and ionic conductivities (sigma_io) of cathode materials.
               # Partial pressures are in [Pa]; if it is a composite, the more electronically conductive material must be material 1.
               <sigma_el_mat1> 1e5 [S/cm] 100e3 [J/mol] -0.25 </sigma_el_mat1>  # =1e5*exp(-10e3/(R*T))*pO2^(-0.25) [S/cm]
               <sigma_el_mat2> 1e3 [S/cm] 100e3 [J/mol] -0.25 </sigma_el_mat2>
               <sigma_io_mat1> 1e2 [S/cm] 100e3 [J/mol] -0.25 </sigma_io_mat1>
               <sigma_io_mat2> 1e5 [S/cm] 100e3 [J/mol] -0.25 </sigma_io_mat2>
          </Functional_Layer>
          <Diffusion_Layer>        # Define diffusion layer electrical properties.
          # Directly specify a selected cathode material. If one of the following options is used, the corresponding Arrhenius
          # parameters don't need to be provided by the user anymore.
          # is_LSC = n             # Set material 1 to be LSC.
          # is_LSCF = n            # Set material 1 to be LSCF.
          # is_GDC = n             # Set material 2 to be GDC (CGO).
          </Diffusion_Layer>
        </Electro_Props>
        # If conductivity values for the functional layer are not provided (neither user-specified Arrhenius parameters provided
        # nor material option used), the code uses a default configuration of LSM as material 1 and YSZ as material 2.
        # If conductivity values for the diffusion layer are not provided, then the code uses the values of the functional layer.
          <Microstructure>               # Define geometrical and microstrutural properties of the cathode section.
            <Functional_Layer>           # Define functional layer (near the electrolyte interface) microstructural properties.
           height = 10.0e-06 [m]         # Height (or thickness) of functional layer.
           ny = 10                       # No. of radial nodes along layer thickness.
           infiltrated = n               # The layer is a conventional cermet, i.e. the infiltrated microstructural model is not used.
           # To set the microstructural properties, only porosity, particle diameter and volume fraction of the functional layer
           # need to be specified mandatorily. The rest are either calculated by the code or have working default values.
           porosity = 0.3                # Porosity of functional layer, i.e. electrode layer near the electrode-electrolyte interface.
           # porosity_grad = 0           # Specify a porosity gradient across the functional layer. If positive, cathode is more porous
                                         # towards the electrolyte and less porous towards the air channel. Default is 0.
           # pore_dia = 0.2e-06 [m]      # Specify the mean pore diameter. Default is calculated from percolation theory.
           part_dia = 0.3e-06 [m]        # Specify the particle size of material 1.
           # part_dia_ratio = 1          # Specify the particle size ratio between materials 1 and 2. Default is 1.
           # A_2pb_mat1_pore = 1e5 [m-1] # Specify the pore-material 1 interfacial area per unit volume.
           # A_2pb_mat1_mat2 = 1e5 [m-1] # Specify the material 1-material 2 interfacial area per unit volume.
                                         # Default is calculated from Percolation theory.
```

```
      # specific_3pb = 1e12 [m-2]   # Specify the (pore-material 1-material 2) triple phase boundary length per unit volume.
                                    # Default is calculated from Percolation theory.
      volume_fraction = 0.5         # Specify the material 1 solid phase volume fraction.
      tort_mat1 = 3                 # Specify the tortuosity factor of the material 1 solid phase.
                                    # Default is calculated from percolation theory.
      tort_mat2 = 3                 # Specify the tortuosity factor of the material 2 solid phase.
                                    # Default is calculated from percolation theory.
      # tort_pores = 2              # Specify the tortuosity factor of the pore phase.
                                    # Default is calculated from an expression relating it to the porosity.
       </Functional_Layer>
         # The following are the properties of the dl, i.e. electrode layer near the gas channel.
       <Diffusion_Layer>
          height = 10.0e-06 [m]    # Height (or thickness) of diffusion layer.
          ny = 10                  # No. of radial nodes along layer thickness.
          # If dl_height is > 0, but the microstructural properties are not specified within this tag, then the values
          # will either be set equal to their corresponding values in the fl, or generated by the code via Percolation theory.
          porosity = 0.6           # Porosity of diffusion layer.
          part_dia = 0.3e-06 [m]   # Particle size of material 1 in diffusion layer.
          volume_fraction = 0.5    # Specify volume fraction of solid phase in diffusion layer that is material 1.
       </Diffusion_Layer>
     </Microstructure>
   <SPECIES>
     <MOLEFRAC>                      # Initial guesses for gas composition in the electrode
       O2  0.21
       N2  *
     </MOLEFRAC>
   </SPECIES>
 </SECTION>
 <SECTION
   name = electrolyte           # Electrolyte section. If applicable,
                                # this section also contains barrier layer coatings between electrodes and electrolyte.
   <Electro_Props>              # Define electrolyte materials electrical properties.
        <Electrolyte>
       # If conductivity is unspecified, the default electrolyte layer is 8YSZ.
       <sigma_io_mat1> 330 [S/cm] 100e3 [J/mol] </sigma_io_mat1>
       # Define an electronic leakage across the electrolyte by specifiying an electronic conductivity >0.
       # <sigma_el_mat1> 3.5 [S/cm] 100e3 [J/mol] </sigma_el_mat1>
       </Electrolyte>
       # If conductivities are unspecified, the default barrier layer coatings on both sides is CGO.
       <Barrier_Layer_Anode>    # Ionic conductivity of barrier layer coating between anode and electrolyte.
       <sigma_io_mat1> 1e5 [S/cm] 100e3 [J/mol] </sigma_io_mat1>
       # Electrolyte-anode barrier layer interfacial contact resistance = k*exp(-Ea/(R*T)) or set temperature-independent to R.
       Rcon_bl_elyt = 0.025e-4 [Ohm m2]
       # k_con_bl_elyt = 0.015 [Ohm m2]
       # Ea_con_bl_elyt = 80e6 [J/mol]
       </Barrier_Layer_Anode>
       <Barrier_Layer_Cathode>  # Ionic conductivity of barrier layer coating between cathode and electrolyte.
       <sigma_io_mat1> 1e5 [S/cm] 100e3 [J/mol] </sigma_io_mat1>
       </Barrier_Layer_Cathode>
   </Electro_Props>
   <Microstructure>             # Define geometrical properties of the electrolyte section.
       <Electrolyte>            # Define central electrolyte layer properties.
          height = 40.0e-06 [m]
          ny = 10
       </Electrolyte>
       <Barrier_Layer_Anode>    # Define dense barrier layer coating layer properties (anode side).
          height = 10.0e-06 [m]
          ny = 5
       </Barrier_Layer_Anode>
       <Barrier_Layer_Cathode>  # Define dense barrier layer coating layer properties (cathode side).
          height = 10.0e-06 [m]
          ny = 5
       </Barrier_Layer_Cathode>
   </Microstructure>
 </SECTION>
 <SECTION>
   name = anode               # Fuel electrode section. Even in electrolysis mode 'anode' represents the fuel electrode.
   # Please look at comments for the cathode above. Only things that are different are commented on in this section.
   <Electro_Props>            # Define anode electrical properties.
     # Anode-electrolyte interfacial contact resistance = k*exp(-Ea/(R*T))*pH2O^pH2Om or set temperature-independent to R
     Rcon_ede_elyt = 0.025e-4 [Ohm m2]
     # k_con_ede_elyt = 0.015 [Ohm m2]
     # Ea_con_ede_elyt = 80e6 [J/mol]
     # A_fac_ede_elyt = 1.0   # Anode-electrolyte interface area per unit of cathode area. Default is 1.
     <Functional_Layer>
          # Define Arrhenius-type expressions for electric (sigma_el) and ionic conductivities (sigma_io) of anode materials.
          # Partial pressures are in [Pa]; if it is a composite, the more electronically conductive material must be material 1.
          <sigma_el_mat1> 1e5 [S/cm] 100e3 [J/mol] -0.25 </sigma_el_mat1>  # =1e5*exp(-10e3/(R*T))*pO2^(-0.25) [S/cm]
          <sigma_el_mat2> 1e3 [S/cm] 100e3 [J/mol] -0.25 </sigma_el_mat2>
          <sigma_io_mat1> 1e2 [S/cm] 100e3 [J/mol] -0.25 </sigma_io_mat1>
          <sigma_io_mat2> 1e5 [S/cm] 100e3 [J/mol] -0.25 </sigma_io_mat2>
          # is_GDC = n
     </Functional_Layer>
     <Diffusion_Layer>
          # ...
     </Diffusion_Layer>
     # If conductivity values for the functional layer are not provided (neither user-specified Arrhenius parameters provided
     # nor material option used), the code uses a default configuration of Ni as material 1 and YSZ as material 2.
```

```
        # If conductivity values for the diffusion layer are not provided, then the code uses the values of the functional layer.
        <Mesh>                      # Define electrical properties of the mesh. Default is Ni.
              <sigma_el_mat1> 1e5 [S/cm] 100e3 [J/mol] </sigma_el_mat1>
        </Mesh>
      </Electro_Props>
      <Microstructure>              # Define geometrical and microstrutural properties of the anode section.
        <Mesh>                      # Contacting mesh layer. Definition is only supported at the anode side.
            height = 100.0e-06 [m]  # Mesh height (or thickness).
            ny = 10                 # No. of radial nodes along layer thickness.
            Bg = 1.5e-08 [m-2]      # Specify the permeability of the mesh.
            porosity = 0.7          # Specify the mesh porosity.
            tort_pores = 1.5        # Specify the mesh pore phase tortuosity.
        </Mesh>
        <Functional_Layer>          # Functional layer.
            height = 25.0e-06 [m]
            ny = 10
            infiltrated = n
            porosity = 0.25
            part_dia = 0.3e-06 [m]
            volume_fraction = 0.5
        </Functional_Layer>
        <Diffusion_Layer>           # Diffusion layer.
            height = 375.0e-06 [m]
            ny = 10
            infiltrated = n
            porosity = 0.6
            part_dia = 0.3e-06 [m]
            volume_fraction = 1
            tort_pores = 1.2
        </Diffusion_Layer>
      </Microstructure>
      <SPECIES>
        <MOLEFRAC>
          H2  0.5
          H2O  *
        </MOLEFRAC>
      </SPECIES>
    </SECTION>
    <SECTION>
      name = fuel                   # Fuel channel section.
      height = 1.0e-03 [m]
      ny = 1
      <INLET>
        u = 1 [m/s]
        temperature = 1073.15 [K]
        pressure = 101325.0 [Pa]
        <SPECIES>
          <MOLEFRAC>
            H2   0.1
            CH4  0.5
            H2O  *
          </MOLEFRAC>
        </SPECIES>
      </INLET>
    </SECTION>
</GEOMETRY>
<ELECTROCHEM>

  # general settings:
  protonic_cell = n           # No protonic ceramic cell (PCC) is modelled, the central electrolyte is an oxygen-ion conductor.
                              # Default is 'n'.
  electro_mechanism = h2      # The electrochemically active pathway is H2 only.
                              # The other options are 'co' or 'mixed' when both H2 and CO pathways are active.

  # set voltage range:
  # Ecell = 1.4 [V]           # Single cell potential.
  Ecell_upper_bound = 1.4 [V] # Upper bound for cell potential to be studied.
  Ecell_lower_bound = 0.5 [V] # Lower bound for cell potential to be studied.
  potential_step = 0.1 [V]    # Increment of cell potential.
  # Here, the code will run simulations for applied cell potentials in the range 1.4V to 0.5V in steps of 0.1V.

  # Alternatively, choose overpotential format:
  # Overpotential_format = y  # This is to plot tafel plots, i.e. log(i) on y-axis vs cell overpotential
  # (= Ecell-Ecell_eq) on x-axis.
  # Ecell_eq = 0.515 [V]      # Open-circuit/equilibrium potential of cell.
  # Ovp_upper_bound = 0.1 [V] # Upper bound of overpotential range to be studied.
  # Ovp_lower_bound = -0.1 [V]# Lower bound of overpotential range to be studied.

  <Charge_Transfer_Kinetics>
  # Exchange current density i = k*exp(-E/(R*T))*p(X)^m_X or directly specified as temperature-independent i.
  # If m_X set to 0, then the code uses exchange current density pressure dependencies derived
  # by Zhu et al. [7].
  # Partial pressures are in [Pa].
       <H2/H2O>
      beta_a = 0.5            # Anodic and cathodic charge transfer coefficient for electrochemical
      beta_c = 0.5            # H2 oxidation/H2O reduction in the fuel electrode.
      # i_tpb = 8.5e03 [A/cm] # Temperature-independent exchange current density.
      k_tpb = 357e-7 [A/cm]   # Pre-exponential factor for exchange current density.
```

```
        E = 108.4e03 [J/mol]    # Activation barrier for exchange current density.
        # mH2 = 0.25            # H2 pressure exponent.
        # mH2O = 0.25           # H2O pressure exponent.
         </H2/H2O>
         # <CO/CO2>
        # beta_a = 0.45         # Anodic and cathodic charge transfer coefficient for electrochemical
        # beta_c = 0.45         # CO oxidation/CO2 reduction in the fuel electrode.
        # i_tpb = 5e03 [A/cm]   # Temperature-independent exchange current density.
        # k_tpb = 148e-7 [A/cm] # Pre-exponential factor for exchange current density.
        # E = 131.38e03 [J/mol] # activation barrier for exchange current density.
        # mCO = 0.25            # CO pressure exponent.
        # mCO2 = 0.25           # CO2 pressure exponent.
         # </CO/CO2>
         <O2>
        beta_a = 0.5            # Anodic and cathodic charge transfer coefficient for O2 electrochemical
        beta_c = 0.5            # reduction/production in the air electrode.
        # i_tpb = 2.8e03 [A/cm] # Temperature-independent exchange current density.
        k_tpb = 10e-7 [A/cm]    # Pre-exponential factor for exchange current density.
        E = 122.5e03 [J/mol]    # Activation barrier for exchange current density.
        # mO2 = 0.25            # O2 pressure exponent.
         </O2>
    </Charge_Transfer_Kinetics>
  </ELECTROCHEM>
  <INIT>
    isothermal=yes              # The simulation is isothermal.
  </INIT>
  <SOLVER>
    t_Begin = 0 [s]             # Start time for the MEA integration.
    t_End = 10 [s]              # End time for the MEA integration.
    dt = 1.0d-12 [s]            # Initial time-step. The solver will then automatically adjust the following time-steps.
    <TOLERANCE>
      abs_Tol = 1.0d-08         # Absolute tolerance of each governing equation solution for each time step.
      rel_Tol = 1.0d-09         # Relative tolerance of each governing equation solution for each time step.
      res_Tol = 1.0d-06         # Average residual tolerance of entire solution set over all time steps.
    </TOLERANCE>
  </SOLVER>
  <OUTPUT>
    species = mole              # Outputs will be mole fractions. It can also be 'mass' for mass fraction outputs.
    file = 5                    # File no corresponding to this simulation.
    monitor = y                 # Show progress of simulation.
                                # It should always be turned on to track the solution process and detect possible errors.
  </OUTPUT>
</SOFC>
```

19.3.3 EIS simulation

The input file for an EIS simulation is almost identical to the steady-state i-V simulation apart from a few changes mainly under the **<ELECTROCHEM>** tag. Line-specific comments in italics are provided for these altered lines from an example input file. Only the relevant line changes with respect to section 19.3.2 are shown.

```
# Button cell isothermal - transient eis simulation.
# ...
<SOFC>
  # ...
  <SECTION>
    name=cathode            # (and/or anode)
  # ...
    <Electro_Props>
    # Double Layer capacitance C_dl = A_cdl*T - B_cdl. If values are not specified,
    # C_dl for both electrodes are taken to equal C_dl of YSZ. See [8] for default values of A_cdl and B_cdl
    # A_dl = 1e-2 [F/m2/K]
    # B_dl = 7 [F/m2]
  # ...
    </Electro_Props>
  </SECTION>
  # ...
  <ELECTROCHEM>
    electro_mechanism = h2  # Choose the electrochemically active pathway.
    Simulate_EIS = y        # Perform an eis simulation.
    symmetric_cell = y      # Implies that it is a symmetric half-cell simulation.
    Ebase = 1.0247 [V]      # The DC cell voltage at which the eis is to be performed. Typically, it is OCV.
    Estep = 0.02 [V]        # The AC voltage signal on top of the base DC voltage.
    tau = 1e-9 [s]          # The time constant of the exponential step input.
                            # This value is optimised for computational speed and accuracy.
    dt_trans = 1d-12 [s]    # Initial time step for eis simulation.
    <Charge_Transfer_Kinetics>
  # ...
    </Charge_Transfer_Kinetics>
  </ELECTROCHEM>
  # ...
</SOFC>
```

The model simulates EIS using an exponential step input and a subsequent Fast Fourier Transform as described by Bessler [9]. This method is preferred over the standard sinusoidal AC input owing to its significantly

greater computational speed with comparable accuracy. However, the accuracy of the technique is sensitive to the time constant of the exponential step input, "*tau*", and should not be less than 1e-7 [9].

19.3.4 Non-isothermal steady-state i-V planar cell/RU simulation

The input file to run an non-isothermal steady-state i-V simulation of a planar cell/RU builds on the input file for an isothermal steady-state i-V button cell simulation, but requires a few additional lines. Apart from the parameters owing to the change in geometry, thermal properties for the electrolyte and the two electrodes under the **<Thermal_Props>** tag as well as radiative properties under the **<RADIATION>** tag need to be provided. Line-specific comments in italics are provided for these additional lines using an example input file. The lines identical to the input file as previously described in section 19.3.2 are not shown.

```
# Non-isothermal planar cell/RU steady-state simulation.
# ...
<SOFC>
  # ...
  <GEOMETRY model=planar>       # Geometry to be modelled is planar, including gas flow perpendicular to the cell thickness.
      <AXIAL_SECTION>           # Define an axial section along planar cell/RU length.
          electrochemistry = n  # Set section to be electrochemically inactive, e.g. an inactive inlet flow region.
          nz = 20               # No. of axial nodes.
          length = 2.00 [cm]    # Length of the section.
      </AXIAL_SECTION>
      <AXIAL_SECTION>           # Define a second axial section along planar cell/RU length.
          electrochemistry = y  # Set section to be electrochemically active, i.e., the MEA is present.
          nz = 70               # No. of axial nodes.
          length = 9.00 [cm]    # Length of the section.
      </AXIAL_SECTION>
      <AXIAL_SECTION>           # Define a third axial section along planar cell/RU length.
          electrochemistry = n  # Set section to be electrochemically inactive, e.g. an inactive outlet flow region.
          nz = 20               # No. of axial nodes.
          length = 2.00 [cm]    # Length of the section.
      </AXIAL_SECTION>
      <INTERCONNECT>            # Define interconnect properties for a planar repeating unit simulation.
        thickness = 0.7e-03 [m] # Thickness of half the interconnect plate only not including the rib,
                                # i.e. t_ic in Fig. 19.1. The rib thickness is already set equal to the gas channel height.
        # Define Interconnect rib-electrode contact layers modelled as stripes underneath the ribs (see Fig. 18.1),
        # i.e., NOT a contact mesh or layer printed onto the full area of the electrode. Such a mesh or layer is defined
        # in the <SECTION> tag, as appropriate, instead. If contact layer thickness is unspecified, then default is 0.
        # contact_layer_an_thickness = 0.1e-03 [m] # Thickness of interconnect rib-anode contact layer.
        # contact_layer_cd_thickness = 0.1e-03 [m] # Thickness of interconnect rib-cathode contact layer.
        # For all properties defined the <Thermal_Props> tag, provide up to 6 coefficients,
        # which will be read in the form of a NASA polynom.
        <Thermal_Props>
        <Interconnect> # Define interconnect thermal properties.
          <cp> 660 0.002 </cp>  # [J/kg/K] Specific heat capacity of dense interconnect material.
          <k> 24 0.001 </k>     # [W/m/K] Thermal conductivity of dense interconnect material.
          <rho> 7700.0 </rho>   # [kg/m3] Density of dense interconnect material.
        </Interconnect>
        <Contact_Layer> # Define contact layer thermal properties.
   <k_anode> 1.0 </k_anode>     # [W/m/K] Thermal conductivity of interconnect rib-anode contact layer
   <k_cathode> 2.0 </k_cathode> # [W/m/K] Thermal conductivity of interconnect rib-cathode contact layer
        </Contact_Layer>
        </Thermal_Props>
      </INTERCONNECT>
      <HOUSING>                 # Define cell housing properties for a planar cell simulation (i.e., not planar RU).
        # thickness = 1e-03 [m] # Thickness of cell housing.
        <Thermal_Props>
        <Housing>
          <cp> 660 0.002 </cp>  # [J/kg/K] Specific heat capacity of dense cell housing material.
          <k> 5.0 0.001 </k>    # [W/m/K] Thermal conductivity of dense cell housing material.
          <rho> 7700.0 </rho>   # [kg/m3] Density of cell housing material.
        </Housing>
        </Thermal_Props>
        # T_surr = 1023.15 [K]  # Surrounding temperature.
        # htc = 0. [W/m2/K]     # Convective heat transfer coefficient of dense cell housing material.
      </HOUSING>
      <SECTION>
        name=air
        height=1.0e-03 [m]
        width=2.98e-03 [m]      # Width of the air channel.
        # ...
      </SECTION>
      <SECTION>
        name=cathode
        width=4.47e-03 [m]      # Width of planar cell/RU. Here, this value is for a RU.
        # ...
        <Electro_Props>
          # Cathode-interconnect interfacial contact resistance = k*exp(-Ea/(R*T)) or set temperature-independent to R.
          # Rcon_ede_ic = 0.01 [Ohm m2]
          # k_con_ede_ic = 0.0 [Ohm m2]
          # Ea_con_ede_ic = 0.0 [J/mol]
        # ...
        </Electro_Props>
```

```
    <Thermal_Props>
    # The thermal properties specified below are the geometric mean of the bulk solid values for material 1 and material 2.
    # They do not factor in the porosity of the electrode. That is done automatically in the code.
    # The thermal properties below need to be re-evaluated every time the volume fraction of
    # material 1 in the electrode in either the diffusion ('dl') or functional layer ('fl') is adjusted.
        <Diffusion_Layer>   # Define thermal properties of cathode diffusion layer.
         <cp> 398.0 </cp>   # [J/kg/K]
         <k> 3.5 </k>       # [W/m/K]
        <rho> 4815.0 </rho> # [kg/m3]
        </Diffusion_Layer>
        <Functional_Layer>  # Define thermal properties of cathode functional layer.
            <cp> 398.0 </cp>
            <k> 3.5 </k>
            <rho> 4815.0 </rho>
        </Functional_Layer>
    </Thermal_Props>
    # ...
  </SECTION>
  <SECTION>
    name=electrolyte
    width=4.47e-03 [m]           # Width in m of planar cell/RU. Here, this value is for a RU.
    # ...
    <PROPERTY>
      # The thermal properties specified below are the respective bulk solid values for the dense
      # electrolyte material and the barrier layers.
    <Thermal_Props>
        <Electrolyte>            # Define thermal properties of the electrolyte layer.
            <cp> 620.0 </cp>
            <k> 2.1 </k>
            <rho> 5938.0 </rho>
        </Electrolyte>
        <Barrier_Layer_Anode>    # Define thermal properties of the anode-side barrier layer.
            <cp> 521.0 </cp>
            <k> 0.98 </k>
            <rho> 7210.0 </rho>
        </Barrier_Layer_Anode>
        <Barrier_Layer_Cathode>  # Define thermal properties of the cathode-side barrier layer.
            <cp> 521.0 </cp>
            <k> 0.98 </k>
            <rho> 7210.0 </rho>
        </Barrier_Layer_Cathode>
    </Thermal_Props>
    </PROPERTY>
  </SECTION>
  <SECTION>
    name=anode
    width=4.47e-03 [m]      # Width of planar cell/RU. Here, this value is for a RU.
    <PROPERTY>
    <Thermal_Props>
        <Mesh> # Define solid phase thermal properties of the anode mesh layer. Do not factor in the porosity.
            <cp> 549.0 </cp>
            <k> 66.2 </k>
            <rho> 8850.0 </rho>
        </Mesh>
      # The thermal properties specified below are the geometric mean of the bulk solid values for material 1 and material 2.
      # They do not factor in the porosity of the electrode. That is done automatically in the code.
      # The thermal properties below need to be re-evaluated every time the volume fraction of
      # material 1 in the electrode in either the diffusion ('dl') or functional layer ('fl') is adjusted.
        <Diffusion_Layer>   # Define thermal properties of anode diffusion layer.
            <cp> 352.8 </cp>
            <k> 4.0 </k>
            <rho> 5140.4 </rho>
        </Diffusion_Layer>
        <Functional_Layer>  # Define thermal properties of anode functional layer.
            <cp> 352.8 </cp>
            <k> 4.0 </k>
            <rho> 5140.4 </rho>
        </Functional_Layer>
    </Thermal_Props>
    # ...
  </SECTION>
  <SECTION>
    name=fuel
    height=1.0e-03 [m]
    width=2.98e-03 [m]      # Width of the fuel channel.
    # ...
  </SECTION>
</GEOMETRY>
<RADIATION>
  # The values below are arbitrary.
  emissivity_ic = 0.3       # Total emissivity of interconnect material.
  emissivity_hg = 0.3       # Total emissivity of cell housing material.
  emissivity_a = 0.5        # Total emissivity of anode composite after factoring in the porosity.
  emissivity_c = 0.8        # Total emissivity of cathode composite after factoring in the porosity.
  view_fac_ede_icplate = 0.7# Based on analytical expression between two parallel rectangular plates.
  view_fac_ede_icrib = 0.1  # Based on analytical expression between two rectangular plates at 90 degrees to one another.
</RADIATION>
<ELECTROCHEM>
```

```
    # ...
  </ELECTROCHEM>
  <INIT>
    isothermal=n              # Choose wheter the simulation is isothermal or not.
    T=1123.15 [K]             # Initial planar cell/RU temp. for non-isothermal simulation.
  </INIT>
  <SOLVER>
    t_Begin = 0
    t_End = 100
    dt = 1.0d-04
    # The 5 parameters below are only necessary for non-isothermal simulations.
    tr_Begin = 0 [s]          # Start time for solid temperature solver.
    tr_End = 1000 [s]         # End time for solid temperature solver. Must be longer than the value for t_End above.
    tr_ini = 1.0e-02 [s]      # Initial time step for solid temperature solver.
    filenr = 3                # File number for non-isothermal simulations.
    max_t_step = 0 [s]        # Maximum allowed time step for solid temperature solver.
                              # If set to 0, then it uses default value of t_end - t_Begin.
    <TOLERANCE>
      abs_Tol = 1.0d-20
      rel_Tol = 1.0d-06
      res_Tol = 1.0d-06
      T_abs_Tol = 1.0d-05     # Absolute tolerance for solid temperature solver for each time step.
      T_rel_Tol = 1.0d-03     # Relative tolerance for solid temperature solver for each time step.
      T_res_Tol = 1.0d-03     # Residual tolerance for solid temperature solver over all time steps.
    </TOLERANCE>
  </SOLVER>
  <T-PROFILE>                 # Define a piecewise linear temperature profile in axial flow direction
   0.01     1123              # with an arbitrary number of pairs in the form (z-position, temperature),
   0.02     1133              # which will be set as prescribed wall (interconnect/housing) temperature.
   0.03     1143
   #...
  </T-PROFILE>
  <OUTPUT>
   species = mole
   # file = 1                 # Not required here, since files are saved with the number specified
                              # in filenr for non-isothermal simulations.
   monitor = y
 </OUTPUT>
</SOFC>
```

19.3.5 DETCHEM$^{\text{MONOLITH}}$ for SOC stack simulation

The first step to run a SOC stack simulation is to modify a single line in the planar SOC input file described in section 19.3.4 to reflect the change in geometry from a planar cell/RU to a planar stack as shown below. All other lines remain unchanged and are not shown.

```
# planar SOC input file for stack simulation
...
<GEOMETRY model=planar_stack> # geometry to be modelled is now planar_stack
...
```

The next step is to create a grid or mesh for the cross-section of the planar stack by specifying its height and width. This is done using ***gridgen.inp*** as shown below.

```
<GRIDGEN>
  <CIRCUMPOINTS>
    name="stack"
    dmax=0.02          # Maximum distance between 2 grid points.
    # Provide co-ordinates of four corners of the cross-section starting with first point at origin, i.e. 0 0
    # This cross-section is then extruded to the length of the stack given in monolith.inp
    0    0             # Bottom left corner of stack.
    0.0894    0        # Bottom right corner of stack. In this example, the width of stack is 0.0894 m.
    0.0894    0.1842   # Top right corner of the stack. In this example case, the height of the stack is 0.1842 m,
                       # which is obtained by multiplying the single RU thickness tRU (see Eq. 18.83) with the no. of cells to be stacked.
    0    0.1842        # Top left corner of the stack.
  </CIRCUMPOINTS>
    #  <LAYER>            # If insulation is to be provided around the stack.
    #    name="insu"
    #    cells=2          # No. of mesh cells along insulation thickness.
    #    thickness=0.02   # [m] Thickness of insulation layer.
    #  </LAYER>
  <BORDER>
    name="bc"
  </BORDER>
</GRIDGEN>
```

Finally, with the cross-sectional mesh created, ***monolith.inp*** can be used to specify the inputs required to run the SOC stack simulation as shown below. For details regarding the supported stack heat loss models in the **<BOUNDARIES>** tag the reader is referred to section 17.4.4.

```
# Steady-state i-V planar stack simulation with heat loss.
# Only comments relevant to running a planar soc stack simulation are provided. Look up the chapter on DETCHEM_MONOLITH
```

```
# for more details about using this input file for non-SOC stack simulations.

{include coelec_stack_adia.inp}  # Includes the planar soc input file to be run.

<MONOLITH3D>
  <PARAMETERS>
    MGRmaxR = 1000
  </PARAMETERS>

  <GRID>
    <XDOMAIN>
      from=  0.                    # Stack must start at 0, negative values possible for upstream insulation layers.
      to=  0.0894 [m]              # Length of planar stack.
      cells= 40                    # No. of mesh cells along stack length.
      aspratio=  1.                # Should be 1 for planar stack.
    </XDOMAIN>
    {include r_grid.inp}           # Includes the cross-sectional mesh file created using gridgen.inp.
    <stack>
      monolithic=yes               # Implies it is a porous domain.
      <names> wall monolith wall </names>
    </stack>
    #   <insu>
    #     monolithic = no
    #     <names> wall insu wall </names>
    #   </insu>
    <bc>
      <names> wall </names>
    </bc>
  </GRID>

  <MATERIALS>
    <TLIMITS>
      Tmin=  353. [K]
      Tmax=  2023. [K]
    </TLIMITS>

    # Thermal properties for porous stack solid layer.
    <monolith>

      # See [1] to find out how to evaluate the thermal properties of the stack.

      <k dir=axial> 15.2921 </k>    # [W/m/K] Thermal conductivity of porous stack solid composite in axial direction.
      <k dir=radial>  5.1091 </k>   # [W/m/K] Thermal conductivity of porous stack solid composite in radial direction.
      <rho>  4899.2 </rho>          # [kg/m3] Density of porous stack solid composite.
      <cp>  547.8175 </cp>          # [J/kg/K] Specific heat capacity of porous stack solid composite.
    </monolith>
    # Thermal properties for insulation layer.
    #   <insu>
    #   <k dir=axial> 0.059 </k>    # [W/m/K]
    #   <k dir=radial>  0.059 </k>  # [W/m/K]
    #   <rho>  480 </rho>           # [J/kg/K]
    #   <cp>  1047 </cp>            # [kg/m3]
    #   </insu>
  </MATERIALS>

  <BOUNDARIES>
    # <inlet type=Dirichlet>
    #   T=1200
    # </inlet>
    <wall type=Neumann>             # Flux type boundary condition with natural convection heat loss
                                    # and radiation heat loss to the stack surroundings.
      htc=  10. [W/m2/K]            # Heat transfer coefficient of stack surroundings.
      Tsurr=  1123.15 [K]           # Temperature of stack surroundings.
      cflux=  0. [W/m2/K]           # Constant heat flux supplied by/lost to the stack surroundings.
      emiss=  0.4                   # Total emissivity of porous stack solid composite.
    </wall>
  </BOUNDARIES>

  <INITIAL>
    T0= 1123.15 [K]                 # Initial stack temperature.
    # file = restart.plt            # Sets initial stack temperature to the distribution specified in restart.plt.
  </INITIAL>

  <SOLUTION>
    tend=  30000. [s]               # End time for simulation.
    dt_ini=  0.03 [s]               # Initial time step.
    dt_max=  1000. [s]              # Maximum time step.
    delta_T= 5. [K]                 # Maximum temperature change in each time step.
  </SOLUTION>

  <OUTPUT>
    write_restart=yes               # Write the restart.plt file.
    # <DO>                          # Write the soc output files and monolith temperature
    #   from=10                     # file every 25 sec from time = 10s to time = 10500 s.
    #   to=10500
    #   step=25
    # </DO>
  </OUTPUT>
```

```
  <CHANNELS>                        # Not used in soc stack simulation. Leave unchanged.
    <channel1>
      ch/cm2=  4.8581
      reverse=no
    </channel1>
  </CHANNELS>

  <CHOOSE>
    delta_T=  5. [K]                # Criteria for temperature clustering.
                                    # Here, all repeating units which have temperatures within
                                    # 5K of each other are considered to be one cluster.
                                    # In this cluster, one repeating unit will be picked as representative
                                    # of the entire cluster and solved.
    pvar_E= 50                      # Criteria for potential clustering.
                                    # Here, repeating units which have a difference in potential of up to 50% of the
                                    # difference between the highest and lowest potential cell can be clustered
                                    # together. In other words, cells of similar temperature are split in 2 clusters
                                    # according to their potential.
    pvar_v=  10.                    # Not used. Leave unchanged.
    pvar_Y=  10.                    # Not used. Leave unchanged.
    memorize=no
  </CHOOSE>

  # <POSTPROCESS>                   # Once simulation has been solved till the value of tend specified in <SOLUTION>, re-run the
                                    # simulation by uncommenting the postprocess tag and its contents to generate
                                    # SOC output files for repeating units at locations specified by the co-ordinates listed below.
  #   file= monolith001.plt
  #   0.0447  0.0921                # Centre of present 40-cell stack.
  #   0 0                           # Bottom left corner of present 40-cell stack.
  #   0.0894  0.0921                # Centre right side of present 40-cell stack.
  # </POSTPROCESS>

  <INLET type=CONSTANT>             # Constant inlet variables.
    E0 = 1.3 [V]                    # Inithial guess for the cell voltage.
    i_ref = -0.1                    # Target current density in [A/cm2]
    relax = 0.9                     # Relaxation coefficient for the potential iteration algorithm.
                                    # Choose values between 1 (time-accuracy, faster) and 0.5 (steady-state accuracy)
                                    # Choose 0 to use constant potential E0 for all cells
    <CHANNEL>
    name = fuel                     # Specify fuel channel inlet conditions.
    u0 = 0.5 [m/s]                  # Inlet velocity.
    T0 = 1123.15 [K]                # Inlet temperature.
      <molefrac>                    # Inlet fuel channel gas composition. Should be same as that in planar soc input file.
        H2   0.1
        CO2  0.323
        H2O  *
      </molefrac>
    </CHANNEL>
    <CHANNEL>
    name = air                      # Specify air channel inlet conditions.
    u0 = 1.0 [m/s]                  # Inlet velocity.
    T0 = 1123.15 [K]                # Inlet temperature.
      <molefrac>                    # Inlet air channel gas composition. Should be same as that in planar soc input file.
        O2   0.21
        N2  *
      </molefrac>
    </CHANNEL>
  </INLET>
  # <INLET type=VARIABLE>           # Variable inlet parameters. For simulation of transient stack operation,
  #   file=sweep.inp                # i.e. switching between different operating conditions.
  # </INLET>
</MONOLITH3D>
```

As stated above in the **<INLET>** tag in ***monolith.inp***, aside from steady-state i-V stack simulations, transient i-V stack simulations can also be performed by changing the type of inlet conditions from constant to variable. To specify the time-varying inlet conditions the ***sweep.inp*** must be included. An example of the file is shown below.

```
# Input file to specify time-varying inlet conditions.
<INLETINFO>
    E0 = 1.3                    # Initial guess does not need to vary with time.
    i_ref = list                # Target current density in [A/cm2] varies with time.
    relax = list                # Relaxation coefficient varies with time. Reduce gradually for better steady-state.
 <CHANNEL>
 name = fuel
 u = 0.6 [m/s]                  # Constant fuel channel inlet velocity.
 T = list                       # Fuel channel inlet temp. varies with time.
   <molefrac list>
   H2                           # Fuel channel inlet gas composition varies with time.
   H2O
   </molefrac>
 </CHANNEL>
 <CHANNEL>
 name = air
 u = 3.0 [m/s]
```

```
 T = list                      # Air channel inlet temp. also varies with time.
   <molefrac list>
    O2                         # Air channel inlet gas composition also varies with time.
    N2
   </molefrac>
 </CHANNEL>
</INLETINFO>
# <INLETDATA> Contains the time-varying inlet conditions as specified. The first column is the time read in [s],
# while the following columns denote the variables set to '=list' in <INLETINFO>, which are read top down such that
# the second line corresponds to 'i_ref' in [A/cm2], the third to 'relax', the fourth to 'T' in [K] of the fuel channel, etc.
<INLETDATA>
#   time i relax   T H2 H2O T O2 N2
  30001 -0.1 0.9 1123.15 0.5 0.5 1023.15 0.21 0.79
  30021 -0.1 0.9 1073.15 0.4 0.6 1073.15 0.22 0.78
  31021 -0.2 0.9 1073.15 0.3 0.7 1073.15 0.23 0.77
  31026 -0.2 0.8 1073.15 0.2 0.8 1073.15 0.24 0.76
  31027 -0.2 0.5 1073.15 0.1 0.9 1073.15 0.25 0.75
</INLETDATA>
```

19.4 Output

19.4.1 Screen Output

The output displayed on the screen whilst the simulation is running enables the monitoring of the simulation's progress. Figure 19.2 are screen captures of the screen outputs for different types of SOC simulations.

(a)

```
Iter   Time   Residual   CurrentDensity
-----------------------------------------
 60    2.3045E-02  5.5122E-04  3.4856E-01
 61    3.1886E-02  3.5011E-04  3.4856E-01
 62    4.3561E-02  1.9529E-04  3.4856E-01
 63    5.6799E-02  1.0103E-04  3.4856E-01
 64    7.2783E-02  4.5621E-05  3.4856E-01
 65    1.7715E-01  2.2894E-07  3.4856E-01
Volatge (V)   Current Density (A cm-2)
-----------------------------------------
 6.0000E-01  3.4856E-01
Eqm. potential of cell =   1.00557244919043
DETCHEM -- copyright by O. Deutschmann -- licence agreement required
For further information visit www.detchem.com
```

(b)

```
  2.092997E+00   2.000000E-02   2.164683E-02
  2.293490E+00   2.000000E-02   2.164602E-02
  2.520281E+00   2.000000E-02   2.164547E-02
  2.749517E+00   2.000000E-02   2.164514E-02
  3.313555E+00   2.000000E-02   2.164482E-02
  3.762235E+00   2.000000E-02   2.164471E-02
  4.139026E+00   2.000000E-02   2.164468E-02
  5.478871E+00   2.000000E-02   2.164466E-02
  6.569488E+00   2.000000E-02   2.164465E-02
  1.000000E+01   2.000000E-02   2.164465E-02
Performing Frequency transform and writing to file ...
Done!
DETCHEM -- copyright by O. Deutschmann -- licence agreement required
For further information visit www.detchem.com
```

(c)

```
Iter   Time   Residual   CurrentDensity
-----------------------------------------
  1    1.0000E-04  1.1918E+00 -2.7201E-01
  2    7.5201E-04  1.3616E-01 -2.7201E-01
  3    2.0843E-03  1.8454E-01 -2.7199E-01
  4    4.5778E-03  1.7260E-02 -2.7195E-01
  5    1.0591E-02  5.5068E-03 -2.7187E-01
  6    2.2491E-02  1.9361E-03 -2.7174E-01
  7    9.4689E-02  1.6336E-04 -2.7165E-01
  8    2.5507E-01  3.8775E-07 -2.7157E-01
Axial position (m)   Volatge (V)   Current Density (A cm-2)
-------------------------------------------------
 8.9400E-02  1.3000E+00 -2.7157E-01
H2:CO mole ratio =    1.95405047469386
\n
Calling temperature solution at    1000.00000000000      s
Temperature solution obtained!
DETCHEM -- copyright by O. Deutschmann -- licence agreement required
For further information visit www.detchem.com
```

Figure 19.2: Screen captures of the screen outputs of (a) isothermal steady-state i-V button cell simulation, (b) transient isothermal EIS button cell simulation and (c) adiabatic steady-state planar repeating unit simulation.

For the steady i-V simulations, applied cell voltage and current density are displayed for each converged step with the axial position also displayed for planar cell/RU/stack simulations as shown in Fig. 19.2 (c). The columns seen in Fig. 19.2 (b) are the integration time, applied step voltage and current response for the EIS simulation. Regardless of the quantities displayed, the most important output to note is the last two lines, viz. "DETCHEM -- copyright ....", which is common to all simulation outputs. These two lines are always output on the screen once a DETCHEM^{SOC} simulation has finished successfully.

After a simulation is complete, the various output files generated by the code can now be analysed. The naming convention of the files for an isothermal simulation is *'<file name>000<file number>.csv'* while for a non-isothermal simulation it is *'<file name>00000<file number>.csv'*. These output files are discussed below.

19.4.2 Electrochemical Output Files

The electrochemical output from the code is split amongst ***effic000*.csv***, ***asr000*.csv***, ***idist000*.csv*** and ***eis000*.csv***. The generation of these files as well as their contents are dependent on the kind of simulation that has been run.

19.4.3 *effic000*.csv*

The file ***effic000*.csv*** contains the cell's essential performance metrics and is written for non-button cell simulations. The column outputs are adapted to the fuel gas inputs (with fuel gas species k = H_2, CO, NH_3, CH_4).

1. Ecell [V] – Cell voltage
2. i_avg [A cm^{-2}] – Length-averaged current density
3. effic_elec [-] – Electrical efficiency ($=\frac{\sum_k(\dot{m}_{o,k}-\dot{m}_{i,k})LHV_k}{w_{ch}E_{cell}\int_0^{L_{cell}} idz}$ in electrolysis mode)
4. effic_tot [-] – Efficiency based on the total (heat+electrical) energy input ($=\frac{\sum_k(\dot{m}_{o,k}-\dot{m}_{i,k})LHV_k}{w_{ch}E_{cell}\int_0^{L_{cell}} idz+\sum ch\left(\dot{m}_{i,ch}\int_{T_\infty}^{T_{i,ch}} C_{p,g,ch}dT\right)}$ in electrolysis mode)
5. Conv_k [-] – Conversion of fuel gas species k ($=\frac{\dot{n}_{i,k}-\dot{n}_{o,k}}{\dot{n}_{i,k}}$)
6. Yield of fuel gas species k ($=\frac{\dot{n}_{o,k}-\dot{n}_{i,k}}{\dot{n}_{theo}}$)
7. m_change_k [kg s^{-1}] – Mass flux produced/consumed of fuel gas species k
8. Tot_conv [-] – Total conversion of all fuel gas species ($=\frac{\sum_k(\dot{n}_{i,k}-\dot{n}_{o,k})}{\sum_k \dot{n}_{i,k}}$)
9. air ratio [-] – Excess air ratio ($=\frac{\dot{n}_{i,O_2}}{\dot{n}_{o,O_2}-\dot{n}_{i,O_2}}$ in electrolysis mode)

If *"species=mole"* under the **<OUTPUT>** tag in the input file, then column 7 contains the corresponding molar flux.

19.4.4 *asr000*.csv*

As the name suggests, the file ***asr000*.csv*** contains the cell's ASR and is useful for generating Tafel plots, i.e. E_{cell} or Eta_{cell} vs. $\log i$, or to analyze the individual resistance contributions (i.e. Ohmic, activation, concentration and contact losses) that sum up to the cell's net ASR. The file is generated for non-EIS simulations. If the *"overpotential_format"* under the **<ELECTROCHEM>** tag is activated, then the columns in the file are as follows,

1. z [mm] – Axial position
2. Ecell [V] – Cell voltage
3. Eta_tot_cell [V] – Cell overpotential (= $E_{cell} - E_{cell,eq}$)
4. i [A cm^{-2}] – Current density
5. log(i) – Base 10 logarithm of i
6. eta_act_cd/an [Ω cm^2] – Anode/cathode activation overpotential
7. R_elyt [Ω cm^2] – Ohmic resistance of the Electrolyte
8. eta_conc_cd/an [Ω cm^2] – Anode/cathode concentration overpotential
9. OCV [V] – Open-circuit voltage of the cell

as well as a second set of columns output below,

1. E_cell [V] – Cell voltage
2. i [A cm^{-2}] – Current density
3. ASR [Ω cm^2] – Total area-specific resistance of the cell (= $\Delta E_{cell}/\Delta i$)

If the *"overpotential_format"* under the **<ELECTROCHEM>** tag is deactivated, then Eta_tot_cell [V] is omitted.

19.4.5 *idist000*.csv*

The file ***idist000*.csv*** contains the spatially resolved electronic and ionic phase currents and potentials. This file can be used to evaluate the electrode utilization length by plotting the distribution of ionic and electronic currents along the electrode thickness. When running a planar simulation, the columns in this file are

1. z [mm] – Axial position
2. y [μm] – Radial position along cell thickness
3. j_el [A cm^{-2}] – Local current density of the electronic phase
4. j_io [A cm^{-2}] – Local current density of the ionic phase
5. phi_el [V] – Local electric potential of the electronic phase
6. phi_io [V] – Local electric potential of the ionic phase
7. delta_phi [V] – Local electric potential difference between the electronic and ionic phases

When running a button cell simulation, column 1 is not output.

19.4.6 *eis000*.csv*

As evident from the name, ***eis000*.csv*** is generated by the code when running a transient EIS simulation. While in case a button cell simulation is performed, a single ***eis000*.csv*** file is produced, multiple ***eis000*.csv*** files are output for a planar cell simulation written at each axial position along the cell length (as defined in the **<GEOMETRY>** tag). In this case, an additional ***eis000*_avg.csv*** file is written that evaluates the global (length-averaged) impedances. In any case, the files contain the Fourier transformed electrochemical impedances of the cell in the frequency domain. The output columns in the file are

1. nu [Hz] – EIS Frequency
2. ImZ [Ω cm^2] – Imaginary part of the AC impedance of the cell
3. ReZ [Ω cm^2] – Real part of the AC impedance of the cell
4. ReZ-R_elyt [Ω cm^2] – Polarization resistance of the cell
5. -ImZ [Ω cm^2]

Since it is common to represent EIS data using either Bode or Nyquist plots, the code also generates two other files ***eis000*b.csv*** and ***eis000*n.csv***. The file ***eis000*b.csv*** contains only columns 1 and 5 of ***eis000*.csv*** for fast plotting of Bode plots while ***eis000*n.csv*** contains only columns 4 and 5 to quickly plot the electrode impedance via a Nyquist plot.

19.4.7 Gas Concentration, Surface Coverage and Temperature Output Files

air000*.csv* and *fuel000*.csv

The files ***air000*.csv*** and ***fuel000*.csv*** contains the local gas mole (or mass) fraction, velocity, mixture density and temperature in the air and fuel channels respectively. For a planar cell simulation with *"species=mole"* under the **<OUTPUT>** tag in the input file, the output columns are

1. z [mm] – Axial position
2. rho [kg m^{-3}] – Local density of gas mixture
3. u [m s^{-1}] – Local gas velocity

4. NRe [-] – Local Reynolds number (= $\rho/\mu \cdot u \cdot d_{\mathrm{h}}$)
5. T [K] – Local gas temperature
6. X_tot – Sum of mole fractions of all species in the air/fuel channel. It should equal 1.

7-n. X_k [-] – Local mole fraction of species k for all species in the air/fuel channel

n+1. X_O2_eq [-] – Local mole fraction of O_2 based on the H_2+0.5∗$O_2 \leftrightarrow H_2O$ thermodynamic equilibrium in the fuel channel.

n+2. Ecell [V] – Cell voltage

If *"species=mass"* under the **<OUTPUT>** tag in the input file, then columns 6, 7, ..., n contain mass fractions denoted by 'Y_' instead of 'X_'. For a button cell simulation, column 4 is not output, whilst column n+1 is only written in ***fuel000*.csv*** in case of a pure H_2/H_2O feed flow.

anode000.csv* and *cathode000*.csv*

The local gas mole (or mass) fraction, mixture density and temperature in the anode and cathode are output in the files ***anode000*.csv*** and ***cathode000*.csv***. For a planar cell simulation with *"species=mole"* under the **<OUTPUT>** tag in the input file, the output columns are

1. z [mm] – Axial position
2. y [μm] – Radial position
3. T [K] – Local electrode temperature
4. rho [kg m^{-3}] – Local density of gas mixture
5. X_tot – Sum of mole fractions of all species in the anode/cathode. It should equal 1.

6-n. X_k [-] – Local mole fraction of species k for all species in the anode/cathode

n+1. X_O2_eq [-] – Local mole fraction of O_2 based on the H_2+0.5∗$O_2 \leftrightarrow H_2O$ thermodynamic equilibrium in the anode.

n+2. Ecell [V] – Cell voltage

If *"species=mass"* under the **<OUTPUT>** tag in the input file, then columns 5, 6, ..., n+1 contain mass fractions denoted by 'Y_' instead of 'X_'. For a button cell simulation, column 1 is not output, whilst column n+1 is only output in ***anode000*.csv*** in case of a pure H_2/H_2O feed flow.

an_props000.csv* and *cd_props000*.csv*

The files ***an_props000*.csv*** and ***cd_props000*.csv*** output the local microstructural properties and effective transport properties used by the model in the anode and cathode respectively. For a planar non-isothermal simulation, the output columns are

1. z [mm] – Axial position
2. y [μm] – Radial position
3. prsty [-] – Local electrode porosity
4. pore_trsty [-] – Local tortuosity factor of the pore phase in electrode
5. vf_mat1 [-] – Local volume fraction of solid phase in electrode that is material 1
6. dp_mat1 [μm] – Local diameter of material 1 particles
7. dp_mat2 [μm] – Local diameter of material 2 particles
8. pore_dia [μm] – Local diameter of electrode pores
9. A_pore_mat1 [cm^{-1}] – Specific interfacial area between pore phase and material 1
10. A_pore_mat2 [cm^{-1}] – Specific interfacial area between pore phase and material 2
11. A_mat1_mat2 [cm^{-1}] – Specific interfacial area between material 1 and material 2
12. vl_3PB_an/vl_3PB_cd [cm^{-2}] – Volumetric three phase boundary length in the anode and cathode respectively
13. C_dl_an/C_dl_cd [F m^{-2}] – Area-specific double layer capacitance
14. sa_el/sc_el [S cm^{-1}] – Electronic conductivity of anode and cathode respectively

15. sa_io/sc_io [S cm^{-1}] – Ionic conductivity of anode and cathode respectively
16. Cp_eff [J kg^{-1} K^{-1}] – Mean specific heat capacity of electrode
17. k_eff [W m^{-1} K^{-1}] – Mean thermal conductivity of electrode
18. Ecell [V] – Cell voltage

If the simulation is isothermal and the geometry is a button cell, then columns 1, 16 and 17 are not written.

covs000*a.csv

The fractional coverages of species on the surface of material 1 (Ni by default) in the anode are output in ***covs000*a.csv***. This file is only created if *"surfchem"* is activated in the input file. The output columns for a planar simulation are

1. z [mm] – Axial position
2. y [μm] – Radial position
3. Th_tot – Sum of fractional coverages of all species on the surface of material 1 in the anode. It should equal 1.

4-n. Th_k [-] – Local fractional coverage of species k on the surface of material 1 for all species in the anode

n+1. Ecell [V] – Cell voltage

Tic_elyt000*.csv*, ***Thg_elyt000*.csv **and** ***Telyt000*.csv***

The files ***Tic_elyt000*.csv***, ***Thg_elyt000*.csv*** or ***Telyt000*.csv*** are only created when running non-isothermal simulations and contains the temperatures of different solid phase cell components. When running an adiabatic planar RU simulation (i.e. with interconnects), the columns of ***Tic_elyt000*.csv*** contain the interconnect (ic) and electrolyte (elyt) temperatures,

1. z [mm] – Axial position
2. y [μm] – Radial position
3. Tic_fc [K] – Local temperature of fuel channel side interconnect
4. Tic_ac [K] – Local temperature of air channel side interconnect
5. Telyt [K] – Local temperature of electrolyte and the barrier layers

If the simulation is for a planar cell containing a cell housing (hg), ***Thg_elyt000*.csv*** is generated instead, and columns 3 and 4 contain the cell housing temperatures Thg_fc [K] and Thg_ac [K]. In case of no cell housing, the file ***Telyt000*.csv*** is output instead, and it does not contain columns 3 and 4.

eheat000*.csv

The file ***eheat000*.csv*** is only created when running non-isothermal simulations. It isolates the various heat fluxes in the cell/RU. The output columns are

1. z [mm] – Axial position
2. ehem_rxn_heat [W m^{-3}] – Overall volumetric electrochemical reaction enthalpy
3. ehem_heat_diss [W m^{-3}] – Overall volumetric heat dissipation due to entropy change
4. ovp_heat_loss [W m^{-3}] – Overall volumetric heat flux due to ohmic and activation overpotentials
5. surf_rxn_heat_anode [W m^{-3}] – Overall volumetric reaction enthalpy due to thermochemical reactions on the surface of material 1 in the anode
6. h_convection [W m^{-3}] – Overall volumetric heat flux due to convection from the gas channels to the electrodes
7. h_radiation [W m^{-3}] – Overall volumetric heat flux due to radiation from the interconnects or cell housing to the electrodes
8. h_loss [W m^{-3}] – Overall volumetric heat loss from the cell housing to the surroundings.

If the column values are positive, then heat is released by the electrodes and if the values are negative then heat is adsorbed from the electrodes. The latter column is only output in case a cell housing is applied.

global.plt

Additionally, a ***global.plt*** file is output whenenver a non-isothermal simulation is run. It can be used to monitor the transient simulation progress, or to plot SOC performance characteristics as a function of time. The file is written in TECPLOT format and contains the following data: *time(s)*, cell voltage *Ecell(V)*, length-averaged current density *Iavg(A cm^{-2})* and spatially-averaged solid phase (electrolyte) temperature *Telyt(K)*.

19.5 Examples

There are several examples in the distribution. These include various relevant types of SOC simulations. The requisite ***sofc.inp*** or ***monolith.inp*** input files, species and mechanism information (***species.inp*** and ***mech.inp***) along with the necessary databases ***thermdata*** and ***moldata*** are included, while the output files are also present.

19.6 Running the code

After configuring the build and compiling the code, the binary ***soc*** should be created in the build directory. To run the code with a ***go*** script in the example directory, it must point to the location of the executable ***soc***. Then, to run DETCHEM^{SOC}, the user can simply write *"./go<space><name of input file>"* in the command shell and execute. For example, if the name of the input file is *"eis_test.inp"*, the executable command in the shell is *"./go<space>eis_test.inp"*. If the program argument is omitted, then *"sofc.inp"* will be run by default.

To run a stack simulation, the script ***go_grid*** must be executed (*"./go_grid"*) first before the ***go*** script can be executed (*"./go"*). The script ***go_grid*** must point to the location of the binary ***gridgen3d*** while the ***go*** script points to the location of the binary ***monolith_soc***. Both these binaries are created in the build directory on compiling the respective DETCHEM targets.

References

[1] A. Banerjee, Y. Wang, J. Diercks, O. Deutschmann, Hierarchical modeling of solid oxide cells and stacks producing syngas via H2O/CO2Co-electrolysis for industrial applications, Appl. Energy. 230 (2018) 996–1013.

[2] A. Bertei, C. Nicolella, Percolation theory in SOFC composite electrodes: Effects of porosity and particle size distribution on effective properties, J. Power Sources. 196 (2011) 9429–9436.

[3] A. Bertei, A. Barbucci, M.P. Carpanese, M. Viviani, C. Nicolella, Morphological and electrochemical modeling of SOFC composite cathodes with distributed porosity, Chem. Eng. J. 207–208 (2012) 167–174.

[4] P. Vijay, M. O. Tad, Z. Shao, M. Ni, Modelling the triple phase boundary length in infiltrated electrodes, Int. J. Hydrogen Energy 42 (2017) 28836-28851.

[5] K. Herrera Delgado, H. Stotz, L. Maier, S. Tischer, A. Zellner, O. Deutschmann, Surface Reaction Kinetics of Steam- and CO2-Reforming as well as Oxidation of Methane over Nickel-based Catalysts, Catalysts 5 (2015) 871-904.

[6] Z. Zhang, C. Karakaya, R. J. Kee, J. D. Way, C. A. Wolden, Barium-Promoted Ruthenium Catalysts on Yittria-Stabilized Zirconia Supports for Ammonia Synthesis, ACS Sustainable Eng. 7 (2019) 18038-18047.

[7] H. Zhu, R. J. Kee, V. M. Janardhanan, O. Deutschmann, D. G. Goodwin, Modeling elementary heterogeneous chemistry and electrochemistry in solid-oxide fuel cells, J. Electrochem. Soc. 152(12) 2005 A2427-A2440.

[8] A. Banerjee, O. Deutschmann, Elementary kinetics of the oxygen reduction reaction on LSM-YSZ composite cathodes, J. Catal. 346 (2017) 30–49.

[9] W. G. Bessler, Rapid Impedance Modeling via Potential Step and Current Relaxation Simulations, J. Electrochem. Soc. 154 (2007) B1186.

Chapter 20

DUO

20.1 Introduction

20.1.1 Background

The code **DUO** was developed to calculate the chemistry and the mass/heat transfer in monolithic reformers. The aim is to allow the user to optimise the design and the operating conditions of reformers based on numerical calculations.

At the entrance of a reformer, the flow is distributed to the channels. On one hand, the flow region upstream of the monolith is often geometrically complex but does not involve complex chemistry. Sometimes, the flow circulates around the monolith. On the other hand, the flow field in the single channels of the monolith is often simple (1D or 2D), but the chemistry is complex. An iterative coupling algorithm was developed that models the geometrically complex flow outside the monolith with one code and calculates the chemically complex reforming process in the channels of the monolith with another code. For this, the chemical models are designed to accommodate detailed gas-phase and catalytic reaction kinetics, possibly including hundreds of species and thousands of reactions. The developed software tool **DUO** controls the interaction between the two programmes.

20.1.2 Modelling of a Monolithic Reformer

Modeling practical systems requires the coupling of complex flow configurations with complex chemical-kinetics. From the point of view of simulation, reformers based on mini channels can be described with the following characteristics:

- three-dimensional laminar flow
- coupled heat transfer inside and between three-dimensional fluid and solid regions
- complex chemistry of gas-phase reactions
- complex chemistry of catalyzed surface reactions
- exchange of heat with the surroundings
- availability of temperature-dependent properties for all the compounds involved.

For the solution of problems of this type, there are, in principle, several CFD (Computational Fluid Dynamics) codes available (e. g., FLUENT STAR-CD, CFX). However, there are three constraints which restrict the simulation:

- availability of reliable, detailed gas-phase and surface mechanisms
- availability of computer power
- availability of finanical resources.

Reformers that are based on monolithic structures include thousands of equally sized small channels. To resolve these structures with numerical methods, grids with high spatial resolution are needed, which leads to large counts of cells. As the type of the basic differential equations is elliptical, the solution process is always iterative. The catalytic reforming kinetics of practical fuels (e.g., diesel) depend on gas-phase and heterogeneous reaction mechanisms, which may involve over a thousand elementary reactions. For each of the hundreds of species, a transport equation in a CFD code has to be solved. This would lead to simulation times in the time-span of weeks up to months or years, even if a large number of processors (e.g., several thousands) were available.

Therefore, direct coupling of chemistry at this level greatly exceeds the capability of computational fluid dynamics (CFD) models which can handle geometric complexity, but not in combination with complex chemistry. Therefore, the development and investigation of reformer systems using solely a common CFD code is not practical. Beyond that, the licensing of a commercial CFD code is very expensive and not suitable for small companies.

20.1.3 Simulation Tools DUO and DC4OpenFOAM

DUO stands for the coupling of the two computer codes DETCHEM Und (German for *and*) OpenFOAM and is a synonym for the joint utilisation of these two programmes. OpenFOAM is an open source CFD tool which enables the calculation of three-dimensional fluid and solid regions. It is free of cost. DETCHEM is a purchasable package of tools specifically designed for the modeling and simulation of reacting flows, particularly for heterogeneous systems such as catalysis, materials synthesis and fuel cells. The simulation tool **DUO** is the gateway between the two codes. The basic idea is to combine the advantages of the two programmes.

Parts of the DETCHEM software can be used in different ways additionally to OpenFOAM. The user has to distinct between two different approaches **DUO** and **DC4OpenFOAM**, which can be applied solely or together.

In **DUO** single tools of DETCHEM are called as external executables from OpenFOAM (*external coupling*). These tools calculate flow and the complex reactions in straight channels in one or two dimensions in space (axial or axial/radial). This feature can be used to calculate monolithic structures with many channels or tubes. In this case the flow regions inside the channels are *not* included in the CFD grid. The data exchange between the codes is managed with ASCII data files.

In **DC4OpenFOAM** some features of DETCHEM are provided by calling a precompiled shared library (*internal coupling*). This feature is used to calculate the fluid properties and the source terms of the chemical reactions in arbitrary regions. This capability is needed, if gas phase reactions in flow regions outside of the channels of a monolith (e. g. upstream of a monolith) have to be considered. Additionally, the surface reactions at walls can be calculated. Therefore, it is also possible to calculate the full three-dimensional flow and the chemistry inside the channels of a monolith. In this case the flow regions inside the single channels would be included in the CFD grid.

Both tools, **DUO** and **DC4OpenFOAM** can be used independently or can be mixed up. The approach allows, for example, the calculation of a reformer in which the combined gas phase and surface chemistry is confined within geometrically simple tubes or channels and additional reactions take place in the gas phase out of the monolith and at the outer surface. In figure 20.1, a schematic drawing of a monolith with nine channels located in a surrounding flow field is shown. The flow covers the region upstream of the monolith as well as the channels in the monolith. Flow over the external surfaces of the catalyst tubes may be used to achieve thermal control. For example, hot exhaust products from an SOFC tail-gas combustor can be used to support endothermic steam reforming. Heat transfer occurs inside the fluid and the solid regions as well as at the interfaces between these regions.

20.1.4 How DUO and DC4OpenFOAM work

A: DUO (External Coupling to OpenFOAM)

As mentioned above, the task is to model a reformer based on a monolithic structure. The approach exploits the structure of reformers in which the combined gas-phase and catalytic chemistry is confined within geometrically simple tubes or channels. In figure 20.1, a schematic drawing of a monolith with nine channels located in a surrounding flow field is shown. The flow covers the region upstream of the monolith as well as the channels in the monolith. Flow over the external surfaces of the catalyst tubes may be used to achieve thermal control. For example, hot exhaust products from an SOFC tail-gas combustor can be used to support endothermic steam reforming. Heat transfer occurs inside the fluid and the solid regions as well as at the interfaces between these regions.

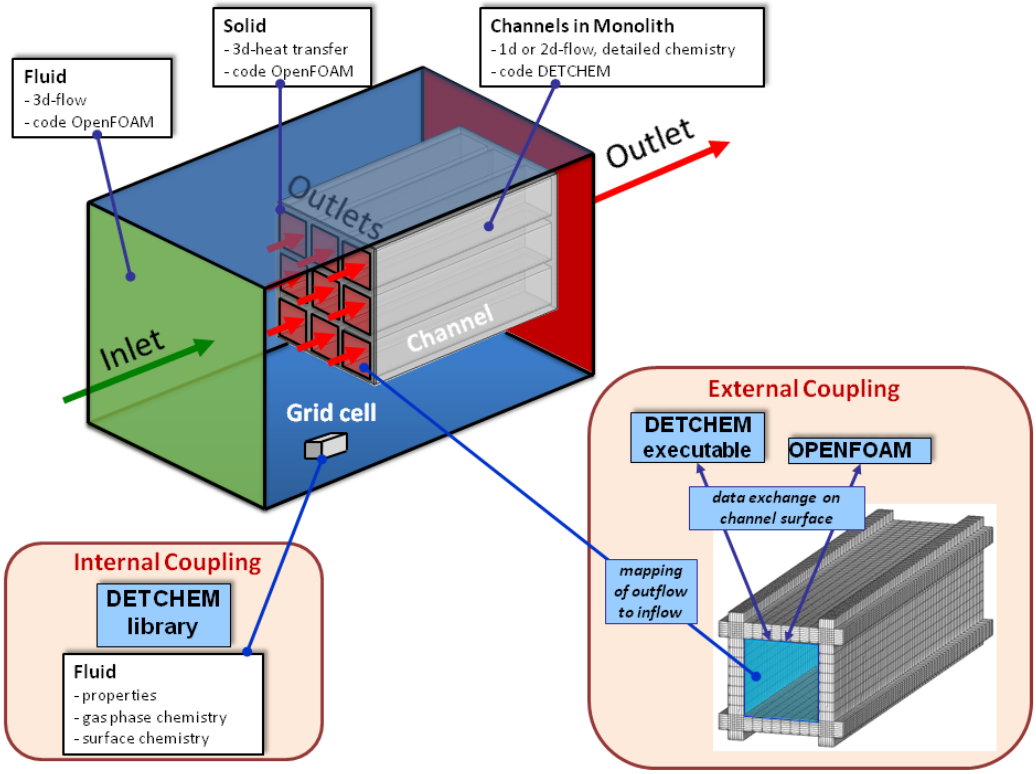


Figure 20.1: Typical configuration of a calculation domain

Using **DUO**, the flow outside of the monolith (see figure 20.1) is calculated with the CFD code OpenFOAM, which handles essentially arbitrary geometry. In addition to the outer fluid flow, the code also solves the conjugate heat transfer in the solid structure of the monolith. This implies the handling of the heat transfer between the solid and fluid regions as well as the exchange of heat with the external environment. In the outer flow regions, the chemistry is often simple or can be fully neglected.

The flow inside the channels is calculated with DETCHEM codes by applying detailed gas-phase and surface chemistry. As the flow is handled as a one or two-dimensional flow, only a simple grid is used. This makes the calculation fast.

DUO handles the exchange of data at the inner surface of the channels between the two codes (see figure 20.2). **DUO** provides the interface between the geometrically complex three-dimensional outer mesh and the one-dimensional axial mesh for the catalyst tubes. The internal chemistry problem is solved using axial wall temperature profiles that are specified by OpenFOAM. Tube wall heat fluxes are predicted as a result of the chemically reacting flow problem within the tubes. These flux profiles are supplied to OpenFOAM by DETCHEM. OpenFOAM then solves the external problem using the tube-wall heat fluxes as boundary conditions. The outer flow solution provides new tube-wall temperature profiles which are used to solve the chemistry problem. The iterative process continues until convergence between the inner and outer problems is achieved. The capability of **DUO** is quite general in the sense that there are essentially no restrictions on the external flow of geometry or the chemistry and transport within the tubes. Therefore **DUO** cannot only be used to model reformers, but also can be applied to simulate arbitrary systems where monolithic structures with internal chemistry are utilised.

DUO accomplishes two additional functions. First, the algorithm must interpolate between the three-dimensional mesh that is used by OpenFOAM and the one-dimensional band mesh used by DETCHEM. Second, **DUO** has to coordinate the iteration strategy between the outer CFD and the catalytic chemistry inside the tubes.

In addition to the fluid flow regions, the CFD mesh resolves all the solid regions. The CFD mesh contains cylindrical “cut-outs” corresponding to the inner walls of the tubes. In other words, the inner tube walls are seen as boundaries for the CFD problem. The cylindrical CFD face mesh on the inside of each tube wall is sectioned into a user-specified number of bands. The band spacing (one-dimensional mesh) is established to resolve gradients that characterize the interior catalytic chemistry problem. In the following discussion, “grid” refers to

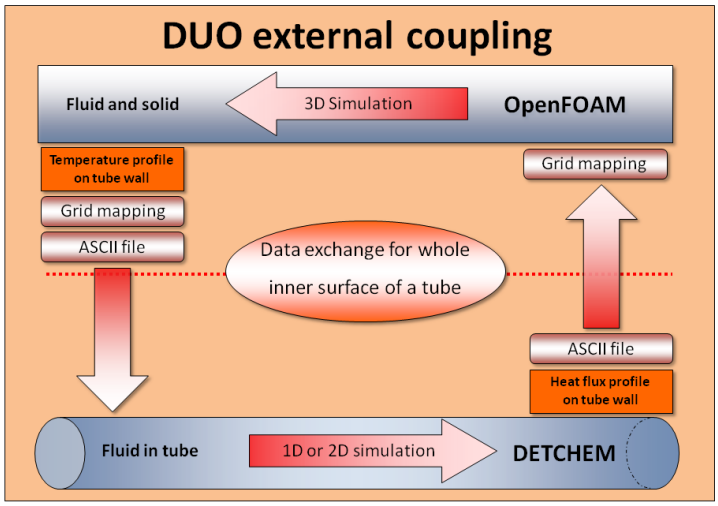


Figure 20.2: External Coupling between DETCHEM and OpenFOAM

the one-dimensional band discretization and "mesh" refers to the collection of faces in the CFD discretization of the tube wall. An integer band number is identified as the linearly increasing cell index for the grid, with the first band in the first grid cell being located at the tube inlet. **DUO** then loops over all mesh faces on a cylindrical tube and calculates the corresponding band number by projecting the face centroid onto the 1D grid.

Figure 20.3 illustrates the relationships between a general triangular surface mesh on a tube and the one-dimensional band grid. All triangular faces whose centroid touches the band are assigned to that band index. As can be seen, no special alignment is needed between the CFD mesh and the band grid. At the beginning of each solver iteration, **DUO** calculates face-averaged temperatures for all bands in a tube. These band averages correspond to azimuthally averaged temperatures on the tube wall faces in the band. **DUO** then creates a file with the list of the band temperatures and executes codes form DETCHEM. Typically, the chemistry model uses a spatial discretization which is different from the CFD grid. Therefore, interpolation of the temperature from the band grid to the chemistry grid is required.

If the user wishes, the conditions at the outflow areas of the CFD mesh, which cover the same areas as the inflows for the single channels, are also written to additional files and used for the calculation of the channels (s. Fig. figure 20.1). After that, **DUO** executes the code DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$ for each channel. When the tube simulation is complete, the tool creates a file containing the heat flux profile for the band grid. **DUO** then analyses these heat-flux profiles and assigns the fluxes for each band to the corresponding collection of mesh faces in that band. This procedure is repeated for all tubes. The CFD code solves the outer flow and heat transfer problem with the prescribed heat fluxes as boundary conditions. The iterative process of averaging tube temperatures in bands, solving the catalytic chemistry problem, transferring the heat fluxes back to the CFD model, and solving the outer flow and temperature is repeated until convergence is achieved.

DUO can perform a user-specified number of CFD flow and thermal iterations between tube chemistry simulations. The inner tube chemistry calculation is typically run to convergence for each intermediate tube wall temperature profile as provided by the CFD model. In contrast, the CFD solution is always iterative, and hence, overall convergence is accelerated by performing several CFD iterations between running the chemistry model if the tube calculations are computationally expensive relative to the CFD calculations. It is possible that a reactor has a very large number of tubes such that modeling detailed chemistry in all the tubes is impractical. Therefore, the algorithm is generalized so that such groups of tubes can be modeled with one representative tube. The chemistry problem is solved for this representative tube, with the resulting heat fluxes being distributed back to all the tubes in the group.

B: DC4OpenFOAM (Internal Coupling to OpenFOAM)

In the standard form OpenFOAM calculates the temperature dependent properties only via the Sutherland law. If the user wants to calculate a monolith with flow regions upstream or around a monolith, it might be insufficient to use this rough approach. Additionally chemical reactions may be of interest in these flow regions. There are solvers in OpenFOAM which can handle gas-phase reactions. But until now there is no capability to handle

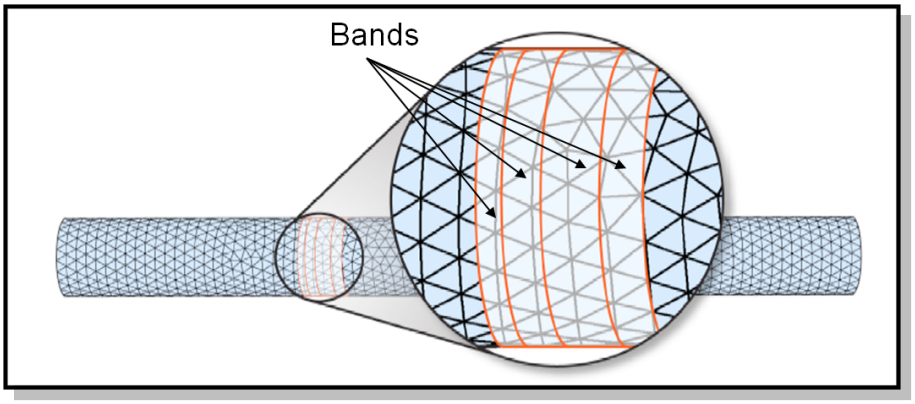


Figure 20.3: Mapping CFD mesh to 1D-grid

surface reactions. Within **DC4OpenFOAM** the calculation of properties and gas phase source terms is managed using a precompiled shared library which was made out of several parts of the DETCHEM toolbox. Figure 20.4 shows the principal structure of the *Internal Coupling*.

For each cell of the fluid domain calls to subroutines which are implied in the library are performed. For each cell the variables the pressure, the gas temperature and the concentrations of the gas phase species have to be provided to the library. From that, the properties heat conductivity, dynamic viscosity and the multi component diffusion coefficients of the gas phase species are send back to the CFD-code. The same holds for the calculation of the source terms of the gas phase kinetics. For the calculation of the surface reactions terms the temperature of the wall is needed additionally.

C: Mixing of External and Internal Coupling

The two couplings can be mixed up. If we want, for example, to calculate a reformer, where the flow in the channels of the monolith should be calculated only in one or two dimensions the *external coupling* (**DUO**) is sufficient. If we want also to consider the flow upstream of the monolith, we can do this within the range of the *external coupling* as long we don‘t have to include chemical reactions in this regions and the determination of the temperature dependent properties based on the Sutherland law is enough. If we want to imply chemical reactions of the gas phase in the upstream region of a monolith and/or to determine the properties in more detail (could be important for mixture effects), we have to include additionally the *internal coupling* **DC4OpenFOAM**. If we want to calculate the flow in the single channels of the monolith in three dimensions including gas phase and surface reactions we have to use the *internal coupling* **DC4OpenFOAM** solely.

20.2 Programming of DUO

In this chapter, some basic information about the programming is given. **DUO** is a collection of subroutines which allows to create a new *solver* or *application* in the world of OpenFOAM. For the external coupling of DETCHEM some changes were made in the original codes DETCHEM$^{\text{PLUG}}$ and DETCHEM$^{\text{CHANNEL}}$ to allow the exchange of data. The code is written in the language C++. The code lines are well commented. In general, the user should not change any code lines. For a deeper understanding of the code or information about the capabilities and the execution of OpenFOAM, the user should refer to the manual.

20.2.1 OpenFOAM

OpenFOAM (Open Field Operation and Manipulation) is a free, open source CFD software package developed by the OpenFOAM Team at SGI Corp and distributed by the OpenFOAM Foundation. It has a large user base across most areas of engineering and science, from both commercial and academic organisations. OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetics. It includes tools for meshing and for pre- and post-processing. Almost everything (including meshing as well as pre- and post-processing) runs in parallel by default, enabling users to take full advantage of the computer hardware at their disposal.

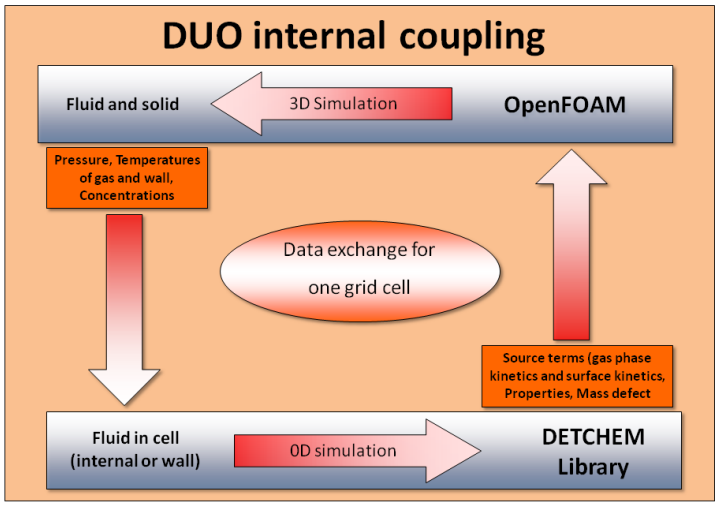


Figure 20.4: Internal Coupling between DETCHEM and OpenFOAM

By being open-source, OpenFOAM offers users complete freedom to customise and extend its existing functionality, either themselves or through support from OpenCFD. The code follows a highly modular design in which collections of functionality (e.g., numerical methods, meshing, physical models, etc.) are each compiled into their own shared library. Then, executable applications are created that are simply linked to the library functionality OpenFOAM, which includes over 80 solver applications that simulate specific problems in engineering mechanics and over 170 utility applications that perform pre- and post-processing tasks, e.g., meshing, data visualisation, etc.

For further information concerning the functionality and use of OpenFOAM, the user should refer to the appropriate manual.

DUO stands for the coupling of the two computer codes DETCHEM Und (German for *and*) OpenFOAM and is a synonym for the joint utilisation of these two programmes. Generally, we use the name **DUO** if we speak about the coupling of DETCHEM and OpenFOAM.

OpenFOAM is a C++ library used primarily to create executables, known as *applications*. OpenFOAM is distributed with a large set of precompiled applications but users also have the freedom to create their own or modify existing ones. The *applications* fall into two categories: *solvers*, that are each designed to solve a specific problem in continuum mechanics, and *utilities*, that are designed to perform simple pre-and post-processing tasks, mainly involving data manipulation and algebraic calculations. The OpenFOAM distribution contains numerous solvers and utilities covering a wide range of problems.

In this terminology, **DUO** is firstly a *solver*. Some coding was done by creating new subroutines which can be used to create a new *solver* or *application*. More precisely, **DUO** is a *solver* made of modules available in OpenFOAM. In the following, the terms *application* and *solver* are used synonymously. Additional code was written to handle the coupling. The resulting *solver* **DUO** calculates laminar flows with reactions in gas phase as well as surface reactions and conjugate heat transfer of connected fluid and solid regions. It calls DETCHEM$^{\mathrm{PLUG}}$ or DETCHEM$^{\mathrm{CHANNEL}}$ as external programs.

20.2.2 Code Files

The file containing the class definition takes a ***".C"*** extension, e.g., a class *nc* would be stored in the file ***nc.C***. As a means of checking errors, the piece of code being compiled must know that the classes it uses and the operations they perform actually exist. Therefore, each class requires a class declaration, contained in a header file with a ***".H"*** file extension, e.g., ***nc.H*** includes the names of the class and its functions. This file is included at the beginning of any piece of code using the class, including the class declaration code itself. Any piece of ***".C"*** code can resource any number of classes and must begin with all the ***".H"*** files required to declare these classes. The classes, in turn, can imply other classes and begin with the relevant ***".H"*** files. By searching recursively down the class hierarchy, we can produce a complete list of header files for all the classes on which the top level ***".C"*** code ultimately depends; these ***".H"*** files are known as the dependencies. With a dependency

list, a compiler can check whether the source files have been updated since their last compilation and selectively compile only those that need to be updated. Header files are included in the code using ♯ include statements, e.g., ♯ *include "otherHeader.H"* causes the compiler to suspend the reading from the current file and to read the file specified instead. Any self-contained piece of code can be put into a header file and included at the relevant location in the main code in order to improve code readability. The application directories can be located anywhere but it is recommended they be within an *application* subdirectory of the user's project directory, i.e., `$HOME/OpenFOAM/$USER-1.7.1`. In the following, the folder where the applications can be found is called `<appDir>`. The code files which were additionally programmed in OpenFOAM to perform the external coupling to DETCHEM within the *application* ***chtMultiRegionSimpleFoamChemDuo*** are described shortly in the following table. They can be found in the directory `<appDir>/chtMultiRegionSimpleFoamChemDuo`.

Solver class files	
chtMultiRegionSimpleFoamChemDuo.C	Includes the OpenFOAM solver. Calls all subsequent routines of **DUO**.
Header files for coupling	
readCouplingParameters.H	Reading and interpretation of keyword file ***couplingParameters***.
initializeCoupling.H	Initialisation of coupling.
callDetchem.H	Calls DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$ and copies input, output and exchange files.
writeOutletConditions.H	Calculates average conditions at outlets of CFD grid for all coupled tubes and writes to files. The data is used to set the inlet conditions for the DETCHEM calculations.
writeWallTemperatures.H	Calculates band-averaged wall temperatures for all coupled tubes and writes them to intermediate files.
Class and header files for boundary handling	
detchemHeatFluxTemperatureFvPatchScalarField.C *detchemHeatFluxTemperatureFvPatchScalarField.H*	Reads wall heat fluxes from intermediate files and maps them on the appropriate cells of the wall patches.

20.3 Usage of DUO

20.3.1 General

As **DUO** is the coupling of two codes, each code can be used as a stand-alone tool. For each program, several directories and files have to be present. For the use of each of the two programs, OpenFOAM and DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$, the user should refer to the appropriate manuals. In this section, only the parts which are necessary for the control of the coupling between the two codes are described.

20.3.2 Installation and Test

A: Folder structure

Figure 20.5 shows the final structure of the folders.

The folders `application` and `run` have to be copied to the folder `$HOME/OpenFOAM/$USER-1.7.1` on your system. The folder `DETCHEM` can be copied to anywhere you want.

B: Operating System

The installation was tested on two different PC systems A and B (s Tab. 3 2). In both cases version 1.7.1 of OpenFOAM was used. This version needs the compilers gcc-4.5 and g++-4.5. This led problems in configuration A, because the standard version of the compilers for this linux version is 4.6.2. For the installation and compiling of each code, the user should refer to the appropriate manuals. However, the installation including some basic commands which are necessary to compile the codes will be given in this chapter.

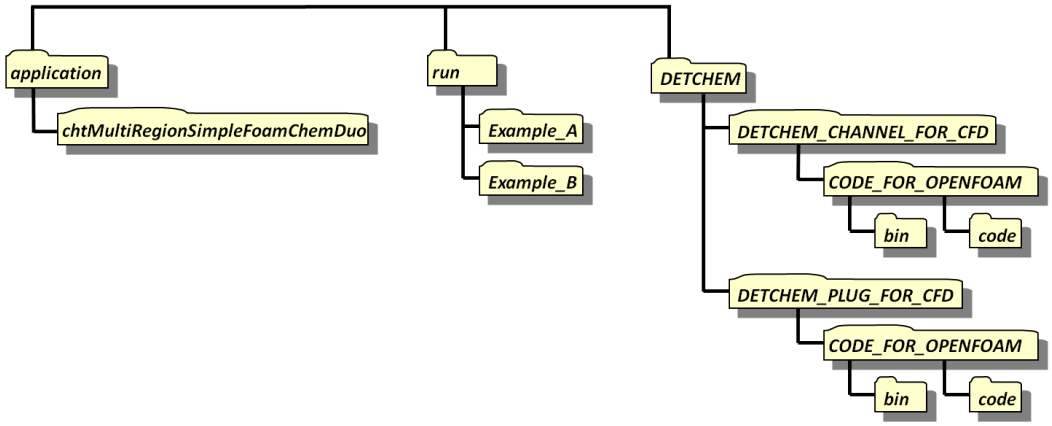


Figure 20.5: Folder Structure

	System A	**System B**
Kernel Name	Linux 3.1.10-1.16-desktop	Linux 2.6.34-12-desktop
System software	openSUSE 12.1 (x86_64)	openSUSE 11.3 (x86_64)
OpenFOAM version	OpenFOAM1.7.1	OpenFOAM1.7.1
Compiler for OpenFOAM	gcc (SUSE Linux) 4.5.0 g++ (SUSE Linux) 4.5.0	gcc (SUSE Linux) 4.5.0 g++ (SUSE Linux) 4.5.0
Compiler for DETCHEM	gfortran	pgf90

C: Compiling of the individual Codes

To perform **DUO**, it is necessary to install the following resources:

1. C++ resources:
   - CFD-Code OpenFOAM-1.7.1
   - OpenFOAM-1.7.1 solver ***chtMultiRegionSimpleFoamChemDuo***
2. Fortran90 resources:
   - DETCHEM executable ***plug***
   - DETCHEM executable ***channel***

Compiling of executable DETCHEM^{PLUG}

DETCHEM^{PLUG} can be compiled in the directory `<caseDir>/detchem/code` by typing the command ***make plug***. The executable ***plug*** is moved automatically to the folder which is one level higher `<caseDir>/detchem`.

Compiling of executable DETCHEM^{CHANNEL}

DETCHEM^{CHANNEL} can be compiled in the directory `<caseDir>/detchem/code` by typing the command ***make channel***. The executable ***channel*** is moved automatically to the folder which is one level higher `<caseDir>/detchem`.

Compiling of Solver *chtMultiRegionSimpleFoamChemDuo*

The solver ***chtMultiRegionSimpleFoamChemDuo*** can be compiled in the directory `<appDir>/chtMultiRegionSimpleFoamChemDuo` by typing the command ***wmake***. If the code changes, it might be helpful to use the commands ***wmake clean*** and ***rmdepall*** firstly.

D: Test of the individual Codes

Before testing **DUO**, the user should ensure that the application ***chtMultiRegionSimpleFoamChemDuo*** and the executables DETCHEM^{PLUG} and DETCHEM^{CHANNEL} work separately.

Test of DETCHEM^{PLUG}

DETCHEM^{PLUG} can be tested in the directory <caseDir>/detchem by typing the command ***plug***. ***plug*** is the executable which has to be compiled before running **DUO** in the directory <caseDir>/detchem/code. Running the code alone (without coupling) by using an input file ***plug.inp*** using all keywords, tags and includes necessary for a normal calculation allows to test the functionality.

Test of DETCHEM^{CHANNEL}

DETCHEM^{CHANNEL} can be tested in the directory <caseDir>/detchem by typing the command ***channel***. ***channel*** is the executable which has to be compiled before running **DUO** in the directory <caseDir>/-detchem/code. Running the code alone (without coupling) by using an input file ***channel.inp*** using all keywords, tags and includes necessary for a normal calculation allows to test the functionality.

Test of OpenFOAM

After reading the instructions in the OpenFOAM manual the user can install and test the system by starting a test case of the standard installation.

Test of DUO

For the test of **DUO** for a new case a grid for a geometry consisting of a fluid part and a solid part has to be generated. All directories and files which are necessary for an OpenFOAM calculation have to be prepared. Firstly, the *application* **DUO** should be tested without any coupling. The internal name of the *application* **DUO** is ***chtMultiRegionSimpleFoamChemDuo***. Each *application* is designed to be executed from a terminal command line, typically reading and writing a set of data files associated with a particular case. The command to start the OpenFOAM application in <caseDir> is: ***chtMultiRegionSimpleFoamChemDuo -case .*** .

To check the functionality of the application, all the requirements for the start of an OpenFOAM run have to be fulfilled. For testing, the coupling has to be disabled in the file <caseDir>/couplingParameters:

```
coupling    off; .
```

20.3.3 Preprocessing

A: Grid Generation in ICEM

OpenFOAM is able to use any mesh which was saved in the format needed for the code FLUENT. The grid can generated with GAMBIT, ICEM or any other software. In the actual section some important facts for the grid generation with the software ANSYS-ICEM are given.

1. Create two bodies named FLUID and SOLID for the two regions which shall be filled with flowing material and solid material. Normally, the body FLUID is inherently present. Add all blocks to the appropriate materials.

2. Each interface for which an *external coupling* is wished in **DUO** has to be in an extra part. That means, that if a monolith has 100 channels you have to define 100 parts. Every interface between fluid and solid for which surface chemistry shall be calculated using the *internal coupling* has to be in one extra part.

3. Convert the mesh to an unstructured mesh using the right mouse click on the entry *PreMesh* in the task bar.

4. Check the mesh carefully inside ICEM. No errors should arise.

5. On the strength of past experience it might happen, that an check in ICEM shows no errors whereas the check in OpenFOAM fails. If possible, firat load the mesh into FLUENT and perform an additional mesh check. If FLUENT accepts the mesh, OpenFOAM is also satisfied.

6. For writing out the grid the following parameters have to be used in the appropriate menu. Use for the item *Output Solver* the option *Fluent_V6* and for the item *Common Structural Solver* the option *ANSYS*. Be sure that the grid has the unit [m]. If not, scale the grid using a scaling factor in the menu for the output.

B: Grid Check in OpenFOAM

To use the grid (which was written in format for *FLUENT Version 6*) the data format has to be transformed, the regions have to be split and the mesh has to be checked. This can be done with the following steps:

1. **Generate case folder**

   Generate the folder <casDir>/system. Copy the following files into this folder (best from a previous calculation): ***controlDict***, ***fvSchemes***, ***fvSolution***.

2. **Transform the mesh**

   Perform the OpenFOAM application: ***fluent3DMeshToFoam <grid file name>.msh -case***. Check carefully the output. If the code recognizes the two materials, some information should be printed which looks like:

   ```
   Zone: 32 name: INTERFACE_SOLID_FLUID type: interface. Reading zone data...done.
   ...
   FINISHED LEXING
   ...
   Creating faceZone 2 name: INTERFACE_SOLID_FLUID type: interface
   ```

3. **Split mesh**

   Perform the OpenFOAM application: ***splitMeshRegions -cellZones -overwrite***. Check carefully the output. If the code handles the two regions rightly, some information should be printed which look like this:

   ```
   Region  Cells
   ------  -----
   0       729
   1       729
   Region  Zone    Name
   ------  ----    ----
   0       0       FLUID
   1       1       SOLID
   Sizes inbetween regions:
   Region  Region  Faces
   ------  ------  -----
   0       1       81
   ```

   Additionally you should find something like this:

   ```
   For interface between region 0 and 1 added patch 8 FLUID_to_SOLID
   ```

4. **Check mesh**

   Perform OpenFOAM application: ***checkMesh -constant***. Check carefully the output. If something is wrong here, the calculation will not work.

C: Boundary Conditions in OpenFOAM

For the "normal" boundary conditions see the example case for the two-zone solver ***chtMultiRegionSimpleFoam***. The "special" boundary conditions, which are necessary to manage the coupling are described later.

D: Directory Structure of a Case (DUO and DC4OpenFOAM)

DUO and **DC4OpenFOAM** need various objects (data files, keyword files, executables, libraries). The structure of a folder <caseDir> is shown in figure 20.6. There are many other objects needed for the calculation. In the folder only the most important files are shown. The objects in grey colour are needed for a calculation of a system with two regions. In this manual they are always named as FLUID and SOLID. The coloured objects have to be changed for the setup of a new case. They should be copied from one of the example cases and adapted afterwards. The objects, their functionality and content are explained in the following sections. The reader should always refer to figure 20.6 to get an overview about the data structure.

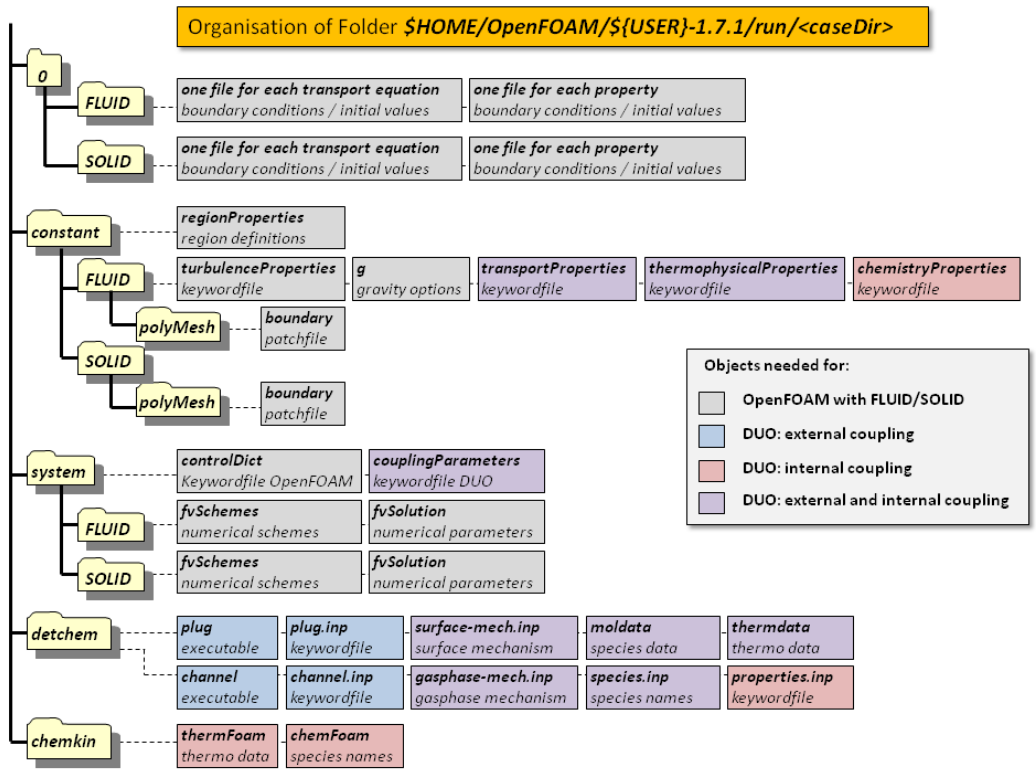


Figure 20.6: Folder structure of a calculation case

E: Data Input for OpenFOAM

Directories and files for the solver The coupling of the codes OpenFOAM and DETCHEM is an *application* in OpenFOAM. The *application*, or *solver*, directories can be located anywhere but it is recommended they be within an application subdirectory of the user's project directory, i.e., `$HOME/OpenFOAM/$USER-1.7.1/-application`. In the following, this folder is called <appDir>. OpenFOAM *applications* are organised using a standard convention according to which the source code of each application is placed in a directory whose name is that of the *application*. The top-level source file takes the application name with the ***".C"*** extension. The *application* name is ***chtMultiRegionSimpleFoamChemDuo***. Therefore, the folder where the application is located has the same name. The directory structure is as follows:

- <appDir>/chtMultiRegionSimpleFoamChemDuo
  - Make
  - fluid
  - solid
  - derivedFvPatchFields

The class and header files of the *application* **DUO**, which are responsible for solving and *external* coupling, are directly located in the folder dirchtMultiRegionSimpleFoamChemDuo. The code files for the handling of the wall patches are located in the subdirectory dirderivedFvPatchFields.

Directories and files for the calculation

A calculation with OpenFOAM including all files is called a ***case***. Normally, a user would assign a name to a calculation ***case***, e.g., ***reformer***. This name becomes the name of a directory in which all the case files and subdirectories are stored. The case directories themselves can be located anywhere but it is recommended they be within a `run` subdirectory of the user's project directory, i.e., *HOME*/*OpenFOAM*/`USER-1.7.1.`One advantage of this is that the `$FOAM_RUN` environment variable is set to `$HOME/OpenFOAM/$USER-1.7.1/run` by default; the user can quickly move to that directory by executing a preset alias, ***run***, at the command line.

In the following, the case folder is called <caseDir>. As the coupling of the codes in **DUO** is controlled via OpenFOAM, the appropriate folder structure for a calculation case also has to be applied. The basic directory structure for a *case* whose name is *reformer* is as follows:

- run/reformer
  - constant
  - system
  - <time>
  - detchem

Directory	Content
constant	This directory contains a full description of the case mesh in a subdirectory polyMesh and files specifying physical properties for the application concerned, e.g. ***transportProperties***.
system	This directory is for setting parameters associated with the solution procedure itself. It contains at least the following files: ***controlDict***, in which run control parameters are set including start/end time, time step and parameters for data output; ***fvSchemes***, in which discretisation schemes used in the solution may be selected at run-time; and ***fvSolution***, in which the equation solvers, tolerances and other algorithm controls are set for the run. Additionally, the keyword file ***couplingParameters*** for control of **DUO** has to be located in this folder.
<time>	These directories contain individual files of data for particular fields. The data can be: either initial values and boundary conditions that the user must specify to define the problem or results written to file by OpenFOAM. Note that the OpenFOAM fields must always be initialised, even when the solution does not strictly require it, as in steady-state problems. The name of each time directory is based on the simulated time at which the data is written. Since we usually start our simulations at time t = 0, the initial conditions are usually stored in a directory named 0 or 0.000000e+00, depending on the name format specified. For example, the velocity field U and the pressure field p are initialised from files 0/FLUID/***U*** and 0/FLUID/***p***, respectively.
detchem	This directory contains all the files necessary for the *external* coupling of DETCHEM. This includes the executables named ***plug*** and/or ***channel*** as well as all other files required for a calculation, such as thermodynamic data, mechanisms, property file and the input files ***plug.inp*** and/or ***channel.inp***. Also, all result files from the DETCHEM calculations can be found in this folder.

1. Perform DETCHEM$^{\text{CHEMINP}}$ which is a pre-processor and translator for DETCHEM species and mechanism information files. DETCHEM$^{\text{CHEMINP}}$ requires one input file named ***cheminp.inp***. This file acts as a container for the species and mechanism information. If this information is stored in external files, e.g. species.inp and mech.inp, then ***cheminp.inp*** may simply look like the following example.

   ```
   <CHEMINP>
   {include species.inp}
   {include mech.inp}
   </CHEMINP>
   ```

   Like in all other DETCHEM applications, the files ***thermdata*** and ***moldata*** have to be present. The library DETCHEM$^{\text{CHEMINP}}$ produces several output files. The file ***chem.new*** contains the species and the mechanism information in a compact form. The files ***thermdata.new*** and ***moldata.new*** contain the thermodynamic data and the molecular properties (respectively) of the required species. The last file is not needed here.

2. Rename the files ***chem.new*** and ***thermdata.new*** to ***chem.inp*** and ***therm.dat***. Remove all information about mechanisms from the file ***chem.new***. Only the parts describing the elements and the species are needed.

3. Apply the OpenFOAM application ***converseChemkinToFoam*** to transform the files into a format which can be used form OpenFOAM. The output will be the two files ***chemFoam*** and ***thermFoam***. However, the user could also generate the files ***chem.inp*** and ***therm.dat*** using CHEMKIN files.

Important!

The user has to guarantee, that the count, the names and the order of the species in the file <caseDir>/chemkin/***chemFoam*** are the same as in the file <caseDir>/detchem/***properties.inp***. The species N_2 should be the last in each case.

F: Data Input for DETCHEM

Before running ***plug*** (which is the executable of DETCHEM^{PLUG}), the user must prepare an input file ***plug.inp*** in the folder <caseDir>/detchem. A typical input file ***plug.inp*** with explanations is shown in the appendix. Before running ***channel*** (which is the executable of DETCHEM^{CHANNEL}, the user must prepare an input file ***channel.inp*** in the folder <caseDir>/detchem. A typical input file ***channel.inp*** with explanations is shown in the appendix. As the input file is the same for all calculated tubes, the file contains so-called *includes*. These tags allow the substitution of values in the input file with values written into intermediate files by **DUO**. Additionally, the following data files are necessary:

thermdata	The thermodynamic database contains enthalpy, entropy and heat capacity data for each species.
moldata	This file contains the molar mass and kinetic theory parameters of chemical species.
gasphase-mech.inp	The gas-phase chemistry mechanism file lists the elementary reactions occurring in the gas phase. This file can be give any name, and this name should be mentioned in the ***mech.inp*** file.
surface-mech.inp	The surface chemistry reaction mechanism file ***surface-mech.inp*** lists the elementary reactions occurring on the surface. This file can be given any name and this name must be mentioned in the ***mech.inp*** file.
species.inp	For any DETCHEM application, the user needs to define the species that are to be considered. In many cases, it is advisable to put the declaration into a separate file, e.g. ***species.inp***.

20.3.4 Control of DUO

As usual, OpenFOAM is managed with the parameters in the ASCII-file ***controlDict*** which can be found in the folder <caseDir>/system. In case of *external coupling* OpenFOAM calls the executables ***plug*** or ***channel*** from DETCHEM. The coupling between the two codes is managed using the ASCII keyword list ***couplingParameters*** which is located in the same directory. Among others, two special types of keywords are used: *dictionaries* and *lists*.

Special boundary conditions for coupling

DUO handles the exchange of data at the inner surface of the channels between the two codes (see figure 20.2). In addition to the fluid flow regions, the CFD mesh resolves all the solid regions. The CFD mesh contains cylindrical "cut-outs" corresponding to the inner walls of the tubes. In other words, the inner tube walls are these boundaries for the CFD problem where data with DETCHEM is interchanged. To handle the data transfer new boundary conditions for OpenFOAM have been defined. They can be found in the directory <appdir>/chtMultiRegionSimpleFoamChemDuo/derivedFvPatchFields/ detchemHeatFluxTemperature. Every tube interchange surface has to be defined with the boundary condition name detchemHeatFluxTemperature in the file <time>/SOLID/***T***. Here an example for the boundary WALL_0101. The values for the parameters K, q, *gradient* and *value* are arbitrary, because they are not used.

```
WALL_0101
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
```

Dictionaries

OpenFOAM uses *dictionaries* as the most common means of specifying data. A *dictionary* is an entity that contains a set of data entries. The keyword entries follow the general format:

```
<keyword>  <dataEntry1> ... <dataEntryN>;}
```

Most entries are single data entries of the form of:

```
<keyword> <dataEntry1>;
```

Most OpenFOAM data files are dictionaries containing a set of keyword entries. Dictionaries provide the means for organising entries into logical categories and can be specified hierarchically so that any dictionary can contain one or more *subdictionary* entries. The format for a dictionary is to specify the dictionary name followed by the entries enclosed in curly braces "{}" as follows:

```
<dictionaryName>
{
... keyword entries ...
};
```

List Structures

A second keyword structure is a *list*. The format of a *list* comprises the specification of the list name followed by the entries enclosed in round parentheses "()" as follows:

```
<listName>
(
... entries ...
);
```

Keywords for DUO

The next table shows all keywords and data elements in the file ***couplingParameters*** required by **DUO**. The column **type** describes the value of the **<dataEntry>**. Entries of the type *R* demand a real value after the keyword name, and those of the type *I* demand an integer value. Keywords of the type *C* demand a character string. The type *Dict* stands for a *dictionary*. The type *List* contains multiple elements of the same type (each of type *C* or *I*). In the following, the term *patch* is used to define a boundary condition. For the purpose of applying boundary conditions, a boundary is generally broken up into a set of *patches*. One *patch* may include one or more enclosed areas of the boundary surface which do not necessarily need to be physically connected.

Caution: All keywords have to be included in the file. There are no default values. The dictionaries and lists also have to be present. If they are not used, they have to be empty (e.g. `namesOfCoupledOutlets( );`).

Keyword structure	Type	Unit	Meaning	Values
General				
`coupling`	*C*	-	Switches the external coupling to DETCHEM and controls which DETCHEM code will be coupled.	plug, channel, off
`debugLevel`	*I*	-	Controls the amount of information output.	0, 1, 2
`coupleOutlets`	*C*	-	Switches mapping of outlets. Requires the dictionary `mapOutletsToDetchemTubes` and the list `namesOfCoupledOutlets.`	on, off
Geometry				

directionOfPatches	*C*	-	Global direction of the wall patches coupled to DETCHEM tubes.	X, x, Y, y, Z, z
lengthOfDetchemTubes	*R*	m	Length of the coupled tubes.	any
numberOfDetchemBands	*I*	-	Number of bands used in DETCHEM calculations.	any
Boundary mapping				
numberOfDetchemTubes	*I*	-	Total number of tubes for which DETCHEM is called.	any

mapPatchesToDetchemTubes
{
namesOfReferencePatches
namesOfCoupledPatches
mapsToDetchemTubes
}
	Dict	-	Dictionary to control for which wall patches a DETCHEM calculation is performed. Requires the lists:
			namesOfReferencePatches
			namesOfCoupledPatches
			mapsToDetchemTubes.
				-

namesOfReferencePatches (Name_1 Name_2);	*List*	-	For each of the wall patches Name_x a DETCHEM calculation is performed. The number of entries in the list has to be the same as the integer numberOfDetchemTubes.	-
namesOfCoupledPatches (Name_1 Name_2);	*List*	-	For each of these wall patches with the name Name_x a DETCHEM calculation is applied. The number of entries in the list has to be the same as the number of entries in the list mapsToDetchemTube.	-
mapsToDetchemTubes (Number_1 Number_2);	*List*	-	For each of these wall patch names given in namesOfCoupledPatches, a tube number has to be allocated. For the equivalent wall patch, the result of a DETCHEM calculation is applied. The number of entries in the list has to be the same as the number of entries in the list namesOfCoupledPatches.	-

mapOutletsToDetchemTubes
{
namesOfCoupledOutlets
}
	Dict	-	Dictionary to control which outlets are used as inlets in the DETCHEMcalculations. Requires the list namesOfCoupledOutlets.	-

`namesOfCoupledOutlets ( Name_1 Name_2 );`	*List*	-	For each of the boundary outlets `Name_x` area-averaged values are calculated and used as inlet conditions by DETCHEM. If `coupleOutlets` is active, the number of entries in the list has to be same as the integer `numberOfDetchemTubes.`	-
Numerical parameters				
`repetitionFactor`	*I*	-	DETCHEM is called at every n^{th} iteration	any

20.3.5 Start of a Calculation with DUO

Start the OpenFOAM application **DUO** by typing the command ***chtMultiRegionSimpleFoamChemDuo***. All the keywords required for the coupling to DETCHEM (see table Tab. 4 1) have to be present in the file `<caseDir>/`***couplingParameters***. Now, the entry for the keyword `coupling` has to be:

`coupling on;` .

For the first run, it is advisable to activate additional output by setting

`debugLevel 2;` .

You could also create an executable shell script named ***start*** with the following entry:

```
#!/bin/csh
chtMultiRegionSimpleFoamChemDuo .
```

If you start the calculation with the command

```
Start > log & ,
```

the job will work in the background and the output will be written into the file ***log***.

20.3.6 Data Exchange

The external coupling between the two codes is done using ASCII data files. These files are written into the folder `system` of the OpenFOAM case directory `<caseDir>`. The procedure of data exchange between several directories might seem a little bit illogical at first sight. This is because the input and output from/to other directories than normally used from OpenFOAM (e.g. to the folder `detchem`) is complicated to handle. Therefore the exchange of data is managed within the folder `system`.

A: Intermediate Data from OpenFOAM to DETCHEM

For each iteration or time step, OpenFOAM creates two input files for every coupled tube. In the following, the symbol *** stands for the identification number of a tube. The temperatures at the outer walls of the coupled tubes are written to the file

`<caseDir>/system/`***walltemp-to-DETCHEM-****** .

The conditions at the outlets of the CFD mesh which shall be used as inlet conditions in the DETCHEM calculations are written to the file

`<caseDir>/system/`***outlet-to-DETCHEM-****** .

For every time step in which a DETCHEM program is called, these files are copied to the folder `<caseDir>/detchem` with a new name (cut of identification number). This is illustrated in the following, in which → represents the copying process:

system/walltemp-to-DETCHEM- → *detchem/walltemp-to-DETCHEM***
system/outlet-to-DETCHEM- → *detchem/outlet-to-DETCHEM***

The filenames ***walltemp-to-DETCHEM*** and ***outlet-to-DETCHEM*** are linked as includes in the files ***plug.inp*** and ***channel.inp*** and interpreted from the codes ***plug*** or ***channel***.

B: Intermediate Data from DETCHEM to OpenFOAM

For each calculation of a tube with the compiled codes ***plug*** or ***channel*** in the directory `detchem`, a file `<caseDir>/detchem/`***heatflux-from-DETCHEM*** containing the profile of the heatflux at the tube wall is created. Afterwards, this file is copied to the folder `<caseDir>/system` with a new name. The file name is ***heatflux-from-DETCHEM-*****, in which the symbol *** stands for the identification number of the tube. This is illustrated in the following, in which → represents the copying process:

$$\textit{detchem/heatflux-from-DETCHEM} \rightarrow \textit{system/heatflux-from-DETCHEM-***} \quad .$$

The heatfluxes in the files ***heatflux-from-DETCHEM-***** are red and mapped to the wall patches of the boundary handling subroutine *detchemHeatFluxTemperature_FvPatchScalarField.C*.

20.3.7 Data Output

A: Information Output

During the analysis of the keywords in the file ***couplingParameters***, the code checks the user parameters and stops if something is specified wrongly. Additionally, several quantities and variables are checked during the calculation. The code stops if something seems to be false. By setting the value of the keyword `debugLevel`, **DUO** produces additional output coming from OpenFOAM, which is always helpful for debugging. All output is displayed on the screen.

Additionally, DETCHEM produces screen output depending on the monitor option specified with the tag **<OUTPUT>** in the input files ***plug.inp*** or ***channel.inp***. With **monitor=1**, the screen output reports the iteration count and the temperature. With **monitor=2**, it reports the iteration count, temperature and the species mass fractions. With **monitor=no** no additional output is generated.

B: Solution Data from OpenFOAM

The `<time>` directories contain the result files. For an example case with the name reformer, the velocity field U for the timestep 10 s is stored in the file `run/reformer/10/`***U***. The solution files can be converted to the TECPLOT format using the utility ***foamToTecplot360***.

C: Solution Data from DETCHEM

Depending on the options set in the input file, DETCHEM$^{\text{PLUG}}$ or DETCHEM$^{\text{CHANNEL}}$ produces several file outputs. All the output files are in the TECPLOT format. In the following, the symbol *** stands for the value of the tag **Filenumber** in ***plug.inp*** or ***channel.inp***. The output files ***outg***.plt*** list velocity, residence times, temperature density and the gas-species mass or mole fractions, depending on whether the species tag in the **<OUTPUT>** element is supplied with mass or mole option. If the **coverage** option is chosen, the files ***outs***.plt*** will be produced, which list the surface coverages along the axial position of the reactor. If the **transfercoeff** option is chosen, the files ***trans***.plt*** will be produced and will list the mass and heat transfer coefficient calculated using the correlations. The file ***summary.dat*** contains the exit temperature and composition of the species listed in the **<SUMMARY>** tag. If the **<SUM>** tag is additionally used, the file ***summary.dat*** contains the sum of the mass/mole fraction of the species listed in the **<SUM>** tag.

20.4 Appendix

20.4.1 Example of a Keyword File *couplingParameters*

In the following, an example of a parameter file named ***couplingParameters*** is printed and described. The syntax is the same as in C++, because the file is handled as an include file. Lines with in the first column and between the symbols /* and */ are comments.

The sequence of the parameters and the positions of the dictionaries are arbitrary. The sequence of the lists in each dictionary and the sequence of entries inside a list are also in a random order.

```
/*---------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  1.7.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
```

```
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      couplingParameters;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

//==================================
//STANDARD PARAMETERS
//==================================
coupling              plug;
coupleOutlets           on;
repetitionFactor        10;
debugLevel               1;
directionOfPatches       z;
numberOfDetchemTubes     2;
lengthOfDetchemTubes  0.01;
numberOfDetchemBands    50;

//==================================
//SUBDICTIONARY 1:
//MAPPING PARAMETERS FOR WALLS
//==================================
mapPatchesToDetchemTubes
{
   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   namesOfReferencePatches
   (
   WALL_0202
   WALL_0204
   );

   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   namesOfCoupledPatches
   (
   WALL_0201
   WALL_0202
   WALL_0203
   WALL_0204
   );

   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   mapsToDetchemTubes
   (
   1
   1
   2
   2
   );
}

//==================================
//SUBDICTIONARY 2:
//MAPPING PARAMETERS FOR OUTLETS
//==================================
mapOutletsToDetchemTubes
{
   //-------------------------------------
   //WORD LIST
   //-------------------------------------
   namesOfCoupledOutlets
   (
   FLUID_OUT_0202
   FLUID_OUT_0204
   );
}

// ************************************************************************* //
```

File Header

After the several comment lines usually used in OpenFOAM files, a block starts, in which some information about the file can be found. In this part, the user does not have to change anything. Next, the standard parameters for the coupling can be found. Make sure that each line ends with the symbol "*;*". After that, the dictionaries including the parameters for the control of the patch mappings are shown.

Standard parameters

The coupling to DETCHEM itself is activated (`coupling on`). The coupling of outlets to DETCHEM inlets is active (`coupleOutlets on`). A calculation with **plug** or **channel** is performed at every 10th timestep (`repetitionFactor 10`) of OpenFOAM. The amount of information output is moderate (`debugLevel 1`). The tubes point into the direction z (`directionOfPatches z`). The count of tubes of the monolith for which a DETCHEM-calculation is performed is 2 (`numberOfDetchemTubes 2`) , The length of the tubes in DETCHEM and OpenFOAM is 0.01 m (`lengthOfDetchemTubes 0.01`). The number of bands or tube wall sections in the DETCHEM calculations is 50 (`numberOfDetchemBands 50`).

Dictionary mapPatchesToDetchemTubes

The word list `namesOfReferencePatches` includes the names of two wall patches. For each of these patches, a DETCHEM calculation of a tube is performed. This means, that for each of the tubes, the distribution of the wall temperature is taken from the corresponding wall patch of OpenFOAM. Caution: the count of names in the word list has to be the same as the value of the parameter `numberOfDetchemTubes`.

The word list `namesOfCoupledPatches` includes the names of four wall patches. For each of these patches, a result of a DETCHEM calculation is applied. For each of the tubes, the distribution of the wall temperature is taken from the corresponding wall patch of OpenFOAM. This means that the wall heat flux of a wall patch is set using a result from a DETCHEM calculation. For each of the coupled patches, a mapping number has to be allocated. Caution: each reference patch is also a coupled patch.

The word list `mapsToDetchemTubes` includes four mapping numbers. Each number belongs to one of the names of the coupled patches. The number represents the index of the tube calculated for which the resulting heat flux is mapped to the wall patch. Caution: each coupled patch has to have its own number. This means that the counts of elements in the list `namesOfCoupledPatches` and the list `mapsToDetchemTubes` have to be the same.

Again, the explanation of the wall mappings:

For the two wall patches `WALL_0202` and `WALL_0204`, two DETCHEM calculations for two tubes will be performed. For each of the calculations, the boundary temperature of the corresponding patch is used as a boundary condition. The result (boundary heat flux) of the first tube calculation is used for TWO boundary patches in OpenFOAM, for `WALL_0201` AND `WALL_0202`. Both patches have the same tube number “1” in the list `mapsToDetchemTubes`. The result of the second tube calculation is used for TWO boundary patches in OpenFOAM, for `WALL_0203` AND `WALL_0242`. Both patches have the same tube number “2” in the list `mapsToDetchemTubes`.

Dictionary mapOutletsToDetchemTubes

The word list `namesOfCoupledOutlets` includes the names of two outlet patches. For each of these patches, the outflow conditions (average velocity and averaged temperature) are calculated and used as input for the DETCHEM calculations. For the calculation of tube 1, the outflow conditions of the patch `FLUID_OUT_0202` are applied, for the calculation of tube 2 the outflow conditions of the patch `FLUID_OUT_0204`.

Boundary conditions

Every tube interchange surface has to be defined with the boundary condition name `detchemHeatFluxTemperature` in the file `<time>/SOLID/`$\boldsymbol{T}$. For the actual case the following entry would be necessary.

```
WALL_0201
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
WALL_0202
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
WALL_0203
    {
        type            detchemHeatFluxTemperature;
        K               K;
```

```
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
WALL_0204
    {
        type            detchemHeatFluxTemperature;
        K               K;
        q               uniform 0;
        gradient        uniform 0;
        value           uniform 0;
    }
```

The values for the parameters *K*, *q*, *gradient* and *value* are arbitrary, because they are not used.

20.4.2 Example of an Input File *plug.inp*

In the following, an example of a parameter file named ***plug.inp***, which is necessary for **DUO** to call the executable DETCHEM$^{\text{PLUG}}$, is printed and described. This file has to be located in the folder `<caseDir>/detchem`.

```
{verbose n}
{include species.inp}
{include mech.inp}
{include walltemp-to-DETCHEM}
{include outlet-to-DETCHEM}

<SURFACE-MODEL>
  <CHEMSURF>
    hini=1.d-10
    time=10.
  </CHEMSURF>
  Fcatgeo=5
</SURFACE-MODEL>

<PFR>
 <GEOMETRY >
   reactor=cylindrical
  <SECTION>
    length =0.0002
    rin=0.000796
    rout=0.000796
    isothermal=n
    adiabatic=n
    mass-transfer=n
    axial-temp-profile=n
    gaschem=n
    surfchem=y
    tw={get T1}
  </SECTION>
...
    <SECTION>
    length =0.0002
    rin=0.000796
    rout=0.000796
    isothermal=n
    adiabatic=n
    mass-transfer=n
    axial-temp-profile=n
    gaschem=n
    surfchem=y
    tw={get T50}
  </SECTION>
</GEOMETRY>
  <INLET>
    u0={get Vinlet}
    p0=1.e+05
    T0={get Tinlet}
    <MOLEFRAC>
     CH4    0.133
     O2     0.0666
     N2     *
    </MOLEFRAC>
   </INLET>
  <SOLVER>
     ini_step=1e-16
     max-h = 2e-4
    <TOLERANCE>
        s_aTol=1.0d-06
        s_rTol=1.0d-05
        t_aTol=1.0d-06
        t_rTol=1.0d-06
        al_aTol=1.0d-06
        al_rTol=1.0d-06
     </TOLERANCE>
   </SOLVER>
```

```
     <OUTPUT>
        file_num={get Filenumber}
        title="CH4"
#       coverage = y
        species = mole
        monitor=no
     </OUTPUT>
</PFR>
```

Description

For the explanation of the keywords, the user should read the manual of DETCHEM^{PLUG}. Here, only the entries which are essential for the performance of **DUO** are explained.

An important fact is that we have a tube which is divided in 50 geometrical sections or bands. The definitions for each band can be found between the tags **<SECTION>** and **</SECTION>** (only two of them are shown in the example).

It should be recapitulated that the calculations for all coupled tubes (or channels) take place in the same directory using the same input file ***plug.inp***. The names of the include files are therefore always constant. But the content of the include files changes for each tube because it is overwritten by **DUO** for each DETCHEM calculation.

The included files are linked in the first lines. In the file ***walltemp-to-DETCHEM***, the wall temperatures for the bands and the tube identification number can be found. Here is an example of the content of the file:

```
{Define Filenumber 2}
{Define T1 1050.}
...
{Define T50 1000.}
```

In the file ***outlet-to-DETCHEM***, the inlet conditions for the tubes can be found. Here is an example of the content of the file:

```
{Define Tinlet 836.70239}
{Define Vinlet 0.54443955}.
```

The name **<name of variable>** of each variable (**T1**, **T2**, **Filenumber**, **Vinlet**, **Tinlet**) is referred to by using the tag {**get <name of variable>**} in ***plug.inp***. This means that the value of the variable is inserted here while running DETCHEM^{PLUG}. Therefore, the "real" content of the file ***plug.inp*** is different for every call.

Caution

If the coupling of the outlets is NOT activated in the parameter file ***couplingParameters*** (`coupleOutlets off`), the values for **u0** and **T0** have to be set by values. Additionally, the appropriate include has to be removed or to be commented (**#**{**include outlet-to-DETCHEM**}).

20.4.3 Example of an Input File *channel.inp*

In the following, an example of a parameter file named ***channel.inp***, which is necessary for **DUO** to call the executable DETCHEM^{CHANNEL}, is printed and described. This file has to be located in the folder `<caseDir>/-detchem`.

```
{verbose n}
{include species.inp}
{include mech.inp}
{include walltemp-to-DETCHEM}
{include outlet-to-DETCHEM}

<SURFACE-MODEL>
  <CHEMSURF>
   hini=1.d-10
   time=1.d0
  </CHEMSURF>
   Fcatgeo=185
  <MIXED_DIFF>
    <ZONE>
       ngrid=10
       aspect=1
       thickness=4.d-5
     </ZONE>
     porosity=0.5
```

```
     tau=3
     diameter=2.3e-8
     <SOLVER>
       time=10
       hini=1.d-4
       rtol=1.d-4
       atol=1.d-20
     </SOLVER>
  </MIXED_DIFF>
   EFF_MODEL = O2
</SURFACE-MODEL>

 <CHANNEL Version=2.0>
 <BASICS>
   title="CH4_auf_Rh"
   ngrid=18
   zmax=0.01
   rmax=0.0004384
 </BASICS>

 <SECTION>
   gaschem=n
   <WALL>
     surfchem=y
     mechanism="Rh"
   </WALL>
 </SECTION>

 <SOLVER>
   hini=1.d-10
   hmax=0.01
 </SOLVER>

 <OUTPUT>
   outg=y
   outs=y
   outflux=y
   summary=n
   file={get Filenumber}
   copy=n
   monitor=1
   molefrac=y
 </OUTPUT>

<INLET>
  u0={get Vinlet}
  p =1.e+05
  T0={get Tinlet}
  <MOLEFRAC>
  # C/O: 1.0
  CH4  0.13
  O2  0.0666
  N2  *
</MOLEFRAC>
</INLET>

<TPROFILE>
{include points.txt}
</TPROFILE>
</CHANNEL>
```

The file ***points.txt*** includes the temperatures of the 50 bands or sections which were choosen here to exchange data between the codes.

Caution: DETCHEM$^{\text{CHANNEL}}$ needs here 51 data points, because the temperature is specified at the boundaries of the sections. The temperatures calculated in OpenFOAM are the average temperatures of each band. Therefore we get 50 values. We use the temperature of band 50 for the left and the right boundary of the last section. Contrary, in DETCHEM$^{\text{PLUG}}$, the temperature is defined for the whole band.

```
<DISCRETE>
0.0 {get T1}
0.0002 {get T2}
0.0004 {get T3}
....
0.0096 {get T49}
0.0098 {get T50}
0.01 {get T50}
</DISCRETE>
```

Description

For the explanation of the keywords, the user should read the manual of DETCHEM$^{\text{CHANNEL}}$. Here, only the entries which are essential for the performance of **DUO** are explained.

In contrast to DETCHEM^{PLUG} we need to define only one section. The number of bands or sections where we want to define a temperature is defined in between the keywords **<TPROFILE>** and **</TPROFILE>**.

It should be recapitulated that the calculations for all coupled tubes (or channels) take place in the same directory using the same input file ***channel.inp*** The names of the include files are therefore always constant. But the content of the include files changes for each tube because it is overwritten by **DUO** for each DETCHEM^{CHANNEL} calculation.

The included files are linked in the first lines. In the file ***walltemp-to-DETCHEM***, the wall temperatures for the bands and the tube identification number can be found. Here is an example of the content of the file:

```
{Define Filenumber 2}
{Define T1 1050.}
...
{Define T50 1000.}
```

In the file ***outlet-to-DETCHEM***, the inlet conditions for the tubes can be found. Here is an example of the content of the file:

```
{Define Tinlet 836.70239}
{Define Vinlet 0.54443955}.
```

The name **<name of variable>** of each variable (**T1**, **T2**, **Filenumber**, **Vinlet**, **Tinlet**) is referred to by using the tag {**get <name of variable>**} in ***channel.inp***. This means that the value of the variable is inserted here while running DETCHEM^{CHANNEL}. Therefore, the "real" content of the file ***channel.inp*** is different for every call.

Caution

If the coupling of the outlets is NOT activated in the parameter file ***couplingParameters*** (`coupleOutlets off`), the values for **u0** and **T0** have to be set by values. Additionally, the appropriate include has to be removed or to be commented (**#**{**include outlet-to-DETCHEM**}).

Chapter 21

DETCHEM Parameters

Up to version 2.1 DETCHEM was mainly written in Fortran 77, which did not offer dynamic memory allocation. In order to give more flexibility to the user, version 2.2 is based on Fortran 90 and can no longer be compiled using a Fortran 77 compiler. The main parameters (e. g. number of species and reactions) can now be changed during run-time. Recompilation is no longer necessary.

The default size parameters set in the distribution of DETCHEM may be not sufficient to meet the needs of the user. If you try to define a case for that more memory allocation is necessary, the program will usually abort with an error message. (In some rare cases an exception is not found – the execution may then terminate surprisingly or give wrong results.)

```
 ERROR: More than  50 cells defined
 increase parameter MGRmaxX in <PARAMETERS> section
```

If you receive such an error message, first check your input file for wrong parameters. If you do not know which option may have caused this error message, use the {**verbose yes**} command in the input file. In the **<SPECIES>**, **<MECHANISM>**, **<CHANNEL>**, and **<MONOLITH>** sections, you can redefine the parameters directly in the input file. For this, insert a **<PARAMETERS>** tag directly after the opening tag of one of these sections. (The **<PARAMETERS>** tag must be the first evaluatable option in these sections.) For instance:

```
<SPECIES>
  <PARAMTERS>
    DCSGMAX = 500   # maximum number of gas-phase species
    DCSMMAX = 50    # maximum number of collision efficiencies
  </PARAMETERS>
  <GASPHASE>
    ...             # list of species
  </GASPHASE>
</SPECIES>

<MECHANISM>
  <PARAMETERS>
     DCRGMAX = 2000 # maximum number of gas-phase reactions
     DCRGTMAX=  100 # maximum number of Troe reactions
     DCRGGMAX=   20 # maximum number of global gas-phase reactions
  </PARAMETERS>
  <GASPHASE>
    ...             # gas-phase mechanism
  </GASPHASE>
</MECHANISM>
...
```

In case dynamic memory allocation is not yet available, you have to change the parameter in the file named by the error message and recompile the application. If you change a parameter in on of the DETCHEM libraries, it is required to recompile the whole package by calling **make** in the DETCHEM root directory.

The following table gives an overview of the parameter files

module	filename	kind of parameters
lib_input	***constINP.F***	line and name length, include depth
	commINPlink.F	tag depth, number of options in one tag
lib_detchem	***parameterDC.f90***	number of species, reactions, and surface types (fully dynamic)
lib_washcoat	***parameterWC.f90***	number of washcoat grid points
CHANNEL	***parameterBL.f90***	number of radial grid points, number of axial sections (fully dynamic)
GRIDGEN3D	***commGRID.F***	number of points, cells, and layers
MONOLITH	***parameterM.f90***	grid size, number of materials and boundaries, output times, channel types (fully dynamic)

List of Scientific Publications Using DETCHEM

1. O. Deutschmann and L. D. Schmidt. Two-Dimensional Modeling of Partial Oxidation of Methane on Rhodium in a Short Contact Time Reactor. Proc. Comb. Inst. 27 (1998), 2283-2291

2. L. D. Schmidt, O. Deutschmann, and C. T. Goralski, Jr. Modeling the Partial Oxidation of Methane to Syngas at Millisecond Contact Times. Natural Gas Conversion V, Studies in Surface Science and Catalysis 119, p. 685-692, Elsevier, Amsterdam, 1998.

3. O. Deutschmann and L. D. Schmidt. Modeling the Partial Oxidation of Methane in a Short Contact Time Reactor. AIChE J. 44 (1998) 2465-2476.

4. O. Deutschmann, L. D. Schmidt, J. Warnatz. Simulation of Reactive Flow in a Partial Oxidation Reactor with Detailed Gas Phase and Surface Chemistry Models, In: Scientific Computing in Chemical Engineering II. Computational Fluid Dynamics, Reaction Engineering, and Molecular Properties. F. Keil, W. Mackens, H. Voß, J. Werther (Eds.). p. 368-375, Springer, 1999.

5. L. L. Raja, R. J. Kee, O. Deutschmann, J. Warnatz, L. D. Schmidt. A Critical Evaluation of Navier-Stokes, Boundary-Layer, and Plug-Flow Models for the Simulation of Flow and Chemistry in a Catalytic Combustion Honeycomb Channel. Catalysis Today 59 (2000) 47-60.

6. O. Deutschmann, L. Maier, U. Riedel, A. H. Stroemann, R. W. Dibble. Hydrogen Asssisted Catalytic Combustion of Methane on Platinum. Catalysis Today 59 (2000) 141-150.

7. J. Braun, T. Hauber, H. T'obben, P. Zacke, D. Chatterjee, O. Deutschmann, J. Warnatz. Influence of Physical and Chemical Parameters on the Conversion Rate of a Catalytic Converter: A Numerical Simulation Study. SAE paper 2000-01-0211 (2000)

8. D. K. Zerkle, M. D. Allendorf, M. Wolf, O. Deutschmann. Modeling of On-Line Catalyst Addition Effects in a Short Contact Time Reactor. Proc. Combust. Inst. 28 (2000) 1365- 1372.

9. D.K. Zerkle, M.D. Allendorf, M. Wolf, O. Deutschmann. Understanding Homogeneous and Heterogeneous Contributions to the Platinum-Catalyzed Partial Oxidation of Ethane in a Short Contact Time Reactor. J. Catal. 196 (2000) 18-39

10. O. Deutschmann, R. Schwiedernoch, L.I. Maier, D. Chatterjee. Natural Gas Conversion in Monolithic Catalysts: Interaction of Chemical Reactions and Transport Phenomena. Natural Gas Conversion VI, Studies in Surface Science and Catalysis 136, E. Iglesia, J.J. Spivey, T.H. Fleisch (eds.), p. 215-258, Elsevier, 2001

11. J. M. Redenius, L. D. Schmidt, O. Deutschmann. Millisecond Catalytic Wall Reactors: I. Radiant Burner. AIChE J. 47(2001) 1177-1184

12. S. Tischer, C. Correa, O. Deutschmann. Transient three-dimensional simulation of a catalytic combustion monolith using detailed models for heterogeneous and homogeneous reactions and transport phenomena. Catalysis Today 69 (2001) 57-62

13. D. Chatterjee, O. Deutschmann, J. Warnatz. Detailed surface reaction mechanism in a three-way catalyst. Faraday Discussions 119 (2001) 371-384

14. J. Braun, T. Hauber, H. T'obben, J. Windmann, P. Zacke, D. Chatterjee, C. Correa, O. Deutschmann, L. Maier, S. Tischer, J. Warnatz. Three-Dimensional Simulation of the Transient Behavior of a Three-Way Catalytic Converter. SAE Technical paper 2002-01- 0065 (2002)

15. R. P. O Connor, L.D. Schmidt, O. Deutschmann. Simulating the Millisecond Oxidation of Cyclohexane by Coupled Chemistry and Fluid Dynamics. AIChE J. 48 (2002), 1241-1256

16. R. Schwiedernoch, S. Tischer, C. Correa, O. Deutschmann, J. Warnatz. Experimental and numerical investigation of the ignition of methane combustion in a platinum-coated honeycomb monolith. Proc. Combust. Inst. 29 (2002)

17. K. Maruta, K. Takeda, J. Ahn, K. Borer, L. Sitzki, P. D. Ronney, O. Deutschmann. Extinction Limits of Catalytic Combustion in Microchannels. Proc. Combust. Inst. 29 (2002).

18. R. Schwiedernoch, S. Tischer, C. Correa, O. Deutschmann. Experimantal and Numerical Study of the Transient Behaviour of a Catalytic Partial Oxidation Monolith. Chem. Eng. Sci. 58 (2003), 633-642.

19. J. Windmann, J. Braun, P. Zacke, S. Tischer, D. Chatterjee, O. Deutschmann, J. Warnatz. Impact of the Inlet Flow Distribution on the Light-Off Behaviour of a 3-Way Catalytic Converter. SAE Technical Paper 2003-01-0937 (2003).

20. R. Schwiedernoch, S. Tischer, H.-R. Volpp, O. Deutschmann. Towards a better understanding of transient processes in catalytic oxidation reactors. Natural Gas Conversion VI, Studies in Surface Science and Catalysis 147 (2004), 511-516.

21. M. Bizzi, R. Schwiedernoch, O. Deutschmann, G. Saracco. Modeling the Partial Oxidation of Methane in a Fixed Bed with Detailed Chemistry. AIChE J. 50 (2004), 1289 - 1299.

22. S. Tischer, O. Deutschmann. Recent advances in numerical modeling of catalytic monolith reactors. Catalysis Today (2005) in press.

23. V. M. Janardhanan, O. Deutschmann. CFD analysis of a solid oxide fuel cell with internal reforming: Coupled interactions of transport, heterogeneous catalysis and electrochemical processes. J. Power. Sources. 162 (2006), 1192-1202.

24. V. M. Janardhanan, O. Deutschmann. Numerical study of mass and heat transport in solid oxide fuel cells running on humidified methane. Chem. Eng. Sci. (in press)

25. V. M. Janardhanan, V. Heuveline, O. Deutschmann. Performance evaluation of planar solid oxide fuel cells. J. Power. Sources. (accepted).