第 I 部
各機能の説明

1　ライブラリ生成アルゴリズムについて

本プログラムは基本的に「ライブラリ生成プログラム」となる。よって、各種パラメータ設定ファイルに加えて、流体計算の場合は”condition.f90”というファイルで直接初期条件と境界条件を設定しなければならない。これらのファイルは”raw”というサフィックス付きでディレクトリにコピーされる。

GUI インターフェースである GUI.py は、checkout.py を動かすための input ファイル (checkout.inp, checkout_model.inp, checkout_chem.inp) を生成し、checkout.py を実行するプログラムであり、主動作は GUI.py を用いた場合でも全て checkout.py が受け持つ。

checkout.py は input ファイルに基づき、各種ライブラリを生成するプログラムである。基本的にライブラリの生成は、該当ファイルをディレクトリからコピーすることによって行い、ある特定のファイルを除いては fortran ソースファイル自体を checkout.py が編集することはない。コピーされるファイルは全て”store”のサブディレクトリ内に入っており、checkout.py またはユーザーによって編集が加えられるファイルには”.raw.”というサフィックスがつけられている。なお、checkout.py 内でそのソースファイルの編集が終了する、つまりそれ以上ユーザーによる編集が不必要になったファイルからは自動的に”.raw.”サフィックスが取り除かれ、そのままコンパイルにかけられるようになる。

1.1　ディレクトリ構造

基本的に類似する機能を持つファイルがまとめられている。例えばオイラー陽解法と LU-SGS 法は”time_scheme”ディレクトリ内に”euler”,”lu-sgs”としてそれぞれ格納されている。また化学計算と流体計算は基本的に同一ライブラリを用いることになっている。以下、全てのディレクトリについて説明を行う。

2　流体コード

checkout.py から呼び出された”store/checkout/checkout_flow.py”によって生成される。生成されたファイルは”checkout”ディレクトリ内に生成される。

2.1　自動生成ファイル

流体コードで自動生成されるファイルは、checkout.py で選択されたディレクトリ内にある”部分的なファイル”をつなげることで生成される。自動生成されるファイル及びその”部分的なファイル”は以下のとおりである。

- main.f90... 主プログラムがかかれたソースとなる。主ループもここで回される。以下の順にファイルの内容が追記されていく。
  - main.top.f90...”program main”からモジュールのインポートまで。
  - main.head.f90... 共通して利用する変数の定義。
  - main.variable.f90... それぞれのスキームで特有に利用される変数の定義。

  - main.part_init.f90... 変数の初期化。グリッドの読み込みやそのバイナリファイルの生成や幾何ヤコビアンの生成、熱力学ライブラリが必要とされる場合にはその初期化も行われる。
  - main.body.raw.f90... 初期化。restart.bin がある場合はそれを用い、ない場合は初期化サブルーチンを用いて初期化を行う。また主ループを回す変数も初期化される。part_primitive という文字列がファイル内に記されており、後述する文字列に置き換えられる。主ループの開始まで記述してある。
  - main.main.raw.f90, main.main.point_implicit.f90... 主ループ内部を記述しており、時間スキームのディレクトリ内に格納されている。part_point_implicit, part_primitive, part_main という文字列がファイル内に記されており、それぞれ以下の文字列に置き換えられる part_point_implicit に文字列が存在する場合 main.main.point_implicit.f90、存在しない場合 main.main.raw.f90 が使われる。なお、point implicit なスキームを受け付けない時間積分法 (ex.Dual Time Step) には main.main.point_implicit.f90 は存在せず、無理やり使おうとする場合コンパイルエラーが起きる。
    * part_point_implicit...point implicit に計算される部分がここに入る。現状、化学生成項の時間積分を進めるサブルーチンがここに入りうる。main.part_point_implicit.f90 にかかれた文字列に置き換えられる。
    * main.part_primitive.f90... 時間刻みの計算やデータアウトプット前にすべき処理、具体的には基本量の計算及び境界条件の設定をするサブルーチンがここに入る。main.part_primitive.f90 にかかれた文字列に置き換えられる。
    * main.part_main.f90... 高次精度化や対流項及び拡散項の計算、時間積分に必要な場合にはそれらの保存量ヤコビアンが計算される。main.part_main.f90 にかかれた文字列に置き換えられる。
  - main.foot.f90... 主ループの終了及び MPI 関数の終了処理、”end program main”を記述してある。
- Makefile...Makefile. 以下の順にファイルの内容が追記されていく。
  - Makefile.head... コンパイラに何を使うのか定義する。アーキテクチャ (PC かスーパーコンピュータ) によって変更される。
  - Makefile.var ... 各種変数の定義
  - Makefile.main... 各オブジェクトファイル作成ルール
  - Makefile.foot... プログラムの最終的なコンパイル部分及び clean の定義
- control.raw.inp...cfl の定義やファイル出力間隔の定義など。使用する場合、MUSCL のパラメータもここで入力される。control.part.inp に記された内容が入力される。

2.2 store/core

どのような場合にも使用されるファイルがまとめられてある。上記自動生成ファイルのための各種ファイルの他、以下のファイルが存在する。

- check_convergence.f90... 収束判定及び途中経過ファイル,restart ファイルの生成を行っている。
- init.vi...vim 用初期化ファイル。これを読み込むと、RUN コマンドで./main を走らせることができる。
- inout.f90...restart ファイル読み書きや途中経過ファイル書き込み。
- n_grid.raw.f90... 重要なパラメータの定義。以下の文字列は checkout.py で置き換えられる。

- NPLANE...multi-block で block の数。muti-block でない場合 1.
- NumI... それぞれの面での i 方向セル数
- NumJ... それぞれの面での j 方向セル数
- NIMAX...i 方向セル数最大値
- NJMAX...j 方向セル数最大値
- NumY...rho の数.ex) 理想気体:1, flame sheet, 完全平衡モデル:2, 素反応モデル:化学種数
- NV... 拡散で空間微分をとるべき次元数.flame sheet で 3, 完全平衡モデルで化学種数.
- GridFileName...plot3d で記述されたグリッドファイルの場所.

また以下の文字列は IC や BC の設定に使える。

- dimq ... q(後述) の次元
- dimw ... w(後述) の次元
- indxg ... w に於ける比熱比のインデックス
- indxht ... w に於ける質量あたり全エンタルピー (J/kg) のインデックス
- indxR ... w に於ける質量あたり気体定数 (J/kg/K) のインデックス
- indxMu ... w に於ける粘性係数 (Pa*s) のインデックス

- prmtr.f90...pi やモルあたりの気体定数"R_uni"の定義.
- read_control.f90...control.inp の読み込み. 全計算に共通する部分に限る。
- scheme.f90... 流束項評価サブルーチン。現在は SLAU のみ。
- variable.f90... 一般的に使われる変数がまとめられてある。
  - q ... 保存量.
    - インデックス 1 から nY：それぞれの気体グループ (理想気体なら全化学種、flame sheet か完全平衡モデルなら燃料と酸化剤、素反応モデルならそれぞれの化学種) の密度 (kg/m$\hat{3}$)
    - インデックス nY+1：x 方向運動量 (kg*m/s)
    - インデックス nY+2：y 方向運動量 (kg*m/s)
    - インデックス nY+3：全エネルギー (J/m$\hat{3}$)
  - qp ... 前ステップの q
  - qpp ... 全ステップの qp
  - w ... 基本量
    - インデックス 1：全化学種の密度の和 (kg/m$\hat{3}$).
    - インデックス 2：x 方向速度 (m/s).
    - インデックス 3：y 方向速度 (m/s).
    - インデックス 4：圧力 (Pa)
    - インデックス 5 ... nY+4：それぞれの気体グループの質量分率
    - インデックス nY+5(=indxg @n_grid.f90)：比熱比
    - インデックス nY+6(=indxht@n_grid.f90)：全エンタルピー (J/kg)
    - インデックス nY+7(=indxR @n_grid.f90)：気体定数 (J/kg/K)
    - インデックス nY+8(=indxMu@n_grid.f90)：粘性係数 (Pa*s)
  - vhi ... それぞれの気体グループの生成熱を含む内部エンタルピー (J/kg)
  - wHli,wHri, wHlj, wHrj ... 高次精度化で予測された w(ex. MUSCL).'l' は'left'、'r' は'right'、'i' は i 方向に +1/2 を足した位置、'j' は j 方向に 1/2 を足した位置。例えば、wHli(:,2,3) は $w_{2+\frac{1}{2},3}$

の左側の値を表す。
- TGi, TGj ... それぞれ i 方向 j 方向の流束項 TG. TG の定義については嶋田テキストの Section 9 参照. これらの値も 1/2 だけセル中心からずれた場所の値であり、ずれ方は wH と同じ。
- TGvi, TGvj ... 粘性項.
- Vol ... 軸対称問題ではそれぞれのセルの体積、二次元問題では面積。
- dsi, dsj ... i 方向 j 方向にそれぞれ垂直なセル辺の長さ。軸対称問題ではさらに半径がかけられる。これもセル中心から 1/2 だけずれた場所の値。
- vni, vnj ... それぞれ dsi, dsj で示された辺の直交正規ベクトル (絶対値 1). 向きは i,j 正の向き。これもセル中心から 1/2 だけずれた場所の値。
- xh, rh ... メッシュ格子点の座標. これもセル中心から 1/2 だけずれた場所の値。
- x, r ... セル中心の座標.
- dt_mat ... local time step のそれぞれのセルでの Δt
- dt_grbl ... global time step の Δt.(スペルミスったので'r' です。)

また全ての計算において、以下の値を control.inp で設定する必要がある。

- Max Step Number... 計算を再開してからの最大ステップ数.
- convergent RMS... 収束とするエネルギー残渣.
- CFL Number...CourantFriedrichsLewy 数.
- File Output Period... ファイル出力周期.

2.3 store/architecture

アーキテクチャ依存のサブルーチン. サブディレクトリは PC と SC でそれぞれ PC とスーパーコンピュータに関するファイルがある. SC には mod_mpi.f90、PC には mod_mpi_dummy.f90 が使われる。スーパーコンピュータの場合は grid_separation.inp からそれぞれのプロセッサが受け持つグリッドの範囲を計算し、変数を初期化する. PC の場合は MPI 関数のダミー関数を入れている。どちらの場合も以下のように受け持つブロックの範囲,i 方向の範囲,j 方向の範囲が定義される。

- ブロックの範囲...nps から npe
- ブロック plane の i 方向の範囲...nxs(plane) から nxe(plane)
- ブロック plane の j 方向の範囲...nys(plane) から nye(plane)

パソコンの場合、nps=1, npe=Nplane である。

2.4 store/dim

軸対称流か二次元流かを選ぶものである。サブディレクトリは 2d と q2d で、それぞれ以下のファイルが含まれている。

- store/dim/2d...geometry.f90 と init_Sq.f90
- store/dim/q2d...geometry.f90 と Sq.f90

Sq は軸対称流のとき式変形の際ソース項に出てくる値を定義しており、二次元流では 0 である。init_Sq.f90 と Sq.f90 はそれぞれ Sq に関する処理を行っている。geometry.f90 には plot3d で記述されたグリッドファイルの読み込み及び可視化用バイナリファイルの生成、幾何ヤコビアンやセルの面積を計算するルーチンがまとめられている。

2.5 store/high_order

高次精度化を行うルーチンである。サブディレクトリは muscl と upwind であり、それぞれ MUSCL スキームと風上差分が入っている。MUSCL を用いる場合、以下の項目を control.inp で設定する必要がある。

- MUSCL ON(1)/OFF(0)...MUSCL の on/off 切り替え。1 で on, 0 で off. その他の値は受け付けない。
- MUSCL Precision Order...MUSCL のオーダー調整。2 と 3 のみ受けつけ、2 の場合は $\kappa = -1$, 3 の場合は $\kappa = \frac{1}{3}$ に MUSCL のパラメータが設定される。

2.6 store/time_step

global time step または local time step が設定される。ファイルは set_dt.f90。またこのどちらかを選ぶことにより、checkout.py の DT_GLOBAL_LOCAL が dt_mat(i,j,plane) または dt_grbl が設定され、時間積分スキームの該当する部分で置き換えられる。

2.7 store/time_scheme

時間スキームの選択。main.main.raw.f90 や main.main.point_implicit.f90, 時間積分サブルーチン (time_...f90) が入っており、主ループ (時間積分ループ) の中身を定義する。

- euler... オイラー陽解法.
- RK2... ルンゲ=クッタ 2 次精度.
- LU-SGS...LU-SGS で実装したオイラー陰解法. 上記の他 sch_lusgs.f90, var_lusgs.f90 で LU-SGS 法を行う。
- NR...Newton-Raphson 法. 同様に LU-SGS 法を使う. 後述するパラメータを用いる. 上記の他 sch_NR.f90, var_NR.f90 で LU-SGS 法を行う。
- precon... 前処理法を用いた Newton-Raphson 法. 同様に LU-SGS 法を使う. 上記の他 sch_precon.f90, var_precon.f90 で LU-SGS 法を行う。仮時間ステップにも前処理行列をかけている. 後述するパラメータを用いる.
- dual...Dual Time Step. 同様に LU-SGS 法を使う. 後述するパラメータを用いる. 上記の他 sch_dual.f90, var_dual.f90 で LU-SGS 法を行う。
- preconLU-SGS... 前処理法をオイラー陰解法 (LU-SGS) で実装している。上記の他 sch_precon.f90, var_precon.f90 で LU-SGS 法を行う。

なお、LU-SGS から preconLU-SGS には”2d”と”q2d”というディレクトリがそれぞれあるが、それぞれ二次元流、軸対称流のコードが収められている。

■NR,precon,dual **で用いられるパラメータについて**　これらのスキームでは以下のパラメータを control.inp で定義する。ただし、ω は内部反復の緩和係数である。

- InternalLoop CFL tau... 疑似時間ステップの cfl 数
- InternalLoop Omega Max... 最大緩和係数。以下 $\omega_{\max}$ とする。
- InternalLoop Omega Min... 最小緩和係数. これを割り込む ω が設定されると、計算を中止する。
- InternalLoop Dqrate Max... 各気体グループの密度の内部反復あたり変化割合の最大値. 以下 $Dq_{\max}$ とする。
- InternalLoop ResRateWarn... 内部反復が収束していないことを標準出力に投げる最小エネルギー残渣比
- InternalLoop ResRateErr... 計算を中止する最小エネルギー残渣比
- InternalLoop OutOmega... ファイル internal_res.dat に緩和係数 ω の履歴を出すか。
- InternalLoop Max Number... 内部ループ数 (固定)

これらのスキームの内部ループで、保存量 q は緩和係数 ω をかけられた Δq により更新される。ω は以下のように定義される。

$$\omega = \min(\omega_{\max}, \frac{Dq_{\max}\rho_i}{\Delta\rho_i}) \tag{1}$$

ただし $i = 1...$nY である。これにより、ρ_i の一内部反復での変化は $Dq_{\max}$ 以下に抑えられる。この定義であると、ω が小さくなりすぎることがある。それを補足するのが”最小緩和係数”のパラメータである。

内部ループの発散は’InternalLoop Omega Min’, ’InternalLoop ResRateWarn’, ’InternalLoop ResRateErr’ で補足されることとなる。ここでエネルギー残渣比は (最後の内部反復でのエネルギー残渣)/(最初の内部反復でのエネルギー残渣) と定義している。なお、NR, precon, dual の時間スキームでは、基本コンセプトは”内部反復が収束”することに基づいているので、エネルギー残渣比が十分小さくならない場合はコンセプト自体が崩壊している。よって、その場合これらのスキームは使うべきではない。(これを見落としている研究は大変多く、私は悲しい。)

パラメータ調整方法　パラメータ調整には”internal_res.dat”が使える。これは”InternalLoop OutOmega”を”true”にセットすることで得られる。(このファイル出力は時間がかかるので、本計算のときには効率向上のため”false”にすべき。) このファイルには 1 列目に内部反復数、2 列目に Dqrate により計算された ω($\omega_{\max}$ で修正する前)、3 列目にエネルギー残渣比が記録されている。以下にこれらの出力を使ってパラメータを調節する手段の一つを記す。

- **’InternalLoop Max Number’ を増やす**... 内部反復数が小さいことが原因で、エネルギー残渣比が指数的に減少しているのに’InternalLoop ResRateErr’ に達していないとき。
- **’InternalLoop Max Number’ を減らす**...’InternalLoop Max Number’ に比べ遥かに少ない内部反復数で’InternalLoop ResRateErr’ に達しているとき。
- **’InternalLoop Omega Max’ を増やす**... 下記の目安に比べ内部反復の収束が遥かに遅く、$\omega_{\max}$ により ω が主に決まっているとき。
- **’InternalLoop Omega Max’ を減らす**...Dqrate により決定された ω(’internal_res.dat’ の第二列目に記録された ω) が指数的に増加せず、一回目の内部反復の ω が $\omega_{\max}$ により決定されている場合。この状況は $\omega\Delta q$ があまりにも大きすぎて、一回目の内部反復における q の推測値が悪くなりすぎ、その

ためその後のステップで正しい値に修正できなくなった場合に起こる。

- **'InternalLoop Dqrate Max' を増やす**... 内部反復の収束が遅く、ω が Dqrate によって主に決定されている場合。
- **'InternalLoop Dqrate Max' を減らす**... ω が指数的に増加せず、一回目の内部反復で Dqrate により ω が決定されている場合。この状況も $\omega\Delta q$ があまりにも大きすぎて、一回目の内部反復における q の推測値が悪くなりすぎ、そのためその後のステップで正しい値に修正できなくなった場合に起こる。

これらの手段は理論的なものではなく、私の経験によるものである。よって間違っている可能性もある。他の方法も試してください。

パラメータの目安

- **0.1 @ Omega Max**... もしこれで動く場合、0.8 や 0.9 も可能な場合もある。Dual Time Step では 0.01 を使わざるを得なかった場合もある。
- **0.001 @ Omega Min**... この値を下回り正しい解を出した計算に出会ったことはない。このパラメータは固定すべきであると思う。
- **0.01 @ Dqrate Max**... 現実問題このパラメータは多分子流にのみ使われうる。Dual Time Step では 0.001 を使わざるを得なかった場合もある。経験上 0.1 は大きすぎる。
- **1.e-5 @ ResRateWarn, 1.e-3 @ ResRateErr**... 経験上、エネルギー残渣比は 3 オーダーは下げなければ正しい解を出さない。よって 1e-3 は固定すべきだと考える。ResRateWarn は警告メッセージを出すだけであるので、好みの値を用いてよい。
- **60 @ Max Number**... 理想気体では経験上 20 で十分。多分子流を Dual Time Step で解いたとき、100 必要だった場合もあった。

2.8 store/viscosity

non-viscous と viscous というディレクトリがあり、viscous にはその中に with-nV と no-nV、さらにそれぞれに"2d"と"q2d"というディレクトリが収められている。non-viscous が非粘性流、viscous が粘性流である。2d と q2d の違いは time_scheme と同じ。

with-nV と no-nV について、化学モデルには"単純に ρ_i の空間微分でエンタルピー拡散を扱えるもの"とそうでないものがあり、それぞれが with-nV と no-nV に対応している。with-nV の場合、"Yv"という"質量拡散によるエンタルピー拡散を考える際考慮しなければならない気体グループの質量分率"が熱力学ライブラリから生成される。例えば、一段総括反応の場合、化学組成は"反応物としての"燃料と酸化剤の質量のみを追いかければいいが、質量拡散によるエンタルピー拡散を計算する場合"生成物としての"燃料、酸化剤、生成物それぞれの質量分率を計算せねばならず、with-nV が使用される。

ファイルは sch_viscous.f90 である。プラントル数 Pr とシュミット数 Sc はここで名前付き定数として共に 1 に定義してあるので、条件を変えたい場合はここを変更するとよい。

2.9 store/therm_lib

therm_lib 内には必ず thermal_mode.f90 が入っており、set_thermo_prop というサブルーチンにより、前ステップに計算された

圧力、気体定数、保存量

から

温度の推定値、質量あたり内部エネルギー、各気体グループの質量分率

を計算し、そこから現ステップでの

圧力、比熱比、質量あたり全エンタルピー、粘性係数、気体定数、各化学種のエンタルピー vhi、各化学種の修正生成熱 DHi

を計算する。また、viscosity の節でも述べたとおり、一部熱力学ライブラリではエンタルピー拡散の計算のために Yv も計算する。(圧縮性流体ではエネルギー保存であるためエンタルピーは圧力依存となる。圧力は温度で変わり、温度は熱力学ライブラリで変わるため、全エンタルピーも基本量としている。) これら出力結果は基本量 w の配列に格納される。

化学ライブラリには大きくわけて理想気体,NASA データベースを用いるもの,chemkin ファイルを用いるものにわけられ、それぞれディレクトリ ideal,NASA,chemkin に対応する。NASA データベースを用いる計算は凍結流、一段総括反応、完全平衡モデルがつかえ、chemkin ファイルを用いる計算は凍結流、素反応モデルが使える。

■vhi と DHi の定義　それぞれのライブラリ詳細の説明を行う前に、vhi と DHi の説明をする。

vhi はエンタルピー拡散の計算に使われ、次元は nV(Yv の次元と同じ.Yv を使わない場合は nY) である。それぞれの気体グループの生成エネルギーを含む単位質量あたりの内部エンタルピーである。拡散の計算に使われるため、vhi に関しては境界での値が必要となる。よって、適切な境界条件を設定しなければならない。

DHi は修正生成熱であり、陰解法や前処理法に用いる流束の保存量でのヤコビアンを求めると出てくる。定義としては以下のとおり。

$$DHi_i = \kappa R_i T - (\kappa - 1) vhi_i \tag{2}$$

ただし、κ は混合ガス全体の平均比熱比、R_i はその気体グループの質量あたりの気体定数 (=(一般気体定数)/(平均分子量)).

DHi_i は理想気体では 0 であるが、実在気体では有限の値を持つ。また、陰解法を許さない熱力学モデルでは、これは 0 にセットされている。

2.9.1 store/therm_lib/ideal

特有モジュール	nY	nV	with-nV or no-nV
gas	1	1	no-nV

理想気体。ファイルは thermal_model.f90 のみである。粘性係数は動粘性係数一定の場合にのみ現在対応している。質量あたり気体定数 R_gas、比熱比 kappa_gas、動粘性係数 nu_gas は”module gas”をインポートすれば使用可能であり、checkout.py でそれぞれ 1.4,287,1.6e-5 に定義されるが、thermal_model.f90 の該当部分を直接編集すれば任意の値に変更可能である。境界条件設定時にはぜひ”module gas”をインポートして上記の名前付き定数を使っていただきたい。

保存量、基本量や vhi は以下のように表される。ただし、ρ, u, v, p は密度、x,y 方向速度、圧力である。

```
q(1)=rho
```

```
q(2)=rho*u
q(3)=rho*v
q(4)=p/(kappa_gas-1d0)

w(1)=rho
w(2)=u
w(3)=v
w(4)=p
w(5)=1d0
w(indxg )=kappa_gas
w(indxht)=kappa_gas/(kappa_gas-1d0)*p/rho+0.5d0*(u**2+v**2)
w(indxR )=R_gas
w(indxMu)=rho*nu_gas

vhi(1)=kappa_gas/(kappa_gas-1d0)*p/rho
DHi(1)=0d0
```

2.9.2 store/therm_lib/NASA

特有モジュール	nY	nV	with-nV or no-nV
const_chem, chem, chem_var	2	モデルによる	with-nV

NASA-CEA 用に公開されたデータベースを用いた熱力学モデル集。凍結流と一段総括反応 (flame sheet)、完全平衡モデルを用意している。

気体グループは 2 つであり、ライブラリでは一つ目に燃料、二つ目に酸化剤を当てている。

全てのモデルで使用される熱力学モデルは core 内に格納されており、以下のファイルがある。

- LU.f90...LU 分解で線形連立方程式を解くモジュール. 完全平衡モデルで利用されている。
- thermo.inp...NASA-CEA 内に格納されている熱力学データ.
- trans.inp...NASA-CEA 内に格納されている熱輸送データ.
- func_chem.f90... 熱力学関数多項式の係数を返す関数ライブラリ.
- mod_chem.raw.f90...NASA データベースを用いた計算に必要なモジュール.(上述特有モジュール) 変数後述.
- sub_chem.f90...mod_chem 内の変数初期化や input ファイル読み込み、上述反応モデルの全てのコア部のサブルーチン.

■mod_chem.f90 の変数について

const_chem　定数と重要な値 (化学種数) を格納している.

- ne = NE !the number of elements
- max_ns = NS !the maximum number of species
- Ru = 8.3144621d0 !universal gas constant

- pst = 1d5 !standard state pressure
- omega = 0.5d0 !relaxation factor
- eps = 1d-9 !epsilon for convergence
- initial_eps= 1d-5 !epsilon to stabilize calculation
- Y_eps = 1d-6 !epsilon for reduction determination
- TSIZE = 1d-11 !epsilon for calc E or H
- TTSIZE = 1d-14 !epsilon for calc n
- ns
- nt

test

```
module chem
   use const\_chem
   implicit none
   integer         ,dimension(max\_ns)::num\_sctn
   double precision,dimension(max\_ns)::MW
   double precision,dimension(2,6,max\_ns)::Trange
   double precision,dimension(9,6,max\_ns)::co

   character*2 ,dimension(ne)::SYM\_ELM
   character*18,dimension(max\_ns)::SYM\_SPC

   !for trans
   double precision trans(4,3,max\_ns),Trange\_trans(2,3,max\_ns)
   character*18     species\_name\_trans(max\_ns)
   integer         ,dimension(max\_ns)::num\_sctn\_trans
   integer         ,dimension(max\_ns)::tr2th

   double precision,dimension(ne,max\_ns)::Ac
end module chem
```

test

```
module chem\_var
   use grbl\_prmtr
   use const\_chem
   implicit none
   double precision n\_save(max\_ns,nimax,njmax,Nplane)

   double precision qf(dimq),wf(dimw)
   double precision rhof,pf,Tf,Ef,Hf,MWf,kappaf,muf
```

```
   double precision Yvf(nV),vhif(nV)
   double precision b0f(ne+2)
   double precision nf(max\_ns)
   double precision nfini(max\_ns)
   integer          nef
   integer          elistf(ne+2)
   integer          nelistf(ne+2)
   double precision maskf(max\_ns)
   double precision maskbf(ne+2)

   double precision qo(dimq),wo(dimw)
   double precision rhoo,po,To,Eo,Ho,MWo,kappao,muo
   double precision Yvo(nV),vhio(nV)
   double precision b0o(ne+2)
   double precision no(max\_ns)
   double precision noini(max\_ns)
   integer          neo
   integer          elisto(ne+2)
   integer          nelisto(ne+2)
   double precision masko(max\_ns)
   double precision maskbo(ne+2)

   !for flame sheet model
   double precision np(max\_ns)
   double precision of
end module chem\_var
```

最後に、境界条件での入力に関して。例えば酸化剤のみのセルで条件を入力する場合、保存量、基本量や vhi は以下のように表される。ただし、u, v は密度、x,y 方向速度、圧力である。DHi については set_thermo_prop で直接入力されるため、手作業で入力されることはない。

```
q(1)=rhoo
q(2)=rhoo*u
q(3)=rhoo*v
q(4)=rhoo*Eo

w(1)=rhoo
w(2)=u
w(3)=v
w(4)=po
w(5)=0d0
```

```
w(6)=1d0
w(indxg )=kappao
w(indxht)=Ho+0.5d0*(u**2+v**2)
w(indxR )=R_uni/MWo
w(indxMu)=muo

vhi=vhio
```

燃料では `w(5)=1d0;w(6)=0d0` になる。

2.10 store/cond

core no-nV with-nV

3 NASA 熱力学データベース自動生成

4 化学コード