TECHNICAL REPORT
IGE–314

A USER GUIDE FOR OPTEX VERSION4

R. Chambon

Institut de génie nucléaire
Département de génie mécanique
École Polytechnique de Montréal
July 4, 2015

Contents

List of Tables

Chapter 1

OPTEX MODULES

1 Fuel Management Optimization

In this section, modules used for fuel management optimization will be described.

1.1 The `FOBJCT:` module

The `FOBJCT:` module is used to define the different parameters for an optimization calculation. These parameters can be decision variables, contraint zone definitions, constraint limits, ... This module can also evaluate the objective function and the constraints values.

The calling specifications are:

Table 1.1: Structure `FOBJCT:`

OPTIM `:=` `FOBJCT:` [*OPTIM*] [*MAPFL*] [*FLUX* [*FLUXP*]] [*MACRO*] [*TRACK INDEX*] `::` **(descfobjct)**

where

OPTIM `character*12` name of the extended OPTIMIZE. If *OPTIM* appears on the RHS, the information previously stored in *OPTIM* is modified if necessary and stored.

MAPFL `character*12` name of the extended MAP. If *MAPFL* appears on the RHS, the information in it will be read for many parameters initialisation.

FLUX `character*12` name of the FLUX linked list. This object is used for the function evaluation. It is recommended to provide it even if no function evaluation is done for many parameter reading.

FLUXP `character*12` name of the FLUX linked list. This object is used for some function evaluation such as void reactivity.

MACRO `character*12` name of the MACROLIB linked list file containing fuel regions description and burnup informations. If it appears on RHS, it means it will be necessary for a function evaluation (objective or constraint).

TRACK `character*12` name of the TRACKING linked list file containing the tracking informations. If it appears on RHS, it means it will be necessary for a function evaluation (objective or constraint) or to memorize the average exit burnup or the fuel cost distribution.

INDEX `character*12` name of the INDEX linked list file containing the index informations. If it appears on RHS, it means it will be necessary for a function evaluation (objective or constraint) or to memorize the average exit burnup or the fuel cost distribution.

(descfobjct) structure containing the data to module `FOBJCT:`.

1.1.1 Data input for module `FOBJCT`*:*

Table 1.2: Structure **(descfobjct)**

[`EDIT` *iprint*]
[`CTRL-ZONE-DF` **(czdf_data)**]
[`FUEL-COST-DF` **(fcdf_data)**]
[`EXIT-B-DIST` `MEMORY`]
[`CST-ZONE-DF` **(cstzdf_data)**]
[`EVAL-OBJ-CST` **(eval_data)**]
;

where

`EDIT` key word used to set *iprint*.

iprint index used to control the printing in module.

`CTRL-ZONE-DF` key word used to define the decision variables and their zones of influence.

(czdf_data) structure containing the data to the option `CTRL-ZONE-DF`.

`FUEL-COST-DF` key word used to define the cost of the uranium is the core.

(fcdf_data) structure containing the data to the option `FUEL-COST-DF`.

`EXIT-B-DIST` key word used to specify that the distribution of the average exit burnup for each volume will be pre-calculated.

`MEMORY` key word used to specify that the distribution of the average exit burnup will be stored in the *OPTIM* object.

`CST-ZONE-DF` key word used to define the constraint (type, value, zone of influence).

(cstzdf_data) structure containing the data to the option `CST-ZONE-DF`.

`EVAL-OBJ-CST` key word used to define and evaluate the objective and / or constraints functions.

(eval_data) structure containing the data to the option `EVAL-OBJ-CST`. This will be treated as a section in itself because other module will refer to it.

Table 1.3: Structure **(czdf_data)**

[`BURNUP-ZONE` $burn_{min}$ $burn_{max}$ { `ALL` | { { Y | N }$_i$,i=1,nbz } }]
[`ENRICH-ZONE` $enri_{min}$ $enri_{max}$ { { Y | N }$_i$,i=1,nez }]

where

`BURNUP-ZONE` key word used to specify that exit-burnup decision variables will be set. This exit-burnup zone were defined previously and are stored in the MAP.

$burn_{min}$ minimum value of the exit-burnup.

$burn_{max}$ maximum value of the exit-burnup.

`ALL` key word used to specify that all burnup-exit zone will be a decision variable.

`Y` key word used to specify that a specific burnup-exit zone will be a decision variable.

`N` key word used to specify that a specific burnup-exit zone will not be a decision variable.

nbz number of average exit burnup zone.

`ENRICH-ZONE` key word used to specify that exit-burnup decision variables will be set. This exit-burnup zone were defined previously and are stored in the MAP.

$enri_{min}$ minimum value of the exit-burnup.

$enri_{max}$ maximum value of the exit-burnup.

`Y` key word used to specify that a specific enrichment zone will be a decision variable.

`N` key word used to specify that a specific enrichment zone will not be a decision variable.

nez number of enrichment zone.

Table 1.4: Structure **(fcdf_data)**

[`FIXED` $cost_i$,i=1,*nez*)
| `DEPENDANT` ε_w C_{NU} C_S C_{FAB} *interest* t_{obt} t_{enr}]
[`MEMORY`]

where

`FIXED` key word used to specify that the price of the fuel is fixed for each enrichment zone.

cost cost of the fuel.

nez number of enrichment zone.

`DEPENDANT` key word used to specify that the price of the fuel is dependant of the enrichment for each enrichment zone.

ε_w U^{235} concentration of waste uranium after the separation work.

C_{NU} natural uranium cost ($/kg).

C_S cost of a separation work unit ($/SWU).

C_{FAB} cost of fabrication of the bundles ($/kg).

interest interest rate (y^{-1}).

t_{obt} time to obtain uranium (y).

t_{enr} time for enrichment (y).

MEMORY key word used to specify that the distribution of the purchase cost of uranium actually in the reactor will be pre-calculated and stored in the *OPTIM* object.

Table 1.5: Structure **(cstzdf_data)**

[KEFF k_{eff}]

[MAXPOWER

...[[ZONE-DEF [[SURV-ZONE n_{surv_zone} [[PLAN i_{plan} { i_{zonej},j=1,n_{cha} | SAME j_{plan} }]]

..................| BUNDLE { ALL | [[PLAN i_{plan} { { 0 | 1 }$_j$,j=1,n_{cha} | SAME j_{plan} }]]

..................| CHANNEL { ALL | { 0 | 1 }$_j$,j=1,n_{cha} }]]

...| VALUE-DEF {{ i_{zone1} cst_{lim} | RANGE i_{zone1} i_{zone2} { cst_{limj},j=i_{zone1},i_{zone2} | ALLSAME cst_{lim} } }]]

...END-MAX-POW]

[VOID-REACT-FC $\rho_{V,FC}$]

[ANALYTIC-FCT cst_{type} cst_{lim}]

where

KEFF key word used to defined k_{eff}.

k_{eff} neutron multiplication factor aimed (this is a constraint of type equal).

MAXPOWER key word used to defined a maximum power in a zone (this is a constraint of type inferior).

ZONE-DEF key word used to specify that the definition of the zone will be provided.

SURV-ZONE key word used to specified that the zone will be defined manually.

n_{surv_zone} total number of surveillance zone.

i_{zone} number of the zone that the bundle is part of (0 if no surveillance zone for this bundle).

BUNDLE key word to specified that surveillance zone are bundles.

CHANNEL key word to specified that surveillance zone are channels.

PLAN key word to specify that the definition of surveillance zone for i_{plan} will follow.

i_{plan} numbers of the plan to be defined.

SAME key word used to specify that the definition of surveillance zone in the plan i_{plan} will be the same one as in the plan j_{plan}.

j_{plan} number of the plan already defined.

ALL key word used to specify that the power in all bundles or channels will be a constraint.

n_{cha} number of channels.

VALUE-DEF key word used to specify that the limit of maximum zone power will be provided.

i_{zone1} first number of surveillance zone.

cst_{lim} constraint limit.

RANGE key word used to specify that the constraint limit will be specified for several zones.

i_{zone2} second number of surveillance zone.

ALLSAME key word used to specify that all the constraint will have the same limit for the zone number between i_{zone1} and i_{zone2}.

END-MAX-POW key word used to specify that the definition of the maximum power surveillance zones is finished.

VOID-REAC-FC key word used to define the full core void reactivity.

$\rho_{V,FC}$ full core void reactivity.

ANALYTIC-FCT key word used to specify that the corresponding constraint will be defined analytically.

cst_{type} type of the analytic constraint (-1 for $\leq$, 0 for $=$ and 1 for $\geq$).

1.1.2 Data input for module functions definition

Because the functions definition is common with other modules, its description will be grouped in this section, as a independent part of the FOBJCT: module description.

The functions definition is based on inverted polish notation logic. Some functions are predefined, but if it is not the case, new functions can be defined manually. In this particular case, variables may be required. Some of them are predefined also, otherwise the user can get them with the same logic as the module GREP. For the functions representing the constraints the user need to specify its number. So it is important to know the order in which constraints where defined.

Table 1.6: Structure **(eval_data)**

[[{ FOBJ | CONSTRAINT i_{cst1} i_{cst2} } { [VARDEF **(vardef_data)**] **(seq_data)** | FUNCT-PREDEF *predef_func* }]]

where

FOBJ key word used to specify that the objective function will be evaluated.

CONSTRAINT key word used to specify that constraint functions between number i_{cst1} and i_{cst2} will be evaluated.

i_{cst1} first number of constraint.

i_{cst2} second number of constraint.

VARDEF key word used to define the variables needed for the function evaluation.

(vardef_data) structure containing the data to the option VARDEF.

(seq_data) structure containing the data used to defined a function directly by the user.

FUNCT-PREDEF key word used to specify that a predefined function will be evaluated.

predef_func name of the predefined function. The predefined function name are :

`UCOST`	
`DX-UCOST`	
`DPHI-UCOST`	
`POWERLIMIT`	
`DX-POWER`	
`DPHI-POWER`	
`KEFF`	
`D-KEFF`	
`VOID-REAC-FC`	
`D-VOID-R-FC`	
`KEFF=KREF`	k_{ref}
`D-KEFF=KREF`	k_{ref}
`MINPCMAX`	q_{moy} i_{cst1} i_{cst2}
`D-MINPCMAX`	q_{moy} i_{cst1} i_{cst2}

Where :
`UCOST` is defined by F_C:

$$F_C = \frac{\langle \frac{C_F(\varepsilon_j)}{B_j}.H, \phi\rangle_{r\acute{e}acteur}}{\langle H, \phi\rangle_{r\acute{e}acteur}} \tag{1.1}$$

`DX-UCOST` is defined by $\frac{\partial F_C}{\partial X_i}$:

$$\begin{aligned} \frac{\partial F_C}{\partial X_i} &= \frac{\langle \frac{\partial Cu}{\partial X_i}, \frac{H}{B}\phi\rangle_{V_i}}{\langle H, \phi\rangle_V} + \frac{\langle Cu\left(\frac{1}{B}\frac{\partial H}{\partial X_i} - \frac{H}{B^2}\frac{\partial B}{\partial X_i}\right), \phi\rangle_{V_i}}{\langle H, \phi\rangle_V} \\ & \quad -F_C.\frac{\langle \frac{\partial H}{\partial X_i}, \phi\rangle_{V_i}}{\langle H, \phi\rangle_V} \qquad i \in (1, n_{var}) \end{aligned} \tag{1.2}$$

`DPHI-UCOST` is defined by $\frac{\partial F_C}{\partial \phi}$:

$$\frac{\partial F_C}{\partial \phi} = S^*_{F_C} = \frac{\frac{Cu}{B}.H(\vec{r}) - F_C.H(\vec{r})}{\langle H, \phi\rangle_V} \tag{1.3}$$

`POWERLIMIT` is defined by q_j:

$$q_j = ZPPF_j.\frac{V}{V_j}\frac{\langle H, \phi\rangle_{V_j}}{\langle H, \phi\rangle_V} \leq \frac{P_{lim}}{\bar{P}} = f_{lim} \tag{1.4}$$

`DX-POWER` is defined by $\frac{\partial q_j}{\partial X_i}$:

$$\begin{aligned} \frac{\partial q_j}{\partial X_i} &= \frac{q_j}{ZPPF_j}.\frac{\partial ZPPF_j}{\partial X_i}.\delta_{ij} + \frac{q_j}{\langle H, \phi\rangle_{V_j}}\langle \frac{\partial H}{\partial X_i}, \phi\rangle_{V_j}.\delta_{ij} \\ & \quad -\frac{q_j}{\langle H, \phi\rangle_V}\langle \frac{\partial H}{\partial X_i}, \phi\rangle_V \qquad i \in (1, I) \text{ et } j \in (1, n_{control-zone}) \end{aligned} \tag{1.5}$$

`DPHI-POWER` is defined by $\frac{\partial q_j}{\partial \phi}$:

$$\frac{\partial q_j}{\partial \phi} = S^*_{q_j} = \frac{ZPPF_j\frac{V}{V_j}.H(\vec{r}_j) - q_j.H(\vec{r})}{\langle H, \phi\rangle_V} \qquad j \in (1, n_{control-zone}) \tag{1.6}$$

where $H(\vec{r}_j) = \begin{cases} H_j & \vec{r} \in V_j \\ 0 & \text{otherwise} \end{cases}$

`KEFF` is defined by k_{eff}: k_{eff} is directly taken in the *FLUX* object.

`D-KEFF` is defined by $\frac{dk_{eff}}{dX_i}$:

$$\frac{dk_{eff}}{dX_i} = -k_{eff}^2 \frac{d\lambda}{dX_i} \tag{1.7}$$

where $\frac{d\lambda}{dX_i}$ is the derivative of the eigenvalue previously calculated with the PERTUR: module.

`VOID-REAC-FC` is defined by ρ_V:

$$\rho_V = \lambda - \lambda_V = \frac{1}{k_{eff}} - \frac{1}{k_{eff,V}} \tag{1.8}$$

where k_{eff} and $k_{eff,V}$ are directly taken in the *FLUX* and *FLUXP* object respectively.

`D-VOID-R-FC` is defined by $\frac{d\rho_V}{dX_i}$:

$$\frac{d\rho_V}{dX_i} = \frac{d\lambda}{dX_i} - \frac{d\lambda_V}{dX_i} \tag{1.9}$$

where $\frac{d\lambda}{dX_i}$ is the derivative of the eigenvalue previously calculated with the PERTUR: module and $\frac{d\lambda_V}{dX_i}$ is the derivative of the eigenvalue previously calculated for a voided reactor with the PERTUR: module.

`KEFF=KREF` is defined by Δk_{eff}:

$$\Delta k_{eff} = \left(k_{eff} - k_{ref}\right)^2 \tag{1.10}$$

where k_{ref} is the required reference multiplication factor.

`D-KEFF=KREF` is defined by

$$\frac{d\Delta k_{eff}}{dX_i} = -2 * \left(k_{eff} - k_{ref}\right) .k_{eff}^2 \frac{d\lambda}{dX_i} \tag{1.11}$$

where k_{ref} is the required reference multiplication factor and $\frac{d\lambda}{dX_i}$ is the derivative of the eigenvalue previously calculated with the PERTUR: module.

`MINPCMAX` is defined by

$$EPmoy = \sum_{j=i_{cst1}|q_j>q_{moy}}^{i_{cst2}} \left(q_j - q_{moy}\right)^{2m} \tag{1.12}$$

where q_{moy} is the average power zone. m can be changed with the module QLPUTL:. The sum is performed from constraint i_{cst1} to i_{cst2}.

`D-MINPCMAX` is defined by

$$\frac{d\Delta EPmoy}{dX_i} = \sum_{j=i_{cst1}|q_j>q_{moy}}^{i_{cst2}} 2m\left(q_j - q_{moy}\right)^{2m-1} \frac{dq_j}{dX_i} \tag{1.13}$$

where q_{moy} is the average power zone. m can be changed with the module QLPUTL:. The sum is performed from constraint i_{cst1} to i_{cst2}.

Table 1.7: Structure **(vardef_data)**

[[`LOAD` *object* [[`DOWN` *repertory*]] `GREP` *data_name* `IN` *var_name*]]
[[`MSYS*FLUX` *object_sys* { `A` | `B` } *object_flux* [`ADJOINT`] `IN` *var_name*]]
[[`PREDEF` *predef_var*]]

where

LOAD key word used to define the object where the data are stored.

object name of the object.

DOWN key word used to go in a sub-directory.

repertory name of the repertory.

GREP key word used to define the name of the data to load.

data_name name of the data to load.

IN key word used to define the name of the local variable.

var_name name of the local variable name.

MSYS*FLUX key word used to specify that a local variable will be calculated by the product of a system matrix and a flux (or adjoint).

object_sys name of the object L_SYSTEM.

A key word used to specify that the system matrix corresponding to the lost of the neutrons will be used. $(\mathbf{A} - \lambda B)\phi = 0$

B key word used to specify that the system matrix corresponding to the production of the neutrons will be used. $(A - \lambda \mathbf{B})\phi = 0$

object_flux name of the object L_FLUX.

ADJOINT key word used to specify that the adjoint flux will be used instead of the flux (default value). In this case the adjoint system matrix are used automatically.

PREDEF key word used to specify that a predefined variable will be load.

predef_var name of the predefined variable. The predefined variable name are define below.

key word	contents	size
FLUX	neutron flux distribution	*nun* × *ngrp*
AFLUX	adjoint flux distribution of the second provided L_FLUX	*nun* × *ngrp*
FLUX2	neutron flux distribution of the second provided L_FLUX	*nun* × *ngrp*
AFLUX2	adjoint flux distribution	*nun* × *ngrp*
FLUX-AV	average flux distribution by channel	*nch* × *ngrp* × *nzone*
FLUX-AX	axial average flux distribution	*nz* × *ngrp*
DIFFX	diffusion coefficient along X abcisse	*nun* × *ngrp*
DIFFY	diffusion coefficient along Y abcisse	*nun* × *ngrp*
DIFFZ	diffusion coefficient along Z abcisse	*nun* × *ngrp*
TOTAL	total cross-sections	*nun* × *ngrp*
NFTOT	fission cross-sections	*nun* × *ngrp*
NUSIGF	number of neutrons per fission time fission cross-sections	*nun* × *ngrp*
H-FACTORS	fission cross section times the energy recovered by fission	*nun* × *ngrp*
CHI	fission spectrum	*nun* × *ngrp*
SIGW-0	isotropic component of the within group of the scattering of the scattering cross-sections	*nun* × *ngrp*

SIGW-1	linearly anisotropic component of the within group of the scattering of the scattering cross-sections	*nun* × *ngrp*
D-TOTAL	derivative of total cross-sections	*nun* × *ngrp*
D-CHI	derivative of fission spectrum	*nun* × *ngrp*
D-DIFFX	derivative of diffusion coefficients along X abcisse	*nun* × *ngrp*
D-DIFFY	derivative of diffusion coefficients along Y abcisse	*nun* × *ngrp*
D-DIFFZ	derivative of diffusion coefficients along Z abcisse	*nun* × *ngrp*
D-NUSIGF	derivative of fission cross-sections	*nun* × *ngrp*
D-NFTOT	derivative of number of neutrons per fission time fission cross-sections	*nun* × *ngrp*
D-HFACT	derivative of fission cross section times the energy recovered by fission	*nun* × *ngrp*
D-SIGW0	derivative of isotropic component of the within group of the scattering of the scattering cross-sections	*nun* × *ngrp*
D-SIGW1	derivative of linearly anisotropic component of the within group of the scattering of the scattering cross-sections	*nun* × *ngrp*
A*PHI	A system matrix times neutron flux vector	*nun* × *ngrp*
B*PHI	B system matrix times neutron flux vector	*nun* × *ngrp*
AP*PHI	perturbated A system matrix times neutron flux vector	*nun* × *ngrp*
BP*PHI	perturbated B system matrix times neutron flux vector	*nun* × *ngrp*
FUNCVALUE	value of the function, usually used when the derivative function is calculated (see the definition of the DX-UCOST predifined function for example).	*ncst*+1
FUNCZVOL	value of the volume on which the function is defined / integrated.	*ncst*+1
KEFF	k_{eff}	*nun* × *ngrp*
KEFF-VOID	k_{eff} corresponding to a pertubated flux	*nun* × *ngrp*
D-LAMBDA	derivative of the eigenvalue	*nun* × *ngrp*
D-LAMBDA-V	derivative of the eigenvalue corresponding to a pertubated flux	*nun* × *ngrp*

Table 1.8: Structure **(seq_data)**

INIT
[[**(data)**
| INTEGRAL { REACTOR | CORE | CST-ZONE | VAR-ZONE | DBL-ZONE | DISCRETE-ALL | DISCRETE-COR | DISCRETE-CST }
... [[**(data)** | ENERGY { ALL | gpr_{from} gpr_{to} } [[**(data)**]] END-ENERGY]]
... END-INTEGRAL]]]]
END

where

INIT key word used to specify that the function definition will follow.

(data) structure containing the data used to defined parts of the function.

INTEGRAL key word used to define an integral over one volume and energy.

REACTOR key word used to specify that the integration volume is the reactor

CORE key word used to specify that the integration volume is the core (all the bundles).

`CST-ZONE` key word used to specify that the integration volume is a control zone of one constraint.

`VAR-ZONE` key word used to specify that the integration volume is a zone where a decision variable applies.

`DBL-ZONE` key word used to specify that the integration volume is on the intersection of a control zone of one constraint and a zone where a decision variable applies.

`DISCRETE-ALL` key word used to specify that the integration will be performed only on the energy for every point of the reactor.

`DISCRETE-COR` key word used to specify that the integration will be performed only on the energy for every point of the core.

`DISCRETE-CST` key word used to specify that the integration will be performed only on the energy for every point of a control zone of one constraint.

`ENERGY` key word used to define the energy part of the integration.

`ALL` key word used to specify that the integration will be performed on all energy groups.

grp_{from} number of the first energy group for the integration.

grp_{to} number of the last energy group for the integration.

`END-ENERGY` key word used to specify that the definition of the integral over energy is finished.

`END-INTEGRAL` key word used to specify that the definition of the integral is finished.

`END` key word used to specify that the definition of the function is finished.

Table 1.9: Structure **(data)**

[[{ *real* | `VAR` *loc_var_name* | *operator* | `VARF` *loc_var_name* }]]

where

real real number.

`VAR` key word used to specify that a local variable will be used.

`VARF` key word used to specify that a local variable which depend with the functional will be used (ex: zone volume).

loc_var_name name of a local variable name. Note : it has to be loaded before.

operator name of a numerical operator. The name must be one of these : `PLUS`, `+`, `MINUS`, `-`, `TIMES`, `*`, `DIVISION`, `/`, `POWER`, `**`, `MAX`, `MIN`, `LOG`, `LN`, `EXP`, `SIN`, `COS`, `TAN`, `ABS`, `SQRT`.

1.1.3 Examples of function definition

We will now give a few examples which will permit users a better understanding of the procedure to define the function for optimization in DONJON.

1. Predefined function:

   ```
   OPTIMIZE := FOBJCT: OPTIMIZE FLUX MACRO ::
    EVAL-OBJ-CST  CONSTRAINT 2 10 PREDEF POWERLIMIT
    ;
   ```

2. Function defiend by user:

   For example, we suppose that a functional u defined by the user is :

   $$f_{cost} = 2 * k_{eff} * \int_{CORE} C_U \int_{allenergygroups} \phi dE.dV \tag{1.14}$$

   Where :
   $\underline{C_U}$ is the fuel cost.
   $\overline{\phi}$ the flux distribution.

   ```
   OPTIMIZE := FOBJCT: OPTIMIZE FLUX MACRO ::
    FUEL-COST-DF  MEMORY
    EVAL-OBJ-CST  FOBJ    VARDEF LOAD FLUX GREP K-EFFECTIVE IN KEFF
                                 PREDEF FLUX
                          2.0
                          VAR KEFF
                          *
                          INTEGRAL CORE
                              VAR FUELCOST
                              ENERGY ALL
                                  VAR FLUX
                                  END-ENERGY
                              *
                              END-INTEGRAL
                          *
                          END
    ;
   ```

1.2 The `QLPUTL:` module

The QLPUTL: module is used to define the optimization options and tools. It is also used to do some pre-calcultaion.

The calling specifications are:

Table 1.10: Structure `QLPUTL:`

OPTIM `:=` `QLPUTL:` *OPTIM* [*FLUX*] [*MAPFL*] [*MACRO* [*MACROP*]] [*SYS* [*SYSP*] *TRACK*] `::` **(descqlputl)**

where

OPTIMIZE `character*12` name of the extended OPTIMIZE.

FLUX `character*12` name of the FLUX linked list. This object is used for the function evaluation. It is recommended to provide it even if no function evaluation is done for many parameters reading. file.

MAPFL `character*12` name of the extended MAP linked list file containing fuel regions description and burnup informations. If *MAPFL* appears on the RHS, the information in it will be red for many parameters initialisation.

MACRO `character*12` name of the MACROLIB linked list file containing the mixtures cross sections. If it appears on RHS, it means it will be necessary for a function evaluation (objective or constraint).

MACROP `character*12` name of the MACROLIB linked list file containing the mixtures perturbated cross sections. If it appears on RHS, it means it will be necessary for a function evaluation (objective or constraint).

SYS `character*12` name of the SYSTEM containing the reference system matrices. SYSTEM must be a linked list. If it appears on RHS, it will be necessary for 'system matrice times flux' calculations.

SYSP `character*12` name of the SYSTEM containing the perturbated system matrices. SYSTEM must be a linked list. If it appears on RHS, it will be necessary for 'perturbated system matrice times flux' calculations.

TRACK `character*12` name of the TRACK (type `L_TRIVAC`) containing the tracking informations. TRACK must be a linked list. If it appears on RHS, it will be necessary for 'system matrice times flux' calculations.

(descqlputl) structure containing the data to module `PQLUTL:`.

1.2.1 Data input for module `QLPUTL`*:*

Table 1.11: Structure **(descqlputl)**

[EDIT *iprint*]
[DEFINITION **(def_data)**]
[STEP-VALID [TEST-CST-VLD] >> *test1* <<]
[STEP-INTERP { PUT | RECOVER >> *test2* << }]
[DX-METHOD { EPS *epsilon* | PREVIOUS }]
[NEW-VAL-UPDT]
[PERTURB-VAR { i_{var1} | RESTORE }]
[BKP-MACRO-P i_{var2}]
[MAT*FLUX { A*PHI | B*PHI | AP*PHI i_{var3} | BP*PHI i_{var3} }]
[LA-PNLT [INITIAL] [F-EVAL] [COEF-UPDATE] [CONV-TEST >> *conv* <<]
... [ALMOST-FSBLE >> *feas* <<]]
[HISTORY i_{iter1} [POWER-CHA] [K-EFFECTIVE] [QUAD-CST]]
... [CONSTRAINT { ALL | RANGE << i_{cst1} >> << i_{cst2} >> | << i_{cst1} >> }] ... [DIRECT << *num* >> { val_i, $i = 1, num$}]] [POWER-CHA#2] [POWER-CHA#3]
;

where

`EDIT` key word used to set *iprint*.

iprint index used to control the printing in module.

`DEFINITION` key word used to define the optimization options.

(def_data) structure containing the data to the option `DEFINITION`.

`STEP-VALID` key word used to verify if the new $\{X_i\}$ ends with a better objective function.

`TEST-CST-VLD` key word used to verify if the new $\{X_i\}$ respects the constraints.

test logical value for the validition of the new decision variables. *test* equals *.true.* if $F_C\left(X_i^{k+1}\right)$ is better than $F_C\left(X_i^k\right)$.

`STEP-INTERP` key word used to specify that an interpolation of the objective function for the midle point between $\{X_i^k\}$ and $\{X_i^{k+1}\}$ will be done.

`PUT` key word used to calculate and store the middle value.

`RECOVER` key word used to verify the middle value.

test2 logical value for the validation of interpolation. If $F_C\left(X_i^{k+\frac{1}{2}}\right)$ is less than $F_C\left(X_i^{k+1}\right)$ then the middle value is kept, otherwise the new value is restored.

`DX-METHOD` key word used to define which method will be used to evaluate the perturbated cross-section.

$$\frac{d\Sigma}{dX_i^k} = \frac{\Sigma(X_{i,p}^k) - \Sigma(X_i^k)}{X_{i,p}^k - X_i^k} \tag{1.15}$$

where $X_{i,p}^{k}$ is the perturbated decision variable.

EPS key word used to define $X_{i,p}^{k}$ by $X_{i}^{k} * (1 + \epsilon)$.

epsilon value of ϵ.

PREVIOUS key word used to define $X_{i,p}^{k}$ by X_{i}^{k-1}.

NEW-VAL-UPDT key word used to update the new decision variables.

PERTURB-VAR key word used to perturbate a decision variable.

i_{var1} number of the decision variable to perturbate.

RESTORE key word used to restore the unperturbated decision variables.

BKP-MACRO-P key word used to store the perturbated macroscopic cross-section. By default all cross-section are stored. To store only some of them, see PQLUTL/DEFINITION/BKP-MCR-P-XS.

i_{var2} number of the decision variable for which the cross-section are perturbated and will be stored.

MAT*FLUX key word used to precalulate the system matrice times the flux.

A*PHI key word used to precalulate the $A.\phi$ (i_{var3}=0 is implicit).

B*PHI key word used to precalulate the $B.\phi$ (i_{var3}=0 is implicit).

AP*PHI key word used to precalulate the $A_p.\phi$.

BP*PHI key word used to precalulate the $B_p.\phi$.

i_{var3} number of the **step** directory where the $A.\phi$ and $B.\phi$ will be stored. i_{var3} represents the decision variable for which the system matrice were perturbated and will be multiplied by the flux for optimization. The result will be stored in L_OPTIMIZE/'STEP//HSIGN ' with WRITE(HSIGN,I8) i_{var3}.

LA-PNLT key word used to specify that a task related to the augmented lagrangian or penalty method is performed.

INITIAL key word used to initialize the constraint weight (if not already done) and the lagrangian coefficient (when augmented lagrangian method is used).

F-EVAL key word used to calculate the augmented lagrangian or penalty function. It can be also used with tabu search to evaluate the corresponding objective function.

COEF-UPDATE key word used to update the constraint weights and lagrangian coefficients (if necessary) in an external iteration.

CONV-TEST key word used to specify that a convergence test for external iteration will be performed.

conv logical value representing the result of the external convergence test.

ALMOST-FSBLE key word used to specify that a test will be permorfed to check if the current point is 'almost feasible'.

feas logical value representing the result of the 'almost feasible' test. The maximum error allowed to set *feas* to .true. is a relative difference between prescribed and current constraint values lower than the convergence crriterium.

HISTORY key word used to store the decision vector and the functionnal values for iteration i_{iter1}.

i_{iter1} integer representing the iteration number.

POWER-CHA key word used to specify that the channel power distribution is also stored.

K-EFFECTIVE key word used to specify that k_{eff} value is also stored.

QUAD-CST key word used to specify that quadratic constraint limit is also stored.

CONSTRAINT key word used to specify that some constraint values are also stored.

ALL key word used to specify that all constraint values are stored.

RANGE key word used to specify that values of a range of constraint are stored.

i_{cst1} integer representing the first or only number of constraint.

i_{cst2} integer representing the second number of constraint.

DIRECT key word used to specify that values provided by the user are stored.

num integer representing the number of values provided by the user.

val_i real representing the values provided by the user.

POWER-CHA#2 same key word as POWER-CHA. It can be used when channel power distribution for a perturbated state of the reactor is also stored.

POWER-CHA#3 same key word as POWER-CHA#2.

Table 1.12: Structure **(def_data)**

[METHOD { SIMPLEX | LEMKE | MAP | AUG-LAGRANG | PENAL-METH }]
[{ MAXIMIZE | MINIMIZE }]
[INN-STEP-LIM *step*]
[VAR-WEIGHT { TYP-BURNUP *weight* | TYP-ENRICH *weight* }]]
[[CST-WEIGHT { i_{cst1} *weight* | RANGE i_{cst1} i_{cst2} { ALLSAME *weight* | $weight_{j}$,j=i_{cst1},i_{cst2} } }]]
[OUT-STEP-LIM *step*]
[INN-STEP-NMX n_{max}]
[OUT-STEP-NMX n_{max}]
[INN-STEP-EPS ϵ_{ext}]
[OUT-STEP-EPS ϵ_{inn}]
[STEP-REDUCT { HALF | PARABOLIC }]
[CST-QUAD-LIM *epsilon4*]
[BKP-MCR-P-XS { ADD | NEW } [[*XS_name*]]]
[F-C-VOLUME [FOBJ {REACTOR | CORE}] [CONSTRAINT i_{cst1} i_{cst2} {REACTOR | CORE | ZONE}]
[CST-WGT-MFAC α]
[CST-VIOL-EPS ϵ_{cst}]
[MIN(PCMX)^2N m]

where

`METHOD` key word used to define the quasi-linear programming method.

`SIMPLEX` key word used to specify that the SIMPLEX method will be used.

`LEMKE` key word used to specify that the LEMKE method will be used.

`MAP` key word used to specify that the MAP method will be used.

`AUG-LAGRANG` key word used to specify that the augmented lagrangian method will be used.

`PENAL-METH` key word used to specify that the penalty method will be used.

`MAXIMIZE` key word used to specify that the optimization problem will be a maximization.

`MINIMIZE` key word used to specify that the optimization problem will be a minimization (default).

`INN-STEP-LIM` key word used to limit the inner step of the optimization problem.

step limit for a step.

`VAR-WEIGHT` key word used to set the weight of the different types of the decision variables for the quadratic limit of the outer step of the optimization problem.

$$\sum w_i.X_i^2 \leq S_k \tag{1.16}$$

`CST-WEIGHT` key word used to set the weight of the constraints.

i_{cst1} number of the (first) constraint to set the weight.

weight weight of the constraint(s).

`RANGE` key word used to specify that several constraint weights will be set.

i_{cst2} number of the last constraint to set the weight.

`ALLSAME` key word used to specify that the several constraint weights will be identical.

`TYP-BURNUP` key word used to set a limit for a burnup type decision varaible.

`TYP-ENRICH` key word used to set a limit for a enrichment type decision varaible.

weight weight for the decision variable.

`OUT-STEP-LIM` key word used to limit the outer step of the optimization problem.

`INN-STEP-NMX` key word used to set the maximum of inner iteration of the optimization problem.

`OUT-STEP-NMX` key word used to set the maximum of outer iteration of the optimization problem.

n_{max} maximum number of iterations.

`INN-STEP-EPS` key word used to set the tolerence of inner iteration convergence criterium of the optimization problem.

ϵ_{ext} tolerence for convergence of inner iterations (real).

`OUT-STEP-EPS` key word used to set the tolerence of outer iteration convergence criterium of the optimization problem.

ϵ_{inn} tolerence for convergence of external iterations (real).

`STEP-REDUCT` key word used to define the method of the reduction of the outer step.

HALF key word used to specify that the step will be reduced by a factor 2.

PARABOLIC key word used to specify that the step will be reduced with the parabolic method.

CST-QUAD-LIM key word to set the parameter *epsilon4* for the quadratic limit of the step.

epsilon4 parameter ϵ_4.

BKP-MCR-P-XS key word used to specify which of the perturbated macroscopic cross-section will be stored on a backup repertory of the L_OPTIMIZE object. (for complementary information see PQLUTL/BKP-MACRO-P)

ADD key word used to add name of cross-section to be stored.

NEW key word used to define a new list of name of cross-section to be stored.

XS_name name of the cross-section to be stored. The list of available name is: DIFFX, DIFFY, DIFFZ, TOTAL, NFTOT, NUSIGF, H-FACTORS, CHI, SIGW-0, SIGW-1, SCAT-0, SCAT-1, CHI, FIXE.

F-C-VOLUME key word used to specify that the volume where the functionals apply will be calculated.

FOBJ key word used to specify that the volume corresponding to the objective function will be computed.

CONSTRAINT key word used to specify that the volume corresponding to the constraints between number i_{cst1} and i_{cst2} will be computed.

i_{cst1} number of the first constraint for which the volume will be calculated.

i_{cst2} number of the last constraint for which the volume will be calculated.

REACTOR key word used to specify that the volume of the functional is the whole reactor.

CORE key word used to specify that the volume of the functional is the core represented by all the fuel channels.

ZONE key word used to specify that the volume of the functional is its corresponding zone.

CST-WGT-MFAC key word used to set the multiplication factor α for the constraint weight update.

α multiplication factor for the constraint weight update.

CST-VIOL-EPS key word used to set the precision ϵ_{cst} when the validation of a new point is done with the constraint validity.

ϵ_{cst} precision for the constraint validity.

MIN(PCMX)^2N key word used to set the coefficient m of the power distribution optimization problem.

m coefficient for channel having power greater than the average.

1.3 The PERTUR: module

The PERTUR: module is used to compute gradients of function using the first order of perturbation theory. Then it can be used to calculate the variation of reactivity of one reactor with a small perturbation of the cross-sections. There is two different approches to solve the problem of reactivity.

The first method uses in fact the module 'SORKEF:' of the previous version. This part of the module computes source terms based on a first order perturbation theory over diffusion equation. The direct diffusion equation for system matrix perturbations ΔA and ΔB can be written for a linear perturbation of the flux $\phi = \phi_o + \Delta\phi$:

$$(A_o - \lambda_o B_o)\Delta\phi = -(\Delta A - \lambda_o \Delta B - \Delta\lambda B_o)\phi_o \tag{1.17}$$

The direct source term is then simply $(\Delta A - \lambda_o \Delta B - \Delta\lambda B_o)\phi_o$ where $\Delta\lambda$ is the first order estimate of the eiganvalue variation, Rayleigh formulation.
The adjoint source terms are easily obtained from a similar expression of the ajoint diffusion equation.

The second method is a part of the optimization modules package. To calculate $\frac{\Delta\lambda}{\Delta X_i}$, the user has to precalculate system matrices * flux. It can be done easily and automaticaly by using the module PQLUTL: with the key word 'MAT*FLUX'. For the specific case of the reactivity, the variation of the inverse of k-effective is given by the following equation:

$$\frac{\partial\lambda}{\partial X_i} = \lambda\left(\frac{\langle\phi^*, \frac{\partial A}{\partial X_i}\phi\rangle}{\langle\phi^*, A\phi\rangle} - \frac{\langle\phi^*, \frac{\partial B}{\partial X_i}\phi\rangle}{\langle\phi^*, B\phi\rangle}\right) \tag{1.18}$$

$$\frac{\Delta\lambda}{\Delta X_i} = \lambda\left(\frac{\langle\phi^*, \frac{A_p}{\Delta X_i}\phi\rangle}{\langle\phi^*, A\phi\rangle} - \frac{\langle\phi^*, \frac{B_p}{\Delta X_i}\phi\rangle}{\langle\phi^*, B\phi\rangle}\right) \quad i \in (1, n_{var}) \tag{1.19}$$

The calling specifications are:

Table 1.13: Structure **(PERTUR:)**

OPTIMIZE := `PERTUR:` *OPTIMIZE FLUX* [*SYS* [*SYSP*] *TRACK*] [*MACRO* [*MACROP*]] : : **(pertur_data)**

where

GPT `character*12` name of the SOURCE containing the source terms. If *GPT* appears on the RHS, the previous values will be updated.

FLUX `character*12` name of the FLUX containing the unperturbed flux, direct or adjoint.

SYS `character*12` name of the SYSTEM containing the reference system matrices. SYSTEM must be a linked list.

SYSP `character*12` name of the SYSTEM containing the perturbation of the system matrices.

TRACK `character*12` name of the TRACK (type `L_TRIVAC`) containing the tracking informations. TRACK must be a linked list.

OPTIMIZE `character*12` name of the OPTIMIZE containing the optimization informations. *GPT* must appear on the RHS to be able to updated the previous values.

(pertur_data) structure containing the data to the second choice for the module `PERTUR:`.

1.3.1 Data input for module `PERTUR`*:*

Table 1.14: Structure **(pertur_data)**

[`EDIT` *iprint*]
[`VARMUN` { i_{var1} i_{var2} | `ALL` }
. . . {`D-LAMBDA` | `D-LAMBDA/DX` | `D-LAMBDA-V` | `D-LAMBDA-V/DX` |**(eval_data)** }] ;

where

`EDIT` key word used to set *iprint*.

iprint index used to control the printing in module.

`VARNUM` key word used to define the decision variable for which the perturbation theory calculations will be done.

i_{var1} number of the first decision variable.

i_{var2} number of the second decision variable.

`ALL` key word used to specify that the perturbation theory calculations will be done for all the decision variables.

`D-LAMBDA` key word used to specify that absolute variation of the eigenvalue perturbation will be calculated for the corresponding perturbated decision variables.

`D-LAMBDA/DX` key word used to specify that derivative of the eigenvalue perturbation will be calculated for the corresponding perturbated decision variables.

`D-LAMBDA-V` key word used to specify that absolute variation of the eigenvalue perturbation will be calculated for the corresponding perturbated decision variables and that the provided system matrix correspond to the voided reactor (or an other configuration of the reactor).

`D-LAMBDA-V/DX` key word used to specify that derivative of the eigenvalue perturbation will be calculated for the corresponding perturbated decision variables and that the provided system matrix correspond to the voided reactor (or an other configuration of the reactor).

(eval_data) see explanations in the module FOBJCT: key word 'EVAL-OBJ-CST' 1.1.2. Some predefined function are described too.

1.4 The `GPTSRC:` module

The GPTSRC: module is used to calculate the sources terms (direct and / or adjoint) for generalized perturbation theory.

The calling specifications are:

Table 1.15: Structure `GPTSRC:`

{ *GPT* `:=` `GPTSRC:` [*GPT*] *OPTIMIZE FLUX* [*SYS* [*SYSP*] *TRACK*] [*MACRO*] [*MAPFL*]
`::` **(gptsrc_data)**

where

GPT `character*12` name of the GPT linked list file containing fuel regions description and burnup informations. If *GPT* appears on the RHS, the information previously stored in *GPT* is modified if necessary and stored.

OPTIMIZE `character*12` name of the extended OPTIMIZE linked list.

FLUX `character*12` name of the FLUX linked list. This object is used for the function evaluation. It is recommended to provide it even if no function evaluation is done for many parameters reading. file.

MACRO `character*12` name of the MACROLIB linked list file containing fuel regions description and burnup informations. If it appears on RHS, is means it will be necessary for a function evaluation (objective or constraint).

MAPFL `character*12` name of the extended MAP. If *MAPFL* appears on the RHS, the information in it will be red for many parameters initialisation.

TABFL `character*12` name of the TABLE linked list file containing fuel regions description and burnup informations. If it appears on RHS, is means it will be necessary for a function evaluation (objective or constraint).

(gptsrc_data) structure containing the data to module `GPTSRC:`.

1.4.1 Data input for module `GPTSRC:`

Table 1.16: Structure `gptsrc_data`

[`EDIT` *iprint*]
[[`DIRECT` { i_{var1} i_{var2} | `ALL` }]]
[[`ADJOINT` **(eval_data)**]]
[[`OTHER` { `DIRECT` | `ADJOINT` } i_{var1} **(eval_data)**]] ;

where

`EDIT` key word used to set *iprint*.

iprint index used to control the printing in module.

`DIRECT` key word used to calculate a direct source term for decision variables S_i.

$$S_i = \frac{\partial \left(A - \lambda B\right)}{\partial X_i}.\phi = \frac{A_p\phi - A\phi}{\Delta X_i} - \frac{\Delta\lambda}{\Delta X_i}.B\phi - \lambda.\frac{B_p\phi - B\phi}{\Delta X_i} \tag{1.20}$$

i_{var1} number of the first decision variable.

i_{var2} number of the second decision variable.

`ALL` key wod used to specify that the direct source terms calculations will be done for all the decision variables.

`ADJOINT` key word used to calculate a adjoint source term for decision variables S_j^*.

$$S_j^* = \frac{\partial G_j}{\partial \phi} \tag{1.21}$$

(eval_data) see explainations in the module FOBJCT: key word 'EVAL-OBJ-CST'. Some predefined function are described too.

1.5 The `GPTGRD:` module

The GPTGRD: module is used to compute the gradient of functions using the generalized perturbation theory. To do that the user must precalculate the sources terms (module GPTSRC) and the generalized adjoints (module GPTFLU).

The GPTGRD: module also allows to define directly values of gradient of functions.

The calling specifications are:

Table 1.17: Structure `GPTGRD:`

OPTIMIZE `:=` `GPTGRD:` *OPTIMIZE FLUXP* [*SYS* [*SYSP* [*SYS2* [*SYS2P*]]] *TRACK*
MACRO [*FLUX*] [*MATEX*] [*MAPFL*] `::` [**(direct_data)**] **(gptgrd_data)**
OPTIMIZE `:=` `GPTGRD:` *OPTIMIZE* `::` **(direct_data)**

where

OPTIMIZE `character*12` name of the OPTIMIZE containing the optimization informations. *GPT* must appear on the RHS to be able to updated the previous values.

FLUXP `character*12` name of the FLUX containing the generalized adjoint flux, explicit or implicit.

TRACK `character*12` name of the TRACK linked list file containing tracking information corresponding to *FLUXP*.

MACRO `character*12` name of the MACROLIB linked list file containing fuel regions description and burnup informations.

FLUX `character*12` name of the FLUX containing the unperturbed flux, direct or adjoint. If it appears on RHS, is means it will be necessary for a function evaluation (objective or constraint).

GPT `character*12` name of the GPT linked list file containing fuel regions description and burnup informations. If it appears on RHS, is means it will be necessary for a function evaluation (objective or constraint).

MATEX `character*12` name of the MATEX object created by the `USPLIT:` module and containing the complete reactor material index including devices.

MAPFL `character*12` name of the MAP linked list file containing the fuel map informations.

(direct_data) structure containing the data to the direct definition of gradient for the module `GPTGRD:`.

(gptgrd_data) structure containing the data to the generalized theory based gradients choice for the module `GPTGRD:`.

1.5.1 Data input for module GTPGRD:

Table 1.18: Structure direct_data

[NEW-VALUE] [REL]

DIRECT-VALUE i_{var1} [i_{var2}] { FOBJ | CONSTRAINT i_{fcn1} i_{fcn1} }

...... *grad* (j=1, $(i_{var2} - i_{var1} + 1).(i_{fcn2} - i_{fcn1} + 1)$)

[;]

where

NEW-VALUE key word used to specify that the value of gradient is set to zero.

REL key word used to recover the epsilon in record OPT-PARAM-R of object *OPTIMIZE*.

DIRECT-VALUE key word used to specify that the value of gradient will be directly given by the user.

i_{var1} first decision variable for which the gradient will be defined.

i_{var2} last decision variable for which the gradient will be defined. If it is not defined, the default value is i_{var1}.

FOBJ key word used to specify that the gradient of the objective function will be defined.

CONSTRAINT key word used to specify that the gradient of conctraints will be defined.

i_{fcn1} first constraint for which the gradient will be defined.

i_{fcn2} last constraint for which the gradient will be defined.

grad value of the gradient.

; this key word has to be provided if **(gptgrd_data)** is not used.

Table 1.19: Structure gptgrd_data

GPT [[DIRECT { i_{var1} i_{var2} | ALL } **(eval_data)**]]

... [[INDIRECT [{ EXPLICIT | IMPLICIT }] { i_{var1} i_{var2} | ALL }] { FOBJ | CONSTRAINT i_{fcn1} i_{fcn2} }]]

;

where

DIRECT key word used to specify that the direct part of the gradient will be calculated.

i_{var1} first decision variable for which the gradient will be defined.

i_{var2} last decision variable for which the gradient will be defined.

ALL key word used to specify that the gradient will be calculated for all decision variables.

(eval_data) see explainations in the module FOBJCT: key word 'EVAL-OBJ-CST'. Some predefined function are described too.

`INDIRECT` key word used to specify that the indirect part of the gradient will be calculated.

`EXPLICIT` key word used to obtain the solution of an direct fixed source eigenvalue problem.

`IMPLICIT` key word used to obtain the solution of an adjoint fixed source eigenvalue problem. If neither '`EXPLICIT`' nor '`IMPLICIT`' are provided the default value will be chosen as a function of n_{var} and $n_{cst} + 1$.

`FOBJ` key word used to specify that the gradient of the objective function will be defined.

`CONSTRAINT` key word used to specify that the gradient of conctraints will be defined.

i_{fcn1} first constraint for which the gradient will be defined.

i_{fcn2} last constraint for which the gradient will be defined.

1.6 The TABU: module

The TABU: module is used to define options and data storage for the tabu search optimization algorithm.

The calling specifications are:

Table 1.20: Structure TABU:

TABUSH [*OPTIM*] := TABU: [*TABUSH*] *OPTIM* :: **(desctabu)**

where

TABUSH `character*12` name of the extended TABU linked list file.

OPTIM `character*12` name of the extended OPTIMIZE linked list file. If *OPTIM* appears on the LHS, decision variables or their limits (for exemple) may be changed for further evaluation of the objective function and constraint.

(desctabu) structure containing the data to module TABU:.

1.6.1 Data input for module TABU:

Table 1.21: Structure **(desctabu)**

[EDIT *iprint*]

[DEFINITION **(def_data)**]

[NEIGHB-CREAT]

[NEIGHB-CHOIC [INIT-PRO-LIST] [NELDER-MEAD] i_{neig}]

[NEIGHB-EVAL [INIT-PRO-LIST] [NELDER-MEAD] i_{neig}]

[NEIGHB-BEST [CONV-TEST >> *Lconv* <<] [PROMISE-TEST [NO-THRESHOLD] >> *Lpro* <<]]

[PROMISE-AREA [NELDER-MEAD] { CREATION | UPDATE }]

[NELDER-MEAD **(nelder_data)**]

;

where

EDIT key word used to set *iprint*.

iprint index used to control the printing in module.

DEFINITION key word used to define the tabu search optimization options.

(def_data) structure containing the data to the option DEFINITION.

NEIGHB-CREAT key word used to create the neighborhood for the decision variable set stored as the current one in the *TABUSH* object.

`NEIGHB-CHOIC` key word used to specify the number i_{neig} within the neighbors which will be evaluated. The corresponding decision variable values are copied in the *OPTIM* object as the current decision variables.

`NEIGHB-EVAL` key word used to specify the number i_{neig} within the neighbors which have been evaluated. The corresponding functional values and the tabu function result are stored in the *TABU* object.

`INIT-PRO-LIST` key word used to specify the initial elements of the promising list are selected and evaluated (and not the neighbors).

`NELDER-MEAD` key word used to specify the initial elements of the polytope for the Nelder-Mead simplex algorithm are selected and evaluated (and not the neighbors).

i_{neig} integer value for a neighbor point to be / which has been evaluated.

`NEIGHB-BEST` key word used to check the neighbors results. The best neighbor result is compared to the fittest solution ever found. An update is performed if necessary. Tests for global convergence and promising area detection can be done. The tabu list is updated.

`CONV-TEST` key word used to verify if global convergence is achieved.

Lconv logical value for the global convergence. *Lconv* equals *.true.* if Nit is greater than Nit_{max}.

`PROMISE-TEST` key word used to verify if a promising area has been detected.

`NO-THRESHOLD` key word used to specify that no threshold limits the acceptance of promising areas.

Lconv logical value for the promising area detection.

`PROMISE-AREA` key word used to specify that calculation based on gradient methods will be performed on a promising area previously detected.

`NELDER-MEAD` key word used to specify the Nelder-Mead simplex algorithm is used instead of the gradient method.

`CREATION` key word used to define the area for the local gradient method optimization algorithm. A backup of original decision variable limits is done in *TABUSH* object and new smaller ones are stored in *OPTIM* object.

`UPDATE` key word used to set the gradient method result for the promising area as the new current decision variable. An update of the best point ever found is done is necessary. The promising list is also updated.

`NELDER-MEAD` key word used to specify the Nelder-Mead simplex algorithm is selected.

(def_data) structure containing the data to the option `NELDER-MEAD` corresponding to the different geometric transformations.

Table 1.22: Structure **(def_data)**

[`ISEED` *seed*]
[`NEIGHBOR-NB` *ngh*]
[`NEIGHBOR-TYP` { `RECTANGLE` | `BALL` }]

continued on next page

Structure **(def_data)** continued from last page

[NEIGHBOR-DIS { GEOMETRIC *fact* | LINEAR | ISOVOLUME }]
[NEIGHBOR-RAD R_n]
[TABU-RAD R_t]
[PROMIS-RAD R_p]
[NIT-MAX-CONV Nit_{max}]
[TABU-LIST-LG { ALL $|Lg_t$ }]
[PROM-LIST-LG { ALL $|Lg_p$ }]
[GET-CURRENT [COMPLETE]]
[PUT-CURRENT]
[INITIALIZE]
[INIT-PRO-LIST]
[RESET-BEST]
[BEST-AS-CURR]
[NELDER-EPS ϵ_{ned}]

where

ISEED key word used to define the seed for random number generation *seed*.

seed integer value for the seed (default given by CLETIM).

NEIGHBOR-NB key word used to define the number of neighbors *ngh*.

ngh integer value for the number of neighbors (default 5).

NEIGHBOR-TYP key word used to specify the type of the neighborhood.

RECTANGLE key word used to specify that the neighborhood will be hyperrectangle crowns (default).

BALL key word used to specify that the neighborhood will be hypersphere crowns.

NEIGHBOR-DIS key word used to specify that the type of discretisation within the neighborhood.

GEOMETRIC key word used to specify that the radius of the crowns are given by a geometric serie. The radius are given by:

$$r_i = R_n \frac{1}{fact^{ngh-i}} \text{ with } i \in \{1, ngh\}$$

fact real number (> 1) for the geometric serie for the radius determination.

LINEAR key word used to specify that the radius of the crowns are given by a linear serie (default). The radius are given by:

$$r_i = R_n \frac{i}{ngh} \text{ with } i \in \{1, ngh\}$$

ISOVOLUME key word used to specify that the radius of the crowns are chosen to have a constant volume for all crowns. The radius are given by:

$$r_i = R_n \sqrt[nvar]{\frac{i}{ngh}} \text{ with } i \in \{1, ngh\}$$

NEIGHBOR-RAD key word used to set the radius $R_n$of the neighborhood.

R_n real number for neighborhood radius. This radius is a fraction of the total limits and then must be between 0.0 and 1.0.

TABU-RAD key word used to set the radius R_t of the hyperrectangle / ball around tabu values. All the points within this small domain are tabu as well.

R_t real number for tabu list radius. This radius is a fraction of the total limits and then must be between 0.0 and 1.0.

PROMIS-RAD key word used to set the radius R_p of the hyperrectangle / ball around tabu values. All the points within this small domain are tabu as well.

R_p real number for promising list radius. This radius is a fraction of the total limits and then must be between 0.0 and 1.0.

NIT-MAX-CONV key word used to specify the number Nit_{max} of required external iteration without improvement of the best solution ever found for global convergence achievement.

Nit_{max} integer value of required iterations for global convergence.

TABU-LIST-LG key word used to specify the maximum length of the tabu list.

PROM-LIST-LG key word used to specify the maximum length of the promising area list.

ALL key word used to specify the values entering in a list are kept until the end of the optimization procedure.

Lg_t integer value of maximum tabu list length.

Lg_p integer value of maximum promising area list length.

GET-CURRENT key word used to specify the decision variable set in *OPTIM* object will be stored as the current one in *TABUSH* object.

COMPLETE key word used to specify the objective function, the constraints and the penalty functions will also be stored with the current values.

PUT-CURRENT key word used to specify the current decision variable set in *TABU* object will be stored as the variable values in *OPTIM* object.

INITIALIZE key word used to initialize (chose a random value) the starting decision variables in the decision space.

INIT-PRO-LIST key word used to specify that the initial promising list is created.

RESET-BEST key word used to reset the best value of the tabu search objective function.

BEST-AS-CURR key word used to set the best value as the current value.

NELDER-EPS key word used to set the convergence limit for the Nelder-Mead simplex algorithm.

ϵ_{ned} real value for the Nelder-Mead convergence criterium (default 0.01).

Table 1.23: Structure **(nelder_data)**

[`FIND-NEW` $>>i_{worst}<<$]
[`COMPARE-NEW` $>>L_{exp}<<$ $>>L_{ctt}<<$ $>>L_{conv}<<$]
[`EXPAN-VLD` $>>L_{conv}<<$]
[`CONTRACTION` $>>L_{mctt}<<$ $>>i_{best}<<$ $>>L_{conv}<<$]

where

`FIND-NEW` key word used to find the worst point of the polytope for the Nelder-Mead simplex algorithm, and compute its reflected point.

i_{worst} integer value for the index of the worst point of the polytope.

`COMPARE-NEW` key word used to compare the reflected point of the worst point of the polytope with the other points. Next geometrical transformation is decided according to the results of the comparison.

L_{exp} logical value for the expansion move.

L_{ctt} logical value for the contraction move.

L_{conv} logical value for convergence.

`EXPAN-VLD` key word used to validate the expansion point (comparison of its results with the reflection point).

`CONTRACTION` key word used to compare the contraction point (comparison of its results with the reflection point). Next geometrical transformation is decided according to the results of the comparison.

L_{ctt} logical value for the multi-contraction move.

i_{best} integer value for the index of the best point of the polytope.

2 Output Data Treatment

In this section, input of output data treatment modules will be given.

2.1 The `ADDOBJ:` module

The `ADDOBJ:` module is used to perform the differences between two objects or to add two objects. For the MACROLIB and FLUX, this is possible only if they contain the same energy group and material mixture numbers.

The calling specifications are:

Table 1.24: Structure `ADDOBJ:`

MACNEW `:=` `ADDOBJ:` [*MACNEW*] *MACRO1 MACRO2* `::` **(addmac_data)**
FLUNEW `:=` `ADDOBJ:` [*FLUNEW*] *FLU1 FLU2* `::` **(addflu_data)**

where

MACNEW `character*12` name of the MACROLIB containing either the nuclear increments, from the calculation of *MACRO1 - MACRO2* or the sum of properties from *MACRO1* + *MACRO2*. Be aware the order of MACROLIB is important, even for addition option.

MACRO1 `character*12` name of a MACROLIB.

MACRO2 `character*12` name of a MACROLIB. When addition is performed, it must contain incremental nuclear properties.

(addmac) structure containing the data to module `ADDOBJ:` with the options for MACROLIB operations.

FLUNEW `character*12` name of the FLUX which will be the result of the addition or the subtraction of the two old one. This object has to be in create mode only.

FLU1 `character*12` name of the first FLUX.

FLU2 `character*12` name of the second FLUX.

(addflu) structure containing the data to module `ADDOBJ:` with the options for FLUX addition.

2.1.1 Data input for module `ADDOBJ:`

Table 1.25: Structure **(addmac_data)**

[`EDIT` *iprt*]
[`STEP` *ilev*]
[{ `ADD` [*frac1* [*frac2*]] | `SUB` }]

continued on next page

Structure **(addmac_data)**

continued from last page

```
[ FROM-MP ]
[ { DDIFF | NODIF } ]
;
```

where

`EDIT` key word used to set *iprt*.

iprt index used to control the printing. <= 2 minimum printing; > 3 macroscopic differences are printed.

`ADD` keyword to specify that the two objects will be added. If only *frac1* is specifeid, *MACRO1* + *frac1*.*MACRO2* will be performed using the options for the diffusions coefficients. If *frac1* and *frac1* are specified, *frac1*.*MACRO1* + *frac2*.*MACRO2* will be performed even for the diffusion coefficients.

frac1 first real that will multiply the value of the first object. (default 1.0)

frac2 second real that will multiply the value of the second object. (default 1.0)

`SUB` keyword to specify that the two objects will be substracted. This is the default option.

`FROM-MP` keyword to specify that the two objects have been partially calculated by different CPU. An addition will be performed to calculate the 'complete' object (*frac1* and *frac2* equal 1.0).

`STEP` key word used to set *ilev*.

ilev number of the perturbed level in MACROLIB. In case of substraction of two MACROLIB: If a single set of increments is stored, it must be equal to 1. This step is used to later compute perturbation system matrices. If this information is absent, incremental cross sections are stored on root directory. In case of addition of two MACROLIB: *ilev* specifies the perturbed level where informations is stored in the two initial MACROLIB. If both MACROLIB have a perturbed level, it must the same. The resulting properties will be stored on root directory.

`DDIFF` keyword to specify a correct treatment of diffusion coefficients. If `SUB` is specified, the resulting incremental diffusion coefficient will be:

$$\Delta D = \frac{1}{\frac{1}{D_1} - \frac{1}{D_2}}$$

where D_1 is taken from the first MACROLIB and D_2 from the second. If `ADD` is specified, the resulting diffusion coefficient will be:

$$D = \frac{1}{\frac{1}{D_1} + \frac{1}{\Delta D_2}}$$

where D_1 is taken from the first MACROLIB and ΔD_2 from the second.

`NODIF` keyword to specify that no addition or substraction of diffusion coefficients will be done. This is the default option.

Table 1.26: Structure **(addflu_data)**

```
[ EDIT iprt ]
{
[ { ADD [ frac1 frac2 ] | SUB } ] [ F-ADJOINT [ ADJ-ONLY ] ]
|
FROM-2-PARTS { FLUX | AFLUX |DFLUX iga1 |ADFLUX iga1 } frac1
......{ FLUX | AFLUX |DFLUX iga2 |ADFLUX iga2 } frac2
......IN { FLUX | AFLUX |DFLUX iga2 |ADFLUX iga2 }
} ;
```

where

`EDIT` key word used to set *iprt*.

iprt index used to control the printing. => 1 structure of the resulting object is printed;

`ADD` keyword to specify that the two objects will be added. This is the default option.

frac1 first real that will multiply the value of the first object. (default 1.0)

frac2 second real that will multiply the value of the second object. (default 1.0)

`SUB` keyword to specify that the two objects will be substracted.

`F-ADJOINT` keyword to specify that the same numerical operation will performed on flux and adjoint flux.

`ADJ-ONLY` keyword to specify that the numerical operation will be performed on the adjoint flux only.

`FROM-2-PARTS` keyword to specify that the two objects contain one part of the result flux. An addition will be performed to calculate the 'complete' object. This option can not be done with other options (except `EDIT`) in the same call of the module.

`FLUX` keyword to specify that the flux will be used or the result of the addition.

`AFLUX` keyword to specify that the adjoint flux will be used or the result of the addition.

`DFLUX` keyword to specify that the explicit generalized adjoint will be used or the result of the addition.

`ADFLUX` keyword to specify that the implicit generalized adjoint will be used or the result of the addition.

iga1 number of the first generalized adjoint if applicable.

iga2 number of the second generalized adjoint if applicable.

`IN` keyword to specify that the type of flux for the result of the addition.

iga3 number of the third generalized adjoint if applicable.

2.2 The `MATLAB:` module

The `MATLAB:` module is used to create an ASCII file executable by MATLAB. Two options are available. First one is used to create a file to draw the gradiants of functions calulated in a optimisation problem. Second option allows to draw maps of the flux distribution.

The calling specifications are:

Table 1.27: Structure `MATLAB:`

{ *ASCII-MAT* `:=` `MATLAB:` *OPTIM* `::` **(descmatlgrd)** |
ASCII-MAT `:=` `MATLAB:` *FLUX TRACK INDEX GEOM* `::` **(descmatlflu)** }

where

ASCII-MAT `character*12` name of the ASCII file executable by MATLAB.

OPTIM `character*12` name of the OPTIMISATION linked list or XSM file containing the gradiants and the stored perturbated values of the functions. Such file is obtained using the module `GPTVRF:`.

(descmatlgrd) structure containing the data input to module `MATLAB:` for gradiants ploting.

FLUX `character*12` name of the FLUX linked list or XSM file containing the flux or adjoint or generalized adjoints or harmonics to be mapped.

TRACK `character*12` name of the TRACK linked list or XSM file containing the tracking datas (TRIVAA is the only type of tracking compatible).

INDEX `character*12` name of the INDEX linked list or XSM file containing the index datas.

GEOM `character*12` name of the GEOM linked list or XSM file containing the geometry description.

(descmatlflu) structure containing the data input to module `MATLAB:` for flux distribution's mapping.

2.2.1 Data input for module `MATLAB:`

Table 1.28: Structure **(descmatlgrd)**

[`EDIT` *iprint*]
`OPT-GRAD-VRF`
;

where

`EDIT` key word used to set *iprint*.

iprint index used to control the printing in module XSFUEL:. =0 for no print (default value); =1 for minimum printing; larger values produce increasing amounts of output.

OPT-GRAD-VRF key word used to select the gradiants verification and ploting option and produce the *ASCII-MAT* file. An exemple of the results is presented on the following figure.

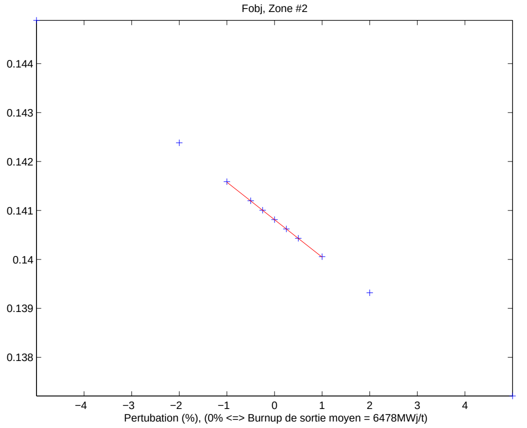


Table 1.29: Structure **(descmatlflu)**

```
[ EDIT iprint ]
MAP-FLUX {FLUX | ADJOINT | GPT-FLU isrc | GPT-ADJOINT isrc | HARMONIC ihrm }
{ NO-GRID | CENTER-GRID | CHANEL-GRID }
;
```

where

EDIT key word used to set *iprint*.

iprint index used to control the printing in module XSFUEL:. =0 for no print (default value); =1 for minimum printing; larger values produce increasing amounts of output.

MAP-FLUX key word used to select the flux distribution mapping option and produce the *ASCII-MAT* file.

FLUX key word used to specify that the distribution of the flux will be drawn.

ADJOINT key word used to specify that the distribution of the adjoint will be drawn.

GPT-FLU key word used to specify that the distribution of the explicit generalized adjoint corresponding to the source number *iscr* will be drawn.

GPT-ADJOINT key word used to specify that the distribution of the implicit generalized adjoint corresponding to the source number *iscr* will be drawn.

iscr identifier for the source number corresponding to generalized adjoint to be drawn.

HARMONIC key word used to specify that the distribution of the flux corresponding to the harmonic number *ihrm* will be drawn.

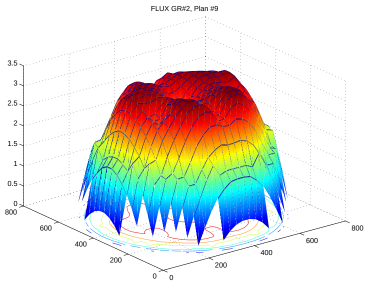
ihrm identifier for the harmonic number corresponding to the flux to be drawn.

`NO-GRID` key word used to specify that no grid will be add on the map of the flux distribution. An exemple of the results is presented on the figure a.

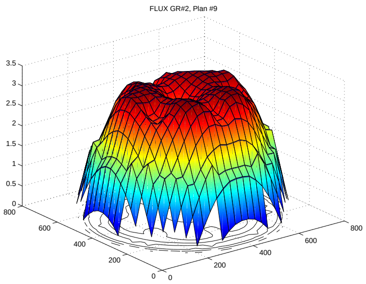
`CENTER-GRID` key word used to specify that a grid will be add on the map of the flux distribution. The nodes of the grid correspond to the center of the volumes where the flux are calculated. An exemple of the results is presented on the figure b.

`CHANEL-GRID` key word used to specify that a grid will be add on the map of the flux distribution. The squares of the grid correspond to the limits of the channels. An exemple of the results is presented on the figure c.

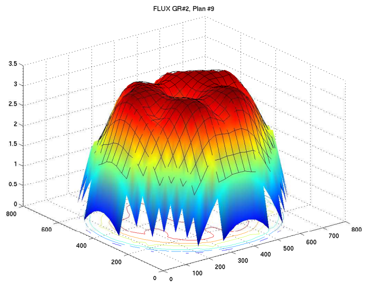
(a)



(b)



(c)



2.3 The `GPTVRF:` module

The GPTVRF: module is used to verify the computation of the gradients with classical and generalized perturbation theories. The analytical gradients calculated with the modules PERTUR: and GPTGRD: are compared with numerical gradients calculated with the Ceshino method using several values of the functionals for pertubated values of the decision variables.

The calling specifications are:

Table 1.30: Structure `GPTVRF:`

OPTIMV `:=` `GPTVRF:` [*OPTIMV*] *OPTIM* `::` **(gptvrf_data)**

where

OPTIMV `character*12` name of the OPTIMIZE containing the pertubated and unpertubated values of the decision variables and of the functionals. If *OPTIMV* is the same as *OPTIM*, then the perturbated datas will be stored in the same OPTIMIZE object.

OPTIM `character*12` name of the OPTIMIZE containing the curent values of the decision variables and of the functionals.

(gptvrf_data) structure containing the data to module `GPTVRF:`.

2.3.1 Data input for module `GPTVRF:`

Table 1.31: Structure `gptvrf_data`

[`EDIT` *iprint*]
[`SAVE-PERT` i_{var} ϵ]
[`SAVE-GPT` {`ABS` | `REL` ϵ_{GPT}}]
[`COMPARE` { i_{var1} i_{var2} | `ALL` } { i_{fct1} i_{fct2} | `ALL` }]
;

where

`EDIT` key word used to set *iprint*.

iprint index used to control the printing in module.

`SAVE-PERT` key word used to store the current values of the functionals for the corresponding perturbation ϵ of the decision variable i_{var}.

i_{var} number of the perturbated decision variable.

ϵ amount of the pertubation (ex: ϵ=0.01 corresponds to 1% of perturbation).

SAVE-GPT key word used to specify that the analytical gradient will be stored.

ABS key word used to specify that the stored analytical gradient correspond to a absolute perturbation or that the derivative are already calculated.

REL key word used to specify that the stored analytical gradient will be devided by the corresponding amount of perturbation to obtain the relative gradient (the derivative).

ϵ_{GPT} amount of the pertubation (ex: ϵ_{GPT}=0.01 corresponds to 1% of perturbation).

COMPARE key word used to specify that the numerical gradient will be computed and compared to the analytical gradients.

i_{var1} number of the first decision variable for which the comparason will be done.

i_{var2} number of the last decision variable for which the comparason will be done.

i_{fct1} number of the first functional for which the comparason will be done.

i_{fct2} number of the last functional for which the comparason will be done.

ALL key word used to specify that the comparason will be done for all decision variables and/or all functionals.

Chapter 2

OPTEX STRUCTURES

1 Contents of a /tabu/ data structure

The /tabu/ specification is used to store the decision variable set used for a tabu search optimization method. The different options of this method are also stored in the /tabu/ data structure.

The signature variable for this data structure must be SIGNA=`L_TABU␣␣␣␣␣␣`. The dimensioning parameters for this data structure, which are stored in the state vector $\mathcal{S}_i^t$, represents:

- The number of neighbors $N_{eig} = \mathcal{S}_1^t$.
- The type of neighborhood $\mathcal{S}_2^t$ where

$$\mathcal{S}_3^t = \begin{cases} 1 & \text{hyperrectangle} \\ 2 & \text{ball} \end{cases}$$

- The type of discretisation of the neigborhood. $\mathcal{S}_3^t$ where

$$\mathcal{S}_3^t = \begin{cases} 1 & \text{geometric} \\ 2 & \text{linear} \\ 3 & \text{isovolume} \end{cases}$$

- The test for external convergence $Nit_{max} = \mathcal{S}_4^t$.
- The current number of iterations without improvement of the best value $Nit = S_5^t$.
- The tabu list maximum size $L_t = \mathcal{S}_6^t$.
- The promising area list maximum size $L_p = \mathcal{S}_7^t$.
- The tabu list current size $L_t^c = \mathcal{S}_8^t$.
- The promising list current size $L_p^c = \mathcal{S}_9^t$.
- The number of decision variables $N_{var} = \mathcal{S}_{10}^t$.
- The number of constraints $N_{cst} = \mathcal{S}_{11}^t$.
- The first number for the random point generation algorithm $\mathcal{S}_{39}^t$ (=TIME()).
- The number of generated random numbers $\mathcal{S}_{40}^t$.

Table 2.1: Main records and sub-directories in /tabu/

Name	Type	Condition	UnitsComment
`SIGNATURE␣␣␣`	C∗12		Signature of the data structure (SIGNA)
`STATE-VECTOR`	I(40)		Vector describing the various parameters associated with this data structure $\mathcal{S}_i^t$ and the tabu search optimization integer options

continued on next page

Main records and sub-directories in /tabu/ continued from last page

Name	Type	Condition	Units	Comment
`TABU-PARAM-R`	R(40)			Vector containing various tabu search optimization real options and several current values.
`CURRENT-VAL␣`	R($N_{var} + N_{cst} + 2$)			The values of the current decision variables, the corresponding functionals and the tabu function
`BEST-VAL␣␣␣␣`	R($N_{var} + N_{cst} + 2$)			The values of the decision variables, the corresponding functionals and the tabu function for the best point ever found
`NEIGHBORHOOD`	Dir			Directory containing the neighbor decision variable set.
`TABU-LIST␣␣␣`	Dir			Directory containing the tabu decision variable set.
`PROMISE-LIST`	Dir			Directory containing the decision variable set corresponding to promising areas.
`NELDER-MEAD␣`	Dir			Directory containing the decision variable set corresponding to the polytope for Nelder-Mead Simplex method.

Notes related with the different limits and values for the iterative calculations of the optimization problem:

1st	R_n	neighborhood radius (fraction $\in [0, 1]$ of the total decision space). (default: 0.01)
2nd	R_t	radius around tabu list decision variable set within points are also tabu (fraction $\in [0, 1]$ of the total decision space). (default: 0.001)
3rd	R_p	radius around promising area list decision variable set within points are also tabu (fraction $\in [0, 1]$ of the total decision space). (default: 0.002)
4th	$fact$	factor in case of a geometric discretisation of the neighborhood. (default: 2)
5th		best tabu search function value ever achieved.

The other value of the record are not used and set to 0.0.

1.1 The sub-directories in /tabu/

The sub-directories are NEIGHBORHOOD, TABU-LIST, PROMISE-LIST and NELDER-MEAD. They all have two main records in common. NELDER-MEAD has several additional records. Those two types of records are presented in the two following tables.

Table 2.2: Main records in sub-directories

Name	Type	Condition	Units	Comment
`ITEM␣␣␣␣␣␣␣␣`	R()			List of decision variable set, its corresponding functional and tabu search function values.
`TABU-F-VAL␣␣`	R(N_{eig})			The tabu search function values of the neighbors.

The length of the ITEM list is given by:

- N_{eig} for NEIGHBORHOOD directory
- $\min\{L_t^c, L_t\}$ for TABU-LIST directory
- $\min\{L_p^c, L_p\}$ for PROMISE-LIST directory
- $N_{var} + 1$ for NELDER-MEAD directory

Table 2.3: Additional records in NELDER-MEAD directory

Name	Type	Condition	Units	Comment
`IWORST␣␣␣␣␣␣`	I(1)			Index corresponding to the summit of the polytope with the worst value of tabu search fonction.
`IWORST2␣␣␣␣␣`	I(1)			Index corresponding to the summit of the polytope with the second worst value of tabu search fonction.
`IBEST␣␣␣␣␣␣␣`	I(1)			Index corresponding to the summit of the polytope with the worst value of tabu search fonction.
`XAVERG␣␣␣␣␣␣`	R(N_{var})			The average values of the decision variables for all the points of the polytope.
`REFLECT-VAL␣`	R($N_{var} + N_{cst} + 2$)			The reflection of the worst point of the polytope with the centroid given by XAVERG. It also includes the corresponding objective function, constraints and tabu search values after their evaluation.
`EXPAN-VLD␣␣␣`	R($N_{var} + N_{cst} + 2$)			The reflection of the centroid of the polytope given by XAVERG with the worst point. It also includes the corresponding objective function, constraints and tabu search values after their evaluation.

2 Contents of a /optimize/ data structure

The /optimize/ specification is used to store the optimization variables and functions values and definitions, limits and options. It is also used in a particular case with the module `GPTVRF:` to store the functions for many perturbated values of the decision variables and the gradients calculated numerically and analytically.

In any case, the signature variable for this data structure must be `SIGNA=L_OPTIMIZE␣␣`. The dimensioning parameters for this data structure, which are stored in the state vector $\mathcal{S}^o_i$, represents:

- The number of decision variables $N_{var} = \mathcal{S}^o_1$.
- The number of constraints $N_{cst} = \mathcal{S}^o_2$.
- The type of optimization. $\mathcal{S}^o_3$ where

$$\mathcal{S}^o_3 = \begin{cases} 1 & \text{minimization} \\ -1 & \text{maximization} \end{cases}$$

- The test for external convergence $\mathcal{S}^o_4$. where

$$\mathcal{S}^o_4 = \begin{cases} 0 & \text{not converged} \\ 1 & \text{converged} \end{cases}$$

- The number of external iterations S^o_5.
- The type of reduction for external step $\mathcal{S}^o_6$. where

$$\mathcal{S}^o_6 = \begin{cases} 1 & \text{half} \\ 2 & \text{parabolic} \end{cases}$$

- The number of inner iterations S^o_7.
- The number of outer iterations S^o_8.
- The resolution's method for the linearized problem $\mathcal{S}^o_9$. where

$$\mathcal{S}^o_9 = \begin{cases} 1 & \text{SIMPLEX/LEMKE} \\ 2 & \text{LEMKE/LEMKE} \\ 3 & \text{MAP} \\ 4 & \text{Augmented Lagragian} \\ 5 & \text{Penalty Method} \end{cases}$$

- The type of perturbation for the decision variables S^o_{10}. where

$$\mathcal{S}^o_{10} = \begin{cases} 1 & \text{epsilon} \\ 2 & \text{previous} \end{cases}$$

- The type of fuel cost definition S^o_{11}. where

$$\mathcal{S}^o_{11} = \begin{cases} 1 & \text{dependent of the enrichment} \\ 2 & \text{fixed} \end{cases}$$

- The test for a realistic decision vector S^o_{12}. where

$$\mathcal{S}^o_{12} = \begin{cases} 0 & \text{do not respect all constraints} \\ 1 & \text{do respect all constraints} \end{cases}$$

- A flag for unsuccessful resolution in QLP S^o_{13}. where

$$\mathcal{S}^o_{13} = \begin{cases} 0 & \text{successful at last iteration} \\ \geq 1 & \text{number of iteration with unsuccessful resolution} \end{cases}$$

- The number of regions $N_r = \mathcal{S}^o_{16}$.
- The number of channels in the core $N_{ch} = \mathcal{S}^o_{17}$.
- The number of bundles per channel $N_k = \mathcal{S}^o_{18}$.
- The number of unknowns per energy group $N_u = \mathcal{S}^o_{19}$.
- The number of energy groups $G = \mathcal{S}^o_{20}$.

Table 2.4: Main records and sub-directories in /optimize/

Name	Type	Condition	Units	Comment
`SIGNATURE␣␣␣`	C∗12			Signature of the data structure (SIGNA)
`STATE-VECTOR`	I(40)			Vector describing the various parameters associated with this data structure $\mathcal{S}^o_i$
`VAR-VALUE␣␣␣`	R(N_{var})			The values of the decision variables
`VAR-TYPE␣␣␣␣`	R(N_{var})			The type of the decision variables Var_{type}.
{*var-zone*}	I($N_{z,v}$)			The definition of the zone where i-th decision variable have an influence on the material properties.
`VAR-MAX-VAL␣`	R(N_{var})			The maximum values of the decision variables can be.
`VAR-MIN-VAL␣`	R(N_{var})			The minimum values of the decision variables can be.
`VAR-WEIGHT␣␣`	R(N_{var})			The weight of the decision variables w_i in the quadratic constraint.
`CST-VOL-TYPE`	I(N_{cst})			Record containing the type for the zone where the constraints apply.
`CST-OBJ␣␣␣␣␣`	R(N_{cst})			The limit value of the contraints. The units depends with the type of the constraint type.
`CST-TYPE␣␣␣␣`	I(N_{cst})			The type of the contraints: =-1 for ≥; =0 for =; =1 for ≤.
{`cst-zone`}	I($N_{z,c}$)			The definition of the zone where the constraint j apply.
`CST-WEIGHT␣␣`	R(N_{cst})			The weight of the constraint η_j and γ_j for the duals and meta-heuristic methods.

continued on next page

Main records and sub-directories in /optimize/ continued from last page

Name	Type	Condition	Units	Comment
`FOBJ-CST-VAL`	R($N_{cst}+1$)			The actual values of the objective function (first value) and the contraints (the following values). The number of the constraints are assigned in the order they have been defined.
`OPT-PARAM-R␣`	R(40)			The different limits and values for the iterative calculations of the optimization problem.
`FUNC-ZON-VOL`	R($N_{cst}+1$)			The first value is the volume where the objective function applies and the followingones correspond to the volumes where the constraints apllie.
`BKP-PERT-XS␣`	I(13)			The indexes of the perturbated material properties to backup.
`FUEL-COST␣␣␣`	R(dependant)			The fixed fuel cost of each enrichment zone if S^o_{11}=-1.0, or the parameters to calculate the enrichment dependant cost if S^o_{11}=-2.0.
`FUEL-C-DIST␣`	R(N_u)		$\$/kg$	The fuel cost distribution corresponding to each point of the neutrons flux unknowns.
`BURN-C-DIST␣`	R(N_u)		MWd/t	The average exit burnup distribution corresponding to each point of the neutrons flux unknowns.
`D-LAMBDA␣␣␣␣`	R(N_{var})			The gradients or the absolute variation depending of the keyword used in the `PERTUR:` module of the eigenvalue with the decision variables.
`D-LAMBDA-V␣␣`	R(N_{var})			The gradients or the absolute variation depending of the keyword used in the `PERTUR:` module of the eigenvalue for the coolant voided reactor with the decision variables.
`GRADIENT␣␣␣␣`	R($N_{var}, N_{cst}+1$)			The gradients of the objective function and the constraints. The gradients of the objective for all the decision variables are in first position, then follow the gradients of the constraints.
`OLD-VALUE␣␣␣`	Dir			Directory containing differents informations of the previous iterations. the values of the decision variables, the objective function, the constraints and the gradients of these functions for the previous external iteration. This repertory will be created by the module `PLQ:` unless it is specified to not do.

continued on next page

Main records and sub-directories in /optimize/

continued from last page

Name	Type	Condition	Units	Comment
`BKP-VALUE␣␣␣`	Dir			Directory containing a backup of the values of the decision variables, the objective function, the constraints and the gradients of these functions for this external iteration. This repertory will be created if the interpolation of the objective function in the middle point is calculated.
`{/stepdir/}`	Dir			Directory containing the perturbated properties and the $A_p\phi$ and $B_p\phi$ vectors.

Notes related with the decision variables definition:
The type of the decision variable are given by Var_{type} which is defined by:

$$Var_{type} = i_{type} * 1000 + i_{zone}$$

where

$$i_{type} = \begin{cases} 1 & \text{average exit burnup} \\ 2 & \text{enrichment} \\ 3 & \text{number of bundles shift} \\ 4 & \text{device thickness} \end{cases}$$

i_{zone} for the number of the corresponding zone in the `L_MAP` for the decision variable.

The zones definition information records {var-zone} will be composed using the following FORTRAN instructions

$$\texttt{WRITE}(\texttt{var}-\texttt{zone},'(\texttt{A8},\texttt{I4})')\ '\texttt{VAR}-\texttt{ZONE}',i$$

for $1 \leq i \leq N_{var}$. The size $N_{z,v}$ of the records depends of the type of the decision variable. For average exit burnup, enrichment and number of bundles shift, $N_{z,v} = N_{ch}$. For device thickness, $N_{z,v} = N_{ch} * N_k$.

Notes related with the constraints definition:
The type of each constraint zones is defined by the following index :

$$j_{type} = \begin{cases} 1 & \text{one value for the whole reactor (ex: } k_{eff}\text{)} \\ 2 & \text{one value for all channels (ex: total power)} \\ 3 & \text{one value for one channel (ex: channel power)} \\ 4 & \text{one value for one bundle (ex: bundle power)} \\ 5 & \text{one value for one zone defined by many bundles} \\ 6 & \text{void reactivity} \\ 7 & \text{analytic function} \end{cases}$$

The zones definition information records {cst-zone} will be composed using the following FORTRAN instructions

$$\texttt{WRITE}(\texttt{cst}-\texttt{zone},'(\texttt{A8},\texttt{I4})')\ '\texttt{CST}-\texttt{ZONE}',j$$

for $1 \leq j \leq N_{cst}$. The size $N_{z,c}$ of the records depends of the type of the constraint.

$$\mathcal{N}_{z,c} = \begin{cases} 1 & \text{if } j_{type} = 3\text{, the record contains the number of the channel} \\ 2 & \text{if } j_{type} = 4\text{, the record contains the number of the channel and the plan} \\ N_{ch} \times N_k & \text{if } j_{type} = 5\text{, the record contains one value for each bundle : =0 out of the} \\ & \text{zone } j\text{, =1 in the zone } j. \end{cases}$$

If $j_{type} = 1$ or 2, no record is necessary to define the volume because it is implicit with the type of the constraint (reactor or core respectively).

Notes related with the different limits and values for the iterative calculations of the optimization problem:

1st	S	external step limit. It is used for the quadratic constraint if applicable. (default: 1.0)
2nd	δ	internal step limit for MAP method. (default: 0.1)
3rd	ε_{ext}	limit for external convergence. (default: 10^{-3})
4th	ε_{int}	limit for internal convergence. (default: 10^{-3})
5th	ε_4	limit for convergence for the quadratic constraint.
6th		expected value for the objective function calculated with the linearized problem.
7th	$\varepsilon_{\Delta X}$	relative variation of the decision variables for the 'epsilon' method of perturbation. (default: 10^{-2})
8th	m	Exponent for a objective function : $\min \sum_j (P_j - \overline{P}^{2m}$. (default: 8.0)
9th	α	multiplication factor for constraint weight if $S_9^o = 4$ or 5. (default: 2.0)
10th	ε_α	limit of the error of the constraints for which function penalty weight have to be adjust.
11th	V_{core}	Volume of the core.
12th	V_{reactor}	Volume of the reactor.

The other value of the record are not used and set to 0.0.

Notes related with the perturbated material properties:

In the record BKP-PERT-XS-, 1 and 0 means that the corresponding perturbated material properties will be backuped or not respectively. The available perturbated material properties are :

1st	DIFFX	diffusion coefficient along x axes (D_x)
2nd	DIFFY	diffusion coefficient along y axes (D_y)
3rd	DIFFZ	diffusion coefficient along z axes (D_z)
4th	TOTAL	total cross-section (Σ_t)
5th	SCAT0	compressed isotropic component of the scattering matrix
6th	SIGW0	isotropic component of the within group of the scattering of the scattering cross-section
7th	SCAT1	compressed linearly anisotropic component of the scattering matrix
8th	SIGW1	linearly anisotropic component of the within group of the scattering of the scattering cross-section
9th	NFTOT	fission cross-section (Σ_f)
10th	NUSIGF	product of the fission cross-section (Σ_f) and the steady-state number of neutron produced per fission (ν^{ss})
11th	H-FACTORS	energy production coefficient (H)
12th	CHI	the steady-state energy spectrum of the neutron emitted by fission
13th	FIXE	fixed sources

Notes related with the fuel cost:

If the cost of the fuel is fixed, the dimension of the record is given by the number of enrichment zones. Otherwise the record contains the 7 parameters necessary to calculate the fuel cost, which are :

1st concentration in ${}^{235}U$ of the waste uranium (ε_w).
2nd cost of natural uranium ($C_{NU}\$/kg$).
3rd cost of a separation work unit ($C_S\$/SWU$).
4th cost of the bundle fabrication ($C_{FAB}\$/kg$).
5th interest rate ($int\%/day$).
6th time to obtain uranium ($days$).
7th time for enrichment ($days$).

The fuel cost is given by the equation:

Notes related with the perturbated materials properties directory:
The directories {stepdir} will be composed using the following FORTRAN instructions

WRITE(stepdir,′(A4, I8)′) ′STEP′, i

for $1 \leq i \leq N_{var}$. Each directory contains also the result of the multiplication of the perturbated system matrix and the flux. $i = 0$ is used to stored the result of the unperturbated system matrix and the flux.

2.1 The sub-directory /OLD-VALUE/ in /optimize/

Table 2.5: Main records and sub-directories in //OLD-VALUE//

Name	Type	Condition	Units	Comment
`VAR-VALUE␣␣␣`	R(N_{var})			The values of the decision variables of the last valid iteration.
`FOBJ-CST-VAL`	R($N_{cst} + 1$)			The values of the objective function and the contraints of the last valid iteration.
`GRADIENT␣␣␣␣`	R($N_{var}, N_{cst} + 1$)			The gradients of the objective function and the constraints of the last valid iteration.
`VAR-VALUE2␣␣`	R(N_{var})			The values of the decision variables of the second-last valid iteration.
`BEST-VAR␣␣␣␣`	R(N_{var})			The values of the decision variables corresponding to the best valid solution ever found.
`BEST-FCT␣␣␣␣`	R(1)			The value of the objective function corresponding to the best valid solution ever found.

2.2 The sub-directory /stepdir/ in /optimize/

Table 2.6: Main records and sub-directories in //stepdir//

Name	Type	Condition	Units	Comment
{`Aphi`}	R(N_u)			The group-dependent vectors representing the multiplication of a system matrix A and the unperturbated flux.
{`Bphi`}	R(N_u)			The group-dependent vectors representing the multiplication of a system matrix B and the unperturbated flux.

continued on next page

Main records and sub-directories in //stepdir// continued from last page

Name	Type	Condition	Units	Comment
{/grpdir/}	Dir			The group-dependent directory containing the perturbated properties. The content is the same as for the /macrolib/ but limited to the properties to be saved only (see. BKP-PERT-XS-).

The records {Aphi} and {Bphi} will be composed using the following FORTRAN instructions:

$$\texttt{WRITE}(\mathsf{Aphi},'\texttt{(A5,I7)}')\ '\texttt{A*PHI}', g$$

and

$$\texttt{WRITE}(\mathsf{Bphi},'\texttt{(A5,I7)}')\ '\texttt{B*PHI}', g$$

respectively where g represent the energy group.

2.3 Contents of a /optimize/ data structure for module GPTVRF:

When the module GPTVRF: is used a new /optimize/ data structure can be created. This new structure contains a copy of the STATE-VECTOR record of the /optimize/ data structure used on the RHS of the module. Then only the bare minimum necessary datas for gradients verification will be stored. This structure can also be deleted when a new point of the optimization procedure is calculated. If the same /optimize/ data structure is used in the module GPTVRF: unnecessary data will be stored for the rest of the optimization calculations. This is why we recommend to use a new /optimize/ data structure on the LHS. In both cases, the /optimize/ data structure will contain the data described in the following table.

Table 2.7: /optimize/ in the particular case of module GPTVRF:

Name	Type	Condition	Units	Comment
SIGNATURE␣␣␣	C∗12			Signature of the data structure (SIGNA)
STATE-VECTOR	I(40)			Vector describing the various parameters associated with this data structure $\mathcal{S}_i^o$, same one or a copy of the previous description.
VAR-VALUE-RF	R(N_{var})			The references values of the decision variables for which the gradients are numerically and analytically calculated.
GRADIENT-GPT	R($N_{var}, N_{cst}+1$)			The gradients calculated analytically. This record is simply a copy of the record GRADIENT of the /optimize/ data structure on the RHS.

continued on next page

/optimize/ in the particular case of module GPTVRF: continued from last page

Name	Type	Condition	Units	Comment
GRADIENT-EXP	R($N_{var}, N_{cst}+1$)			The gradients calculated numerically of the objective function and the constraints. The gradients are stored the same way as for the record GRADIENT of the first description.
/varpertdir/	Dir			Directory containing the perturbation values and the corresponding values of the functions for a decision variable i.

Notes related with the perturbated decision variables directory:
The directories {varpertdir} will be composed using the following FORTRAN instructions

$\texttt{WRITE}(\mathsf{varpertdir},'(\texttt{A8},\texttt{I4})')\ '\texttt{VAR}-\texttt{PERT}', i$

for $1 \leq i \leq N_{var}$. Each directory contains also the records described in the following table.

Table 2.8: The sub-directory /varpertdir/ in /optimize/

Name	Type	Condition	Units	Comment
EPSILON␣␣␣␣␣	R($N_{\varepsilon,i}$)			The value of the perturbation of the decision variable i.
{fvalpert}	R($N_{cst}+1$)			The values of the objective function and constraints for the perturbation p of the decision variable i.

The number of perturbation $N_{\varepsilon,i}$ can include the unperturbated case ($\varepsilon = 0.0$) and can be different for each decision variable i. Moreover, the values of the perturbation do not have to be in increasing or decreasing order.

The records {fvalpert} will be composed using the following FORTRAN instructions

$\texttt{WRITE}(\mathsf{fvalpert},'(\texttt{A9},\texttt{I3})')\ '\texttt{F}-\texttt{C}-\texttt{VAL}-\texttt{P}', p$

for $1 \leq p \leq N_{\varepsilon,i}$.

Index