**A guide for using user defined constraint feature of the VLBI Analysis
software system Calc/Solve**

Leonid Petrov

Abstract:

*This document describes the way how to imposed user defined constraints on parameters in
Calc/Solve*

Table of contents:

1   Overview

User program for imposing constraints should create equations of
constraints, right-hand side of constraints (usually zero), reciprocal weights
of constraints and off-diagonal terms of constraints (may be zero).

User constraint program is called in the contest of Solve process.
It should read some Solve variable and write equations of constraints in the
special format which is understood by Solve. Of course, in a principle,
a user is in a position to write in this file directly, although it is
assumed that a user uses exactly the same subroutines for defining and
applying constraints as for Solve built-in constraints.

First executable statement of a user program is PRE_PROG(), the last
executable statement is END_PROG()

Usually at the beginning a user program need to read Solve scratch files:

```
    CALL USE_GLBFIL   ( 'OR' )
    CALL USE_GLBFIL_2 ( 'R'  )
    CALL USE_GLBFIL_4 ( 'RC' )
    CALL USE_PARFIL   ( 'ORC' )  ! Reading  prfil.i
    CALL USE_COMMON   ( 'ORC' )  ! Reading  socom.i
    CALL SOCOM_EXT
```

Constraints are put in the object of derived type CNSTROBJ. Type definition
is defined in include file cnstr.i

A user is supposed to do the following operations under the object CNSTROBJ:

1) Initialization. This routine is called only once at the beginning.

Routine INIT_CNS   ($MK4_ROOT/progs/solve/proc/add_cns.f)

2) Define the name of the constraint, units, right-hand side of the constraint
   equation, flag local/global. This routine is called for each new constraint
   equation.

   Routine ADDCNS_NAM   ($MK4_ROOT/progs/solve/proc/add_cns.f)

3) Set flag that this constraint is a user-defined. It is done in order to
   distinguish user-defied constraints from built-in constraint and prevent
   traps of internal control. This routine is called for each new constraint
   equation.

   Routine SET_USER_CNS   ($MK4_ROOT/progs/solve/proc/add_cns.f)

4) Insert the coefficient of constraint equation. This routine is called for
   each non-zero element of constraint equation.

   Routine ADDCNS_EQU   ($MK4_ROOT/progs/solve/proc/add_cns.f)

5) Write down constants of CNSTROBJ in disk. Routine for that is called at
   after all constraints are defined.

   Routine WRITE_CNSTR   ($MK4_ROOT/progs/solve/proc/io_cnstr.f)

   User constraints can be imposed on both global and local parameters.
Argument CNI_MODE of WRITE_CNSTR specifies on which mode constraints are
to be imposed: CNI__UGL for global parameters and CNI__ULC for local
parameters.

2 Restrictions

   1. A user should avoid using abbreviations of Solve built-in constraints
unless he or she wants to interfere Solve built-in constraints and really
understands what he or she is doing.

   2. Name of units of user constraint equations are used directly by procedure
write_sinex without transformation. Therefore, if a user intends to write
listings of his solution in Sinex format, he should use internal units which
Solve uses for the estimated parameters.

3 Example of user constraint program

   An example can be find in
       $MK4_ROOT/example/user_cons_example.f
       $MK4_ROOT/example/user_cons_example.mak

```
For compiling and linking execute a command:
make -f $MK4_ROOT/example/user_cons_example.mak
```

Questions and comments about this guide should be sent to:

Leonid Petrov (sgdass@lists.nasa.gov)

Last update: 2002.10.04