4 PROGRAM GUIDE

Variables in *UDEC* are stored in "linked lists," which consist of addresses (the first memory location for a particular item – e.g., block or contact) and offsets (which prescribe the memory location of the variable relative to the address). The program guide contains the contents of these linked-list groups in the data structure. Figures 4.1 through 4.4 schematically show the linkage of these various groups, and should assist the user in following through the data groups and parameters. Figure 4.1 shows the linked-list arrangement of the main data arrays. Figures 4.2, 4.3 and 4.4 illustrate the conventions for linkages in the block data, domain data and contact data arrays, respectively.

4.1 Linked-List Addresses and Offsets

Addresses for data objects can be found using the **PRINT** command. In most cases, the first integer given is the address. Commands in *UDEC* allow commonly used real variables to be tracked (using the **HISTORY** command) or changed (using the **CHANGE** or **RESET** command). It is also possible to access variables via *FISH* to track or change less commonly used variables, or to define a dependency of one variable on another (e.g., shear displacement-dependent joint friction; see Section 3.7.3 in the **User's Guide** for an example).

As mentioned in Section 2.5.6, *FISH* programs have access to some of *UDEC*'s linked-list data structures. The global indices that point to these data structures are provided as *FISH* scalar variables (see Section 2.5.1). The various data blocks, and the offsets of items within the blocks, are contained in a series of files supplied with *UDEC*. These files have the extension ".FIN" (for **F**ish **IN**clude file); they provide symbolic names for offsets and current numerical values (which may change in future versions of *UDEC*).* The ".FIN" files serve two purposes: first, they document the meanings of the various data items; second, the files may be **CALL**ed from a data file – they automatically execute and define appropriate symbols. The symbols are all preceded by the **$** sign, so that they are invisible to the casual user who gives a **PRINT fish** command. The *FISH* programmer may simply use numbers or the symbols provided in the ".FIN" files for offsets. It is better to specify offsets in symbolic form because the resulting *FISH* program will work correctly with future versions of *UDEC*, in which offsets and block sizes may be different. (Itasca will make every effort to retain the same symbolic names in future versions of *UDEC*.)

Access to the data structure is via the **fmem** and **imem** functions, as described in Section 2.5.6. Be sure to note whether the variable in the ".FIN" file is of floating-point or integer type before using these functions. Also note that uncontrolled alterations of memory contents can crash the computer. So, any operations that alter memory contents should be undertaken with care.**

---

* The ".FIN" files are contained in the "\Datafiles\Fish\FIN" directory.

** A note to experienced *UDEC* users: The old commands **RSET** and **ISET** are no longer supported. The indices used with those commands *are not the same* as those used with *FISH*. This documentation provides the *FISH* indices only. You are strongly urged to change any data files using these commands to the *FISH* equivalent for use with this version of *UDEC*.

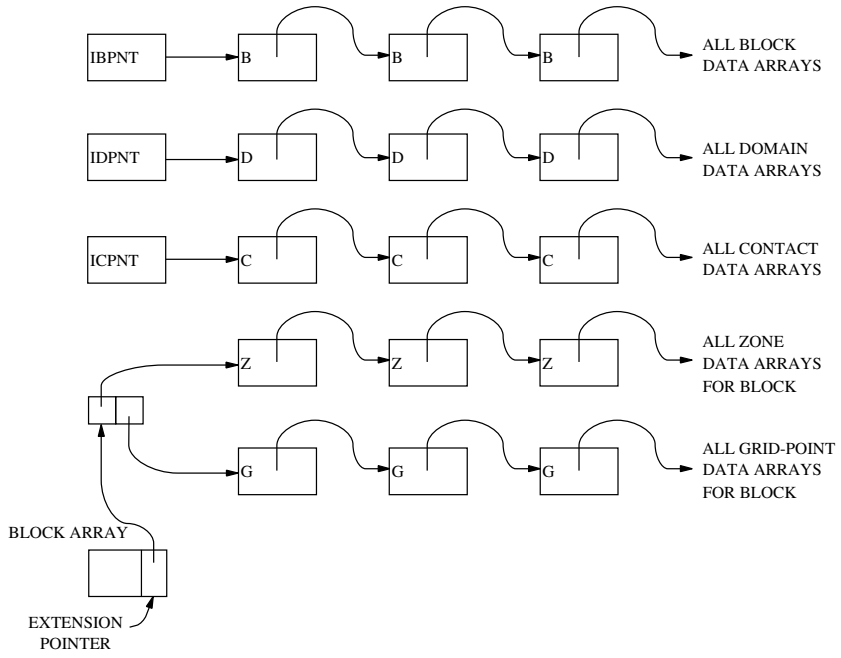


Figure 4.1 Linked lists for main data arrays
IBPNT : block_head
IDPNT : domain_head
ICPNT : contact_head

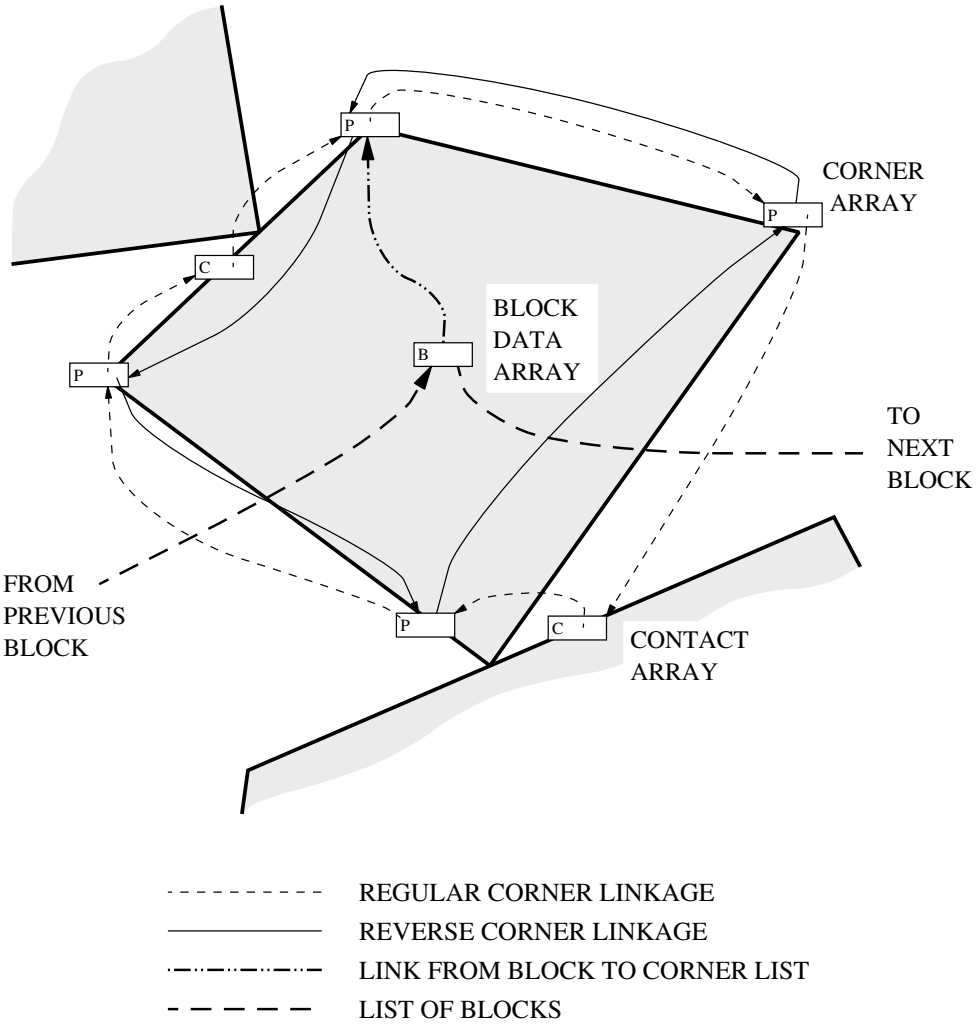


Figure 4.2 Block linkage and reverse corner links

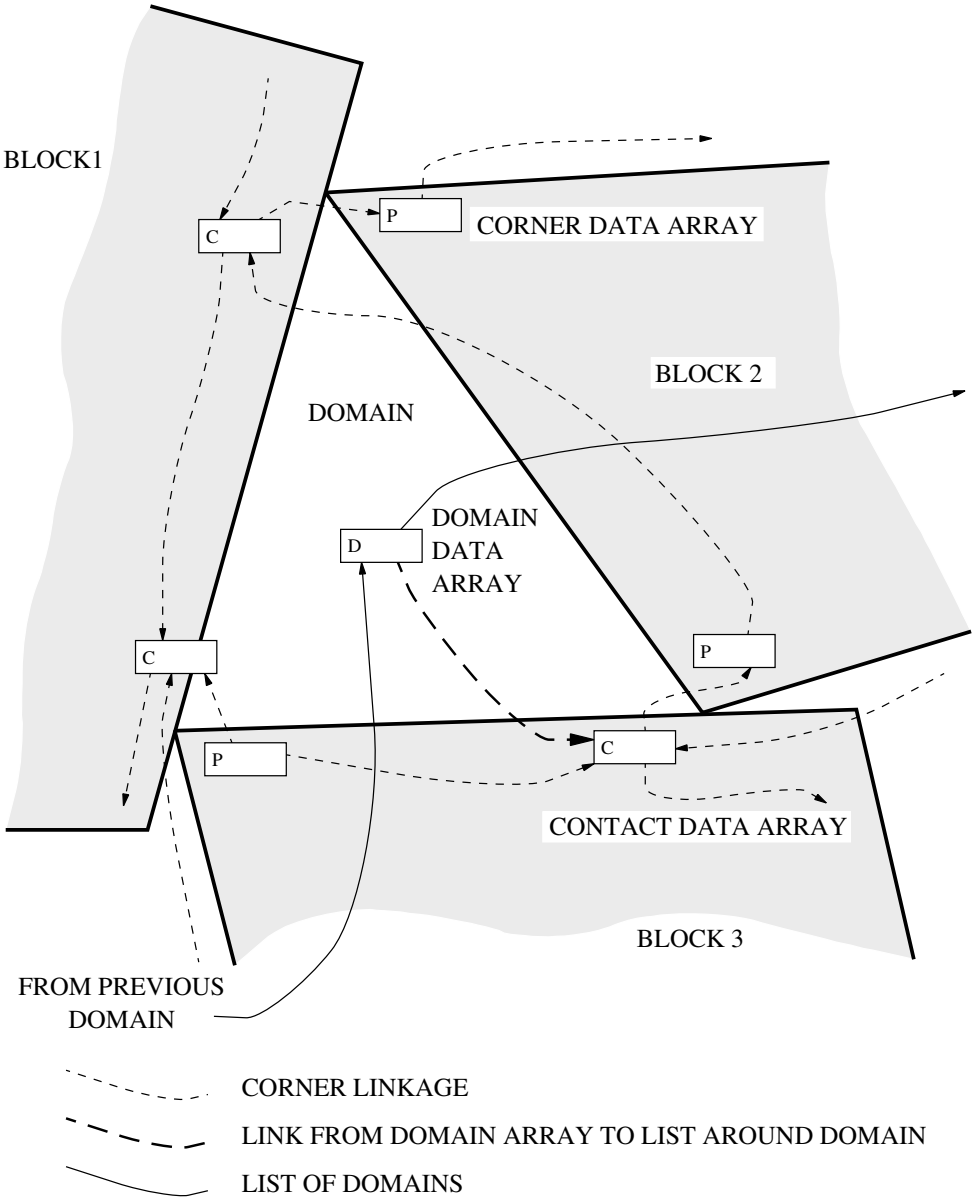
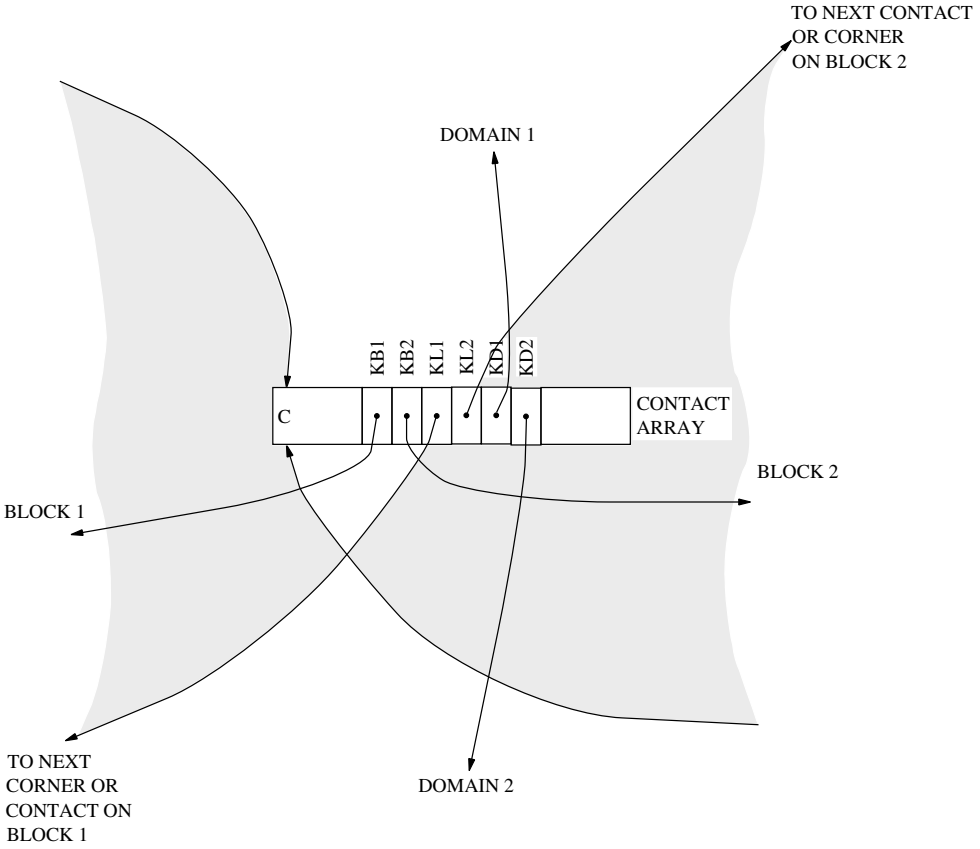


Figure 4.3 Domain linkages



NOTE: KB1, KB2 -- KD2 REFER TO THE OFFSETS

Figure 4.4 Convention used for linkages within a contact array

The list in Table 4.1 provides the names of *FISH* scalar headers and file names for the main data groups: blocks, boundary corners, contacts and domains.

Table 4.1 Global indices and include file names for main data groups

Global index	File name	Data structure
block_head	block.fin	lists of block, corner, gridpoint and zone data
bou_head	boucnr.fin	lists of boundary corner and boundary condition data
contact_head	contact.fin	list of contact data
domain_head	domain.fin	list of domain data

The list in Table 4.2 provides the names of *FISH* scalar headers and file names for the structural element data groups: reinforcement, cables, structural elements and support elements.

Table 4.2 Global indices and include file names for structural element data groups

Global index	File name	Data structure
cable_elem_head	cable.fin	list of cable element data
cable_node_head	cable.fin	list of cable node data
r_head	reinf.fin	list of reinforcement data
str_elem_head	str.fin	list of structural element data
str_node_head	str.fin	list of structural node data
str_int_head	str.fin	list of structural element interface data
sup_head	support.fin	list of support element data

It is also possible to access and change material properties that are stored locally for block zones and joint contacts (via the **ZONE** and **JOINT** commands). The extension offset for local zone properties, **z_zex**, is accessed in one of two ways: either via the block header, **block_head**, and zone index, **b_zone**, or by specifying an x,y global coordinate position with **z_near*(x,y)***. Likewise, the extension offset for local joint properties, **c_jex**, is accessed either via the contact header, **contact_head**, or by specifying an x,y global coordinate position with **c_near*(x,y)***. Table 4.3 lists the *FISH* include files that contain the local property offset names and values.

Table 4.3 Extension indices and include file names for local material properties

Extension names	File name	Data structure
c_jex	jmat.fin	list of joint constitutive models and properties
z_zex	zmat.fin	list of zone constitutive models and properties

All ".FIN" files are listed in the following sections.

4.2 Main Data Group ".FIN" Files

4.2.1 "BLOCK.FIN"

"BLOCK.FIN" provides access to block, corner, gridpoint and zone data. The block data array is accessed via the **block_head** *FISH* variable. The corner data array is accessed from the **b_corner** *FISH* variable. The gridpoint data array is accessed from the **b_gp** *FISH* variable, and the zone data array is accessed from the **b_zone** *FISH* variable.

Example 4.1 Data File "BLOCK.FIN"

```
set echo off
def $block_fin
; Block data array --- FISH include file
;---------------------------------------
; index to top of block data array is block_head
;    all values are floating-point type unless
;    it is declared otherwise
$KTYPE    =  0 ; block type number (integer)
               ; MRIG=1(rigid block)
               ; MFDEF=3(fully-deformable)
$KBNEXT   =  1 ; index of  next block in block list (integer)
$KP       =  2 ; index of one corner in block's corner list (integer)
$KMAT     =  3 ; material number (integer)
$KCONS    =  4 ; constitutive number (integer)
$KBCOD    =  5 ; code number (integer) :
               ; 0 (free block)
               ; 1 (fixed block)
$KX       =  6 ; x-coordinate of centroid
$KY       =  7 ; y-coordinate of centroid
$KXD      =  8 ; x-velocity
$KYD      =  9 ; y-velocity
$KTD      = 10 ; angular velocity (counterclockwise positive)
$KAREA    = 11 ; block area
$KBM      = 12 ; block mass
$KBI      = 13 ; moment of inertia
$KBDSF    = 14 ; density scaling factor
$KBFX     = 15 ; x-centroid force-sum
$KBFY     = 16 ; y-centroid force-sum
$KBFT     = 17 ; centroid moment sum
$KXL      = 18 ; x-load applied to block centroid
$KYL      = 19 ; y-load applied to block centroid
$KBEX     = 20 ; extension pointer (to data) (integer)
$KBUMAX   = 21 ; maximum block displacement (not used)
$KBPOR    = 22 ; code number (integer) :
```

```
                ; 0 nonporous block
                ; 1 porous block
$KBFMAG   = 23 ; block out of balance force
$KBLEXTRA = 24 ; unused offset available to users
$KBID     = 25 ; Block ID used by GIIC (integer)
$KBGROUP  = 26 ; pointer to group name (integer)
$KBSEQ    = 27 ; Block code used for hiding (integer)
$KBFIX    = 28 ; Block fixity state bits
                ;  = 1 all
                ;  = 2 X
                ;  = 4 Y
                ;  = 8 Rotation
$KBTEM    = 29 ; block centroid temperature
$KBDT1    = 30 ; block centroid temperature change in one cycle
$KBTHM    = 31 ; block centroid ''thermal mass''
$KBFIX    = 32 ; flag for fixed centroid temperature (integer)
$KBDTEM   = 33 ; accumulated block centroid temperature change
end
$block_fin


def $corner_fin
; Corner data array --- FISH include file
;---------------------------------------
; index to top of corner data array is via the FISH function b_corner(ib)
;    all values are floating-point type unless
;    it is declared otherwise
;
$MCOR     =  0 ; = 4, denotes corner (integer)
$KL       =  1 ; index of next corner or contact on block,
                ; in clockwise direction (integer)
$KR   =  2 ; index of next corner in counterclockwise
                ; direction (integer)
$KNB      =  3 ; index of host block (integer)
$KXP      =  4 ; x-coordinate of corner
$KYP      =  5 ; y-coordinate of corner
$KXCP     =  6 ; x-coordinate of local circle center
$KYCP     =  7 ; y-coordinate of local circle center
$KRAD     =  8 ; radius of local circle
$KXDP     =  9 ; x-velocity of corner
$KYDP     = 10 ; y-velocity of corner
$KGP      = 11 ; index of corresponding gridpoint if block
                ; is deformable (integer)
$KBDEX    = 12 ; extension pointer to boundary corner array (integer)
                ; >0 for exterior boundaries
                ; <0 for interior boundaries
```

```
$KPUX     = 13 ; x-displacement of corner
$KPUY     = 14 ; y-displacement of corner
$KCOREXTRA= 15 ; unused offset available to users
$KPID     = 16 ; gridpoint ID (integer)
$KTHPF    = 22 ; flag for fixed corner temperature (integer)
$KTEMP    = 23 ; corner temperature
$KDTEM1   = 24 ; corner temperature change in one cycle
$KTHMA    = 25 ; corner ``thermal mass''
$KDTEM    = 26 ; accumulated corner temperature change
end
$corner_fin


def $grid_fin
; Gridpoint data array --- FISH include file
;-----------------------------------------
; index to gridpoint data array is via the FISH function b_gp(ib)
;    all values are floating-point type unless
;    it is declared otherwise
;
$KGNEXT   =  0 ; index of next gridpoint in gridpoint list (integer)
$KCOR     =  1 ; index of corresponding block corner (integer)
$KXG      =  2 ; x-coordinate
$KYG      =  3 ; y-coordinate
$KXDG     =  4 ; x-velocity
$KYDG     =  5 ; y-velocity
$KGFX     =  6 ; x-force sum
$KGFY     =  7 ; y-force sum
$KGPM     =  8 ; gridpoint mass
$KGDSF    =  9 ; density scaling factor
$KGUX     = 10 ; x-displacement
$KGUY     = 11 ; y-displacement
$KGFMAG   = 12 ; accumulated gridpoint force magnitude
$KGPEXTRA = 13 ; unused offset available to users
$KGFOR1   = 18 ; force used in calculating damping
$KGFOR2   = 19 ; force used in calculating damping
$KGID     = 20 ; gridpoint ID (integer)
$KGSTR    = 21 ; projected gridpoint stress
$KTHGF    = 27 ; flag for fixed gridpoint temperature (integer)
$KGTEMP   = 28 ; gridpoint temperature
$KGDTEM1  = 29 ; gridpoint temperature change in one cycle
$KTHMG    = 30 ; gridpoint 'thermal mass'
$KGDTEM   = 31 ; accumulated gridpoint temperature change
end
$grid_fin
```

```
def $zone_fin
; Zone data array --- FISH include file
;--------------------------------------
; index to zone data array is via the FISH function b_zone(ib)
;    all values are floating-point type unless
;    it is declared otherwise
;
$KZNEXT   =  0 ; index of next zone in zone list (integer)
$KZG      =  1 ; start of triple pointer to 3 surrounding
                 ; gridpoints (integer)
$KZS11    =  4 ; xx-stress component
$KZS12    =  5 ; xy-stress component
$KZS22    =  6 ; yy-stress component
$KZM      =  7 ; zone mass
$KZROT    =  8 ; zone rotation
$KZPP     =  9 ; zone pore pressure
$KZPLAS   = 10 ; plasticity indicator (integer):
                 ; 0 (elastic)
                 ; 1 (currently at yield in shear and/or volume)
                 ; 2 (elastic but previously at yield)
                 ; 3 (currently at yield in tension);
                 ; 6 (ubiquitous joints at yield in shear)
                 ; 7 (ubiquitous joints previously at yield)
                 ; 8 (ubiquitous joints at yield in tension)
$KZEX     = 11 ; extension index to other constitutive relations (integer)
$KZS33    = 12 ; out-of-plane stress component
$KZONEXTRA= 13 ; unused offset available to users
$KZMDCOD  = 14 ; code used in mixed discretization (integer)
$KZMDZ2   = 15 ; address of other zone in pair (integer)
$KZMDZ4   = 16 ; address of zone in overlay pair (integer)
$KZID     = 17 ; zone ID used by GIIC (integer)
$KZGROUP  = 18 ; pointer to zone group name (integer)
$KZDENS   = 19 ; zone density
$KZSTR    = 20 ; used in stress contouring
;
; Thermal extension to Zone data array ---
;-----------------------------------------
; This data will only be present if CONFIG THERMAL is used
;
$KZCOND   = 23 ; Thermal conductivity (if same in X and Y)
$KZSPEC   = 24 ; Specific heat
$KZTHEXP  = 25 ; Thermal expansion coefficient
$KZXCOND  = 26 ; X direction thermal conductivity
$KZYCOND  = 27 ; Y direction thermal conductivity
end
```

```
$zone_fin
set echo on
```

4.2.2 "BOUCNR.FIN"

"BOUCNR.FIN" provides access to boundary corner data. The boundary corner array is accessed with the **bou_head** *FISH* variable. The generalized boundary condition array is accessed from the **$KBDROT** offset index. Note that unlike most linked lists in *UDEC*, the outer boundary corner list is circular and is not terminated by a zero index. You need to test the index from the start of the loop to detect that the entire list has been covered. Also unlike the outer boundary corner list, the interior boundary corner list is terminated with a zero index.

Example 4.2 Data File "BOUCNR.FIN"

```
set echo off
def $boucnr_fin
; Boundary corner array --- FISH include file
;-----------------------------------------------
; index to top of boundary corner array is bou_head
;   (all values are floating-point type unless declared otherwise)
;
$KBDC     =  0 ; index of next boundary corner (integer)
$KBDCOR   =  1 ; index of block corner (integer)
$KBDX     =  2 ; type of boundary condition in the x-direction (integer)
$KBDY     =  3 ; type of boundary condition in the y-direction (integer)
               ; Boundary Conditions:
               ; 1 (load boundary)
               ; 2 (boundary element boundary)
               ; 3 (viscous boundary)
               ; 4 (velocity boundary)
$KBDFX    =  4 ; total x-force
$KBDFY    =  5 ; total y-force
$KBDAPX   =  6 ; applied x-force increment or x-velocity
$KBDAPY   =  7 ; applied y-force increment or y-velocity
$KBDXH    =  8 ; x-load history
$KBDYH    =  9 ; y-load history
$KBEN     = 10 ; index of boundary element node (integer)
$KBE2     = 11 ; index of second boundary element node (integer)
$KBEF     = 12 ; boundary element load distribution factor
$KBDROT   = 13 ; index of generalized boundary condition array (integer)
$KBDFF    = 14 ; free-field indicator (integer)
$KBDPP    = 15 ; pore pressure
$KBDNWPP  = 16 ; non-wetting pore pressure
$KBDPER   = 17 ; permeability condition (integer):
               ; 0 (permeable)
               ; 1 (impermeable)
$KBDNWPER = 18 ; non-wetting permeability condition (integer)
               ; 0 (permeable)
               ; 1 (impermeable)
```

```
$KBDSAT   = 19 ; fluid saturation
$KBDPH    = 20 ; pore pressure history (integer)
$KBDNWPH  = 21 ; non-wetting pore pressure history (integer)
$KBDPF    = 22 ; pore pressure history value
end
$boucnr_fin


def $genbou_fin
; Generalized boundary condition array --- FISH include file
;-------------------------------------------------------------
; index to top of boundary corner array is via the
; boundary corner array index $KBDROT
;   (all values are floating-point type unless declared otherwise)
$KBDRS    =  0 ; rotated shear boundary condition (integer)
$KBDRN    =  1 ; rotated normal boundary condition (integer)
               ; Boundary Conditions:
               ; 4 (velocity boundary)
$KBDRVS   =  2 ; boundary shear velocity
$KBDRVN   =  3 ; boundary normal velocity
$KBDRA    =  4 ; boundary angle
$KBDRCS   =  5 ; boundary cosine
$KBDRSN   =  6 ; boundary sine
$KBDR8    =  7 ; not used
end
$genbou_fin

def $iboucnr_fin
; Interior boundary corner array --- FISH include file
;-----------------------------------------------------
; index to top of interior boundary corner array is ibou_head
;   (all values are floating-point type unless declared otherwise)
;
$KBIN     =  0 ; index of next interior boundary corner (integer)
$KBICOR   =  1 ; index of block corner (integer)
$KBIX     =  2 ; type of boundary condition in the x-direction (integer)
$KBIY     =  3 ; type of boundary condition in the y-direction (integer)
               ; Boundary Conditions:
               ; 1 (load boundary)
               ; 2 (boundary element boundary)
               ; 3 (viscous boundary)
               ; 4 (velocity boundary)
$KBIFX    =  4 ; total x-force
$KBIFY    =  5 ; total y-force
$KBIFXT   =  6 ; applied x-force increment
$KBIFYT   =  7 ; applied y-force increment
```

```
$KBIVX    =  8 ; specified x velocity
$KBIVY    =  9 ; specified y velocity
$KBIXH    = 10 ; x-load history
$KBIYH    = 11 ; y-load history
$KBIGP    = 12 ; index of associated gridpoint
$KBIBD    = 13 ; not used
$KBIROT   = 14 ; index of generalized interior
               ; boundary condition array (integer)
$KBIPP    = 15 ; interior boundary pressure (not currently used)
$KBIPER   = 16 ; interior boundary permeability (not currently used)
$KBIPH    = 17 ; interior pressure history (not currently used)
$KBIPF    = 19 ; interior pore pressure history value (not currently used)
end
$iboucnr_fin
set echo on
```

4.2.3 "CONTACT.FIN"

"CONTACT.FIN" provides access to contact data. The contact array is accessed with the **contact_head** *FISH* variable. The hydraulic location array is accessed from the **$KCH** offset index. The continuously yielding joint model extension array is accessed from the **$KEX** offset index.

Example 4.3 Data File "CONTACT.FIN"

```
set echo off
def $contact_fin
; Contact data array --- FISH include file
;----------------------------------------
; index to top of contact data array is contact_head
;   (all values are floating-point type unless declared otherwise)
;
$MCON     =  0 ; = 5, denotes contact (integer)
$KCNEXT   =  1 ; index of next contact in contact list (integer)
$KB1      =  2 ; index of first block involved in contact (integer)
$KB2      =  3 ; index of second block involved in contact (integer)
$KL1      =  4 ; index of next item in clockwise list of block
                 ;  corresponding to KB1 (integer)
$KL2      =  5 ; same as KL1 but for block KB2 (integer)
$KD1      =  6 ; index of domain to left of contact going from
                 ; block KB1 to KB2 (integer)
$KD2      =  7 ; index of domain to right of contact going from
                 ; block KB2 to KB1 (integer)
$KCM      =  8 ; material type number (integer)
$KCC      =  9 ; constitutive number (integer)
$KXC      = 10 ; contact x-coordinate
$KYC      = 11 ; contact y-coordinate
$KXDC     = 12 ; relative x-velocity (of block KB2 relative to block KB1)
$KYDC     = 13 ; relative y-velocity
$KCS      = 14 ; relative shear displacement
$KCN      = 15 ; relative normal displacement
$KCFS     = 16 ; shear force
$KCFN     = 17 ; normal force (compression positive)
$KCCOD    = 18 ; code number (integer):
                 ; 1 (corner/corner contact)
                 ; 2 (corner/edge contact)(KB1 ... corner, KB2 ... edge)
                 ; 3 (edge/corner contact)(KB1 ... edge, KB2 ... corner)
$KCAP     = 19 ; mean aperture for joint
$KCQ      = 20 ; flow-rate across joint or contact
$KCL      = 21 ; length associated with joint
$KGAM     = 22 ; fracture indicator for CLJ5 (stored as real):
                 ; 0.0 (unfractured)
                 ; 1.0 (fractured)
```

```
$KCNX     = 23 ; contact normal cosines
$KCNY     = 24 ; contact normal cosines
$KCRAT1   = 25 ; interpolation factor
$KCRAT2   = 26 ; interpolation factor
$KC1A     = 27 ; index of corner in block 1 (integer)
$KC2A     = 28 ; index of corner in block 2 (integer)
$KCOP     = 29 ; normal displacement at zero normal stress
$KCNMIN   = 30 ; maximum closure
$KHEX     = 31 ; extension index for other constitutive model
                 ; parameters (integer):
                 ; see $CY_JME_fin  if CHANGE JCONS = 3 command used
$KCH      = 32 ; extension index for hydraulic array (integer)
$KCAZ     = 33 ; contact aperture at zero normal stress
$KCONEXTRA= 36 ; unused offset available to users
$KCQNW    = 37 ; non-wetting flow rate
$KCID     = 38 ; joint ID (integer)
$KCGROUP  = 39 ; pointer to group name (integer)
end
$contact_fin


def $hydraulic_fin
; Hydraulic location array --- FISH include file
;-----------------------------------------------
; index to hydraulic location array is via the contact data
; array index $KCH
;   (all values are floating-point type)
;
$KCHN     =  0 ; contact normal displacement
$KCHX     =  1 ; contact hydraulic x-location
$KCHY     =  2 ; contact hydraulic y-location
$KCHVN    =  3 ; contact hydraulic volume
$KCHSNP   =  4 ; contact adjusted normal stress
; The following are allocated only if incompressible transient
; flow is used
$KCHAO    =  5 ; old aperture
$KCHPDO   =  6 ; old pressure difference
$KCHPD1   =  7 ; new pressure difference
end
$hydraulic_fin


def $CY_JME_fin
; CY joint model extension --- FISH include file
;-----------------------------------------------
; index to CY joint model extension is via the contact data
```

```
; array index $KHEX
;   (all values are floating-point type)
$KECYF    =  1 ; bounding friction angle in radians
$KECYR    =  2 ; reversal factor
$KECYDS   =  3 ; old shear displacement increment
end
$CY_JME_fin
set echo on
```

4.2.4 "DOMAIN.FIN"

"DOMAIN.FIN" provides access to domain data. The domain array is accessed with the **domain_head** *FISH* variable. The domain extension for fluid flow is accessed from the **$KDH** offset index.

Example 4.4 Data File "DOMAIN.FIN"

```
set echo off
def $domain_fin
; Domain data array --- FISH include file
;---------------------------------------
; index to top of domain data array is domain_head
;   (all values are floating-point type unless declared otherwise)
;
$MDOM     =  0 ; =6, denotes domain (integer)
$KDNEXT   =  1 ; index of next domain in domain list (integer)
$KDAR     =  2 ; domain area
$KPP      =  3 ; pore pressure for domain
$KUMAX    =  4 ; fictitious domain displacement
$KDLOOP   =  5 ; index of one object in counterclockwise list
               ; around domain (contact or corner)(integer)
               ; 0'th offset of object array indicates the type of data
               ; where:
               ; 4 = corner array
               ; 5 = contact array
$KDCOD    =  6 ; code number(integer):
               ; 0 (domain pressure not controlled)
               ; 1 (domain pressure controlled)
$KDH      =  7 ; domain hydraulic extension (integer)
$KDX      =  8 ; x-coordinate of domain center
$KDY      =  9 ; y-coordinate of domain center
$KDOMEXTRA= 10 ; unused offset available to users
$KNWPP    = 11 ; non-wetting domain pressure
$KDNWCOD  = 12 ; non-wetting code (integer)
$KDAPER   = 13 ; domain aperture
$KDM      = 14 ; domain material property number (integer)
$KDINI    = 15 ; pressure initialization flag (integer)
$KDID     = 16 ; domain ID (integer)
end
$domain_fin


def $dom_ext_fin
; Domain extension for fluid flow --- FISH include file
;-----------------------------------------------------
```

```
; index to domain extension for fluid flow is via the
; domain data array index $KDH
;   (all values are floating-point type unless declared otherwise)
;
$KPPO     =  0 ; domain pore pressure
$KDHIS    =  1 ; pore pressure history index (integer)
$KDHSAT   =  2 ; domain saturation
$KPPNWO   =  3 ; non-wetting pore pressure
$KDNWHIS  =  4 ; non-wetting pressure history (integer)
$KDFTEMP  =  5 ; fluid temperature
; the following are only allocated if incompressible transient
; flow is used
$KDHDENS  =  6 ; fluid density
$KDHBULK  =  7 ; fluid bulk modulus
$KDHVF    = 13 ; unbalanced domain volume due to flow
$KDHDV    = 14 ; residual unbalanced domain volume
$KDHPP    = 15 ; new domain fluid pressure
$KDHVU    = 16 ; unbalanced volume
$KDHVO    = 17 ; old unbalanced volume
$KDHPO    = 18 ; old domain fluid pressure
$KDHFAC   = 19 ; permeability factor
end
$dom_ext_fin
set echo on
```

4.3 Structural Element Data Group ".FIN" Files

4.3.1 "CABLE.FIN"

"CABLE.FIN" provides access to cable element data. The cable element data array is accessed via the **cable_elem_head** *FISH* variable. The cable node data array is accessed from the **cable_node_head** *FISH* variable.

Example 4.5 Data File "CABLE.FIN"

```
set echo off
def $cab_el_fin
; Cable reinforcing axial element array --- FISH include file
;-------------------------------------------------------------
; index to top of cable element array is cable_elem_head
;   (all values are floating-point type unless declared otherwise)
;
$KELID    =  1 ; user-given (or generated) ID num (integer)
$KELN1    =  2 ; address of end-1 node (integer)
$KELN2    =  3 ; address of end-2 node (integer)
$KELMAT   =  4 ; material number - steel (integer)
$KELARS   =  5 ; area of steel cable (now a property)
$KELFAX   =  6 ; axial force
$KELUT1   =  7 ; x-component of tangential unit vector
$KELUT2   =  8 ; y-component of tangential unit vector
$KELLEN   =  9 ; current length
$KELFAL   = 10 ; failure (state) flag (integer)
$KELEAX   = 11 ; axial strain (steel)
$KELSTF   = 12 ; current stiffness (force / strain) for steel element
$KELGID   = 13 ; not used
end
$cab_el_fin


def $cab_nd_fin
; Cable node array --- FISH include file
;---------------------------------------
; index to top of cable node array is cable_node_head
;   (all values are floating-point type unless declared otherwise)
;
$KNDID    =  1 ; user-given (or generated) ID num (integer)
$KNDX     =  2 ; x-coordinate
$KNDY     =  3 ; y-coordinate
$KNDUD1   =  4 ; x-velocity
$KNDUD2   =  5 ; y-velocity
```

```
$KNDU1    =  6 ; x-displacement
$KNDU2    =  7 ; y-displacement
$KNDSMS   =  8 ; stiffness-scaled mass (tangential direction)
$KNDMAT   =  9 ; material number - grout (integer)
$KNDZON   = 10 ; zone address the node maps into (integer)
$KNDF1    = 11 ; unbalanced x-force
$KNDF2    = 12 ; unbalanced y-force
$KNDAP1   = 13 ; applied x-force
$KNDAP2   = 14 ; applied y-force
$KNDXFX   = 15 ; = 1 if fixed in x (integer) (not implemented in UDEC)
$KNDYFX   = 16 ; = 1 if fixed in y (integer) (not implemented in UDEC)
$KNDEFL   = 17 ; effective length
$KNDT1    = 18 ; x-component of tangential unit vector
$KNDT2    = 19 ; y-component of tangential unit vector
$KNDBFL   = 20 ; flag to indicate if grout has failed (integer)
$KNDW1    = 21 ; weight-1
$KNDW2    = 22 ; weight-2
$KNDW3    = 23 ; weight-3
$KNDFS    = 24 ; grout (shear) force
$KNDDMS   = 25 ; 'correct' dynamic mass (tangential direction)
$KNDRSV   = 26 ; tangential relative velocity (node relative to grid)
end
$cab_nd_fin
set echo on
```

4.3.2 "REINF.FIN"

"REINF.FIN" provides access to reinforcement data. The reinforcement data array is accessed via the **r_head** *FISH* variable.

Example 4.6 Data File "REINF.FIN"

```
set echo off
def $reinf_fin
; Local reinforcement data array --- FISH include file
;-----------------------------------------------------
; index to top of local reinforcement data array is r_head
;   (all values are floating-point type unless declared otherwise)
;
$MLRCON   =  0 ; = 6, denotes local reinforcement (integer)
$KLRCNEXT =  1 ; index of next reinforcement in the list (integer)
$KLRB1    =  2 ; index of first block involved in reinforcement (integer)
$KLRB2    =  3 ; index of second block involved in reinforcement (integer)
$KLRL1    =  4 ; index of next item in clockwise list of block
                 ;  corresponding to KB1 (integer)
$KLRL2    =  5 ; same as KL1 but for block KB2 (integer)
$KLRD1    =  6 ; index of domain to left of reinforcement going from
                 ;  block KB1 to KB2 (integer)
$KLRD2    =  7 ; index of domain to right of reinforcement going from
                 ;  block KB2 to KB1 (integer)
$KLRCM    =  8 ; material type number (integer)
$KLRCC    =  9 ; constitutive number (integer)
$KLRXC    = 10 ; x-coordinate
$KLRYC    = 11 ; y-coordinate
$KLRXDC   = 12 ; relative x-velocity (of block KB2 relative to block KB1)
$KLRYDC   = 13 ; relative y-velocity
$KLRCS    = 14 ; relative shear displacement on joint
$KLRCN    = 15 ; relative normal displacement on joint
$KLRCFS   = 16 ; shear force at joint
$KLRCFN   = 17 ; normal force at joint

$KLRCAP   = 19 ; orientation of reinforcement relative to joint
$KLRCQ    = 20 ; total axial displacement of reinforcement
$KLRCL    = 21 ; axial force on reinforcement
$KLRGAM   = 22 ; shear force on reinforcement
end
$reinf_fin
set echo on
;
```

4.3.3 "STR.FIN"

"STR.FIN" provides access to structural element data. The structural element lumped mass (node) array is accessed via the **str_node_head** *FISH* variable. The structural element data array is accessed from the **str_elem_head** *FISH* variable. The structural element interface array is accessed from the **str_int_head** *FISH* variable.

Example 4.7 Data File "STR.FIN"

```
set echo off
def $str_fin
;  Structural element parameters -- FISH Include file
;  ---------------------------------------------------
; Structural element lumped mass array accessed via str_node_head
;   (all values are floating-point type unless declared otherwise)
;
$SNDNEXT    =  0 ; index of next node on list (integer)
$SNDID      =  1 ; nodal point ID number (integer)
$SNDX       =  2 ; x-coordinate of nodal point
$SNDY       =  3 ; y-coordinate of nodal point
$SNDUD1     =  4 ; x-velocity of nodal point
$SNDUD2     =  5 ; y-velocity of nodal point
$SNDU1      =  6 ; x-displacement of nodal point
$SNDU2      =  7 ; y-displacement of nodal point
$SNDSMS     =  8 ; stiffness scaled mass (tangential direction)
$SNDF1      = 11 ; unbalanced x-force
$SNDF2      = 12 ; unbalanced y-force
$SNDAP1     = 13 ; applied x-force
$SNDAP2     = 14 ; applied y-force
$SNDEFL     = 17 ; effective length
$SNDT1      = 18 ; x-component of tangential unit vector
$SNDT2      = 19 ; y-component of tangential unit vector
$SNDDMS     = 25 ; actual mass (tangential direction)
$SNDUDR     = 27 ; rotational velocity
$SNDUR      = 28 ; rotational displacement
$SNDFR      = 29 ; unbalanced moment force
$SNDAPR     = 30 ; applied moment force
$SNDDIN     = 31 ; dynamic rotational inertia
$SNDSIN     = 32 ; maximum of scaled mass or dynamic inertia
$SNDTINC    = 34 ; temperature increment of node
$SNDILYR    = 35 ; multiple layer temp flag used in thermal
$SNDEXTRA   = 36 ; unused offset available to users
; The following are used for rockbolts only
; --- rockbolt node data array ---
;    (uses structural/cable element offset names for first 36 items)
$KNDTYP     = 37 ; property ID
```

```
$KNDTAD     = 38 ; index for property data array
$KNDCOD     = 39 ; = 1 rockbolt node outside of any zones
;                  = 4 rockbolt node inside a zone
$KNDCOD2    = 40 ; = 5 node in zone
$KNDPIN     = 41 ; not currently used in UDEC
$KNDMA1     = 42 ; not currently used in UDEC
$KNDMA2     = 43 ; not currently used in UDEC
$KNDNFL     = 44 ; = 0 normal spring elastic
;                  = 2 normal spring yielding
$KNDFN      = 45 ; contact spring normal force
$KNDUP      = 46 ; contact spring normal gap
$KNDUM      = 47 ; contact spring ?
$KNDUA      = 48 ; contact spring shear displacement
$KNDUNR     = 49 ; not currently used in UDEC
$KNDHFR     = 50 ; not currently used in UDEC
$KNDHTD     = 51 ; not currently used in UDEC
$KNDHTH     = 52 ; not currently used in UDEC
$KNDHIN     = 53 ; not currently used in UDEC
$KNDHMI     = 54 ; not currently used in UDEC
$KNDHTR     = 55 ; not currently used in UDEC
$KNDEP      = 56 ; mean effective pressure in plane
                 ; perpendicular to rockbolt
$KNDNC      = 57 ; not currently used in UDEC
$KNDNC1     = 58 ; not currently used in UDEC
$KNDLO1     = 59 ; not currently used in UDEC
$KNDLO2     = 60 ; not currently used in UDEC
$KNDFOR1    = 61 ; X direction nodal force sum
$KNDFOR2    = 62 ; Y direction nodal force sum
$KNDUS      = 63 ; spring shear displacement
$KNDCSTR    = 64 ; not currently used in UDEC
$KNDCAREA   = 65 ; not currently used in UDEC
$KNDSFCE    = 66 ; not currently used in UDEC
$KNDDT      = 67 ; not currently used in UDEC
$KNDGP      = 68 ; not currently used in UDEC
$KNDSXXI    = 69 ; XX stress in zone when node was added
$KNDSXYI    = 70 ; XY stress in zone when node was added
$KNDSYYI    = 71 ; YY stress in zone when node was added
$KNDSZZI    = 72 ; ZZ stress in zone when node was added
$KNDPPI     = 73 ; not currently used in UDEC
$KNDSI      = 74 ; flag used to indicate installation stress assigned
$KNDCFAC    = 75 ; not currently used in UDEC
$KNDAXF1    = 76 ; not currently used in UDEC
$KNDAXF2    = 77 ; not currently used in UDEC
$KNDRSD     = 78 ; not currently used in UDEC
$KNDXO      = 79 ; not currently used in UDEC
$KNDYO      = 80 ; not currently used in UDEC
```

```
$KNDACC1    = 81 ; not currently used in UDEC
$KNDACC2    = 82 ; not currently used in UDEC
$KNDACCTH   = 83 ; not currently used in UDEC
$KNDSP7     = 84 ; not currently used in UDEC
$KNDSP8     = 85 ; not currently used in UDEC
$KNDSP9     = 86 ; not currently used in UDEC
;
;
;Structural element array access via str_elem_head
;   (all values are floating-point type unless declared otherwise)
;
$SELNEXT    =  0 ; index of next element (integer)
$SELID      =  1 ; structural element ID number (integer)
$SELN1      =  2 ; address of lumped mass 1 (integer)
$SELN2      =  3 ; address of lumped mass 2 (integer)
$SELMAT     =  4 ; material number (integer)
$SELARS     =  5 ; area of structural element
$SELFAX     =  6 ; axial force  NOTE: internally compression negative.
                 ; The sign is switched when printing to be consistent
                 ; with cable elements.
$SELUT1     =  7 ; x-component of tangential unit vector
$SELUT2     =  8 ; y-component of tangential unit vector
$SELLEN     =  9 ; length of element
$SELFAL     = 10 ; axial failure indicator (integer):
                 ; 0  (elastic)
                 ; 1  (yield tension)
                 ; 2  (yield compression)
$SELSTF     = 12 ; element stiffness
$SELFS      = 14 ; shear force
$SELM1      = 15 ; total moment at node 1
$SELM2      = 16 ; total moment at node 2
$SELTH      = 17 ; structural element thickness
$SELIN      = 18 ; structural element second moment of area
$SELSH      = 19 ; structural element shape ratio
$SELFAL2    = 20 ; second axial failure indicator (integer):
                 ; 0  (elastic)
                 ; 1  (yield tension)
                 ; 2  (yield compression)
$SELCONS    = 21 ; structural element constitutive model (integer)
$SELEXTRA   = 22 ; unused offset available to users
;
; structural element interface array accessed via str_int_head
;   (all values are floating-point type unless declared otherwise)
;
$SCCONT     =  0 ; contact type (integer)
$SCNEXT     =  1 ; index of next interface (integer)
```

```
$SB1        =  2 ; address of lumped mass (integer)
$SB2        =  3 ; address of block (integer)
$SL2        =  5 ; index of next item in clockwise list of block (integer)
$SCM        =  8 ; material number (integer)
$SCC        =  9 ; constitutive number (= 1) (integer)
$SXC        = 10 ; x-coordinate of contact
$SYC        = 11 ; y-coordinate of contact
$SXDC       = 12 ; relative x velocity (of block relative to lumped mass)
$SYDC       = 13 ; relative y velocity (of block relative to lumped mass)
$SCS        = 14 ; relative shear displacement
$SCN        = 15 ; relative normal displacement
$SCFS       = 16 ; shear force
$SCFN       = 17 ; normal force
$SCCOD      = 18 ; contact type (integer)
$SCNX       = 23 ; x shift due to element thickness
$SCNY       = 24 ; y shift due to element thickness
$SCRAT1     = 25 ; force weighting factor 1
$SCRAT2     = 26 ; force weighting factor 2
;
end
$str_fin
set echo on
```

4.3.4 "SUPPORT.FIN"

"SUPPORT.FIN" provides access to support element data. The support element array is accessed via the **sup_head** *FISH* variable.

Example 4.8 Data File "SUPPORT.FIN"

```
set echo off
def $support_fin
; Support element data array --- FISH include file
;-----------------------------------------------------
; index to top of support element data array is sup_head
;   (all values are floating-point type unless declared otherwise)
;
$KSUEXT   =  1 ; spare extension (not used)
$KSUT1    =  2 ; x-component unit vector
$KSUT2    =  3 ; y-component unit vector
$KSUMAT   =  4 ; material number (integer)
$KSUFN    =  5 ; normal force (compression positive)
$KSUFS    =  6 ; shear force (not used)
$KSUUN    =  7 ; total normal displacement
$KSUUS    =  8 ; total shear displacement
$KSUTAD   =  9 ; address of top block (integer)
$KSUBAD   = 10 ; address of bottom block (integer)
$KSUTRA   = 11 ; ratio (i.e., weighting factor) for top
$KSUBRA   = 12 ; ratio (i.e., weighting factor) for bottom
$KSUTCO   = 13 ; address of top corner (integer)
$KSUBCO   = 14 ; address of bottom corner (integer)
$KSUSUB   = 15 ; pointer to sub-elements (integer)
$KSUNSM   = 16 ; number of elements (integer)
$KSUNMX   = 17 ; max normal displacement in past
$KSULIM   = 18 ; axial displacement at maximum force
$KSUTYP   = 19 ; support type (1 = rate dependent) (integer)
$KSUFSTAT = 20 ; maximum static force
$KSUCSTIF = 21 ; stiffness constant
$KSUDE10  = 22 ; previous force increment
$KSUTBMX  = 23 ; maximum x-value in table
$KSULDMX  = 24 ; maximum axial force achieved in past
$KSUN0    = 25 ; axial displacement at zero axial force
$KSUMAS   = 26 ; support deletion flag (integer)
$KSUEXTRA = 27 ; unused offset available to users
$KSU28    = 28 ; not used
$KSU29    = 29 ; not used
end
$support_fin
set echo on
```

```
;
```

4.4 Local Material Model and Properties Data Group ".FIN" Files

4.4.1 "JMAT.FIN"

"JMAT.FIN" provides access to joint constitutive models and properties that are stored locally for joint contacts using the **JOINT** command. The extension for local joint properties is the **c_jex**(*ci*) *FISH* variable.

Example 4.9 Data File "JMAT.FIN"

```
set echo off
def $jmat_fin
;  JOINT material model parameters -- FISH Include file
; ------------------------------------------------
;  index to zone property list is c_jex(zi)
;   (all values are floating-point type unless declared otherwise)

;  point contact --- Coulomb slip (JOINT MODEL POINT)
;  ------------------------------------------------
$pc_kn     =  1 ; contact normal stiffness (force/displacement)
$pc_ks     =  2 ; contact shear stiffness (force/displacement)
$pc_coh    =  3 ; contact cohesion (force)
$pc_phi    =  4 ; contact friction angle (degrees)
$pc_psi    =  5 ; contact dilation angle (degrees)
$pc_ten    =  6 ; contact tensile strength (force)
$pc_perm   =  7 ; contact permeability
$pc_ares   =  8 ; residual aperture at high stress
$pc_azero  =  9 ; aperture at zero normal stress
$pc_nwjperm= 10 ; non-wetting joint permeability

;  joint area contact --- Coulomb slip (JOINT MODEL AREA)
;  ------------------------------------------------
$ac_kn     =  1 ; joint normal stiffness (stress/displacement)
$ac_ks     =  2 ; joint shear stiffness (stress/displacement)
$ac_coh    =  3 ; joint cohesion (stress)
$ac_phi    =  4 ; joint friction angle (degrees)
$ac_psi    =  5 ; joint dilation angle (degrees)
$ac_ten    =  6 ; joint tensile strength (stress)
$ac_tab    =  7 ; table number for normal displacement vs
                  ; normal stress (integer)
$ac_zerdil =  8 ; shear displacement for zero dilation
$ac_ares   =  9 ; residual aperture at high stress
$ac_azero  = 10 ; aperture at zero normal stress
$ac_empb   = 11 ; empirical multiplier for fluid flow law
$ac_expa   = 12 ; exponent of joint hydraulic multiplier
```

```
$ac_perm   = 13 ; joint permeability
$ac_grdex  = 14 ; gradient exponent
$ac_nwjperm= 15 ; non-wetting joint permeability

;  joint area contact --- Coulomb slip with residual strength
;                         (JOINT MODEL RESIDUAL)
;  ------------------------------------------------------------
$rs_kn     =  1 ; joint normal stiffness (stress/displacement)
$rs_ks     =  2 ; joint shear stiffness (stress/displacement)
$rs_coh    =  3 ; joint cohesion (stress)
$rs_phi    =  4 ; joint friction angle (degrees)
$rs_psi    =  5 ; joint dilation angle (degrees)
$rs_ten    =  6 ; joint tensile strength (stress)
$rs_tab    =  7 ; table number for normal displacement vs
                  ; normal stress (integer)
$rs_zerdil =  8 ; shear displacement for zero dilation
$rs_ares   =  9 ; residual aperture at high stress
$rs_azero  = 10 ; aperture at zero normal stress
$rs_empb   = 11 ; empirical multiplier for fluid flow law
$rs_expa   = 12 ; exponent of joint hydraulic multiplier
$rs_perm   = 13 ; joint permeability
$rs_grdex  = 14 ; gradient exponent
$rs_phir   = 15 ; residual friction angle (degrees)
$rs_cohr   = 16 ; residual cohesion (stress)
$rs_tenr   = 17 ; residual tensile strength (stress)
$rs_frac   = 18 ; fracture condition
                  ; 0 - not fractured
                  ; 1 - joint has fractured
$rs_nwjperm= 19 ; non-wetting joint permeability


;  joint area contact --- continuously yielding (JOINT MODEL CY)
;  ------------------------------------------------------------
$cy_kn     =  1 ; joint normal stiffness (stress/displacement)
$cy_ks     =  2 ; joint shear stiffness (stress/displacement)
$cy_coh    =  3 ; joint cohesion (stress)
$cy_phi    =  4 ; joint friction angle (degrees)
$cy_psi    =  5 ; joint dilation angle (degrees)
$cy_ten    =  6 ; joint tensile strength (stress)
$cy_zerdil =  7 ; shear displacement for zero dilation
$cy_ares   =  8 ; residual aperture at high stress
$cy_azero  =  9 ; aperture at zero normal stress
$cy_empb   = 10 ; empirical multiplier for fluid flow law
$cy_expa   = 11 ; exponent of joint hydraulic multiplier
$cy_perm   = 12 ; joint permeability
$cy_knexp  = 13 ; exponent of joint normal stiffness
```

```
$cy_ksexp  = 14 ; exponent of joint shear stiffness
$cy_jf     = 15 ; joint initial friction angle (degrees)
$cy_jr     = 16 ; joint roughness parameter (length)
$cy_knmax  = 17 ; maximum value of joint normal stiffness
$cy_knmin  = 18 ; minimum value of joint normal stiffness
$cy_ksmax  = 19 ; maximum value of joint shear stiffness
$cy_ksmin  = 20 ; maximum value of joint shear stiffness
$cy_kecyds = 21 ; old shear displacement increment
$cy_grdex  = 22 ; gradient exponent
$cy_kecyr  = 23 ; reversal factor
$cy_nwjperm= 24 ; non-wetting joint permeability

end
$jmat_fin
set echo on
```

4.4.2 "ZMAT.FIN"

"ZMAT.FIN" provides access to block zone constitutive models and properties that are stored locally for zones using the **ZONE** command. The extension for local zone properties is the **z_zex** *FISH* variable. All values are stored as real numbers. This means that only the function **fmem** is used with the offsets. The function **lmem** should not be used. These offsets are also used with the **z_prop** *FISH* function for setting and retrieving zone model values.

The mapping of the zone model numbers is as follows:

SS = 3
Mohr-Coulomb = 5
DY = 6
Ubiquitous = 7
Null = 8
Elastic = 9

Example 4.10 Data File "ZMAT.FIN"

```
set echo off
def $zmat_fin
;  ZONE material model parameters -- FISH Include file
; ------------------------------------------------
;  index to zone property list is z_zex(zi)
;   (all values are floating-point type unless declared otherwise)

;  linear elastic (ZONE MODEL ELASTIC)
;  -----------------------------------
$le_kgmod   = 1 ; shear modulus
$le_kk      = 2 ; bulk modulus
$le_ke1     = 3 ; bulk + 4/3*shear
$le_ke2     = 4 ; bulk - 2/3*shear
$le_kg2     = 5 ; 2 * shear modulus

;  mohr-coulomb (ZONE MODEL MOHR)
; ------------------------------
$mc_kgmod   =  1 ; shear modulus
$mc_kk      =  2 ; bulk modulus
$mc_kcoh    =  3 ; cohesion
$mc_kphi    =  4 ; friction angle in degrees
$mc_kpsi    =  5 ; dilation angle in degrees
$mc_kind    =  6 ; plasticity indicator (integer)
$mc_kten    =  7 ; tensile strength
$mc_kcsnp   =  8 ; 2.0 * cohesion * sqrt(anphi)
$mc_knphi   =  9 ; anphi  =  (1+sin(phi))/(1-sin(phi))
```

```
$mc_knpsi   = 10 ; anpsi  =  (1+sin(psi))/(1-sin(psi))
$mc_ke1     = 11 ; bulk + 4/3*shear
$mc_ke2     = 12 ; bulk - 2/3*shear
$mc_kx1     = 13 ; ke1 - (ke2 * anpsi) - ((ke2 - ke1 * anpsi) * anphi)

;  ubiquitous joint (ZONE MODEL UBIQUITOUS)
; ----------------------------------------
$ubj_kgmod  =  1 ; shear modulus
$ubj_kk     =  2 ; bulk modulus
$ubj_kcoh   =  3 ; cohesion
$ubj_kphi   =  4 ; friction angle in degrees
$ubj_kpsi   =  5 ; dilation angle in degrees
$ubj_kind   =  6 ; plasticity indicator (integer)
$ubj_kten   =  7 ; tensile strength
$ubj_kja    =  8 ; ubiquitous joint angle
$ubj_kjf    =  9 ; ubiquitous joint friction angle
$ubj_kjd    = 10 ; ubiquitous joint dilation angle
$ubj_kjcoh  = 11 ; ubiquitous joint cohesion
$ubj_kjten  = 12 ; ubiquitous joint tension
$ubj_kcsnp  = 13 ; 2.0 * cohesion * sqrt(anphi)
$ubj_knphi  = 14 ; anphi  =  (1+sin(phi))/(1-sin(phi))
$ubj_knpsi  = 15 ; anpsi  =  (1+sin(psi))/(1-sin(psi))
$ubj_ke1    = 16 ; bulk + 4/3*shear
$ubj_ke2    = 17 ; bulk - 2/3*shear
$ubj_kx1    = 18 ; ke1 - (ke2 * anpsi) - ((ke2 - ke1 * anpsi) * anphi)

;  strain-softening/hardening mohr-coulomb (ZONE MODEL SS)
;  -------------------------------------------------------
$ss_kgmod  =  1 ; shear modulus
$ss_kk     =  2 ; bulk modulus
$ss_kcoh   =  3 ; cohesion
$ss_kphi   =  4 ; friction angle in degrees
$ss_kpsi   =  5 ; dilation angle in degrees
$ss_kind   =  6 ; plasticity indicator (integer)
$ss_kepdev =  7 ; plastic deviatoric strain
$ss_kctab  =  8 ; cohesion softening table (integer)
$ss_kftab  =  9 ; friction softening table (integer)
$ss_kdtab  = 10 ; dilation softening table (integer)
$ss_kttab  = 11 ; tension  softening table (integer)
$ss_kten   = 12 ; tensile strength
$ss_kepten = 13 ; plastic tensile strain
$ss_kcsnp  = 14 ; 2.0 * cohesion * sqrt(anphi)
$ss_knphi  = 15 ; anphi  =  (1+sin(phi))/(1-sin(phi))
$ss_knpsi  = 16 ; anpsi  =  (1+sin(psi))/(1-sin(psi))

;  double yield (ZONE MODEL DY)
```

```
; ------------------------------
$dy_kgmod   = 1 ; shear modulus
$dy_kk      = 2 ; bulk modulus
$dy_kcoh    = 3 ; cohesion
$dy_kphi    = 4 ; friction angle in degrees
$dy_kpsi    = 5 ; dilation angle in degrees
$dy_kind    = 6 ; plasticity indicator (integer)
$dy_kepdev  = 7 ; plastic deviatoric strain
$dy_kctab   = 8 ; cohesion softening table (integer)
$dy_kftab   = 9 ; friction softening table (integer)
$dy_kdtab   = 10 ; dilation softening table (integer)
$dy_kttab   = 11 ; tension  softening table (integer)
$dy_kepvol  = 12 ; volumetric strain
$dy_kpcmax  = 13 ; plastic cap
$dy_kten    = 14 ; tensile strength
$dy_kpctab  = 15 ; cap softening table (integer)
$dy_kmdmul  = 16 ; cap multiplier
$dy_kdystat = 17 ; cap plasticity indicator
$dy_kepten  = 18 ; tensile plastic strain
$dy_kcsnp   = 19 ; 2.0 * cohesion * sqrt(anphi)
$dy_knphi   = 20 ; anphi  =  (1+sin(phi))/(1-sin(phi))
$dy_knpsi   = 21 ; anpsi  =  (1+sin(psi))/(1-sin(psi))

end
$zmat_fin
set echo on
```