A GUIDE TO USE OF DELIGHT.STRUCT.

Mark A. Austin

Department of Civil Engineering.
University of California.
Berkeley,
California 94720.

ABSTRACT

DELIGHT.STRUCT is an optimization based design system for statically and dynamically loaded structures whose response may be linear or nonlinear. This report provides a step-by-step guide through the computational stages of the program as applied to the seismic loading of planar steel frames, including braced frames. A comprehensive example is presented.

June 13, 1983

II. ACKNOWLEDGEMENTS.

This research was supported by the National Science Foundation under Grant PFR-7908261 with the University of California, Berkeley. Computing facilities were provided in part by Equipment Grant ENG-7810992 from the National Science Foundation.

III. TABLE OF CONTENTS.

1. INTRODUCTION.

This is the third in a series of reports looking at the optimal design of planar moment-resistant steel frames subject to earthquake excitation. The general capabilities of a computer-aided design environment called DELIGHT.STRUCT are discussed in two previous reports by Balling et al.[1,2]. A philosophy for steel frame design is introduced and the results of a 4 story-3 bay optimal design are presented as an example.

The primary objective of this report is to provide the uninitiated 'user' of DELIGHT.STRUCT with a step by step guide through the computational stages of this program. It is assumed here that the frame preprocessor *finput* written specifically for the expedient input of both moment-resistant and braced steel frames will be used. Consequently, this report covers only a subsection of the total capabilities of the design system. A comprehensive example is included so the 'user' can see exactly what is required in working through the various stages of optimization. The second purpose of the report is to provide information that is both supplementary and complementary to that already contained within reports Balling et al.[1], Balling et al.[2] and Bhatti[3]. In particular, information exemplifying the dynamic storage allocation for the array *Resp* is provided for the Workman braced-frame[4]. Any further post-processing software development requires a detailed knowledge of this matrix a priori. It is hoped this example will be sufficiently complete in displaying the general structure of *Resp*. A glossary of define statements, interactive commands and files is also located in the Appendices. The 'user' should utilize the Appendices to become aquainted with the capabilities of this program.

1.1 Report Outline.

The following chapter examines the program structure of a subsection of DELIGHT.STRUCT. It emphasizes the strategy employed to complete a single iteration of optimization, and stresses key points to keep in mind while working through a problem.

Chapter 3 introduces the 'example' problem, also known as the Workman frame[4]. The frame geometry, earthquake loading, design variables and desired failure mechanism are briefly discussed to aid the reader in understanding the steps that follow. Echo printouts of the minimum volume design for the example problem are located in Appendix 1. The reader can use this in conjunction with the footnotes appended and Appendix 5 of Balling et al.[1] to work through his/her own problem. A glossary of terms and command descriptions follows in Appendix 2. In Appendix 3, a discussion of the dynamic array allocation in *Resp* for the example problem is presented. Finally, Appendix 4 gives a summary of the steps to work through when recompiling DELIGHT.STRUCT.

1.2 Background Reading.

The following sections of Balling et al.[1] and Balling et al.[2] are recommended as minimal background,and preliminary 'user' reading before continuing on to the remaining chapters of this report.

Balling et al.[1]

Section 1	: Introduction
Section 2	: Background to DELIGHT.STRUCT.
Section 3.3	: Seismic Resistant Design of Frames.
Section 3.3.1	: Problem Definition Phase.
Section 3.3.2	: Computation Phase.

Balling et al.[2]

Section 1	: Introduction.
Section 2.1	: Loading.
Section 2.2.1	: Geometry of Model.
Section 2.2.2	: Simulation Procedure.
Section 2.2.3	: Element Models.
Section 2.4	: Cost Function.
Section 2.5	: Constraint Functions.

If the 'user' intends to add to the present software in DELIGHT.STRUCT a comprehensive knowledge of its capabilities is essential. A suggested 'getting-started' procedure is to first read the aforementioned references, work through this report with a simple example, then return to Balling et al.[2] and read chapters two,three and four, before proceeding to modify the program.

The reader should consult [7] for an introduction to DELIGHT.

2. PROGRAM STRUCTURE AND SOLUTION STRATEGY.

When the 'user' wishes to initiate a new problem, existing software may or may not be employed. New software will need to be written if the problem is not concerned with the optimization of moment-resistant steel frames subject to seismic excitation. A detailed knowledge of the array *Resp* that stores the response history of the structure is required before routines describing the conventional and functional constraints and objective functions, as well as some plotting procedures can be assembled. An example of these routines for a single story-one bay braced frame is located in Balling et al.[1]; Appendix 5. This report, however, assumes that *finput* will be used in conjuction with fortran software already written to describe constraints and objective functions relevant to the optimal design of moment-resistant steel frames. The reader is referred to Figures 3,4, and 5 of Balling et al.[1] for a schematic of the fortran subroutines and rattle procedures used at each stage.

This chapter is divided into three sections. The first two highlight key points to remember when the problem is being defined and the iterations of optimization are being carried out. Part three discusses software features available to facilitate the running of long jobs.

2.1 Problem Formulation.

The main points to remember when setting up a problem are :

(1) The optimization problem is reduced to finding a design vector *X(NPARAM)* such that all constraints on performance are satisfied while simultaneously minimizing a cost function within a feasible domain. The forms of constraint and cost functions applicable to the seismic resistant design of frames are discussed in Sections 2.4 and 2.5 of Balling et al.[2].

(2) Curves of best fit relating wide flange section dimensions introduced by Walker[6] and discussed by Balling et al.[2]: Section 2.3.2 are used. This enables the frame to be considered as an assemblage of elements, each characterized by a single parameter in the design vector.

(3) The dimension of the design vector is *NPARAM*. The first *NSIMPAR* variables correspond to the unknown element sizes, and those remaining are dummy design parameters utilized only when story drifts, or other time-dependent cost functions are being minimized. The latter terms require special treatment and the 'user' should read Section 2.4.2 of Balling et al.[1] and the define *fprint* in Appendix 2 of this report to see what is required.

(4) Excessive weighting on any given design variable may significantly distort the objective function contours. This may result in slow numerical convergence at the optimization stage. It may also cause difficulties in selecting perturbations for the calculation of numerical derivatives. Potential problems due to bad scaling are therefore mitigated by requiring component values in the design vector to lie within the range [-1,1]. Maximum and minimum allowable section sizes are read from the file *assumptions* and the fortran routine *fsectn.f* scales this range by a linear mapping on to the interval [-1,1]. Constraint functions are also normalized to the form shown in the following section.

(5) The correct formatting for *assumptions* is located in the fortran subroutine *fassum.f.* Because *assumptions* is only read at the *finput* stage, the 'user' has to return to this procedure if subsequent problem modifications are required.

2.2 The Optimization Process

DELIGHT.STRUCT uses the method of feasible directions for the constrained optimization problem. An explanation of the mathematical formulation of this method has been presented by Bhatti[3]. The objective of this section is to outline the strategy DELIGHT.STRUCT uses in completing 1 iteration of optimization. It does not aim review the theory. The key points to note beforehand are :

(1) The permissible design space is a hyper-cube of dimension *NPARAM*. This region is subdivided into two subdomains. If a given X in the space satisfies all the constraints, the point is characterized as feasible. Otherwise, it is termed infeasible.

(2) Constraints are divided into two classes. Conventional constraints are time independent scalar valued functions of the form:

$$PSI = \left[\frac{g(X)}{g(\mathit{allowable})} \right] - 1$$

Functional constraints are time dependent. Their dimensionless form is :

$$PSI = \left\{ \left[\frac{g(X,t)}{g(\mathit{allowable})} \right] - 1 \right\}_{\mathit{max\ over\ time}}$$

The manner in which they are incorporated into the feasible directions algorithm is discussed in Bhatti[3]: Chapter 5.

An iteration of optimization using this algorithm is composed of two stages, direction finding and steplength calculation.

2.2.1 The Direction Vector.

If the design vector is infeasible, a new direction vector will be chosen so that X moves as close as possible to the feasible domain. This is achieved by shifting X so that a reduction in the maximum constraint violation occurs. The cost or objective function plays no role in this case. However, within the feasible domain a direction is calculated so that a maximum decrease in cost function is obtained. The program strategy is as follows:

(1) The procedure *actgrad* sweeps all the constraints and decides which should be included in the gradient calculation. Its criteria are:

Let psi* = max[PSI,0.0]

A constraint is incorporated into the direction vector
calculation and termed 'active' if :

constraint % $\geqslant$ *(psi* - Eps + 1)* $\cdot$ *100%*

otherwise, the constraint plays no further role.

The rattle parameter *nact* is the sum of constraints included at this step.

(2) A jacobian matrix of dimension(*nact, NPARAM*) containing the partial derivatives of constraints with respect to the design variables *X* is then calculated. Each derivative is estimated by a first order forward finite difference approximation. The reader is encouraged to look at the rattle procedure *perturb* to see how the first *NSIMPAR* elements of *X* are successively perturbed by an amount *Deltax*. Each is followed by a simulation and the calculation of the column of cost function and active constraint partial derivatives relating to the perturbed component. The remaining terms in the jacobian matrix are zeros. The most uncertain and critical part in this stage is the choice of *Deltax*. Guidelines and discussion pertaining to its setting are located in Appendix 2. We also note here that the computational work required at this stage is dependent on *NSIMPAR* and the number of frame elements. It is relatively insensitive to the number of constraints within each loadcase[1].

(3) A quadratic program is now called to find the required direction vector. In the simple example for one active constraint and a cost function shown in Figure(1), vectors $\underline{a}$ and $\underline{b}$ are the directions of maximum decrease for the constraint and objective function respectively. A line segment joining the ends of these vectors is constructed and the direction vector is chosen so that it is parallel to the line of shortest distance between the origin of the constraint function gradients and the line segment. Figure(2) shows the more general case where the cost function and numerous active constraints influence the direction vector calculation. A region bounded by the ends of the individual gradient

[1] The exception is when no constraints are active. A simulation will not be done for such cases.

vectors, known as the convex hull, is defined. The direction vector chosen is parallel to the shortest line segment joining origin of the constraint gradients and the convex hull. In both examples the direction vector points away from the active constraint boundaries while also leading to a local decrease in the cost function. This numerical strategy is adopted to inhibit the design vector zig-zagging against nonlinear constraints during the initial stages of convergence. The active constraint width, *Eps*, may be reduced during the latter stages of optimization to allow the design vector to move against a constraint.

The direction vector is now scaled with respect to its Euclidean norm. Finally, angles between the cost function, directions of maximum active constraint increase and the direction vector are calculated. The 'user' should check that these angles lie in the range 90^0 - 180^0 and are preferably not close to 90^0. It should be apparent from Figure(2) that the algorithm will work best when the directions of maximum active constraint and cost function decrease are all parallel, thus forming an angle of 180^0 with respect to the negative direction vector. The jacobian matrix will have rank = 1 in such cases. If the angles calculated are close to 90^0, only a small decrease in maximum constraint violation can be expected. The method of feasible directions fails when the origin of the constraint function gradients lies within the convex hull. In such cases, it is impossible to satisfy the aforementioned condition of all angles having to be greater than 90^0.

2.2.2 Armijo step length

The Armijo step length is a simple numerical procedure that finds the approximate minimum of a function along the direction vector. If the design vector is initially infeasible, this minimum is such that a maximum reduction in greatest constraint violation is achieved. If the vector is otherwise feasible, a step length is chosen to locally minimize the cost function without entering the infeasible domain. A mathematical description of this procedure is presented in Bhatti[3] ; Section 5.3.

In DELIGHT.STRUCT this calculation is carried out in the rattle procedure *armijo.* The reader can easily verify that the new trial vector is:

xnew = X + Dist * direct( )

where: xnew = new trial design vector.
X = present iteration design vector.
direct( ) = direction vector.
Dist = armijo step length scaling factor.

The parameter *Dist* takes the value *Diststart* at the beginning of the first iteration. The default is set at 0.2. If the first trial step length satisfies all constraint reduction or objective function requirements, *Dist* is premultiplied by a factor β^{-1}, successively increasing the step length until it is maximized either by violating a constraint or exceeding the parameter *Distmax*. The more common case however is that a constraint violation or objective function increase will occur at the first time step. The step length is decreased by multiples of β until either all requirements are satisfied, or the default of *Maxiter* trials is reached. This case is shown in Figure(3). Although the default for *Maxiter* is set at 50, resimulation of a new design vector will not occur when the maximum design vector element discrepancy between the present and previous stored response is less than *Tolx*[2].

When this routine is entered for subsequent iterations, the starting value of *Dist* is equal to the final value taken at the end of the previous iteration. Further guidelines to the setting of other parameters in *armijo* are presently unavailable.

2.5 Running Long Jobs

It is considerate to use the *nice* command which gives priority to other users in peak load time if a lengthy job is being run. The following example demonstrates how this is utilized when DELIGHT.STRUCT is started.

```
% nice +2 DELIGHT.STRUCT '<memframe>'
```

This is equivalent to giving the command *go '<memframe>'* except that when the *nice* command is utilized, the alias *go* has to be replaced by DELIGHT.STRUCT. The computer will automatically default the program to *nice +4* after ten minutes of run time if it has not

[2] See *Maxiter* in Appendix 2 for an approximate guideline to a more appropriate setting for this parameter.

specified. A job will be completed at constant priority if *nice* has been specified. Therefore, it may be more efficient to run a complete task at *nice +2.*

The second useful tool is the pajama mode as outlined in Balling et al.[1], Section 3.2.2. A file *pajama* is created that contains various define statements. For example, it might include:

```
stop_opt
simulate
fprint
saveall
quit
```

The implementation of *pajama* typically follows the setting up of the optimization algorithm. For example :

```
> pajama
> sleep 10000 ; run 3
```

The program suspends for 10000 seconds. The file *pajama* is accessed following 3 iterations of optimization. Each command is sequentially completed and the program is held in a suspended state in the memfile *memprob.* Further interaction with the program is possible following the commands ;

```
% go '<memprob>'
> fonward
> start_opt
```

3. EXAMPLE PROBLEM.

3.1 Geometry and Modelling

A ten story, single bay friction braced steel frame as introduced by Workman[4] is utilized as the 'example' frame for this report. Its geometry and starting section sizes are shown in Figure(4). Each beam and column is represented by its moment of inertia and the friction braces by their cross-sectional area. Section 2.2.3 of Balling et al.[2] discusses the lumped-plasticity parallel component modelling assumptions used by ANSR. The braces are assumed to be constructed with mechanical devices that offer zero resistance to compression. Incipient buckling is therefore prevented. In tension they deform in an elasto-plastic manner, acting to brake the motion, redistribute member forces throughout the frame and dissipate a large percentage of the earthquake input energy. The overall effect is an enhanced structural performance when compared to either the purely moment-resistant frame or the traditional concentrically braced frame.

The element numbers assigned by the frame preprocessor *finput* are shown in Figure(5). Reference[8] comprehensively discusses the design philosophy adopted for these frames. The guidelines of this approach discourage column plastic hinges except at the base of the frame. The preliminary calculations indicated no column hinging for the given earthquake loading. Thus, for the purposes of the investigation, it was decided to retain this desirable feature throughout the optimization process by fixing the columns at their starting inertias, and treating only the beams and braces as design variables. In addition, the beams and braces were subjectively grouped as shown in Figure(6). Repetition of this type is essential for economical construction. Consequently, in the modelling of this frame the design vector dimension has been reduced from a maximum of 60 to 18, of which only 8 correspond to element parameters and the final 10 to dummy story drift variables.

3.2 Earthquake Loading.

The earthquake record chosen for this study is a scaled version of the El Centro N40W component that occurred on October 15, 1979. It is the E6 record employed by Balling. A discussion of the scaling procedure used is in Balling et al.[2], Section 2.1.3.

4. DISCUSSION.

The user should be aware of the following limitations associated with the performance of DELIGHT.STRUCT.

1. DELIGHT.STRUCT uses a Newton method to calculate the constraint and cost function gradients. Convergence to a global minimum is not guaranteed and in general convergence to a local minimum will occur.
2. The starting design should be as close as possible to the expected optimal design. This increases the possibility of converging to the global minimum. If this is not the case, the 'user' should be prepared to use engineering judgement to over-ride the algorithm and position the design vector as close as possible to the anticipated final design. The design algorithm will not produce miracles.
3. The method of feasible directions will fail when the origin of the constraint and cost function gradients lies within the convex hull[1]. For example, if the Workman frame is modelled with zero percent Rayleigh damping, both girder yield moment under moderate earthquake and girder end energy dissipation constraints will be violated. The former constraint requires that girder inertias be decreased to attract less bending moment. The latter constraint implies that girder size be increased to make the frame less flexible. A conflict in requirements results and the feasible directions algorithm approach is inapplicable.

[1] Also see section 2.2.1.

5. REFERENCES.

1. R.J. Balling, K.S. Pister, and E. Polak, "DELIGHT.STRUCT: A Computer-Aided Design Environment for Structural Engineering", Report No EERC 81-19, Earthquake Engineering Research Center, Univ of Cal, Berkeley, Ca, December, 1981.
2. R.J. Balling,V. Ciampi,K.S. Pister, and E. Polak, "Optimal Design of Seismic-Resistant planar Steel Frames", Report No EERC 81-20,Earthquake Engineering Research Center,Univ of Cal,Berkeley,Ca,December,1981.
3. M.A. Bhatti," Optimal Design of Localized Nonlinear Systems with Dual Performance Criteria under Earthquake Excitations", Report No 79-15, Earthquake Engineering Research Center, Univ of Cal, Berkeley,Ca,July,1979.
4. G.H. Workman, "The Inelastic Behavior of Multistory Braced Frame Structures Subjected to Earthquake Excitation", Research Report,University of Michigan,Ann Arbor; Sept,1969.
5. A.S. Pall,C. Marsh,"Response of Friction Damped Braced Frames", Journal of the Structural Division, ASCE, pp 1313-1323, Vol(108), No ST6, June 1982.
6. N.D. Walker, "Automated Design of Earthquake Resistant Multistory Steel Building Frames", Report No 77-12,Earthquake Engineering Research Center,Univ of Cal,Berkeley,Ca,May,1977.
7. W.T. Nye,A.L. Tits, "Delight For Beginners", ERL memo no UCB/ERL M82/55 University of California,Berkeley,July,1982.
8. M.A. Austin,K.S. Pister, "Optimal Design of Friction-Braced Frames under Seismic Loading". Report No UCB/EERC-83/10 Earthquake Engineering Research Center, Univ of Cal, Berkeley, Ca, June, 1983.

APPENDIX 1.

This appendix contains the DELIGHT.STRUCT echo output for the Workman frame minimum volume design. It was initiated by giving the rattle command *echo_io_to Out*, whence all terminal input and output has been echoed to the file *Out.* The intention of this section is twofold. It is included to show the 'user' in a simple manner the step-by-step procedure needed to complete an optimal design. The second purpose is to provide a supplement to the explanation of terms within the Appendix 2. Notes are appended at the end of this section to highlight some of the less obvious pitfalls to be wary of when using DELIGHT.STRUCT. These are denoted by a **( )** in the output.

Getting started

The 'user' should be in the directory *framework.d*. The computation phase is initiated with the command;

```
% go '<memframe>'
```

```
>>
>> # Workman frame : 5% damping : 20th November
>>
>> terminal 2648
>> finput
INTERACTIVE GRAPHICAL INPUT OF GEOMETRY AND LOADS
  (you should be on a color terminal, Sir)
  (units are inches, kips, and seconds throughout)

Type the number of stories:
>> 10
Type the number of bays:          **(1)**
>> 1
Type story heights from bottom to top (one per line):
>> 144
>> 144
>> 144
>> 144
>> 144
>> 144
>> 144
>> 144
>> 144
>> 144
Type bay widths from left to right (one per line):
>> 240
--------------------------------------------------------
From this point on the user inputs lists of various data.
Each line of the list may contain one or more elements.
The list is terminated by a line consisting of a single zero.
While making a list, errors may be undone. This is done by
  typing the negative of the first element of the bad line.
--------------------------------------------------------
Type a list of story and bay numbers of braced panels:
>> 1 1
>> 2 1
>> 3 1
>> 4 1
>> 5 1
>> 6 1
>> 7 1
>> 8 1
>> 9 1
>> 10 1
>> 0
Type a list of numbers of elements to be erased:
>> 0
Type a list of numbers of nodes to be erased:
>> 0
Type a list of shear girder element numbers:
>> 0
Type a list of dissipator element numbers:
>> 0
Type a list of rubber bearing element numbers:
```

```
>> 0
Type a list of loaded girder numbers with their uniform load:
>> 21 .3
>> 22 .3
>> 23 .3
>> 24 .3
>> 25 .3
>> 26 .3                    **(2)**
>> 27 .3
>> 28 .3
>> 29 .3
>> 30 .3
>> 0
Type a list of loaded node numbers with their point loads:
>> 0
Type a list of constrained node numbers with their code:
  (code is 3-digit integer where digits represent hori-disp,
   vert-disp, rotation with 0=free, 1=constrained)
  (default code is 111 for the base nodes):
>> 0
Type a list of numbers of elements not subject to design:
>> 1
>> 2
>> 3
>> 4
>> 5                       **(3)**
>> 6
>> 7
>> 8
>> 9
>> 10
>> 11
>> 12
>> 13
>> 14
>> 15
>> 16
>> 17
>> 18
>> 19
>> 20
>> 0
Type a list of numbers of elements not subject to constraint:
>> 0                       **(4)**
Type a list of groups of elements constrained equal during design
  (end each line with a zero):
>> 1 2 0
>> 3 4 5 6 0
>> 7 8 9 10 0
>> 11 12 13 14 0               **(5)**
>> 15 16 17 18 0
>> 19 20 0
>> 21 0
>> 22 23 24 25 0
>> 26 27 28 29 30 0
>> 31 32 33 34 0
>> 35 36 37 38 0
>> 39 40 41 42 0
>> 43 44 45 46 0
>> 47 48 49 50 0
>> 0
Type a list of element numbers with their initial design values
   (moments of inertia for columns and girders, areas for braces
    thickness for truss girders, and edge lengths for bearings)
   (elements are systematically erased as values are specified)
   (default values are the max values for the elements):
>> 1 1500
>> 3 1373
>> 7 1165
>> 11 851
>> 15 542
>> 19 339
>> 21 1287
>> 22 984
>> 26 800
>> 31 3.38
>> 35 2.94
>> 39 2.88
>> 43 2.88
>> 47 2.88
>> 0                          **(6)**
>>
>> fstartoff
Input cost function coefficients for:
  volume of designed elements:
>> 0.00002                    **(7)**
  sum of squares of moderate quake story drifts:
>> 0
  energy input from severe quake:
>> 0
  inelastically dissipated energy from severe quake:
>> 0
  fuse dissipated energy from severe quake:
>> 0
  column dissipated energy from severe quake:
>> 0
PARAMETER:  Cosvol = Cosvol : volume cost coefficient
PARAMETER:  Cosdri = Cosdri : drift cost coefficient
PARAMETER:  Volmax = MAXREAL : max structural volume
PARAMETER:  Drift = .005 : max moderate story drift
PARAMETER:  Accel = .5 : max moderate floor accel in gs
PARAMETER:  Sway = .01 : max severe structure sway
PARAMETER:  Colax = .5 : gravity column axial force factor
PARAMETER:  Colgra = .6 : gravity column yield factor
PARAMETER:  Colyld = 1 : moderate column yield factor
PARAMETER:  Colduc = 3 : severe column ductility
PARAMETER:  Girgra = .6 : gravity girder yield factor
```

```
PARAMETER:  Girdef = .00417 : gravity girder midspan deflection
PARAMETER:  Giryld = 1 : moderate girder yield factor
PARAMETER:  Girduc = 6 : severe girder ductility
PARAMETER:  Bragra = .6 : gravity brace yield factor
PARAMETER:  Brayld = 1 : moderate brace yield factor
PARAMETER:  Braduc = 25 : severe brace ductility
PARAMETER:  Tolx = 0.0001 : tolerance for resimulating
PARAMETER:  Numsim = 0 : number of simulations
PARAMETER:  Numsteps = 0 : number of simulation time steps
PARAMETER:  Numref = 0 : number stiffness reformulations
PARAMETER:  Maxerr = 0.0 : maximum energy error ratio
PARAMETER:  Print = 1 : simulation printout code
Type "1" if a response file exists, otherwise type "0":
>> 0              **(8)**
---------- Next CRUNCH is forced. ----------
CRUNCH

>>
>> # reset parameters
>>
>> Braduc = 75        **(9)**
>> Sway = 0.015
>> Tolx = 0.00005
>> Volmax = 70000     **(10)**
>>
>> simulate           **(11)**
begin simulation...
end simulation...
begin simulation...
end simulation...
begin simulation...
end simulation...
>>
>>
>> fprint
Created file "frame"
>> list frame
-------------------- Begin frame --------------------
Values Of Different Terms In The Cost Function:
   Volume of designed elements   = 5.342e+4
   Sum of squares of story drifts = 1.012e-4
   Severe quake input energy     = 2.666e+3 **(12)**
   Severe quake dissipated energy = 2.030e+4
   Fuse dissipated energy        = 0.000
   Column dissipated energy      = 0.000

Element Section Design Variable Values:
  X1 = moment of inertia   = 1287
  X2 = moment of inertia   = 984
  X3 = moment of inertia   = 800
  X4 = brace section area  = 3.380
  X5 = brace section area  = 2.940
  X6 = brace section area  = 2.880
  X7 = brace section area   =  2.880
  X8 = brace section area   =  2.880

Dummy Variables : reset only for minimum story drift design
  X9 =  1.000,  should be = -.6729
  X10 =  1.000,  should be =  7.053e-2
  X11 =  1.000,  should be =  .3329
  X12 =  1.000,  should be =  .3243
  X13 =  1.000,  should be =  .1151
  X14 =  1.000,  should be =  8.305e-2
  X15 =  1.000,  should be = -.1641
  X16 =  1.000,  should be = -.3735
  X17 =  1.000,  should be = -.7174
  X18 =  1.000,  should be = -.8989
-------------------- End frame --------------------
>>
>> # resetting dummy parameters to just less than 1.0
>>
>> X(9) = 0.98
>> X(10) = 0.98
>> X(11) = 0.98           **(13)**
>> X(12) = 0.98
>> X(13) = 0.98
>> X(14) = 0.98
>> X(15) = 0.98
>> X(16) = 0.98
>> X(17) = 0.98
>> X(18) = 0.98
>>
>> simulate             **(14)**
>>
>> printstate
Created file "state"
>> list state
-------------------- Begin state --------------------
*** COST =  1.068         **(15)**
   PSI  = -9.996e-2   IPSI = 46 JPSI = 293

   INEQ(1): percent = 76
   INEQ(2): percent = 52  **(16)**
   INEQ(3): percent = 52

   ----lines deleted----

   INEQ(189): percent = 20
   INEQ(190): percent = 12
   INEQ(191): percent = 12

   FINEQ(1,386): percent = 35
   FINEQ(2,385): percent = 18
   FINEQ(3,279): percent = 35

   ----lines deleted----
   FINEQ(99,286): percent = 27
   FINEQ(100,288): percent = 28
   FINEQ(101,386): percent = 17
   FINEQ(102,388): percent = 54
   FINEQ(103,291): percent = 67
   FINEQ(104,294): percent = 67
   FINEQ(105,296): percent = 56  **(17)**
   FINEQ(106,298): percent = 55
   FINEQ(107,300): percent = 42
   FINEQ(108,302): percent = 32
   FINEQ(109,303): percent = 14
   FINEQ(110,305): percent = 5
   FINEQ(111,301): percent = 69
-------------------- End state --------------------
>>
>>
>>
>> optimize
Which optimization algo do you wish to use?
  0 = user supplied
  1 = feasible directions
  2 = conjugate gradient
  Type the corresponding number:
>> 1
PARAMETER:  Eps = 0.2 : active constraint width
PARAMETER:  Dist = 0.2 : step length
PARAMETER:  Delta = 1.0 : eps reduction control
PARAMETER:  Epstol = 0.00001 : convergence tolerance on eps
PARAMETER:  Psitol = 0.00001 : convergence tolerance on psi
PARAMETER:  Geomslope = 100.0 : geometric constraint slope
PARAMETER:  Alpha = 0.2 : cost or psi reduction angle
PARAMETER:  Beta = 0.5 : step length search factor
PARAMETER:  Distmax = 1.0 : maximum step length
PARAMETER:  Scale = 1.0 : scale for active constraints
PARAMETER:  Gamma = 1.0 : infeasible cost penalty
PARAMETER:  Maxiter = 50 : maximum number of armijo loops
Type "1" if you have supplied a gradient scheme, otherwise type "0":
>> 0
PARAMETER:  Deltax = 0.0002 : gradient perturbation
What is the maximum number of iterations you expect to complete?
>> 20          **(18)**
ERROR: Echo output has already been set!
>>             **(19)**
>> # Modify design paramters
>>
>> Eps = 0.25
>> Deltax = 0.0001 **(20)**
>> Tolx = 0.00005
>> Maxiter = 20   **(21)**
>>
>> pajama         **(22)**
>> sleep 10;run 5
PARAMETER   SOURCE FILE        VALUE   DESCRIPTION
Cosvol      <Fstartofffile>   2.000e-5  volume cost coefficient
```

```
Cosdri      <Fstartofffile>     0.000  drift cost coefficient
Volmax       <Fstartofffile>  7.000e+4  max structural volume
Drift       <Fstartofffile>  5.000e-3  max moderate story drift
Accel       <Fstartofffile>     .5000  max moderate floor accel in gs
Sway        <Fstartofffile>  1.500e-2  max severe structure sway
Colax       <Fstartofffile>     .5000  gravity column axial force fac
                              tor
Colduc       <Fstartofffile>    3.000  severe column ductility
Girgra      <Fstartofffile>     .6000  gravity girder yield factor
Girdef      <Fstartofffile>  4.170e-3  gravity girder midspan deflect
                              ion
Giryld      <Fstartofffile>     1.000  moderate girder yield factor
Girduc       <Fstartofffile>    6.000  severe girder ductility
Bragra      <Fstartofffile>     .6000  gravity brace yield factor
Brayld      <Fstartofffile>     1.000  moderate brace yield factor
Braduc       <Fstartofffile>  7.500e+1  severe brace ductility
Tolx        <Sansrsimfile>   5.000e-5  tolerance for resimulating
Numsim        <Sansrsimfile>     3.000  number of simulations
Numsteps      <Sansrsimfile>  2.203e+3  number of simulation time steps
Numref       <Sansrsimfile>  2.329e+3  number stiffness reformulations
Maxerr       <Sansrsimfile>  2.479e-5  maximum energy error ratio
Print       <Sansrsimfile>     1.000  simulation printout code
Eps         <Afeasdirfile>     .2500  active constraint width
Dist        <Afeasdirfile>     .2000  step length
Delta       <Afeasdirfile>     1.000  eps reduction control
Epstol      <Afeasdirfile>  1.000e-5  convergence tolerance on eps
Psitol      <Afeasdirfile>  1.000e-5  convergence tolerance on psi
Geomslope     <Afeasdirfile>  1.000e+2  geometric constraint slope
Alpha        <Afeasdirfile>     .2000  cost or psi reduction angle
Beta        <Afeasdirfile>     .5000  step length search factor
Distmax      <Afeasdirfile>     1.000  maximum step length
Scale       <Afeasdirfile>     1.000  scale for active constraints
Gamma         <Afeasdirfile>     1.000  infeasible cost penalty
Maxiter      <Afeasdirfile>  2.500e+1  maximum number of armijo loops
Deltax      <Sperturbfile>  1.000e-4  gradient perturbation

ITER = 0, COST = 1.068, PSI = -9.996e-2
Column X(18):
 -2.479300e-3
 -.2528926
 -.4049587
 -.3936842   **(23)**
 -.4863158
 -.4989474
 -.4989474
 -.4989474
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
 .9800000
 .9800000
--------simulation output-------
number of simulations = 3
number of time steps = 2203
number of stiffness reformulations = 2329
maximum energy error ratio =  2.479e-5

begin simulation...
end simulation...
begin simulation...

----lines deleted----

end simulation...
begin simulation...
end simulation...


ITER = 1, COST =  .9888, PSI = -5.756e-2
Column X(18):
 -2.225102e-2
 -.3440237
 -.5320309
 -.4487421
 -.5413737
 -.5540053
 -.5540053
 -.5540053
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
--------algorithm output--------
Eps =  6.250e-2, theta = -7.463e-2, Dist =  .2000
direction finding phase:
Column direct(18):
 -9.885861e-2
 -.4556555      **(24)**
 -.6353608
 -.2752896
 -.2752896
 -.2752896
 -.2752896
 -.2752896
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
   COST  norm = .3863  angle = 1.800e+2
Matrix jacobian(1,18):
  3.819347e-2  .1760399    .2454681    .1063566    .1063566    .1063566
     .1063566    .1063566    0.000000    0.000000    0.000000    0.00
0000    0.000000    0.000000    0.000000    0.000000    0.000000
0.000000
steplength determination phase:
   first try in armijo did it
   nothing was violated
   constraints violated
--------simulation output-------
number of simulations = 25
number of time steps = 12849
number of stiffness reformulations = 13198
maximum energy error ratio = 2.596e-5


begin simulation...
end simulation...
begin simulation...

----3 iteration outputs deleted----

begin simulation...
end simulation...
begin simulation...
end simulation...


ITER = 5, COST = .9435, PSI = -1.585e-4
Column X(18):
 -3.138512e-2
 -.3930830
 -.6066237
 -.4740846
 -.5666297
 -.5816874
 -.5812434
 -.5829577
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
  .9800000
--------algorithm output--------
Eps = 3.125e-2, theta = -3.557e-2, Dist = 7.813e-4
direction finding phase:
Column direct(18):
  .2491448
 -.4857420
 -.4460768
  .2371166
 -.2725755
 -.5000144
  .2808440
 -.2087765
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  0.000000
  COST norm = .4293 angle = 1.284e+2 **(25)**
  FINEQ(51,398) norm = 8.381 angle = 9.138e+1
  FINEQ(52,304) norm = 1.479e+1 angle = 9.315e+1
Matrix jacobian(3,18):
 3.875706e-2  .1944792    .2973475    .1063566    .1063566    .1063566
    .1063566    .1063566   0.000000    0.000000    0.000000    0.00
0000    0.000000    0.000000    0.000000    0.000000    0.000000
0.000000
 -2.671077   -1.518632   -4.352461   -4.272515   -.7830400    .7740612
   -4.571883   -1.219823   0.000000    0.000000    0.000000    0.00
0000    0.000000    0.000000    0.000000    0.000000    0.000000
0.000000
 -5.004753   -2.841162   -7.068334   -7.655132   -1.350487    1.641632
   -8.206798   -2.266458   0.000000    0.000000    0.000000    0.00
0000    0.000000    0.000000    0.000000    0.000000    0.000000
0.000000
steplength determination phase:
  armijo loop was decreasing
  constraints violated
  constraints violated
  nothing was violated
--------simulation output-------
number of simulations = 93
number of time steps = 51973
number of stiffness reformulations = 53805
maximum energy error ratio = 2.596e-5


Interrupt...     **(26)**
```

```
>> # I'm back !!!
>> # Continuation after 5 iterations
>>
>> display_time  **(27)**
 TOTAL   DIRECT  NUMBER
SECONDS SECONDS OF CALLS PROCEDURE/MACRO NAME
7.64e+4 -------  ---  Cpu-time since last "clear_time"
6.73e+4 2.12e+1  142   ansrsim
5.97e+4 5.97e+4   90   analys
4.08e+4 3.55e+3   15   state
4.05e+4 4.08      5   armijo
3.55e+4 5.18e+3    8   actgrad
3.02e+4 3.63      8   sensitivity
3.02e+4 .100      7   gradients
3.02e+4 1.66e+1    7   perturb
7.62e+3 7.62e+3  144   resman
1.97e+2 1.24e+1  3161   functional
>>
>> # find present state
>>
>> fonward
>> fprint
Created file "frame"
>> list frame
------------------- Begin frame -------------------
Values Of Different Terms In The Cost Function:
   Volume of designed elements    = 4.717e+4
   Sum of squares of story drifts = 1.178e-4
   Severe quake input energy      = 2.587e+3
   Severe quake dissipated energy = 1.819e+4
   Fuse dissipated energy         = 0.000
   Column dissipated energy       = 0.000

Element Section Design Variable Values:
  X1 = moment of inertia   = 1252
  X2 = moment of inertia   = 814
  X3 = moment of inertia   = 556
  X4 = brace section area  = 2.998
  X5 = brace section area  = 2.559
  X6 = brace section area  = 2.487
  X7 = brace section area  = 2.489
  X8 = brace section area  = 2.481

Dummy Variables : reset only for minimum story drift design
  X9 = .9800, should be = -.7027
  X10 = .9800, should be = 7.732e-2
  X11 = .9800, should be = .4389
  X12 = .9800, should be = .4499
  X13 = .9800, should be = .2650
  X14 = .9800, should be = .3409
  X15 = .9800, should be = .1633
  X16 = .9800, should be = -.1410
  X17 = .9800, should be = -.6056
  X18 = .9800, should be = -.8618
```

```
-------------------- End frame --------------------
>>
>> printstate
Created file "state"
>> list state
------------------- Begin state -------------------
   COST = 9.435
   PSI = -8.663e-4   IPSI = 111 JPSI = 305

   INEQ(1): percent = 67
   INEQ(2): percent = 52

   ----lines deleted----

   INEQ(190): percent = 13
   INEQ(191): percent = 13

   FINEQ(1,387): percent = 33
   FINEQ(2,386): percent = 17

   ----lines deleted----

   FINEQ(109,309): percent = 20
   FINEQ(110,311): percent = 7
   FINEQ(111,305): percent = 72
-------------------- End state --------------------
>>
>> # Set up data file for plotting
>>
>> plot_fenergy
Which energies do you wish to see (max=7)?
  1 = quake energy      2 = work done by loads
  3 = elastic energy    4 = inelastic energy
  5 = damped energy     6 = kinetic energy
  7 = error energy
Type the corresponding numbers on a single line.
>> 4
>>
>> quit     **(28)**
```

Notes

(1) The maximum allowable product (bays $\times$ stories) is 12.

(2) The uniform load is divided into dead and live load components. The live uniform load to total uniform load ratio is set in the file *assumptions.*

(3) Elements not subject to design will retain the value assigned in the last part of *finput.*

(4) Note that although element nos 1-20 are not subject to design, they are subject to constraint. Therefore, space in the array *Resp* is allocated for all elements in this example. If an element isn't subject to constraint, space for it will not be allocated in *Resp,* even though it could be subject to design.

(5) Elements subject to both equal constraint and design share the same design variable. Consequently, in this example there are only 8 design parameters and 10 dummy story drift parameters. During the optimization process each design variable will be guided by the most critical constraint within the element group.

(6) The files *sizefile, xfile, description, ansrdata1, ansrdata2,* and *ansrdata3* are now created. Example files are shown in Balling et al.[1], Appendix 6.

(7) The volume cost coefficient is chosen so the value of *COST* will be of the same order as the design variables.

(8) If a file response did exist at this stage, typing 1 would cause it to be copied into the array *Resp.*

(9) A brace ductility was chosen at this stage so that energy and section area restrictions would not be active constraints.

(10) If volume is not being minimized, it is wise to constrain its value.

(11) At this stage no previous response exists. A full 3 part simulation is needed. See *simulate* in Appendix 2.

(12) See Balling et al.[2], Section 2.4.1 for a discussion of the energy formulation.

(13) Read *fprint* in Appendix 2 for an explanation of why the story drift parameters are being modified.

(14) A full resimulation is not required because the first *NSIMPAR* elements of the present design vector differ by less than *Tolx* from the previous design vector. The constraints are calculated using the previous response.

(15) The *COST* function is of the same order as the design variables. As *PSI* is less than 0, the starting design vector is within the feasible domain. *IPSI* is the number of the maximum constraint violation and *JPSI* is the timestep at which it occurs.

(16) Look at the file *description* to see how the constraints have been ordered.

(17) The file *description* indicates that functional constraint nos 101-110 correspond to story drift constraint violations. They are not close to 100% and will not be included in the direction vector calculation unless *Eps* is greater than 0.32.

(18) Array dimensions for the file *history* are set here.

(19) Recall that an echo of terminal input and output has already been directed to the file *Out*. Otherwise it would have automatically been directed to the file *dialogue* from this point on.

(20) See comments on *Deltax* in Appendix 2.

(21) See comments on *Maxiter* in Appendix 2.

(22) See Balling et al.[1], Section 3.2.2 and Section 2.5 of this report for information on the *pajama* command.

(23) Note that all elements in the design vector are within the range [-1,1].

(24) When story drift is not being minimized only the first *NSIMPAR* elements of the direction vector are nonzero.

(25) Only a very small decrease in the cost function is expected at the 5th iteration. This is accompanied by close to perpendicular angles between the direction of maximum cost function decrease and directions of maximum descent for the active constraints.

(26) The *pajama* file is accessed at this point.

(27) You should see Section 10 of Delight for Beginners[7] for an explanation of this command. The command *savehist* could also be given at this stage, generating the file *history* containing a summary of iteration results.

(28) The *quit* command returns the user to UNIX.

APPENDIX 2.

This appendix contains a summary of files, defines, and interactive variables often employed in the running of DELIGHT.STRUCT. Its purpose is to help the 'user' interpret terms. It should be used in conjunction with Appendix 1 of this report and Appendices 1 and 2 of Balling et al.[1]. Terms omitted from this list will be located in Appendix 2 of reference [1].

1. DEFINES

finput	This define activates the frame pre-processor *Finput.* Its capabilities are discussed in Section 3.3.1 of Balling et al.[1] and an example of its use is in Appendix 1. After the 'user' has finished inputing his/her problem the files *xfile, sizefile,* and *description* are created. In addition, because the frame preprocessor is being used, the ANSR simulation input files *ansrdata1, ansrdata2,* and *andrdata3* are also generated at this stage.
fonward	Recalls parameters from the rattle file *<Cstartofffile>*. It also reads the previous ANSR simulation response from the binary file *response* into the array *Resp.* We note that this define is equivalent to giving the command *fstartoff* and typing a 1 when asked if a response file exists.
fprint	This define prints information on the cost function and design variables to the file *frame.* The following guidelines should be followed in modifying the design vector.
(1)	If the file frame suggests that variables be reset to values outside the range [-1,1], the parameter *Drift* is too small. Recall that convergence at the armijo step length stage will not occur for values of *abs(X)* $\geqslant$ *1.0.* Consequently, *Drift* should be increased until all the suggested story drift parameters have values less than 1.

(2) If story drift is being minimized, the dummy parameters should be set to a value just larger than the value suggested by *fprint.* This has the effect of making the story drift parameters active in the direction gradient calculations; ie , % constraints in the file *state* should now be in the range 95-99% . The 'user' is cautioned against resetting the dummy parameters to the exact values suggested in the file *frame.* These are only given to 4 decimal places and a small residual error in *PSI* could accidentally put the design vector in the infeasible domain.

(3) These constraints should not be included in the gradient calculation if story drifts are not being minimized. The 'user' can ensure this by first resetting all the dummy design variables to just less than 1. This takes care of the aforementioned residual error problem. The second point to check is whether or not the percentage of constraint violations is higher than that defined to be included in the direction calculation. This is done by comparing the values in *state* with the criteria in procedure *actgrad* for choosing active constraints. We note :

Let psi* = max [PSI,0.0]
A constraint is chosen to be 'active' if

$$\textit{constraint}\ \% \geqslant [\,psi^* - Eps + 1\,] \cdot 100\ \%$$

If none of the story drifts are included, no adjustment is necessary. If some constraints are however picked up, they can be removed by either increasing the parameter *Drift,* or decreasing the starting active constraint band-width *Eps.*

fstruct Gives a screen display of the frame geometry as it is defined in the frame preprocessor.

graph The 'user' should see the latter section of Balling et al.[1], Appendix 5 for an example of the Rattle commands *erase* and *window_list* before proceeding to graph results on the terminal. The routines available for graphing on the screen are :

graph_cost	graph_felhyst
graph_elenergy	graph_fenergy
graph_elforce	graph_psi
graph_faccel	graph_time
graph_fdrift	graph_x
graph_fsway	graph_frecord

Sections 3.3.1 and 3.3.2 of Balling et al.[1] discuss the capabilities of each graph define.

onward — Reads the previous response from the file *response* into the array *Resp.*

optimize — Includes the file *Coptimizefile.*

pajama — This command is used when running long jobs. The 'user' should read the appropriate section of this report and the final paragraph of Section 3.2.2 of Balling et al.[1].

plot — Data files for report plots are generated by substituting the command *plot* in place of *graph.* These may be need to be converted to a non-exponent format to become compatible with plotter software.

run — Starts the optimization algorithm running for 'n' iterations.

saveall — This command stores all the values of DELIGHT variables into the memfile *memprob.* It then stores the contents of the last ANSR simulation response vector *Resp* into a file *response.* Finally, the contents of the current design vector *X* are put in the file *xfile.*

savehist — A concise summary iteration results is put in the file *history.* The *COST, PSI,* design vector *X* and cpu iteration times are available for each iteration. Array dimensions for this information storage are allocated at the end of the *optimize* command.

simulate — This define computes the cost and constraint functions via the rattle procedure *State.* The present design vector *X* is first compared to the previous vector. If the difference in each element is less than the variable *Tolx,* the response of the previous iteration is considered to be sufficiently close to the present, and no resimulation is required. The constraints are calculated and the present design vector is put in the file *xfile*. In most cases this test will fail and a 3 part simulation will be initiated to get the new response. As stated in Balling et al.[2]: Section 2.2.2, these are :

(1) A static nonlinear analysis made under gravity loads only.

(2) A static linear analysis under gravity loads followed by a dynamic linear analysis under moderate earthquake loading.

(3) A static nonlinear analysis under gravity loads followed by a dynamic nonlinear analysis under severe earthquake loading.

Further information on the numerical time stepping method and the choice of damping model is located in the aforementioned reference.

sleep	Causes DELIGHT.STRUCT to sit idle for 'n' seconds.
start_opt	Resumes the cpu-time measurement and echo to the file *dialogue*.
stop_opt	Suspends echo to a file and cpu-time measurement.

2. INTERACTIVE COMMANDS.

Deltax	Each component in the design vector *X* is successively perturbed by *Deltax* to obtain the active constraint and objective function partial derivatives in the direction vector calculation. The truncation error in the derivative calculation is Order(Deltax), so it should be chosen as small as possible without entering the realm of numerical roundoff. If it is set too large then gradient calculation errors may lead to an increase in *COST* or *PSI* along the direction vector. This is most likely to happen after several iterations when the expected change in *COST* or constraint violation approaches zero. In the example problem, *Deltax* = 0.001 leads to an increase along the direction vector. *Deltax* = 0.00001 is recommended.
Eps	The active constraint band-width, *Eps,* is used to decide which constraints are active in the direction vector calculation. The criteria used by the rattle procedure *actgrad* in making this choice is discussed in Chapter 2.

Maxiter This variable limits the number of trial step lengths in the Armijo step calculation. The default is set at 50; however, it could be interactively reduced so that the following inequality is only just satisfied.

$$\beta^{[\,Maxiter\ -1\,]} \geqslant \left[\frac{Tolx}{Dist} \right]$$

Print This DELIGHT variable is set by default to 1. However, if this is interactively set to 0, followed by the command simulate, a large response file *ansroutput* will be generated This contains nodal accelerations etc, that may be inspected if program debugging is required.

PSI If $g(X)$ is the value of the constraint function :

$$PSI = \left[\frac{g(X)}{g(allowable)} \right] - 1$$

Tolx Tolx is the tolerance for ANSR resimulation. It must be less than *Deltax.*

APPENDIX 3.

The common block *bigres* contains the array *resp* that stores the present structural response. The purpose of this section is to outline the dynamic array allocation as it applies to the Workman frame[4].

Figure(7) shows a simple schematic of the way *resp* is related to the main sections of DELIGHT.STRUCT. The reader should see Section 3.1.2. of Balling et al.[1] for a brief discussion of its contents. Modifications have since been made and these are now explained.

The original array *resp* was divided into two main sections containing the present and previous structural responses. Its maximum length was limited to 300000 double precision words. This was too small for the Workman frame, so the subroutine *resman* was modified to store the previous response in a binary file *savenl.* This doubled the response storage capacity. An additional array *rsto(50)* was also created to store the previous design vector. The 'user' can look at *resman* to see how the shuffling of data follows comparisons between present and past design vectors.

The length of this array may be changed by updating the common block *bigres* which is located in the directories *ansr.d, framefort.d,* and *element.d.* The "% grep -n bigres *" command can be used to find all the appropriate routines within each directory. The program will need to be recompiled and the reader is referred to Appendix 4 for the required procedure.

The storage order within the array depends upon the flags and lists read from the files *ansrdata1, ansrdata2,* and *ansrdata3.* Since *finput* is being used, the ANSR files are created by the subroutine *fandat,* which also automatically sets the flags and lists.

The array storage is divided into four blocks.

1. The simulation design variables. For the *example* problem this corresponds to the 8 design parameters.

2. The ANSR response according to *ansrdata1*. For the *example* problem this corresponds to the gravity load response.

3. The ANSR response according to *ansrdata2*. For the *example* problem this corresponds to the moderate earthquake response.

4. The ANSR response according to *ansrdata3*. For the *example* problem this corresponds to the severe earthquake response.

The variables *kstatic, kmoder,* and *ksever* determined within the subroutine *fcrnch,* indicate the beginning array numbers for blocks 2, 3, and 4 respectively. These are calculated at the *finput* stage and a check is made to ensure the required storage *NRESP* does not exceed 300000. Consequently, *fcrnch* must also be modified if this array length is being altered.

The storage order within blocks 2,3 and 4 is now explained. If the analysis is static each item stored (ie: beam end moment; etc) needs only one storage space. However, if the analysis is static/dynamic a complete response history each of length (1 + no time steps) is necessary for each item stored. Within each block the ordering is:

Number Of Items.	Description
1	zero
ISTENE * 7	Energies for the structure. (ISTENE = 1) Input. Work of Gravity Loads. Elastic. Inelastic. Damped. Kinetic. Error.

(ISTDIS + ISTVEL + ISTACC)*NODSX	X Direction nodal Displacements (ISTDIS = 1) and Velocities (ISTVEL = 1), Accelerations (ISTACC = 1) for the nodes listed in LNXDH(NODSX)
(ISTDIS + ISTVEL + ISTACC)*NODSY	Y Direction nodal Displacements (ISTDIS = 1) and Velocities (ISTVEL = 1), Accelerations (ISTACC = 1) for the nodes listed in LNYDH(NODSY)
(ISTDIS + ISTVEL + ISTACC)*NODSZ	Z Direction nodal Displacements (ISTDIS = 1) and Velocities (ISTVEL = 1), Accelerations (ISTACC = 1) for the nodes listed in LNZDH(NODSZ)

2* (Sum over beam-column elements of the number of non-zero digits in KOUTDT.)

For Beam-Column elements, storage depends on the flag KOUTDT for each element as follows

Axial force at both ends.	(KOUTDT contains the digit 1)
Plastic Rotation at both ends.	(KOUTDT contains the digit 2)
Energy Dissipation at both ends.	(KOUTDT contains the digit 3)
Moment at both ends.	(KOUTDT contains the digit 4)
Rotation at both ends.	(KOUTDT contains the digit 5)

(Sum over the truss elements the number of
non-zero digits in KTHO)

For the truss elements storage depends on the
flag KTHO for each element as follows.

Axial force. (KTHO contains the digit 1)

Axial Deformation. (KTHO contains the digit 2)

Energy Dissipation. (KTHO contains the digit 3)

The ANSR subroutine *storsp* stores the energies, nodal displacements, velocities and accelerations. Information pertaining to the braces and beam-columns is stored by the element subroutines *stor1* and *stor2* respectively.

We now consider the Workman frame. The following table summarizes the flags and lists set by the frame preprocessor.

FLAG OR LIST.	ANSRDATA1	ANSRDATA2	ANSRDATA3
ISTENE	0	0	1
ISTDIS	0	1	1
ISTVEL	0	0	0
ISTACC	0	1	0
NODSX	0	10	1
NODSY	0	0	0
NODSZ	0	0	0
LNXDH(NODSX)	0	3,5,7,9 11,13,15 17,19,21	21
LNYDH(NODSY)	-	-	-

LNZDH(NODSZ)	-	-	-
KOUTDT elements (1-20)	41	4	3
KOUTDT elements (21-30)	54	4	3
KOUTDT elements (31-50)	1	1	213

The layout of *Resp* for the *example* problem is now given.

DESCRIPTION	NO.	SUB-TOTAL
BLOCK 1 (Design Variables)		
Simulation design variables	8	8
BLOCK 2 (Gravity Loads)		
zero :	1	
For each Column :		
Axial Force I end		
Axial Force J end		
Moment I end		
Moment J end	4*20	
For each Beam :		
Moment I end		
Moment J end		
Rotation I end		
Rotation J end	4*10	

For each Brace :		
Axial Force	20	141

BLOCK 3 (Moderate Earthquake)

zero :	1101

For each story :

X - Displacement	
X - Acceleration	2*10*1101

For each column :

Moment	I end	
Moment	J end	2*20*1101

For each beam :

Moment	I end	
Moment	J end	2*10*1101

For each brace :

Axial Force	20*1101	111201

BLOCK 4 (Severe Earthquake)

zero :	1101
Energies :	7*1101
Top Floor X-Displacement :	1101

For Each Column :

Energy Dissipated	I end	
Energy Dissipated	J end	2*20*1101

For each Beam :

Energy Dissipated	I end	
Energy Dissipated	J end	2*10*1101

For each Brace :

Energy Dissipated		
Axial Force		
Axial Displacement	3*20*1101	253379

(Total Length) NRESP = 253379

```
Kstatic  = 1 + 8                      = 9
Kmoder   = 1 + 8 + 141                = 150
Ksever   = 1 + 8 + 141 + 111201 = 111351
```

APPENDIX 4.

All fortran routines should be compiled before recompilation begins. A typical example is % f77 -c -g fassum.f. The command ls-tl can then be given. The object code sources should come before the fortran. DELIGHT.STRUCT may not have to be recompiled if minor system modifications are made. For example, a modified Rattle routine can be included with either of the commands: include <filename> or use <filename>.

Otherwise, the procedure to be followed in recompiling DELIGHT.STRUCT is:

(1) Move to the directory *delight.d* and give the command % load.delight . If the response is DELIGHT.STRUCT(bad file memfile) the program needs to be recompiled and the command % make.memfile should then be given.

(2) Move to the directory *structrattle.d* and give the command % make.memstruct .

(3) Move to the directory *framerattle.d* and give the comand % make.memframe .

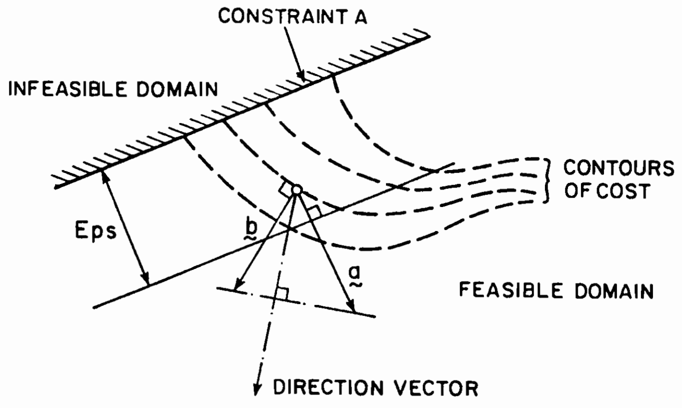


FIG (1) : Direction Vector Calculation for 1 Active Constraint and a cost function.

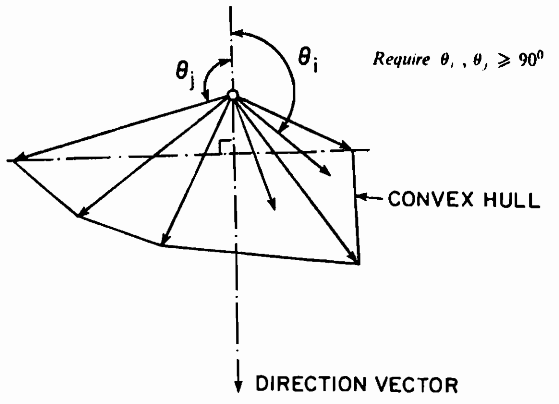
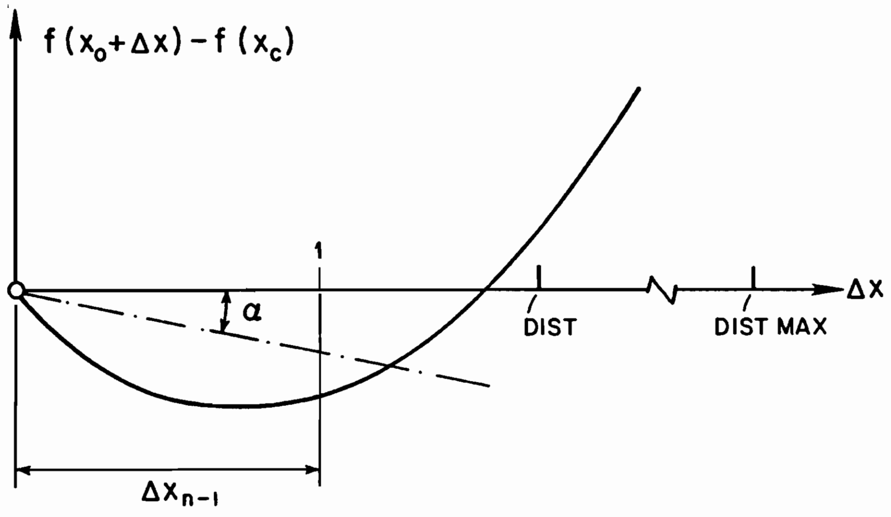


FIG (2) : Direction Vector Calculation for Multiple Active Constraints.



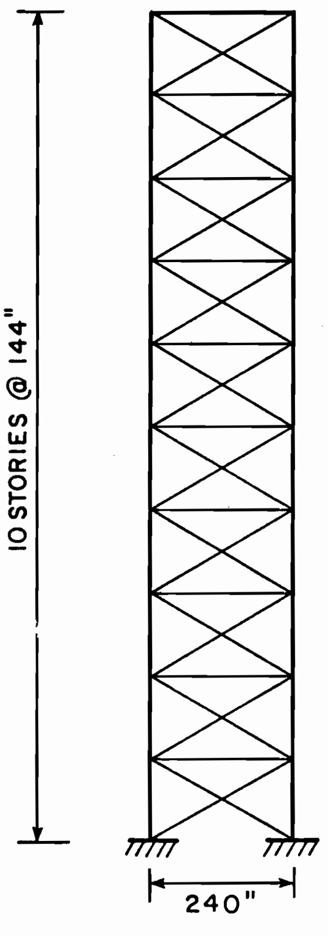
The Armijo Step Length is satisfied when

$$f(x_0+\Delta x_{n-1}) - f(x_0) \leqslant -\alpha \cdot \Delta x_{n-1}$$

where

x_0 =	*design vector.*
Δx_{n-1} =	$\beta^{n-1} \cdot$ *Dist* $\cdot$ *direct( )*
direct( ) =	*Direction vector.*
f( ) =	*Cost or Constraint function.*
Dist =	*Starting direction vector scaling factor.*
α =	*COST or PSI reduction angle.*
β =	*Step length line-search factor.*
n =	*nth trial at Armijo step length.*

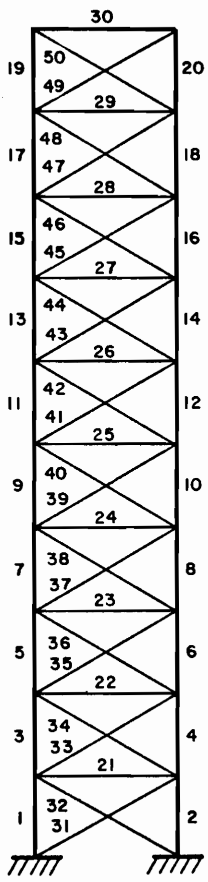
FIG (3) : Diagram of the Armijo Step Length.



BEAM INERTIA	COLUMN INERTIA	BRACE AREA
800		
	339	2.88
800		
	542	2.88
800		
	542	2.88
800		
	851	2.88
800		
	851	2.88
984		
	1165	2.88
984		
	1165	2.94
984		
	1373	2.94
984		
	1373	3.38
1287		
	1500	3.38

UNITS INERTIA IN^4
AREA IN^2

FIG (4) : Example Frame ; Starting Column and Girder Inertias ; Brace Cross Section Areas.



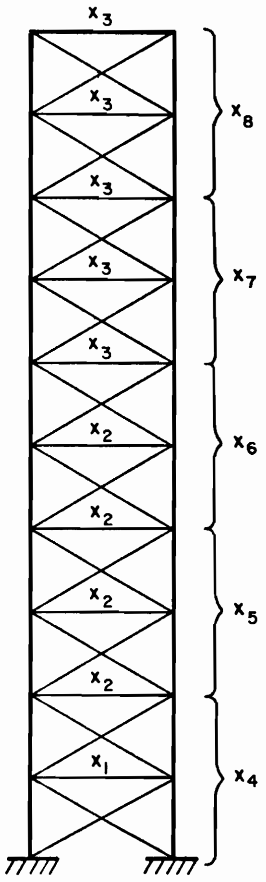
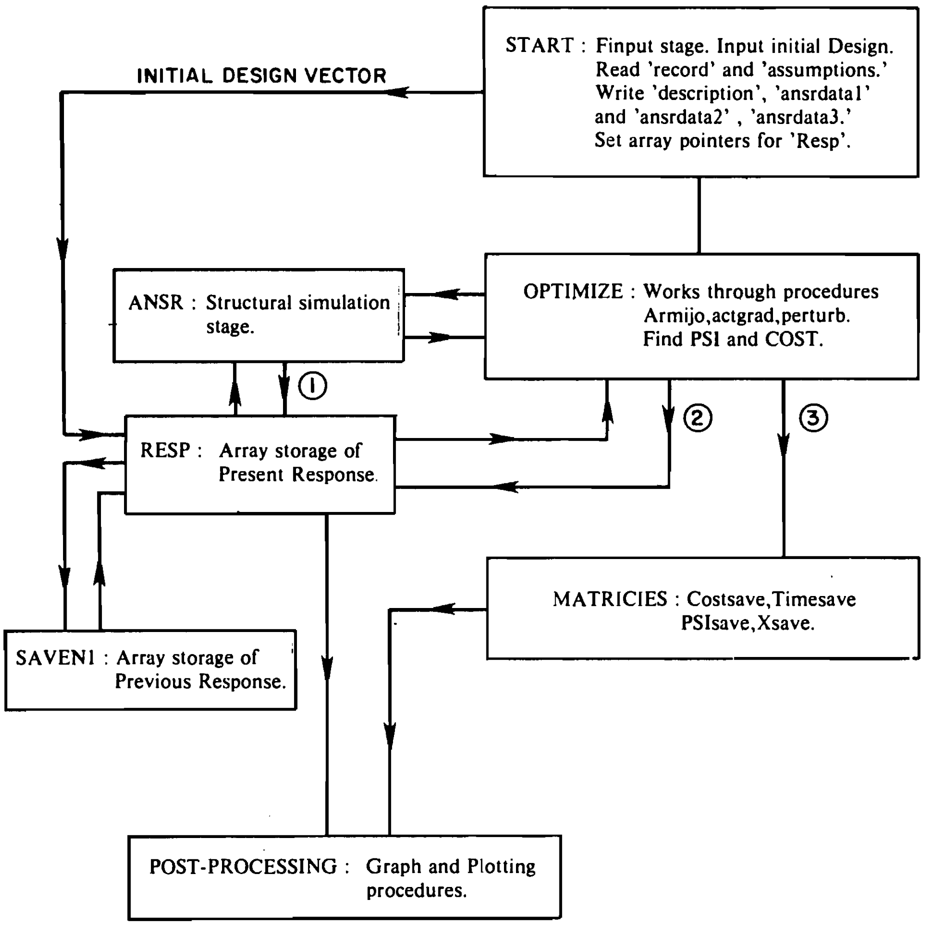


FIG (5) : Element Numbering for Example Frame.

FIG (6) : Element Grouping for Optimal Designs.


INITIAL DESIGN VECTOR
START : Finput stage. Input initial Design. Read 'record' and 'assumptions.' Write 'description', 'ansrdata1' and 'ansrdata2' , 'ansrdata3.' Set array pointers for 'Resp'.
ANSR : Structural simulation stage.
OPTIMIZE : Works through procedures Armijo,actgrad,perturb. Find PSI and COST.
1
2
3
RESP : Array storage of Present Response.
MATRICIES : Costsave,Timesave PSIsave,Xsave.
SAVEN1 : Array storage of Previous Response.
POST-PROCESSING : Graph and Plotting procedures.

1 : Storage of response data at the end of each time-step; via 'storsp','stor1','stor2'.
2 : Check present design variables against stored variables to see if a resimulation is needed.
3 : Storage of results at the end of each iteration. Summarized in the file 'history'

FIG (7) : Schematic of how 'Resp' is related to Major sections of DELIGHT.STRUCT.