UNITED STATES DEPARTMENT OF THE INTERIOR

GEOLOGICAL SURVEY

SEISRISK II

A COMPUTER PROGRAM FOR SEISMIC HAZARD ESTIMATION

by Bernice Bender and David M. Perkins

Open File Report 82-293

1982

SEISRISK II

A Computer Program for Seismic Hazard Estimation

by Bernice Bender and David M. Perkins

Abstract

The computer program SEISRISK II calculates probabilistic ground motion values for use in seismic hazard mapping. SEISRISK II employs a model that allows earthquakes to occur as points within source zones and as finite-length ruptures along faults. It assumes that earthquake occurrences have a Poisson distribution, that occurrence rates remain constant during the time period considered, that ground motion resulting from an earthquake is a known function of magnitude and distance, that seismically homogeneous source zones are defined, that fault locations are known, that fault rupture lengths depend on magnitude, and that earthquake rates as a function of magnitude are specified for each source.

SEISRISK II calculates for each site on a grid of sites the level of ground motion that has a specified probability of being exceeded during a given time period. The program was designed to process a large (essentially unlimited) number of sites and sources efficiently and has been used to produce regional and national maps of seismic hazard. It is a substantial revision of an earlier program SEISRISK I, which has never been documented. SEISRISK II runs considerably faster and gives more accurate results than the earlier program and in addition includes rupture length and acceleration variability which were not contained in the original version.

We describe the model and how it is implemented in the computer program and provide a flowchart and listing of the code.

Introduction

Given a model of earthquake occurrences, the level of ground motion (as characterized by acceleration, velocity or some other measure) which has a specified probability q of being exceeded at a site during a time period t can be determined. When the ground motion values corresponding to probability q and time period t are calculated at a grid of sites, an earthquake hazard map that consists of contours of various levels of motion can be constructed.

SEISRISK II has been used to calculate the ground motion levels contoured in national and other maps of seismic hazard. (It does not do the contouring). It employs a model which permits earthquakes to occur as points within a set of seismic source zones and as finite-length ruptures along well-defined faults. The ground motions calculated using the model depend upon assumptions regarding source zone and fault locations, attenuation function, and earthquake rates and magnitudes.

Because earthquakes from many seismic sources may contribute to ground motion at a site and because motions are calculated at a large number of sites in producing seismic hazard maps, the need to provide the ability to process a large number of sites and sources governed the architecture of the program. A major goal has been to make the program efficient and computer time as short as possible. The model, data required for inputs, and the version of the program, SEISRISK I, used to create 1976 national earthquake hazard maps are summarized by Algermissen and Perkins (1976). The code was published without documentation in Algermissen and others, 1976. For the same inputs, results using SEISRISK II are consistent with those obtained earlier, but the computer time required has been considerably reduced and the accuracy of some calculations increased. The revised program, however, is considerably more complex than the original version. The model and its implementation in

SEISRISK II will be described in detail in the next section. Changes that were made in source-area and fault-line computations will be discussed so that alternate approaches may be noted and compared. A flow chart, program listing, and information regarding data input and formats are included.

Seismic Hazard Model

In this model, earthquakes may occur within quadrilateral source zones or along linear fault segments. Seismic source zones and faults are defined to be such that the area enclosed within a zone (or length of a fault) can be regarded as seismically homogeneous, that is, earthquakes are equally likely to occur as point sources anywhere within the zone; ruptures of a given length may, with equal likelihood, be centered at any point on the fault for which neither end of the rupture would extend beyond the end of the fault.

Earthquakes in a given zone or fault are restricted to occur within a specified magnitude range; the range may be different for different zones and faults. More specifically, for each zone (fault) a maximum magnitude m_{max}, a minimum magnitude m_0, and a magnitude interval Δm are assumed. All magnitudes occurring within a magnitude interval are grouped at the center of the interval, that is, at a set of n distinct magnitudes m_i:

$$m_i = m_0 + i\frac{\Delta m}{2}, \quad 0 \leq i \leq n-1, \quad \Delta m = \frac{m_{max} - m_0}{n}.$$

The rate of earthquakes per unit time in the zone (or on the fault) in each magnitude interval is specified by the program user; it frequently follows (but is not required to follow) a relationship of the form

$$\log N_m = a - b\,m \tag{1}$$

where a and b are constants.

If N_{m_j} is the total number of earthquakes in the j^{th} magnitude interval, the fractional occurrences expected in any small subarea ΔA is

$$\frac{\Delta A}{A(tot)} N_m(i)$$

where $A(tot)$ = total area of zone.

Seismicity is assumed to remain constant during the time periods being considered, that is the average rate of earthquakes per unit time for each magnitude interval does not change with time.

Ground motion--let us say acceleration--at a site resulting from an earthquake is set up as a tabular function of earthquake magnitude and distance from the site: $a = f(m,d)$. (Acceleration is assumed to be monotonically increasing in magnitude and decreasing in distance.) The table is provided as input to the program to represent the attenuation function selected.

A histogram of the yearly rate of accelerations within 100 acceleration intervals is constructed for each site. Accelerations in the range $a(i-1) < a \leq a(i)$ are accumulated into the i^{th} entry of the histogram, as described below.

For the given attenuation, let $d_m(i)$ be the distance at which a magnitude m earthquake produces acceleration $a(i)$ at a site. The probability that an acceleration in the range $a(i-1) < a \leq a(i)$ occurs at the site is the probability that a magnitude m earthquake occurs at a distance $d_m(i) \leq d < d_m(i-1)$ integrated (summed) over all permissible magnitudes. (Note $d_m(i) \leq d_m(i-1)$ since shorter distances produce higher accelerations.)

The probability that an earthquake of magnitude m occurring within the zone (fault) occurs within the distance interval $d_m(i) \leq d < d_m(i-1)$ is equivalent to the fraction of the area of the zone (length of fault) that lies within this interval. The area (or length) within the interval may be expressed as a function of interval width

$$\Delta d(i) = d_m(i-1) - d_m(i)$$

and interval midpoint

$$\overline{d}_m(i) = \frac{d_m(i-1) + d_m(i)}{2}$$

or

$$Area[m(i)] = f[\overline{d}_m(i), \Delta d_m(i)]. \tag{2}$$

The rate of accelerations in the range $a(i-1) < a \leq a(i)$ at the site resulting from magnitude m earthquakes within the zone then is equal to the rate of magnitude m earthquakes within $Area[m(i)]$:

$$\rho_m(i) = \frac{Area[m(i)]}{Area(tot)} N_m \tag{3}$$

where

N_m = rate of magnitude m earthquakes for the entire zone (or fault)

$Area(tot)$ = total area of zone (or total fault length).

The program determines $Area[m(i)]$ for each magnitude and adds the corresponding rate $\rho_m(i)$ into the i^{th} histogram entry. The total rate accumulated for accelerations in the range $a(i-1) < a \leq a(i)$ from one source zone then is the sum over magnitude of

$$\rho(i) = \sum_{j=1}^{n} \rho_{m_j}(i) \tag{4}$$

where n = total number of magnitude intervals.

The process is repeated for all $1 \leq i \leq 100$, for all sources. Using the histogram, the exceedance rate $Ex[a(i)]$ of acceleration $a(i)$, that is the yearly occurrence rate of accelerations $a > a(i)$, can be determined. $Ex[a(i)]$ is defined as:

$$Ex[a(i)] = \sum_{j=i+1}^{100} \rho(j) \tag{5}$$

Note that exceedance has the same dimensions as rate $[\rho(i)]$ or frequency (N), that is earthquakes/year/area, and so is an annual rate.

We are now in a position to determine the probability associated with the exceedance of any acceleration. Earthquake occurrences are assumed to have a Poisson distribution, that is, the probability that an earthquake will occur is the

same at all times regardless of when the last earthquake occurred. For a Poisson distribution with mean rate μ, the probability of exactly k events during time t is

$$P(k,t) = \frac{(\mu t)^k \exp(-\mu t)}{k!} \tag{6}$$

The probability of no event during the same time interval is therefore

$$P(0,t) = \exp(-\mu t). \tag{7}$$

Setting $\mu = Ex(a)$ = yearly rate of exceedances of a, the probability of not exceeding a in t years is

$$P(0,t) = \exp[-Ex(a)\,t] \tag{8}$$

Taking the natural logarithm (ln) of both sides of equation (8) gives

$$Ex(a) = -\frac{ln[P(0,t)]}{t}. \tag{9}$$

That is, the value of a for which equation 9 is satisfied is the acceleration which has probability $p = P(0,t)$ of not being exceeded during the time interval t. (Equivalently, a has probability $q = 1-P(0,t)$ of being exceeded one or more times during t.) Assuming a table of $Ex[a(i)]$ exceedance rate versus acceleration $a(i)$has been calculated using equation 5, $Ex(a)$, the solution to equation 9, may be found by interpolation. By specifying any two of the three parameters (p,a,t) we can determine the remaining parameter. However, SEISRISK II expects that a probability level p and several values (up to twenty) of time are specified and the corresponding values of a are desired.

We shall describe the model in greater detail and discuss the differences between SEISRISK II and SEISRISK I in the following section on computational procedure.

Computational Procedure.

1. Geometry.

SEISRISK I uses a spherical earth for all computations. It determines the great circle distance between points on the earth and computes spherical arcs and areas. Such computations involve repeated evaluation of trigonometric functions and are quite time consuming. We found we could reduce computer time by using instead a rectangular coordinate system and computing linear distances, planar areas, and the like. SEISRISK II, therefore, transforms the region of interest to an area surrounding a new equator and then uses a rectangular coordinate system in which:

x = (new) longitude in radians · km

y = (new) latitude in radians · km.

More precisely, given the latitudes and longitudes of two selected points $(long_1, lat_1)$ and $(long_2, lat_2)$, the great circle through these two points (and center of the earth) is rotated to become the new equator so that

$$(\lambda_2, \varphi_2) = (long_2, lat_2) \rightarrow (0,0)$$

$$(\lambda_1, \varphi_1) = (long_1, lat_1) \rightarrow (d,0)$$

where

$$d = R \arccos[(\cos(\lambda_1)\cos(\lambda_2)(\cos(\varphi_1)\cos(\varphi_2) + \sin(\lambda_1)\sin(\varphi_2)) + \sin(\lambda_1)\sin(\lambda_2)]$$

= great circle distance between (λ_1, phi_1), (λ_2, φ_2)

R = radius of earth in kilometers

(Transformation equations are given in Appendix A.)

All subsequent input coordinates--for instance, end points of source areas and fault segments--are transformed relative to the new equator. Because of convergence of parallels of latitude, the error could be substantial if the original $(long, lat)$ coordinates were treated as being on a flat surface. However, this error is minimized if the coordinates are transformed to lie in the vicinity of a

new equator, so that, for example, within a 12 degree square region centered on this equator the maximum error introduced by using the straight line distance between two points (x_i, y_i), (x_j, y_j) instead of the great circle distance is less than .6%.

Felt points or sites are expressed as points on a rectangular m by n grid in which the site (x_i, y_j) is

$$x_i = \text{longitude of row } i \cdot \text{ck}, \qquad 1 \leq i \leq m$$

$$y_j = \text{latitude of col } j \cdot \text{ck}, \qquad 1 \leq j \leq n$$

where ck=conversion factor for degrees to kilometers at the equator.

Coordinates $(long, lat)$ of two opposite corners (upper left, lower right) of the seismic felt area in the initial system, and grid spacing are input to the program (figure 1). Sites may also be points evenly spaced on a number of lines; in this case, $(long, lat)$ of two end points of each line, number of sites per line, and number of lines are specified as input.

2. Acceleration Table.

A table of 100 acceleration levels $ac(k)$, such that $ac(k) < ac(k+1)$ for $1 \leq k \leq 100$, is constructed. A scale factor (input) allows the original range of acceleration values in the table to be expanded or contracted. For a scale factor of one, accelerations are in the range .01-4.0 g. (A scale factor of .5 would permit values from .005g-2.0g.) (The table is created in the subroutine "box" and other values could be substituted by replacing this subroutine.)

Given $ac(k)$, a corresponding array $reg(k)$ is set aside to accumulate fractional acceleration occurrences so that occurrences in the range $ac(j-1) < a \leq ac(j)$ are placed in $reg(j)$. Accelerations less than $ac(1)$ are placed in $reg(1)$ and those exceeding the highest entry $ac(max)$ (where max = 100) are placed into $reg(max+1)$.

Earthquakes as point events within quadrilateral source areas and as finite

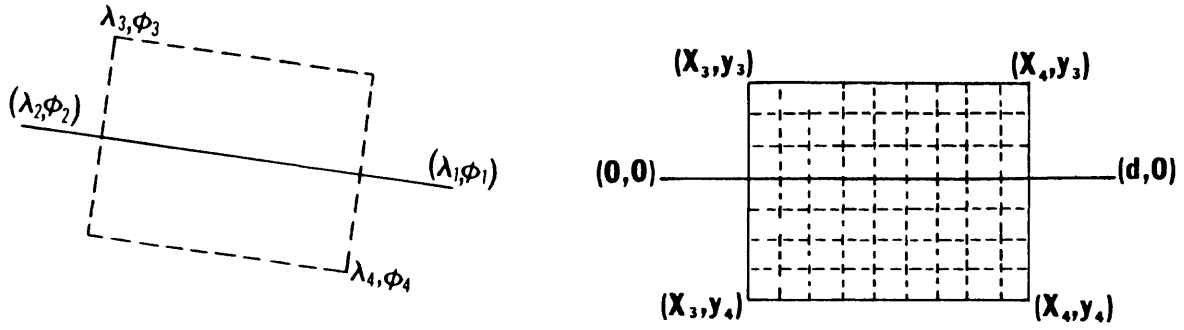


Figure 1. Four points define transformation and seismic felt area.
Great circle through (λ_1,φ_1), (λ_2,φ_2), and (0,0) becomes new equator.

$$(\lambda_1,\lambda_2)\rightarrow(d,0)$$
$$(\lambda_2,\varphi_2)\rightarrow(0,0)$$

(λ_3,φ_3) and (λ_4,φ_4) become endpoints of rectangular area containing seismic felt points.

$$(\lambda_3,\varphi_3)\rightarrow(x_3,y_3)$$
$$(\lambda_4,\varphi_4)\rightarrow(x_4,y_4)$$

Sites in new coordinate system are at even increments in $\Delta x,\Delta y$ at centers of gridded areas. At end of computation, sites are transformed to the coordinates they would have in the original system.

length ruptures along linear fault segments are assumed to contribute to accelerations at a site. We shall discuss these two cases separately.

3. Source Zones.

A seismic source zone is defined as a seismically homogeneous area enclosed by one or more arbitrary quadrilaterals, connected or disjoint (figure 2). In SEISRISK I, an area which includes all of the sources is divided into subareas by a grid spaced at even increments in latitude and even (but possibly different) increments in longitude. Contributions to the total acceleration at the site are computed for each gridded subarea which overlaps a source. The distance from the center of the subarea to the site is computed and the acceleration a for an earthquake of specified magnitude at this distance is determined. The fraction of the source region within this subarea multiplied by the earthquake rate per

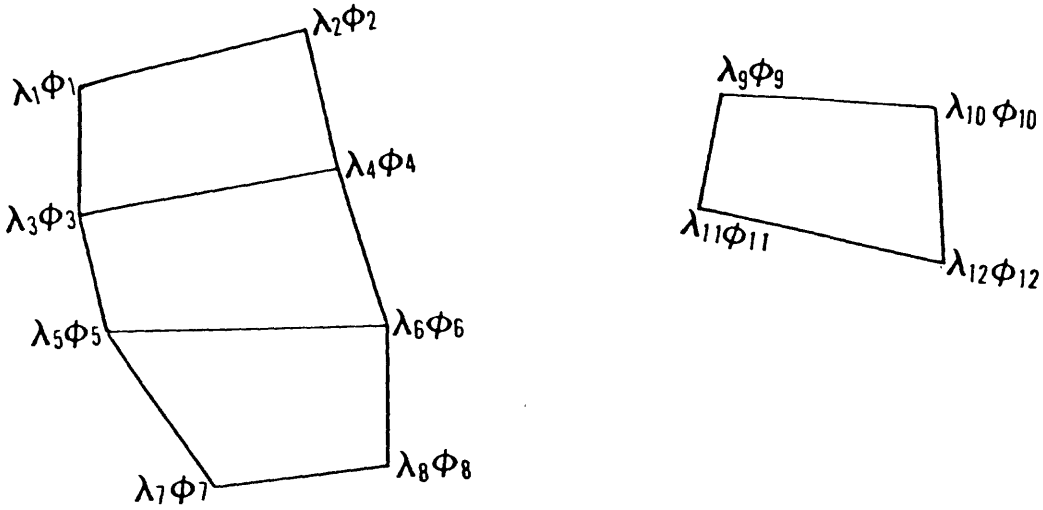


Figure 2. Seismic source zone composed of two sets of quadrilaterals

Input: For each quadrilateral set

(1) num fr tot

num=number +1 of quadrilaterals in this set

fr identifies current set, $1 \leqslant fr \leqslant tot$

tot=total number of sets of quadrilaterals

(2) pairs of quadrilateral corner points.

In this example, input is

```
 4 1 2
λ1 φ1 λ2 φ2
λ3 φ3 λ4 φ4
λ5 φ5 λ6 φ6
λ7 φ7 λ8 φ8
 2 2 2
λ9 φ9 λ10 φ10
λ11 φ11 λ12 φ12
```

year for this magnitude gives the total contribution to the acceleration histogram for this subarea and magnitude. If a is such that $ac(j-1) < a \leqslant ac(j)$ the fractional expected occurrences are placed in $reg(j)$, the accumulator for accelerations in this range (figure 3). Note that because all magnitude m earthquakes in a subarea are assumed to occur at the same distance from a site and are placed into a single $reg(j)$, the size of the entry in $reg(j)$ is a function of grid spacing. Furthermore, contributions from adjacent subareas are not necessarily added into adjacent $reg(j)$, $reg(j+1)$, and a "lumpy" histogram may result. Acceleration densities near the maximum acceleration at a site

especially may be biased unless an effort is made to correct to zero grid spacing (Perkins, 1978).

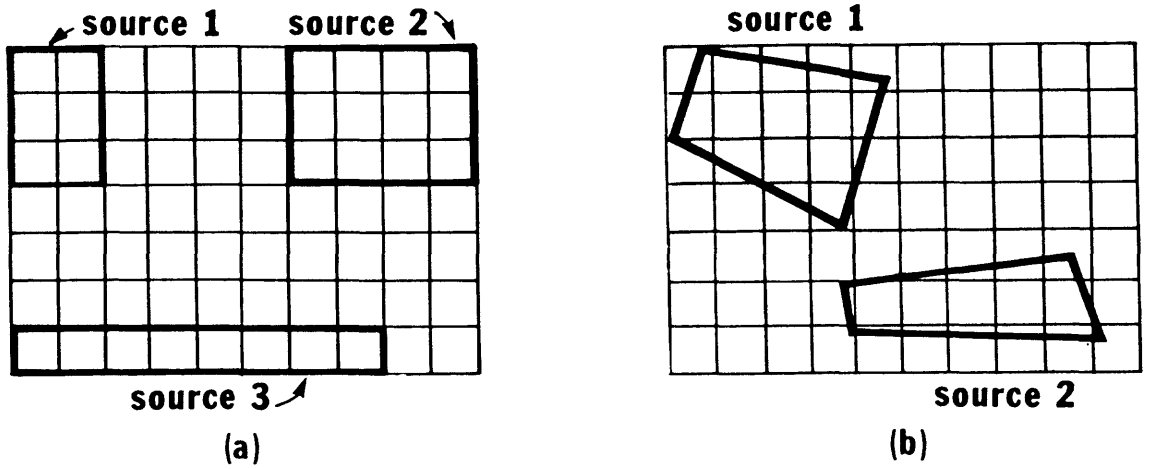


Figure 3. Grid in original Algermissen and Perkins program (1976) contains source areas and sites at which hazard is to be evaluated.
(a) ideal case. Edges of source areas are on grid lines. If source contains n blocks, each block expects $1/n$ of the earthquakes.
(b) usual case. Each small gridded area expects earthquakes in proportion to the total source area it contains.
Distance from a site to an earthquake is the distance to the center of the block in which the earthquake occurs.

Accelerations for sites inside a source area are particularly sensitive to grid size (figure 4).

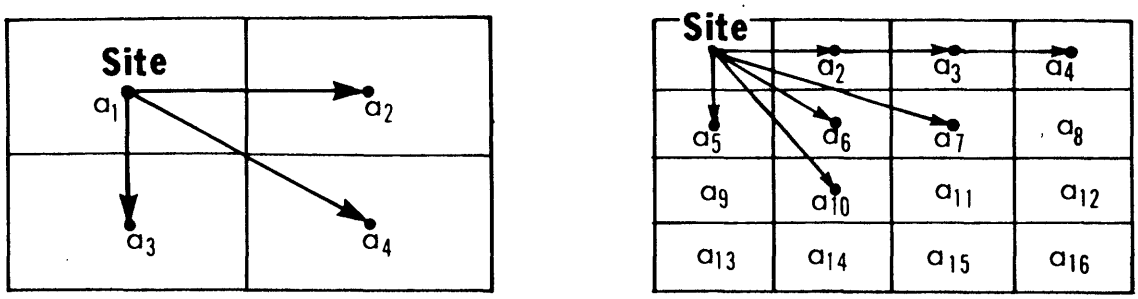


Figure 4. In original Algermissen and Perkins program, accuracy depends upon grid spacing. Attenuation function is evaluated at a distance measured to the center of each gridded area.

(A site in either source area in figure 4 is at zero distance from an earthquake occurring in the rectangle containing the site, but in the first case four times as many earthquakes occur "at zero distance" as in the second. This

"discretization error" may be significant if acceleration falls off rapidly with distance.)

Increasing the number of grid points will increase the accuracy and decrease the bias, which may be important for sites near the source. However, since grid size remains fixed for a computer run, decreasing the grid spacing by a factor two in longitude and latitude results in four times as many distance-area computations for each site. Because this is a mapping program and sites typically cover a large area, many sites are far from the source where the additional computations gain little in accuracy. We overcame this difficulty by incorporating a modification of the approach used by McGuire (1976). We shall discuss McGuire's approach, some possible problems and our solution to them.

McGuire does the distance-area computation in the reverse order: instead of finding distances of gridded subareas from the site, he determines an approximation to the fraction of the total quadrilateral area contained within each of a set of annular rings at fixed radial distances from the site (figure 5).

For sites outside the source, he computes the closest ($r_{\min}$) and farthest (r_{max}) distances of the quadrilateral from the site and then divides the difference ($r_{\max} - r_{\min}$) into a number (nstep) of evenly spaced intervals by constructing arcs of circles with radii $r = r_{min}, r_{\min}+dr, r_{\min}+2dr, \ldots r_{\max}$ and centers at the site. The entire interval is assumed to contribute to the acceleration level calculated for the center of the interval, that is, at the radial distance of the circle midway between two bounding arcs.

The area of one interval is estimated by

$$area = angle \cdot r \cdot dr \tag{10}$$

where $angle$ = arc length in radians of that portion of the circle of radius $r = r_{\min} + (i + \frac{1}{2})dr$ that lies wholly within the quadrilateral,

dr = interval width (figure 5).

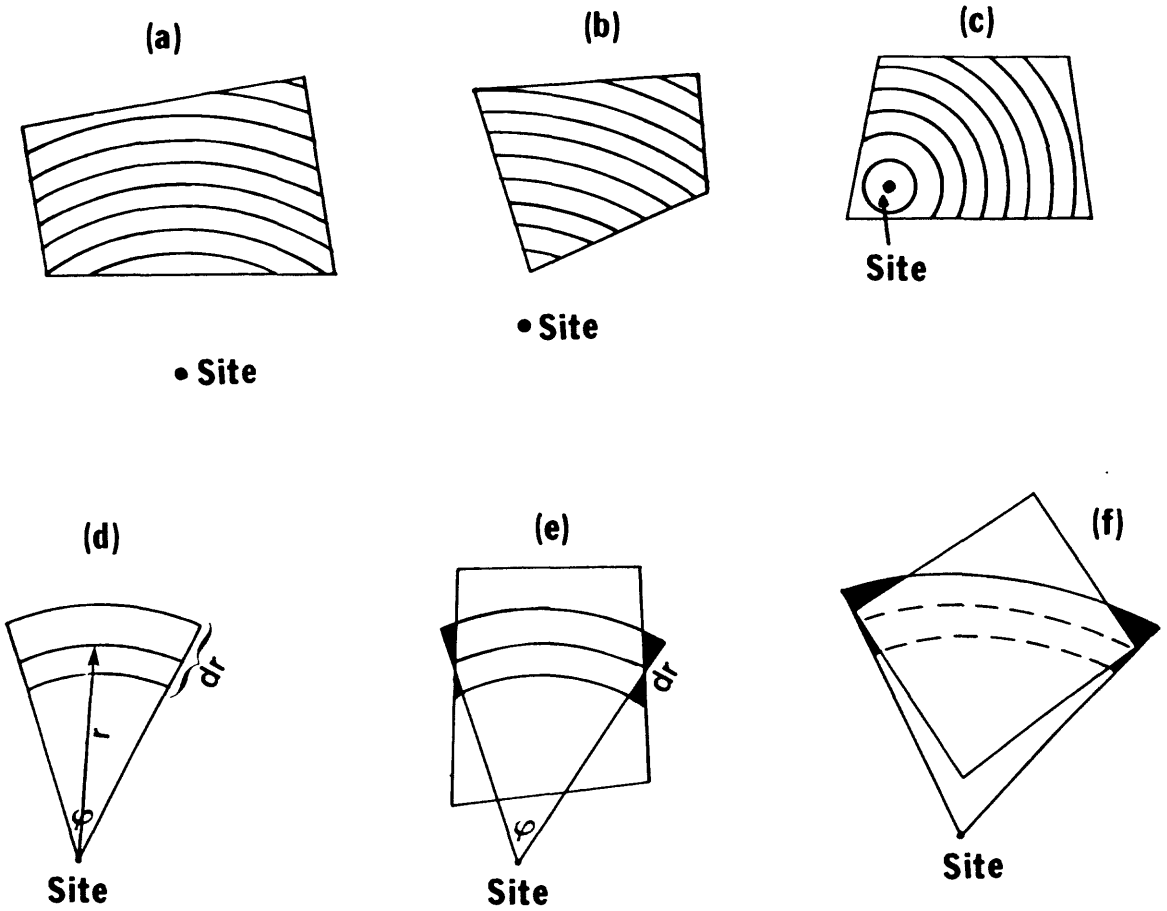


Figure 5. Arcs of circles with centers at the site are used to approximate the area of a quadrilateral.

Area = $\varphi r\, dr$ in (d), (e) and (f). Shaded area in (e) and (f) is difference between quadrilateral area and arc area. In (e) errors nearly cancel; in (f) the arc area is an overestimate.

For sites within the source area, the intervals are as above for $r_{min} \leq r \leq r_{max}$, and are annular rings for those circles wholly contained within the quadrilateral for $0 \leq r \leq r_{min}$ (figure 5).

McGuire noted that arc-areas can give a poor approximation to quadrilateral areas if too few arcs are used. On the other hand, considerable computer time is required to determine the intersections of quadrilaterals and circles and the angles involved. (See subroutines "inside" and "outsid" in the program listing.)

McGuire discusses the question of how many intervals are required to approximate the quadrilateral area within a specified tolerance. He notes that

in his risk program, the computer time needed is roughly proportional to the number of intervals; doubling the number of intervals approximately doubles the total running time.

We investigated the interval question further and reached some additional conclusions:

When acceleration is evaluated at a set of n equally spaced radii (as in McGuire's program), accelerations are restricted to these n values. Since acceleration $a = f(m,d)$ may be a highly nonlinear function of magnitude and distance, the acceleration densities obtained may be inaccurate even though the approximation to the quadrilateral area is quite good using these radii.

It is particularly important that areas and densities for the annuli closest to the site be accurate since most of the higher (and usually most important) accelerations at the site occur within these annuli.

We note that the same number ($nstep$) of intervals even for the same source can give varying accuracy depending on the orientation of the region with respect to the site. In the first quadrilateral of figure 6, a much greater distance from the nearest point to the farthest means that the acceleration is evaluated at four times as large an interval spacing dr as in the second case. However, the arc areas in both cases give about the same percentage error in estimating source areas.

Thus it would appear that computing accelerations for areas on a fixed grid and computing accelerations for annular areas at fixed radii have similar tradeoffs in time versus accuracy of the contributions to the acceleration histogram (or boxes).

Taking advantage of the histogram-based architecture of the program, we found we were able to considerably improve the accuracy and simultaneously decrease the running time as follows:

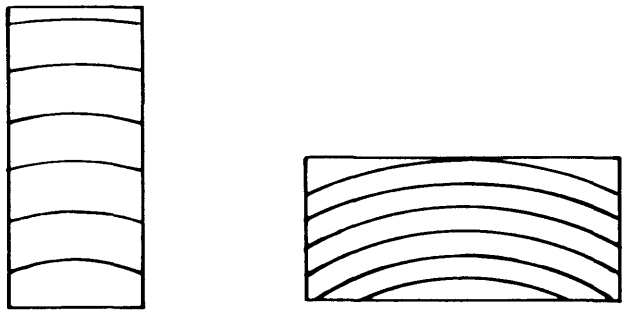


Figure 6. Source area with two orientations relative to site.

Recall that an acceleration "box" represents acceleration values in some range $ac(j-1) < a \leq ac(j)$. For a specified magnitude, box boundaries $[ac(j-1), ac(j)]$ correspond to radii $[r(j-1), r(j)]$. The contribution to accelerations $ac(j-1) \leq a \leq ac(j)$ produced by magnitude m earthquakes in a quadrilateral is proportional to the area within the quadrilateral at distances $r(j) \leq r < r(j-1)$. Instead of estimating areas for radii at fixed distances, we may increase our accuracy by estimating the area bounded by the radii $[r(j), r(j-1)]$. However, this could involve computing distances for a large number of radii and, as we have seen, be quite time consuming.

We decided it might be possible to save computation time by interpolating in a table of distance versus arc-length (for a predetermined set of radii) to obtain areas and hence acceleration densities as desired. The usefulness of this approach depends upon how many tabular entries are required in order that areas can be approximated with reasonable accuracy by linear interpolation. We concluded that we could obtain satisfactory results by interpolation using seven evenly spaced radii per quadrilateral plus from two to eight additional radii to vertices and edges depending upon the location of the site relative to the source. The tests, geometric considerations, and conclusions are discussed

in Appendix B.

Let us assume that a table of distance versus arc-length has been constructed and that arc-length can be interpolated as needed. Because an interpolation involves only a fraction of the time required to compute an angle, we proceed as follows:

Given a pair of radii, $[r(j-1), r(j)]$ we determine the corresponding quadrilateral area for magnitude m earthquakes:

$$Area[m(j)] \approx \sum_{k=1}^{np} arc(k)\,\Delta d \tag{11}$$

where

$arc(k)$ = arc length (km) at radial distance $d(k)$

(obtained by interpolation)

$d(k)$ = midpoint of an interval of width $\Delta d(k)$

np = number of points at which $d(k)$ is evaluated, $np \geq 1$.

[If the bounding radii $r(j-1)$ and $r(j)$ lie between a single pair of distance entries in the distance-arc length table, $np = 1$ and $d(1) = \frac{r(j-1)+r(j)}{2}$; otherwise np = 1 + number of tabular distance entries spanned by $r(j-1) < r \leq r(j)$.]

We normalize the result to fractional area and multiply by earthquake rate to obtain the acceleration entry for magnitude m to be added into $reg(j)$. Summing the contributions for each magnitude gives the total entry which has been placed in $reg(j)$ for this source.

Because only a few angles are calculated and linear interpolations are done rapidly, this annular approach has proven considerably faster than computing distances on a grid. It also inherently gives a smoother set of acceleration densities. Because it resulted both in decreased computer running time and greater accuracy, the arc method was incorporated into the programs and gridding of source areas eliminated. Angles are now evaluated for a set of radii between r_{min} and r_{max} and saved in a table for later (linear) interpolation.

3. Faults

A fault zone consists of one or more seismically homogeneous faults, that is, earthquakes are distributed among faults in a zone in proportion to the length of the faults. An individual fault is composed of up to 24 connected straight line segments. (The maximum number of segments may be altered within the program.)

As with source areas, the amplitude of ground motion at a site is a function of earthquake magnitude and distance. However, in the case of faults, distance from the site to a rupture is always taken to be the distance to the rupture point nearest to the site. Rupture lengths are a function of magnitude, and are log-normally distributed with median length $bl(m)$ and standard deviation σ_l. Ruptures are assumed to occur with equal likelihood anywhere along the fault, so long as they are wholly contained within the fault.

A rupture of a magnitude m earthquake is defined to have length $bl_i(m)$ with probability $p(i)$, where

$$log[bl_i(m)] = a + b\,m + fr(i)\,\sigma_l \qquad (12)$$

a, b are constants.

SEISRISK II allows either one or five rupture lengths per magnitude to be used depending upon whether variability is to be modelled in the length versus magnitude relationship. Thus

$$fr(i) = \begin{cases} 0 \text{ if only median rupture length is used} \\ \text{(one rupture length per magnitude),} \\ \text{five values in the range } -2 \leq fr(i) \leq 2 \\ \text{(five lengths per magnitude).} \end{cases}$$

$p(i)$ = probability of rupture of length $bl_i(m)$

$$p(i) = \frac{1}{\sqrt{2\pi}} \int_{f(i)}^{f(i+1)} exp\left(\frac{-x^2}{2}\right) dx \qquad (13)$$

where $f(i)$ is selected so that $fr(i)$ is

$$f(i) < fr(i) \leq f(i+1); \qquad f(1) = -\infty; \qquad f(nr+1) = \infty.$$

$$\sum_{i=1}^{nr} p(i) = 1$$

nr = number of rupture lengths per magnitude.

A fault is composed of one or more connected line segments of total length L. Ruptures of length $bl_i(m)$ are permitted to occur only at a set of n distinct locations such that rupture centers are evenly spaced at increments $\Delta bl_i(m)$ on the fault. The k^{th} rupture center is at a distance $d_i(k)$ from one end of the fault where

$$d_i(k) = \frac{bl_i(m)}{2} + (k-1)\,\Delta bl_i(m) \tag{14}$$

and

$$\Delta bl_i(m) = \frac{L - bl_i(m)}{n-1} \qquad 1 \leq k < +n, \tag{15}$$

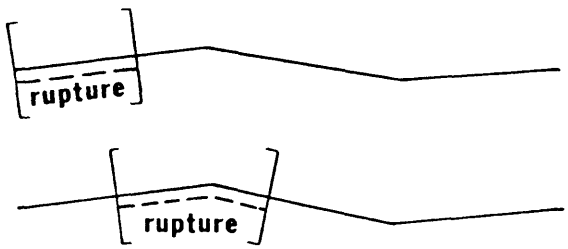


Figure 7. Fault composed of three connected straight line segments. A rupture may span several segments. Closest distance from rupture to site is used to determine ground motion at the site.

Ruptures of length $bl_i(m)$ are assumed to occur with equal probability at all permissible rupture locations. The fractional seismicity $C(i,m)$ associated with one such rupture is

$$C(i,m) = \rho(m)\frac{p(i)}{n}\,frl \tag{16}$$

where $\rho(m)$ = rate of magnitude m earthquakes for the set of faults

frl = length of this fault/total length of faults in set.

For each rupture the closest distance to the site is determined. Since rup-

tures may span several fault segments (or even be as long as the entire fault) the closest point to the site is the closest point of the rupture on any fault segment containing it (figure 7). For one such rupture of length $bl_i(m)$ the acceleration "box" level is found for the acceleration corresponding to the rupture distance and earthquake magnitude. The associated seismicity $C(i,m)$ is added into the accelerations accumulated at this level.

When the computation of nearest distance from the site to a rupture is done repeatedly as one assumes different positions for the rupture along the fault, the computer time required may be substantial. This time can be considerably reduced if some minor transformations are made, as discussed in Appendix C. Using the procedure described, we can now quite efficiently find the nearest distance to a single rupture. However, choosing a discrete set of rupture locations can give only an approximation to the fraction of possible magnitude m ruptures of length bl yielding accelerations in the range $ac(j-1) < a \leq ac(j)$. In some cases we can determine this fraction exactly without calculating values for ruptures moved incrementally along the fault. (See figure 8, cases (a) and (b)).

In these cases either the point on the fault nearest the site is at one end of the fault and the distance to the site increases (or decreases) monotonically as one moves along the fault, or the distance to the site decreases as one moves along the fault from one end to some interior point and then increases as one continues along the fault to the other end. (One of these cases always holds, for example, when the fault is a single segment.)

We shall indicate briefly how we determine the length of fault containing centers of ruptures whose closest distance to the site lies in the range $d[m(j)] \leq d < d[m(j-1)]$. This length of fault represents a fraction of all possible rupture centers for magnitude m earthquakes and contributes a corresponding fractional acceleration $ac(j-1) < a \leq ac(j)$. A more detailed and

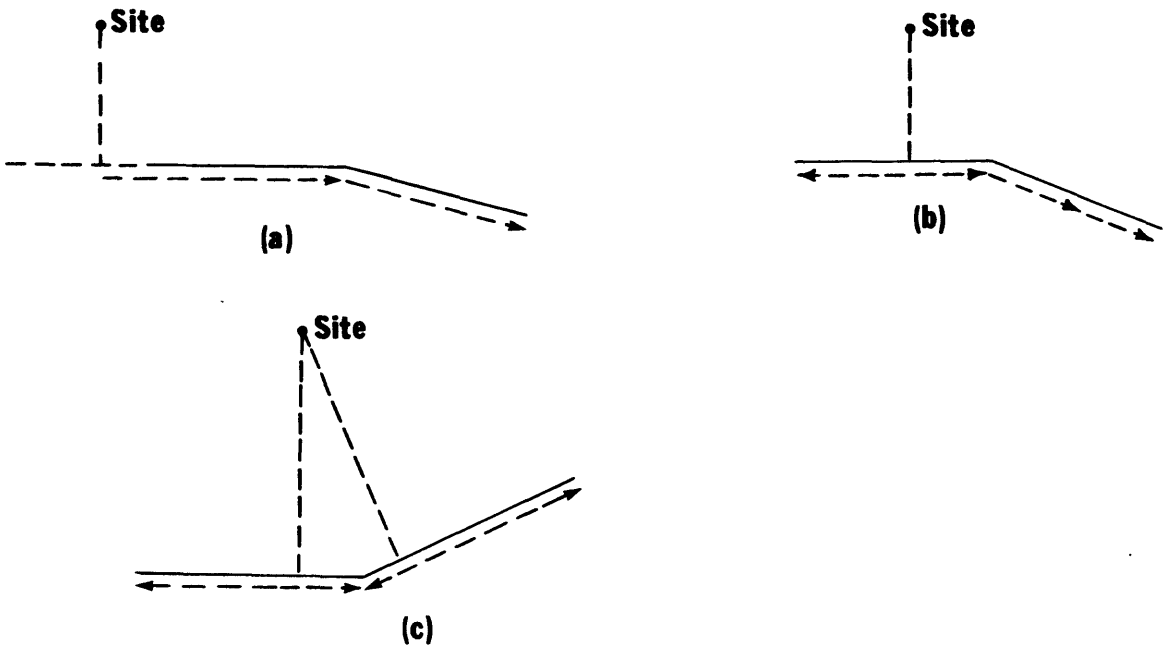


Figure 8. Arrows indicate direction of increasing distance along the fault from the site.
(a) Closest distance from site to fault is at left end of fault. Distance increases monotonically along the fault as one moves in the direction of the arrows. (b) Distance to site decreases as one moves along fault from left end of fault to point A, then increases monotonically as one continues to other end. (c) Distance to site alternately decreases and increases as one moves along the fault.

mathematical description is contained in Appendix C.

To simplify the discussion, let us assume that the fault consists of a single line segment extending from (0,0) to $(L,0)$ and the site is located at (X,P).

Case 1. $X<0$ (Equivalently, $X>L$).

In this case, the point on the fault nearest the site is at (0,0) with the distance to the site increasing monotonically along the fault. Because a rupture cannot extend beyond the end of the fault, a rupture of length $\delta l = \delta l_i(m)$ must have its midpoint l_{mid} on the fault between

$$\frac{\delta l}{2} \leq l_{mid} \leq L - \frac{\delta l}{2}.$$

But the point on the rupture closest to the site is the endpoint l_{end} nearest to

(0,0) on the fault and must lie in the range:

$$0 \leq l_{end} \leq L - bl .$$

All permissible positions of l_{end} are equally likely.

Let $pt(j) = x$ coordinate of the point on the fault which is at distance $d_{m(j)}$ from (X,P). The length of the fault corresponding to distances in the range $d_m(j) \leq d < d_m(j-1)$ is the largest interval $\Delta x(j)$ for which

$$0 \leq pt(j) \leq \Delta x(j) \leq pt(j-1) \leq L - bl$$

and the fractional accelerations in the range $ac(j-1) < a \leq ac(j)$ therefore are (assuming rupture length less than fault length)

$$fr(j,m) = \frac{\Delta x(j)}{L - bl} \rho(m) frl \qquad (17)$$

where $\rho(m)$ and frl are defined as in equation 16. For those $d_m(j)$ for which there is no corresponding $pt(j)$ on the fault, $fr(j,m) = 0$.

Case 2. $0 < X < L$.

The distance from the fault to the site decreases monotonically as one moves from (0,0) along the fault to $(X,0)$ then increases steadily as one continues to the other end to $(L,0)$.

In this case, all magnitude m ruptures with midpoints on the fault between

$$X - \frac{bl}{2} \leq x \leq X + \frac{bl}{2}$$

overlap $(X,0)$ and are all at the same closest distance P from (X,P). Hence all these ruptures will produce the same acceleration at the site, and contribute to a single $reg(j)$.

For all ruptures with midpoints

$$\frac{bl}{2} < l_{mid} < X - \frac{bl}{2},$$

the point closest to the site will be a rupture endpoint at $l_{end} = (l_{mid} + \frac{bl}{2})$ (where $bl \leq l_{end} < X$). Similarly, all ruptures with midpoints

$$X + \frac{bl}{2} < l_{mid} < L - \frac{bl}{2}$$

will have their closest endpoint on the fault at

$$X < l_{end} < L - bl.$$

Now the problem reduces exactly to the monotonically increasing (or decreasing) situation of Case 1, if we regard the fault as being composed of two segments extending from (0,0) to (X,0), and (X,0) to (L,0), after accounting for the ruptures which overlap (X,0).

For these cases including rupture-length variability can also be done easily without substantially increasing computation time. This is also discussed in Appendix C.

Acceleration Variability

Accelerations are assumed to be lognormally distributed with standard deviation σ_a around the mean (log) values given by the attenuation function. (In a lognormal distribution, the mean value of $ln[f(x)]$ is actually the median value of $f(x)$.) This states that given mean log acceleration $\overline{a}$, the probability of an acceleration in the range $a_1 \leq a \leq a_2$ is the area under the normal probability integral:

$$pr(a_1, a_2) = \frac{1}{\sqrt{2\pi}\sigma_a} \int_{x=\ln a_1}^{\ln a_2} \exp[\frac{-(x-\ln \overline{a})^2}{2\sigma_a{}^2}]dx \qquad (18)$$

To model this variability, instead of placing the fractional occurrences of $\overline{a}$ into the box corresponding to $\overline{a}$, we distribute these occurrences into acceleration boxes in proportion to the probability $pr(a_{j-1}, a_j)$ for each j.

If σ_a is not a function of magnitude and distance, we can postpone this spreading until after the entire histogram of mean (log) acceleration densities for a site has been accumulated. For each acceleration level, $ac(j-1) < a \leq ac(j)$, the rate accumulated in reg(j) can be redistributed into a new set of accumulators regs(k). If the entry in $reg(j)$ is assumed to be concentrated at acceleration $\overline{a}$ where

$$\ln(\bar{a}) = \frac{\ln[ac(j-1)] + \ln[ac(j)]}{2},$$

then from equation 18, the portion placed in $regs(k)$ is

$$pr[ac(k-1), ac(k)]\, reg(j) \rightarrow regs(k).$$

This computation may be performed by evaluating the appropriate normal probability area in each case. It may be approximated by using a set of n (odd) points each with weight $\frac{1}{n}$ such that for the k^{th} of these:

$$ln(a_k) = \ln(\bar{a}) + dev(k), \tag{19}$$

$$pt(k)\,\sigma_a < dev(k) \leq pt(k+1)\,\sigma_a \tag{20}$$

and $pt(k), pt(k+1)$ are such that

$$\frac{1}{\sqrt{2\pi}} \int_{pt(k)}^{pt(k+1)} exp\left(\frac{-x^2}{2}\right) = \frac{1}{n}, \quad 1 \leq k \leq n.$$

A fraction $(\frac{1}{n})$ of the rate in $reg(j)$ is redistributed for this k into the $regs(i)$ for which:

$$ln[ac(i-1)] < ln(a_k) \leq ln[ac(i)].$$

The approximation (subroutine wts) uses n=19 and σ_a (input). Input parameter "nterp"=1 causes the approximation to be used, "nterp"=2 the integration, equation 18.

Note that acceleration variability is accounted for only at the end, after median rates have been determined for all the areas and faults. When earthquakes from many sources are added in, this results in a considerable time saving over methods which include acceleration variability at each step in the computation.

Attenuation Function.

The program computes ground motion (acceleration) by interpolating in a table of acceleration as a function of magnitude and distance. Interpolations are linear in distance and linear in log acceleration and log magnitude.

The table is read in for up to eight magnitudes and twenty distances.

Velocity or some other ground motion parameter may be substituted for acceleration. If the ground motion values read in are in a range different from .01 to 4, the appropriate scale factor must be input to make the "box" levels compatible with these values.

Algermissen and Perkins (1976) in producing their 1976 seismic risk maps, used modifications of curves for acceleration done by Schnabel and Seed (1973). (Use of an attenuation function of the type $\ln a = c_1 + c_2 m + c_3 \ln R$ (Esteva, 1969) may require a change in the manner of interpolation.)

Fault Length Inputs.

The values of a, b, and σ_l may be provided as input for the rupture-length relationship

$$\log_{10}(bl) = a + b\,m + fr\,\sigma_l \qquad (16)$$

If a, b, σ_l are all zero or blank, previous values are used. If no previous values were provided, default values of Bonilla and Buchanan (1970) are used:

$a = -1.085 \quad b = .389 \quad \sigma_l = .52$

If $a = 0$, $b = 0$, and $\sigma_l = 0$ are input, a single rupture-length value is used

length $= .000630\,exp(1.52\,m)$ (where m = magnitude)

for consistency with the original national maps produced by Algermissen and Perkins (1976).

Output

As stated earlier, we seek the acceleration that, with probability q, will be exceeded (or probability $p = 1-q$ will not be exceeded) during t years. We have seen that this is the value a of acceleration for which $Ex(a)$ the yearly exceedance rate of a at the site is

$$Ex(a) = \frac{-ln(p)}{t}.$$

Assuming yearly exceedance rates (as in equation 5) have been calculated for each of 100 acceleration levels, the solution a that corresponds to the specified (p, t) may be found by interpolating between the calculated values.

Solutions are computed for up to 24 time periods t using rates derived for median accelerations only, and also for the same set of times using rates that include acceleration variability.

The results, including the acceleration histogram, may optionally be written to file 16 for later printing. Summary results may be saved on file 2 for future plotting.

Effects of the earthquakes in a source area or fault set are calculated for all sites before proceeding to the next source, with acceleration occurrences being added to those obtained previously for each site. Therefore, partial results may be examined any stage. Output options are given for each source set by the input parameter *iprint*.

iprint =
- -1: continue to next set; no print, no statistics
- +1: compute solutions a; write histogram and solutions to file 16, then continue to next set
- +2: also same as +1 but save the solutions on file 2
- +3: no write to file 16; save solutions on file 2

Run Continuation:

Rates that have been accumulated into the $reg(j)$ for each site are saved in file 3 at the end of a run. These values then may be read back into $reg(j)$ for a subsequent run. This makes it possible, for example, to accumulate accelerations from sources having different attenuation functions.

Parameter $isw=1$ for run continuation, $isw=0$ for new run.

Conclusions

The SEISRISK II computer program was designed to process a large number of sites and source zones efficiently in a single computer run. The program is organized to complete calculations for all sites for a single source before proceeding to another source. Extensive use is made of coordinate transformations in order to reduce distance calculations. A number of calculations are simplified by linear interpolation of slowly varying variables. The program exploits the histogram summary of ground motions by limiting the distance calculations to those that correspond to histogram boundaries. This is particularly effective in shortening the number of calculations required to adquately represent the site effects of fault rupture sources.

SEISRISK II is considerably faster than the original version (SEISRISK I). While timings, of course, depend on the actual input values (for instance, grid spacing in the original program), the newer version runs at least three times faster than the earlier one at 1/4-degree grid spacing. It is of the order of 10-100 times faster than McGuire's seismic risk programs (1976, 1978).

References

Algermissen, S.T., and Perkins, David M., 1976, A probabilistic estimate of maximum acceleration in rock in the contiguous United States: U.S. Geological Survey Open-File Report 74-416, 45 p.

Algermissen, S. T., Perkins, D. M., Isherwood, W., Gordon, D., Reagor, B.G., and Howard, C., 1976, Seismic Risk Evaluation of the Balkan Region: Proceedings of the Seminar on Seismic Zoning Maps, Skopje, Yugoslavia, Oct 27-Nov 4, 1975, v. 2, p. 172-240.

Bonilla, M. G., and Buchanan J. M., 1970, Interim report on world wide historic and surface faulting: U. S. Geological Survey Open-File Report, 32 p.

Esteva, L., 1969, Seismicity prediction: a Bayesian approach : Proceedings World Conference Earthquake Engineering, 4th, Santiageo, Chile, 2, p.172-184.

McGuire, Robin, 1976, FORTRAN computer program for seismic risk analysis: U.S. Geological Survey Open-File Report 76-67, 90 p.

McGuire, Robin, 1978, FRISK: Computer program for seismic risk analysis using faults as earthquake sources: U.S. Geological Survey Open-File Report 76-67, 90 p.

Perkins, D.M., 1978, Acceleration Hazard Map Sensitivity to Input Seismic Parameters, Bollettino di Geofisica Teorica ed Applicata v. XX, no. 78, p. 188-197.

Schnabel, P.B., and Seed, H.B., 1973, Acceleration in rock for earthquakes in the Western United States: Bulletin of Seismological Society of America v. 63, p. 501-516.

Wallace, R.E., 1970, Earthquake recurrence intervals on the San Andreas fault: Geological Society of America Bulletin, v. 81, p. 2875-2890.

Appendix A

Equations to transform spherical coordinates to coordinates referred to an arbitrary great circle as a new equator.

Given the longitude and latitude (λ_1,φ_1) and (λ_2,φ_2) of two points P_1, and P_2 on a sphere of unit radius, transform the great circle through these two points (and the center of the sphere) so that $(\lambda_1,\varphi_1)\rightarrow(0,0)$, and $(\lambda_2,\varphi_2)\rightarrow(d,0)$ where d = great circle distance between P_2 and P_1. The transformation matrix $[R_{ij}]$ is the product of three rotation matrices:

$$[R_{ij}]=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\vartheta & \sin\vartheta\\ 0 & -\sin\vartheta & \cos\vartheta\end{bmatrix}\cdot\begin{bmatrix}\cos\varphi_1 & 0 & \sin\varphi_1\\ 0 & 1 & 0\\ -\sin\varphi_1 & 0 & \cos\varphi_1\end{bmatrix}\cdot\begin{bmatrix}\cos\lambda_1 & \sin\lambda_1 & 0\\ -\sin\lambda_1 & \cos\lambda_1 & 0\\ 0 & 0 & 1\end{bmatrix}$$

where $\sin\vartheta=\dfrac{\beta}{\sqrt{\beta^2+\gamma^2}}$; $\quad\cos\vartheta=\dfrac{\alpha}{\sqrt{\beta^2+\gamma^2}}$

$$\alpha=\cos\varphi_1\cos\varphi_2\cos(\sin\varphi_1-\lambda_1)+\sin\varphi_1\sin\varphi_2$$

$$\beta=\cos\varphi_2\sin(\lambda_2-\lambda_1)$$

$$\gamma=-\sin\varphi_1\cos\varphi_2\cos(\lambda_2-\lambda_1)+\cos\varphi_1\sin\varphi_2$$

If P_3 is an arbitrary point on the sphere with coordinates (λ_3,φ_3) or rectangular coordinates:

$$x_3=\cos\varphi_3\cos\lambda_3$$

$$y_3=\cos\varphi_3\sin\lambda_3$$

$$z_3=\sin\varphi_3$$

P_3 is transformed relative to the new equator:

$$[R_{ij}][F']=[G]$$

where $F=[x_3,y_3,z_3]$ and the new coordinates are $[G]=[x_3',y_3',z_3']$. New longitude, latitude coordinates are therefore:

$$\varphi_3=\arcsin(\varphi_3)$$

$$\lambda_3=\begin{cases}\arcsin(\lambda_3) & \text{if } y_3\geq 0\\ -\arcsin(\lambda_3) & \text{if } y_3<0.\end{cases}$$

Appendix B

Determination of number of radii to be used in a table of radius versus arc length (km) for interpolation in quadrilateral area computations.

A circle of radius r with its center at a site will intersect one or more edges of a quadrilateral source area if $r_{min} < r < r_{max}$ where

r_{min} = distance from site to closest point (on edge or vertex) of quadrilateral;

r_{max} = distance from site to farthest vertex of quadrilateral.

The quadrilateral area contained within an annular ring centered at r is approximated by $A_r \approx angle\ r\ \Delta r$ where

angle = angle in radians included in that portion of the circle of radius r that lies wholly within the quadrilateral.

Δr = interval width.

We wish to determine the radii to be used in a table of radius versus arc length (km) in order that quadrilateral areas may be evaluated with sufficient accuracy (as discussed in the text) by linear interpolation of tabular values.

Let us evaluate angles at a fixed number n of radii, at $r_0 = r_{min}$ and at r_i, $1 \le i \le n$, where

$$r_1 = r_{min} + \frac{\Delta r}{2}, \quad r_2 = r_1 + \Delta r, \quad r_3 = r_2 + \Delta r, \quad r_n = r_{max} - \frac{\Delta r}{2}$$

$$n = \frac{r_{max} - r_{min}}{\Delta r}$$

We shall construct a table of radius versus arc length (angle · radius). We note that $angle(r)$ is a continuous function of radius r and

$angle(r) = 2\pi$ for $0 < r \le r_{min}$ if the site is contained within the source;

$angle(r_{min}) = 0$ if the site is outside the source area;

$angle(r_{max}) = 0$ in both cases.

Now let us assume a table of distance versus arc length has been created. We shall interpolate (as in equation 11 in the text) to estimate the area within the quadrilateral contained between a pair of bounding radii r_1 and r_2.

Expanding on equation 11, we illustrate by way of an example how the area is calculated using tabular values when the bounding radii are r_1 and r_2. Let d_i, $d_1 < d_2 < \ldots . d_n$ be the tabular distance values. Let $d_2 < r_1 < d_3$; $d_4 < r_2 < d_5$. Let $Inv(d)$=arc length interpolated in the table at distance d. Then the quadrilateral area contained between r_1 and r_2 may be approximated by

$$A \approx Inv\frac{(r_1+d_2)}{2}(d_2-r_1) + Inv\frac{(d_2+d_3)}{2}(d_3-d_2) + Inv\frac{(r_2+d_4)}{2}(r_2-d_4)$$

Summing arc areas gives an estimate of the total quadrilateral area. This estimate may be compared with the "true" quadrilateral area which is easily obtained as the sum of the areas of two triangles.

To test the accuracy of the area computations for a fixed number of radii, a set of 5656 quadrilateral-site pairs (used in a California study) were evaluated for n= 5, 7, 9 11, 13 radii per quadrilateral in the table.

The following table gives in each case the number of times for which the fractional error in

["arc estimate" area - "true" area] / "true" area

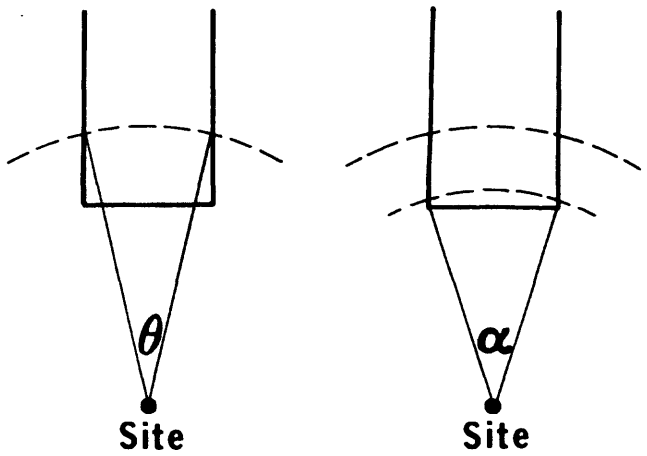
exceeded the value shown.

number with error >	number of radii used				
	5	7	9	11	13
.01	5391	4969	4042	2579	1729
.02	4873	3229	1634	726	274
.03	3962	1662	454	92	21
.04	2775	665	73	14	3
.05	1739	189	14	9	2
.06	903	38	8	5	0
.07	349	10	4	1	0
.08	143	9	1	0	0
.09	39	6	0	0	0
.10	8	2	0	0	0

Note that when as few as seven arcs per area are used, fewer than 5% of the areas are in error by more than 5%, (the largest single error being 11%.)

However, while the overall error may be considered satisfactorily small for as few as seven radii, in some cases the error in computing the area contained

in the intervals closest to the site was relatively large. Thus for example, for the quadrilateral and site shown, interpolating between zero at the closest edge and ϑr at the first arc cuts off a high fraction of the closest area and causes an underestimate of the acceleration densities at the highest accelerations.



We observed that computing the $\alpha r(angle \cdot distance)$ to the closest vertex in this case would largely solve the problem.

It became obvious that certain radii represented inflection points in arc length as a function of distance and should be included in the table. These were radii to vertices and perpendiculars to edges.

Therefore, we decided to determine arc lengths at radii for those vertices that are not at distance r_{min} or r_{max}. In addition, if the perpendicular from the site to a line containing an edge of the quadrilateral intersects the edge, we compute the arc length for a radius at this perpendicular distance.

We now repeated the above area test using a table including the radii to vertices and edges in addition to the radii at fixed intervals. The results were as follows:

number with error >	number of fixed radii				
	5	7	9	11	13
.01	617	356	175	98	58
.02	124	51	19	2	2
.03	40	5	0	0	0
.04	7	0	0	0	0
.05	0	0	0	0	0

We concluded that using seven fixed radii plus a variable number of radii (from two to six depending on the geometry) would suffice. The number of fixed radii "itst=7" is set in subroutine "rrisk"; it may be altered by recompiling.

Appendix C

Details of rupture distance computations.

The model permits a fault to consist of n connected straight line segments. The rupture may be wholly contained within one segment or overlap any number of fault segments. It may be as long as the fault, but cannot extend beyond the end of the fault. The distance from a rupture to a site is defined as the distance from the site to the point on the rupture nearest to the site.

Repeatedly determining site to rupture distances for ruptures of varying lengths and locations along a fault can be quite time consuming unless certain transformations are made. The following describes the transformations made in SEISRISK II and how they are used in the distance computations.

For an arbitrary fault segment $1 \leq i \leq n$, let:

$[x_1(i)\,,\,y_1(i)]\,,\,[x_2(i)\,,\,y_2(i)]$ = coordinates of endpoints of segment i

$L(i) = \sqrt{[x_1(i)-x_2(i)]^2+[y_1(i)-y_2(i)]^2}$ = length of segment i

(Fx,Fy) = coordinates of site

Coordinates are transformed so that fault endpoints

$[x_1(i),y_1(i)] \rightarrow (0,0)$

$[x_2(i),y_2(i)] \rightarrow [L(i),0]$

and the site

$(Fx,Fy) \rightarrow [X(i),P(i)]$

where

$$X(i) = [Fx-x_1(i)]\cos\alpha + [Fy-y_1(i)]\sin\alpha$$

$$P(i) = -[Fx-x_1(i)]\sin\alpha + [Fy-y_1(i)]\cos\alpha$$

= perpendicular distance from the site to line i.

$$\cos\alpha = \frac{x_2(i)-x_1(i)}{L(i)} \qquad \sin\alpha = \frac{y_2(i)-y_1(i)}{L(i)}$$

Once $L(i),X(i),P(i)$ have been determined the fault segment may be regarded as located on the x-axis extending from (0,0) to $[L(i),0]$ and the site

located at $[X(i), P(i)]$ (figure 9).

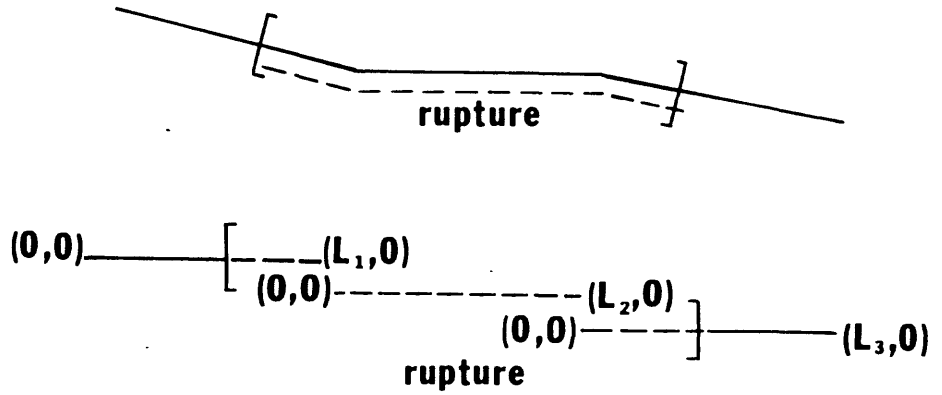


Figure 9. Fault segment and site in (a) original coordinates and (b) relocated so that segment lies on X-axis.

(a) Segment of fault extends from (x_1, y_1) to (x_2, y_2) ; site is at (F_x, F_y). Perpendicular from site intersects extended fault line (x_3, y_3).

(b) Fault and site moved so that fault extends from (0,0) to $(L,0)$. Site moved to (X,P).

We compute and save $L(i), X(i), P(i)$ for each segment of the fault, $1 \leq i \leq n$, and subsequently use these quantities rather than the original coordinates as follows:

Consider now an arbitrary rupture which has one end point on segment j and the second on segment k, where $1 \leq j \leq k \leq n$. Define the coordinates of the end points of the part of the rupture which lies on segment i (where $j \leq i \leq k$) as $[R_1(i), 0]$, $[R_2(i), 0]$ (figure 10).

$0 \leq R_1(i) \leq R_2(i) \leq L(i)$ for $i = j$ or $i = k$

$R_1(i) = 0$, $R_2(i) = L(i)$ for $j < i < k$ (If a rupture extends over three or more fault segments, the rupture overlap with the interior segments must be the entire segments.)

If the rupture overlaps segment i, let $CD(i)$ = square of closest distance from the site to the the rupture on segment i. When the rupture extends over several segments the value of $CD(i)$, which is the minimum for all such segments, is required. $CD(i)$ is defined as follows:

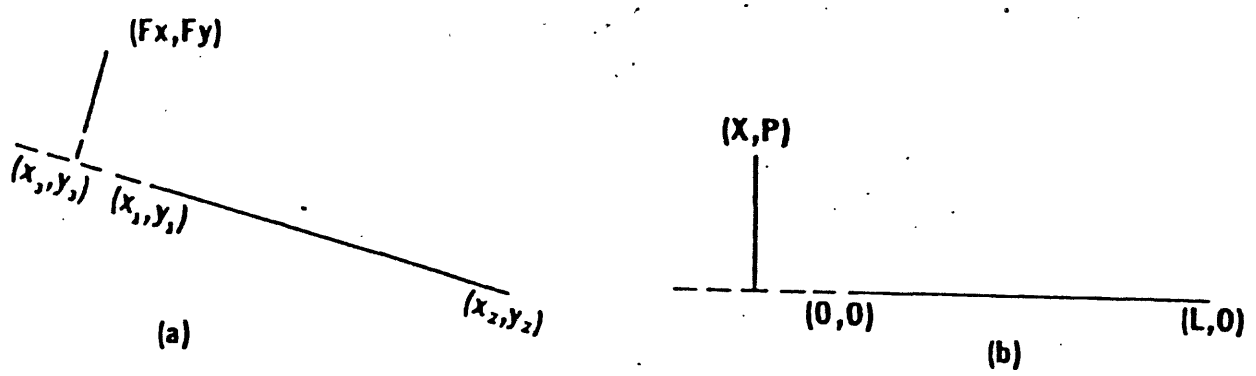


Figure 10. Rupture overlaps three fault segments. Each segment is relocated to lie on the X-axis. Endpoints of the part of the rupture on segment i are $[R_1(i), 0]$ and $[R_2(i), 0]$.

$$CD(i) = [R_1(i) - x(i)]^2 + P(i)^2 \quad \text{if } X(i) < R_1(i),$$

$$CD(i) = [X(i) - R_2(i)]^2 + P(i)^2 \quad \text{if } X(i) > L(i),$$

$$CD(i) = P(i)^2 \quad \text{if } R_1(i) \leq X(i) \leq R_2(i).$$

Thus, since we have saved $L(i)$, $P(i)$, $X(i)$, determining the (square of the) closest distance requires only a simple computation for each fault segment that the rupture overlaps. (Because the squares of the distances are compared, only one square root must be computed in determining closest distance.)

In the new system, the relative end and overlap points $R_1(i), R_2(i)$ are determined almost trivially from the total distance dl along the fault of the initial end point (start) of the given rupture and from the length of the rupture and fault segments.

Because the fault is continuous and the distance along the fault is the sum of preceding fault segment lengths to that point, define

$Rem(j) = dl - \sum_{i=0}^{j-1} L(i)$, where $L(0) = 0$. The first segment containing the rupture beginning at dl is the segment j for which $Rem(j) > 0$ and $Rem(j+1) < 0$. (In this case, $R_1(j) = Rem(j)$.) The segment containing the final end point of a rupture of length bl is determined similarly by substituting $(dl + bl)$ for (dl)

above.

In certain cases it is not necessary to compute accelerations at a set of distinct rupture locations, and accelerations at the site in the range $ac(j-1) < a \leq ac(j)$ resulting from ruptures of magnitude m earthquakes can be calculated directly. We shall discuss these cases using the notation already developed.

Case 1. $X(i) < 0$ for all $1 \leq i \leq n$ (or $L(i) < X(i)$ for all $1 \leq i \leq n$.)

In this case, the point on the rupture nearest the site is at one end of the rupture with the the distance to the site increasing (or decreasing) monotonically along the rupture.

If $X(i) < 0$ for all i, for any rupture overlapping segments j through k, where $j \leq k$, $[R_1(j),0]$ is the point closest to the site. Similarly if $X(i) > L(i)$ for all i, $(R_2(k),0)$ is the point of the rupture closest to the site.

Assume $X(i) < 0$, for all i.

($L(i) < X(i)$ could be considered in a symmetrical argument.)

In this case the nearest distance to the site is determined by $R_1(i)$ for the lowest i containing the rupture. Recall that for a rupture of length bl, rupture centers may be located at any distance dc along the fault for which

$$\frac{bl}{2} \leq dc \leq \frac{L-bl}{2},$$

or equivalently, the lowest end point of a rupture is at any distance dl for which

$$0 \leq dl \leq L - bl.$$

The acceleration produced by a rupture of magnitude m and length $bl_i(m)$ decreases as the rupture moves along the fault from a maximum acceleration $a_{max}[bl_i(m)]$ for a rupture with closest end point at $dl = 0$ to a minimum acceleration $a_{min}[bl_i(m)]$ for a rupture with closest end point at $dl = L - bl(m)$.

Therefore, for an acceleration range (or "box") such that

$$a_{min}[bl_i(m)] \leq ac(j-1) < a \leq ac(j) \leq a_{max}[bl_i(m)]$$

there is a corresponding interval along the fault $df_m(j) \leq dl \leq df_m(j-1)$ for magnitude m earthquakes in which (closest) end points of ruptures producing these accelerations may be located. Because these ruptures are at a distance $d_m(j) \leq d \leq d_m(j-1)$ from the site,

$$df_m(j) = \sqrt{d_m(j)^2 - P(i)^2} + X(i) \quad [X(i)<0]$$

$$df_m(j-1) = \sqrt{d_m(j-1)^2 - P(i)^2} + X(i).$$

Ruptures at distance along the fault $df(j) \leq dl < df(j-1)$ then contribute to accelerations in the range $ac(j-1) < a \leq ac(j)$ at the rate

$$rat[ac(j), bl_i(m)] = \rho(m) \frac{df_m(j-1) - df_m(j)}{L - bl_i(m)} frl\, p(i)$$

where:

frl = length of this fault/total length of faults in set

$p(i)$ = probability of rupture length $bl_i(m)$.

$\rho(m)$ = rate of magnitude m earthquakes for this fault.

$Rat(ac(j), bl_i(m))$ is the rate for ruptures of length $bl_i(m)$. For n rupture lengths, given that

$$bl_1(m) > bl_2(m) .... > bl_n(m),$$

there is a corresponding section of fault $0 \leq dl \leq L - bl_i(m)$ that may contain closest end points of ruptures and

$$L - bl_n(m) > L - bl_{n-1}(m) > ..... > L - bl_1(m).$$

Now define $df_{m,i}(j) = df_m(j)$ if $df_m(j) < L - bl_i(m)$

$= L - bl_i(m)$ otherwise, $1 \leq i \leq n$.

The total contribution to the acceleration $ac(j-1) < a \leq ac(j)$ then is the sum over rupture lengths for magnitude m ruptures:

$$rat(a) = \rho(m) \sum_{i=1}^{n} \frac{df_{m,i}(j) - df_{m,i}(j-1)}{L - bl_i(m)} frl\, p(i)$$

where $df_{m,i}(0) = 0$.

Thus, one need only compute one set of distances $df_m(j)$ per magnitude

along the fault rather than repeating the process for each rupture length, so that including rupture length variability in the calculation does not substantially increase the computation time.

Case 2. The distance from the fault to the site decreases monotonically as one moves a distance dc along the fault to some point on segment k [$(Ik,0)$ in the new system] and then increases steadily as one continues to the other end. This correponds to:

$X(i) < 0$ for $1 \leq i \leq k-1$

$$X(k) = Ik = dc - \sum_{i+1}^{k-1} L(i)$$

$X(i) > L(i)$ for $k+1 \leq i \leq n$. In this case, all ruptures of length $bl_i(m)$ with midpoints at a distance dl along the fault

$$dc - \frac{bl_i(m)}{2} < dl < dc + \frac{bl_i(m)}{2}$$

overlap dc and hence are regarded as being at the same closest distance from the site. The acceleration at the site produced by these magnitude m ruptures is the acceleration at distance $P(k)$, the perpendicular distance to the k^{th} segment. Because ruptures cannot extend past the end of the fault, the location of possible rupture centers is restricted to a part of the fault and the contribution at $P(k)$ becomes

$$rat[ac(j),m,bl_i(m)] = \rho(m)\, frl \sum_{i=1}^{n} \frac{Du_i(m) - Dl_i(m)}{L - bl_i(m)} p(i),$$

where

$$Du_i(m) = min[dc + \frac{bl_i(m)}{2}\ ,\ L - \frac{bl_i(m)}{2}]$$

$$Dl_i(m) = max[dc - \frac{bl_i(m)}{2}\ ,\ \frac{bl_i(m)}{2}].$$

Now the bookkeeping becomes more complicated, but we can look at all ruptures of length $bl_i(m)$ with centers at distance dl on the fault

$$\frac{bl_i(m)}{2} \leq dl \leq L - \frac{bl_i(m)}{2}$$

and remove those already accounted for, those with centers

$$dc - \frac{bl_i(m)}{2} \leq dl < dc + \frac{bl_i(m)}{2}.$$

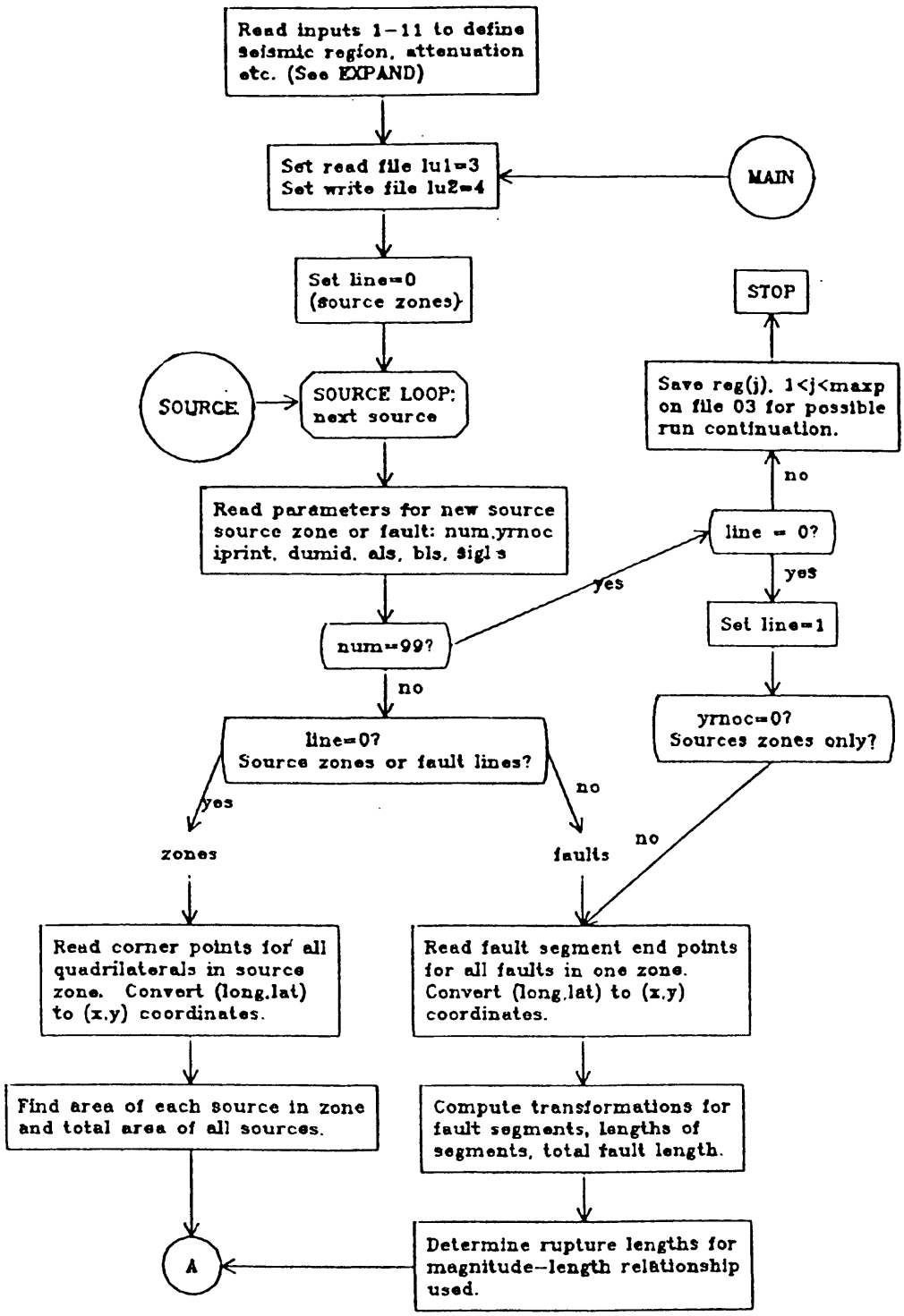
The remaining ruptures may be separated into two sets: those (if any) with centers $\frac{bl_i(m)}{2} \leq dl < dc - \frac{bl_i(m)}{2}$ and the those for which

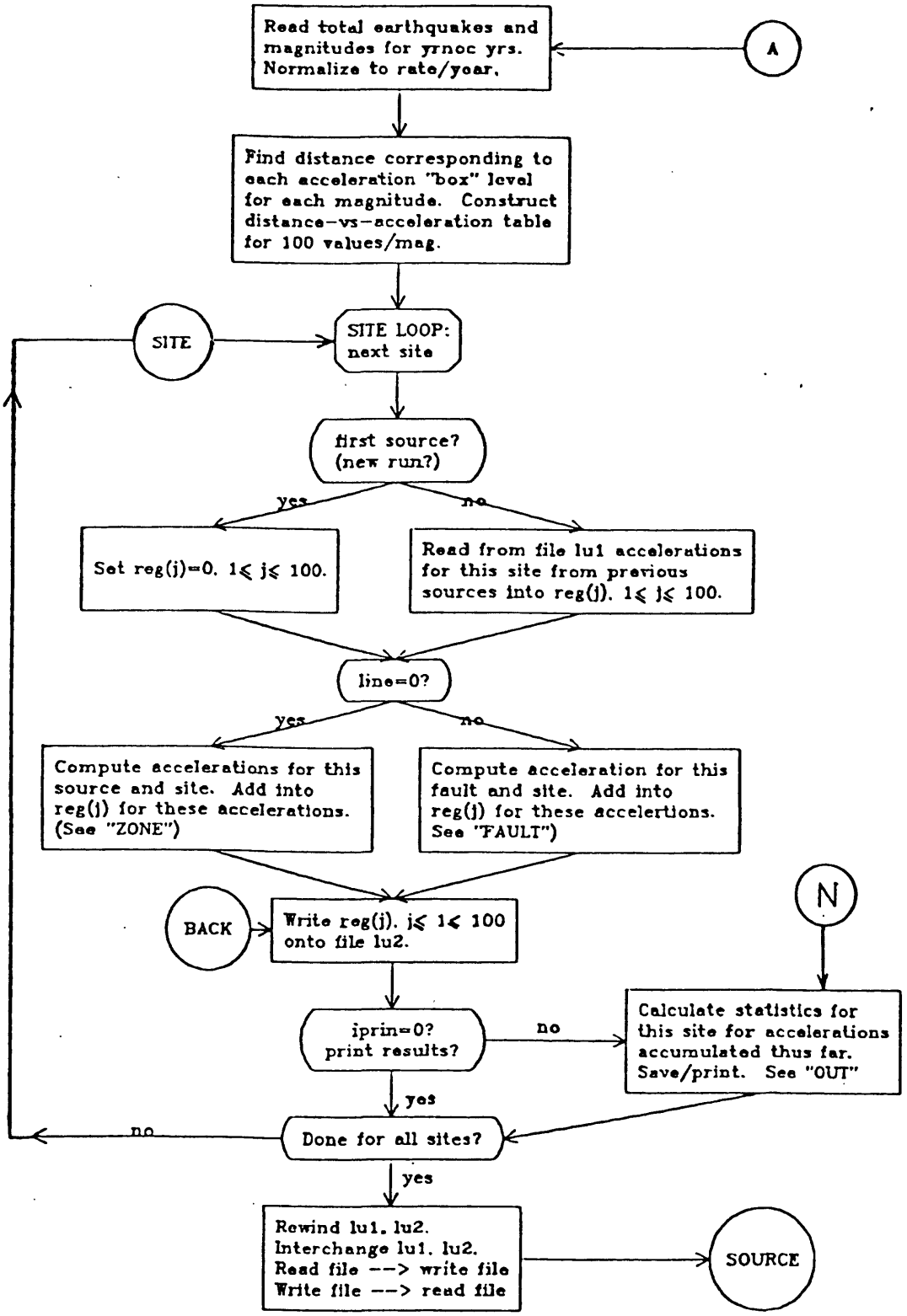
$$dc + \frac{bl_i(m)}{2} < dl \leq L - \frac{bl_i(m)}{2}.$$

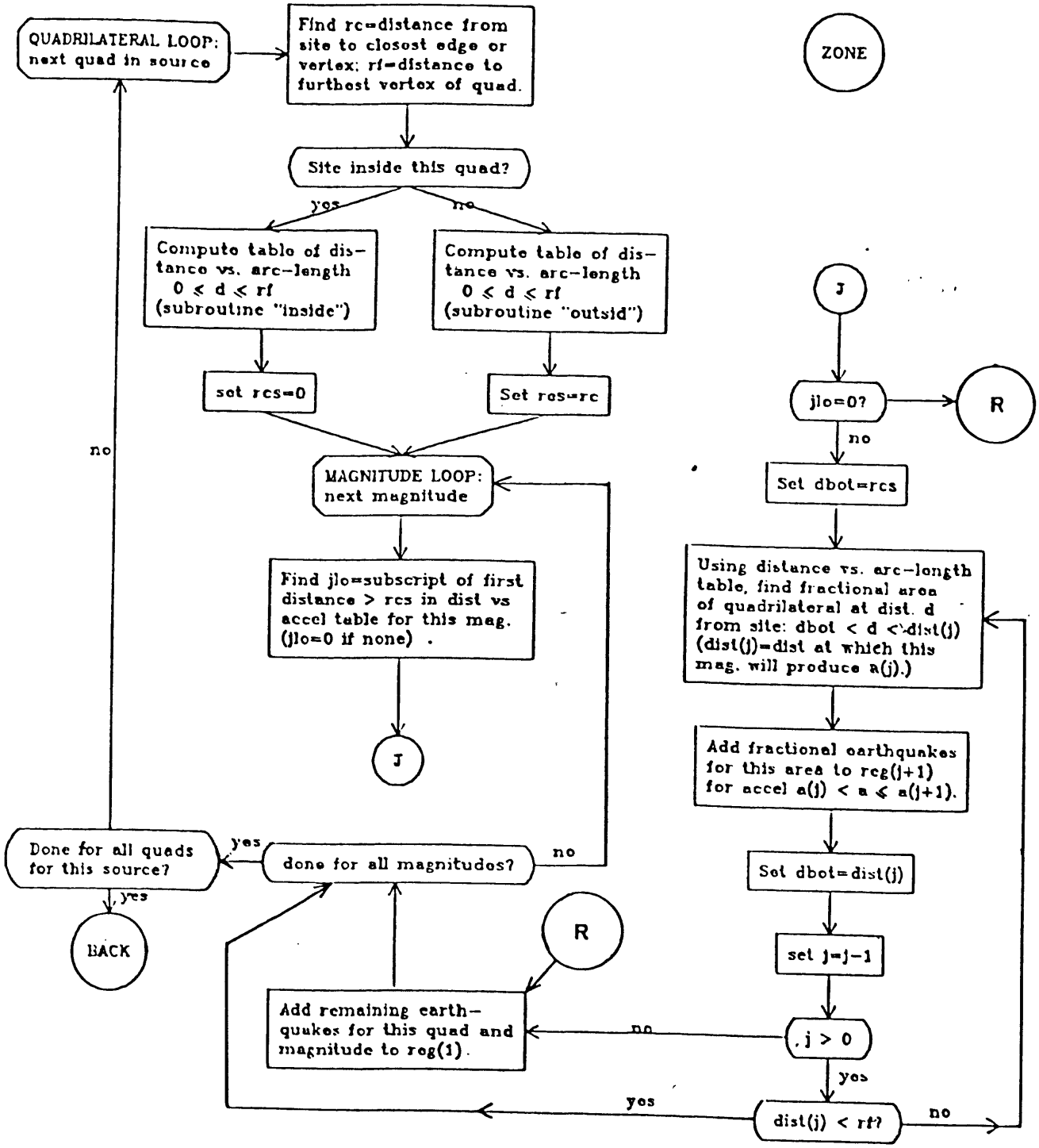
The latter two intervals correspond to ruptures whose closest end points to the site are at dl where $min[bl_i(m), dc] \leq dl < dc$ for one set of ruptures and at $dc < dl \leq max[dc, L - bl_i(m)]$ for the other set. These two sets of ruptures do not overlap and the fault may be regarded as consisting of two distinct segments in which the distance from the rupture to the site decreases (or increases) monotonically as one moves along the fault. Hence each of these segments may be treated separately as in Case I, and the accelerations resulting from ruptures along these segments may be added to those already accounted for.

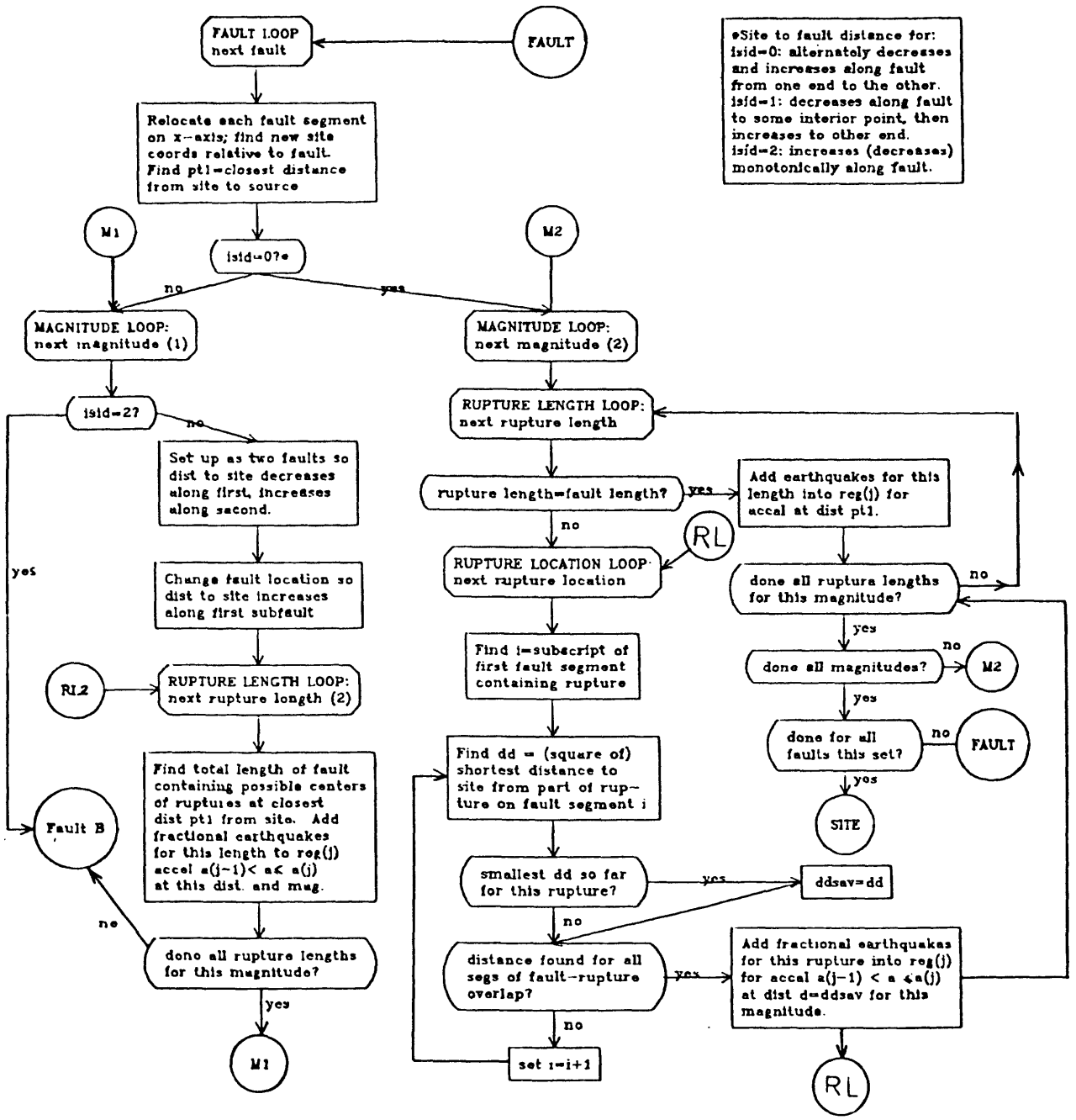
Appendix D

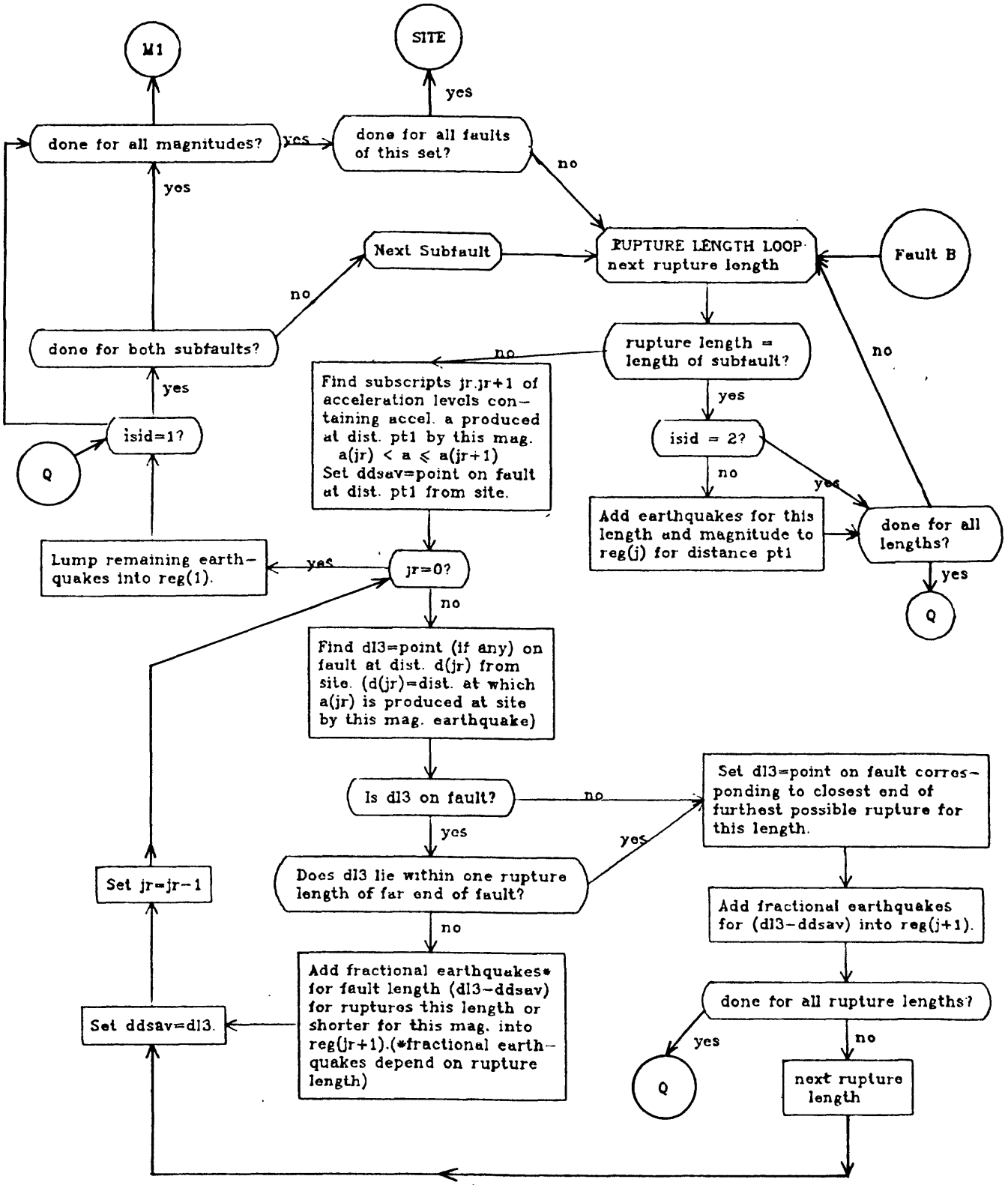
Flowchart

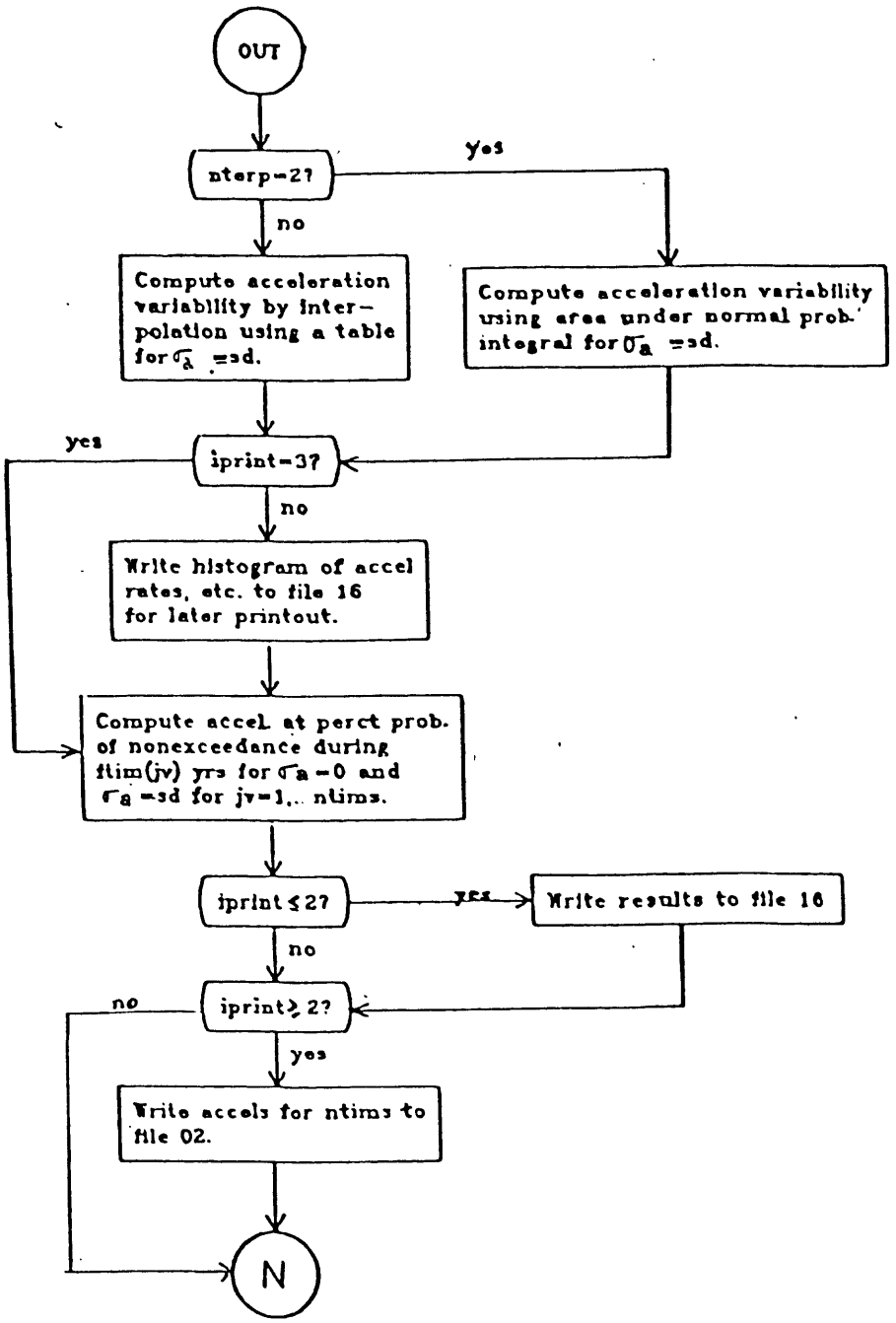

Read inputs 1-11 to define seismic region, attenuation etc. (See EXPAND)
Set read file lu1=3
Set write file lu2=4
MAIN
Set line=0 (source zones)
SOURCE
SOURCE LOOP: next source
Read parameters for new source source zone or fault: num,yrnoc iprint, dumid, als, bls, sigl s
num=99?
yes
no
line = 0?
yes
no
Save reg(j), 1<j<maxp on file 03 for possible run continuation.
STOP
Set line=1
yrnoc=0? Sources zones only?
no
line=0? Source zones or fault lines?
yes
no
zones
faults
Read corner points for all quadrilaterals in source zone. Convert (long,lat) to (x,y) coordinates.
Read fault segment end points for all faults in one zone. Convert (long,lat) to (x,y) coordinates.
Find area of each source in zone and total area of all sources.
Compute transformations for fault segments, lengths of segments, total fault length.
A
Determine rupture lengths for magnitude-length relationship used.

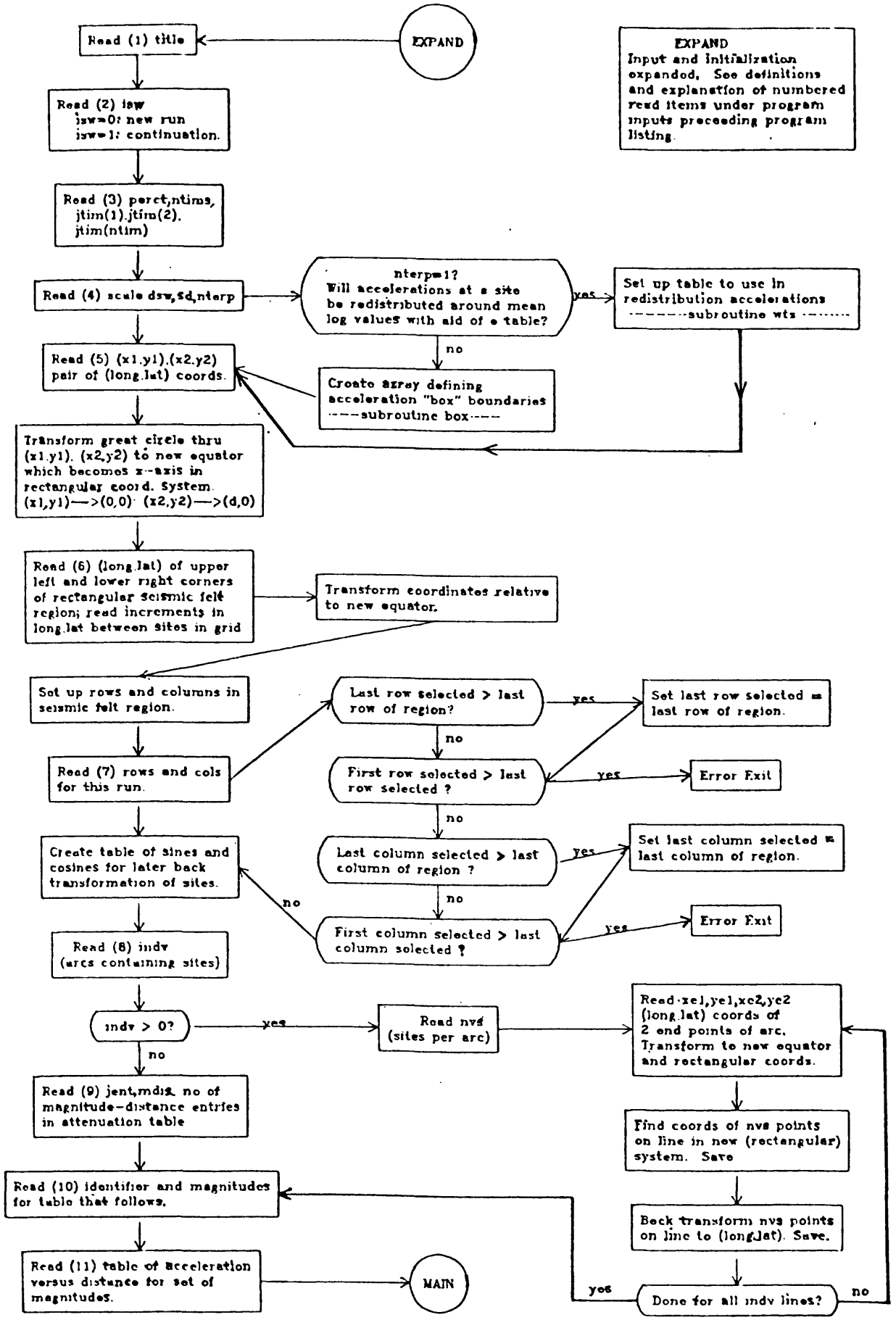

Read total earthquakes and magnitudes for yrnoc yrs. Normalize to rate/year.
A
Find distance corresponding to each acceleration "box" level for each magnitude. Construct distance-vs-acceleration table for 100 values/mag.
SITE
SITE LOOP: next site
first source? (new run?)
yes
no
Set reg(j)=0. 1≤ j≤ 100.
Read from file lu1 accelerations for this site from previous sources into reg(j). 1≤ j≤ 100.
line=0?
yes
no
Compute accelerations for this source and site. Add into reg(j) for these accelerations. (See "ZONE")
Compute acceleration for this fault and site. Add into reg(j) for these accelertions. See "FAULT")
BACK
Write reg(j). j≤ 1≤ 100 onto file lu2.
N
iprin=0? print results?
no
Calculate statistics for this site for accelerations accumulated thus far. Save/print. See "OUT"
yes
no
Done for all sites?
yes
Rewind lu1, lu2.
Interchange lu1, lu2.
Read file --> write file
Write file --> read file
SOURCE


QUADRILATERAL LOOP: next quad in source
Find rc=distance from site to closest edge or vertex; rf=distance to furthest vertex of quad.
ZONE
Site inside this quad?
yes
no
Compute table of distance vs. arc-length 0 ≤ d ≤ rf (subroutine "inside")
Compute table of distance vs. arc-length 0 ≤ d ≤ rf (subroutine "outsid")
J
jlo=0?
R
no
set rcs=0
Set rcs=rc
Set dbot=rcs
MAGNITUDE LOOP: next magnitude
Find jlo=subscript of first distance > rcs in dist vs accel table for this mag. (jlo=0 if none) .
Using distance vs. arc-length table, find fractional area of quadrilateral at dist. d from site: dbot < d < dist(j) (dist(j)=dist at which this mag. will produce a(j).)
J
Add fractional earthquakes for this area to reg(j+1) for accel a(j) < a ≤ a(j+1).
Done for all quads for this source?
yes
done for all magnitudes?
no
Set dbot=dist(j)
yes
BACK
R
set j=j-1
Add remaining earthquakes for this quad and magnitude to reg(1).
no
j > 0
yes
yes
dist(j) < rf?
no


FAULT LOOP
next fault
FAULT
Relocate each fault segment on x-axis; find new site coords relative to fault. Find pt1=closest distance from site to source
*Site to fault distance for:
isid=0: alternately decreases and increases along fault from one end to the other.
isid=1: decreases along fault to some interior point, then increases to other end.
isid=2: increases (decreases) monotonically along fault.
M1
isid=0?*
M2
no
yes
MAGNITUDE LOOP:
next magnitude (1)
MAGNITUDE LOOP:
next magnitude (2)
isid=2?
no
yes
Set up as two faults so dist to site decreases along first, increases along second.
Change fault location so dist to site increases along first subfault
RL2
RUPTURE LENGTH LOOP:
next rupture length (2)
Find total length of fault containing possible centers of ruptures at closest dist pt1 from site. Add fractional earthquakes for this length to reg(j) accel a(j-1)< a< a(j) at this dist. and mag.
Fault B
no
dono all rupture lengths for this magnitude?
yes
M1
RUPTURE LENGTH LOOP:
next rupture length
rupture length=fault length?
yes
Add earthquakes for this length into reg(j) for accel at dist pt1.
no
RL
RUPTURE LOCATION LOOP
next rupture location
Find i=subscript of first fault segment containing rupture
Find dd = (square of) shortest distance to site from part of rupture on fault segment i
smallest dd so far for this rupture?
yes
ddsav=dd
no
distance found for all segs of fault-rupture overlap?
yes
Add fractional earthquakes for this rupture into reg(j) for accel a(j-1) < a <a(j) at dist d=ddsav for this magnitude.
no
set i=i+1
RL
done all rupture lengths for this magnitude?
no
yes
done all magnitudes?
no
M2
yes
done for all faults this set?
no
FAULT
yes
SITE


M1
SITE
yes
done for all magnitudes?
yes
done for all faults of this set?
no
yes
RUPTURE LENGTH LOOP next rupture length
Fault B
Next Subfault
no
done for both subfaults?
rupture length = length of subfault?
no
no
yes
Find subscripts jr,jr+1 of acceleration levels containing accel. a produced at dist. pt1 by this mag. a(jr) < a ≤ a(jr+1) Set ddsav=point on fault at dist. pt1 from site.
isid=1?
yes
isid = 2?
Q
no
yes
Add earthquakes for this length and magnitude to reg(j) for distance pt1
done for all lengths?
yes
Q
Lump remaining earthquakes into reg(1).
yes
jr=0?
no
Find dl3=point (if any) on fault at dist. d(jr) from site. (d(jr)=dist. at which a(jr) is produced at site by this mag. earthquake)
Is dl3 on fault?
no
Set dl3=point on fault corresponding to closest end of furthest possible rupture for this length.
yes
yes
Does dl3 lie within one rupture length of far end of fault?
Add fractional earthquakes for (dl3-ddsav) into reg(j+1).
Set jr=jr-1
no
Add fractional earthquakes* for fault length (dl3-ddsav) for ruptures this length or shorter for this mag. into reg(jr+1).(*fractional earthquakes depend on rupture length)
done for all rupture lengths?
Set ddsav=dl3.
yes
no
Q
next rupture length


OUT
nterp=2?
yes
no
Compute acceleration variability by interpolation using a table for σ_a =sd.
Compute acceleration variability using area under normal prob. integral for σ_a =sd.
iprint=3?
yes
no
Write histogram of accel rates, etc. to file 16 for later printout.
Compute accel at perct prob. of nonexceedance during ftim(jv) yrs for σ_a =0 and σ_a =sd for jv=1,.. ntims.
iprint≤2?
yes
Write results to file 16
no
iprint≥2?
no
yes
Write accels for ntims to file 02.
N


EXPAND
Input and initialization expandod, See definitions and explanation of numbered read items under program inputs preceeding program listing.
EXPAND
Read (1) title
Read (2) isw
isw=0: new run
isw=1: continuation.
Read (3) perct,ntims, jtim(1).jtim(2). jtim(ntim)
Read (4) scale dsw,sd,nterp
nterp=1? Will accelerations at a site be redistributed around mean log values with aid of a table?
yes
Set up table to use in redistribution accelerations ------subroutine wts ------
no
Read (5) (x1,y1),(x2,y2) pair of (long,lat) coords.
Create array defining acceleration "box" boundaries ---subroutine box----
Transform great circle thru (x1,y1). (x2,y2) to new equator which becomes x-axis in rectangular coord. System. (x1,y1)—>(0,0) (x2,y2)—>(d,0)
Read (6) (long,lat) of upper left and lower right corners of rectangular seismic felt region; read increments in long,lat between sites in grid
Transform coordinates relative to new equator.
Set up rows and columns in seismic felt region.
Last row selected > last row of region?
yes
Set last row selected = last row of region.
no
Read (7) rows and cols for this run.
First row selected > last row selected ?
yes
Error Exit
no
Create table of sines and cosines for later back transformation of sites.
Last column selected > last column of region ?
yes
Set last column selected = last column of region.
no
First column selected > last column selected ?
yes
Error Exit
no
Read (8) indv (arcs containing sites)
indv > 0?
yes
Read nvs (sites per arc)
Read xe1,ye1,xe2,ye2 (long,lat) coords of 2 end points of arc. Transform to new equator and rectangular coords.
Find coords of nvs points on line in new (rectangular) system. Save
Back transform nvs points on line to (long,lat). Save.
Done for all indv lines?
yes
no
no
Read (9) jent,mdis no of magnitude-distance entries in attenuation table
Read (10) identifier and magnitudes for table that follows.
Read (11) table of acceleration versus distance for set of magnitudes.
MAIN

Appendix E

Example showing data inputs and outputs--sample problem.

Sample inputs and outputs are provided in this appendix. An explanation of the input variables (file 15) and formats is given at the beginning of the program listing in Appendix F.

The output (file 16) is organized to give location and ground motion statistics for each site. The site coordinates (longitude, latitude) in decimal degrees appear first, followed by the shortest distance from the site to the closest fault. Then two arrays are given, one for zero attenuation variability and one for attenuation variability σ_a. The columns in each array give for the ground motion (g.m.) in the i^{th} row:

occ/yr: occurrence rate per year of earthquakes producing motions at the site in the range $\text{g.m.}(i-1) < \text{g.m.} \leq \text{g.m.}(i)$.

exc/yr: exceedance rate per year of g.m.(i);

r(events): the average number of events required to produce an exceedance of g.m.(i), $\quad r(\text{events}) = \dfrac{\text{total yearly events}}{\text{exc/yr of g.m.}(i)}$;

r(yrs): return period in years, $r(\text{yrs}) = \dfrac{1}{\text{exc/yr of g.m.}(i)}$.

R(yrs) does not change for ground motions above level i if additional sources are added which do not produce ground motions larger than level g.m.(i) at the site. R(yrs) does change if all the earthquake rates are multiplied by a constant factor. R(events), on the other hand, changes if any source is added; however, it does not change if all the rates are multiplied by a constant factor.

Following the arrays is "total yearly events," the total number of earthquakes from all sources in this run. This total should be nearly the same at all sites; slight differences are due to inaccuracies in calculating areas and assigning fractional seismicity to these areas as discussed in the text. Large differences probably indicate an error in the computation.

The concluding lines for each site read

"xxx ext prob = yyy for zzz years": the extreme probability is xxx that ground motion yyy will not be exceeded in zzz years.

Included also in this appendix are a listing of a program to read (binary) output file 02 and the output of that program for the sample problem. File 02 contains only the summary information needed for mapping purposes: site coordinates (long, lat) and the ground motions calculated at the specified probability level for the times of interest (for both zero attenuation variability and one value of σ_a).

Sample input data on file 15:

```
SEISRISK EXAMPLE
0
.90 3 10 50 250
1.  0 .5 1
123.0 38.0 120.0 38.0
123.00     39.00      120.00     37.00       .100       .100
5 7 5 8
1
5
123.  39.  123.  38
6 12
'watashh79'          8.5        7 6        6.6        5.6        5.2        4.?
3.22        .74         .73         .67         .45         .195        .072
6.43        .64         .62         .53         .36         .135        .047
16.09       .49         .43         .32         .19         .052        .02
32.18       .36         .28         .17         .09         .02         .0052
64.3        .21         .14         .06         .035        .0051       .0013
96.5        .12         .07         .03         .0138       .0023       .00042
160.9       .045        .025        .015        .005        .00083      .0001
321.8       .013        .0076       .0026       .0012       .00021      .00003
643.0       .0034       .0019       .00065      .0003       .00005      .00001
1288.0      .00085      .00047      .00016      .00007      .00001      .00001
2570.0      .00021      .00012      .00004      .00002      .00001      .00001
5140.0      .0001       .00006      .00002      .00001      .00001      .00001
00    1.-1zn24
 12   1  1
117.43 34.13 117.20 34.21
118.17 34.52 118.00 34.66
118.71 34.71 118.63 34.86
119.05 34.77 119.00 34.92
119.31 34.83 119.20 34.97
119.63 35.00 119.49 35.11
121 34 36.60 121.20 36.72
121.77 36.88 121.60 37.00
122.20 37.22 122.00 37.32
123.85 38.97 123.68 39.05
124.12 39.32 123.89 39.40
124.28 39.90 124.03 39.95
.106   .281   .74    1.97
6 100 5.500 4.900 4.300
00 65.4-1zn25
  2   1   1
119 05 34.09 118.26 34.20
119.40 33 50 118.77 33.39
.031   .073   .172   .405   .955   2.248
7.300 6.700 6.100 5.500 4.900 4.300
00   1.0-1zn34
  7   1   2
120.40 34.68 120.00 34.78
120 56 34.93 120.28 35.00
120.78 35.20 120.38 35.39
121.00 35.48 121.00 36.00
121.51 36 00 121.47 36.50
121.82 36.45 121.80 36.80
```

```
122.00 36.73 122.21 37.11
  2  2  2
121.80 36.80 121.77 36.88
122.21 37.11 122.20 37.22
.0198 .0641 .2074 .6712
6.100 5.500 4.900 4.300
99  1.0-1ft24-1.085  .389  .50
 12  1  1
117.37    34.17     118.09    34.59     118.67    34.79     119.03    34.85
119.26    34.90     119.54    35.06     121.27    36.66     121.69    36.94
122.10    37.27     123.77    39.01     124.01    39.36     124.16    39.93
.0021 .0057 .015  .040
8.500 7.900 7.300 6.700
00 10.0+2ft39
  4  1  4
122.40    38.10     122.90    38.66     123.38    39.10     123.94    39.79
  3  2  4
122.60    38.68     123.10    39.20     123.74    39.95
  5  3  4
122.16    38.16     122.46    38.80     122.75    38.98     123.18    39.56
123.40    39.98
  3  4  4
122.50    39.10     122.80    39.25     122.95    39.43
.0072 .0200 .057
7.900 7.300 6.700
99
```

```
*************************************************************
Sample output on file 16.
(Detailed results are shown only for the first site.
Summarized results are provided on file 02 for all
sites in this example and are shown at end.)

SEISRISK EXAMPLE
 isw=0:  new run--no previous results included
 extreme probability  0.900
   for exposure times (years)    0   50  250
 scale factor for ground motion "box" levels=   1.00
 coordinates input in decimal degrees
    coordinates are printed in decimal degrees
 variability in attenuation, sigma=  0.50
 interpolation code for attenuation variability, nterp=1
 grid oriented parallel to great circle thru ( 123.00, 38.00),( 120.00, 38.00)
 corners of gridded area-upper left= 123.00,  39.00
                        lower right= 120.00,  37.00
 longitude increment= 0.1000 (decimal degrees)
 latitude increment = 0.1000 (decimal degrees)
 gridded region contains  21 rows,   24 cols
 for this run begin at col   5 end col   7, begin row   5 end row   8
 sites are also located on   1 line(s)
   5 sites per line
 line  1 end points at 123.000, 39.000 and 123.000, 38.000
 attenuation function watashh79
                              magnitude
```

```
   dist(km)     8.50       7.60       6.60       5.60       5.20       4.20
       3.22    0.74000    0.73000    0.67000    0.45000    0.19500    0.07200
       6.43    0.64000    0.62000    0.53000    0.36000    0.13500    0.04700
      16.09    0.49000    0.43000    0.32000    0.19000    0.05200    0.02000
      32.18    0.36000    0.28000    0.17000    0.09000    0.02000    0.00520
      64.30    0.21000    0.14000    0.06000    0.03500    0.00510    0.00130
      96.50    0.12000    0.07000    0.03000    0.01380    0.00230    0.00042
     160.90    0.04500    0.02500    0.01500    0.00500    0.00083    0.00010
     321.80    0.01300    0.00760    0.00260    0.00120    0.00021    0.00003
     643.00    0.00340    0.00190    0.00065    0.00030    0.00005    0.00001
    1288.00    0.00085    0.00047    0.00016    0.00007    0.00001    0.00001
    2570.00    0.00021    0.00012    0.00004    0.00002    0.00001    0.00001
    5140.00    0.00010    0.00006    0.00002    0.00001    0.00001    0.00001
 in the following
 iprint=-1: omit statistics, no printout for this source
 iprint =1: print cumulative histogram and statistics for earthquakes
            occurring in this and all sources previously input
 iprint =2: same as iprint =1 but also save summary on file 2
 iprint =3: summary information only on file 2

 coordinates are west longitude and latitude
Oyrnoc=      1.  iprint=-1 for area zn24
 source  1 of  1
    117.430     34.130    117.200     34.210
    118.170     34.520    118.000     34.660
    118.710     34.710    118.630     34.860
    119.050     34.770    119.000     34.920
    119.310     34.830    119.200     34.970
    119.630     35.000    119.490     35.110
    121.340     36.600    121.200     36.720
    121.770     36.880    121.600     37.000
    122.200     37.220    122.000     37.320
    123.850     38.970    123.680     39.050
    124.120     39.320    123.890     39.400
    124.280     39.900    124.030     39.950
 nr of levels of seismicity =  4
 earthquake rate / year
 occurrences=  0.106000  0.281000  0.740000  1.970000
  magnitudes=      6.10      5.50      4.90      4.30
 zn24 area=   17455. sq km, rate/sq km= 0.11286E-03 for mags  4.00- 4.60
Oyrnoc=     65.  iprint=-1 for area zn25
 source  1 of  1
    119.050     34.090    118.260     34.200
    119.400     33.500    118.770     33.390
 nr of levels of seismicity =  6
 before normalizing to rate/year
 occurrences=  0.031000  0.073000  0.172000  0.405000  0.955000  2.248000
 earthquake rate / year
 occurrences=  0.000474  0.001116  0.002630  0.006193  0.014602  0.034373
  magnitudes=      7.30      6.70      6.10      5.50      4.90      4.30
 zn25 area=    5130. sq km, rate/sq km= 0.67002E-05 for mags  4.00- 4.60
Oyrnoc=      1.  iprint=-1 for area zn34
 source  1 of  2
    120.400     34.680    120.000     34.780
    120.560     34.930    120.280     35.000
    120.780     35.200    120.380     35.390
```

```
    121.000     35.480    121.000     36.000
    121.510     36.000    121.470     36.500
    121.820     36.450    121.800     36.800
    122.000     36.730    122.210     37.110
 source  2 of  2
    121.800     36.800    121.770     36.880
    122.210     37.110    122.200     37.220
 nr of levels of seismicity =  4
 earthquake rate / year
 occurrences=  0.019800  0.064100  0.207400  0.671200
  magnitudes=      6.10      5.50      4.90      4.30
 zn34 area=    10075. sq km, rate/sq km= 0.66621E-04 for mags  4.00- 4.60
Oyrnoc=      1.  iprint=-1 for area ft24
 fault  1 of  1
     117.37   34.17,     118.09   34.59,     118.67   34.79,     119.03   34.85,
    119.26   34.90,     119.54   35.06,     121.27   36.66,     121.69   36.94,
    122.10   37.27,     123.77   39.01,     124.01   39.36,     124.16   39.93,
 nr of levels of seismicity =  4
 earthquake rate / year
 occurrences=  0.002100  0.005700  0.015000  0.040000
  magnitudes=      8.50      7.90      7.30      6.70
 fault rupture length parameters a1= -1.085 b1=  0.389  sig1=  0.50
Oyrnoc=     10.  iprint= 3 for area ft39
 fault  1 of  4
     122.40   38.10,     122.90   38.66,     123.38   39.10,     123.74   39.79,
 fault  2 of  4
     122.60   38.68,     123.10   39.20,     123.74   39.95,
 fault  3 of  4
     122.16   38.16,     122.46   38.80,     122.75   38.98,     123.18   39.56,
    123.40   39.98,
 fault  4 of  4
     122.50   39.10,     122.80   39.25,     122.95   39.43,
 nr of levels of seismicity =  3
 before normalizing to rate/year
 occurrences=  0.007200  0.020000  0.057000
 earthquake rate / year
 occurrences=  0.000720  0.002000  0.005700
  magnitudes=      7.90      7.30      6.70
```

SEISRISK EXAMPLE
site at long 122.415, lat 38.556
shortest dist to fault= 5.756 km

zero attenuation variability

g.m.	occ/yr	exc/yr	r(events)	r(yrs)
0.0100	4.07487	0.11539	36.3	8.7
0.0113	0.00802	0.10737	39.0	9.3
0.0131	0.00983	0.09754	43.0	10.3
0.0151	0.01029	0.08725	48.0	11.5
0.0173	0.01084	0.07641	54.8	13.1
0.0196	0.01210	0.06430	65.2	15.6
0.0222	0.00975	0.05455	76.8	18.3
0.0250	0.00706	0.04749	88.2	21.1
0.0280	0.00568	0.04181	100.2	23.9
0.0312	0.00420	0.03761	111.4	26.6
0.0347	0.00328	0.03433	122.0	29.1
0.0383	0.00328	0.03105	134.9	32.2
0.0422	0.00338	0.02767	151.4	36.1
0.0463	0.00295	0.02473	169.5	40.4
0.0507	0.00235	0.02238	187.3	44.7
0.0553	0.00214	0.02024	207.1	49.4
0.0600	0.00174	0.01850	226.5	54.1
0.0650	0.00168	0.01682	249.2	59.5
0.0702	0.00457	0.01225	342.1	81.6
0.0756	0.00055	0.01170	358.3	85.5
0.0813	0.00038	0.01132	370.3	88.4
0.0871	0.00038	0.01094	383.1	91.4
0.0931	0.00038	0.01055	397.0	94.8
0.0993	0.00040	0.01015	412.8	98.5
0.1056	0.00045	0.00970	432.1	103.1
0.1122	0.00063	0.00907	462.0	110.2
0.1189	0.00206	0.00701	597.8	142.7
0.1258	0.00027	0.00674	621.8	148.4
0.1328	0.00017	0.00657	638.1	152.3
0.1400	0.00018	0.00639	655.6	156.5
0.1473	0.00019	0.00620	675.4	161.2
0.1548	0.00020	0.00600	698.5	166.7
0.1624	0.00025	0.00575	729.1	174.0
0.1702	0.00140	0.00435	964.1	230.1
0.1781	0.00016	0.00419	1000.7	238.8
0.1862	0.00011	0.00407	1028.8	245.5
0.1943	0.00012	0.00395	1059.6	252.9
0.2026	0.00013	0.00382	1097.0	261.8
0.2110	0.00069	0.00313	1340.2	319.8
0.2196	0.00074	0.00239	1755.8	419.0
0.2282	0.00006	0.00233	1797.9	429.1
0.2370	0.00006	0.00227	1841.9	439.6
0.2459	0.00006	0.00222	1888.5	450.7
0.2549	0.00005	0.00217	1934.9	461.8
0.2640	0.00006	0.00211	1985.6	473.9
0 2733	0.00022	0.00189	2222.2	530.3
0 2827	0.00027	0.00162	2590.6	618.2
0.2921	0.00003	0.00159	2636.7	629.2
0.3018	0.00003	0.00156	2685.0	640.8
0.3115	0.00003	0.00153	2734.7	652.6
0.3213	0.00003	0.00150	2785.7	664.8
0.3313	0.00003	0.00147	2844.2	678.8
0 3414	0.00021	0.00126	3312.6	790.6
0.3517	0.00002	0.00125	3358.4	801.5
0.3620	0.00002	0.00123	3404.8	812.5
0.3725	0.00002	0.00121	3452.0	823.8

variability in atten, sigma= 0.50

g.m.	occ/yr	exc/yr	r(events)	r(yrs)
0.0100	4.07168	0.11859	35.3	8.4
0.0113	0.01046	0.10812	38.8	9.2
0.0131	0.01037	0.09775	42.9	10.2
0.0151	0.01111	0.08664	48.4	11.5
0.0173	0.00919	0.07745	54.1	12.9
0.0196	0.00820	0.06925	60.5	14.4
0.0222	0.00740	0.06186	67.7	16.2
0.0250	0.00729	0.05457	76.8	18.3
0.0280	0.00507	0.04950	84.7	20.2
0.0312	0.00587	0.04363	96.0	22.9
0.0347	0.00481	0.03882	107.9	25.8
0.0383	0.00438	0.03444	121.7	29.0
0.0422	0.00268	0.03176	131.9	31.5
0.0463	0.00372	0.02804	149.4	35.7
0.0507	0.00273	0.02531	165.5	39.5
0.0553	0.00289	0.02242	186.9	44.6
0.0600	0.00191	0.02051	204.3	48.8
0.0650	0.00190	0.01861	225.2	53.7
0.0702	0.00160	0.01701	246.3	58.8
0.0756	0.00106	0.01596	262.6	62.7
0.0813	0.00138	0.01458	287.5	68.6
0.0871	0.00135	0.01322	316.9	75.6
0.0931	0.00102	0.01220	343.4	82.0
0.0993	0.00102	0.01118	374.9	89.5
0.1056	0.00051	0.01067	392.8	93.7
0.1122	0.00102	0.00964	434.5	103.7
0.1189	0.00092	0.00873	480.1	114.6
0.1258	0.00033	0.00839	499.2	119.1
0.1328	0.00063	0.00776	539.7	128.8
0.1400	0.00066	0.00710	589.8	140.8
0.1473	0.00037	0.00673	622.2	148.5
0.1548	0.00040	0.00634	661.0	157.7
0.1624	0.00031	0.00603	694.7	165.8
0.1702	0.00044	0.00559	749.5	178.9
0.1781	0.00051	0.00508	825.5	197.0
0.1862	0.00027	0.00481	871.2	207.9
0.1943	0.00029	0.00452	927.1	221.3
0.2026	0.00024	0.00428	979.8	233.8
0.2110	0.00024	0.00404	1038.4	247.8
0.2196	0.00024	0.00380	1102.8	263.2
0.2282	0.00017	0.00363	1153.4	275.2
0.2370	0.00025	0.00338	1237.9	295.4
0.2459	0.00016	0.00323	1298.7	309.9
0.2549	0.00019	0.00303	1381.0	329.6
0.2640	0.00017	0.00286	1462.8	349.1
0.2733	0.00011	0.00275	1523.3	363.5
0.2827	0.00009	0.00266	1577.5	376.5
0.2921	0.00020	0.00246	1703.7	406.6
0.3018	0.00009	0.00237	1764.9	421.2
0.3115	0.00022	0.00216	1943.0	463.7
0.3213	0.00008	0.00208	2013.7	480.6
0.3313	0.00010	0.00198	2113.7	504.4
0.3414	0.00017	0.00182	2306.1	550.3
0.3517	0.00003	0.00179	2344.6	559.5
0.3620	0.00013	0.00166	2525.4	602.7
0.3725	0.00003	0.00163	2578.0	615.2

0.3835	0.00002	0.00120	3501.7	835.7		0.3835	0.00007	0.00155	2699.7	644.3
0.3950	0.00002	0.00118	3554.2	848.2		0.3950	0.00007	0.00148	2829.9	675.4
0.4070	0.00008	0.00110	3804.0	907.8		0.4070	0.00006	0.00142	2955.2	705.3
0.4197	0.00002	0.00109	3858.7	920.9		0.4197	0.00008	0.00134	3127.9	746.5
0.4330	0.00002	0.00107	3917.0	934.8		0.4330	0.00006	0.00128	3266.1	779.5
0.4469	0.00002	0.00105	3979.4	949.7		0.4469	0.00016	0.00112	3739.0	892.3
0.4616	0.00002	0.00104	4045.2	965.4		0.4616	0.00006	0.00106	3960.5	945.2
0.4771	0.00002	0.00102	4116.6	982.4		0.4771	0.00004	0.00102	4112.0	981.3
0.4935	0.00002	0.00100	4195.8	1001.3		0.4935	0.00007	0.00095	4431.1	1057.5
0.5108	0.00002	0.00098	4285.7	1022.8		0.5108	0.00003	0.00092	4562.0	1088.7
0.5291	0.00002	0.00095	4393.3	1048.5		0.5291	0.00007	0.00085	4947.0	1180.6
0.5485	0.00003	0.00093	4521.6	1079.1		0.5485	0.00008	0.00077	5468.4	1305.0
0.5690	0.00054	0.00039	10703.2	2554.3		0.5690	0.00009	0.00068	6189.4	1477.1
0.5909	0.00001	0.00038	10985.2	2621.6		0.5909	0.00005	0.00063	6643.1	1585.4
0.6142	0.00001	0.00037	11387.1	2717.5		0.6142	0.00005	0.00058	7185.6	1714.8
0.6391	0.00024	0.00013	32840.7	7837.4		0.6391	0.00006	0.00053	7954.7	1898.4
0.6656	0.00013	0.00000	0.0	0.0		0.6656	0.00002	0.00051	8283.6	1976.9
0.6940	0.00000	0.00000	0.0	0.0		0.6940	0.00006	0.00045	9404.1	2244.3
0.7245	0.00000	0.00000	0.0	0.0		0.7245	0.00004	0.00040	10405.0	2483.1
0.7571	0.00000	0.00000	0.0	0.0		0.7571	0.00005	0.00035	12004.2	2864.8
0.7923	0.00000	0.00000	0.0	0.0		0.7923	0.00002	0.00033	12828.6	3061.5
0.8302	0.00000	0.00000	0.0	0.0		0.8302	0.00006	0.00027	15566.5	3714.9
0.8711	0.00000	0.00000	0.0	0.0		0.8711	0.00001	0.00025	16461.5	3928.5
0.9154	0.00000	0.00000	0.0	0.0		0.9154	0.00006	0.00020	21154.4	5048.5
0.9634	0.00000	0.00000	0.0	0.0		0.9634	0.00001	0.00019	22469.3	5362.3
1.0156	0.00000	0.00000	0.0	0.0		1.0156	0.00004	0.00014	29439.6	7025.7
1.0724	0.00000	0.00000	0.0	0.0		1.0724	0.00001	0.00013	32853.9	7840.5
1.1344	0.00000	0.00000	0.0	0.0		1.1344	0.00004	0.00008	50108.3	11958.3
1.2023	0.00000	0.00000	0.0	0.0		1.2023	0.00001	0.00007	56173.9	13405.8
1.2768	0.00000	0.00000	0.0	0.0		1.2768	0.00001	0.00006	69503.8	16587.0
1.3587	0.00000	0.00000	0.0	0.0		1.3587	0.00001	0.00005	81428.5	19432.8
1.4490	0.00000	0.00000	0.0	0.0		1.4490	0.00000	0.00005	85910.1	20502.3
1.5489	0.00000	0.00000	0.0	0.0		1.5489	0.00003	0.00002	99999.9	49810.6
1.6596	0.00000	0.00000	0.0	0.0		1.6596	0.00001	0.00001	99999.9	99999.9
1.7828	0.00000	0.00000	0.0	0.0		1.7828	0.00001	0.00000	99999.9	99999.9

total yearly events 4 19026

zero attenuation variability

0.900 ext prob = 0.093 for 10 years
0.900 ext prob = 0.264 for 50 years
0.900 ext prob = 0.567 for 250 years

variability in atten, sigma=0.50

0.106 for 10 years
0.317 for 50 years
0.706 for 250 years

ratio 250 yr 0.900 extreme value to 10 yr val= 6.08 6.64

```
c    program to read summary file02 created by seisrisk ii.
c
c   file 02 contains for each site the level of ground motion
c   that has probability "perct" of not being exceeded during
c   "jtim" years for each of "ntim" values of "jtim".
c   ground motion values are given first for "zero variability in
c   attenuation" for all times "jtim", then for "variability
c   in attenuation  sigma = sd" for all values of "jtim".
c
c   first record contains:
c    irow1,irow2,icol1,icol2 = first and last rows, first and last
c           columns in gridded area containing sites for which
c           ground motions were calculated in seisrisk ii.
c    indvpt = total number of sites on lines
c    ntims = number of exposure times
c    jtims = exposure times in years
c    sd = standard deviation in attenuation variability.
c    perct: ground motion calculated has probability "perct"
c        of not being exceeded during "jtim" years
c
c    all records after first:
c    xlong,ylat = longitude, latitude of site in decimal degrees
c    sol(i) = ground motion at (xlat,ylong) corresponding to
c   "perct" probability and jtim(k) years where
c      i=1,2,.....ntim,    k=i  for zero attenuation variability
c      i = ntim+1, ntim+2.....2*ntim,   k=i-ntim
c                      for variability in attenuation sigma = sd
c
      dimension sol(48),jtim(24)
      read (2) irow1,irow2,icol1,icol2,indvpt,ntim, (jtim(jv),jv=1,
     &ntim),perct,sd
      ntim2=2*ntim
      write(44,10),(jtim(jv),jv=1,ntim),(jtim(jv),jv=1,ntim)
   10 format(30x'exposure times'//'  long     lat '6i9)
c     format for printout will depend on scale used for solution
   20 read (2,end=40) xlong,ylat,(sol(jt),jt=1,ntim2)
      write (44,30),xlong,ylat,(sol(jt),jt=1,ntim2)
   30 format(1h 2f7.3,6f9.3)
      go to 20
   40 call exit
      end
```

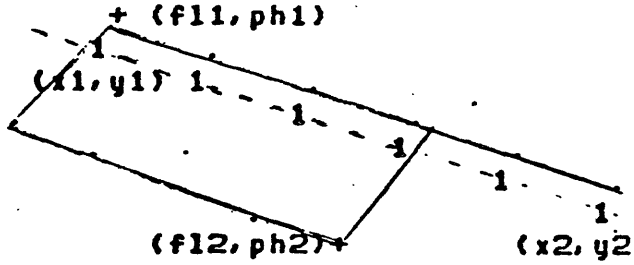
The following are the summarized results that were written to file 02 for the sample problem.

		exposure times					
long	lat	10	50	250	10	50	250
122.415	38.556	0.093	0.264	0.567	0.106	0.318	0.711
122.287	38.557	0.079	0.196	0.611	0.087	0.266	0.724
122.159	38.558	0.067	0.162	0.352	0.073	0.210	0.460
122.032	38.558	0.057	0.137	0.265	0.061	0.168	0.352
122.414	38.456	0.095	0.266	0.460	0.117	0.330	0.628
122.286	38.457	0.083	0.208	0.781	0.095	0.285	0.819
122.159	38.458	0.071	0.174	0.417	0.081	0.227	0.508
122.031	38.458	0.061	0.150	0.272	0.067	0.184	0.376
122.413	38.356	0.105	0.272	0.434	0.131	0.332	0.603
122.285	38.357	0.083	0.234	0.710	0.107	0.302	0.732
122.158	38.358	0.072	0.196	0.503	0.088	0.245	0.549
122.030	38.358	0.063	0.164	0.308	0.074	0.193	0.392
123.000	39.000	0.140	0.373	0.548	0.145	0.423	0.793
123.000	38.750	0.160	0.333	0.854	0.184	0.455	1.022
123.000	38.500	0.257	0.383	0.467	0.259	0.528	0.808
123.000	38.250	0.376	0.771	0.788	0.429	1.031	1.621
123.000	38.000	0.268	0.432	0.500	0.284	0.601	0.931

Appendix F

SEISRISK II Computer Program Listing

```
c                        SEISRISK II
c      computer program for determining ground motions
c          for use in seismic hazard mapping
c
c      files:
c         file 15:   inputs
c         file 16:   outputs
c         file 02:   summary for plots--see text
c         file 03:   intermediate results
c                    file 03 must be saved i f next run is to be
c                    a continuation of this run
c         file 04:   intermediate results
c
c                              inputs
c
c  1. title-up to 80 characters   (free field)
c  2. isw   (free field)
c      isw=0 new run (usual case)  isw=1 restart or continue
c         from previous run
c  3. prob,ntims, jtim(1), jtim(2).... jtim(ntims)
c     acceleration is sought for which there is probability prob of
c     being exceeded (1-prob of not) in jtim(1), jtim(2), ... jtim(ntim)
c     years
c     prob = extreme probability in decimal
c     ntims = number of times for which calculation is done
c     jtim(i)  =  durations  (years) for which extreme motions are to be
c              calculated at the prob probability level, 1 <= i <= ntims
c  4. scale,dsw,sd,nterp  (free field)
c        scale=scaling factor for ground motion boxes
c        scale=1 for motions .01 to 4.--scale accordingly
c        dsw:  =1 if inputs are degrees and minutes
c              =0 if inputs are decimal degrees
c        sd=standard deviation in log acceleration
c      for acceleration variability around mean value:
c        nterp: =1 interpolate in log acceleration using table
c               =2: determine area under normal probability integral
c                 more accurate, slower to run than interp=1
c  5. x1,y1, x2,y2 (long,lat) in decimal degrees (free field)
c     transform great circle thru (x1,y1), (x2,y2), (0,0)
c      to equator for new coordiate system
c  6. f11,ph1,f12,ph2,flinc,phinc
c     (f11,ph1) upper left    (f12,ph2) lower right  (long, lat)
c     corners of seismic felt region for risk computation
c     rectangular region with sides parallel to arc
c     of great circle thru (x1,y1), (x2,y2) (0,0) --defined by 3.
c     other two sides perpendicular and thru given end points
c          + (f11,ph1)
c
c     (x1,y1)
c
c
c
c
c
c           (f12,ph2)+                (x2,y2)
c
c
```



```
c      area within dots represents felt region
c     '1' s represents line joining (x1,y1) (x2,y2)
c     flinc,phinc (long,lat) increment in degrees (in new coordinate
c        system) for which risk is to be computed
c
c  7. irow1, irow2, icol1, icol2   (free field)
c        starting and ending rows and columns for this run
c        accelerations computed for sites in these rows
c        and columns--may be subset of seismic felt region defined
c       in input 6.    ^lso irow1=irow2=icol1=icol2=o permitted
c       if only individual sites on lines (input 6) are selected.
c  8. indv (free field)
c        number of line segments containing individual sites at
c        which acceleration is to be computed; zero if
c        only fixed grid is used.
c     If indv > 0 read next inputs, otherewise skip to input 8.
c      nvs (free field)
c        number of sites per line segment
c      xe1,ye1,xe2,ye2 (free field--indv cards, one pair per card)
c        end points (long, lat) of line segment:  sites will be
c        evenly spaced on line in new coordinate system
c  9. jent,mdis (free field)
c       jent=no of magnitudes for which acceleration is
c          tabulated as a function of distance   (maximum 8)
c      mdis=number of distances in attenuation table (maximum 20)
c  10. nam, tm (jent values of tm)
c       nam=identifier up to 10 characters-for example 'schn-seed'
c          identifies attenuation curves used
c       tm=magnitude for which table of distance versus acceleration
c         values follows  (free field)
c   *******magnitudes must be in descending order***
c  11. rtab(i), (atab(i,j),j=1,jent)   (mdis cards, i=1,mdis)
c     rtab(i)=ith tabular distance in kilometers from earthquake source
c     atab(i,j)=mean peak acceleration (or other ground motion parameter
c               at i th distance for j th magnitude.
c
c
c     source area inputs:
c
c     each source deck - -
c  1s. num, yrnoc, iprint, dumid,als,bls,sigls
c     format(i2, f5.0, i2,  a4, 2f6.2, f5.2)
c     num=0 for seismic source areas
c        =99 for 1s fault set
c        =0 for all fault sets after 1st
c        =99 at end of computation
c     yrnoc=number of years over which the earthquake occurrences
c               take place
c     iprint = -1  no statistics for this (intermediate) set of
c                  occurrences
c            =1 statistical calculations and printout
c            =2 same as=1 plus summary file for plot, etc (unit 2)
c            =3 omit printout; do summary file for plot (unit 2)
c    dumid = four character identifier for source zone or fault
c    als, bls, sigls ignored here but used for faults
c
```

```
c  2s.  jseg,ifr,itot
c    jseg=num of pairs of quadrilaters1 corner points in this source
c        set (jseg-1 quads).
c    seismicity to be apportioned by fractional area among itot sources.
c     ifr  = set number (ifr =1,2...itot in sequence)
c     itot=10 max;     total quad pairs for itot sources maximum 30
c  3s.  jseg quadrilateral endpoint cards (two points per card):
c     quad corner points (left long, left lat), (right long, right lat),
c     in decimal degrees (if dsw=0)  degrees, minutes (if dsw=1)
c
c       x1,y1 _____________ x2,y2
c            1             1
c            1             1
c            1             1
c            1             1
c       x3,y3 _____________ x4,y4
c              1           1
c               1           1
c                1           1
c                 1           1
c                  1_________1
c            x5,y5             x6,y6
c
c     jseg=3 in this example:  quad endpoint pairs are
c                    (x1,y1)---(x2,y2)
c                    (x3,y3)---(x4,y4)
c                    (x5,y5)---(x6,y6)
c     subregions of a set are defined as shown
c     subregions of one set are separate from those of other sets
c     repeat 2s, 3s for itot sources
c  4s.  noc(l)  l=1,lev  (lev=12f6.2)   (lev=12 maximum)
c     numbers of events expected in yrnoc years in each
c          magnitude interval for earthquake occurrences
c  5s.  fm(l), l=1,lev  (12f6.2)
c     fm(l)=center of magnitude interval for which noc(l) occur
c
c       repeat 1s. thru 5s. for remaining sources
c
c          fault inputs:
c
c  1f.   num, yrnoc, iprint, dumid, als, bls, sigls
c     parameters defined as in 1s. above
c  **note num=99 for first fault, =0 thereafter for successive faults
c     als,bls,sigls rupture length parameters
c     if als, bls non zero:
c          length=10**(als +bls*m +fr*sigl)
c          where m=magnitude;    sigl=standard deviation (in log length)
c     fr=(normally) 5 values in range (-2,+2)  if sigl non zero
c     fr=0 (1 value)--mean rupture length only if sigl=0
c     if als,bls,sigls=0 (or blank) previous values are used
c     if no previous values input, default valuesare used:
c          als=-1.085     bls=.389    sigls=.52
c     if als not zero, bls=0, sigl=0 defaults to Algermissen-Perkins
c     old single rupture length vaules (for consistency with original
c     risk map runs)
c             length=.000630*exp(1.52*m)
```

```
c 2f.  jseg, ifr, itot  (free field)
c     jseg=num of fault segment end points to be connected
c         into single fault   (jseg-1 segments)
c     ifr, itot as defined in 2s
c 3f.  (xl(i,i(fr),yl(i,i(fr), i=1,jseg)   (free field)
c       xl=long  yl=lat (degrees)  (ifr=fault number)
c      jseg=24 max          itot=26 max
c 4f.  noc(1)   same as in 4s.
c 5f.  fm(1) same as 5s.
c     repeat 2f, 3f for itot faults
c
c     repeat 1f. thru 5f. for remaining faults
c
c     end with num=99 (omit other inputs on line)
c * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
c
      character  idsw*16, nam*10,tytle*80,dumid*4
      dimension idsw(2),ata(20,15),ibr(10),lsub(26),dtot(26)
      dimension xpt2(500),ypt2(500),xpt1(500),ypt1(500)
      dimension prls(10),rlls(10,12,
      dimension nrlsv(12),prlrl(10,12),rll(10,12)
      dimension disl(50),angl(50)
      common/inout/mdum(4),stsize
      common/slinds/sld1(4),perpa(4),sld2(4),sllq(4),
     & mina(4),perpaq(4),sldum2(8)
      common/extra/isid,shdis,distl(24),inear
      common /xlyl/xll(4),yl(4),xr(4),yr(4),xc(4),yc(4)
      common/itabs/dtab(105,12),maxa(12)
      common/wty/ wt(101),dev(101)
      dimension tmdif(8),atab(20,8),rtab(20),tm(8),atadif(20,8)
       dimension rtdif(20)
      common/xyarea/xsav(2,50),ysav(2,50),sarea(50)
      dimension reg(106)
      common/wds/max,maxp
      common/ext/prob,ex,exl
      common/tims/ftim(20),jtim(20),ntims,nwt,iprint
       common/sincos/sinx(100),cosx(100),siny(100),cosy(100)
      common/quad/maxq,ac(105),aclog(105)
      dimension x(4),y(4),noc(12),fmag(13),fm(13),fml(13)
      common/linps/dist(24,26),disq(24,26),jseg,jsegm,
     & xl(24,26),yp(24,26),coa(24,26),sia(24,26)
      common/lindis/dls(24),perp(24),min(24),persq(24)
      dimension jm(26), disrem(24)
       real noc
       integer dsw
      data linset,idef,line,nrls/0,0,0,0/
      data idsw/'decimal degrees ','degrees, minutes'/
      data r,pi2,pi,pih/6378.,6.2831854,3.1415927,1.570963/
      rad=pi/180.
c     input on unit 15
      rewind 15
c     units 3 and 4 used for intermediate results
c     saved on unit 3 for later use or continuation
      rewind 3
      rewind 4
      allerr=0.
```

```
   10 format(i1,i4,f10.5,2i2,a12,a12)
      dlevsv=0.
      flevst=0.
      levno=0.
      tot5=0.
      read(15,20) tytle
   20 format(a80)
      write(16,2420) tytle
c  isw tells whether you include data from a previous run, zero if no.
      read(15,*)isw
      if(isw.eq.0) write(16,30)
   30 format(' isw=0: new run--no previous results included')
      if(isw.eq.1) write (16,40)
   40 format(' isw=1: run continuation; add to previous results')
c     prob=extreme probability in decimal
c     ntims are number of 'jtims' for which calculation is done
c     jtims are the durations for which the extreme motions are
c     to be calculated at the prob extreme probability level
      read (15,*) prob, ntims, (jtim(jv),jv=1,ntims)
      if (prob.lt.1.) go to 60
      write(16,50) prob
   50 format(' prob ='f8.3' must be decimal less than one')
      call exit
   60 continue
      write (16,70) prob,(jtim(jv),jv=1,ntims)
   70 format(' extreme probability 'f6.3
     &/'   for exposure times (years) '2(10i5/))
      ex=-alog(prob)
      ex1=alog(ex)
      do 80 jv=1,ntims
   80 ftim(jv)=jtim(jv)
c
      read(15,*) scale,dsw,sd,nterp
      write(16,90) scale,idsw(dsw+1),idsw(dsw+1),sd,nterp
   90 format(' scale factor for ground motion "box" levels='
     &f7.2/' coordinates input in ' a16,
     &/'    coordinates are printed in 'a16,
     &/' variability in attenuation, sigma='f6.2,
     &/' interpolation code for attenuation variability, nterp='i1)
c        scale=multiplicative factor for acceleration levels
c        dsw=0 if input is decimal degrees; =1 for degrees,minutes
c        sd=standard deviation for log acceleration
c        nterp=1:  use table for interpolating acceleration variability
c              =2: compute acceleration variability (slower-more accurate
c
c   read (x1,y1), (x2,y2) =long,lat in dec. degrees of 2 points
c   to transform to equator
c      (x1,y1) to (0.,0)(x2,y2) to (dist. between points,0)
  100 read(15,*) x1,y1,x2,y2
      write (16,110) x1,y1,x2,y2
  110 format(' grid oriented parallel to great circle thru ('
     & f7.2','f6.2'),('f7.2','f6.2')')
c  convert to decimal degrees unless dsw=0.
      if(dsw.eq.0) go to 120
      call condec(x1)
      call condec(x2)
```

```
      call condec(y1)
      call condec(y2)
  120 continue
      x1=x1*rad
      x2=x2*rad
      y1=y1*rad
      y2=y2*rad
c    transform to equator
      call inieqr(x2,y2,x1,y1)
c
c   read in upper left and lower right corners of affected rectangle
c   -- degrees, minutes ie 20.30=20 degrees,30 minutes (dsw=1)
c        (decimal degrees 20.50 = 20 1/2 degrees (dsw=0) )
c   opposite corners of gridded area (in latitude and longitude)
c   gridded area becomes rectangle surrounding new equator.
c   sites are at uniform increments in latitude and longitude
c   within this gridded area in new coordinate system.
c   in order to limit affected area you can specify
c    beginning and ending rows and columns as irowi and 2
c    and icoli and 2 (next input card)
c       ph1=phi=latitude
c       fl1=lambda=longitude
c
      read(15,*) fl1,ph1,fl2,ph2,flinc,phinc
      write (16,130) fl1,ph1,fl2,ph2
  130 format(' corners of gridded area-upper left='f7.2','f7.2,
     &/23x' lower right='f7.2','f7.2)
c
c   convert to decimal degrees (unless dsw=0)
      if(dsw.eq.0) go to 140
      call condec(ph1)
      call condec(fl1)
      call condec(ph2)
      call condec(fl2)
      call condec(phinc)
      call condec(flinc)
  140 continue
c
c     determine number of subregions in map area and set up subscripting
c     irow=no of rows
c     icol=no of cols
      write(16,150) flinc,phinc
  150 format(' longitude increment='f7.4' (decimal degrees)'
     &/' latitude increment ='f7.4' (decimal degrees)')
      flinc=flinc*rad
      phinc=phinc*rad
      dphinc=phinc*r
      dflinc=flinc*r
      fl1=fl1*rad
      ph1=ph1*rad
      fl2=fl2*rad
      ph2=ph2*rad
c     transform to equator and x-y rectangular coordinate system
c     convert to kilometers
      call toeqr(fl1,ph1,flam1,phi1)
      call to eqr(fl2,ph2,flam2,phi2)
```

```
      if(phi1.gt.0) go to 180
      if(phi2.gt.0) go to 190
c     apparent error in inputs or xformataion
  160 write(16,170)
  170 format(' probable input error--felt region upper left,'
     &' lower right'/' must be on opposite sides of new equator')
      call exit
  180  if(phi2.le.0) go to 200
      go to 160
  190 phs=phi1
      phi1=phi2
      phi2=phs
      fls=flam1
      flam1=flam2
      flam2=fls
c     have proper orientation--continue
  200 ph1=phi1
      if(flam1.lt.0) go to 210
      fl1=flam1
c    find total number of cols
      go to 230
  210 write(16,220)flam1,flam2
  220 format(' apparent error in xforming lat and long limits'
     & 'to eqr--'/'  flam1='f9.3'  fl2='f9.3)
      call exit
  230 continue
c     set up rows and columns
      irow=(ph1-phi2)/phinc+1.01
      icol=(fl1-flam2)/flinc+1.01
      if(irow.le.99) go to 260
  240 write(16,250) irow,icol
  250 format( 1h i3' rows'i5' cols needed -redimension sinx,siny'
     &',cosx,cosy arrays  and fix print test')
      call exit
  260 if(icol.gt.99) go to 240
      write(16,270)irow,icol
  270 format(' gridded region contains'i4,' rows, 'i4' cols')
      read(15,*)irow1,irow2,icol1,icol2
c
c    limits rows and columns of felt points for which calculation is
c    to be made.
c
      write(16,280)irow1,irow2,icol1,icol2
  280 format(' for this run begin at col 'i3' end col 'i3
     & ', begin row 'i3' end row 'i3)
c   read in number of line segments containing sites at which
c   calculation is to be done  (in addition to or instead of
c   computing for sites on a fixed grid)
c   zero if no line segments
      read(15,*)indv
      write(16,290) indv
  290 format(' sites are also located on 'i3' line(s)')
      if (indv.eq.0) go to 370
c     calculate accelerations at nvs sites on each of indv lines
c     read in end point pairs (long,lat) xe1,ye1,xe2,ye2
c     for each of indv lines
```

```
c      nvs sites are evenly spaced on line*
c      nvs=2 gives accelerations at end points of line;
c      nvs=3 acceleration at 2 end points plus center of line,etc.
       read(15,*) nvs
       write(16,300) nvs
  300 format(i4' sites per line')
       indvpt=0
       do 360 i=1,indv
       read(15,*) xe1,ye1,xe2,ye2
       write(16,310) i xe1,ye1,xe2,ye2
  310 format(' line 'i2' end points at 'f7.3','f7.3' and 'f7.3
     &','f7.3)
       if(dsw.eq.0) go to 320
       call condec(xe1)
       call condec(xe2)
       call condec(ye1)
       call condec(ye2)
  320 continue
       xe1=xe1*rad
       xe2=xe2*rad
       ye1=ye1*rad
       ye2=ye2*rad
       call toeqr(xe1,ye1,xo1,yo1)
       call toeqr(xe2,ye2,xo2,yo2)
       fr=0.
       if (nvs.gt.1) frinc=1./(nvs-1)
       do 350 iq=1,nvs
       indvpt=indvpt+1
       if(indvpt.le.500) go to 340
       write(16,330)
  330 format(' more than 500 single points requested as sites'
     & 1x,'do 500')
       go to 370
  340 x11=xo1+fr*(xo2-xo1)
       yp1=yo1+fr*(yo2-yo1)
       cosy(1)=cos(yp1)
       siny(1)=sin(yp1)
       sinx(1)=sin(x11)
       cosx(1)=cos(x11)
       call backw(1,1,xpt1(indvpt),ypt1(indvpt))
       xpt2(indvpt)=x11*r
       ypt2(indvpt)=yp1*r
       fr=fr+frinc
  350 continue
  360 continue
  370 continue
c
c      jent=number of magnitude entries in table of acceleration as a
c           function of magnitude and distance
c      mdis=number of distance entries
       read(15,*) jent,mdis
       if (jent.le.8) go to 390
       write (16,380) jent
  380 format(' jent='i3' too large--max 8 allowed')
       call exit
  390 if (mdis.le.20) go to 410
```

```
      write(16,400) mdis
  400 format(' mdis='i4'  too large--max 20 distances allowed')
      call exit
  410 continue
c     read identifier and magnitudes for which acceleration versus
c     distance table follows.  magnitudes must be in decreasing order
c
      read(15,*)nam,(tm(i),i=1,jent)
c    print magnitude headings for attenuation function.
      write (16,420) nam
  420 format(' attenuation function 'a10/30x'magnitude')
      write(16,430) (tm(i),i=1,jent)
  430 format('   dist(km) 'f7.2,9f10.2)
c    since interpolation in attenuation function tables is in logs of
c    magnitudes, transform magnitudes to log magnitudes.
      do 440 ll=1,jent
  440 tm(ll)=alog(tm(ll))
c  read in table of acceleration vs distance values for set
c  of magnitudes (distance increasing in table)
      do 500 m=1,mdis
      read(15,*) rtab(m),(atab(m,j),j=1,jent)
      if(atab(1,1).lt.10) go to 460
      write (16,450) rtab(m),(atab(m,j),j=1,jent)
  450 format(1h f10.2,8f10.2)
      go to 480
  460 write(16,470) rtab(m),(atab(m,j),j=1,jent)
  470 format(1h f10.2,8f10.5)
c  take logs of tabular values since interp. is in log acc. etc.
  480 do 490 j=1,jent
  490 atab(m,j)=alog(atab(m,j))
  500 continue
      do 510 m=2,mdis
  510 rtdif(m)=rtab(m)-rtab(m-1)
      ll=3
      l2=4
c    set up weights etc for distribution of acceleration values
c    if nterp=2, omit
      if (nterp.eq.2) go to 520
      call wts(nwt,sd)
c
c      define limits for acceleration boxes:   accelerations in the
c      range ac(j-1) < ac < ac(j) will be collected the jth box.
c      max=number of boxes;  acceleration values too large for
c      any of the box levels correspond to maxp=max+1
c      as here constituted boxes have limits ranging from
c     .01 to about 4.  Greatest accuracy is in range (about)
c     .2 to .7.  For motions exceeding 4(g) of 4(m/sec) provide
c     appropriate distance vs ground table and select 'scale'
c     for multiplying boxes to get consistent range
c
  520 call box(scale,sd)
      ac(maxp)=9.999*scale
c
c     basic step size for distances in source areas
      do 530 i=1,maxp
  530 aclog(i)=alog(ac(i))
```

```
c    number of digits after decimal for print of
c      accelerations (subroutine out) determined by scale
       naf=4
       if(scale.ge.100 ) naf=2
c
c      reset irow2, icol2 to max. allowed if too large
       if(irow2.gt.irow) irow2=irow
       if(icol2.gt.icol) icol2=icol
       if(irow1.le.irow2) go to 550
       write(16,540) irow1
  540  format(' irow1 ='i4'  too large--error stop')
       call exit
  550  if(icol1.le.icol2) go to 570
       write(16,560) icol1
  560  format(' icol1 ='i4'   too large--error stop')
       call exit
c      identify rows and cols for file02
  570  write(2) irow1,irow2,icol1,icol2,indvpt,ntims,(jtim(jv),jv=1,
      &ntims),prob,sd
c      create table of sines and cosines for later back transformation
c      of felt points
       if(icol.eq.0) go to 610
       yp1=(ph1-(irow1-.5)*phinc)
       x11=(fl1-(icol1-.5)*flinc)
       irow2p=irow2+1
       icol2p=icol2+1
       do 580 ira=irow1,irow2p
       cosy(ira)=cos(yp1)
       siny(ira)=sin(yp1)
  580  yp1=yp1-phinc
       do 590 ii=icol1,icol2p
       sinx(ii)=sin(x11)
       cosx(ii)=cos(x11)
  590  x11=x11-flinc
       write (16,600)
  600  format(' in the following'/' iprint=-1: omit statistics, no
      & printout for this source'/' iprint =1: print cumulative
      & histogram and statistics for earthquakes '
      &/11x,' occurring in this and all sources previously input'
      &/ ' iprint =2: same as iprint =1 but also save summary on file 2'
      &/ ' iprint =3: summary information only on file 2'
      &// ' coordinates are west longitude and latitude')
c
c
c   start new source area (or fault line) computation
  610  read(15,630)num,yrnoc,iprint,dumid,als,bls,sigls
c      num=0 for seismic sources
c         =99 for first fault set
c         =0 for all faults after first set
c         =99 at end of computation (last input)
c      yrnoc=number of years over which the earthquake occurrences
c           input for this source region (fault set) take place
c
c   iprint = zero if statistics are not to be calculated at the
c      end of computation for this source area (fault)
c      for accelerations accumulated thus far
```

```
c  iprint = one for statistical calculations and printout
c  iprint =2 same as =1 plus summary file for plot (unit 2)
c  iprint=3 omit printout; summary file for plot (unit 2)
c
  620 format('0yrnoc= 'f6.0,' iprint='i2,' for area 'a4)
  630 format(i2,f5.0,i2 ,a4,2f6.2,f5.2)
c
c
      if(num.ne.99) go to 640
      if(line.eq.1) go to 2480
c  yrnoc=0 here if only source areas, no faults included
      if(yrnoc.eq.0.) go to 2480
      line=1
  640 write(16,620)yrnoc,iprint,dumid
  650 if(line.eq.1) go to 750
      nbr=0
      ist=0
      iend=0
c  read identifiers for source area inputs
  660 read(15,*)jseg,ifr,itot
      write(16,670) ifr,itot
  670 format(' source 'i2' of 'i2)
c     jseg=number of pairs of quadrilateral end points in this
c          single source
c     itot=number of sources in this source area
c     ifr=identifies which source,  ifr=1,2,....itot
      if (itot.le.10) go to 690
      write(16,680) itot,jseg,ifr
  680 format('  itot ='i4'  to large--max 10  jseg='i4
     & ' ifr='i4)
      call exit
  690 ist=iend+1
      iend=ist+jseg-1
      if (iend.le.50) go to 710
      write(16,700)
  700 format('  too many quadrilaterals in source--max 50')
      call exit
c  read in boundaries for quadrilaterals for all subareas
c  in this source
c       read limits of seismic area x(i)=lambda(i),   y(i)=phi(i)
c       upper left(x,y) then  upper right(x,y) --1st quadrilateral
c       2nd card   lower left,lower right 1st quadrilateral=
c       2nd card   upper left,upper right   2nd quadrilateral,  etc.
c
  710 do 730 ii=ist,iend
      read(15,*)  (xsav(i,ii),ysav(i,ii), i=1,2)
  720 format(1h 4f10.3)
      write(16,720)(xsav(i,ii),ysav(i,ii),i=1,2)
  730 continue
      if(ifr.eq.itot) go to 740
      nbr=nbr+1
      ibr(nbr)=iend
      go to 660
  740 num=iend
      go to 870
c     fault line read
```

```
  750 j=1
      dsum=0.
c     fault inputs
  760 read(15,*) jseg ,ifr,itot
      write(16,770) ifr,itot
  770 format(' fault 'i2' of 'i2)
c     jseg=number of end points of connected segments
c          for the current fault
c          jseg=one plus number of segments
c     itot=number of di-tinct faults in this zone
c     ifr identifies current fault, ifr=1,2,...itot
      if (jseg.le.24) go to 800
  780 write (16,790) jseg,itot
  790 format('  jseg='i4' (max 24) itot='i4' (max 26)-stop')
  800 if(itot.gt.26) go to 780
c     read in end points long,lat of each segment
      read(15,*)(xl(i,j),yp(i,j),i=1,jseg)
      write(16,810)(xl(i,j),yp(i,j),i=1,jseg)
  810 format(1h 4(f10.2,f8.2,',',))
      jm(j)=jseg
      do 830 i=1,jseg
c     convert to decimal degrees unless dsw=0
      if(dsw.eq.0) go to 820
      call condec(yp(i,j))
      call condec(xl(i,j))
  820 continue
c    transform to equator--rectangular coordinates
      yin=yp(i,j)*rad
      xin=xl(i,j)*rad
      call to eqr(xin,yin,xout,yout)
      xl(i,j)=xout*r
      yp(i,j)=yout*r
  830 continue
c    compute line parameters, lengths for fault segments
      jsegm=jseg-1
      dtot(j)=0.
      do 840 i=1,jsegm
      xdelta=xl(i+1,j)-xl(i,j)
      ydelta=yp(i+1,j)-yp(i,j)
      disq(i,j)=xdelta*xdelta+ydelta*ydelta
      dist(i,j)=sqrt(disq(i,j))
      coa(i,j)=(xl(i+1,j)-xl(i,j))/dist(i,j)
      sia(i,j)=(yp(i+1,j)-yp(i,j))/dist(i,j)
  840 dtot(j)=dtot(j)+dist(i,j)
      dsum=dsum+dtot(j)
      if(ifr.eq.itot) go to 860
  850 j=j+1
      go to 760
c     read in num of occurrences for each magnitude (12f6.2)
c     both area and fault input
  860 jtot=j
  870 read (15,990) (noc(i),i=1,12)
      lev=12
  880 if(noc(lev).ne.0.0) go to 900
      lev=lev-1
      if(lev.gt.0) go to 880
```

```
      write(16,890)
  890 format(' apparent input err for no of occurrences at each level')
      go to 2520
  900 continue
      write(16,910)lev
  910 format(' nr of levels of seismicity = 'i2)
c
c  read in corresponding mag. interval center points. (lev of them)
c
  920 format(' before normalizing to rate/year')
  930 format( ' earthquake rate / year')
      read(15,990)(fm(i),i=1,lev )
      if(yrnoc. eq. 1.) go to 950
      write(16,920)
      write(16,960) (noc(11),11=1,lev)
c     this is a normalization to rate per year
      do 940 l=1,lev
  940 noc(l)=noc(l)/yrnoc
  950 write(16,930)
      write(16,960)(noc(l),l=1,lev)
  960 format(' occurrences='8f10.6)
      write(16,970)(fm(l),l=1,lev)
  970 format('  magnitudes='8f10.2)
      do 980 i=1,lev
      fml(i)=.000630*exp(1.52*fm(i))
      fmag(i)=fm(i)
  980 fm(i)=alog(fm(i))
  990 format(12f6.2)
c
      if (line.eq.0) go to 1110
c    determine which rupture length magnitude relationship to use
      if(als.eq.0)   go to 1080
      al=als
      bl=bls
      sigl=sigls
c     compute break length
 1000 write(16,1010) al,bl,sigl
 1010 format(' fault rupture length parameters al='f7.3' bl='f7.3
     &'  sigl='f6.2)
      linset=1
      if (sigl.gt.0) go to 1070
c     which single break default do we want
c     if bl=0, al non zero use old Algermissen-Perkins default values
      if(bl.eq.0) go to 1040
c     use parameter values supplied for one break only
 1020 do 1030 i=1,lev
 1030 rlls(1,i)=10.**(al+bl*fmag(i))
      idef=2
      go to 1060
c     Algermissen-Perkins old default values
 1040 do 1050 i=1,lev
 1050 rlls(1,i)=fml(i)
      idef=1
 1060 prls(1)=1.
      nrls=1
      go to 1220
```

```
c     set up rupture lengths and probabilities as fct of magnitude
c     nrls lengths per magnitude
c     al, bl, sigl non zero
 1070 call pbreak(fmag,al,bl,sigl,rlls,prls,lev,nrls)
      go to 1220
 1080 if (nrls.eq.1) go to 1090
      if(linset.eq.1) go to 1070
c     compute default values
      al=-1.085
      bl=.389
      sigl=.52
      linset=1
      go to 1000
 1090 if(idef.eq.1) go to 1040
      if(idef.eq.2) go to 1020
      write(16,1100) idef
 1100 format(' fault break length error--idef='i3' quit')
      call exit
c  convert quad corner points to decimal degrees  (if dsw =1)
c
 1110 do 1130 ii=1,num
      do 1130 i=1,2
      if(dsw.eq.0) go to 1120
      call condec(xsav(i,ii))
      call condec(ysav(i,ii))
 1120 continue
      xin=xsav(i,ii)*rad
      yin=ysav(i,ii)*rad
      call to eqr(xin,yin,xout,yout)
      xsav(i,ii)=xout*r
      ysav(i,ii)=yout*r
 1130 continue
c
 1140 nm=num-1
      stot=0.
      do 1180 ii=1,nm
      if(nbr.eq.0) go to 1160
      do 1150 iq=1,nbr
      if(ii.eq.ibr(iq)) go to 1180
 1150 continue
c     set line parameters for subsource
 1160 call setreg(ii)
      do 1170 i=1,2
      x(i)=xsav(i,ii)
      x(i+2)=xsav(i,ii+1)
      y(i)=ysav(i,ii)
 1170 y(i+2)=ysav(i,ii+1)
c     find area of subregion ii
      s1=abs(x(1)*(y(2)-y(3))+x(2)*(y(3)-y(1))+x(3)*(y(1)-y(2)))
      s2=abs(x(4)*(y(2)-y(3))+x(2)*(y(3)-y(4))+x(3)*(y(4)-y(2)))
      sarea(ii)= (s1+s2)/2.
      stot=stot+sarea(ii)
 1180 continue
      eua=noc(1)
c    find earthquake rate per unit area for lowest magnitude
c    rate will be printed but not used in program
```

```
      if(noc(lev).gt.eua) go to 1190
      delm=(fmag(2)-fmag(1))/2.
      fm1=fmag(1)+delm
      fm2=fmag(1)-delm
      go to 1200
 1190 eua=noc(lev)
      delm=(fmag(lev)-fmag(lev-1))/2.
      fm1=fmag(lev)+delm
      fm2=fmag(lev)-delm
 1200 eua=eua/stot
      write(16,1210) dumid,stot,eua,fm1,fm2
 1210 format(1h a4' area='f9.0' sq km, rate/sq km='e12.5,
     & ' for mags 'f5.2'-'f5.2)
 1220 dlev=fm(1)-fm(2)
c     create table of log accelerations for set of magnitudes
c     in (tm) for the original set of mdis distances as a function
c     of distance
c     next three cards test whether table has already
c      been computed--if so skip over
      if(lev.gt.levno) go to 1230
      if(fm(1).ne.flevst) go to 1230
      if(dlev.eq.dlevsv) go to 1390
 1230 do 1240 ill=1,jent
 1240 tmdif(ill)=tm(ill)-tm(ill-1)
      do 1290 ll=1,lev
      do 1250 il=2,jent
      if(fm(ll).gt.tm(il)) go to 1260
 1250 continue
      il=jent
 1260 ilm=il-1
      do 1270 j=1,mdis
 1270 ata(j,ll)=atab(j,ilm)+(fm(ll)-tm(ilm))/tmdif(il)
     & *(atab(j,il)-atab(j,ilm))
      do 1280 j=2,mdis
 1280 atadif(j,ll)=ata(j,ll)-ata(j-1,ll)
 1290 lsub(ll)=il
c      find distance corresponding to each acceleration in
c      table ac (set up in subroutine box) for each magnitude
      do 1380 l=1,lev
      do 1370 j=1,maxp
      do 1300 i=2,mdis
      if(aclog(j).gt.ata(i,l)) go to 1320
 1300 continue
c      beyond table
 1310 dtab(j,l)=rtab(mdis)
      go to 1370
 1320 if(atadif(i,l).ne.0) go to 1360
c      no change in acceleration at this distance
 1330 if(i.eq.2) go to 1350
      write(16,1340) i,l,j
 1340 format(' apparent error indist-table i,l,j='3i4)
      call exit
 1350 dtab(j,l)=0.
      go to 1370
 1360 dtab(j,l)=rtab(i)-(rtab(i)-rtab(i-1))*(ata(i,l)-aclog(j))
     &/atadif(i,l)
```

```
      if(dtab(j,1).lt.0) dtab(j,1)=0.
 1370 continue
 1380 continue
      dlevsv=dlev
      levno=lev
      flevst=fm(1)
 1390 continue
      dtmax=dtab(1,1)
      if(dlev.lt.0) dtmax=dtab(1,lev)
c     find entry for each magnitude in distance-acceleration table
c     for zero distanc  (highest acceleration for that magnitude)
      do 1410 l=1,lev
      j=maxp
 1400 if(dtab(j,1).ne.0.) go to 1410
      j=j-1
      go to 1400
 1410 maxa(l)=j
c
       irow1s=irow1
      irow2s=irow2
      icol1s=icol1
      icol2s=icol2
      if(icol2s.gt.icol) icol2s=icol
      do 2460 ispt=1,2
      if(ispt.eq.1) go to 1420
c    case of single points
      if(indvpt.eq.0) go to 2460
      irow1s=1
      irow2s=1
      icol1s=1
      icol2s=indvpt
      go to 1430
 1420 if(irow1.eq.0) go to 2460
 1430 do 2460 ira=irow1s,irow2s
c
      ymid=(ph1-(ira-.5)*phinc)*r
      do 2450 ii=icol1s,icol2s
c
      if(ispt.eq.2) go to 1440
      fii=ii
      xmid=(fl1-(fii-.5)*flinc)*r
      go to 1450
 1440 xmid=xpt2(ii)
      ymid=ypt2(ii)
 1450 if(isw.eq.0) go to 1460
c
c     after first time thru read values from unit lu=3 or 4 as
c     appropriate or if using data from previous run, read tape
c     (on unit 3) first time thru.
c
      read(11) reg
      go to 1480
c
c  first time thru compute limits if not read from tape
c
 1460 do 1470 ll=1,maxp
```

```
 1470 reg(11)=0.
      reg(maxp+1)=9999.999
c     set reg(maxp+1) to min dist from fault
c
c     for fixed  affected area in row ira, col ii
c
c         source area computation
 1480 if(line.eq.1) go to 1870
      do 1860 k=1,nm
      if(nbr.eq.0) go to 1500
      do 1490 iq=1,nbr
      if(k.eq.ibr(iq)) go to 1860
 1490 continue
 1500 approx=0.
      do 1510 i=1,2
      xll(i)=xsav(i,k)
      xll(i+2)=xsav(i,k+1)
      yl(i)=ysav(i,k)
      yl(i+2)=ysav(i,k+1)
 1510 continue
      xr(3)=xll(1)
      yr(3)=yl(1)
      xr(1)=xll(2)
      yr(1)=yl(2)
      xr(2)=xll(4)
      yr(2)=yl(4)
      xr(4)=xll(3)
      yr(4)=yl(3)
      aread=sarea(k)/stot
c     determine whether site is inside or outside k th quadrilateral
c     subregion.  find distance rc to closest endpoint
c     and rf to farthest end point.  Set up table of distance versus
c     arc length for distances in the range rc to rf using
c     subroutines outsid or inside.
c      itst=no of radii at fixed increments between rc and rf
c             at which arc lengths are to be evaluated in table
c      itst=7   set in subroutine rrisk--recompile to change
      call rrisk(xmid,ymid,in,k,dis,rc,rf,itst)
      if(in.eq.1) go to 1540
      if(rc.lt.dtmax) go to 1530
      do 1520 ll=1,lev
 1520 reg(1)=reg(1)+aread*noc(ll)
      go to 1860
 1530 rcs=rc
      disl(1)=rc
      angl(1)=0.
      itst=itst+1
      ibeg=2
      go to 1550
 1540 disl(1)=0.
      angl(1)=0
      disl(2)=rc
      angl(2)=pi2*rc
      ibeg=3
      if(rc.eq.0) ibeg=2
      rcs=0.
```

```
      itst=itst+2
c    compute distance versus arc length table for even increments
c    in distance
 1550 do 1580 iii=ibeg,itst
      if(in.eq.0.) go to 1560
      call inside(xmid,ymid,k,dis,angle)
      go to 1570
 1560 call outsid(xmid,ymid,k,dis,angle)
 1570 continue
      disl(iii)=dis
      angl(iii)=angle*dis
 1580 dis=dis+stsize
      itst=itst+1
      disl(itst)=rf
      angl(itst)=0.
c    compute distance versus arc length table for distances
c    to selected vertices and perpendiculars to edges
      rcsq=rc*rc
      rfsq=rf*rf
      do 1690 i=1,4
      ire=0
      if(in.eq.0) ire=2
      if(sllq(i).le.rcsq+1.) go to 1680
      if(sllq(i).ge.rfsq-1.) go to 1680
      dis=sqrt(sllq(i))-.001
      if(dis.le.rc) goto 1680
c     add angle, distance corresponding to vertex
      if (in.eq.0) go to 1600
      call inside(xmid,ymid,k,dis,angle)
      go to 1630
 1590 if(mina(i).ne.3) go to 1690
c    closest side next
      dis=abs(perpa(i))
      if(dis.le.rc+.5) go to 1690
      if(dis.ge.rf-.5) go to 1690
      dis=dis-.001
      call inside(xmid,ymid,k,dis,angle)
      ire=1
      go to 1630
 1600 continue
      call outsid(xmid,ymid,k,dis,angle)
      go to1630
 1610 if(mina(i).ne.3) go to 1690
      dis=abs(perpa(i))
      ire=1
      do 1620 iql=1,4
      if(abs(sllq(iql)-perpaq(i)).le. 2.) go to 1690
 1620 continue
      if(dis.le.rc+1.) go to 1690
      if(dis.ge.rf-1.) go to 1690
      dis=dis-.001
      call outsid(xmid,ymid,k,dis,angle)
 1630 continue
      do 1640 ll=1,itst
      if(dis-disl(ll)) 1650,1670,1640
 1640 continue
```

```
c    insert new distance, angle
 1650 jk=itst
      do 1660 mm=11,itst
      angl(jk+1)=angl(jk)
      disl(jk+1)=disl(jk)
 1660 jk=jk-1
      disl(11)=dis
      angl(11)=angle*dis
      itst=itst+1
 1670 if(ire.eq.0) go to 1590
      if(ire.eq.2) go to 1610
      go to1690
 1680 if(in)1610,1610,1590
 1690 continue
      itst=itst+1
      disl(itst)=rf
      angl(itst)=0.
c    for each magnitude, source, determine fractional source area giving
c    acceleration a:  ac(i-1) < a < ac(i)  for each entry in acceleration
c    table ac.  equivalent to finding fractional source area at distance
c    from site between successive entries in acceleration-distance
c    table:  dtab(i,1) < d < dtab(i-1,1) for magnitude 1.
c    interpolates for distance is disl vs angl table
c    area=integral(angle*dist*delta(dist))  from dtab(i-1,1) to dtab(i,1)
      do 1840 11=1,lev
      stnoc=noc(11)/stot
      iswd=0
      maxm=maxa(11)
      if(maxm.le.0.) go to 1710
      approx=0.
      if(in.eq.0) go to 1700
      jlo=maxm
      go to 1730
 1700 if(rcs.lt.dtab(1,11)) go to 1720
 1710 reg(1)=reg(1)+aread*noc(11)
      go to 1830
 1720 jlo=indpt(dtab(1,11),rcs,maxm)
 1730 j=jlo
      iqs=2
      do 1810 jk=1,jlo
      if(jk.gt.1) go to 1740
      dbot=rcs
      ddis=dtab(j,11)
      if(dtab(j,11).lt.rf) goto 1760
      reg(j+1)=reg(j+1)+sarea(k)*stnoc
      go to 1830
 1740 if(dtab(j,11).lt.rf) go to 1750
      dbot=dtab(j+1,11)
      ddis=rf
      iswd=1
      go to 1760
 1750 ddis=dtab(j,11)
      dbot=dtab(j+1,11)
 1760 anarea=0.
      do 1770 iq=iqs,itst
      if(dbot.le.disl(iq)) go to 1780
```

```
 1770 continue
      iq=itst
c     distance exceeds last value-error since table set up to handle this
 1780 dtop=disl(iq)
      if(disl(iq).gt.ddis) dtop=ddis
      dmid=(dtop+dbot)/2.
      delr=dtop-dbot
      fr=(dmid-disl(iq-1))/(disl(iq)-disl(iq-1))
      ang=angl(iq-1)+(angl(iq)-angl(iq-1))*fr
      anarea=anarea+ang*delr
      if(dtop.eq.ddis) go to 1800
      dbot=dtop
      iq=iq+1
       if(iq.le.itst) go to 1780
      write(16,1790)
 1790 format(' err in arc area computation')
      go to 1780
c     interpolate in angle , distance table
 1800 approx=approx+anarea
      reg(j+1)=reg(j+1)+anarea*stnoc
      j=j-1
      if(iswd.eq.0) go to 1810
      arerr=(approx-sarea(k))/sarea(k)
      allerr=allerr+1.
      if (abs(arerr).lt. .05) go to 1840
      tot5=tot5+1.
      if(abs(arerr).le. .10) go to 1840
      write(16,1850) arerr,ira,ii,rc,rf,k,ll,in,dumid
      go to 1840
 1810 iqs=iq
c      need to add remaining area if any to lowest acceleration
      dif=sarea(k)-approx
      if(dif.lt.0) go to 1830
 1820 reg(1)=reg(1)+dif*stnoc
 1830 approx=sarea(k)
 1840 continue
 1850 format(' int err='f6.3' row 'i2' col 'i2' rc='
     & f7.2' rf='f7.2' k='i2' ll='i2' in='i1,1x,a4)
 1860 continue
      go to 2400
c     fault computation
 1870 do 2390 j=1,jtot
      jseg=jm(j)
      jsegm=jseg-1
      call cldis(xmid,ymid,j)
      do 1910 ll=1,lev
c     fill in rupture lengths--make  certain that rupture length
c     does not exceed total fault length
      nrl=nrls
      do 1880 irl=1,nrl
      rll(irl,ll)=rlls(irl,ll)
 1880 prlrl(irl,ll)=prls(irl)
      if(rll(1,ll).lt.dtot(j)) go to 1910
      rll(1,ll)=dtot(j)
      if(nrl.eq.1) go to 1910
 1890 if(rll(2,ll).lt.dtot(j)) go to 1910
```

```
      prlrl(i,ll)=prlrl(i,ll)+prlrl(2,ll)
      nrl=nrl-1
      if(nrl.eq.1) go to 1910
      do 1900 i=2,nrl
      prlrl(i,ll)=prlrl(i+1,ll)
 1900 rll(i,ll)=rll(i+1,ll)
      go to 1890
 1910 nrlsv(ll)=nrl
      pt1=sqrt(shdis)
      if(pt1.lt.reg(maxp+1)) reg(maxp+1)=pt1
      if (isid.eq.0) go to 2160
c     distance is monotonically increasing for all segments
c        (the decreasing case has been converted to increasing)
c     or single shortest distance is interior to fault
      do 2150 ll=1,lev
      nrl=nrlsv(ll)
      frcon=dtot(j)/dsum*noc(ll)
      maxm=maxa(ll)
      jlo=indpt(dtab(1,ll),pt1,maxm)
      if (jlo.gt.0) go to 1920
c    distance beyond table--lump acceleration in first box
      reg(1)=reg(1)+frcon
      go to 2150
 1920 continue
      if (isid.eq.1) go to 1940
 1930 ibeg=1
      ifin=jsegm
      ixt=1
      dnear=dtot(j)
      inear=jsegm
      go to 1980
 1940 ibeg=1
      ifin=inear
      ixt=2
c    dnear=distance along fault from one end to point on fault
c    closest to the site
c    dnear=total length of segments 1,2,...inear
c    inear defined sub cldis--in common/extra/
      dnear=0.
      do 1950 iq=1,inear
 1950 dnear=dnear+distl(iq)
      dfar=dtot(j)-dnear
      do 1970 irl=1,nrl
      smd=rll(irl,ll)
      if(smd.lt.dtot(j)) go to 1960
c      break is entire segment
      reg(jlo+1)=reg(jlo+1)+frcon*prlrl(irl,ll)
      go to 1970
 1960 smd2=smd/2.
c     find fraction of fault at closest distance to site
c     for interior site
      arcn=smd2
      arcf=smd2
      if(dnear.lt.smd) arcn=dnear-smd2
      if(dfar.lt.smd) arcf=dfar-smd2
      arct=arcn+arcf
```

```
      if(arct.lt. .0001) go to 1970
      reg(jlo+1)=reg(jlo+1)+frcon/(dtot(j)-smd)*arct*prlrl(irl,ll)
 1970 continue
c    find frac of fault in distance range dtab(jr,ll) < d <dtab(jr-1,ll)
c    for each magnitude ll
 1980 do 2140 ithru=1,ixt
      jr=jlo
      iii=ibeg
      casum=0.
      irl=1
      dsav=-dls(iii)
 1990 frac=0.
      smd=rll(irl,ll)
      if(smd.lt.dtot(j)) go to 2000
      if(ixt.eq.2)go to 2120
c     add in entire fault length
      reg(jlo+1)=reg(jlo+1)+frcon*prlrl(irl,ll)
      go to 2120
 2000 cuml=dnear-smd
      if(cuml.le.0) go to 2120
      do 2010 iq=irl,nrl
 2010 frac=frac+prlrl(iq,ll)/(dtot(j)-rll(iq,ll))
      frac=frac*frcon
      dlen=0.
      do 2020 iq=ibeg,ifin
      disrem(iq)=distl(iq)
      dlen=dlen+disrem(iq)
      if (dlen.ge.cuml) go to2040
 2020 continue
      write(16,2030)dlen,ibeg,ifin,cuml
 2030 format(' error for fault line dlen='e12.5' ibeg='i3
     &' ifin='i3' cuml='e12.5)
      call exit
 2040 disrem(iq)=cuml-dlen+disrem(iq)
 2050 ifins=iq
 2060 dend=disrem(iii)-dls(iii)
c     dsav=starting distance along segment iii
 2070 dl3=sqrt(dtab(jr,ll)**2-persq(iii))
      if(dl3.gt.dend) go to 2110
      carea=dl3-dsav
      casum=casum+carea
      reg(jr+1)=reg(jr+1)+carea*frac
      jr=jr-1
      if (jr.eq.0) go to 2090
 2080 dsav=dl3
      go to 2070
c    at top of table--remaining dist beyond table
 2090 do 2100 iq=irl,nrl
      cuml=dnear-rll(iq,ll)
 2100 reg(1)=reg(1)+(cuml-casum)*prlrl(iq,ll)*frcon
     & /(dtot(j)-rll(iq,ll))
      go to 2130
 2110 carea=dend-dsav
      casum=casum+carea
      reg(jr+1)=reg(jr+1)+carea*frac
      dsav=dend
```

```
c       at end of segment iii
        if(iii.eq.ifins)  go to 2120
        iii=iii+1
        dsav=-dls(iii)
        go to 2060
 2120   irl=irl+1
        if(irl.le.nrl) go to 1990
        if(ithru.eq.ixt) go to2150
 2130   irl=1
        ibeg=inear+1
        ifin=jsegm
        dnear=dtot(j)-dnear
 2140   continue
 2150   continue
        go to 2390
c       more than one turning point
c       brute force.  find contribution for single break
c       increment break centers by distance deltal along fault from
c       one end to other.
 2160   continue
        deltal=10.
        if(pt1.gt.100) deltal=20.
        if(pt1.lt.15.) deltal=5.
        do 2380 ll=1,lev
        nrl=nrlsv(ll)
        do 2380 irl=1,nrl
        smd=rll(irl,ll)
        prnoc=noc(ll)*prlrl(irl,ll)
        dj=dtot(j)-smd
        nd=dj/deltal+1.5
        if(nd.gt.1) go to 2170
        nd=2
        if(dj.gt.1.) go to 2170
c       break is entire fault length
        maxm=maxa(ll)
        jlo=indpt(dtab(1,ll),pt1,maxm)
        reg(jlo+1)=reg(jlo+1)+prnoc*dtot(j)/dsum
        go to 2380
 2170   continue
        dmvp=prnoc*dtot(j)/(dsum*(nd-1))
        del=dj/(nd-1)
        do 2370 m=1,nd
        dst=(m-1)*del
        do 2180 i=1,jsegm
        if(dst.lt.dist(i,j)) go to 2200
        dst=dst-dist(i,j)
 2180   continue
        write(16,2190)dst,j
 2190   format(' apparent error fault dist--dst='e12.5'  j='i4)
        go to 2520
 2200   ddsav=1.e8
        dfin=dst+smd
 2210   dover=0.
        if(dfin.le.dist(i,j)) go to 2220
        dover=dfin-dist(i,j)
        dfin=dist(i,j)
```

```
 2220 if(dls(i).lt.dst) go to 2230
      if(dls(i).gt.dfin) go to 2240
c      shortest distance to segment at point within segment
      dd=persq(i)
      go to 2250
c      shortest dist at start of segment
 2230 dd=(dls(i)-dst)**2+persq(i)
      go to 2250
c      shortest distance at end end of segment
 2240 dd=(dls(i)-dfin)**2+persq(i)
 2250 continue
      if(dd.lt.ddsav) ddsav=dd
      if(abs(dover).lt.0.0001) go to 2270
      dst=0
      i=i+1
      dfin=dover
      if(i.le.jsegm) go to 2210
 2260 format(' i too large i='i2)
      write(16,2260) i
      call exit
 2270 dis=sqrt(ddsav)
c  have distance to subsegment of arc of fault
c
      if(m.gt.1) go to 2280
      jen2=indpt(dtab(1,11),dis,maxa(11))
      disav=dis
      go to 2370
 2280 disdif=dis-disav
      if(disdif.eq.0) go to 2290
      dfac=dmvp/disdif
c     spread out contributions into accel boxes
      if(disdif) 2300,2290,2330
 2290 reg(jen2+1)=reg(jen2+1)+dmvp
      go to 2370
c      distance decreasing
 2300 jen2=jen2+1
 2310 distab=dis-dtab(jen2,11)
      if (distab.lt.0) go to 2320
      reg(jen2)=reg(jen2)+(dis-disav)*dfac
      jen2=jen2-1
      disav=dis
      go to 2370
 2320 reg(jen2)=reg(jen2)+(dtab(jen2,11)-disav)*dfac
      disav=dtab(jen2,11)
      jen2=jen2+1
      go to 2310
 2330 if(jen2.eq.0) go to 2350
 2340 distab=dis-dtab(jen2,11)
      if(distab.gt.0) goto 2360
 2350 reg(jen2+1)=reg(jen2+1)+(dis-disav)*dfac
      disav=dis
      go to 2370
 2360 reg(jen2+1)=reg(jen2+1)+(dtab(jen2,11)-disav)*dfac
      disav=dtab(jen2,11)
      jen2=jen2-1
      if (jen2.ge.1) go to 2340
```

```
      go to 2350
 2370 continue
 2380 continue
c     above ends loop for more than one turning point
 2390 continue
 2400 continue
      write(12) reg
c     write accel box counts on tape after looping thru all
c     source subareas
c
      if(iprint.ge.0) go to 2410
      go to 2450
 2410 continue
      if(iprint.le.2) write(16,2420)tytle
      if (ispt.eq.1) go to 2430
      xout1=xpt1(ii)
      yout1=ypt1(ii)
      go to 2440
 2420  format('1',a80)
c     back transform point from equator to dec degrees
 2430 call backw(ii,ira,xout1,yout1)
c     results calculated and printed in subroutine out
 2440 call out(reg,naf,xout1,yout1,sd,nterp)
 2450 continue
c     ends loop for one site, one area or fault
c     repeat for all sites for this source before going on to next
c     source
 2460 continue
c     one source completed for all sites
       rewind 4
       rewind 3
 247  continue
      l1s=l1
      l1=l2
      l2=l1s
      isw=1
      go to 610
c     finished
 2480 if(l2.eq.4) go to 2520
c     transfer to unit 3 for possible continuation or restart
 2490 read(4,end=2510) reg
 2500 write(3) reg
      go to 2490
 2510  continue
 2520 continue
c     write(16,2530) tot5,allerr
 2530 format(/' sources with error in area > .05='f7.0
     & ' out of 'f7.0' sources ')
 2540 format(' number of brute force points='i4)
      stop
      end
c ***************************************************************
      subroutine condec(phi)
      i=phi
c     fr=phi-i
      phi=float(i)+(phi-float(i))/.60
```

```
      return
      end
c ***************************************************************
      subroutine lin(x1,y1,x2,y2,a,b,c,x3,y3)
c   coefficients a, b, c are determined so that points which are in the
c   same half plane as x3,y3 will be a positive distance from the line
c   joining x1,y1 and x2,y2
c       a x + b y + c=0
      if(x1.eq.x2) go to 10
      a=(y1-y2)/(x1-x2)
      b=-1
      c=y1-a*x1
      go to 20
   10 a=1.
      b=0
      c=-x1
   20 d=a*x3+b*y3+c
      if(d.gt.0.) return
      a=-a
      b=-b
      c=-c
      return
      end
c ***************************************************************
      subroutine linint(reg,t,sol)
      common/tims/dummy(42),iprint
      common/quad/maxp,ac(105),aclog(105)
      common/ext/prob,ex,ex1
      dimension reg(106)
      ii=maxp-1
      sum=0.
   10 if(ii.le.1) go to 20
      if(reg(ii).gt.0.) go to 40
      ii=ii-1
      go to 10
   20 sol=0.
      if(iprint.le.2) write(16,30)
   30 format(' sol not obtained for time='f7.0)
      return
   40 sum=sum+reg(ii)
      if(t*sum.gt.ex) go to 50
      ii=ii-1
      if(ii.ge.1) go to 40
      go to 20
   50 if(sum.eq.reg(ii)) go to 70
      y2=alog(t*(sum-reg(ii)))
      y1=alog(t*sum)
      ex1t=ex1
   60 a=(y1-y2)/(ac(ii-1)-ac(ii))
      b=y1-a*ac(ii-1)
      sol=(ex1t-b)/a
      return
   70 y1=t*sum
      y2=0.
      ex1t=ex
      if(iprint.le.2) write(16,80)
```

```
   80 format(' log interpolation fails--use linear')
      go to 60
      end
c ***************************************************************
      subroutine wts(nwt,sd)
c    set up nwt weights for later redistributing (log) mean peak
c    accelerations normally around mean values
c    standard deviation in log acceleration = sd
      common/wty/wt(101),dev(101)
      dimension temp( 0)
      data(temp(i),i=1,10)/0.,.132,.267,.407,.555,.717,.900,1.119,
     & 1.413,1.938/
      w=1./19.
      np=10
      do 10 i=1,10
      dev(i+9)=temp(i)
      dev(np)=-temp(i)
   10 np=np-1
      nwt=19
      do 20 i=1,nwt
      wt(i)=w
   20 dev(i)=sd*dev(i)
      return
      end
c ***************************************************************
      subroutine box(scale,sd)
c     determine boundaries ac(i) for acceleration levels
c     accelerations in the rangeac(i-1) < a <ac(i) will be
c     accumulated in reg(i)
c     scale multiplies the basic levels computed
c     scale=1 for accelerations in the range .01 to 5.
      common /quad/maxq,ac(105),aclog(105)
      common/wds/max,maxp
      max=104
      maxp=105
      yul=.513
      xul=1.135
      ac(1)=.01*scale
      do 10 i=2,55
      ac(i)=ac(i-1)*xul
      yul=yul*.95
      xul=1.02+.28*yul
   10 continue
      a1=ac(55)
      a0=a1*a1
      do 20 i=56,100
      a1=a1/xul
      ac(i)=a0/a1
      yul=yul/.95
      xul=1.02+.28*yul
   20 continue
c ----------------note important-------------
c     the following inserted to allow acceleration boxes for small
c     values below range of interest for sd computation
c     this should be changed later as desired
      j=100
```

```
      do 30 i=1,100
      ac(j+4)=ac(j)
   30 j=j-1
      j=4
      efac=exp(-sd/2.)
      do 40 i=1,4
      ac(j)=ac(j+1)*efac
   40 j=j-1
      return
      end
c ***************************************************************
      subroutine inieqr(x1,y1,x2,y2)
c     read xlong, ylat in radians for 1st point
c     read xlong, ylat in radians for 2nd point
c     points 1 and 2 to be transformed to equator
c     point 1 will be transformed to (0,0)
c     point 2 will be transformed to (d,0)
c    where d=great circle distance between (x1,y1) and (x2,y2)
c     output is transformation matrices f (forward) fi (backward)
c       generated and stored in comm.)
      common/eqr/ r1(3,3),r2(3,3),r3(3,3),d(3,3),f(3,3),vec(3)
      common/inv/fi(3,3),res(3)
      cp1=cos(y1)
      cp2=cos(y2)
      cl1=cos(x1)
      sp1=sin(y1)
      sp2=sin(y2)
      sl1=sin(x1)
c  initialize matrices
      do 10 i=1,3
      do 10 j=1,3
      r1(i,j)=0.
      r2(i,j)=0.
   10 r3(i,j)=0.
      r1(1,1)=1.
      r2(2,2)=1.
      r3(3,3)=1.
      r2(1,1)=cp1
      r2(3,3)=cp1
      r2(1,3)=sp1
      r2(3,1)=-sp1
      r3(1,1)=cl1
      r3(2,2)=cl1
      r3(1,2)=sl1
      r3(2,1)=-sl1
      s1=cp1*sp2-sp1*cp2*cos(x2-x1)
      c1=cp2*sin(x2-x1)
      den=sqrt(c1*c1+s1*s1)
      sinth=s1/den
      costh=c1/den
      r1(2,2)=costh
      r1(3,3)=costh
      r1(2,3)=sinth
      r1(3,2)=-sinth
      call matmpy(r1,r2,d)
      call matmpy(d,r3,f)
```

```
c    transformation matrix in f! inverse matrix in fi
c    compute inverse
       do 20 i=1,3
       do 20 j=1,3
       r1(i,j)=0.
       r2(i,j)=0.
       r3(i,j)=0.
   20  continue
       r1(3,3)=1.
       r2(2,2)=1.
       r3(1,1)=1.
       r1(1,1)=cl1
       r1(2,2)=cl1
       r1(2,1)=sl1
       r1(1,2)=-sl1
       r2(1,1)=cp1
       r2(3,3)=cp1
       r2(1,3)=-sp1
       r2(3,1)=sp1
       r3(2,2)=costh
       r3(3,3)=costh
       r3(3,2)=sinth
       r3(2,3)=-sinth
       call matmpy(r1,r2,d)
       call matmpy(d,r3,fi)
       return
       end
c ***************************************************************************
       subroutine toeqr(x3,y3,flam6,phi6)
c     transform point at (x3,y3)  (long,lat in radians) to new
c     equator using transformation defined in subroutine ineqr
       common/eqr/  r1(3,3),r2(3,3),r3(3,3),d(3,3),f(3,3),vec(3)
       common/inv/fi(3,3),res(3)
       cxmid=cos(x3)
       cymid=cos(y3)
       sxmid=sin(x3)
       symid=sin(y3)
       vec(1)=cxmid*cymid
       vec(2)=sxmid*cymid
       vec(3)=symid
       call vecmpy(f,vec,res)
       phi6=asin(res(3))
       cp6=cos(phi6)
       cl6=res(1)/cp6
       sl6=res(2)/cp6
       if(cl6.gt.1.) cl6=1.
       flam6=acos(cl6)
       if(sl6.lt.0.) flam6=-flam6
       return
       end
c ***************************************************************************
       subroutine backw(ix,iy,flii,phii)
c      for printout and identification transformed point defined
c      by  row ix and col iy in new coordinate system back to
c      location (long, lat. in dec degrees ) where it would have
c      been initailly
```

```
      common/inv/fi(3,3),res(3)
      dimension vec(3)
c     need cos, sins in common array
      common/sincos/sinx(100),cosx(100),siny(100),cosy(100)
      data rad/.0174533/
      sl6=sinx(ix)
      cl6=cosx(ix)
      cp6=cosy(iy)
      sp6=siny(iy)
      vec(1)=cl6*cp6
      vec(2)=sl6*cp6
      vec(3)=sp6
      call vecmpy(fi,vec,res)
      if(abs(res(3)).lt.1.0) go to 30
      if(res(3).ge.1.) go to 10
      phii=-90.
      go to 20
   10 phii=90.
   20 flam=0.
      return
   30 phii=asin(res(3))
      cpi=cos(phii)
      phii=phii/rad
      cli=res(1)/cpi
      sli=res(2)/cpi
      flii=acos(cli)
      if(sli.lt.0.) flii=-flii
      flii=flii/rad
      return
      end
c ***************************************************************************
      subroutine matmpy(a,b,c)
c     multiply two 3 x 3 matrices A x B = C
      dimension a(3,3),b(3,3),c(3,3)
      do 10 i=1,3
      do 10 j=1,3
      c(i,j)=0.
      do 10 k=1,3
   10 c(i,j)=c(i,j)+a(i,k)*b(k,j)
      return
      end
c ***************************************************************************
      subroutine vecmpy(f,vec,r)
c     vector multiplication used by subroutines backw, toeqr
      dimension f(3,3),vec(3),r(3)
      do 10 i=1,3
      r(i)=0.
      do 10 k=1,3
   10 r(i)=r(i)+f(i,k)*vec(k)
      return
      end
c ***************************************************************************
      subroutine csqdis(x,y,xl,yl,xr,yr,isub)
      dimension xl(4),yl(4),xr(4),yr(4)
      common/slinp/fsa(4,50),fsb(4,50),fsc(4,50),rta(4,50),dist(4,50)
     & ,disq(4,50)
```

```
      common /slinds/dls(4),perp(4),sl1(4),slsq(4),min(4),persq(4)
     &,srsq(4),dlp(4)
      do 10 i=1,4
   10 perp(i)=(fsa(i,isub)*x+fsb(i,isub)*y+fsc(i,isub))/rta(i,isub)
      do 20 i=1,4
      slsq(i)=(x-xl(i))**2+(y-yl(i))**2
      srsq(i)=(x-xr(i))**2+(y-yr(i))**2
   20 continue
      do 50 i=1,4
      persq(i)=perp(i)*perp(i)
      dl1=slsq(i)-persq(i)
      dl2=srsq(i)-persq(i)
      if(dl1.lt.0.) dl1=0.
      if(dl2.lt.0.) dl2=0.
      if(dl1+dl2.le.disq(i,isub)) go to 40
c      shortest distance external to segment
      if(dl1.lt.dl2) go to 30
      min(i)=1
      dlp(i)=sqrt(dl2)
      dls(i)=dist(i,isub)+dlp(i)
      go to 50
   30 min(i)=2
      dls(i)=-sqrt(dl1)
      go to 50
   40 min(i)=3
      dls(i)=sqrt(dl1)
      dlp(i)=0.
   50 continue
      return
      end
c ***************************************************************************
      subroutine rrisk(xnot,ynot,in,isub,r,rc,rf,itst)
       common/inout/ic,ics,ifar,azimf,stsize
       common/xyarea/xsav(2,50),ysav(2,50),sarea(50)
       common/slinds/dls(4),perp(4),sl1(4),slsq(4),min(4),persq(4)
     &,srsq(4),dlp(4)
       common/slinp/aa(4,50),bb(4,50),cc(4,50),rta(4,50),dist(4,50),
     & disq(4,50)
       common/xlyl/xl(4),yl(4),xr(4),yr(4),xc(4),yc(4)
c          subroutine which loads temporary arrays with this
c            subsource's corners and determines if this site is
c            within or without the subsource.
      data pi2/6.2831853/
c    fixed number of radii at which distance-vs arc length
c    table is computed is set here   itst=7 ***************
      itst=7
      rc2=10000000.
      ics=0
      call csqdis(xnot,ynot,xl,yl,xr,yr,isub)
      rf2=0.
      do 20 ii=1,4
      if(slsq(ii).gt.rc2) go to 10
      rc2=slsq(ii)
   10 if(slsq(ii).lt.rf2) go to 20
      rf2=slsq(ii)
      ifar=ii
```

```
   20 continue
      do 30 ii=1,4
      if(min(ii).ne.3) go to  30
      if(persq(ii).gt.rc2) go to 30
      rc2=persq(ii)
      ics=ii
   30 continue
      rc=sqrt(rc2)
      rf=sqrt(rf2)
      stsize=(rf-rc)/itst
c        step thru source area.
      r=rc+(0.5)*stsize
c        determine if site is inside source area.
   40 do 50 ii=1,4
      d=aa(ii,isub)*xnot+bb(ii,isub)*ynot+cc(ii,isub)
      if(d.lt.0)go to 90
   50 continue
      in=1
c           determine azimuth of farthest point with respect to site
      azimf=acos((x1(ifar)-xnot)/rf)
      if (y1(ifar)-ynot) 60,70,70
   60 azimf=pi2 - azimf
   70 continue
c       rc is now distance from site to closest side.
c       rf is now distance from site to farthest corner.
c         pick step size based on fraction of area left
   80 return
c        if do loop finished, point lies within area.
   90 in=0
  100 return
      end
c ***************************************************************
      subroutine setreg(isub)
c    find equation of line thru each side of quadrilateral
c   find length of each side, etc.
      common/outs/xouts,youts
      dimension x1(4),y1(4),xr(4),yr(4)
      common/xyarea/xsav(2,50),ysav(2,50),sarea(50)
      common/slinp/aa(4,50),bb(4,50),cc(4,50),rta(4,50),dist(4,50),
     & disq(4,50)
      data r/6378./
      do 10 i=1,2
      x1(i)=xsav(i,isub)
      x1(i+2)=xsav(i,isub+1)
      y1(i)=ysav(i,isub)
      y1(i+2)=ysav(i,isub+1)
   10 continue
      xr(3)=x1(1)
      yr(3)=y1(1)
      xr(1)=x1(2)
      yr(1)=y1(2)
      xr(2)=x1(4)
      yr(2)=y1(4)
      xr(4)=x1(3)
      yr(4)=y1(3)
c        determine if any sides are vertical lines
```

```
      do 70 ii=1,4
      if(xl(ii).eq.xr(ii)) go to 20
      aa(ii,isub)=(yl(ii)-yr(ii))/(xl(ii)-xr(ii))
      bb(ii,isub)=-1.
      cc(ii,isub)=yl(ii)-aa(ii,isub)*xl(ii)
      go to 30
   20 aa(ii,isub)=1.
      bb(ii,isub)=0.
      cc(ii,isub)=-xl(ii)
   30 if(ii.gt.2) go to 40
      d=aa(ii,isub)*xl(3)+bb(ii,isub)*yl(3)+cc(ii,isub)
      go to 50
   40 d=aa(ii,isub)*xl(2)+bb(ii,isub)*yl(2)+cc(ii,isub)
   50 if (d.gt.0.) go to 60
      aa(ii,isub)=-aa(ii,isub)
      bb(ii,isub)=-bb(ii,isub)
      cc(ii,isub)=-cc(ii,isub)
   60 disq(ii,isub)=(xr(ii)-xl(ii))**2+(yr(ii)-yl(ii))**2
      dist(ii,isub)=sqrt(disq(ii,isub))
   70 rta(ii,isub)=sqrt(aa(ii,isub)*aa(ii,isub)+bb(ii,isub)*
     &bb(ii,isub))
      return
      end
      function indpt(y,yval,n)
c      points in descending order--non zero values from points
c      1 to n:  find first value in table exceeding yval
      dimension y(102)
      if(yval.lt.y(1)) go to 10
      indpt=0
      return
   10 if(yval.gt.y(n)) go to 20
      indpt=n
      return
   20 num=n
      idiv=2
   30 nh=num/idiv
      if(nh.le.1) go to 50
      if(yval.lt.y(nh)) go to 40
      idiv=2*idiv
      go to 30
c    yval lies approx between nh and 2*nh or 2*nh
   40 idiv=2*idiv
      nh2=nh+num/idiv
      if(nh.eq.nh2) goto 80
      if(yval.gt.y(nh2)) go to 40
c    yval further down on table than y(nh2)
      nh=nh2
      go to 40
   50 do 60 i=1,n
      if(y(i).lt.yval) go to 70
   60 continue
   70 indpt=i-1
      return
   80 continue
      do 90 i=nh,n
      if(y(i).lt.yval) go to70
```

```
   90 continue
      end
c ***************************************************************
      subroutine cldis(x,y,jl)
c     subroutine to compute shortest distance, etc to line
c     containing fault segment i of jl th fault (for each i)
c     determines whether closest point to fault is at
c     low end (min(i)=2)
c     interior  (min(i)=3)
c     high end  (min(i)=1)
c     set isid=0 if multiple turning points
c         isid=1 if single closest distance from site to fault
c         interior to fault
c         isid=2 if closest distance from site to fault at one end
c         of the fault
      dimension dlp(20)
      common/linps/distl(24,26),disq(24,26),jseg,jsegm,
     & xl(24,26),yp(24,26),coa(24,26),sia(24,26)
      common/extra/isid,shdis,dist(24),inear
      common /lindis/dls(24),perp(24),min(24),persq(24)
      shdis=1.e9
      do 10 ii=1,jsegm
   10 dist(ii)=distl(ii,jl)
      i1=0
      i2=0
      i3=0
      isid=0
      do 20 i=1,jseg
      slsq=(x-xl(i,jl))**2+(y-yp(i,jl))**2
      if(slsq.lt.shdis) shdis=slsq
   20 continue
      do 50 i=1,jsegm
      xd=x-xl(i,jl)
      yd=y-yp(i,jl)
      perp(i)=-xd*sia(i,jl)+yd*coa(i,jl)
      dls(i)=xd*coa(i,jl)+yd*sia(i,jl)
      persq(i)=perp(i)*perp(i)
      if(dls(i).lt.0) go to 30
      if(dls(i).le.dist(i)) go to 40
c      shortest distance external to segment
      min(i)=1
      i1=i1+1
      dlp(i)=dls(i)-dist(i)
      go to 50
   30 min(i)=2
      i2=i2+1
      go to 50
   40 min(i)=3
      i3=i3+1
      if(persq(i).lt.shdis) shdis=persq(i)
      dlp(i)=0.
   50 continue
      if(i1.ne.jsegm)  go to 90
      iup=jsegm
      isid=2
   60 iuh=iup/2
```

```
      if(2*iuh.ne.iup) dls(iuh+1)=-dlp(iuh+1)
      if(iup.eq.1)  go to 80
      j=iup
      do 70 i=1,iuh
      t=persq(i)
      persq(i)=persq(j)
      persq(j)=t
      dls(i)=-dlp(j)
      dls(j)=-dlp(i)
      t=dist(i)
      dist(i)=dist(j)
      min(i)=2
      min(j)=2
      dist(j)=t
   70 j=j-1
   80 return
   90 if(i2.ne.jsegm)  go to 100
      isid=2
      return
  100 if(i3.gt.1) return
      if(min(1).eq.2) return
      iturn=0
      iq=1
      do 110 k=1,jsegm
      if(min(k).ne.1) go to 120
      iturn=iturn+1
  110 continue
  120 if(iturn.ne.i1) return
      if(i3.eq.1 .and. min(k).eq.2) return
c    we have single turning point
      isid=1
      inear=k-1
      if(min(k).ne.3) go to 140
      inear=k
c       adjust to 2 sep segments
      j=jseg
      do 130 i=k,jsegm
      dls(j)=dls(j-1)
      persq(j)=persq(j-1)
      dist(j)=dist(j-1)
  130 j=j-1
      jsegm=jseg
      jseg=jseg+1
      dist(k+1)=dist(k+1)-dls(k)
      dist(k)=dls(k)
      dls(k+1)=0
  140 iup=inear
      go to 60
      end
c ***************************************************************************
      subroutine pbreak(fm,al,bl,sigl,rlls,prls,lev,nrls)
      dimension fm(12),prls(12),rlls(10,12)
      dimension pmid(5),prang(5)
      data pmid/1.575,.74,0.,-.74,-1.575/
      data prang/.1151,.2295,.3108,.2295,.1151/
      data nrl/5/
```

```
c       evaluate al+bl*fm(l)+pmid(i)*sigl
c      where  1 <l <lev      and 1 < i < nrl
c      prang probability associated with jth break
c      for each magnitude compute nrl rupture lengths
c       :::::::: must be ordered in descending order--largest break
c       ::::::::: length first for each magnitude
       do 10 j=1,lev
       con=al + bl*fm(j)
       do 10 i=1,nrl
       c=con+pmid(i)*sigl
    10 rlls(i,j)=10.**c
       do 20 i=1,nrl
    20 prls(i)=prang(i)
       nrls=nrl
       return
       end
c ***************************************************************
       subroutine out(reg,naf,xouts,youts,sd,num)
       common/tims/ftim(20),jtim(20),ntims,nwt,iprint
       dimension qp(140),sol(20),sols(20)
       common/quad/maxq,ac(105),aclog(105)
       dimension reg(106)
       common/ext/prob,ex,ex1
       common/wds/max,maxp
       common/wty/wt(101),dev(101)
       dimension regs(106),cdfs(105),cdf(105)
       if(iprint.le.2) write(16,10) xouts,youts,reg(maxp+1),sd
    10 format(' site at long 'f8.3', lat 'f8.3,
      &/' shortest dist to fault='f9.3' km'/10x
      &'zero attenuation variability'18x'variability in atten, sigma='
      & f5.2/2( '     g.m.    occ/yr     exc/yr  r(events) r(yrs) '))
       ict=0
       do 20 l=1,maxp
    20 regs(l)=0
       if(num.eq.2) go to 70
       do 60 l=1,max
       if(reg(l+1).eq.0.) go to 60
       asav=(aclog(l)+aclog(l+1))/2
       kl=1
       do 50 nw=1,nwt
       a=asav+dev(nw)
    30 if(a.le.aclog(kl)) go to 40
       kl=kl+1
       if(kl.le.max) go to 30
    40 regs(kl)=regs(kl)+reg(l+1)*wt(nw)
    50 continue
    60 continue
       go to 120
    70 sd6=4.4*sd
       kls=1
       do 110 l=1,max
       if(reg(l+1).eq.0) go to 110
       asav=(aclog(l)+aclog(l+1))/2.
       kl=kls
    80 if(aclog(kl)-asav.ge.-sd6) go to 90
       kl=kl+1
```

```
      if(kl.le.max) go to 80
      go to 110
   90 kls=kl
      w=(aclog(kl)-asav)/sd
      call bbgaus(w,g1)
      if(kl.eq.1) regs(1)=regs(1)+(1.-g1)*reg(1+1)
  100 kl=kl+1
      wh=(aclog(kl)-asav)/sd
      call bbgaus(wh,g2)
      regs(kl)=regs(kl)+(g1-g2)*reg(1+1)
      if(wh.ge.4.4) go to 110
      if(kl.ge.maxp) go to 110
      g1=g2
      go to 100
  110 continue
c  we are really not interested in 1st 4 acceleration boxes but used
c  them for computation of variability in acceleration--correct so
c  box 5 accumulates all accelerations below ac(5)
c  ac(5) is the first value printed
  120 continue
      regs(1)=regs(1)+reg(1)
      regsav=reg(5)
      regssv=regs(5)
      do 130 i=1,4
      regs(5)=regs(5)+regs(i)
  130 reg(5)=reg(5)+reg(i)
      cdfs(maxp)=0.
      cdf(maxp)=0.
      maxq=maxp
      k=max
      do 140 i=5,max
      cdfs(k)=cdfs(k+1)+regs(k+1)
      cdf(k)=cdf(k+1)+reg(k+1)
  140 k=k-1
      if((cdfs(5).ne.0.0).and.(cdf(5).ne.0.0)) go to 160
      if(iprint.gt.2) write(16,150)
  150 format(' no acceleration events accumulated. statistics calc.
     &bypassed')
      sol(ntims)=0.0
      sols(ntims)=0.
      go to 370
  160 continue
c
c  if accelerations do not exceed ac(5), the first acceleration
c  of interest, skip statistics calculation to avoid dividing by
c  zero or upsetting interpolation routines.
c
      cnums=cdfs(5)+regs(5)
      cnum=cdf(5)+reg(5)
      yrncnum=1./cnum
      nrep=0
      idone=0
      if(iprint.eq.3) go to 320
      do 290 i=5,maxp
      if(cdf(i).eq.0.0) go to 180
      if(reg(i).ne.0) go to 170
```

```
      if(regs(i).eq.0.) go to 290
c
c   compute return period in number of events and number of years
c   return period in events = average number of earthquakes needed
c   to produce an acceleration exceeding a
c       r(events)=(yearly earthquakes)/(yearly exceedances of a)
c   return period in years = average time between earthquakes
c   which cause acceleration a to be exceeded
c      r(yrs)=1/(yearly exceedances of a)
c
  170 retev=cnum/cdf(i)
      retyr=retev*yrn cnum
c
c   compute acceleration which has probability prob of not being
c   exceeded during t years.  this is given by the value of a
c   for which
c      prob = exp(-t * number of exceedances of a per year)
c
      go to 190
  180 retev=0.
      retyr=0
  190 if(cdfs(i).eq.0.) go to 200
      retevs=cnums/cdfs(i)
      retyrs=retevs/cnums
      go to 210
  200 retevs=99999.9
      retyrs=99999.9
      if(cdiv.ne.0.) go to 210
      maxq=i
      idone=1
  210 continue
      qp(nrep+1)=ac(i)
      qp(nrep+2)=reg(i)
      qp(nrep+3)=cdf(i)
      if(retev.ge.100000.) retev=99999.9
      if(retyr.ge.100000.) retyr=99999.9
      qp(nrep+4)=retev
      qp(nrep+5)=retyr
      qp(nrep+6)=ac(i)
      qp(nrep+7)=regs(i)
      qp(nrep+8)=cdfs(i)
      if(retevs.ge.100000.) retevs=99999.9
      if(retyrs.ge.100000.) retyrs=99999.9
      qp(nrep+9)=retevs
      qp(nrep+10)=retyrs
      if(nrep.ne.130) go to 250
      if(naf.le.3) go to 220
      write(16,270) qp
      go to 230
  220 write (16,260) qp
  230 continue
      nrep=0
c  start new page if ict=4
      ict=ict+1
      if(ict.eq.4) write(16,240)
  240 format(1h1)
```

```
      if(idone.eq.1) go to 320
      go to 290
  250 nrep=nrep+10
  260 format(1h f8.2,2f9.5,2f8.1,f12.2,2f9.5,2f8.1)
  270 format(1h f8.4,2f9.5,2f8.1,f12.4,2f9.5,2f8.1)
  280 if (idone.eq.1) go to 300
  290 continue
  300 if(nrep.eq.0) go to  320
      if(naf.le.3) go to 310
      write(16,270) (qp(11),11=1,nrep)
      go to 320
  310 write(16,260) (qp(11),11=1,nrep)
  320 if (iprint.gt.2) go to 340
      write(16,330) cnums,sd
  330 format('   total yearly events ' f10.5,
     &/10x'zero attenuation variability'18x,
     & 'variability in atten, sigma='f4.2)
c
c    compute acceleration which has probability prob of not being
c    exceeded during t years.  this is given by the value of a
c    for which
c       prob = exp(-t * number of exceedances of a per year)
c
  340 do 350 jv=1,ntims
      call linint(reg,ftim(jv),sol(jv))
      call linint(regs,ftim(jv),sols(jv))
      if(iprint.le.2) write(16,360)prob,sol(jv),jtim(jv),sols(jv),
     &jtim(jv)
  350 continue
  360 format(f10.3' ext prob ='f6.3' for 'i4' years'20x
     & f6.3' for 'i4' years')
  370 rat=0.0
      if(sol(ntims).ne.0.0) rat=sol(ntims)/sol(1)
      ratsd=0
      if(sols(ntims).ne.0) ratsd=sols(ntims)/sols(1)
      if(iprint.le.2)write(16,380)jtim(ntims),prob,jtim(1),rat, ratsd
      if(iprint.ge.2) write(2) xouts,youts,(sol(jt),jt=1,ntims),
     &(sols(jt),jt=1,ntims)
  380 format('   ratio'i5' yr'f6.3' extreme value to'i5' yr val
     &='f7.2,7x,f7.2)
      reg(5)=regsav
      regs(5)=regssv
      return
      end
c ***************************************************************
      subroutine bbgaus(x,gg)
c    computes normal probability integral gg from -(infinity) to xx
c     sets gg=1 if xx < -6;   gg=0  if xx > +6
c    uses approximation Handbook of Mathematical Functions-
c    NBS Applied Math Series 55--p299 7.1.26 for error function
      ax=abs(x)/1.4142136
      if(ax-4.2426408) 20,20,10
   10 gg=0.
      if(x.lt.0) gg=+1.
      return
   20 continue
```

```
      d=1.+((((((4.30638e-5*ax+2.765672e-4)*ax+1.520143e-4
     & )*ax+9.2705272e-3)*ax+4.2282012e-2)*ax+7.0523078e-2)*ax
      d=d*d
      d=d*d
      d=d*d
      gg=.5/(d*d)
      if(x) 30,30,40
   30 gg=1.-gg
   40 return
      end
c ****************************************************************
      subroutine outsid(xnot,ynot,isub,r,angle)
      common/inout/ic,ics,ifar,azimf,stsize
      common/xysav/xsav(2,50),ysav(2,50),sarea(50)
       common/xlyl/xl(4),yl(4),xr(4),yr(4), xc(4),yc(4)
       common/slinp / aa(4,50),bb(4,50),cc(4,50),rta(4,50),dist(4,50)
     &,disq(4,50)
      common/slinds/dls(4),perp(4),sll(4),slsq(4),min(4),persq(4)
     &,srsq(4),dlp(4)
      dimension jsub(3)
c     subroutine for calculating angle when site is outside
c          (quadrilateral) source area.
c      r is the radial distance for which angle is sought
c
c     (xnot,ynot)=point outside quadrilateral source region isub
c     r=radius of circle (or radius of annular ring of width dr)
c     centered at (xnot,ynot)
c     isub = index identifying quadrilateral source region
c
c    the area contained in the intersection of the region and annular
c     ring is determined by the fraction of the ring within the region.
c            area = angle x r x dr
c     where angle = the angle (in radians) subtended by the
c     part of the annular ring contained within the region.
c
c          (rc is closest distance between site and source.)
c          (rf is furthest distance between site and source)
c       subroutine csqdis has computed distance to vertices of subregion
c       and shortest distance to each side
c       adapted from subroutine by mcguire
c
      data jsub/2,3,1/
      npt=0
      angle=0.
      signal=1.
c        loop on each side
      rsq=r*r
      do 140 ii=1,4
      if(min(ii)-2) 10,40,70
c     closest point is xr(ii),yr(ii)
c       at most one intersection
   10 if(rsq.gt.slsq(ii)) go to 140
      if(rsq.lt.srsq(ii)) go to 140
      d=(sqrt(rsq-persq(ii))-dlp(ii))/dist(ii,isub)
   20 y1=(yl(ii)-yr(ii))*d+yr(ii)
      x1=(xl(ii)-xr(ii))*d+xr(ii)
```

```
c  intersects in range
c           does circle (radius r) intersect it?
c         store first point
   30 npt=npt+1
      xc(npt)=x1
      yc(npt)=y1
      go to 140
c   closest point is at xl(ii),yl(ii)
   40 if(rsq.lt.slsq(ii)) go to 140
      if(rsq.gt.srsq(ii)) go to 140
   50 d=(sqrt(rsq-persq(ii))+dls(ii))/dist(ii,isub)
   60 x1=xl(ii)+(xr(ii)-xl(ii))*d
      y1=yl(ii)+(yr(ii)-yl(ii))*d
      go to 30
   70 if(rsq.lt.persq(ii)) go to 140
      if(rsq.le.slsq(ii)) go to 80
      if(rsq.gt.srsq(ii)) go to 140
c    single intersection
c      on side nearer xr,yr
      go to 50
   80 if(rsq.gt.srsq(ii)) go to 90
c      two intersections on this side
c     can compute angle directly without determining coordinates
      arg=abs(perp(ii))/r
      as=acos(arg)
      go to 100
   90 d=(dls(ii)-sqrt(rsq-persq(ii)) )/dist(ii,isub)
      go to 60
c     single intersection
c         see if this side is closest to point, if so, treat specially.
  100 if (ii-ics) 110,120,110
c           both points are on boundary, calculate angle betwen them.
  110 sign=-1.
      go to 130
  120 sign=1.
      signal=-1.
  130 angle=sign*2.*as + angle
c         see if second point only is on boundary
  140 continue
      if(npt.gt.0) go to 150
      if(signal) 310,320,320
  150 go to (320,240,160,250),npt
c      this is an error unless radius is on a vertex and
c       was counted twice.
  160 do 170 i=1,3
      j=jsub(i)
      if(abs(xc(i)-xc(j)) .ge. .01) go to 170
      if(abs(yc(i)-yc(j)) .le. .01) go to 220
  170 continue
c    check for one of three points being a vertex
      do 210 i=1,4
      do 180 j=1,3
      if(abs(xc(j)-xl(i)).gt. .001) go to 180
      if(abs(yc(j)-yl(i)) .le. .001) go to 190
  180 continue
      go to 210
```

```
  190 if(j-2) 200,200,230
  200 xc(j)=xc(3)
      yc(j)=yc(3)
      go to 230
  210 continue
      go to 320
  220 if(i.ne.1) go to 230
      xc(2)=xc(3)
      yc(2)=yc(3)
  230 npt=2
      go to 240
  240 ad=sqrt((xc(1)-xc(2))*(xc(1)-xc(2))+(yc(1)-yc(2))*(yc(1)-yc(2)))
      angle=angle + signal*2.*asin(ad/(2.*r))
      go to 310
c           four intersection points (each on a different side).
c           determine angle by finding closest 2 intersections to
c           farthest corner, calculate angle between, and add angle
c           between other two intersections.
  250 dist1=10000000.
      i1=0
      i2=0
      i3=0
      i4=0
      do 300 jj=1,4
      distn=(xl(ifar)-xc(jj))*(xl(ifar)-xc(jj))
     &  +(yl(ifar)-yc(jj))*(yl(ifar)-yc(jj))
      if (distn-dist1) 260,260,270
  260 dist2=dist1
      dist1=distn
      i4=i3
      i3=i2
      i2=i1
      i1=jj
      go to 300
  270 if (distn-dist2) 280,280,290
  280 dist2=distn
      i4=i3
      i3=i2
      i2=jj
      go to 300
  290 i4=i3
      i3=jj
  300 continue
c            calculate angle between 2 closest points to farthest corner.
      ad=sqrt((xc(i1)-xc(i2))*(xc(i1)-xc(i2))
     &  + (yc(i1)-yc(i2))*(yc(i1)-yc(i2)))
      angle=2.*asin(ad/(2.*r))
c          calculate angle between 2 points farthest from
c          farthest corner and add to previous angle.
      ad=sqrt((xc(i3)-xc(i4))*(xc(i3)-xc(i4))
     &  +(yc(i3)-yc(i4))*(yc(i3)-yc(i4)))
      angle=2.*asin(ad/(2.*r)) + angle
  310 continue
c         compute rate of earthquakes in this annular source
      return
c         error printout
```

```
  320 write(16,330)isub,ic,npt,xnot,ynot,(xl(i),yl(i), i=1,4),r,angle,
     &(xc(i),yc(i),i=1,4)
  330 format(' ***** error in subroutine outsid. source no. ',i3,
     &' debug values follow..... ',/10x,2i5,5(/10x,4f14.6))
  340 call exit
      end
c ***************************************************************
      subroutine inside(xnot,ynot,isub,r,pangle)
      common/xyarea/xsav(2,50),ysav(2,50),sarea(50)
      common/inout/ic,ics,ifar,azimf,stsize,stepo
      common/slinds/dls(4),perp(4),sll(4),slsq(4),min(4),persq(4)
     &,srsq(4),dlp(4)
      common/slinp/aa(4,50),bb(4,50),cc(4,50),rta(4,50),dist(4,50),
     & disq(4,50)
      common/xlyl/xl(4),yl(4),xr(4),yr(4),xc(4),yc(4)
      dimension jsub(3)
c
c     (xmid,ymid)=point inside quadrilateral source region ii
c     r=radius of circle (or radius of annular ring of width dr)
c     centered at (xmid,ymid)
c     ii = index identifying quadrilateral source region
c
c    the area contained in the intersection of the region and annular
c     ring is determined by the fraction of the ring within the region.
c           area = pangle x r x dr
c     where pangle = the angle (in radians) subtended by the
c     part of the annular ring contained within the region.
c     subroutine insid determines pangle.
c
c       adapted from subroutine by mcguire
      data pi,pi2/3.1415927,6.2831853/
      data jsub/2,3,1/
      rsq=r*r
      npt=0
      angle=0.
c         loop on each side
      do 110 ii=1,4
      if(min(ii)-2) 10,40,70
c     closest point is xr(ii),yr(ii)
c       at most one intersection
   10 if(rsq.gt.slsq(ii)) go to 110
      if(rsq.lt.srsq(ii)) go to 110
      d=(sqrt(rsq-persq(ii))-dlp(ii))/dist(ii,isub)
   20 yl=(yl(ii)-yr(ii))*d+yr(ii)
      xl=(xl(ii)-xr(ii))*d+xr(ii)
c  intersects in range
c          does circle (radius r) intersect it?
c        store first point
   30 npt=npt+1
      xc(npt)=xl
      yc(npt)=yl
      go to 110
c   closest point is at xl(ii),yl(ii)
   40 if(rsq.lt.slsq(ii)) go to 110
      if(rsq.gt.srsq(ii)) go to 110
   50 d=(sqrt(rsq-persq(ii))+dls(ii))/dist(ii,isub)
```

```
   60 x1=xl(ii)+(xr(ii)-xl(ii))*d
      y1=yl(ii)+(yr(ii)-yl(ii))*d
      go to 30
   70 if(rsq.lt.persq(ii)) go to 110
      if(rsq.le.slsq(ii)) go to 80
      if(rsq.gt.srsq(ii)) go to 110
c    single intersection
c      on side nearer xr,yr
      go to 50
   80 if(rsq.gt.srsq(ii)) go to 90
c      two intersections on this side
c     can compute angle directly without determining coordinates
      arg=abs(perp(ii))/r
      as=acos(arg)
      go to 100
   90 d=(dls(ii)-sqrt(rsq-persq(ii)))/dist(ii,isub)
      go to 60
c     single intersection
c           both points are on boundary, calculate angle betwen them.
  100 angle=2.*as + angle
c         see if second point only is on boundary
  110 continue
      if (npt) 570,120,140
  120 if (angle-0.001) 570,130,130
c         following is for case of no single intersection points;
c         angle is 2 * pi - angle calculated so far.
  130 pangle=pi2-angle
      go to 560
  140 go to (570,230,150,500),npt
c     this is an error unless radius is on a vertex and
c      was counted twice.
  150 do 160 i=1,3
      j=jsub(i)
      if(abs(xc(i)-xc(j)) .ge. .01) go to 160
      if(abs(yc(i)-yc(j)) .le. .01) go to 210
  160 continue
c    check for one of three points being a vertex
      do 200 i=1,4
      do 170 j=1,3
      if(abs(xc(j)-xl(i)).le. .001) go to 170
      if(abs(yc(j)-yl(i)) .gt. .001) go to 180
  170 continue
      go to 200
  180 if(j-2) 190,190,220
  190 xc(j)=xc(3)
      yc(j)=yc(3)
      go to 220
  200 continue
      go to 570
  210 if(i.ne.1) go to 220
      xc(2)=xc(3)
      yc(2)=yc(3)
  220 npt=2
      go to 230
c          2 intersection points; determine azimuths.
  230 if (xc(1)-xnot-r) 260,250,240
```

```
  240 if (xc(1)-xnot-r-0.001) 250,250,570
  250 azim1=0.0
      go to 300
  260 if (xc(1)-xnot+r) 270,280,290
  270 if (xc(1)-xnot+r+0.001) 570,280,280
  280 azim1=pi
      go to 320
  290 azim1=acos((xc(1)-xnot)/r)
  300 if (yc(1)-ynot) 310,320,320
  310 azim1=pi2  - azim1
  320 if (xc(2)-xnot-r) 350,340,330
  330 if (xc(2)-xnot-r-0.001) 340,340,570
  340 azim2=0.0
      go to 390
  350 if (xc(2)-xnot+r) 360,370,380
  360 if (xc(2)-xnot+r+0.001) 570,370,370
  370 azim2=pi
      go to 410
  380 azim2=acos((xc(2)-xnot)/r)
  390 if (yc(2)-ynot) 400,410,410
  400 azim2=pi2 -azim2
  410 pangle=azim2-azim1
      if (pangle) 420,570,460
  420 if (azim1-azimf) 430,570,440
  430 pangle=pi2 +pangle -angle
      go to 560
  440 if (azimf-azim2) 430,570,450
  450 pangle=-pangle-angle
      go to 560
  460 if (azim2-azimf) 470,570,480
  470 pangle=pi2 -pangle -angle
      go to 560
  480 if (azimf-azim1) 470,570,490
  490 pangle=pangle-angle
      go to 560
c        four intersection points (each on a different side).
c        determine angle by finding closest 2 intersections to
c        farthest corner, calculate angle betwen, and add angle
c        between other two intersections.
  500 dist1=10000000.
      i1=0
      i2=0
      i3=0
      i4=0
      do 550 jj=1,4
      distn=(xl(ifar)-xc(jj))*(xl(ifar)-xc(jj))
     &    +(yl(ifar)-yc(jj))*(yl(ifar)-yc(jj))
      if (distn-dist1) 510,510,520
  510 dist2=dist1
      dist1=distn
      i4=i3
      i3=i2
      i2=i1
      i1=jj
      go to 550
  520 if(distn-dist2) 530,530,540
```

```
  530 dist2=distn
      i4=i3
      i3=i2
      i2=jj
      go to 550
  540 i4=i3
      i3=jj
  550 continue
c         calculate angle between 2 closest points to farthest corner
      ad=sqrt((xc(i1)-xc(i2))*(xc(i1)-xc(i2))
     &  + (yc(i1)-yc(i2))*(yc(i1)-yc(i2)))
      pangle=2.*asin(ad/(2.*r))
c          calculate angle between 2 farthest points from
c          farthest corner and add to previous angle.
      ad=sqrt((xc(i3)-xc(i4))*(xc(i3)-xc(i4))
     & + (yc(i3)-yc(i4))*(yc(i3)-yc(i4)))
      pangle=2.*asin(ad/(2.*r))+pangle
c        angle for this radius is now known, calculate risk
  560 return
c     anarea=pangle*r*stsize
c           error printout
  570 write (16,580) isub,ifar,npt,xnot,ynot,(x1(i),y1(i), i=1,4),r,
     &pangle,(xc(i),yc(i),i=1,4)
  580 format (' ***** error in subroutine inside. source no. ',
     & i3,' debug values follow..... ',/10x,2i10,10(/10x,2f12.6))
  590 call exit
      end
```