COMPUTATION RESEARCH GROUP
Stanford Linear Accelerator
Stanford, California

CGTM NO. 193
December 1980
(Rev. Dec. 1981)

DIVONNE4

A Program for Multiple Integration and Adaptive Importance Sampling

Jerome H. Friedman

Computation Research Group
Stanford Linear Accelerator Center

and

Margaret H. Wright

Systems Optimization Laboratory
Stanford University

The DIVONNE4 package is a collection of subroutines for aiding in the numerical integration of functions of several variables. It can also be used to sample points in multidimensional coordinate spaces from a user specified probability density function. The program adaptively partitions a multidimensional coordinate space (integration volume) into a set of axis oriented hyperrectangular regions, based on a user provided function of the coordinate values. The goal is to adjust the size and shape of the regions so that the range of function values within each is small. These regions are then used as the basis for a stratified sampling estimate of the integral of the function, or to sample random vectors from the coordinate space with probability density that of the function. (See Friedman and Wright, 1981a, 1981b, for a detailed description of the algorithms).

General Note: Integration in moderate to high dimension of even an apparently simple function can often be a difficult problem. The user is urged to experiment with the various parameters and options available for the purpose of validating the results. Another useful validation procedure (when possible) is to calibrate the various parameters and options on a function similar to the integrand but for which the integral is known.

Structure

Subroutine subprogram

User Entry Names: PARTN, INTGRL, USRINT, GENPNT, RANGEN, TREDUMP
BUKDMP, USRTRM, DFUN

Internal Entry Names:	SPLIT, QUASI, RECPAR, BOUNDS, TREAUD, NODAUD, BUCMVE, FREQN, NOCUT, TSTEXT, DELSLV, BUFOPT, BNDOPT, SETTOL, BNDTST, COPY, GRDCMP, DELETE, BFGS, MODCHL, NMDCHL, LDLSOL, SHRNK, FEASMV, ADDBND, MULCHK, DELBND, LOCSCH, ORTHVC, MXSTEP, NEWPTQ, RLEN, EXMBUC, QUAD, DOT.
Files References:	TAPE6 (output) and TAPE NFILE: optional user defined external file (see below)
External routines referenced:	FORTRAN library
Common Blocks:	/DATE/, /PRINT/, /ISTRGE/, /RSTRGE/, /LIMITS/, /INSMPL/, /QUADRE/, /START/, /EXFILE/, /DISPOS/, /DEPTHS/, /MXMZER/, /SAMPLE/, /SPCNTL/, /MLIMIT/, /TRESZE/, /EXMCTL/, /BUKSZE/, /GENINL/, /FUNN/, /ANSWER/, /MAXERR/, /SIGSPL/, /STRAND/, /WITCH/, /BNDLMT/, /PRSTOP/, /ZOOO1/, /CUTLOS/, /ZEETRM/

NOTE: This documentation is intended to describe DIVONNE4 (12/17/80). A maximum coordinate space dimensionality is defined at compile time. For this version of the program, it is set to 10.

Function Specification

The function (integrand or probability density) is defined by a user coded function subprogram:

```
      DOUBLE PRECISION FUNCTION DFUN (N,X)
      INTEGER N
      DOUBLE PRECISION X(N)
        .
        .
        .
      DFUN=
        .
        .
        .
      RETURN
      END
```

Both arguments are input to the subprogram. The first N is the number of integration variables (dimension of integration variable space) and the second is a vector containing the coordinates of a point in the integration volume. The function subprogram must return (in DFUN) the value of the integrand or probability density for the coordinate values stored in X.

WARNING: The values stored in X must not be changed in this subprogram.

Integration - Automatic Invocation

Integration is most simply invoked by the subroutine Call:

```
CALL DIVON(N, XMINUS, XPLUS, ERRMAX, MAXNUM, ANS, ERROR)
Integer N, MAXNUM
REAL XMINUS(N), XPLUS(N), ERRMAX, ANS, ERROR
```

Input parameters (specified by user):

N: dimension of integration variable space.

XMINUS(I), I=1,N: lower integration limit on Ith integration variable

XPLUS(I), I=1,N: upper integration limit on Ith integration variable

ERRMAX: permissible uncertainty of integral estimate.
(Algorithm terminates when estimated uncertainty

is below ERRMAX.)

MAXNUM: maximum number of function evaluations. (Algorithm terminates when the number of calls to DFUN exceeds MAXNUM, regardless of uncertainty estimate.)

Output parameters:

ANS: integral estimate

ERROR: associated uncertainty estimate

NOTE: The hyperrectangle defined by the limits XMINUS, XPLUS must enclose the actual region of integration. If the integration region is smaller than this hyperrectangle (e.g., nonrectangular), the value of the integrand (DFUN) must be set to zero whenever the input vector (X) lies outside the true region of integration.

WARNING: DIVON terminates the partitioning and chooses the integration sample size based on certain assumptions concerning the accuracy of quasi-uniform Monte Carlo integration. These assumptions tend to be valid for relatively easy integrands but may not be valid for difficult integration problems. For difficult problems, DIVON may terminate the partitioning too soon and/or choose too small a sample for the final integration. For these cases, the partitioning and integration phases should be invoked separately under user control as described below.

Integration - Detailed Invocation

Greater power and flexibility can be achieved by invoking the partitioning and integration phases of the algorithm separately.

The partitioning is invoked by a subroutine call

```
CALL PARTN (N, XMINUS, XPLUS, SPRDMX, MAXPTS).
```

All arguments are input to the subroutine and must be specified by the user. The parameters N, XMINUS, XPLUS are described above. SPRDMX and MAXPTS control termination of the partitioning.

If SPRDMX <0, then partitioning terminates when the root sum squared spread of function values (ΔF) summed over the current set of regions is less than |SPRDMX|.

If SPRDMX ≥0, then partitioning terminates when $\Delta F/|I| \leq$ SPRDMX. Here I is an approximate estimate of the integral of the function maintained while partitioning.

Under usual conditions (functions that are not excessively difficult), values of SPRDMX from 1.0 to 2.0 are reasonable. For difficult functions, some experimentation may be required to determine a good value.

MAXPTS = maximum number of integrand evaluations. Control is returned to the calling program at the end of the first iteration for which the number of integrand evaluations (calls to DFUN) exceed MAXPTS.

To obtain a more accurate estimate of the definite integral, the subroutine call

```
CALL INTGRL(N,JDEG,NPT,ANS,ERROR)
```

must be executed after control returns from PARTN. The first three arguments are input and the last two output. N is the dimension of the integration variable space. JDEG controls the several options supplied for estimating the integral within each hyperrectangular region. The total integral estimate is taken as the sum over all regions of the individual region estimates.

JDEG=0: quasi-uniform Monte Carlo estimate

JDEG=1: pseudo random Monte Carlo estimate with importance sampling

JDEG=2: second degree quadrature formula estimate

JDEG=3: third degree quadrature formula estimate

JDEG=5: fifth degree quadrature formula estimate

JDEG<0: estimate by user provided procedure (see below)

Experience indicates that the quasi-uniform Monte Carlo (JDEG=0) tends to produce the most accurate integral estimates. Uncertainty estimates (ERROR) are returned only for the Monte Carlo (JDEG=0,1) and user provided (JDEG<0, see below) estimates.

NPT controls the number of sample points used for the final integral estimate in INTGRL when Monte Carlo methods are used. For quasi-uniform Monte Carlo (JDEG=0), NPT is the number of samples drawn in each region. For pseudo-random Monte Carlo (JDEG=1), NPT times the number of regions is the total number of samples drawn. The number used in each region is proportional to the spread, S_i, within the region. For quadrature formula estimation, the number of samples drawn in each region depends upon the particular formula. For those provided here, the numbers are as follows:

2nd degree formula (JDEG=2) N+1 points/region

3rd degree formula (JDEG=3) 2*N points/region

5th degree formula (JDEG=5) 2*N**2+1 points/region

The output quantities, ANS and ERROR, are the estimated values of the integral and associated uncertainty (when provided), respectively.

It should be noted that the function used to partition the coordinate space need not be the same as that being integrated with INTGRL. In particular, a single partitioning may be used in the integration of several not too dissimilar functions (over the same region of integration) by repeated calls to INTGRL. If the integrand is to change between such calls, there must be user coded logic in both the calling program and DFUN, to ensure that the correct integrand is evaluated. Communication between the calling program and DFUN can be accomplished with labeled commons.

Importance Sampling

There are two ways to use DIVONNE4 to sample points from a user specified density function. For both methods, the density function is coded in DFUN and the partitioning is invoked by executing a call to PARTN as described above. The user density (as coded in DFUN) need not be normalized. Before the call to PARTN, the labeled common

```
COMMON/QUADRE/IDEG
```

must be declared and the value of IDEG set to 1 (IDEG=1). This causes PARTN to calculate the necessary quantities for the importance sampling application.

In order to randomly draw vectors in the coordinate space with probability density approximating the user defined function, the subroutine call

```
CALL GENPNT(N,X,WT)
```

must be repeatedly executed after control returns from PARTN. Each such execution will store the N coordinates of a new random point in the vector X, and a corresponding weight in WT. Applying this weight with the point X, results in distributions that are statistically equivalent to a sample of points distributed with the density coded in DFUN.

In order to randomly draw vectors in the coordinate space with the precise probability density defined in DFUN, the subroutine call

```
CALL RANGEN(N,X)
```

must be repeatedly executed after control returns from PARTN. Each such execution will store the N coordinates of a new random point in the vector X. These points will be a sample drawn from the density coded in DFUN.

The decision whether to use weighted (GENPNT) or unweighted (RANGEN) points depends on the nature of the application. If the sample is to be used for statistical summaries only, and the computation performed on an observation is small compared to the computation needed to generate it (in GENPNT), then

weighted observations are the most efficient. On the other hand, if this is not the case, then unweighted observations can be more efficient.

Method

The partitioning is accomplished by a nested refinement procedure. The procedure is iterative and, at any stage, it is only concerned with one particular hyperrectangular region. Associated with each region is a measure of the spread of function values within the region. At any stage of the partitioning, the region for which this measure is largest is the one chosen for further paritioning (initially, the whole integration region is the one under consideration).

The quantity used to characterize the variation V_i of the function values within each region, R_i, is the range or difference of extreme values

$$V_i = \max_{\vec{x} \in R_i} f(\vec{x}) - \min_{\vec{x} \in R_i} f(\vec{x}) \qquad (1)$$

The extreme values are found by a special numerical algorithm. This choice of a measure transforms the estimation problem to an optimization problem. The quantity

$$S_i = 1/2\ V_i \cdot \mathrm{Vol}\ (R_i) \qquad (2)$$

(called the spread) bounds the uncertainty of any Monte Carlo or quadrature estimate of the integral within the region and is used as a relative measure of the contribution of the region to the uncertainty of the global integral estimate.

The purpose of refining the region under consideration into two or more subregions is to reduce its contribution to the global uncertainty, thus reducing the global uncertainty. After the refinement, the daughter subregions are merged into the list of all regions and the one with the largest spread measure, S_i, is the next one to be considered for further refinement. This successive refinement continues until the root sum of squares of the spread

values

$$S = \left[\sum_{i=1}^{M} S_i^2 \right]^{1/2} \qquad \text{(M=number of regions)}$$

is less than the user specified limit, or until the number of function evaluations exceeds the user specified maximum.

After partitioning, the variation of the function values within each hyperrectangular region is considerably smaller than the variation of the function over the total integration volume. One can then apply standard numerical integration techniques to evaluate the integral within each subregion. These techniques tend to work very well when the variation of integrand values is small. The standard techniques provided in DIVONNE4 are quasi-uniform and pseudo-random Monte Carlo methods, and three quadrature formulas. The user can also provide additional methods at his/her discretion. Of the methods provided with DIVONNE4, the quasi-uniform Monte Carlo seems to perform the best when there are not large discontinuities in the function within the region of integration. When there are, the pseudo-random Monte Carlo seems to work best. The quadrature formulae tend to perform poorly in all but the very easiest problems.

Random vectors are sampled by choosing randomly a hyperrectangular region with probability proportional to the estimated integral within the region. A very approximate (but adequate for this purpose) estimate of the integral is maintained during partitioning. If a more accurate estimate is desired, INTGRL may be called before sampling begins. Once a region is chosen, a vector is drawn randomly with uniform probability (GENPNT) or with probability proportional to the function (RANGEN) from within the region.

Additional Options (Bells and Whistles)

Although the minimum necessary to use the program has been described above, there are several user options that can enhance the power, flexibility and usefulness of the program. These options are invoked by changing the value of an internal parameter stored in a labeled common. Default values for these parameters are set via a BLOCK DATA subprogram internal to the DIVONNE4 package at compilation time. they can be changed by declaring the appropriate labeled common in a user routine and then reset via an executable statement.

Speed Option

```
COMMON/BNDLMT/FLOW, FHIGH
```

FLOW and FHIGH are upper and lower bounds (respectively) on the value of the user defined function (DFUN) within the coordinate space (integration region). If either or both of these values are known, setting FLOW and/or FHIGH to the appropriate value may cause more efficient operation of the algorithm. Default values are -9.9E71 and 9.9E71.

Numerical Options

```
COMMON/SAMPLE/NPOINT
```

As part of the partitioning procedure for each subregion, the program draws NPOINT quasi-uniform vectors within each region. The vector for which the function value is largest/smallest is used as the starting point for the maximization/minimization. The average function value for these vectors is used to obtain an approximate estimate of the integral within the region. Decreasing NPOINT may reduce the number of function evaluations used in the par-

titioning. Increasing its value increases the robustness of the optimization procedure and increases the accuracy of the approximate integral estimate. NPOINT must be greater than two and less than a maximum value set at compile time (currently 200). The default value is 47.

COMMON/QUADRE/IDEG

In addition to the quasi-uniform estimate of the approximate integral within each region during the partitioning, the user may invoke quadrature formula estimates. These estimates may be useful for defining alternate criteria for termination of partitioning. Usually, these quadrature estimates are not useful during partitioning and simply waste function evaluations.

IDEG = 0: no numerical quadrature

IDEG = 1: importance sampling (see above)

IDEG = 2: 2nd degree quadrature within each subregion

IDEG = 3: 2nd and 3rd degree quadrature within each subregion

IDEG = 5: 2nd, 3rd and 5th degree quadrature within each region

Default = 0

COMMON/PRSTOP/NSTOP

NSTOP controls the partitioning termination when integration is invoked by calling DIVON. The iterative partitioning is terminated when the estimated total number of integrand evaluations (partitioning plus final integration) does not decrease for NSTOP successive iterations.

Default value = 5

Control Options

COMMON/PRINT/IPRINT

Controls printed output. Iteration information is printed when the iteration number is an exact multiple of IPRINT and upon termination. Setting IPRINT to zero inhibits all printed output.

default = 1 (maximum printing)

COMMON/EXFILE/NFILE

Defines FORTRAN file number (TAPE NFILE) for external storage to save intermediate results between runs (see below).

default = 1

COMMON/DISPOS/IDISP

Controls the storage on an external file of information necessary to continue partitioning with a future job (or job step) at the last iteration of the present one.

IDISP = 0 : no action

IDISP ≠ 0 just before PARTN returns control to calling program, TAPE NFILE is rewound and information is written onto TAPE NFILE for continuing the partitioning at that point at which it left off.

default = 0

COMMON/START/ISTART

Defines initialization state at time of call to PARTN.

ISTART = 1 : new problem, begin partitioning

ISTART = 2 : subsequent call, continue partitioning from last iteration of previous call to PARTN in this job step. This value (ISTART=2) is not

valid for the first call to PARTN within a job step.

ISTART = 3 : continue partitioning from last iteration of previous job (or job step) by reading information from TAPE NFILE

default = 1

Output

During the execution of the program, the user can have access to intermediate results by declaring the appropriate labeled commons. These values are the current estimates of the various quantities and are updated with each iteration. For the correct functioning of the algorithm, these quantities should never be reset by the user.

```
COMMON/FUNN/NFUN, NOPT, NCUT
```

NFUN = Total number of integrand evaluations

NOPT = Number of those used for numerical optimization

NCUT = Number of those used for finding cut points

```
COMMON/ANSWER/FINTGL,SPRD,SPRDMX,RANEFF,QUAD2,QUAD3,QUAD5,NRGN,MAXRGN
```

FINTGL = Approximate quasi-Monte Carlo estimate of integral

SPRD = Root sum squared spread

SPRDMX = Largest region spread

RANEFF = Estimted sampling efficiency for RANGEN

QUAD2 = 2nd degree quadrature estimate of integral

QUAD3 = 3rd degree quadrature estimte of integral

QUAD5 = 5th degree quadrature estimate of integral

NRGN = Number of subregions

MAXRGN = Subregion number with largest spread

RANGEN efficiency exists only if IDEG has been set (IDEG=1) before the call to PARTN. The quadrature estimates exist only if IDEG has been set appropriately (IDEG=2,3,5) before the call to PARTN.

User Defined Termination

When invoked, PARTN returns control to the calling program only if the specified root sum squared spread has been reached or the specified number of integrand evaluations have been exceeded. Based on his/her own logic and the values of the various output quantities defined above (for example, consistency among the various quadrature formulae and the quasi-Monte Carlo estimate or sufficient generation efficiency for RANGEN), the user may wish to define other stopping criteria. This can be done by coding a user defined function subroutine.

```
LOGICAL FUNCTION USRTRM (ITER)
   .
   .
   .
RETURN
END
```

The argument is input to the routine and contains, as a value, the iteration number. Returning a value .FALSE. for USRTRM will cause continued iteration subject to the usual convergence criteria. Returning a value of .TRUE. will cause immediate termination of the partitioning procedure.

User Provided Integration Procedure

The numerical integration procedures provided with DIVONNE4, for obtaining the integral estimates within each subregion, are by no means exhaustive and the user may wish to try others. This is accomplished by setting IDEG < 0 in the call to INTGRL and by supplying a subroutine with the following header.

SUBROUTINE USRINT (RGNL,RGNU,FINTIN,SPRDIN,FNTOUT,ERRSQ)

The first four quantities are input to the subroutine and the last two are output.

RGNL(N)	=	lower limits on each coordinate that define the region
RGNU(N)	=	upper limits on each coordinate that define the region
FINTIN	=	approximate quasi-Monte Carlo estimate of integral within the region obtained during partitioning
SPRDIN	=	function spread measure within region
FNTOUT	=	estimate of integral within region
ERRSQ	=	squared error estimate of integral evaluation within region

The total integral is taken to be the sum of the individual region estimates and the uncertainty is taken to be the square root of the sum of the squared error estimates.

ACKNOWLEDGMENT

The authors wish to thank David Carey and Fred James for helpful suggestions

REFERENCES

Friedman, J.H. and Wright, M.H. (1981a). A Nested Partitioning Procedure for Numerical Multiple Integration. ACM Transactions on Mathematical Software, March 1981.

Friedman, J.H. and Wright, M.H. (1981b). Adaptive Importance Sampling, Stanford Linear Accelerator Center, SLAC PUB, Feb. 1981.

APPENDIX

This section is added for completeness. It discusses internal aspects of the program that should seldom, if ever, concern the user.

Storage Management

The memory necessary to manage the partitioning and the associated hyper-rectangular regions is declared in two labeled commons:

```
COMMON/ISTRGE/MXRGNS, ISTOR (4*MXRGNS)
COMMON/RSTRGE/RSTSZE, RSTOR (RSTSIZE)
```

The first is for storage of integer quantities and the second is for real variables. MXRGNS is the maximum number of regions for which there is sufficient storage. As distributed, the program permits a maximum of 3000 regions. As indicated, ISTOR must be dimensioned four times MXRGNS. RSTSZE is simply an integer that stores the dimension of RSTOR. If RSTSZE is not large enough for the maximum number of regions specified (MXRGNS), then the program will automatically reduce MXRGNS so that there is sufficient storage. As distributed, RSTSZE=18001, which is enough storage for 3000 regions if no quadrature formula estimation is used during partitioning. If the computer used to run DIVONNE4 does not have sufficient memory, these quantities and the associated dimension declarations can be reduced.

If, during partitioning, the total number of regions is about to exceed the maximum number for which there is sufficient storage, the partitioning terminates and control is returned to the calling program. If IDISP has been set nonzero, then information for continuing the partitioning has been written to TAPE NFILE. The user may then increase the available storage by declaring the above commons in the calling program and increasing the values of MXRGNS and RSTSZE (as well as the associated dimensions of ISTOR and RSTOR), and continue the partitioning where it left off at the previous job.

Control of Recursion

The discussion of the partitioning procedure above is slightly over-simplified in that it was described as a purely iterative procedure. For efficiency considerations, the iteration is augmented by recursion. That is, after a region has been partitioned into daughter subregions, each daughter is immediately under consideration for further splitting without its being incorporated into the list of all regions and updating the list. Each daughter that is further partitioned is also under immediate consideration for further refinement and so forth. This recursion stops when all daughters have values of the spread measure less than the second largest value in the list (remember that the region with the largest value was the one chosen for partitioning) or until a maximum recursion depth is reached.

The maximum recursion depth is specified in a labeled common

COMMON/DEPTHS/ISTDPH, INCDPH

The first integer (ISTDPH) is the maximum recursion depth for the very first region under consideration, namely, the whole integration volume. INCDPH is the maximum recursion depths for all further subregions under consideration. Default values set in the program are ISTDPH=3 and INCDPH=5.

Control of Optimization

The optimization procedure is described in some detail in Friedman and Wright (1981a), and hence only a brief overview will be given here. The method used is a non-derivative bound-constrained quasi-Newton method (see Gill, Murray and Wright, 1981, for a duscussion of non-derivative method and the treatment of bound constraints). Optimization methods typically include various parameters that control portions of the algorithm. In the present

program, these parameters are set to default values that work well on most problems; however, it may be helpful (and in some cases, necessary) to alter these parameters.

The subroutine SETTOL sets the values of parameters used in the optimization procedure. The user should alter the FORTRAN statements that assign any of the values to be changed. The parameters will be described in the following paragraphs.

The value FTOL is a tolerance used to test for termination of the optimization within the current subspace. Roughly speaking, the optimization is considered to have converged if the change in the objective function or the change in the vector of free variables is less than FTOL. If FTOL is positive, the test involves $\Delta F/(1+|F|)$, a combination of relative and absolute tolerances, where F is the function being minimized, and ΔF is the change in F during the last iteration (with a similar test for the change in the free variables). If FTOL is negative, a purely relative test is used (with some safeguards to prevent overflow if $|F|$ becomes very small). The default value of FTOL is 5×10^{-2}, and corresponds to a fairly loose convergence test.

The value GTOL is a tolerance that is also used to test for convergence of the optimization in the current subspace. If the length of the gradient with respect to the free variables is less than GTOL, the optimization is terminated . If the length of the gradient with respect to the free variables is less than GTOL at the beginning of the optimization, a special local search is performed in order to avoid being stuck at a stationary point (see Friedman and Wright, 1981a, for details of the local search). GTOL must be positive; the default value is 10^{-2}, which corresponds to a fairly loose convergence test.

The parameter DELTA gives the finite-difference interval to be used in computing forward-difference approximations to the gradient vector. For a well scaled problem in which the objective function can be calculated to full machine precision, the value of DELTA should be no less than the square root of machine precision. However, a larger value of DELTA may be necessary when the calculation of F includes noise. The size of DELTA is also used to define when a variable is "on" a bound, i.e. the variable is set to its bound whenever it lies within DELTA of the bound. Since a single value of DELTA is used for all the variables, it is extremely important for the variables to be scaled uniformly with respect to one another. A detailed discussion of problem scaling is given in Chapter 8 of Gill, Murray and Wright (1981). The default value of DELTA is 10^{-5}.

At each iteration of the optimization procedure, a linear search is performed to obtain a lower function value. The parameter ETA, which lies in (0,1), controls the accuracy of the line search as follows. If ETA is very close to zero, a highly accurate line search is performed, i.e. a close apporximation to the minimum along each search direction is computed; if ETA is close to unity, the line search will terminate as soon as a lower point has been found. A small value of ETA tends to produce a more accurate quasi-Newton approximation to the Hessian, and hence may allow convergence in fewer iterations. On the other hand, a larger value of ETA means that fewer function evaluations are required during each iteration. The default value of ETA is 0.2.

The value ALFMAX imposes an upper bound on the change in length of the vector of free variables at each iteration. Within the optimization algorithm, the step to be taken at each iteration is also bounded above by the step to

to the nearest bound, and hence a large value of ALFMAX does not usually affect the course of the optimization. The value of ALFMAX should be set to a smaller value than the default when it is desirable to try to force the optimization algorithm to converge to a solution that is close to the starting point. The default value of ALFMAX is 100.0.

The parameter MAXFUN is the maximum number of function evaluations that will be allowed in attempting to compute the minimum and the maximum of the function within a given region. The default value is 50n, where n is the number of variables. This value may be too small if the function is very badly behaved.

The value IPRINT indicates the frequency of printout during the optimization process. If IPRINT is negative (the default value), no intermediate printing will occur. If IPRINT is positive, printing will occur at every IPRINT-th iteration. If IPRINT is zero, printing will occur at the initial point and at the final iteration.

An additional control on the optimization procedure may be exercised through the parameters FLOBD and FUPBD, which are contained in the labelled COMMON block BNDLMT. FLOBD and FUPBD are intended to be known lower and upper bounds on the value of the function in the region. The default values are effectively minus and plus infinity (-9.9×10^{71} and 9.9×10^{71}). However, the effeciency of the optimization procedure can in some cases be significantly increased if better bounds are known. For example, consider a situation in which the objective function is a sum of squares, so that its value must be nonnegative (and hence FLOBD could be set to zero). If the function value is zero at any of the points in the initial random sample, then the given point is known to be a minimum, and no further function values need to be

expended in searching for a minimum.

A second way in which the user can improve the efficiency of the optimization involves the calculation of the function to be optimized. In some instances, the function is known to be extremely "flat" throughout much of the region, in the sense that the vlaues of the function are effectively constant (e.g., an exponential may assume values like 10^{-15} and 10^{-30}, which are for practical purposes identical). If the user-programmed subroutine checks for this situation and sets all such values to a constant, the optimization procedure will not need to expend function evaluations to determine the lowest point.

Gill, P.E., Murray, W. and Wright, M.H. (1981) Practical Optimization, Academic Press, London and New York.

Control of Splitting

The method used to subdivide the region of interest is described in Friedman and Wright (1981a). The user control of the splitting procedure occurs mainly through the labelled COMMON block CUTOLS. The parameters in CUTOLS are set to default values that have worked well in practice, but it may be possible to increase efficiency by adjusting their values.

The parameter BNDTOL, which lies in (0,1), can be viewed as expressing a tolerance for deciding when the cut along a given direction would not be significant. In particular, if the distance from a given component of the major extremum in a region to either of its bounds is less than BNDTOL times the distance bewteen the lower and upper bounds, no cut will be made in that direction from the extremum. The default value of BNDTOL is .05.

The value FRACT corresponds to the value α in equation (10) in Friedman and Wright (1981a). The initial trial cut along each direction is given by FRACT times the distance from the extremum to the corresponding bound. The default value of FRACT is 0.5.

The desired set of cuts corresponds to the solution of a system of nonlinear equations that represent the differences of function values. The parameters RGNTOL and FNLTOL control the termination of the iterative procedure that solves these nonlinear equations.

One test for terminating the procedure involves the desision that the regions corresponding to successive trial sets of cuts are sufficiently close in size. This test uses the value RGNTOL, which lies in (0,1), and its inverse. If the ratio of region sizes lies between RGNTOL and 1/RGNTOL, the procedure is considered to have converged. The default value of RGNTOL is 0.9.

The second test for termination is based in FNLTOL. The procedure terminates when the ratio of the maximum function difference to the maximum

function value is less than FNLTOL. The default value of FNLTOL is 0.1.

The logic of the splitting procedure is also controlled by two local variables (BIG and SINGTL) in the subrouting DELSLV. These variables are set in DATA statements, and may be latered by the user if appropriate. The value of BIG is used to provide a safeguard against unreasonably large numbers that might occur while iterating to solve the system of nonlinear equations. In particular, BIG provides an upper bound on the size of the elements in the triangular factors of the approximate Jacobian matrix, and on the value of each component of the vector of trial cuts. The default value of BIG is 10^{10}. The value SINGTL is a tolerance that is meant to define overly "small" diagonal elements in the approximate Jacobian matrix. The default value of SINGTL is 10^{-4}.

Storage Dumps

Two subroutines are available for dumping the contents of the ISTOR and RSTOR arrays in readable formats. These may be invoked after control returns from PARTN to obtain a detailed understanding of the resulting partitioning of the integration volume.

CALL TREDMP

causes the binary tree associated with the partitioning to be printed, one line for each nonterminal node, with the following format

NONTERMINAL NODE(I) = IM IL IR IX PRTN

IM = maximum terminal region number of left subtree of node I

IL = left son of node I

IR = right son of node I

(negative node numbers indicate terminal regions)

IX = coordinate upon which split was made at the Ith partitioning

PRTN = value of split point on IX-th coordinate

CALL BUKDMP(N)

causes the information concerning each hyperrectangular region to be printed. The argument N is the dimension of the integration variable space. This information includes the limits on each coordinate, $X_i(\ell)$ and $X_i(u)$, that define the region, along with the approximate quasi-uniform Monte Carlo estimate of the integral within the region, the spread measure squared for the region, the coordinate and split point for the next partition of the region, and the various quadrature formula integral estimates (if present, IDEG=2, 3 or 5). For IDEG=1 (importance sampling), several additional internally used quantities are printed.