Scenario and Stress Testing using *SAS/IML®*

Dmitry Donin, Dmytro Korol, Mykola Tkachenko

Bank of Montreal

April 07, 2011

Stress Testing Objectives

Stress Testing is aimed to capture extreme values of a model under consideration on unlikely scenarios.

- *Unlikely scenarios at level q* are the scenarios corresponding to the q-th quantile of the probability distribution of all scenarios.
- *A stressed value of a model at level q* is its extreme value when restricted to the set of unlikely scenarios at level q.

This approach is intrinsic to the model. No possible external drivers are considered.

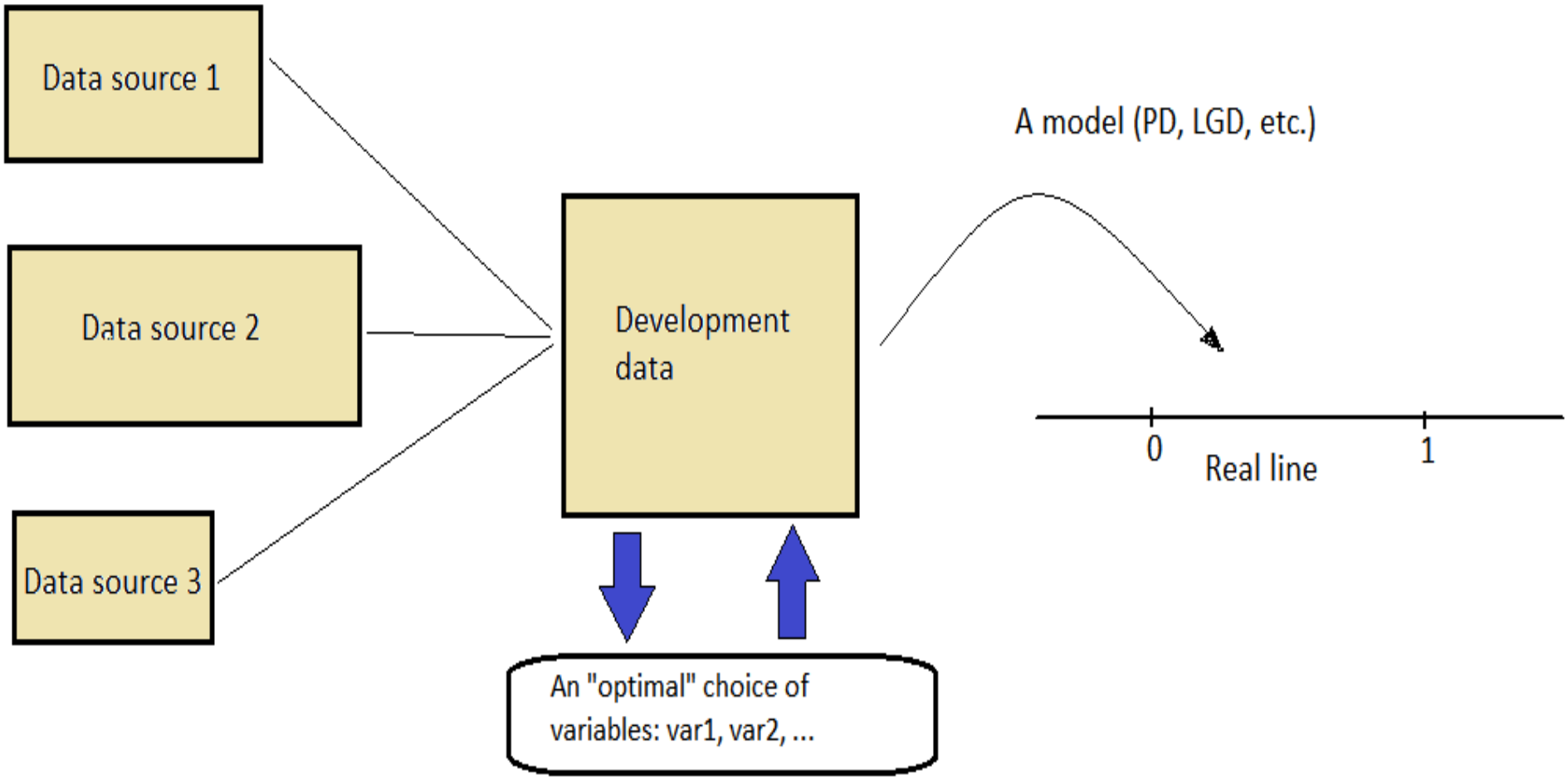
Main references

J. Berkowitz (2004), *A coherent framework for stress-testing*, In Jorion, P., editor, Innovations in Risk Management: Seminal Papers from the Journal of Risk. Risk Books.

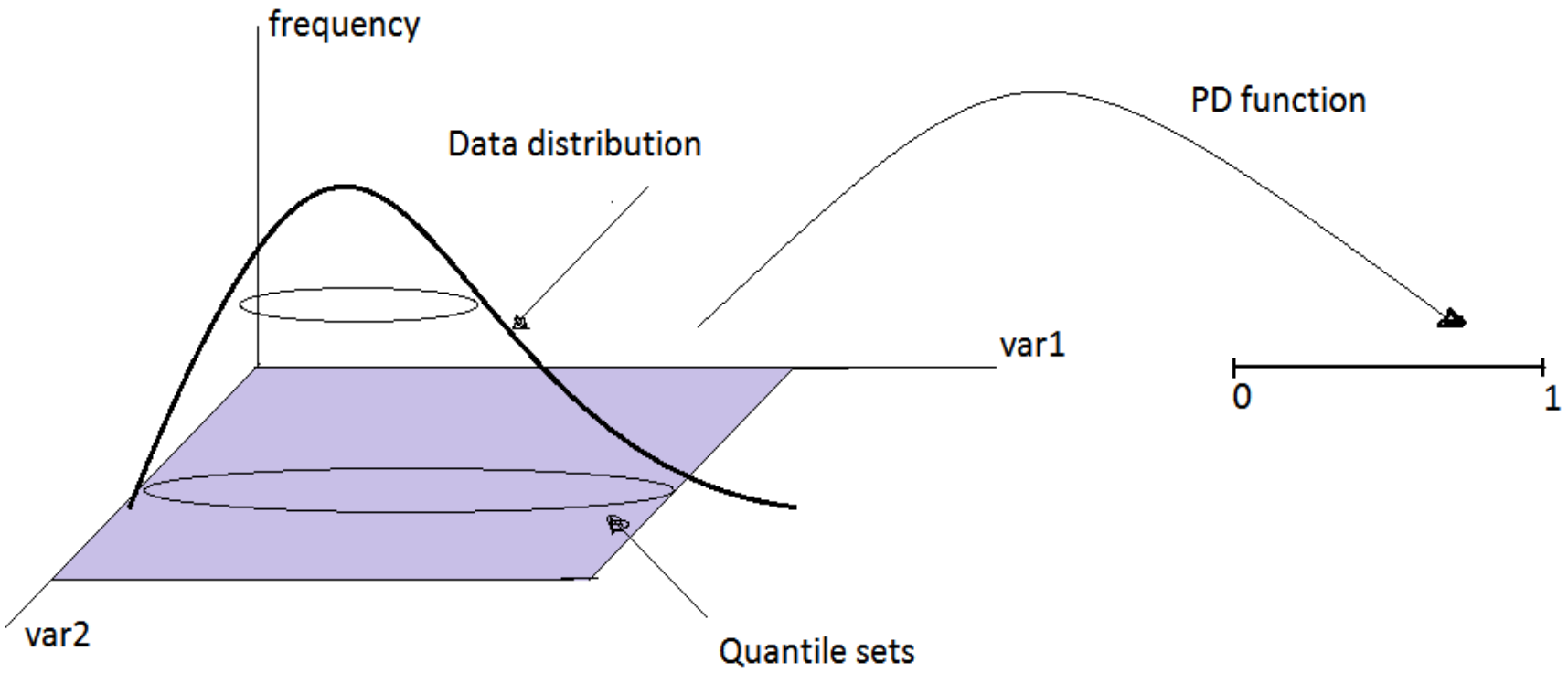
A. J. McNeil, A. D. Smith (2010), *Multivariate Stress Testing* for Solvency II.

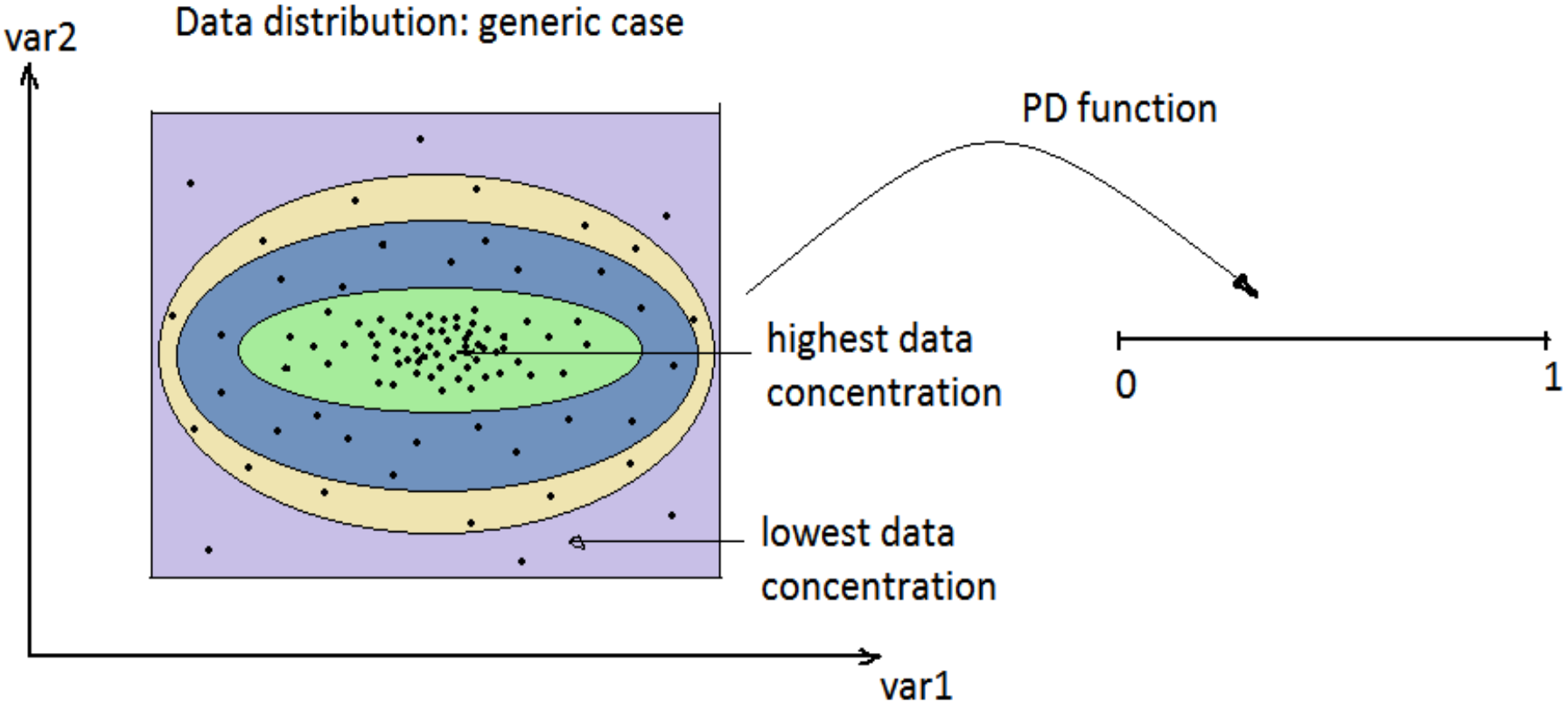
D. E. Goldberg (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Professional; 1 edition.

C. G. E. Boender, R. J. Caron, J. F. McDonald, A. H. G. Rinnooy Kan, H. E. Romeijn, R. L. Smith, J. Telgen, A. C. F. Vorst (1991), *Shake-and-bake algorithms for generating uniform points on the boundary of bounded polyhedra*, Operations Research, Volume 39, Issue 6, 945-954


Data source 1
Data source 2
Data source 3
Development data
A model (PD, LGD, etc.)
0
Real line
1
An "optimal" choice of variables: var1, var2, ...

Distribution of the development sample:

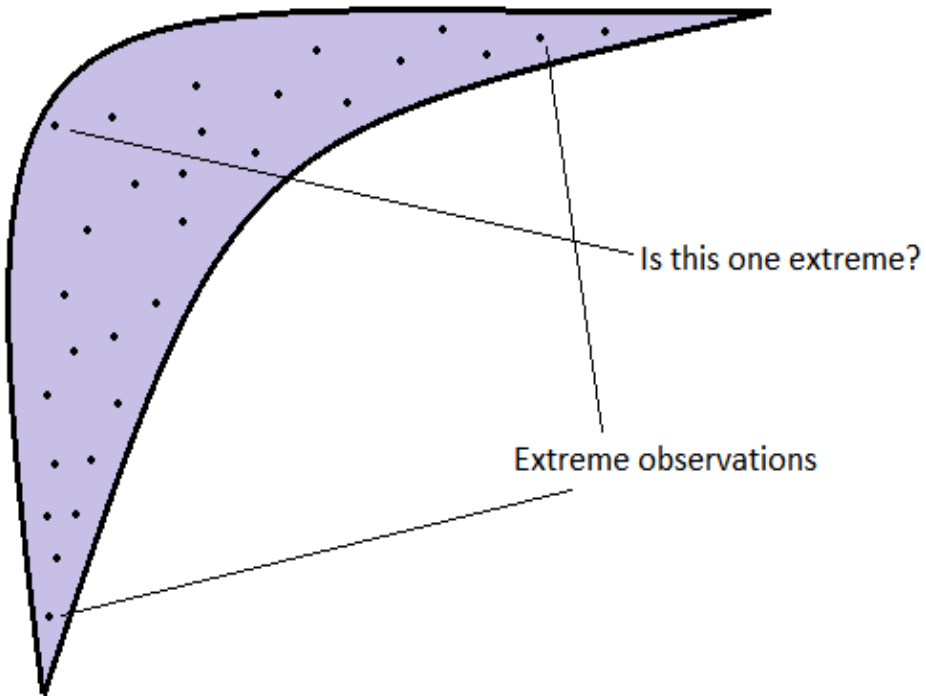




Generic case =

1) the "quantiles" of the data distribution are convex concentric subsets;
2) the PD function is monotonic along radial directions.

“Bad” data distribution: non-convex case



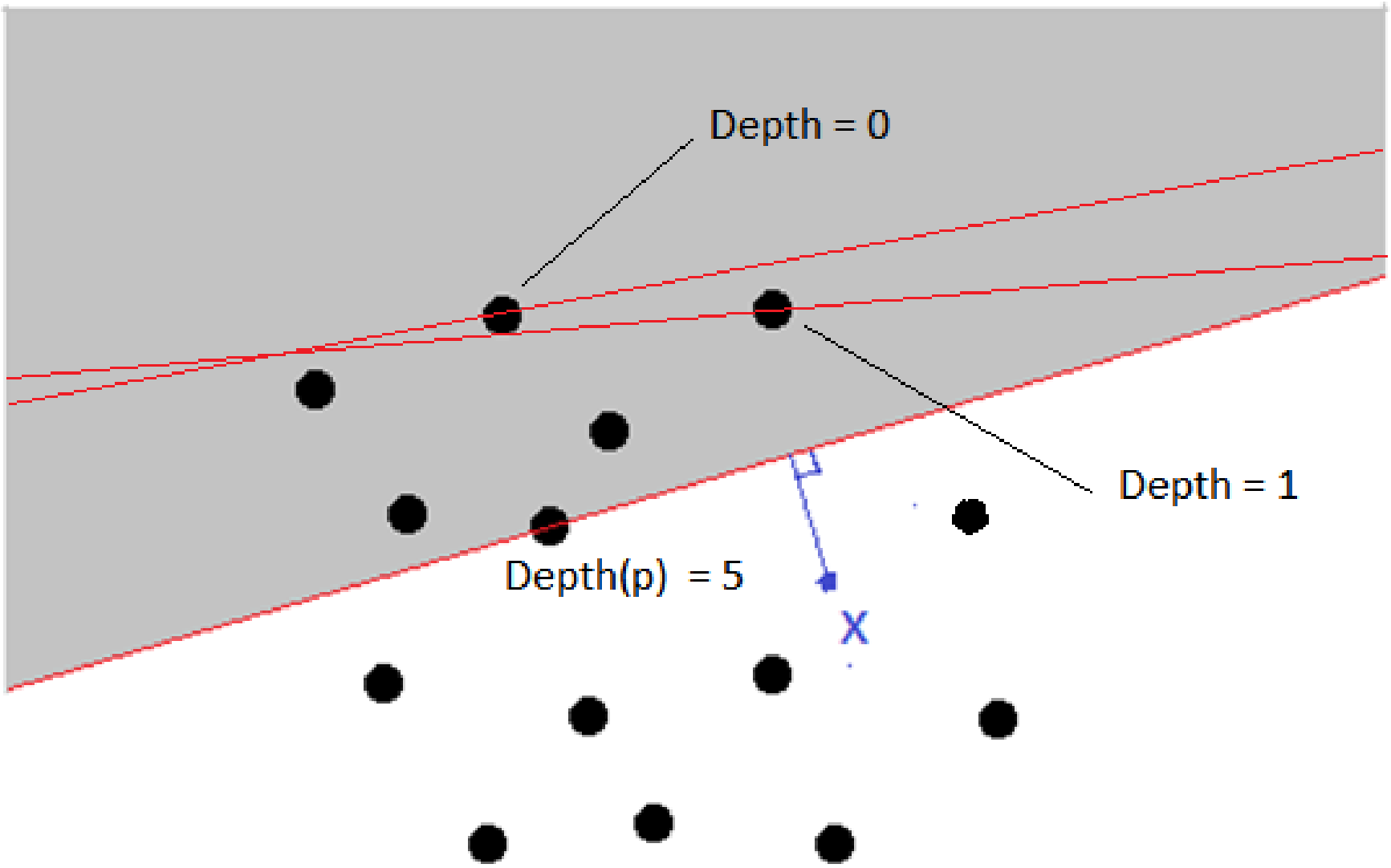
Problems associated with this approach

1. Probability distribution function of all scenarios is usually unknown.
2. For multivariate distributions the notion of quantile is not defined.
3. Finding extremes of a model with many variables can be very time consuming.
4. For complicated models such an approach may not give global extremes of the model. E.g. a model may achieve its global maxima on an event occurring with relatively high probability.

Half-space depth

To define extreme scenarios or quantiles of a multivariate distribution, we use the notion of half-space depth:

The *half-space depth* of a given point relative to a data set is the minimum number of points in the data set, lying in any closed half-plane determined by a line through this point:



$$\min_{x \in \mathbb{R}^d \setminus 0} |\{q \in S | \langle x, q \rangle \leq \langle x, p \rangle\}|$$

Our approach to Scenario and Stress Testing using *SAS/IML*®

1. Use '*proc model'* to generate data from probability distribution with known marginal distributions and known correlation matrix.
2. Use the notion of half-space depth to define quantiles of multivariate distributions.
3. Find extremes of the model using the following two algorithms:
   - Genetic Algorithm (implemented in *SAS/IML*)
   - A variation of Shake-and-Bake Algorithm

BMO Bank of Montreal

SAS Code – Generate Data for Procedure
1. Motivation

Data Generation is done to demonstrate the behavior and flexibility of the overall approach. In a business setting a natural data set would be available to act as a representative sample of the overall population.

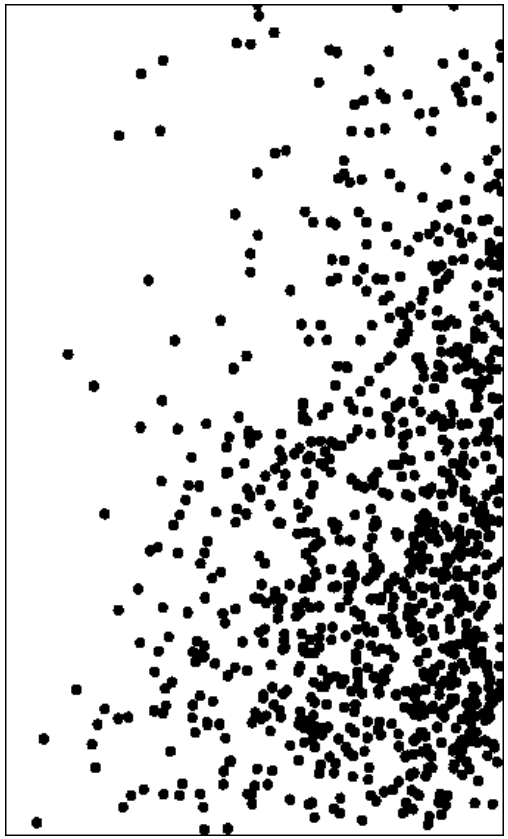
SAS Code – Generate Data for Procedure

2. Implementation

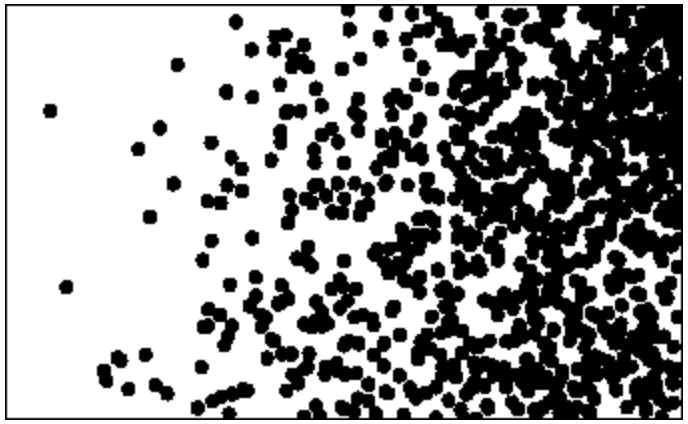
```
data s;
   _NAME_ = "aa";
    aa = 1; ab = 0.1;
    output;
    _NAME_ = "ab";
    aa = 0.1; ab = 1;
    output;
run;
data all;
    y = 0;
    z = 0;
    output;
run;
%let nsim = 100;
proc model data=all ;
    aa = 0;
    ab = 0;
    errormodel aa ~ uniform(0,1);
    errormodel ab ~ uniform(0,1);
    solve aa ab  / random = &nsim  sdata = s out = sim_uni(where=(_rep_>0)) ;
run;
```

SAS Code – Generate Data for Procedure

3. Results



Generated Sample Data. Y is gamma, X is beta. Rho is 0.2



Generated Sample Data. Y is uniform, X is beta. Rho is 0.2

SAS Code – Normalized Order Statistics

1. Motivation

Two things can be said about each point in a sample: what value does the model take at that point and what is the relative location with respect to other points in the sample. To answer the first question we need to know the real variable values which define this point. To answer the second question it is enough to know the normalized order statistic of each variable.

Half-space depth is unaffected if the variables are transformed into order statistics. On the other hand, it is much easier to set up the Genetic Algorithm using normalized order statistics.

SAS Code – Normalized Order Statistics Part 1

2. Implementation

```
%macro constraint;
data tempVar;
set constVar;
run;
%do i = 1 %to &nvars;
          proc sort data=tempVar;
          by &&var&i.;

          data tempVar (drop = ret prevCount);
          retain ret prevCount;
          set tempVar;
          if _n_ ^= 1 then
                    if &&var&i. = ret then
                              ind&&var&i. = prevCount;
                    else
                              ind&&var&i. = prevCount+1;
          else
                    ind&&var&i. = 1;
          ret = &&var&i.;
          prevCount = ind&&var&i.;
          run;
```

SAS Code – Normalized Order Statistics Part 2

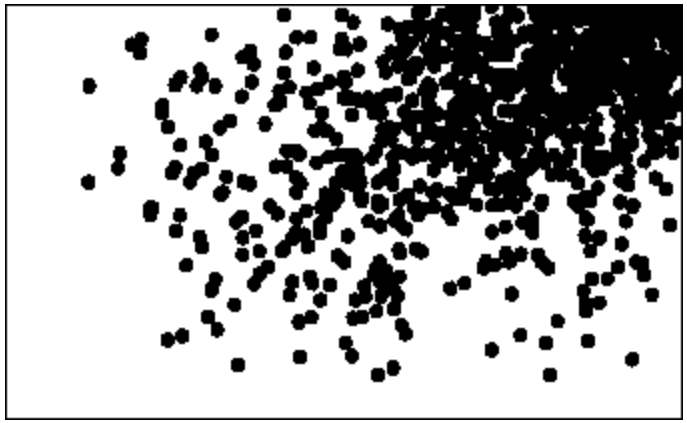
2. Implementation

```
            proc sql;
            create table distvar&i. as select distinct(&&var&i.) as &&var&i. from constVar;

            proc sort data=distvar&i.;
            by &&var&i.;
%end;

proc sql;
create table indVar as select ind&var1/max(ind&var1) as &var1
%do i = 2 %to &nvars;, ind&&var&i./max(ind&&var&i.)as &&var&i. %end;
from tempVar;

quit;
%mend constraint;
%constraint;
```

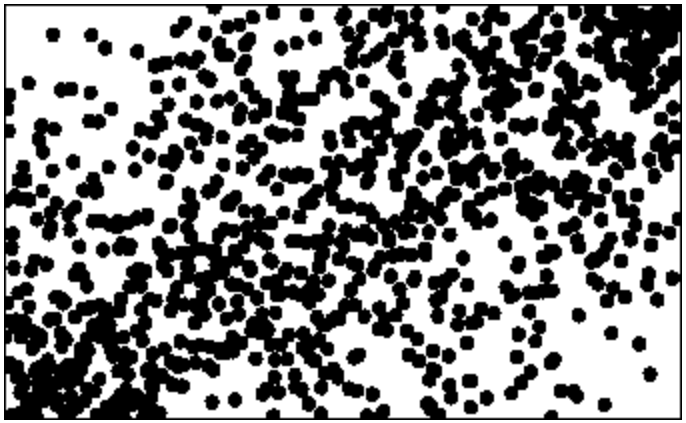
BMO Bank of Montreal

SAS Code – Normalized Order Statistics Part 1

3. Results



Raw Values



Normalized Order Statistics

SAS Code – Proc IML

```
proc iml;
use indVar;
read all into rankedVariables[colname=name];
numberRecords = nrow(rankedVariables);
%do i = 1 %to &nvars;
          use distvar&i.;
          read all into distinctVarValue&i[colname=&&var&i.];
          maxVarCounts = maxVarCounts||nrow(distinctVarValue&i);
%end;
pointForDepth = 0;
bestPlanecoeffs= 0;
constantForPlane = 0;
```

Need to turn SAS data sets into IML matrices. Matrix rankedVariables contains order statistics while matrices distvar1 and higher contain actual variable values

SAS Code – Approximate Half-Space Depth

1. Motivation

Calculating half-space depth exactly requires complicated algorithms and becomes extremely slow for even a moderate number of variables. An alternative is to use the genetic algorithm feature of IML to approximate the half-space depth. This approach will require two functions. The first function will calculate the number of points in one half-space and return this value. The second function will try to use the genetic algorithm to minimize the value of the first function, thus calculating the half-space depth of a given point.

SAS Code – Count Points in Half-space

2. Implementation

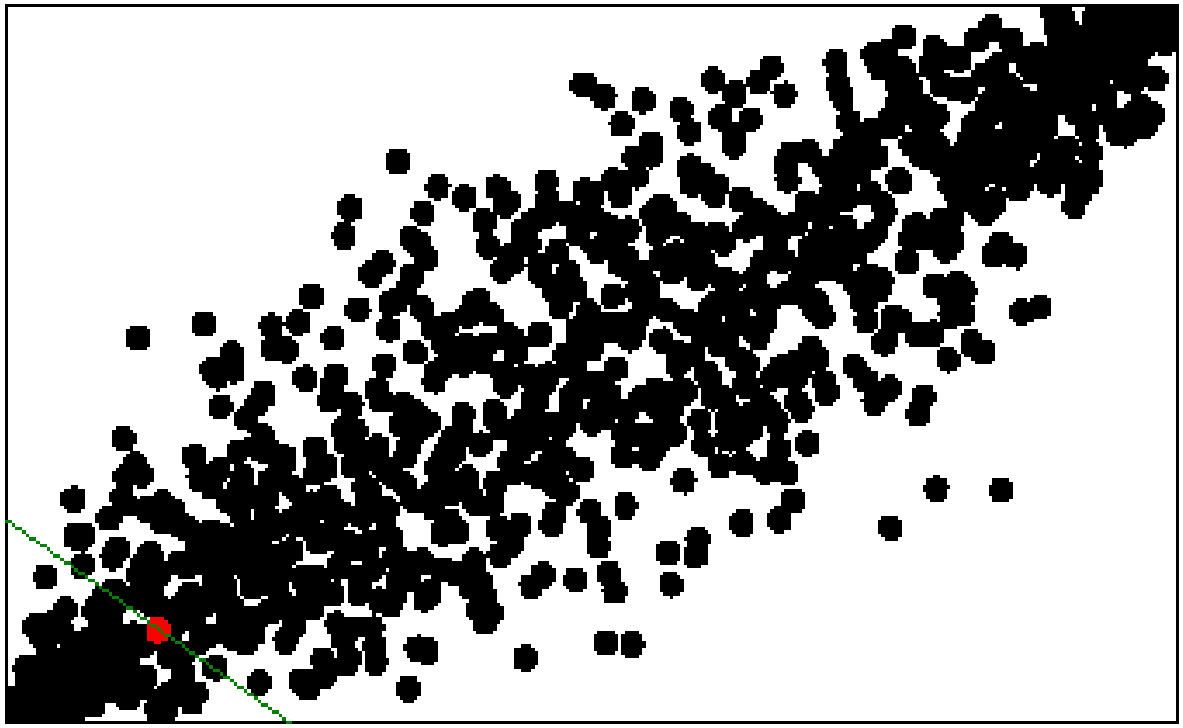
```
start halfspaceCount (planecoeffs) global (rankedVariables, numberRecords,
maxVarCounts, pointForDepth);

     if (planecoeffs*planecoeffs`) = 0 then
               return (numberRecords);
     constant=(pointForDepth / maxVarCounts)*planecoeffs`;
     diff = rankedVariables*planecoeffs` -  J(numberRecords,1,constant);
     halfspaceCount = 0;
     halfspaceCount = ncol(loc(abs(diff)+diff))+(numberRecords - ncol(loc(diff)));

     return (halfspaceCount);
finish halfspaceCount;
```

SAS Code – Approximate Half-Space Depth
2. Implementation

```
start halfspaceDepth(point) global(rankedVariables,numberRecords, maxVarCounts,
 pointForDepth, bestPlanecoeffs, constantForPlane);

          pointForDepth = point;
          id = gasetup(1,&nvars ,0);
          call gasetobj(id,0,"halfspaceCount");
          call gasetcro(id,1,1);
          call gasetmut(id,0.05,1);
          call gasetsel(id,10,1,1);
          call gainit(id,100,J(1,&nvars,-1)//J(1,&nvars,1));

          iterations = 10;
          do i = 2 to iterations;
                    call garegen(id);
          end;

          call gagetmem(bestMember, depth, id, 1);
          bestPlanecoeffs = bestMember/sqrt(ssq(bestMember));
          constantForPlane = (pointForDepth / maxVarCounts)*bestPlanecoeffs`;

          return (depth/numberRecords);
finish halfspaceDepth;
```

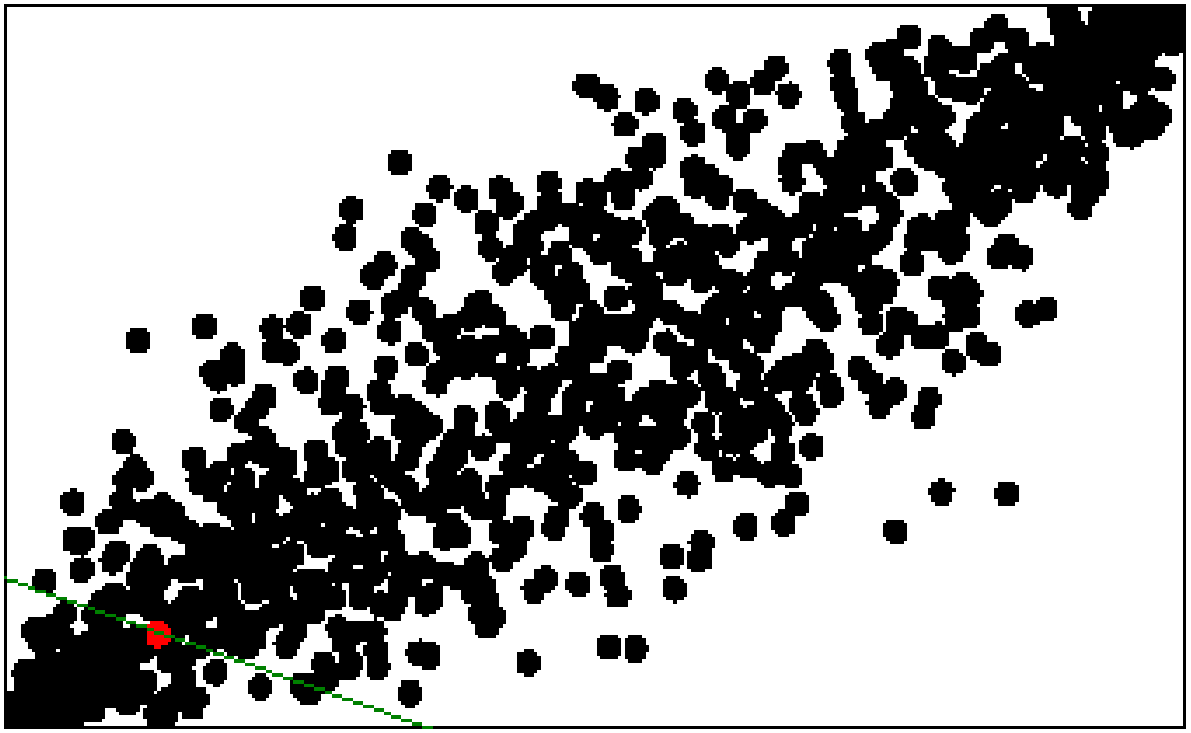
SAS Code – Approximate Half-Space Depth

3. Results



SAS Code – Approximate Half-Space Depth

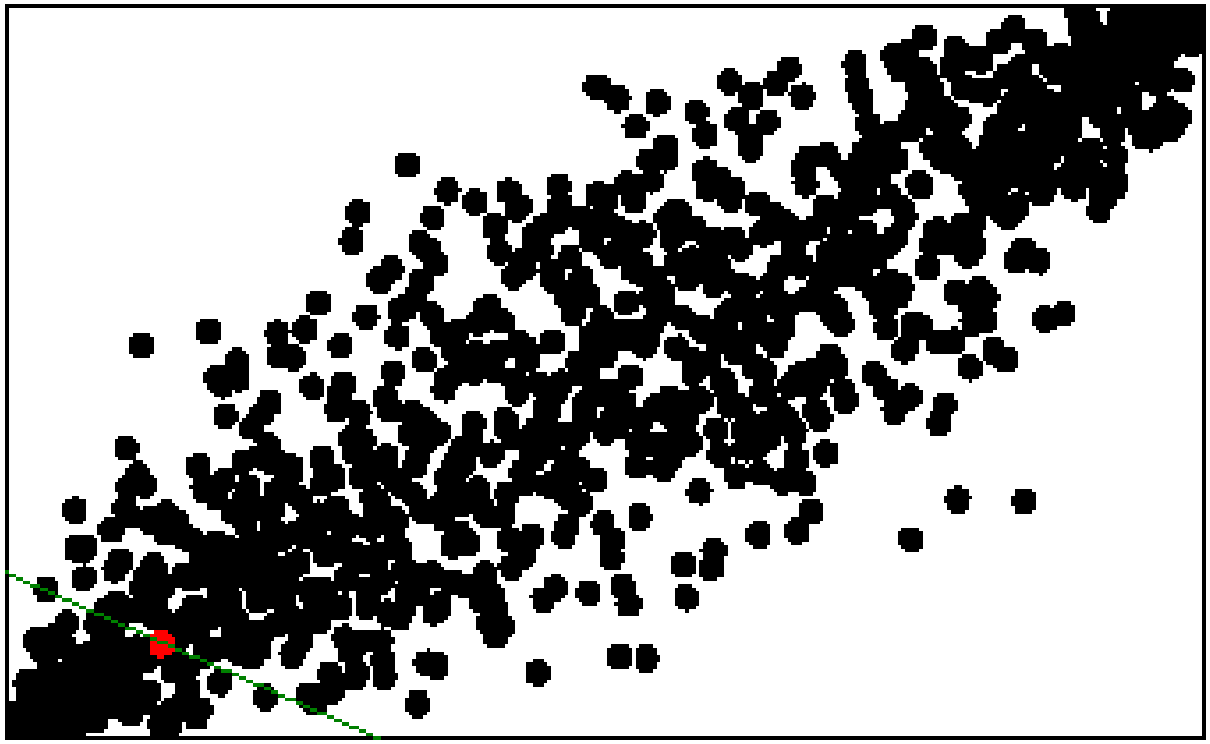
3. Results



BMO Bank of Montreal

SAS Code – Approximate Half-Space Depth

3. Results



BMO Bank of Montreal

SAS Code – Setting Up Shake-and-Bake

1. Methodology

Once all the planes at a given depth are available they need to be sampled to find the point which is located close to a given quantile and which causes the model to achieve either a maximum or a minimum. To do this the Shake-and-Bake algorithm is implemented in IML.

To get an intuitive picture of the Shake-and-Bake algorithm imagine a ball bouncing off the walls of a room. Every time the ball hits a wall we check the value of the model at that point and see how it compared to the previous points we had.

SAS Code – Setting Up Shake-and-Bake

2. Implementation

```
start PD(a) global(distinctVarValue1 %do i = 2 %to &nvars;,distinctVarValue&i %end;);
          return (distinctVarValue1[max(a[1,1],1)]
          %do i = 2 %to &nvars;+distinctVarValue&i[max(a[1,&i],1)] %end;);
finish PD;

do i = 1 to numberRecords;
          if abs(&quantile -halfspaceDepth(ceil(rankedVariables[i,]#maxVarCounts)))/
          &quantile < 0.1 then do;
                    equiDepth = equiDepth//i;
                    planes = planes//bestPlanecoeffs;
                    constants = constants//constantForPlane;
          end;
end;

do i = 1 to &nvars;
          k = J(1,&nvars,0);
          k[i] = 1;
          planes = planes//k;
          constants = constants//1;
          k[i] = -1;
          planes = planes//k;
          constants = constants//0;
end;
```

BMO Bank of Montreal

SAS Code – Shake and Bake

2. Implementation

```
if (nrow(equiDepth) >= 2*&nvars) then do;
          x = rankedVariables[equiDepth[1],];
          c = 1;
          do i = 1 to 1000;
                    u = normal(repeat(0,1,&nvars));
                    do while (ssq(u) = 0);
                              u = normal(repeat(0,1,&nvars));
                    end;
                    u = u / sqrt(ssq(u));
                    r = uniform(0);
                    v = r/sqrt(1-(planes[c,]*u`)**2)*u-(r*(planes[c,]*u`)/sqrt(1-
                    (planes[c,]*u`)**2)+sqrt(1-r**2))*planes[c,];
                    l =  (constants - planes*x`)/(planes*v`);
                    ind = 1;
          do j = 2 to nrow(l);
                    if (l[ind] <= 10**-14 | (l[j] < l[ind] & l[j] > 10**-14)) then
                              ind = j;
          end;
```

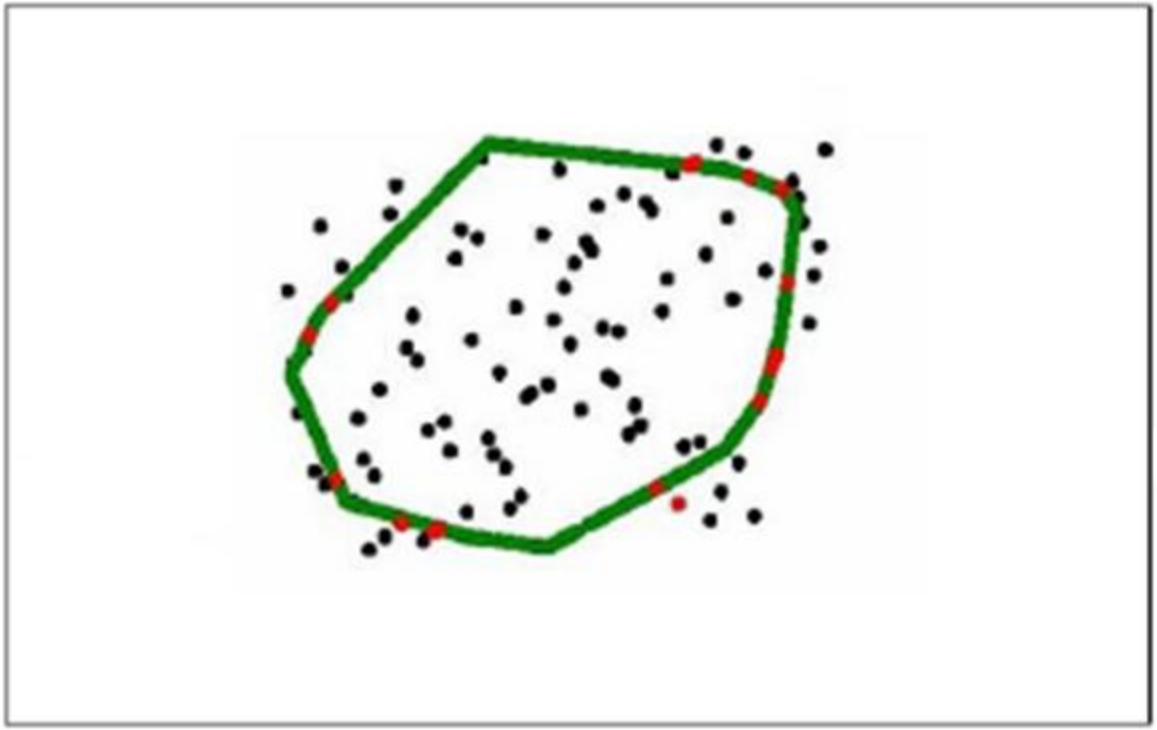
SAS Code – Show Results and Exit Proc IML

2. Implementation

```
            c = ind;
            totx = (x||PD(ceil(x#maxVarCounts)))//totx;
            x = x + l[ind]*v;
            end;
            call sortndx(ind,totx,{%eval(&nvars+1)},{%eval(&nvars+1)});
            print "The highest PD found is:"(PD(ceil(totx[ind[1],1:&nvars]#maxVarCounts)));
            print (halfspaceDepth(ceil(totx[ind[1],1:&nvars]#maxVarCounts)));
            print (ceil(totx[ind[1],1:&nvars]#maxVarCounts));
            call gstart;
            call gwindow({-1 -1  2 2});
            call gpoint(rankedVariables[,1],rankedVariables[,2],"dot","black");
            call gpoint(totx[,1],totx[,2],"dot","green");
            call gpoint(rankedVariables[equiDepth,1],rankedVariables[equiDepth,2],"dot","re
            call gshow;
            call gstop;
end;
quit;
```

BMO Bank of Montreal

Plot of Results



Depth=5 contour. The result of the combination of Genetic and Shake-and-Bake algorithms on a set of 100 data points

Questions?

BMO Bank of Montreal