Appendix for "Risk Quantization by Magnitude and Propensity"

by Olivier P. Faugeras and Gilles Pagès.

Some Mathematica code:

1. Empirical computation of (m_X, p_X) by Lloyd's Method

```
  (*Input: data= dataset;
          x0=starting value of the threshold search (take the mean
          of the data set for  example)
    Ouput: {m_X,p_X}
  *)
    mplloyd[data_, x0_] :=
    Module[{a},
        iteratefunction[a_] := Mean[Select[data, # > a &]]/2;
        a = FixedPoint[iteratefunction[#] &, x0];
        {2 a, NProbability[x > a, x \[Distributed]
        EmpiricalDistribution[data]]}
          ]
```

2. Multilevel quantization by direct Minimization

```
(* objective function *)
    q2[ma_, mb_, data_] :=
 Mean[MapThread[
   Min, {SquaredEuclideanDistance[#, ma] & /@ data,
    SquaredEuclideanDistance[#, mb] & /@ data,
    SquaredEuclideanDistance[#, 0] & /@ data}]]

(* create Lognormal data *)
dataone = Exp[RandomVariate[NormalDistribution[], 1000]];

(* compute propensities*)
Clear[solb, ma, mb]
solb = NMinimize[{q2[ma, mb, dataone], ma < mb}, {ma, mb}]
magnitudesb = Prepend[Values[solb[[2]]], 0](* computes magnitudes *)

Clear[distdatamagnitude, output];
distdatamagnitude =
  Table[SquaredEuclideanDistance[dataone[[i]], magnitudesb[[j]]], {i,
    1, Length[dataone]}, {j, 1,
    Length[magnitudesb]}]; (* this creates a matrix of the squared \
euclidean distance of the datapoints to the magnitude centers*)
```

```
output = {};(* initialization. *)
For[i = 1, i < Length[dataone] + 1, i++,
 AppendTo[output,
  Flatten[Position[
    distdatamagnitude[[i,
      All]], _?(# <= Min[distdatamagnitude[[i, All]]] &)]]]
 ] (*output is the list of indices of the smallest distance to center \
*)
output = Flatten[output];
pxb = Table[Count[output, i], {i, 1, 3}]/Length[dataone] // N
```

3. Multivariate quantization by direct Minimization

```
(* objective function to be minimized *)
   (*input: a,b are the m_X coordinates, data the empirical data to be \
applied to
 it creates pure functions in order to apply them to lists
then, Mapthread the Min between these lists,
 then takes the mean (empirical expectation) *)
    qmulti[a_, b_, data_] :=
 Mean[MapThread[
   Min, {SquaredEuclideanDistance[#, {a, b}] & /@ data,
    SquaredEuclideanDistance[#, {0, 0}] & /@ data}]]

*(compute magnitudes *)
magnitudes[
   sol_] := (Last[sol] /. Rule -> List )[[All,
    2]] (* extract magnitudes from solution*);

(* compute propensities *)
inside[mx_, my_, data_] :=
 Select[data,
  EuclideanDistance[#, {0, 0}] > EuclideanDistance[#, {mx, my}] &]
propensity[data_, mx_, my_] :=
 Length[inside[mx, my, data]]/Length[data] // N

(* example of running code *)
Clear[data3, sol3, mx3, my3, plot3, px3]
data3 = RandomVariate[
   CopulaDistribution[
    "Maximal", {UniformDistribution[{0, 1}],
     UniformDistribution[{0, 1}]}], 5000];
sol3 = NMinimize[qmulti[a, b, data3], {a, b}] (* compute solution *)
{mx3, my3} = magnitudes[sol3]
px3 = propensity[data3, mx3, my3]
```

4. Portfolio magnitude-propensity profile:

```
q[m_, data_] :=
    Mean[MapThread[Min,{SquaredEuclideanDistance[#, m] & /@ data,
     SquaredEuclideanDistance[#, 0] & /@ data}]];
propensityunivariate[data_, mx_]:=Length[
    Select[data, EuclideanDistance[#, {0, 0}] > EuclideanDistance[#, mx] &]]/ Length[da
    Clear[dataport, solport, mxport, pxport, risk]
dataport =
  RandomVariate[
   CopulaDistribution[{"Multinormal", {{1, 0.5}, {0.5,
       1}}}, {GammaDistribution[1.5, 2], GammaDistribution[0.5, 2]}],
   5000];
mxport = {};
pxport = {};

For[i = 0, i <= 20, i++,
 alpha = 0.05 {i, 20 - i};
 datain = dataport . alpha;
 solport = NMinimize[q[a, datain], {a}] (* compute solution *);
 AppendTo[mxport, magnitudes[solport]];
 AppendTo[pxport, propensityunivariate[datain, mxport]];
 ]
risk = Partition[Riffle[Flatten[mxport], pxport], 2]
```