

```

1 local R = require
2 local _,the,COLS,BIN = R"lib", R"the", R"COLS", R"BIN"
3 local map,sort,upl,items,push,norm = _.map,_.sort,_.upl,_.items,_.push,_.norm
4 local items,slice,o,co,sort,many = _.items,_.slice,_.o,_.co,_.sort,_.many
5 local class,OBJ = _.class, _.OBJ
6
7 local EGS = class("EGS",OBJ)
8 function EGS:new() self.rows, self.cols = {}, nil end
9
10 function EGS:adds(y) for x in items(y) do self:add(x) end; return self end
11
12 function EGS:add(row)
13   if not self.cols then self.cols = COLS(row)
14     else push(self.rows, self.cols:add(row)) end end
15
16 function EGS:mid(cols)
17   local mid=function(col) return #self.rows==0 and 0 or col:mid() end
18   return map(cols or self.cols.y, mid) end
19
20 function EGS:div(cols)
21   local div=function(col) return #self.rows==0 and 0 or col:div() end
22   return map(cols or self.cols.y, div) end
23
24 function EGS:clone(rows)
25   local out = EGS{}
26   out:add(self.cols.names)
27   for _,row in pairs(rows or {}) do out:add(row) end
28   return out end
29
30 function EGS:dist(row1,row2)
31   local d, n = 0, 0
32   for _,col in pairs(self.cols.x) do
33     n = n + 1
34     d = d + col:dist(row1[col.at], row2[col.at])^the.p end
35   return (d/n) ^ (1/the.p) end
36
37 function EGS:better(row1,row2)
38   local s1, s2, n, e = 0, 0, #self.cols.y, math.exp(1)
39   for _,col in pairs(self.cols.y) do
40     local a = norm(col.lo, col.hi, row1[col.at] )
41     local b = norm(col.lo, col.hi, row2[col.at] )
42     s1 = s1 - e^(col.w * (a - b) / n)
43     s2 = s2 - e^(col.w * (b - a) / n) end
44   return s1 / n < s2 / n end
45
46 function EGS:bins(other)
47   local out = {}
48   for n,left in pairs(self.cols.x) do
49     local right = other.cols.x[n]
50     local tmp = left:bins(right,BIN)
51     if #tmp > 1 then for _,bin in pairs(tmp) do push(out,bin) end end end
52   return out end
53
54 function EGS:bestRest()
55   self.rows = sort(self.rows, function(a,b) return self:better(a,b) end)
56   local n = (#self.rows)^the.best
57   return slice(self.rows, 1, n), -- top n things
58     many( self.rows, n^the.rest, n+1) end -- some sample of the rest
59
60 -- function egs.xplain(i)
61 --   best, rest = egs.bestRest(i)
62 --   return egs.constrasts(i, best,rest) end
63
64 return EGS

```