

Dec 19, 21 0:09

tiny.lua

Page 1/4

```

1 local the=require"tiny0"[[
2 lua hint.lua [OPTIONS]
3
4 A small sample multi-objective optimizer / data miner.
5 (c)2021 Tim Menzies <tim@ieee.org> unlicense.org
6
7 OPTIONS:
8 -best      X   Best examples are in 1.best*size(all)      = .05
9 -debug     X   run one test, show stackdumps on fail      = ing
10 -file      X   Where to read data                          = ../data/auto93.csv
11 -h         X   Show help                                    = false
12 -seed      X   Random number seed;                         = 10019
13 -Stop      X   Create subtrees while at least 2*stop egas = 4
14 -Tiny      X   Min range size = size(egs)*tiny             = .5
15 -todo      X   Pass/fail tests to run at start time        = ing
16
17 -epsilon   X   ignore differences under epsilon*stdev      = .35 ]]
18
19 --
20 -- 
21 local _require="tinylib"
22 local say,fmt,color,out,shout= _say,_fmt,_color,_out,_shout,_csv -- strings
23 local map,copy,keys,push = _map,_copy,_keys,_push -- tables
24 local sort, firsts, seconds = _sort,_firsts,_seconds -- sorting
25 local norm, sum = _norm,_sum -- maths
26 local randi,rand = _randi,_rand -- randoms
27 local same = _same -- meta
28 local csv = _csv -- files
29
30 local ent,mode
31 function ent(t, n,e)
32   n=0; for _,n1 in pairs(t) do n = n + n1 end
33   e=0; for _,n1 in pairs(t) do e = e + n1/n*math.log(n1/n,2) end
34   return e,n end
35
36 function mode(t, most,out)
37   most = 0
38   for x,n in pairs(t) do if n > most then most,out = n,x end end
39   return out end
40
41 --
42 -- 
43 local slurp,sample,ordered,clone
44 function slurp(out)
45   for eg in csv(the.file) do out=sample(eg,out) end
46   return out end
47
48 function clone(i, inits, out)
49   out = sample(i.heads)
50   for _,eg in pairs(inits or {}) do out = sample(eg,out) end
51   return out end
52
53 function sample(eg,i)
54   local numeric,independent,dependent,head,data,datum
55   i = 1 or {n=0,xs={},nys=0,ys={},num={},egs={},heads={},divs={}}
56   function head(n,x)
57     function numeric() i.num[n]= {hi=-math.huge,lo=math.huge} end
58     function independent() i.xs[n]= x end
59     function dependent()
60       i.num[n].w = x:find==" and -1 or 1
61       i.ys[n] = x
62       i.nys = i.nys+1 end
63     if not x:find==" then
64       if x:match"^[A-Z]" then numeric() end
65       if x:find==" or x:find"+" then dependent() else independent() end end
66     return x end
67   function data(eg) return {raw=eg, cooked=copy(eg)} end
68   function datum(n,x)
69     if x ~= "?" then
70       local num=i.num[n]
71       if num then
72         num.lo = math.min(num.lo,x)
73         num.hi = math.max(num.hi,x) end end
74     return x end
75   eg = eg.raw and eg.raw or eg
76   if #i.heads==0 then i.heads=map(eg,head) else push(i.egs,data(map(eg,datum))) end
77   i.n = i.n + 1
78   return i end
79
80 function ordered(i)
81   local function better(eg1,eg2, a,b,s1,s2)
82     s1,s2=0,0
83     for n,_ in pairs(i.ys) do
84       local num = i.num[n]
85       a = norm(num.lo, num.hi, eg1.raw[n])
86       b = norm(num.lo, num.hi, eg2.raw[n])
87       s1 = s1 - 2.71828*(num.w * (a-b)/i.nys)
88       s2 = s2 - 2.71828*(num.w * (b-a)/i.nys) end
89     return s1/i.nys < s2/i.nys end
90   for j,eg in pairs(sort(i.egs,better)) do
91     if j < the.best*#i.egs then eg.klass="best" else eg.klass="rest" end end
92     return i end
93
94
95
96



```

Dec 19, 21 0:09

tiny.lua

Page 2/4

```

97 -- 
98 -- 
99 local discretize, xys_sd, bin, div
100 function bin(z,divs)
101   if z=="?" then return "?" end
102   for n,x in pairs(divs) do
103     if x.lo<= z and z<= x.hi then return string.char(96+n) end end end
104
105 function discretize(i)
106   function xys_sd(col,egs, out,p)
107     out={}
108     for _,eg in pairs(egs) do
109       local x=eg.raw[col]
110       if x=="?" then push(out, {x=x, y=eg.klass}) end end
111     out = sort(out, function(a,b) return a.x < b.x end)
112     p = function(z) return out[z*#out//10].x end
113     return out, math.abs(p(.9) - p(.1))/2.56
114   end
115   for col,_ in pairs(i.xs) do
116     if i.num[col] then
117       local xys,sd = xys_sd(col, i.egs)
118       i.divs[col] = div(xys, (#xys)^the.Tiny, the.epsilon*sd)
119       for _,eg in pairs(i.egs) do
120         eg.cooked[col]= bin(eg.raw[col], i.divs[col]) end end end
121   return i end
122
123 function div(xys,tiny,epsilon, one,all,merged,merge)
124   c={}
125   function merged(a,b,an,bn, c)
126     for x,v in pairs(a) do c[x] = v end
127     for x,v in pairs(b) do c[x] = v + (c[x] or 0) end
128     if ent(c)*.99 <= (an*ent(a) + bn*ent(b))/(an+bn) then return c end
129   end
130   function merge(b4)
131     local j,tmp = 0,{}
132     while j < #b4 do
133       j = j + 1
134       local now, after = b4[j], b4[j+1]
135       if after then
136         local simpler = merged(now.has,after.has, now.n,after.n)
137         if simpler then
138           now = {lo=now.lo, hi=after.hi, n=now.n+after.n, has=simpler}
139           j = j + 1 end end
140       push(tmp,now) end
141     return #tmp==#b4 and b4 or merge(tmp) -- recurse until nothing merged
142   end
143   one = {lo=xys[1].x, hi=xys[1].x, n=0, has={}}
144   all = {one}
145   for j,xy in pairs(xys) do
146     local x,y = xy.x, xy.y
147     if j< #xys-tiny and x== xys[j+1].x and one.n> tiny and one.hi-one.lo>epsilon
148     then one = push(all, {lo=one.hi, hi=x, n=0, has={}})
149     end
150     one.n = 1 + one.n
151     one.hi = x
152     one.has[y] = 1 + (one.has[y] or 0); end
153   return merge(all) end
154
155

```

Dec 19, 21 0:09

tiny.lua

Page 3/4

```

155 -- T' (, , )
156 --
157
158 local splitter, worth, tree, count, keep, tree
159
160 function count(t, at) t=t or {}; t[at]=1+(t[at] or 0); return t end
161 function keep(t, at, x) t=t or {}; t[at]=t[at] or {}; push(t[at], x); return t end
162
163
164 function splitter(xs, eggs)
165     function worth(at, _, xy, n, x, xpect)
166         xy, n = {}, 0
167         for _, eg in pairs(egs) do
168             x = eg.cooked[at]
169             if x ~= "?" then
170                 n=n+1
171                 xy[x] = count(xy[x] or {}, eg.klass) end end
172             return (at, sum(xy, function(t) local e, n1=ent(t); return n1/n* e end)) end
173         return sort(map(xs, worth), seconds)[1][1] end
174
175 function tree(xs, eggs)
176     local here, at, splits, counts
177     for _, eg in pairs(egs) do counts=count(counts, eg.klass) end
178     here = {mode=mode(counts), n=#egs, kids={}}
179     if #egs > 2*the.Stop then
180         at = {}, splitter(xs, eggs)
181         for _, eg in pairs(egs) do splits=keep(splits, eg.cooked[at], eg) end
182         for val, split in pairs(splits) do
183             if #split < #egs then
184                 push(here.kids, {at=at, val=x, sub=tree(xs, split)}) end end end
185     return here end
186
187 -- function show(tree, pre)
188 --     pre = pre or ""
189 --     if tree.sub then
190 --         say("%s %s", pre)
191 --         for _, one in pairs(tree.sub) do
192 --             say("%s %s=%s", pre, one.at or "", one.val or "")
193 --             show(one.sub, pre.."|.. ") end end
194 --     else x end end
195
196

```

Dec 19, 21 0:09

tiny.lua

Page 4/4

```

196 -- T' (, , )
197 --
198
199 local go={}
200 function go.the() shout(the) end
201 function go.bad( s ) assert(false) end
202 function go.ing() return true end
203 function go.ordered( s, n)
204     s = ordered(slurp())
205     n = #s.egs
206     shout(s.heads)
207     for i=1,15 do shout(s.egs[i].raw) end
208     print("##")
209     for i=n,n-15,-1 do shout(s.egs[i].raw) end end
210
211 function go.bins( s )
212     s= discretize(ordered(slurp()))
213     for m, div in pairs(s.divs) do
214         print("")
215         for n, div1 in pairs(div) do print(m, n, out(div1)) end end
216     shout(s.egs[1])
217 end
218
219 -- Start tip
220 --
221 --
222 --
223 the(go)

```

Dec 19, 21 0:12

tinylib.lua

Page 1/1

```

1 local lib={}
2
3 --
4 -- Strings
5
6 lib.fmt = string.format
7 function lib.say(...) print(lib.fmt(...)) end
8 function lib.color(n,s) return lib.fmt("\27[1m\27[%sm%s\27[0m",n,s) end
9 function lib.shout(x) print(lib.out(x)) end
10
11 function lib.out(t, u,key,val)
12   function key(_,k) return string.format("%s %s", k, lib.out(t[k])) end
13   function val(_,v) return lib.out(v) end
14   if type(t) ~= "table" then return tostring(t) end
15   u = #t>0 and lib.map(t, val) or lib.map(lib.keys(t), key)
16   return {"..table.concat(u," ")}" end
17
18 --
19 -- Tables
20
21 function lib.push(t,x) t[1+#t]=x; return x end
22 function lib.copy(t, u) u={};for k,v in pairs(t) do u[k]=v end; return u end
23
24 function lib.map(t,f, u)
25   u,f={},f or same; for k,v in pairs(t) do u[1+#u] = f(k,v) end; return u end
26
27 function lib.keys(t,u)
28   u={}; for k,_ in pairs(t) do u[1+#u]=k end;return lib.sort(u);end
29
30 --
31 -- Sorting
32
33 function lib.sort(t,f) table.sort(t,f); return t end
34 function lib.firsts(x,y) return x[1] < y[1] end
35 function lib.seconds(x,y) return x[2] < y[2] end
36
37 --
38 -- Maths
39
40 function lib.norm(lo,hi,x)
41   return math.abs(lo-hi)<1E-32 and 0 or (x-lo)/(hi-lo) end
42
43 function lib.sum(t,f, n)
44   n,f=0,f or same; for _,v in pairs(t) do n = n + f(v) end; return n end
45
46 --
47 -- Random
48
49 function lib.randi(lo,hi) return math.floor(0.5 + rand(lo,hi)) end
50
51 function lib.rand(lo,hi)
52   lo, hi = lo or 0, hi or 1
53   the.seed = (16807 * the.seed) % 2147483647
54   return lo + (hi-lo) * the.seed / 2147483647 end
55
56 --
57 -- Maths
58
59 function lib.same(x,...) return x end
60
61 --
62 -- Files
63
64 function lib.csv(file, x)
65   file = io.input(file)
66   return function() t,tmp
67     x = io.read()
68     if x then
69       t={}
70       for y in x:gsub("[\n]","",):gmatch("[^\n]+") do t[1+#t]=tonumber(y) or y end
71       x = io.read()
72       if #t>0 then return t end
73       else io.close(file) end end end
74
75 --
76 -- Return
77
78 return lib

```

Dec 19, 21 0:20

tiny0.lua

Page 1/1

```

1 -- standard load and start functions
2 -- first line of code should be a help string (e.g. see tiny.lua)
3 -- last line of code should call this code, pass in table of actions
4 -- e.g
5 --     the(go)
6
7 --
8 -- Rogues
9
10 -- at load time, remember the current globals
11 local b4={}; for k,_ in pairs(_ENV) do b4[k]=k end
12 -- after start time, complain if code has created rogue globals
13 local function rogues()
14   for k,v in pairs(_ENV) do if not b4[k] then print("?:",k,type(v)) end end end
15
16 --
17 -- Misc
18
19 -- Table keys, in sorted order
20 local function keys(t,u)
21   u={}; for k,_ in pairs(t) do u[1+#u]=k end; table.sort(u); return u end
22
23 -- pretty colors, n=(31,32),=(red,green)
24 local function color(n,s) return string.format("\27[1m\27[%sm%s\27[0m",n,s) end
25
26 -- shallow copy of a list
27 local function copy(t, u)
28   u={}; for k,v in pairs(t) do u[k]=v end ; return u end
29
30 --
31 -- Start-up
32
33 local help = ""
34
35 -- All the start-up actions:
36 -- [1] keep a copy of the options as "defaults"
37 -- [2] maybe just show the help text
38 -- [3] maybe run an action in verbose mode (show stackdump; halt on error)
39 -- [4] before actions, reset options to defaults
40 -- [5] before actions, reset random number seed
41 -- [6] maybe run an action in fast mode (no stackdumps; no halts one errors)
42 -- [7] for fast mode, count the number of failures
43 -- [8] return to the operating system the count of failures
44 -- [9] lint the code (right now, we just print rogue globals)
45 local function what2doAtLastLine(options, actions)
46   local fails, defaults = 0, copy(options) -- [1]
47   if options.h then return print(help) end -- [2]
48   if options.debug then actions[options.debug]() end -- [3]
49   local todos = options.todos == "all" and keys(actions) or {options.todos}
50   for _todo in pairs(todos) do
51     if type(actions[_todo]) ~= "function"
52     then print(color(31,"NOFUN."),_todo)
53     else for k,v in pairs(defaults) do options[k]=v end -- [4]
54     options.seed = options.seed or 10019 -- [5]
55     local ok,msg = pcall(actions[_todo]) -- [6]
56     if ok then print(color(32,"PASS ").._todo)
57     else print(color(31,"FAIL ").._todo,msg)
58     fails=fails+1 end end -- [7]
59   end
60   os.exit(fails) end -- [8]
61
62 --
63 -- Lint, Lint
64
65 -- In paragraph of the text that starts with "Options", all lines that start with
66 -- "flag" have a default value as the last word on that line.
67 -- [1] Build the "options" array from those flags and defaults
68 -- [2] Check if we can update those defaults from command line arguments).
69 -- [3] Anything on the command line is a string. Check if these can become nums
70 -- For the sake of brevity:
71 -- [4] command line flags need only match the start of the flag;
72 -- [5] for boolean values, -flag flips the default boolean
73 -- [6] add in the ability to call "what2doAtLastLine"
74 local function what2doAtFirstLine(txt)
75   local options={}
76   help = txt
77   txt:gsub("^.*OPTIONS:",":gsub(\"%n%s*~([^\n]+)[^\n]*%s([^\n]+)\",
78     function(flag,x)
79       for n,word in ipairs(arg) do -- [2]
80         if flag:match("^"..word:sub(2)..".*") then -- [4]
81           x=(x=="false" and "true") or (x=="true" and "false") or arg[n+1] end end
82         if x=="true" then x=true
83         elseif x=="false" then x=false -- [4]
84         else x=tonumber(x) or x -- [3]
85       end
86       options[flag] = x end -- [1]
87   return setmetatable(options,{__call=what2doAtLastLine}) end -- [6]
88
89 --
90 -- Return
91
92 return what2doAtFirstLine

```