

















|  |
| --- |
| $ontext  Cardinality constrained mean-variance portfolio model.  Branch-and-bound algorithm written in GAMS.  Data from an AMPL example by Robert Vanderbei,  Princeton University.  Erwin Kalvelagen, 2003  $offtext  set i 'asset categories' /  1 tbill 'US 3 month t-bills'  2 bonds 'US government long bonds'  3 sp 's&p 500'  4 wfiv 'wilshire 5000'  5 qqq 'Nasdaq composite'  6 lbcorp 'Lehman Brothers corporate bonds index'  7 eafe 'Morgan Stanley EAFE Index'  8 gold  /;  alias (i,j);  set t 'years' /1973\*1994/;  table r(t,i) 'returns'  t bill bonds sp wfiv qqq lbcorp eafe gold  1973 1.075 0.942 0.852 0.815 0.698 1.023 0.851 1.677  1974 1.084 1.020 0.735 0.716 0.662 1.002 0.768 1.722  1975 1.061 1.056 1.371 1.385 1.318 1.123 1.354 0.760  1976 1.052 1.175 1.236 1.266 1.280 1.156 1.025 0.960  1977 1.055 1.002 0.926 0.974 1.093 1.030 1.181 1.200  1978 1.077 0.982 1.064 1.093 1.146 1.012 1.326 1.295  1979 1.109 0.978 1.184 1.256 1.307 1.023 1.048 2.212  1980 1.127 0.947 1.323 1.337 1.367 1.031 1.226 1.296  1981 1.156 1.003 0.949 0.963 0.990 1.073 0.977 0.688  1982 1.117 1.465 1.215 1.187 1.213 1.311 0.981 1.084  1983 1.092 0.985 1.224 1.235 1.217 1.080 1.237 0.872  1984 1.103 1.159 1.061 1.030 0.903 1.150 1.074 0.825  1985 1.080 1.366 1.316 1.326 1.333 1.213 1.562 1.006  1986 1.063 1.309 1.186 1.161 1.086 1.156 1.694 1.216  1987 1.061 0.925 1.052 1.023 0.959 1.023 1.246 1.244  1988 1.071 1.086 1.165 1.179 1.165 1.076 1.283 0.861  1989 1.087 1.212 1.316 1.292 1.204 1.142 1.105 0.977  1990 1.080 1.054 0.968 0.938 0.830 1.083 0.766 0.922  1991 1.057 1.193 1.304 1.342 1.594 1.161 1.121 0.958  1992 1.036 1.079 1.076 1.090 1.174 1.076 0.878 0.926  1993 1.031 1.217 1.100 1.113 1.162 1.110 1.326 1.146  1994 1.045 0.889 1.012 0.999 0.968 0.965 1.078 0.990  ;  scalar  rho 'minimum return' /1.1/  maxasset 'max number of assets in portfolio' /5/  ;  parameters  mean(i) 'average returns'  rtilde(t,i) 'mean corrected return'  cov(i,j) 'covariance'  ;  mean(i) = sum(t,r(t,i)) / card(t);  rtilde(t,i) = r(t,i) - mean(i);  cov(i,j) = sum(t, rtilde(t,i)\*rtilde(t,j))/card(t);  variables  x(i) 'weights'  z 'objective variable'  y(i) 'indicator wether in portfolio'  w(t) 'auxiliary variable'  ;  positive variable x(i);  binary variable y(i);  equations  obj 'objective'  budget 'total weight = 1'  minreturn 'minimum return'  cardinality 'max number of assets in portfolio'  wdef(t) 'calculate w(t)'  indicator(i) 'if y(i)=0 => x(i)=0'  ;  obj.. z =e= sum(t, sqr(w(t)));  budget.. sum(i,x(i)) =e= 1;  minreturn.. sum(i, mean(i)\*x(i)) =g= rho;  cardinality.. sum(i, y(i)) =l= maxasset;  wdef(t).. w(t) =e= sum(i, x(i)\*rtilde(t,i));  indicator(i).. x(i) =l= y(i);  \*-------------------------------------------------------------  \* simple branch and bound algorithm  \*-------------------------------------------------------------  model m /all/;  option rminlp=conopt;  option limrow=0;  option limcol=0;  option solprint=off;  \*---------------------------------------------------------  \* datastructure for node pool  \* each node has variables x(i) which is either  \* still free, or fixed to zero or with a lowerbound  \* of minx.  \*---------------------------------------------------------  set node 'maximum size of the node pool' /node1\*node20/;  parameter bound(node) 'node n will have an obj <= bound(n)';  set fixedzero(node,i) 'variables y(k,j) are fixed to zero in this node';  set fixedone(node,i) 'variables y(k,j) are fixed to one in this node';  scalar bestfound 'lowerbound in B&B tree' /+INF/;  scalar bestpossible 'upperbound in B&B tree' /-INF/;  set newnode(node) 'new node (singleton)';  set waiting(node) 'waiting node list';  set current(node) 'current node (singleton except exceptions)';  parameter bblog(node,\*) 'logging information';  scalar done 'terminate' /0/;  scalar first 'controller for loop';  scalar first2 'controller for loop';  scalar subobj 'objective of subproblem';  scalar maxx;  set wait(node) 'waiting nodes';  parameter nodenumber(node);  parameter bestx(i) 'record best solution';  parameter besty(i) 'record best solution';  nodenumber(node) = ord(node);  fixedzero(node,i) = no;  fixedone(node,i) = no;  set fx(i) 'used in fixing variables';  alias (nd,node);  \*  \* add root node to the waiting node list  \*  waiting('node1') = yes;  current('node1') = yes;  newnode('node1') = yes;  bound('node1') =INF;  loop(node$(not done),  {  \*  \* node selection  \*  bestpossible = smin(waiting(nd), bound(nd));  current(nd) = no;  current(waiting(nd))$(bound(nd) = bestpossible) = yes;  first = 1;  \*  \* only consider first in set current  \*  loop(current$first,  {  first = 0;  bblog(node,'node') = nodenumber(current);  bblog(node,'lb') = bestpossible;  \*  \* remove current node from waiting list  \*  waiting(current) = no;  \*  \* clear bounds  \*  y.lo(i) = 0;  y.up(i) = 1;  \*  \* set appropriate bounds  \*  fx(i) = fixedzero(current,i);  y.fx(fx) = 0;  fx(i) = fixedone(current,i);  y.fx(fx) = 1;  display y.lo,y.up;  \*  \* solve subproblem  \*  solve m minimizing z using rminlp;  display y.l;  \*  \* check for optimal solution  \*  bblog(node,'solvestat') = m.solvestat;  bblog(node,'modelstat') = m.modelstat;  abort$(m.solvestat <> 1) "Solver did not return ok";  if (m.modelstat = 1 or m.modelstat = 2,  \*  \* get objective  \*  subobj = z.l;  bblog(node,'obj') = subobj;  \*  \* check "integrality"  \*  maxx = smax(i, min(y.l(i), 1-y.l(i)));  if (maxx = 0,  \*  \* found a new "integer" solution  \*  bblog(node,'integer') = 1;  if (subobj < bestfound,  \*  \* improved the best found solution  \*  bblog(node,'best') = 1;  bestfound = subobj;  \*  \* record best solution  \*  bestx(i) = x.l(i);  besty(i) = y.l(i);  \*  \* remove all waiting nodes with bound > bestfound  \*  wait(nd) = no; wait(waiting) = yes;  waiting(wait)$(bound(wait) > bestfound) = no;  );  else  \*  \* "fractional" solution  \*  fx(i) = no;  fx(i)$(min(y.l(i), 1-y.l(i))=maxx) = yes;  first2 = 1;  loop(fx$first2,  first2 = 0;  \*  \* create 2 new nodes  \*  newnode(nd) = newnode(nd-1);  fixedzero(newnode,i) = fixedzero(current,i);  fixedone(newnode,i) = fixedone(current,i);  bound(newnode) = subobj;  waiting(newnode) = yes;  fixedzero(newnode,fx) = yes;  newnode(nd) = newnode(nd-1);  fixedzero(newnode,i) = fixedzero(current,i);  fixedone(newnode,i) = fixedone(current,i);  bound(newnode) = subobj;  waiting(newnode) = yes;  fixedone(newnode,fx) = yes;  );  );  else  \*  \* if subproblem is infeasible this node is fathomed.  \* otherwise it is an error.  \*  abort$(m.modelstat <> 4 and m.modelstat <> 5)  "Solver did not solve subproblem";  );  bblog(node,'waiting') = card(waiting);  );  \*  \* are there new waiting nodes?  \*  done$(card(waiting) = 0) = 1;  \*  \* display log  \*  display bblog;  );  display bestx,besty,bestfound; |

The final version:

|  |
| --- |
| $ontext  Cardinality constrained mean-variance portfolio model.  Branch-and-bound algorithm written in GAMS.  Data from an AMPL example by Robert Vanderbei,  Princeton University.  Erwin Kalvelagen, 2003  $offtext  set i 'asset categories' /  tbill 'US 3 month t-bills'  bonds 'US government long bonds'  sp 's&p 500'  wfiv 'wilshire 5000'  qqq 'Nasdaq composite'  lbcorp 'Lehman Brothers corporate bonds index'  eafe 'Morgan Stanley EAFE Index'  gold  /;  alias (i,j);  set t 'years' /1973\*1994/;  table r(t,i) 'returns'  tbill bonds sp wfiv qqq lbcorp eafe gold  1973 1.075 0.942 0.852 0.815 0.698 1.023 0.851 1.677  1974 1.084 1.020 0.735 0.716 0.662 1.002 0.768 1.722  1975 1.061 1.056 1.371 1.385 1.318 1.123 1.354 0.760  1976 1.052 1.175 1.236 1.266 1.280 1.156 1.025 0.960  1977 1.055 1.002 0.926 0.974 1.093 1.030 1.181 1.200  1978 1.077 0.982 1.064 1.093 1.146 1.012 1.326 1.295  1979 1.109 0.978 1.184 1.256 1.307 1.023 1.048 2.212  1980 1.127 0.947 1.323 1.337 1.367 1.031 1.226 1.296  1981 1.156 1.003 0.949 0.963 0.990 1.073 0.977 0.688  1982 1.117 1.465 1.215 1.187 1.213 1.311 0.981 1.084  1983 1.092 0.985 1.224 1.235 1.217 1.080 1.237 0.872  1984 1.103 1.159 1.061 1.030 0.903 1.150 1.074 0.825  1985 1.080 1.366 1.316 1.326 1.333 1.213 1.562 1.006  1986 1.063 1.309 1.186 1.161 1.086 1.156 1.694 1.216  1987 1.061 0.925 1.052 1.023 0.959 1.023 1.246 1.244  1988 1.071 1.086 1.165 1.179 1.165 1.076 1.283 0.861  1989 1.087 1.212 1.316 1.292 1.204 1.142 1.105 0.977  1990 1.080 1.054 0.968 0.938 0.830 1.083 0.766 0.922  1991 1.057 1.193 1.304 1.342 1.594 1.161 1.121 0.958  1992 1.036 1.079 1.076 1.090 1.174 1.076 0.878 0.926  1993 1.031 1.217 1.100 1.113 1.162 1.110 1.326 1.146  1994 1.045 0.889 1.012 0.999 0.968 0.965 1.078 0.990  ;  scalar  rho 'minimum return' /1.1/  maxasset 'max number of assets in portfolio' /5/  ;  parameters  mean(i) 'average returns'  rtilde(t,i) 'mean corrected return'  cov(i,j) 'covariance'  ;  mean(i) = sum(t,r(t,i)) / card(t);  rtilde(t,i) = r(t,i) - mean(i);  cov(i,j) = sum(t, rtilde(t,i)\*rtilde(t,j))/card(t);  variables  x(i) 'weights'  z 'objective variable'  y(i) 'indicator wether in portfolio'  w(t) 'auxiliary variable'  ;  positive variable x(i);  binary variable y(i);  equations  obj 'objective'  budget 'total weight = 1'  minreturn 'minimum return'  cardinality 'max number of assets in portfolio'  wdef(t) 'calculate w(t)'  indicator(i) 'if y(i)=0 => x(i)=0'  ;  obj.. z =e= sum(t, sqr(w(t)));  budget.. sum(i,x(i)) =e= 1;  minreturn.. sum(i, mean(i)\*x(i)) =g= rho;  cardinality.. sum(i, y(i)) =l= maxasset;  wdef(t).. w(t) =e= sum(i, x(i)\*rtilde(t,i));  indicator(i).. x(i) =l= y(i);  \*-------------------------------------------------------------  \* simple branch and bound algorithm  \*-------------------------------------------------------------  model m /all/;  option rminlp=conopt;  option limrow=0;  option limcol=0;  option solprint=off;  \*---------------------------------------------------------  \* datastructure for node pool  \* each node has variables x(i) which is either  \* still free, or fixed to zero or with a lowerbound  \* of minx.  \*---------------------------------------------------------  set node 'maximum size of the node pool' /node1\*node1000/;  parameter bound(node) 'node n will have an obj <= bound(n)';  set fixedzero(node,i) 'variables y(k,j) are fixed to zero in this node';  set fixedone(node,i) 'variables y(k,j) are fixed to one in this node';  scalar bestfound 'lowerbound in B&B tree' /+INF/;  scalar bestpossible 'upperbound in B&B tree' /-INF/;  set newnode(node) 'new node (singleton)';  set waiting(node) 'waiting node list';  set current(node) 'current node (singleton except exceptions)';  parameter bblog(node,\*) 'logging information';  scalar done 'terminate' /0/;  scalar first 'controller for loop';  scalar first2 'controller for loop';  scalar subobj 'objective of subproblem';  scalar maxx;  set wait(node) 'waiting nodes';  parameter nodenumber(node);  parameter bestx(i) 'record best solution';  parameter besty(i) 'record best solution';  nodenumber(node) = ord(node);  fixedzero(node,i) = no;  fixedone(node,i) = no;  set fx(i) 'used in fixing variables';  alias (nd,node);  \*  \* add root node to the waiting node list  \*  waiting('node1') = yes;  current('node1') = yes;  newnode('node1') = yes;  bound('node1') =INF;  loop(node$(not done),  \* if all elements of set "node" is not same with the value of 'done', the loop can continue.  \*  \* node selection  \*  bestpossible = smin(waiting(nd), bound(nd));  current(nd) = no;  current(waiting(nd))$(bound(nd) = bestpossible) = yes;  first = 1;  \*  \* only consider first in set current  \*  loop(current$first,  first = 0;  bblog(node,'node') = nodenumber(current);  bblog(node,'lb') = bestpossible;  \*  \* remove current node from waiting list  \*  waiting(current) = no;  \*  \* clear bounds  \*  y.lo(i) = 0;  y.up(i) = 1;  \*  \* set appropriate bounds  \*  fx(i) = fixedzero(current,i);  y.fx(fx) = 0;  fx(i) = fixedone(current,i);  y.fx(fx) = 1;  display y.lo,y.up;  \*at the second iteration, fx(i)=fixedzero(node2,i), so f(i)only has the fixedzero(node2,'tbill'). y.fx(node2,'tbill')=0.  \* fx(i)=fixed(node2,i). the set is empty, so y.fx(node2,i)=1 is impossible.  \*  \* solve subproblem  \*  solve m minimizing z using rminlp;  display x.l;  display y.l;  \*  \* check for optimal solution  \*  bblog(node,'solvestat') = m.solvestat;  bblog(node,'modelstat') = m.modelstat;  abort$(m.solvestat <> 1) "Solver did not return ok";  if (m.modelstat = 1 or m.modelstat = 2,  \*  \* get objective  \*  subobj = z.l;  bblog(node,'obj') = subobj;  \*  \* check "integrality"  \*  maxx = smax(i, min(y.l(i), 1-y.l(i)));  if (maxx = 0,  \*  \* found a new "integer" solution  \*  bblog(node,'integer') = 1;  if (subobj < bestfound,  \*  \* improved the best found solution  \*  bblog(node,'best') = 1;  bestfound = subobj;  \*  \* record best solution  \*  bestx(i) = x.l(i);  besty(i) = y.l(i);  \*  \* remove all waiting nodes with bound > bestfound  \*  wait(nd) = no; wait(waiting) = yes;  waiting(wait)$(bound(wait) > bestfound) = no;  );  else  \*  \* "fractional" solution  \*  fx(i) = no;  fx(i)$(min(y.l(i), 1-y.l(i))=maxx) = yes;  first2 = 1;  loop(fx$first2,  first2 = 0;  \*  \* create 2 new nodes  \*  newnode(nd) = newnode(nd-1);  fixedzero(newnode,i) = fixedzero(current,i);  fixedone(newnode,i) = fixedone(current,i);  bound(newnode) = subobj;  waiting(newnode) = yes;  fixedzero(newnode,fx) = yes;  \* at the first iteration, node2 = node 1, fixedzero(node2,i)= fixedzero(node1,i), here fixedzero(node1,i) is empty; fixedone(node1,i) is empty as well; bound(node2)=subobj;  \* in the waiting list, node2 is there. fixedzero(node2,'tbill')=yes. It means that the 'tbill' is fixed as zero at node2.  newnode(nd) = newnode(nd-1);  fixedzero(newnode,i) = fixedzero(current,i);  fixedone(newnode,i) = fixedone(current,i);  bound(newnode) = subobj;  waiting(newnode) = yes;  fixedone(newnode,fx) = yes;  \*node3 = node2, fixedzeor(node3,i)=fixedzero(node1,i), here fixedzero(node1,i) is empty; fixedone(node1,i) is empty as well; bound(node3)=subobj;  \*in the waiting list, node3 is there. fixedzero(node3,'tbill')=yes. It means that the 'tbill' is fixed as one at node3.  );  );  else  \*  \* if subproblem is infeasible this node is fathomed.  \* otherwise it is an error.  \*  abort$(m.modelstat <> 4 and m.modelstat <> 5)  "Solver did not solve subproblem";  );  bblog(node,'waiting') = card(waiting);  );  \*  \* are there new waiting nodes?  \*  done$(card(waiting) = 0) = 1;  \*  \* display log  \*  display bblog;  );  display bestx,besty,bestfound; |