SOME MINLP SOLUTION ALGORITHMS

ERWIN KALVELAGEN

Abstract. This document describes several methods to solve Mixed-Integer Nonlinear Programming (MINLP) problems within a GAMS environment. We demonstrate Generalized Benders Decomposition (GDB), Outer Approximation (OA) and Branch-and-bound (BB) using algorithms compactly implemented in the GAMS language.

1. Introduction

The GAMS language is powerful enough to implement reasonable complex algorithms. Examples include Lagrangian Relaxation with subgradient optimization [18], Benders Decomposition [16], Column Generation [17] and Dantzig-Wolfe Decomposition [19] algorithms .

In this document we will show how some algorithms for solving Mixed Integer Nonlinear Programming (MINLP) problems can be coded compactly in GAMS. In many cases, such algorithms written in GAMS are not competitive performance-wise to their siblings coded in traditional programming languages[7, 13]. However, they have substantial value for prototyping, research and educational purposes, and they can have use in special cases where a black-box solver is not appropriate.

MINLP models have gained significant popularity, largely due to problems and algorithmic work from chemical engineering institutions. We want to mention the work of Ignacio Grossmann and Chris Floudas (both chemical engineers) which has been very influential.

Many of the algorithms can use existing (local) NLP and (linear) MIP solvers as building blocks in the form of sub-solvers. This has an enormous advantage: all the algorithmic development available in state-of-the-art solvers is automatically and immediately productive in the MINLP algorithm.

Some of the more important solution strategies are branch-and-bound, generalized Benders Decomposition, and Outer-Approximation. We will illustrate these methods with an actual working GAMS model. For a general overview of MINLP solution methods see for instance [10, 1, 12].

The algorithms in this document are written in the GAMS language. There are also a number of ready-to-run MINLP solvers available under GAMS:

- DICOPT, from Carnegie-Mellon, outer-approximation and generalized Benders decomposition
- SBB, a branch-and-bound method
- BARON, if global solutions are required
- OQNLP, using a stochastic search method

Date: September, 2003, revised January, 2005; May 2009.

2. Cardinality-constrained portfolio problem

To illustrate the algorithms we consider the design of a portfolio x using a standard Mean-Variance quadratic programming formulation [26, 27] but with the additional condition that no more than P instruments can be in the portfolio.

The standard Mean-Variance portfolio problem can be formulated as Quadratic Programming (QP) problem:

$$\boxed{\begin{array}{lll}\text{PORTFOLIO} & \underset{x}{\text{minimize}} & \displaystyle\sum_i \sum_j x_i Q_{i,j} x_j \\ & \text{subject to} & \displaystyle\sum_i r_i x_i \geq \rho \\ & & \displaystyle\sum_i x_i = 1 \\ & & x_i \geq 0 \end{array}}$$

where x are the holdings, r indicated the returns with ρ being a minimum portfolio return and Q is the variance-covariance matrix.

The cardinality-constrained version of this model can be formulated as

$$\boxed{\begin{array}{lll}\text{CARDPORT} & \underset{x}{\text{minimize}} & \displaystyle\sum_i \sum_j x_i Q_{i,j} x_j \\ & \text{subject to} & \displaystyle\sum_i r_i x_i \geq \rho \\ & & \displaystyle\sum_i x_i = 1 \\ & & x_i \leq y_i \\ & & \displaystyle\sum_i y_i \leq P \\ & & x_i \geq 0, y_i \in \{0,1\} \end{array}}$$

For larger problems the generation and evaluation of the quadratic form $x^T Q x = \sum_i \sum_j x_i Q_{i,j} x_j$ becomes quite expensive. GAMS does not exploit the particular characteristics of this expression, and a large number of non-linearities and partial-derivatives are being generated. However, we can apply a somewhat no-intuitive reformulation. Often the number of observations to form the variance-covariance matrix is smaller than the number of possible instruments. The reason is that very old observations are not deemed to provide good information on the current situation. This means that the $(n \times n)$ matrix Q is larger than the $(n \times T)$ matrix holding the original time-series. We can exploit this by incorporating the original data $d_{i,t}$ in the model, using the definition of the covariance directly:

$$q_{ij} = \frac{1}{T} \sum_t (d_{i,t} - \mu_i)(d_{j,t} - \mu_j) \tag{1}$$

where $\mu_i = (1/T) \sum_t d_{i,t}$. This allows us to write:

$$\begin{aligned}\sum_{i,j} x_i q_{i,j} x_j &= \sum_{i,j}\left[\frac{1}{T} x_i x_j \sum_t (d_{i,t}-\mu_i)(d_{j,t}-\mu_j)\right]\\ &= \frac{1}{T}\sum_t\left[\left(\sum_i x_i(d_{i,t}-\mu_i)\right)\left(\sum_j x_j(d_{j,t}-\mu_j)\right)\right]\\ &= \frac{1}{T}\sum_t w_t^2\end{aligned} \tag{2}$$

where

$$w_t = \sum_k x_k (d_{k,t}-\mu_k) \tag{3}$$

which is a linear equation.

We end up here with a separable objective function which consists of a simple sum of squares. Notice that we proved in the derivation that Q is a positive semi-definite matrix:

$$x^T Q x = \frac{1}{T} w^T w \geq 0 \tag{4}$$

for all x. This proves again that the model is convex, and global solutions can be found.

The cardinality-constrained portfolio model can now be formulated as:

$$\boxed{\begin{array}{lll}\text{CARDPORTREF} & \underset{x}{\text{minimize}} & \sum_t w_t^2 \\ & \text{subject to} & \sum_i r_i x_i \geq \rho \\ & & \sum_i x_i = 1 \\ & & x_i \leq y_i \\ & & \sum_i y_i \leq P \\ & & w_t = \sum_k x_k(d_{k,t}-\mu_k) \\ & & x_i \geq 0, y_i \in \{0,1\}\end{array}}$$

A complete model formulated in GAMS can now be formulated as:

Model portf.gms. [1]

```
$ontext

  Cardinality constrained mean-variance portfolio model.

  Data from an AMPL example by Robert Vanderbei,
  Princeton University.

  Erwin Kalvelagen, 2000

$offtext


set i 'asset categories' /
```

[1] http://www.amsterdamoptimization.com/models/minlp/portf.gms

```
       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);

model m /all/;

option optcr=0;
option minlp=sbb;
solve m using minlp minimizing z;

display x.l,y.l,z.l;
```

We solve the model with SBB, a branch-and-bound method. The log is:

```
 Root node solved locally optimal.
 Resetting optcr to 1.0e-09
   Node  Act. Lev.  Objective  IInf  Best Int.   Best Bound   Gap  (0 secs)
      0     2   0       0.0671   5           -       0.0671         -
      1     3   1       0.0676   5           -       0.0671         -
      2     4   2       0.0696   5           -       0.0671         -
      3     5   3       0.0703   4           -       0.0671         -
      4     6   4       0.1183   3           -       0.0671         -
      5     5   5   infeasible   -           -       0.0671         -
      6     6   5       0.1183   2           -       0.0671         -
      7     7   6       0.1352   2           -       0.0671         -
      8     8   7       0.1356   1           -       0.0671         -
*     9     7   8       0.1356   0       0.1356      0.0671  1.020471
     10     8   1       0.0671   4       0.1356      0.0671  1.020471
     11     9   2       0.0671   3       0.1356      0.0671  1.020471
     12    10   2       0.0925   3       0.1356      0.0671  1.020471
     13    11   3       0.0768   4       0.1356      0.0671  1.020471
     14    12   3       0.0671   2       0.1356      0.0671  1.020471
     15    13   4       0.0671   1       0.1356      0.0671  1.020471
     16    14   4       0.0828   1       0.1356      0.0671  1.020471
*    17    10   5       0.1097   0       0.1097      0.0671  0.635182
*    18     0   5       0.0671   0       0.0671      0.0671  0.000000
 Integer solution
 Statistics:
    Iterations    :             161
    NLP Seconds   :        0.550781
    B&B nodes     :              18
    MIP solution  :        0.067105 found in node 18
    Best possible :        0.067105
    Absolute gap  :        0.000000     optca :  0.000000
    Relative gap  :        0.000000     optcr :  0.000000
    Model Status  :               8
    Solver Status :               1

 NLP Solver Statistics
    Total Number of NLP solves   :         20
    Total Number of NLP failures:           0
    Details:         conopt
      # execs           20
      # failures         0
 Terminating.
```

I.e. it needs 19 nodes, plus one additional NLP evaluation, to find the optimal configuration:

```
----    106 VARIABLE x.L  weights

tbill  0.359,    wfiv   0.088,    lbcorp 0.318,    eafe   0.128,    gold   0.107
```

```
----    106 VARIABLE y.L  indicator wether in portfolio

tbill  1.000,    wfiv   1.000,    lbcorp 1.000,    eafe   1.000,    gold   1.000


----    106 VARIABLE z.L                  =        0.067  objective variable
```

This model is solved using heuristics in [6]. A parallel branch-and-bound SQP method applied to the model is described in [24].

3. Branch-and-bound Methods

In this section we implement a branch-and-bound algorithm for MINLP models. Branch-and-bound methods are used extensively for mixed-integer linear programming models and go back to [23]. The basic method is directly applicable to models with nonlinear functions in which case a nonlinear solver needs to evaluate the relaxed sub-problems. A standard branch-and-bound algorithm for MINLP problems with integer variables can look like [4, 14, 25]:

Branch & Bound Algorithm.

```
{Initialization}
$LB := -\infty$ {Best possible}
$UB := +\infty$ {Best found}
Store root node in waiting node list
**while** waiting node list is not empty **do**
  {Node selection}
  Choose a node from the waiting node list
  Remove it from the waiting node list
  Solve subproblem
  **if** infeasible **then**
    Node is fathomed
  **else if** optimal **then**
    **if** integer solution **then**
      **if** $obj < UB$ **then**
        {Better integer solution found}
        $UB := obj$
        Remove nodes $j$ from list with $LB_j > UB$
      **end if**
    **else**
      {Variable selection}
      Find variable $x_k$ with factional value $v$
      Create node $j_{new}$ with bound $x_k \le \lfloor v \rfloor$
      $LB_{j_{new}} := obj$
      Store node $j_{new}$ in waiting node list
      Create node $j_{new}$ with bound $x_k \ge \lceil v \rceil$
      $LB_{j_{new}} := obj$
      Store node $j_{new}$ in waiting node list
    **end if**
  **else**
    Stop: problem in solving subproblem
  end if
  $LB = \min_j LB_j$
end while
{End of algorithm}
```

The root node is the problem with all integer restrictions relaxed.

Branch-and-bound methods are based on the concept of *relaxations*: sub-problems with one or more of the discrete variable relaxed to continuous variables. The number of relaxations to explore is often very large for problems with many integer variables. Mixed-integer linear programming branch-and-bound solvers put much emphasis on solving relaxations very fast (e.g. by using dual methods). Modern solvers also do lots of work on preprocessing: presolving the model to make it smaller and adding cuts to make the feasible region smaller (although this makes the problem larger in terms of number of constraints). Many of these techniques are not available in MINLP branch-and-bound solvers.

Although most descriptions of branch-and-bound methods emphasize the tree structure underlying the algorithm, this simple version of the algorithm does not need any tree-like data-structures. All we use is a list of waiting nodes with a corresponding upper-bound. Such a list can be easily implemented in GAMS using some sets and parameters.

The complete GAMS implementation of the above algorithm may look like:

Model portfbb.gms. [2]

```
$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
```

[2] http://www.amsterdamoptimization.com/models/minlp/portfbb.gms

```
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),

*
* 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 presented log is:

```
----    310 PARAMETER bblog  logging information

           node       lb  solvestat  modelstat      obj  integer     best   waiting

node1     1.000     +INF      1.000      2.000    0.067                       2.000
node2     2.000    0.067      1.000      2.000    0.093                       3.000
node3     3.000    0.067      1.000      2.000    0.067                       4.000
```

```
node4     6.000     0.067      1.000      2.000     0.077                           5.000
node5     7.000     0.067      1.000      2.000     0.067                           6.000
node6    10.000     0.067      1.000      2.000     0.083                           7.000
node7    11.000     0.067      1.000      2.000     0.067                           8.000
node8    14.000     0.067      1.000      2.000     0.110                           9.000
node9    15.000     0.067      1.000      2.000     0.067                          10.000
node10   18.000     0.067      1.000      2.000     0.068                          11.000
node11   19.000     0.067      1.000      2.000     0.067     1.000     1.000
```

The algorithm can solve this problem quickly and gives the correct global optimal objective function of 0.067. The number of nodes needed is 11, which is actually less than SBB needs. The solution is the same:

```
----    315 PARAMETER bestx  record best solution

tbill  0.359,    wfiv   0.088,    lbcorp 0.318,    eafe   0.128,    gold   0.107


----    315 PARAMETER besty  record best solution

tbill  1.000,    wfiv   1.000,    lbcorp 1.000,    eafe   1.000,    gold   1.000


----    315 PARAMETER bestfound             =        0.067  lowerbound in B&B tree
```

4. Integer cuts

In the following algorithms we will use a construct that forbids a known integer solution to be considered again. Assume we have already considered a set integer solutions $\bar{y}_i^{(k)}$ for $k = 1, \ldots, K$. We can forbid y_i to assume any of these values by the inequality:

$$\sum_i |y_i - \bar{y}_i^{(k)}| \neq 0 \text{ for } k = 1, \ldots, K \tag{5}$$

This is a non-linear constraint that can be converted into a linear inequality as follows:

$$\begin{aligned}
&\sum_i |y_i - \bar{y}_i^{(k)}| \neq 0 \\
\Rightarrow &\sum_i |y_i - \bar{y}_i^{(k)}| \geq 1 \\
\Rightarrow &\sum_{i|\bar{y}_i^{(k)}=0} |y_i| + \sum_{i|\bar{y}_i^{(k)}=1} |y_i - 1| \geq 1 \\
\Rightarrow &\sum_{i|\bar{y}_i \in N^{(k)}} y_i + \sum_{i|\bar{y}_i \in P^{(k)}} (1 - y_i) \geq 1 \\
\Rightarrow &\sum_{i|\bar{y}_i \in N^{(k)}} y_i + |P^{(k)}| - \sum_{i|\bar{y}_i \in P^{(k)}} y_i \geq 1 \\
\Rightarrow &\sum_{i|\bar{y}_i \in P^{(k)}} y_i - \sum_{i|\bar{y}_i \in N^{(k)}} y_i \leq |P^{(k)}| - 1
\end{aligned} \tag{6}$$

We introduced here the sets $P^{(k)} = \{i|\bar{y}_i^{(k)} = 1\}$ and $N^{(k)} = \{i|\bar{y}_i^{(k)} = 0\}$. The expression $|P^{(k)}|$ denotes the cardinality (the number of elements) of the set $P^{(k)}$.

The last constraint:

$$\sum_{i|\bar{y}_i \in P^{(k)}} y_i - \sum_{i|\bar{y}_i \in N^{(k)}} y_i \leq |P^{(k)}| - 1 \tag{7}$$

is easily formulated in a GAMS environment.

5. Generalized Benders Decomposition

Benders Decomposition, introduced in [3] for linear models (for a GAMS implementation see [16]), was generalized by [11], allowing for nonlinearities.

Assume we have an MINLP model in the form of:

$$\begin{aligned} \min_{x,y}\ & c^T y + f(x) \\ & h(x) = 0 \\ & g(x) \leq 0 \\ & Cx + By \leq b \\ & Ay \leq a \\ & x \in \mathcal{R}^n \\ & y \in \{0,1\}^m \end{aligned} \tag{8}$$

Here x are the non-linear variables and y are the binary variables, appearing linearly in the model. This is not a real loss of generalization as we can introduce an extra variable and extra linking equations in case a binary variable appears non-linearly in the model.

The Generalized Benders' Decomposition algorithm [11, 9] can be stated as:

Generalized Benders Decomposition.

```
{Initialization}
$K := 1$
$UB := \infty$
select an initial configuration for the integer variables $\bar{y}^{(K)}$
{Step 1: NLP subproblem}
Solve
```

$$\begin{aligned} \min_{x}\ & z_{nlp} = c^T \bar{y}^{(K)} + f(x) \\ & h(x) = 0 \\ & g(x) \leq 0 \\ & Cx + B\bar{y}^{(K)} \leq b \\ & x \in X \end{aligned} \tag{9}$$

```
where $\lambda$ are the duals for inequality $Cx + B\bar{y}^{(K)} \leq b$
Let the solution be $\bar{x}^{(K)}$ and $\bar{\lambda}^{(K)}$
**if** $z_{nlp} < UB$ **then**
    {Record better solution}
    $UB := z_{nlp}$
    $x^* := \bar{x}^{(K)}$
    $y^* := \bar{y}^{(K)}$
**end if**
```

```
{Step 2: MIP Master Problem}
Let
```

$$\mathcal{L}(x^{(k)}, y, \lambda^{(k)})) = c^T y + f(x^{(k)}) + (\lambda^{(k)})^T (Cx^{(k)} + By - b) \tag{10}$$

```
Solve
```

$$\begin{aligned} &\min_{y,\mu} z_{mip} = \mu \\ &\mu \ge \mathcal{L}(\bar{x}^{(k)}, y, \bar{\lambda}^{(k)})) \text{ for } k = 1, \ldots, K \\ &Ay \le a \\ &y \in \{0,1\}^m \end{aligned} \tag{11}$$

```
Let the solution be $\bar{y}^{K+1}$.
if $z_{mip} \ge UB$ then
  Stop
else
  $K := K + 1$
  Go to step 1
end if
{End of algorithm}
```

The NLP subproblem (also known as the *primal problem* is just the MINLP problem with the integer variables fixed. The Master problem is using nonlinear duality theory in its formulation.

We can construct a GAMS model that implements this.

Model portfgbd.gms. [3]

```
$ontext

  Cardinality constrained mean-variance portfolio model.
  Generalized Benders Decomposition (GBD) 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
```

[3]http://www.amsterdamoptimization.com/models/minlp/portfgbd.gms

```
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);



*--------------------------------------------------------------
* GBD sub problem
* this is the NLP model resulting from fixing the integer
* variables. In our implementation, this is a RMINLP model
* with the constraint CARDINALITY removed (it is only
* in the integer variables which are all fixed).
*-------------------------------------------------------------

model sub /obj,budget,minreturn,wdef,indicator/;




*-------------------------------------------------------------
* GBD master problem
* This is a linear MIP model.
* We add integer cuts to disallow previously considered integer
* configurations.
*-------------------------------------------------------------

set iter 'cycle numbers' /iter1*iter50/;
set cycle(iter) 'dynamic subset: cycles done';

variable  mu;
equation  mastercut(iter)  'benders cut';
equation  integercut(iter) 'integer cut';

parameter lambda(iter,i) 'duals of equation INDICATOR';
parameter wkeep(iter,t)  'continuous variables from previous cycles';
parameter xkeep(iter,i)  'continuous variables from previous cycles';
set icutzero(iter,i);
set icutone(iter,i);
set icutset(iter);


mastercut(cycle)..
    mu =g= sum(t, sqr(wkeep(cycle,t))) +
           sum(i, lambda(cycle,i)*(xkeep(cycle,i)-y(i)));

integercut(icutset)..
    sum(icutone(icutset,i), y(i)) - sum(icutzero(icutset,i), y(i))
      =l= sum(icutone(icutset,i),1)-1;

icutset(iter) = no;
icutzero(iter,i) = no;
icutone(iter,i) = no;

model master /mastercut,cardinality,integercut/;


*-------------------------------------------------------------
* initial integer configuration
*-------------------------------------------------------------

set initportf(i) 'initial portfolio' /
     qqq     'Nasdaq composite'
     lbcorp  'Lehman Brothers corporate bonds index'
/;
y.l(i) = 0;
y.l(initportf) = 1;

*-------------------------------------------------------------
* GENERALIZED BENDERS DECOMPOSITION
*-------------------------------------------------------------
option mip=cplex;
option rminlp=conopt;
option limrow=0;
option limcol=0;
option optcr=0;
option solprint=off;


set problem /sub,master/;
parameter gbdlog(iter,problem,*) 'algorithm log';
parameter ybest(i) 'record best solution';
parameter xbest(i) 'record best solution';
```

```
*
* initialization
*
scalar UB 'upperbound' /INF/;
scalar done /0/;


loop(iter$(not done),


*
* fix integer variables
*
   y.fx(i) = round(y.l(i));
   display y.l;


*
* solve NLP subproblem
*
   solve sub minimizing z using rminlp;
   abort$(sub.solvestat <> 1) "NLP Solver did not return OK";
   abort$(sub.modelstat = 3 or sub.modelstat >= 6) "NLP Solver did not return OK";

   gbdlog(iter,'sub','solvestat') = sub.solvestat;
   gbdlog(iter,'sub','modelstat') = sub.modelstat;



*
* if optimal/locally optimal...
*
   if(sub.modelstat = 1 or sub.modelstat = 2,

        gbdlog(iter,'sub','obj') = z.l;

*
* do we have a better solution?
*
        if(z.l < UB,
            UB = z.l;
            gbdlog(iter,'sub','better') = 1;
*
* record best solution
*
            ybest(i) = y.l(i);
            xbest(i) = x.l(i);
        );

   );


*
* remember data for constructing Bender's cuts
*

   xkeep(iter,i) = x.l(i);
   wkeep(iter,t) = w.l(t);
   lambda(iter,i) = -indicator.m(i);
   cycle(iter) = yes;


*
* remove fixed bounds
*
   y.lo(i) = 0;
   y.up(i) = 1;

*
* solve master problem
*
   solve master minimizing mu using mip;

   gbdlog(iter,'master','solvestat') = master.solvestat;
   gbdlog(iter,'master','modelstat') = master.modelstat;
   gbdlog(iter,'master','obj') = mu.l;

*
* make sure this integer solution is not visited again
*
   icutset(iter) = yes;
   icutzero(iter,i)$(y.l(i)<0.5) = yes;
   icutone(iter,i)$(y.l(i)>0.5) = yes;

*
* stopping criterion
*
   if (mu.l >= UB,
      done = 1;
   );


);


display gbdlog;
display xbest,ybest,ub;
```

The cycle log is as follows:

```
----    276 PARAMETER gbdlog  algorithm log

               solvestat   modelstat         obj      better

iter1.sub          1.000       2.000       0.217       1.000
iter1.master       1.000       1.000      -2.452
iter2.sub          1.000       2.000       0.077       1.000
iter2.master       1.000       1.000      -0.001
iter3.sub          1.000       2.000       0.130
iter3.master       1.000       1.000      -0.001
iter4.sub          1.000       2.000       0.093
iter4.master       1.000       1.000      -0.001
iter5.sub          1.000       2.000       0.118
iter5.master       1.000       1.000      -0.001
iter6.sub          1.000       2.000       0.070       1.000
iter6.master       1.000       1.000      -0.001
iter7.sub          1.000       2.000       0.083
iter7.master       1.000       1.000      -0.001
iter8.sub          1.000       2.000       0.110
iter8.master       1.000       1.000       0.016
iter9.sub          1.000       2.000       0.067       1.000
iter9.master       1.000       1.000       0.067
```

The results are as we have seen before:

```
----    277 PARAMETER xbest  record best solution

tbill  0.359,    wfiv   0.088,    lbcorp 0.318,    eafe   0.128,    gold   0.107


----    277 PARAMETER ybest  record best solution

tbill  1.000,    wfiv   1.000,    lbcorp 1.000,    eafe   1.000,    gold   1.000


----    277 PARAMETER UB                   =        0.067  upperbound
```

We did not encounter infeasible NLP subproblems. We could handle infeasible subproblems by simply adding an integer cut: the integer configuration leading to this infeasibility will be forbidden. For another approach using explicit slack variables see [1]. We also only used a Benders optimality cut; a slight variation can be used in case of NLP infeasibility, called a feasibility cut.

For another example of Generalized Benders Decomposition expressed in GAMS see [2], based on the algorithmic framework APROS by [15].

6. Outer Approximation

In the Generalized Benders Algorithm the MIP master model did not contain any x variables: only the integer variables y were present. In the Outer Approximation (OA) the variables x actually are present in the linearizations in the Master MIP problem. Assume again the MINLP problem to be solved is:

$$\begin{aligned}
&\min_{x,y} c^T y + f(x) \\
&\quad h(x) = 0 \\
&\quad g(x) \le 0 \\
&\quad Cx + By \le b \\
&\quad Ay \le a \\
&\quad x \in \mathcal{R}^n \\
&\quad y \in \{0,1\}^m
\end{aligned} \tag{12}$$

The NLP subproblem is as we have seen before the NLP problem formed by fixing the integer variable to an initial suggestion or to a proposal generated by the Master MIP problem. This Master Problem in an outer approximation framework[20, 21, 22] is somewhat more complicated:

$$\begin{aligned}
&\min \mu \\
&\quad \mu \ge c^T y + f(\bar{x}^{(k)}) + \nabla f(x)^T (x - \bar{x}^{(k)}) \\
&\quad 0 \ge g(\bar{x}^{(k)}) + \nabla g(x)^T (x - \bar{x}^{(k)}) \\
&\quad 0 \ge T^{(k)} [h(\bar{x}^{(k)}) + \nabla h(x)^T (x - \bar{x}^{(k)})] \\
&\quad Cx + By \le b \\
&\quad Ay \le a \\
&\quad y \in \{0,1\}
\end{aligned} \tag{13}$$

where the matrix $T^{(k)}$ is a diagonal matrix defined by:

$$T_{i,i}^{(k)} = \begin{cases} -1 & \text{if } \lambda_i^{(k)} < 0 \\ +1 & \text{if } \lambda_i^{(k)} > 0 \\ 0 & \text{if } \lambda_i^{(k)} = 0 \end{cases} \tag{14}$$

where $\lambda^{(k)}$ are the duals of the equality constraints $h(x) = 0$ at cycle k.

The structure of the model makes the linearizations easy as only the objective contains non-linear terms w_t^2. The OA algorithm is therefore stated readily as:

Model portfoa.gms. [4]

[4] http://www.amsterdamoptimization.com/models/minlp/portfoa.gms

```
$ontext

  Cardinality constrained mean-variance portfolio model.
  Outer-Approximation (OA) 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);



*-------------------------------------------------------------
* OA sub problem
* this is the NLP model resulting from fixing the integer
* variables. In our implementation, this is a RMINLP model
* with the constraint CARDINALITY removed (it is only
* in the integer variables which are all fixed).
*-------------------------------------------------------------

model sub /obj,budget,minreturn,wdef,indicator/;


*-------------------------------------------------------------
* OA master problem
* This is a linear MIP model.
* We add integer cuts to disallow previously considered integer
* configurations.
*-------------------------------------------------------------

set iter 'cycle numbers' /iter1*iter50/;
set cycle(iter) 'dynamic subset: cycles done';

variable  mu;
equation  flin(iter)  'linearization of f';
equation  integercut(iter) 'integer cut';

parameter wkeep(iter,t)  'continuous variables from previous cycles';
set icutzero(iter,i);
set icutone(iter,i);
set icutset(iter);


flin(cycle)..
    mu =g= sum(t, sqr(wkeep(cycle,t))) +
           sum(t, 2*wkeep(cycle,t)*(w(t)-wkeep(cycle,t)));


integercut(icutset)..
    sum(icutone(icutset,i), y(i)) - sum(icutzero(icutset,i), y(i))
      =l= sum(icutone(icutset,i),1)-1;

icutset(iter) = no;
```

```
icutzero(iter,i) = no;
icutone(iter,i) = no;

model master /flin,cardinality,budget,minreturn,wdef,indicator,integercut/;


*-------------------------------------------------------------
* initial integer configuration
*-------------------------------------------------------------

set initportf(i) 'initial portfolio' /
     qqq     'Nasdaq composite'
     lbcorp  'Lehman Brothers corporate bonds index'
/;
y.l(i) = 0;
y.l(initportf) = 1;

*-------------------------------------------------------------
* OUTER APPROXIMATION ALGORITHM
*-------------------------------------------------------------
option mip=cplex;
option rminlp=conopt;
option limrow=0;
option limcol=0;
option optcr=0;
option solprint=off;


set problem /sub,master/;
parameter oalog(iter,problem,*) 'algorithm log';
parameter ybest(i) 'record best solution';
parameter xbest(i) 'record best solution';


*
* initialization
*
scalar UB 'upperbound' /INF/;
scalar done /0/;


loop(iter$(not done),


*
* fix integer variables
*
   y.fx(i) = round(y.l(i));
   display y.l;


*
* solve NLP subproblem
*
   solve sub minimizing z using rminlp;
   abort$(sub.solvestat <> 1) "NLP Solver did not return OK";
   abort$(sub.modelstat = 3 or sub.modelstat >= 6) "NLP Solver did not return OK";

   oalog(iter,'sub','solvestat') = sub.solvestat;
   oalog(iter,'sub','modelstat') = sub.modelstat;



*
* if optimal/locally optimal...
*
   if(sub.modelstat = 1 or sub.modelstat = 2,

        oalog(iter,'sub','obj') = z.l;

*
* do we have a better solution?
*
        if(z.l < UB,
            UB = z.l;
            oalog(iter,'sub','better') = 1;
*
* record best solution
*
            ybest(i) = y.l(i);
            xbest(i) = x.l(i);
        );

   );


*
* remember data for constructing linearizations
*

   wkeep(iter,t) = w.l(t);
   cycle(iter) = yes;


*
* remove fixed bounds
*
   y.lo(i) = 0;
   y.up(i) = 1;

*
* solve master problem
*
   solve master minimizing mu using mip;

   oalog(iter,'master','solvestat') = master.solvestat;
   oalog(iter,'master','modelstat') = master.modelstat;
   oalog(iter,'master','obj') = mu.l;

*
* make sure this integer solution is not visited again
*
   icutset(iter) = yes;
   icutzero(iter,i)$(y.l(i)<0.5) = yes;
   icutone(iter,i)$(y.l(i)>0.5) = yes;

*
* stopping criterion
*
   if (mu.l >= UB,
      done = 1;
   );


);


display oalog;
display xbest,ybest,ub;
```

The log is:

```
----    271 PARAMETER oalog  algorithm log

               solvestat   modelstat         obj      better

iter1.sub          1.000       2.000       0.217       1.000
iter1.master       1.000       1.000      -0.757
iter2.sub          1.000       2.000       2.669
iter2.master       1.000       1.000      -0.434
iter3.sub          1.000       2.000       0.094       1.000
iter3.master       1.000       1.000      -0.013
iter4.sub          1.000       2.000       0.099
iter4.master       1.000       1.000       0.025
```

```
iter5.sub          1.000       2.000       0.078       1.000
iter5.master       1.000       1.000       0.039
iter6.sub          1.000       2.000       0.070       1.000
iter6.master       1.000       1.000       0.053
iter7.sub          1.000       2.000       0.067       1.000
iter7.master       1.000       1.000       0.067
iter8.sub          1.000       2.000       0.083
iter8.master       1.000       1.000       0.067
```

with the same solution as we have seen before:

```
----    272 PARAMETER xbest  record best solution

tbill  0.359,    wfiv   0.088,    lbcorp 0.318,    eafe   0.128,    gold   0.107


----    272 PARAMETER ybest  record best solution

tbill  1.000,    wfiv   1.000,    lbcorp 1.000,    eafe   1.000,    gold   1.000


----    272 PARAMETER UB                   =        0.067  upperbound
```

For more information on outer approximation methods see e.g. [10]. A computational comparison between branch-and-bound and outer-approximation was carried out in [5, 8].

7. Linear approximation

A linearized version of the cardinality constrained portfolio model can be developed by replacing the objective

$$\min \sum_t w_t^2 \tag{15}$$

by

$$\min \sum_t |w_t| \tag{16}$$

Using variable splitting we can write:

$$\begin{aligned} w_t &= w_t^+ - w_t^- \\ w_t^+ w_t^- &= 0 \\ w_t^+ &\ge 0 \\ w_t^- &\ge 0 \end{aligned} \tag{17}$$

The absolute value is now:

$$|w_t| = w_t^+ + w_t^- \tag{18}$$

As we are minimizing the absolute values, the non-linear constraint $w_t^+ w_t^- = 0$ will hold automatically, and can be dropped. The resulting linear mixed integer model in formulated GAMS could look like:

Model portflin.gms. [5]

[5]http://www.amsterdamoptimization.com/models/minlp/portflin.gms

```
$ontext

  Cardinality constrained mean-variance portfolio model.
  Linearized version.

  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'
   wplus(t) 'auxiliary variable (plus part)'
   wmin(t)  'auxiliary variable (min part)'
;
positive variable x(i),wplus(t),wmin(t);
binary variable y(i);

equations
   obj          'objective'
   linobj       'linear 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'
   varsplit(t)  'variable splitting'
;


linobj.. z =e= sum(t, wplus(t)+wmin(t));

obj.. z =e= sum(t, sqr(w(t)));

varsplit(t).. w(t) =e= wplus(t) - wmin(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);

*------------------------------------------------------------------
* solve the linearized version
*------------------------------------------------------------------

model mlin /linobj,varsplit,budget,minreturn,cardinality,wdef,indicator/;
option optcr=0;
option mip=cplex;
solve mlin using mip minimizing z;
display x.l,y.l,z.l;
```

This model gives as portfolio:

```
----    117 VARIABLE x.L  weights

tbill  0.299,    lbcorp 0.424,    eafe   0.162,    gold   0.115


----    117 VARIABLE y.L  indicator wether in portfolio

tbill  1.000,    lbcorp 1.000,    eafe   1.000,    gold   1.000
```

This is not exactly the same as the optimal portfolio, but it is close. Of course this solution can be used as an initial configuration or bound in MINLP algorithms.

8. Conclusion

We have shown how GAMS allows direct implementation of a number of MINLP algorithms: branch-and-bound, generalized Benders and outer-approximation.

9. Aknowledgements

I would like to thank Wilfredo Sifuentes for pointing out an error in an earlier version of this paper.

References

1. C. S. Adjiman, C. A. Schweiger, and C. A. Floudas, *Mixed-integer nonlinear optimization in process synthesis*, Handbook of Combinatorial Optimization (D.-Z. Du and P. M. Pardalos, eds.), Kluwer, 1998.
2. A. Aggarwal and C. A. Floudas, *Synthesis of general distillation sequences*, Model library file: `http://www.gams.com/modlib/libhtml/nonsharp.htm`.
3. J. F. Benders, *Partitioning procedures for solving mixed-variables programming problems*, Numerische Mathematik **4** (1962), 238–252.
4. Brian Borchers, *(MINLP) Branch and Bound Methods*, Encyclopedia of Optimization, vol. 3, Kluwer, 2001, pp. 331–335.
5. Brian Borchers and John E. Mitchell, *A computational comparison of branch and bound and outer approximation algorithms for 0-1 mixed integer nonlinear programs*, Tech. report, Mathematics Department, New Mexico Tech and Department of Mathematical Sciences, Rensselaer Polytechnic Institute, December 1996.
6. T.-J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha, *Heuristics for cardinality constrained portfolio optimisation*, Computers and Operations Research **27** (2000), 1271–1302.
7. GAMS Development Corp., *SBB*, `http://www.gams.com/docs/solver/sbb.pdf`, April 2002.
8. Roger Fletcher and Sven Leyffer, *Solving mixed integer nonlinear programs by outer approximation*, Tech. report, University of Dundee, October 1996.
9. O. E. Flippo and A. H. G. Rinnooy Kan, *Decomposition in general mathematical programming*, Mathematical Programming **60** (1993), 361–382.
10. Christodoulos A. Floudas, *Nonlinear and mined-integer optimization*, Oxford University Press, 1995.
11. A. M. Geoffrion, *Generalized Benders decomposition*, Journal of Optimization Theory and Applications **10** (1972), no. 4, 237–260.
12. Ignacio E. Grossmann, *Review of nonlinear mixed-integer and disjunctive programming techniques*, Tech. report, Department of Chemical Engineering, Carnegie Mellon University, June 2001.
13. Ignacio E. Grossmann, Jagadisan Viswanathan, Aldo Vecchietti, Ramesh Raman, and Erwin Kalvelagen, *DICOPT: A discrete continuous optimization package*, `http://www.amsterdamoptimization.com/pdf/dicopt.pdf`, January 2003.
14. O. K. Gupta and V. Ravindran, *Branch and bound experiments in convex nonlinear integer programming*, Management Science **31** (1985), no. 12, 1533–1546.
15. G. E. Paules IV and C. A. Floudas, *APROS: Algorithmic development methodology for discrete-continuous optimization problems*, Operations Research **37** (1989), no. 6, 902–915.
16. Erwin Kalvelagen, *Benders decomposition with GAMS*, `http://www.amsterdamoptimization.com/pdf/benders.pdf`, December 2002.
17. ______, *Column generation with GAMS*, `http://www.amsterdamoptimization.com/pdf/colgen.pdf`, December 2002.
18. ______, *Lagrangian relaxation with GAMS*, `http://www.amsterdamoptimization.com/pdf/lagrange.pdf`, December 2002.
19. ______, *Dantzig-Wolfe decomposition with GAMS*, `http://www.amsterdamoptimization.com/pdf/dw.pdf`, May 2003.
20. G. R. Kocis and I. E. Grossmann, *Relaxation strategy for the structural optimization of process flow sheets*, Industrial and Engineering Chemistry Research **26** (1987), no. 9, 1869.
21. ______, *Global optimization of nonconvex MINLP problems in process synthesis*, Industrial and Engineering Chemistry Research **27** (1988), no. 8, 1407–1421.
22. ______, *Computational experience with DICOPT solving MINLP problems in process synthesis*, Computers & Chemical Engineering **13** (1989), 307.
23. A. H. Land and A. G. Doig, *An automatic method for solving discrete programming problems*, Econometrica **28** (1960), no. 3, 497–520.

24. E. K. Lee and J. E. Mitchell, *Computational experience of an interior-point sqp algorithm in a parallel branch-and-bound framework*, Working paper, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, 1997.
25. Sven Leyffer, *Integrating SQP and branch-and-bound for mixed integer nonlinear programming*, Computational Optimization and Applications **18** (2001), 295–309.
26. H. M. Markowitz, *Portfolio selection*, Journal of Finance **7** (1952), 77–91.
27. A. D. Roy, *Safety first and the holding of assets*, Econometrica **20** (1952), 413–449.

Amsterdam Optimization Modeling Group, Washington DC/The Hague
E-mail address: erwin@amsterdamoptimization.com