BENDERS DECOMPOSITION FOR STOCHASTIC PROGRAMMING WITH GAMS

ERWIN KALVELAGEN

Abstract. This document describes an implementation of Benders Decomposition for solving two-stage Stochastic Linear Programming problems using GAMS.

1. Introduction

The two-stage stochastic linear programming problem can be stated as [2, 3, 4]:

$$\boxed{\text{SLP}\qquad \begin{aligned} \underset{x}{\text{minimize}}\ & c^T x + E_\omega Q(x,\omega)\\ & Ax = b\\ & x \geq 0 \end{aligned}}$$

where

$$\begin{aligned} Q(x,\omega) = \min_y\ & d_\omega^T y\\ & T_\omega x + W_\omega y = h_\omega\\ & y \geq 0 \end{aligned} \tag{1}$$

Here E_ω is the expectation, and ω denotes a scenario or possible outcome with respect to the probability space (Ω, P). We will consider discrete distributions P only, so we can write:

$$E_\omega Q(x,\omega) = \sum_{\omega\in\Omega} p(\omega) Q(x,\omega) \tag{2}$$

Using this we can formulate a large LP that forms the *deterministic equivalent* problem:

$$\begin{aligned} \min\ & c^T x + \sum_\omega p(\omega) d_\omega^T y_\omega\\ & Ax = b\\ & T_\omega x + W_\omega y_\omega = h_\omega\\ & x \geq 0, y_\omega \geq 0 \end{aligned} \tag{3}$$

Date: 17 januari 2003.

The structure of equation $T_\omega x + W_\omega y_\omega = h_\omega$ is called L-shaped, which can be made visible by writing it out:

$$
\begin{array}{lllllll}
T_1 x & +W_1 y_1 & & & & = & h_1 \\
T_2 x & & +W_2 y_2 & & & = & h_2 \\
T_3 x & & & +W_3 y_3 & & = & h_3 \\
 & & & & & \vdots & \\
T_K x & & & & +W_K y_K & = & h_K
\end{array} \tag{4}
$$

2. Benders Algorithm

The Benders' algorithm[1, 7] to solve this problem can be formulated as follows (we largely follow the notation in[6]).

Step 1: Initialization
$\nu := 1$ {Iteration number}
$UB := \infty$ {Upper bound}
$LB := -\infty$ {Lower bound}
Solve initial master problem:

$$
\begin{aligned}
\min\ & c^T x \\
& Ax = b \\
& x \geq 0
\end{aligned} \tag{5}
$$

$\overline{x}^\nu := x^*$ {Optimal values}

Step 2: Sub problems
for $\omega \in \Omega$ **do**
Solve the sub problem:

$$
\begin{aligned}
\min\ & d_\omega^T y_\omega \\
& W_\omega y_\omega = h_\omega - T_\omega \overline{x}^\nu \\
& y_\omega \geq 0
\end{aligned} \tag{6}
$$

$\overline{y}_\omega^\nu := y_\omega^*$ {Optimal values}
$\overline{\pi}_\omega^\nu := \pi_\omega^*$ {Optimal dual values}
end for
$UB := \min\{UB, c^T \overline{x}^\nu + \sum_{\omega\in\Omega} p_\omega d_\omega^T \overline{y}_\omega^\nu\}$

Step 3: Convergence test
if $(UB - LB)/(1 + LB) \leq TOL$ **then**
Stop: required accuracy achieved
Return $\overline{x}^\nu$
end if

Step 4: Master problem

Solve the Master problem:

$$
\begin{aligned}
&\min c^T x + \theta \\
&\quad Ax = b \\
(7) \qquad &\quad \theta \geq \sum_{\omega\in\Omega} p_\omega \left(-\overline{\pi}^{\ell}_{\omega}[T_\omega x + W_\omega \overline{y}^{\ell}_{\omega} - h_\omega]\right),\ \ell = 1,\ldots,\nu-1 \\
&\quad x \geq 0
\end{aligned}
$$

$\overline{x}^{\nu} := x^*$ {Optimal values}
$\overline{\theta}^{\nu} := \theta^*$
$LB := c^T\overline{x}^{\nu} + \overline{\theta}^{\nu}$
go to step 2.

If the subproblems are infeasible, we need to include slightly different formulated cuts. The optimality cuts shown here assume the subproblems could be solved to an optimal feasible solution.

3. Example

The standard transportation model can be stated as:

$$
\text{TRANSPORT} \qquad \begin{aligned}
\underset{x}{\text{minimize}} \ & \sum_{i,j} c_{i,j} x_{i,j} \\
& \sum_j x_{i,j} = s_i \ \forall i \\
& \sum_i x_{i,j} = d_j \ \forall j \\
& x_{i,j} \geq 0
\end{aligned}
$$

where $c_{i,j}$ are the unit transportation costs, s_i is the supply at plant i and d_j is the demand at location j.

Now consider the case that demand d_j is stochastic. We assume that products are shipped before the actual demand is observed. After the products arrive at each location j, actual demand is known. If demand is not met, the sales is lost. If the shipment is larger than the final demand, then the remaining products need to be disposed of as they have no shelf life (i.e. they spoil). Unit disposal costs are known.

To model this, first assume we somehow know the demand. This means we can build a non-stochastic version of the model, also called the *core model*:

$$
\begin{aligned}
&\max \sum_j p_j \mathit{Sales}_j - \sum_{i,j} c_{i,j} \mathit{Ship}_{i,j} - \sum_i c_i \mathit{Prod}_i - \sum_j c_j \mathit{Waste}_j \\
&\quad \mathit{Prod}_i = \sum_j \mathit{Ship}_{i,j} \\
(8) \qquad &\quad \mathit{Prod}_i \leq \mathit{cap}_i \\
&\quad \sum_i \mathit{Ship}_{i,j} = \mathit{Sales}_j + \mathit{Waste}_j \\
&\quad \mathit{Sales}_j \leq \mathit{demand}_j \\
&\quad \mathit{Sales}_j \geq 0, \mathit{Ship}_{i,j} \geq 0, \mathit{Prod}_i \geq 0, \mathit{Waste}_j \geq 0
\end{aligned}
$$

In this model we try to maximize profit, i.e. revenue minus costs, where costs can be broken down in production costs, transportation costs and waste disposal costs. This model is just a little bit more complex than the transportation model.

When we make $demand_j$ stochastic, the problem becomes more involved. We assume there are three scenarios: pessimistic, average and optimistic. The deterministic equivalent of the stochastic model could look like:

$$
\begin{aligned}
\max \mathit{Profit} = & \sum_{j,\omega} p_j prob_\omega \mathit{Sales}_{j,\omega} - \sum_{i,j} c_{i,j} \mathit{Ship}_{i,j} \\
& - \sum_i c_i \mathit{Prod}_i - \sum_{j,\omega} c_j prob_\omega \mathit{Waste}_{j,\omega} \\
& \mathit{Prod}_i = \sum_j \mathit{Ship}_{i,j} \\
& \mathit{Prod}_i \le cap_i \\
& \sum_i \mathit{Ship}_{i,j} = \mathit{Sales}_{j,\omega} + \mathit{Waste}_{j,\omega} \\
& \mathit{Sales}_{j,\omega} \le demand_j \\
& \mathit{Sales}_{j,\omega} \ge 0, \mathit{Ship}_{i,j} \ge 0, \mathit{Prod}_i \ge 0, \mathit{Waste}_{j,\omega} \ge 0
\end{aligned}
\tag{9}
$$

The following model implements the core model and the deterministic equivalent formulation in GAMS:

3.0.1. *Model stochlp.gms.* [1]

```
$ontext

   Transportation problem with stochastic demands.
   Simple three scenario problem.

   Erwin Kalvelagen, January 2003


$offtext

sets
  i 'factories' /f1*f3/
  j 'distribution centers' /d1*d5/
  s 'scenarios' /lo,mid,hi/
;

parameter capacity(i) /f1 500, f2 450, f3 650/;

table demand(j,s) 'possible outcomes for demand'
      lo  mid   hi
  d1 150  160  170
  d2 100  120  135
  d3 250  270  300
  d4 300  325  350
  d5 600  700  800
;

parameter prob(s) 'probabilities' /
   lo 0.25
   mid 0.5
   hi 0.25
/;

table transcost(i,j) 'unit transportation cost'
```

[1] http://www.amsterdamoptimization.com/models/stochbenders/stochlp.gms

```
       d1    d2    d3    d4    d5
  f1   2.49  5.21  3.76  4.85  2.07
  f2   1.46  2.54  1.83  1.86  4.76
  f3   3.26  3.08  2.60  3.76  4.45
;

scalar prodcost  'unit production cost' /14/;
scalar price     'sales price' /24/;
scalar wastecost 'cost of removal of overstocked products' /4/;



*-----------------------------------------------------------------------
* first we formulate a non-stochastic version of the model
* we just use 'mid' values for the demand.
*-----------------------------------------------------------------------

variables
   ship(i,j)   'shipments'
   product(i)  'production'
   sales(j)    'sales (actually sold)'
   waste(j)    'overstocked products'
   profit
;
positive variables ship,product,sales,waste;

equations
   obj
   production(i)
   selling(j)
;


obj.. profit =e= sum(j, price*sales(j)) - sum((i,j), transcost(i,j)*ship(i,j))
                 - sum(j, wastecost*waste(j)) - sum(i,prodcost*product(i));


production(i).. product(i) =e= sum(j, ship(i,j));
product.up(i) = capacity(i);

selling(j).. sum(i, ship(i,j)) =e= sales(j)+waste(j);
sales.up(j) = demand(j,'mid');

model nonstoch /obj,production,selling/;
solve nonstoch maximizing profit using lp;

display ship.l,product.l,sales.l,waste.l;

*-----------------------------------------------------------------------
* now we formulate a stochastic version of the model
* We form here the deterministic equivalent
*-----------------------------------------------------------------------

variables
   salesw(j,s)  'stochastic version of sales'
   wastew(j,s)  'stochastic version of waste'
   received(j)  'amount of product received in distribution center'
;
positive variable salesw,wastew;

equations
   objw
   sellingw(j,s)
   receive(j)
;

objw.. profit =e= sum((j,s),price*prob(s)*salesw(j,s))
                 - sum((i,j), transcost(i,j)*ship(i,j))
                 - sum((j,s), wastecost*prob(s)*wastew(j,s))
                 - sum(i,prodcost*product(i));

receive(j)..     received(j) =e= sum(i, ship(i,j));
```

```
sellingw(j,s).. received(j) =e= salesw(j,s)+wastew(j,s);
salesw.up(j,s) = demand(j,s);

model stoch /objw,production,receive,sellingw/;
solve stoch maximizing profit using lp;

display ship.l,product.l,salesw.l,wastew.l,profit.l;
```

The optimal levels for the stage one variables from the deterministic model are:

```
----    113 VARIABLE ship.L  shipments

            d1          d2          d3          d4          d5

f1                                                     500.000
f2     150.000                             300.000
f3                 100.000     270.000                 100.000
```

4. Benders formulation

This model is quite easily to solve using Benders' Decomposition in GAMS. A complete formulation is given below.

4.0.2. *Model stochbenders.gms.* [2]

```
$ontext

    Benders decomposition applied to a two-stage
    stochastic linear programming problem.

    Erwin Kalvelagen, januari 2003


$offtext


sets
  i 'factories' /f1*f3/
  j 'distribution centers' /d1*d5/
  s 'scenarios' /lo,mid,hi/
;

parameter capacity(i) /f1 500, f2 450, f3 650/;

table demand(j,s) 'possible outcomes for demand'
      lo  mid   hi
  d1 150  160  170
  d2 100  120  135
  d3 250  270  300
  d4 300  325  350
  d5 600  700  800
;

parameter prob(s) 'probabilities' /
   lo 0.25
   mid 0.5
   hi 0.25
/;

table transcost(i,j) 'unit transportation cost'
       d1    d2    d3    d4    d5
  f1   2.49  5.21  3.76  4.85  2.07
  f2   1.46  2.54  1.83  1.86  4.76
  f3   3.26  3.08  2.60  3.76  4.45
;
```

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

```
scalar prodcost  'unit production cost' /14/;
scalar price     'sales price' /24/;
scalar wastecost 'cost of removal of overstocked products' /4/;


*---------------------------------------------------------------------
* Form the Benders master problem
*---------------------------------------------------------------------

set
   iter 'max Benders iterations' /iter1*iter25/
   dyniter(iter) 'dynamic subset'
;

positive variables
   ship(i,j)        'shipments'
   product(i)       'production'
   slackproduct(i)  'slack'
   received(j)      'quantity sent to market'
;
free variables
   zmaster          'objective variable of master problem'
   theta            'extra term in master obj'
;

equations
   masterobj        'master objective function'
   production(i)    'calculate production in each factory'
   receive(j)       'calculate quantity to be send to markets'
   prodcap(i)       'production capacity'
   optcut(iter)     'Benders optimality cuts'
;

parameter
   cutconst(iter)   'constants in optimality cuts'
   cutcoeff(iter,j) 'coefficients in optimality cuts'
;

masterobj..
    zmaster =e=  sum((i,j), transcost(i,j)*ship(i,j))
                + sum(i,prodcost*product(i)) + theta;

receive(j)..       received(j) =e= sum(i, ship(i,j));
production(i)..    product(i) =e= sum(j, ship(i,j));
prodcap(i)..       product(i) + slackproduct(i) =e= capacity(i);
optcut(dyniter)..  theta =g= cutconst(dyniter) +
                              sum(j, cutcoeff(dyniter,j)*received(j));

model masterproblem /masterobj, receive, production, prodcap, optcut/;

*---------------------------------------------------------------------
* Form the Benders' subproblem
* Notice in equation selling we use the level value received.l, i.e.
* this is a constant
*---------------------------------------------------------------------

positive variables
   sales(j)         'sales (actually sold)'
   waste(j)         'overstocked products'
   slacksales(j)    'slack'
;
free variables
   zsub             'objective variable of sub problem'
;

equations
   subobj           'subproblem objective function'
   selling(j)       'part of received is sold'
   selmax(j)        'upperbound on sales'
;

parameter demnd(j) 'demand';
```

```
subobj..
   zsub =e= -sum(j, price*sales(j)) + sum(j, wastecost*waste(j));

selling(j)..  sales(j) + waste(j) =e= received.l(j);
selmax(j)..   sales(j) + slacksales(j) =e= demnd(j);


model subproblem /subobj,selling,selmax/;


*-------------------------------------------------------------------
* solver options
*-------------------------------------------------------------------

option limrow = 0;
option limcol = 0;
subproblem.solprint = 2;
masterproblem.solprint = 2;
subproblem.solvelink = 2;
masterproblem.solvelink = 2;

*-------------------------------------------------------------------
* Benders algorithm
*-------------------------------------------------------------------


*
* step 1: solve master without cuts
*

display "-----------------------------------------------------------------",
        "Master without cuts",
        "-----------------------------------------------------------------";
dyniter(iter) = NO;
cutconst(iter) = 0;
cutcoeff(iter,j) = 0;
theta.fx = 0;
solve masterproblem minimizing zmaster using lp;
display zmaster.l;



*
* repair bounds
*
theta.lo = -INF;
theta.up = INF;

scalar lowerbound /-INF/;
scalar upperbound /INF/;
parameter objsub(s);
scalar objmaster;
objmaster = zmaster.l;

scalar iteration;
scalar done /0/;

loop(iter$(not done),

   iteration = ord(iter);
   display "-----------------------------------------------------------------",
           iteration,
           "-----------------------------------------------------------------";

*
* solve subproblems
*
   dyniter(iter) = yes;
   loop(s,
       demnd(j) = demand(j,s);
```

```
      solve subproblem minimizing zsub using lp;
      objsub(s) = zsub.l;
      cutconst(iter) = cutconst(iter) - prob(s)*sum(j,(-selmax.m(j))*demand(j,s));
      cutcoeff(iter,j) = cutcoeff(iter,j) - prob(s)*(-selling.m(j));
   );

   upperbound = min(upperbound, objmaster + sum(s, prob(s)*objsub(s)));

*
* convergence test
*
   display lowerbound,upperbound;
   if( (upperbound-lowerbound) < 0.001*(1+abs(lowerbound)),

      display  "Converged";
      done = 1;

   else

*
* solve masterproblem
*
      solve masterproblem minimizing zmaster using lp;
      display ship.l;

      lowerbound = zmaster.l;
      objmaster = zmaster.l-theta.l;

   );

);


abort$(not done) "Too many iterations";

display "-----------------------------------------------------------------",
        "Optimal stage one solution",
        "-----------------------------------------------------------------",
        ship.l;
```

The optimal levels for the stage one variables from this formulation are:

```
----    221 VARIABLE ship.L  shipments

            d1          d2          d3          d4          d5

f1                                                     500.000
f2     150.000                                 300.000
f3                 100.000     270.000                 100.000
```

5. Dual subproblem

In the model above we used a primal subproblem and its marginals (duals) in the calculation of the cuts. We can also solve a dual subproblem directly. Such a problem can be formulated as:

```
*-------------------------------------------------------------------
* Dual subproblem
*-------------------------------------------------------------------

variables
   pi_selling(j)    'dual of equation selling'
   pi_selmax(j)     'dual of equation selmax'
;

equations
   dualobj
   dualcon(j)
;
```

```
dualobj..
   zsub =e= sum(j, received.l(j)*pi_selling(j)) + sum(j, demnd(j)*pi_selmax(j));

dualcon(j).. pi_selling(j) + pi_selmax(j) =l= -price;

pi_selling.up(j) = wastecost;
pi_selmax.up(j) = 0;

model dualsubproblem /dualobj,dualcon/;
```

The subproblem solution step now becomes:

```
*
* solve subproblems
*
   dyniter(iter) = yes;
   loop(s,
       demnd(j) = demand(j,s);
       solve dualsubproblem maximizing zsub using lp;
       objsub(s) = zsub.l;
       cutconst(iter) = cutconst(iter) - prob(s)*sum(j,(-pi_selmax.l(j))*demand(j,s));
       cutcoeff(iter,j) = cutcoeff(iter,j) - prob(s)*(-pi_selling.l(j));
   );
```

This problem is rather simple and can even be solved directly without invoking an LP solver:

```
   loop(j,
      if (received.l(j)>demnd(j),
         pselling(j) = wastecost;
         pselmax(j) = -price-pselling(j);
      else
         pselling(j) = -price;
         pselmax(j) = 0;
      );
   );
```

A slightly different variant of this problem is described in [5]: there the probabilities are considered independent leading to 243 scenarios. This paper also implements the last suggestion, to solve the subproblems directly inside GAMS instead of as using an LP problem.

References

1. J. F. Benders, *Partitioning procedures for solving mixed-variables programming problems*, Numerische Mathematik **4** (1962), 238–252.
2. John R. Birge and François Louveaux, *Introduction to Stochastic Programming*, Springer Series in Operations Research, Springer, 1997.
3. Gerd Infanger, *Planning Under Uncertainty – Solving Large-Scale Stochastic Linear Programs*, Boyd & Fraser, 1994.
4. Peter Kall and Stein W. Wallace, *Stochastic Programming*, Wiley, 1994.
5. Erwin Kalvelagen, *Two Stage Stochastic Linear Programming with GAMS*, `http://www.amsterdamoptimization.com/pdf/twostage.pdf`.
6. S. S. Nielsen and S.A. Zenios, *Scalable Parallel Benders Decomposition for Stochastic Linear Programming*, Parallel Computing **23** (1997), 1069–1088.
7. R. M. van Slyke and R. J.-B. Wets, *L-Shaped Linear Programs with Applications to Optimal Control and Stochastic Programming*, SIAM Journal on Applied Mathematics **17** (1969), 638–663.

Amsterdam Optimization Modeling Group LLC, Washington D.C./The Hague
E-mail address: `erwin@amsterdamoptimization.com`