On the role of bounds
in stochastic linear programming

P. Kall and J. Mayer
IOR, University of Zurich

Abstract

Stochastic linear programming (SLP) models involve multivariate integrals. Although in the discretely distributed case these integrals become sums they typically contain a large amount of terms. The purpose of this paper is twofold: On the one hand we discuss the usage of bounds concerning integrals for constructing SLP algorithms and secondly we point out the role of bounds–based algorithms for solving SLP problems. The conceptual considerations are demonstrated in the last section by computational results. The tests have been carried out by utilizing SLP–IOR, our model management system for SLP.

Keywords: Stochastic programming; Successive discrete approximation; Boole–Bonferroni bounds; Testing of solvers; Model management systems

1 Stochastic linear programming models

In this section we summarize the stochastic linear programming (SLP) model types which will be considered in the paper. For a detailed description including basic properties see e.g. Kall and Wallace [20].

Two stage fixed recourse models can be formulated as follows:

\[
\begin{align*}
\min & \{c^T x + E_\omega Q(x, \omega)\} \\
\text{s.t.} & \quad Ax = b \\
& \quad x \geq 0,
\end{align*}
\]

where

\[
\begin{align*}
Q(x, \omega) &= \min q^T(\omega)y \\
\text{s.t.} & \quad Wy = h(\omega) - T(\omega)x \\
& \quad y \geq 0.
\end{align*}
\]
with $A$ being an $m_1 \times n_1$ and $W$ an $m_2 \times n_2$ matrix and the remaining arrays dimensioned accordingly; $\omega \in \Omega$ with $(\Omega, \mathcal{F}, P)$ being a probability space.

Problem (2) is called the second stage or recourse problem, matrix $W$ is the recourse matrix. The model (1) is a complete recourse problem if $\{z \mid z = W y, y \geq 0\} = \mathbb{R}^{m_2}$ and $\{u \mid W^T u \leq q(\omega)\} \neq \emptyset$ w.p. 1 hold. A special case of complete recourse is simple recourse where $W = (I, -I)$, $T(\omega) \equiv T$ and $q(\omega) \equiv q$ are fulfilled. We will only consider complete and simple recourse problems with $q(\omega) \equiv q$. The stochastic entries are modeled via the following affine sums:

$$h(\omega) = h^0 + \sum_{j=1}^{r} h^j \xi_j(\omega), \quad T(\omega) = T^0 + \sum_{j=1}^{r} T^j \xi_j(\omega) \tag{3}$$

with $\xi$ being a random vector on a probability space $(\Omega, \mathcal{F}, P)$.

An important class of two stage models with recourse consists of two stage models with a finite discrete distribution. In this special case $\xi$ has a finite discrete distribution and then problem (1) can equivalently be formulated as the following deterministic LP problem with a dual block–angular structure:

$$\begin{align*}
\min \left[ c^T x + p_1 q^T y^1 + \ldots + p_L q^T y^L \right] \\
A x & = b \\
T^1 x + W y^1 & = h^1 \\
\vdots & \quad \vdots \\
T^L x + W y^L & = h^L \\
x & \geq 0 \\
y^i & \geq 0 \quad \forall i.
\end{align*} \tag{4}$$

where $T^i$ and $h^i$ are the realizations and $p_i$ the corresponding probabilities, $i = 1, \ldots, L$, of the random vector $\xi$. With $x$ being fixed (4) decomposes into $L$ subproblems. We will call these subproblems diagonal subproblems. Notice that the number of the diagonal subproblems equals the number of realizations of $\xi$.

**Jointly chance constrained models (models with probabilistic constraints)** can be formulated as

$$\begin{align*}
\min \quad c^T x \\
P(\{\omega \mid T x \geq h(\omega)\}) & \geq \alpha \\
A x & = b \\
x & \geq 0 \tag{5}
\end{align*}$$
with $0 < \alpha < 1$ being some (high) probability level. Notice that only the RHS is stochastic. This assumption is needed to ensure the convexity of (5) for a broad class of multivariate distributions including multinormal, see Kall and Wallace [20] or Prékopa [25]. We will assume that $h(\omega)$ has a multinormal distribution. The joint chance constraint can obviously be replaced by

$$F(y) \geq \alpha$$
$$Tx - y \geq 0$$

with $F(y)$ being the joint probability distribution function of $h(\omega)$.

2 Numerical difficulties

Both type of problems involve multivariate integrals which makes them difficult to solve numerically.

For two stage recourse problems this integral appears in the objective function in the expectation term as $E_{\omega}Q(x, \omega)$ with $Q(x, \omega)$ being the optimal value function of an LP with a RHS parametrized by a linear affine function of $x$.

As mentioned above (1) can equivalently be formulated in the discretely distributed case as a deterministic LP problem. The existence of the equivalent LP does not imply however that solving (1) simply reduces to solving the LP problem (4) by some LP algorithm which perhaps utilizes the structure. To see this let us consider the case when we have 6 independent random variables ($r = 6$). If each of them has 5 realizations, the number of diagonal blocks will be $L = 5^6 = 15625$ which may already result in an LP (4) intractable on personal computers by LP techniques. Taking 10 independent random variables with each of them having 5 realizations amounts in $L = 5^{10} = 9765625$. Proceeding this way it is easy to conceive realistic problem formulations where due to the combinatorial explosion LP techniques become infeasible even on large mainframe computers. This remains true if we take into account computer developments in a foreseeable future.

For chance constrained problems the chance constraint involves the joint probability distribution function $F(y) = P(\{\omega \mid h(\omega) \leq y\})$ i.e. $F(y)$ is in our case given as a multivariate integral involving the density function of the multivariate normal distribution.

The main idea pursued in this paper is the usage of bounds on these integrals for the construction of algorithms. In successive discrete approximation methods for complete recourse problems the computation of $E_{\omega}Q(x, \omega)$ can be avoided completely. For chance constrained problems the computational time is largely
reduced by employing bounds in the linesearch procedures. In the paper we only consider exact bounds; stochastic bounds have been given by Norkin, Ermoliev and Ruszczyński [24].

3 Algorithmic approaches

In this section we give an outline of the main approaches for solving SLP problems. To each approach we list some of the algorithms belonging to that class; a more complete listing is beyond the scope of this paper. For the citations not explicitly given here see the references in [20] and [25].

- Two stage problems with a finite discrete distribution: The first approach for these type of problems consists of solving the equivalent LP (4) by algorithms which utilize the structure. Algorithms belonging to this class are e.g. the L–shaped method, Van Slyke and Wets (1969), Birge and Louveaux (1988), Gassmann (1990); regularized decomposition, Ruszczyński (1986), Ruszczyński and Święatanowski (1997) [26]; and interior point methods, Mészáros (1997) [23].

This approach is clearly constrained by available computer capacity to problems with “not too many” joint realizations (scenarios). From this point of view basically two types of solvers can be identified in the class.

The first type of solvers requires data for the explicitly formulated problem (4) as a large–scale LP (e.g. IP methods). In this case available storage capacity sets a bound on the problems which can be set up and solved (independently on the fact how good the solver otherwise is). To see the point let us consider the case where the components of the random vector $\xi$ are stochastically independent. For setting up the LP (4) all of the joint realizations of $\xi$ are to be generated. Thus the combinatorial explosion already enters the problem setup phase.

With the second type of solvers (decomposition methods) this difficulty can be overcome by using the S–MPS input dataformat [1], [10] which provides a facility for specifying problem data for (1) without explicitly formulating (4). In particular when $\xi$ has stochastically independent components the distribution is specified componentwise. This approach however also involves a serious “built in” constraint on the size of problems which can be solved by it. The decomposition methods belonging to this class require solving of all diagonal subproblems several times during a run. This implies that all diagonal subproblems must explicitly be set up and be dealt
with, i.e. in particular all joint realizations of $\xi$ must again be generated. This sets a bound on problem size according to available computing speed (and possibly also storage capacity).

- For two stage problems with a discrete distribution and “many” realizations or with a continuous distribution the main approaches are the following:

In this paper we discuss bounds–based algorithms. An outstanding feature of these methods is that at each iteration exact lower and upper bounds on the optimal objective value of problem (1) are available. The size of solvable problems for this approach is essentially constrained by the dimension of the random vector $\xi$; the number of joint realizations plays a subordinate role. This feature identifies those two stage problems with complete recourse for which the bounds–based approach is best suited.

- Chance constrained problems:

4 Bounds

In this section we summarize the usage of bounds. We keep the discussion short; the interested reader may find a detailed discussion of the Boole–Bonferroni bounds in [20], [25] and of bounds concerning recourse problems and successive discrete approximation schemes in [2], [3], [4], [5], [15], [16], [18] and [20].

Boole–Bonferroni bounds on $F(x)$
Let $A_i = \{ \omega \mid h_i(\omega) \leq x_i \}$ and $B_i = A_i^c = \Omega \setminus A_i, \forall i$. The distribution function can be expressed as

$$F(x) = P(\bigcap_{i=1}^{r} A_i) = 1 - P(\bigcup_{i=1}^{r} B_i)$$

The bounds are based on binomial moments. The $k$’th binomial moment is

$$S_{k,r} = \sum_{1 \leq i_1 \leq \ldots \leq i_k \leq r} P(B_{i_1} \cap \ldots \cap B_{i_k})$$

for $k \geq 1$

The basic Boole–Bonferroni bounds are in terms of first and second binomial moments:

$$1 - S_{1,r} + \frac{2}{r} S_{2,r} \leq F(x) \leq 1 - \frac{2}{k+1} S_{1,r} + \frac{2}{k(k+1)} S_{2,r}$$

with $k = \lceil \frac{2S_{2,r}}{S_{1,r}} \rceil + 1$.

For sharper bounds (e.g. Hunter’s bound) see [25].

**Lower bounds on $Q(x, \xi)$**

The lower bounds are based on the Jensen–inequality. Let $\Xi$ be an interval approximation of the support of $\xi$ possibly resulting from truncation and let us consider a partition of $\Xi$ into subintervals:

$$\Xi = \prod_{i=1}^{r} [a_i, b_i] \quad \Xi = \bigcup_{l=1}^{L} \Xi_l; \quad \Xi_l \cap \Xi_k = \emptyset (l \neq k)$$

We denote by $C = \sigma(\Xi_1, \ldots, \Xi_L)$ the $\sigma$–algebra generated by the partition and let

$$p_l = P(\{ \omega \mid \xi(\omega) \in \Xi_l \}), \quad \tilde{\xi}^l = E_{\Xi_l}(\xi) = \frac{1}{P(\Xi_l)} \int_{\Xi_l} \xi P(d\xi), \forall l.$$ 

Then we have—with $E^C$ the conditional expectation with respect to $C$—by Jensen’s inequality:

$$Q(x, E^C(\xi)) \leq E^C(Q(x, \xi)).$$

which results in a lower bound given by the optimal objective value of the following LP:

$$\min c^T x + \sum_{l=1}^{L} p_l Q(x, \tilde{\xi}^l)$$

$$\begin{array}{ll}
Ax & = b \\
x & \geq 0,
\end{array}$$

6
It is easy to see that a refined partition results in a nondecreasing lower bound.

**Upper bounds on** $Q(x, \xi)$

These bounds are based on the Edmundson–Madansky inequality. In the one-dimensional case ($r = 1$) the bound is the following:

$$\int_{\xi_l} Q(x, \xi) P(d\xi) = \int_{a_l}^{a_{l+1}} Q(x, \xi) P(d\xi) \leq p_l \left( \frac{a_{l+1} - \xi_l}{a_{l+1} - a_l} Q(x, a_l) + \frac{\xi_l - a_l}{a_{l+1} - a_l} Q(x, a_{l+1}) \right)$$

For $r > 1$ and $\xi_1, \ldots, \xi_r$ stochastically independent, successive integration (Fubini) gives the upper bound. For the dependent case a similar argument gives an upper bound, which however involves all moments. For $x$ fixed, a refined partition results in a nonincreasing upper bound.

For the case of simplices instead of intervals Frauendorfer [8] gave upper bounds; for general polyhedral domains Edirisinghe and Ziemba [4] have developed bounds.

**Usage of bounds in SLP algorithms**

*For two stage recourse problems* the bounds are utilized for constructing successive discrete approximation schemes by successively refining the partition. The refinement strategies have two main components. The first one is based on the bounds computed for the cells separately and concerns choosing the cells which should be further partitioned. The second component of the strategy is for choosing the coordinate according to which the cut of a chosen cell should be carried out. Computing the lower bound involves solving an LP with a dual block angular structure. The number of joint realizations (diagonal blocks) is equal to the number of subintervals (cells) in the current partition of $\Xi$. For solving these LP’s an efficient hot start procedure has been given by Kall and Stoyan [19]. Computing the upper bound requires to solve an LP of the size of the diagonal block LP’s in (4) (i.e. the size of $W$), in turn for every vertex in the current partition.

*For jointly chance constrained algorithms* the bounds are used within NLP algorithmic frameworks in the linesearch procedures. Bisection–type procedures are employed with a combination of computing bounds and function values, where the latter are only computed when interval reduction based on bounds is no more possible. Computing the bounds amounts in computing the distribution function of the one– and two dimensional marginal distributions. With a multinormal distribution the marginal distributions are also normal and the bounds can be
computed efficiently.

5 Test runs

The test runs have been carried out on an IBM/PC 400MHz Pentium computer with 128 MB storage. The testing environment has been SLP–IOR, our model management system for SLP, see Kall and Mayer [17]. For a detailed description of the solvers used in the tests see Mayer [22].

The first three tests concern complete recourse problems with a finite discrete distribution and an increasing number of realizations. The goal is to show the limitations of the LP–based decomposition approach and to demonstrate that the bounds–based algorithms still work quite efficiently for problems with a large amount of joint realizations.

Test #1 We consider complete recourse problems which are well solvable in our computing environment both with regularized decomposition and successive discrete approximation.

The battery ZB3, generated by SLP–IOR, contains 10 randomly generated complete recourse problems. \( A \) and \( W \) have dimensions \( 20 \times 40 \) and \( 6 \times 12 \), respectively. The nonzero densities for \( c, A, b \) are 100\%, 30\%, 100\%, respectively. The densities of \( T, h, W, q \) have been chosen as 40\%, 100\%, 40\%, 50\%.

The RHS \( h \) and \( T \) are stochastic, we took 6 standard normally distributed random variables. The arrays in the affine sums (3) have 30\% densities of the nonzero entries. This results altogether in 78 highly dependent random entries in (2).

Subsequently we discretized the distribution by utilizing SLP–IOR; we took 3 realizations for each component, i.e. we have \( 3^6 = 729 \) joint realizations. The equivalent LP (4) has the size \( 4394 \times 8788 \) with 124171 nonzeros.

We have solved the problems with DAPPROX, which is our implementation of the successive discrete approximation scheme, and with the regularized decomposition solver QDECOM, implemented by A. Ruszczyński (1985).

Remark: The regularized decomposition method has been substantially improved since 1985 by Ruszczyński and Świątanoski, see [26]. The authors have implemented their method as the solver DECOMP. The reason why we used QDECOM in our tests is the following: DECOMP requires the input data in the standard S–MPS dataformat [1] which does not allow for the affine sums (3). We prefer however to carry out our tests with test batteries involving (3) because these kind
of problems proved to be much harder from the numerical point of view. Notice that in our batteries we have 76 dependent random variables in the problems. Let us point out an important new development concerning standard dataformats: Gassmann and Schweitzer [10] have developed a substantial extension of S–MPS which among several new features also accounts for the affine sums (3).

The solution with DAPPROX has been computed with 0.05 relative accuracy concerning bounds. The results are shown in Table 1. The upper part in this Tableau contains the optimal objective values for QDECOM and lower bounds for DAPPROX. The lower part displays computing time in seconds. The results show a similar performance for both solvers.

Test #2 Next we increased the number of realizations by slightly increasing the number of realizations for each of the random variables $\xi_i$.

The battery ZB5 is the same as in ZB3 consequently the problems have the same characteristics as those in Test #1. The only difference is that we now discretized with 5 realizations for each component. We have 15625 joint realizations and an LP equivalent (4) of the size $93770 \times 187500$ and $\sim 2650000$ nonzeros.

These problems proved to be quite hard for QDECOM: The average computing time was $\sim 1600$ seconds. As our primary goal in this part of tests is not the investigation of comparative solver efficiency we did not include the detailed computational times for QDECOM in the paper. Table 2 just displays the results with DAPPROX; the results show that the problems in ZB5 can still quite efficiently be solved by DAPPROX.

Test #3 This battery differs from ZB5 only in the discretization: now we discretized with 7 realizations for each component. We have now 117649 joint realizations and an LP equivalent (4) of the size $705914 \times 1411828$ with $\sim 14824000$ nonzeros.

These problems proved to be too large for QDECOM, the available storage was insufficient for setting up the working arrays of the solver. The second part of Table 2 displays the results with DAPPROX; the problems in ZB7 are still quite efficiently be solvable by DAPPROX.

Test #4 The goal of this test is to show the performance of successive discrete approximation algorithms in the simple recourse case.

The battery ZSMUN contains 10 randomly generated simple recourse problems. $A$ $100 \times 200$, $W$ $50 \times 100$; the nonzero densities were 10% throughout (except for $W = (I, -I)$). Only the RHS is stochastic, we have 50 normally distributed
### Table 1: Test #1: complete recourse problems

<table>
<thead>
<tr>
<th></th>
<th>DAPPROX</th>
<th>QDECOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZB3-1</td>
<td>12.22393</td>
<td>12.54898</td>
</tr>
<tr>
<td>ZB3-2</td>
<td>3.604689</td>
<td>3.695223</td>
</tr>
<tr>
<td>ZB3-3</td>
<td>39.08614</td>
<td>39.99893</td>
</tr>
<tr>
<td>ZB3-4</td>
<td>11.22241</td>
<td>11.52075</td>
</tr>
<tr>
<td>ZB3-5</td>
<td>12.97690</td>
<td>13.43307</td>
</tr>
<tr>
<td>ZB3-6</td>
<td>11.85668</td>
<td>12.20840</td>
</tr>
<tr>
<td>ZB3-7</td>
<td>14.44288</td>
<td>14.69232</td>
</tr>
<tr>
<td>ZB3-8</td>
<td>2.900206</td>
<td>2.999599</td>
</tr>
<tr>
<td>ZB3-9</td>
<td>25.15192</td>
<td>25.81004</td>
</tr>
<tr>
<td>ZB3-10</td>
<td>13.98828</td>
<td>14.21381</td>
</tr>
</tbody>
</table>

- **min** | 6.48 | 4.29 |
- **max** | 23.13 | 18.24 |
- **mean** | 14.31 | 10.33 |
- **sdev** | 6.20 | 3.66 |

### Table 2: Tests #2 and #3: complete recourse problems

<table>
<thead>
<tr>
<th></th>
<th>DAPPROX</th>
<th>DAPPROX</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZB5-1</td>
<td>14.16345</td>
<td>ZB7-1</td>
</tr>
<tr>
<td>ZB5-2</td>
<td>3.945236</td>
<td>ZB7-2</td>
</tr>
<tr>
<td>ZB5-3</td>
<td>42.19052</td>
<td>ZB7-3</td>
</tr>
<tr>
<td>ZB5-4</td>
<td>12.80241</td>
<td>ZB7-4</td>
</tr>
<tr>
<td>ZB5-5</td>
<td>14.66621</td>
<td>ZB7-5</td>
</tr>
<tr>
<td>ZB5-6</td>
<td>12.83067</td>
<td>ZB7-6</td>
</tr>
<tr>
<td>ZB5-7</td>
<td>16.16763</td>
<td>ZB7-7</td>
</tr>
<tr>
<td>ZB5-8</td>
<td>3.174562</td>
<td>ZB7-8</td>
</tr>
<tr>
<td>ZB5-9</td>
<td>26.61366</td>
<td>ZB7-9</td>
</tr>
<tr>
<td>ZB5-10</td>
<td>15.75017</td>
<td>ZB7-10</td>
</tr>
</tbody>
</table>

- **min** | 22.35 | 81.35 |
- **max** | 152.09 | 1183.7 |
- **mean** | 67.50 | 381.9 |
- **sdev** | 37.45 | 343.0 |

10
random variables with standard deviation chosen as 10% of the magnitude of the corresponding RHS component. The arrays in the affine sums (3) are unit arrays.

Table 3 shows the results with SRAPPROX, our implementation of the discrete approximation scheme for simple recourse. Due to the special structure the recourse objective is separable in $\chi = Tx$ which allows for an efficient implementation. The results illustrate that simple recourse problems can quite efficiently be solved even with a large amount of random variables in the model. Notice that even a rough discretization with 3 realizations for each component would amount in $3^{50}$ joint realizations, which is clearly far beyond the scope of solvers based on the LP equivalent (4), and this will remain so in a foreseeable future.

\textit{Test #5} The final test intends to show the performance of solvers utilizing the bounds, for jointly chance constrained problems.

The battery ZJMUN consists of 10 randomly generated jointly chance constrained problems. A $50 \times 70$, the nonzero densities are 10% throughout. Only the RHS is stochastic having an 8-dimensional multinormal distribution with randomly generated correlation matrices. The standard deviations are 10% of the magnitude of the corresponding RHS ($h^0$) entries. The probability level is $\alpha = 0.9$.

The results of the test runs are shown in Table 4. PCSPIOR is our implementation of a variant of Veinott’s cutting plane method whereas PROBALL is an implementation of a variant of the Elzinga–Moore central cutting plane method. The missing entry means that none of the solvers could solve ZJMUN8 due to difficulties in finding a starting feasible solution in the first stage. $F(x)$ and $\nabla F(x)$ as well as the Boole–Bonferroni bounds have been computed by utilizing the subroutine package PCSPNOR3 which is the new version of NORSUBS, both developed by Szántai [28].
Table 3: Test #4 ZSMUN: simple recourse problems

<table>
<thead>
<tr>
<th>ZSMUN</th>
<th>SRAPPROX</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZSMUN1</td>
<td>5.198125</td>
</tr>
<tr>
<td>ZSMUN2</td>
<td>3.959444</td>
</tr>
<tr>
<td>ZSMUN3</td>
<td>16.67837</td>
</tr>
<tr>
<td>ZSMUN4</td>
<td>22.54663</td>
</tr>
<tr>
<td>ZSMUN5</td>
<td>16.70969</td>
</tr>
<tr>
<td>ZSMUN6</td>
<td>12.69672</td>
</tr>
<tr>
<td>ZSMUN7</td>
<td>15.93974</td>
</tr>
<tr>
<td>ZSMUN8</td>
<td>27.31383</td>
</tr>
<tr>
<td>ZSMUN9</td>
<td>14.79209</td>
</tr>
<tr>
<td>ZSMUN10</td>
<td>13.87318</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>min</th>
<th>max</th>
<th>mean</th>
<th>sdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>2.03</td>
<td>2.63</td>
<td>2.35</td>
<td>0.17</td>
</tr>
</tbody>
</table>

Table 4: Test #5 ZJMUN: jointly chance constrained problems

<table>
<thead>
<tr>
<th>ZJMUN</th>
<th>PCSPIOR</th>
<th>PROBALL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZJMUN1</td>
<td>2133.891</td>
<td>2133.883</td>
</tr>
<tr>
<td>ZJMUN2</td>
<td>3707.396</td>
<td>3707.396</td>
</tr>
<tr>
<td>ZJMUN3</td>
<td>4947.498</td>
<td>4947.516</td>
</tr>
<tr>
<td>ZJMUN4</td>
<td>874.7920</td>
<td>874.7912</td>
</tr>
<tr>
<td>ZJMUN5</td>
<td>3039.521</td>
<td>3039.507</td>
</tr>
<tr>
<td>ZJMUN6</td>
<td>774.3646</td>
<td>774.3635</td>
</tr>
<tr>
<td>ZJMUN7</td>
<td>1924.839</td>
<td>1924.839</td>
</tr>
<tr>
<td>ZJMUN8</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>ZJMUN9</td>
<td>3578.791</td>
<td>3578.791</td>
</tr>
<tr>
<td>ZJMUN10</td>
<td>4981.208404</td>
<td>4981.223997</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>min</th>
<th>max</th>
<th>mean</th>
<th>sdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>3.30</td>
<td>19.67</td>
<td>8.51</td>
<td>4.92</td>
</tr>
<tr>
<td>max</td>
<td>0.94</td>
<td>13.62</td>
<td>5.88</td>
<td>4.08</td>
</tr>
</tbody>
</table>
References


