Optimisation under uncertainty: software tools for modelling and solver support

Christian Valente, Gautam Mitra, Victor Zverovich
 Agenda

- Background and motivation
- Modelling & computational architecture
- SAMPL:
  - Stochastic Programming
  - Chance Constraints
  - Integrated Chance Constraints
  - Robust Optimization
- Solver support
- Conclusions
Positioning of Tools

• Introduction
• We are preoccupied with tools...
• Financial analytics tools..as important as finance itself...!
• View SP today as at the beginning of Gold Rush of analytics...!!
Donald Rumsfeld sets out uncertainty.........!!
Analytics comes of age: modelling paradigms

- Donald Rumsfeld’s utterances:
- **Known Known**...deterministic world{ Time tableing , TV advert scheduling...air line scheduling }
- **Known Unknown**... uncertain world ...but can model uncertainty >> scenario generation{ All BFSI applications supply chain logistics, energy systems....}
- **Unknown Unknown**... information is not revealed

Optimisation = Decision Making

Optimisation under uncertainty >> focus on BFSI
The Message

• The embedding of analytics within IS based solutions is becoming necessary...gaining acceptance

• Analytics = Application of (Models + Software realisations)

• In order to sell analytics it is necessary to understand the different categories of models and how these are integrated in a total solution.
Analytics comes of age: modelling paradigms

- *Descriptive Models* as defined by a set of mathematical relations, which simply predicts how a physical, industrial or a social system may behave.

- *Normative Models* constitute the basis for (quantitative) decision making by a superhuman following an entirely rational that is, logically scrupulous set of arguments. Hence quantitative decision problems and idealised decision makers are postulated in order to define these models.

- *Prescriptive Models* involve systematic analysis of problems as carried out by normally intelligent persons who apply intuition and judgement. Two distinctive features of this approach are uncertainty analysis and preference (or value or utility) analysis.

- *Decision Models* are in some sense a derived category as they combine the concept underlying the normative models and prescriptive models.
SP comes of age: scope of models

- Data Model
- Decision Model: Constrained optimisation
- Descriptive Model: Simulation and Evaluation

Ex ante decision
Ex post evaluation (simulation)
SP comes of age: decision...simulation...evaluation engine

Motivation
Architecture
SP
CCP
ICCP
Robust Optimisation
Solver
Conclusions

Ex-ante decision models
- Expected value LP
- Two-stage SP - recourse
- Multistage SP - recourse
- Chance-constrained SP
- Integrated chance constraints
- Robust optimisation

Models of (parameter) randomness
- Scenario Generator 1
- Scenario Generator 2

Simulation and Decision evaluation

Performance measures
- Statistical measures: mean, variance, skewness, kurtosis
- Stochastic measures: EVPI, VSS
- Risk measures: VaR, CVaR, standard deviation
- Performance measures: Solvency ratio, Sharpe ratio, Sortino ratio

Ex-ante decision models:
- Expected value LP
- Two-stage SP - recourse
- Multistage SP - recourse
- Chance-constrained SP
- Integrated chance constraints
- Robust optimisation

Models of (parameter) randomness:
- Scenario Generator 1
- Scenario Generator 2

Simulation and Decision evaluation:

Performance measures:
- Statistical measures: mean, variance, skewness, kurtosis
- Stochastic measures: EVPI, VSS
- Risk measures: VaR, CVaR, standard deviation
- Performance measures: Solvency ratio, Sharpe ratio, Sortino ratio
Research problems

• Mathematical Programming is a useful and widely used paradigm for decision making

• A few approaches are now established for quantitative decision making under uncertainty:
  – Dynamic Stochastic Programming
  – Stochastic Programming
  – Robust Optimisation

• These differ in respect of:
  – their formulation
  – assumptions and models for uncertain parameters
Focus of the development

• To simplify the formulation of SP and RO problems with a modelling language
• To facilitate the investigation of a given decision problem under alternative modelling frameworks
• To automate the steps necessary for their solution
• To connect scalable solution algorithms which match the models
SP modelling process

Modelling phase
- Which scenario generators?
  - Model(s) of randomness
- Which decision model?
  - Decision Model
- Which solution method?
  - Solution algorithm

Runtime phase
- SG Parameters
- Algebraic model
- Solver Controls

Scenario Generator -> Modelling System -> Solver
Modelling Languages

- Algebraic Modelling Languages conveniently express MPs in a format both easy to understand and that can be processed by a solver (i.e. AMPL, GAMS, AIMMS, MPL, OPL, ...)
- Both SP and RO problems have different requirements
  - in language constructs
  - in specification of parameters uncertainty
  - in interacting with the solver
- SAMPL extends AMPL adding syntax to support such problem classes
Supported problems classes

- Dynamic Stochastic Programming
- SP Problems
- Robust Optimisation
- Distribution Problems
- Recourse Problems
- Problems with Chance Constraints
- Problems with ICC
- Stochastic Measures: EVPI and VSS
  - Wait and See
  - Expected Value
  - Distribution based
  - Scenario based

Motivation Architecture SP CCP ICCP Robust Optimisation Solver Conclusions
Modelling Languages

- Sometimes it is not practical or not possible to find distributions of the random parameters -> no scenarios
- A robust model can be formulated using the Robust Optimisation framework

SCENARIO GENERATION
Modelling distribution of random parameters

DECISION MODEL
Optimum Allocation Modelling

UNCERTAINTY not revealed
Parameters defined by uncertainty sets (Soyster, Bertsimas and Sim, Ben-Tal and Nemirovski)

Stochastic Programming

Robust Optimisation
Introducing ALM Example

• ALM model:
  – individual has incomes to invest in the stock market
  – In various time periods, he has liabilities to match
  – Asset prices are uncertain

• Objective: maximise terminal wealth of investor

• Constraints:
  – Asset inventory constraints
  – Cash balance constraints (buys and sales subject to transaction costs)
# Example: entities

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Notation</th>
<th>Description</th>
<th>Range//Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indices</td>
<td>ASSETS</td>
<td>I</td>
<td>Assets classes</td>
<td>i = 1 ... I;</td>
</tr>
<tr>
<td></td>
<td>TIME</td>
<td>T</td>
<td>Time periods</td>
<td>t = 1 ... T;</td>
</tr>
<tr>
<td>Parameters</td>
<td>price</td>
<td>$P_{it}$</td>
<td>Price of asset $i$ at time period $t$</td>
<td>ASSETS,TIME,</td>
</tr>
<tr>
<td></td>
<td>liabilities</td>
<td>$L_t$</td>
<td>Liability at time period $t$</td>
<td>TIME</td>
</tr>
<tr>
<td></td>
<td>Incomes</td>
<td>$F_t$</td>
<td>Incomes at time period $t$</td>
<td>INCOMES</td>
</tr>
<tr>
<td></td>
<td>Tcost</td>
<td>$G$</td>
<td>Transaction cost as % of trade value</td>
<td></td>
</tr>
<tr>
<td></td>
<td>hold0</td>
<td>$H_0$</td>
<td>Initial holdings of each asset</td>
<td>ASSETS</td>
</tr>
<tr>
<td>Variables</td>
<td>hold</td>
<td>$H_{it}$</td>
<td>Quantity of assets $i$ to hold in time period $t$</td>
<td>ASSETS,TIME</td>
</tr>
<tr>
<td></td>
<td>sell</td>
<td>$S_{it}$</td>
<td>Quantity of assets $i$ to sell in time period $t$</td>
<td>ASSETS,TIME</td>
</tr>
<tr>
<td></td>
<td>buy</td>
<td>$B_{it}$</td>
<td>Quantity of assets $i$ to buy in time period $t$</td>
<td>ASSETS,TIME</td>
</tr>
</tbody>
</table>
Deterministic version

```plaintext
set ASSETS;
set TIME;
param NT;
param Tcost;
param liabilities{TIME};
param income{TIME};
param hold0{ASSETS};
param price{TIME, ASSETS};

var hold{TIME, ASSETS} >=0;
var buy{TIME, ASSETS} >=0;
var sell{TIME, ASSETS} >=0;

maximize wealth: sum{a in ASSETS} price[NT,a]*hold[NT,a];

subject to
sinv1{a in ASSETS}: hold[1,a]=hold0[a]+buy[1,a]-sell[1,a];
sinv2{a in ASSETS,t in 2..NT}:
    hold[t,a]=hold[t-1,a]+buy[t,a]-sell[t,a];

cashbalance{t in TIME}:
    (1-Tcost)*(sum{a in ASSETS}price[t,a]*sell[t,a]) + income[t]
    - (1+Tcost)*(sum{a in ASSETS}price[t,a]*buy[t,a]) >=
    liabilities[t];
```
## Example: entities stochastic

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Notation</th>
<th>Description</th>
<th>Range/Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indices</td>
<td>ASSETS</td>
<td>I</td>
<td>Assets classes</td>
<td>i = 1 ... I;</td>
</tr>
<tr>
<td></td>
<td>TIME</td>
<td>T</td>
<td>Time periods</td>
<td>t = 1 ... T;</td>
</tr>
<tr>
<td></td>
<td>SCENARIO</td>
<td>S</td>
<td>Scenarios</td>
<td>S = 1..NS;</td>
</tr>
<tr>
<td>Parameters</td>
<td>price</td>
<td>P_{its}</td>
<td>Price of asset i at time period t in each SCENARIO</td>
<td>ASSETS,TIME, SCENARIO</td>
</tr>
<tr>
<td></td>
<td>liabilities</td>
<td>L_t</td>
<td>Liability at time period t</td>
<td>TIME</td>
</tr>
<tr>
<td></td>
<td>Incomes</td>
<td>F_t</td>
<td>Incomes at time period t</td>
<td>INCOMES</td>
</tr>
<tr>
<td></td>
<td>Tcost</td>
<td>G</td>
<td>Transaction cost as % of trade value</td>
<td></td>
</tr>
<tr>
<td></td>
<td>hold0</td>
<td>H_0</td>
<td>Initial holdings of each asset</td>
<td>ASSETS</td>
</tr>
<tr>
<td>Variables</td>
<td>hold</td>
<td>H_{its}</td>
<td>Quantity of assets i to hold in time period t in each SCENARIO</td>
<td>ASSETS,TIME, SCENARIO</td>
</tr>
<tr>
<td></td>
<td>sell</td>
<td>S_{its}</td>
<td>Quantity of assets i to sell in time period t in each SCENARIO</td>
<td>ASSETS,TIME, SCENARIO</td>
</tr>
<tr>
<td></td>
<td>buy</td>
<td>B_{its}</td>
<td>Quantity of assets i to buy in time period t in each SCENARIO</td>
<td>ASSETS,TIME, SCENARIO</td>
</tr>
</tbody>
</table>
Introducing Scenarios

Deterministic (EV)

Multiple scenarios (WS)

Multiple scenarios (TSSP)
Non anticipativity

• Example: Variable hold(asset, time, scenario)

The variables of the first time period represent all the same decision -> Fix their values to be the same with a constraint (NON ANTICIPATIVITY)
Stochastic version AMPL

```AMPL
set SCENARIO;

param prob{SCENARIO};

param price{TIME, ASSETS, SCENARIO};

var hold{TIME, ASSETS, SCENARIO} >= 0;
var buy{TIME, ASSETS, SCENARIO} >= 0;
var sell{TIME, ASSETS, SCENARIO} >= 0;

maximize wealth: sum{a in ASSETS, s in SCENARIO} prob[s]* price[NT,a,s]*hold[NT,a,s];

subject to
Appropriate scenario indices in each constraint
And...
Non anticipativity:
NAHold{s in SCENARIO,a in ASSETS}:hold[1,a,s]=hold[1,a,1];
NABuy{s in SCENARIO,a in ASSETS}:buy[1,a,s]=buy[1,a,1];
NASell{s in SCENARIO,a in ASSETS}:sell[1,a,s]=sell[1,a,1];
```

And...
Stochastic version SAMPL

```plaintext
scenario set SCENARIO;
probability prob{SCENARIO};
random param price{TIME, ASSETS, SCENARIO};
var hold{TIME, ASSETS, SCENARIO} >=0;
suffix stage if t=1 then 1 else 2;
var buy{TIME, ASSETS, SCENARIO} >=0;
suffix stage if t=1 then 1 else 2;
var sell{TIME, ASSETS, SCENARIO} >=0;
suffix stage if t=1 then 1 else 2;
maximize wealth: sum{a in ASSETS, s in SCENARIO} prob[s]* price[NT,a,s]*hold[NT,a,s];
subject to
  Appropriate scenario indices in each constraint
  and...
  Non-anticipativity:
  NAHold{s in SCENARIO,a in ASSETS} hold[1,a,s]=hold[1,a,1];
  NABuy{s in SCENARIO,a in ASSETS} buy[1,a,s]=buy[1,a,1];
  NASell{s in SCENARIO,a in ASSETS} sell[1,a,s]=sell[1,a,1];
and.. no non-anticipativity!!
```

Motivation
Architecture
SP
CCP
ICCP
Robust Optimisation
Solver
Conclusions
Chance Constraints

- Introduced in Charnes and Cooper, 1959
- Constraint(s) must hold with probability $p_i$
Integrated Chance Constraints

- Introduced in Haneveld, 1986
- Expected violation of constraint(s) $\leq$ shortfall $\beta_i$
- Individual ICC
  $$E_\omega [\eta_i(x, \omega)^{-}] \leq \beta_i, \quad \beta_i \geq 0, i \in I$$
- Joint ICC
  $$E_\omega [\max_{i \in I} \eta_i(x, \omega)^{-}] \leq \beta_i, \quad \beta_i \geq 0$$

where $\eta_i(x, \omega)^{-}$ represents the violation that occurs in constraint $i$ under realisation $\omega$
(Integrated) Chance Constraints

subject to
cashbalance\{t \text{ in} \text{ TIME}\}:

\text{probability}\{s \text{ in} \text{ SCENARIO}:
(1 - Tcost)*(\text{sum}\{a \text{ in ASSETS} \text{ price}[t,a,s]\text{sell}[t,a,s]) + \text{income}[t]
-(1+Tcost)*(\text{sum}\{a \text{ in ASSETS} \text{ price}[t,a,s]\text{buy}[t,a,s]\} >=
\text{liab}[t]

} >= \text{reliability};

subject to
cashbalance\{t \text{ in} \text{ TIME}\}:

\text{expectation}\{s \text{ in} \text{ SCENARIO}\} (\text{liab}[t] + (1+Tcost)*(\text{sum}\{a \text{ in ASSETS} \text{ price}[t,a,s]\text{buy}[t,a,s]\}
\text{less}
((1-Tcost)*(\text{sum}\{a \text{ in ASSETS} \text{ price}[t,a,s]\text{sell}[t,a,s]) + \text{income}[t])

) <= \text{level};
Robust Optimisation

- Consider the linear optimisation problem:
  \[ Z = \max \; cx \]
  subject to \[ Ax \leq b \]
  \[ l \leq x \leq u \]

- The uncertainty model is the following:

For a particular row \( i \) of the matrix \( A \) let \( J_i \) represent the set of coefficients in row \( i \) that are subject to uncertainty. Each entry \( a_{ij}, j \in J_i \) is modelled as a **symmetric and bounded** random variable \( \tilde{a}_{ij}, j \in J_i \) that takes values in \([a_{ij} - \tilde{a}_{ij}, a_{ij} + \tilde{a}_{ij}]\). Associated with the uncertain data \( \tilde{a}_{ij} \), we define the random variable \( \eta_{ij} = (\tilde{a}_{ij} - a_{ij})/\tilde{a}_{ij} \), which obeys an **unknown but symmetric** distribution, and takes values in \([-1,1]\).
Robust Optimisation

- There are three well known formulations:

**Soyster**

\[
\sum_{j} a_{ij} x_j + \sum_{j \in J_i} \tilde{a}_{ij} y_j \leq b_i \quad \forall i
\]

\[-y_j \leq x_j \leq y_j \quad \forall j
\]

\[y \geq 0\]

**Ben-Tal and Nemirovski**

\[
\sum_{j} a_{ij} x_j + \sum_{j \in J_i} \tilde{a}_{ij} y_{ij} + \Omega_i \sqrt{\sum_{j \in J_i} \tilde{a}_{ij}^2 z_{ij}^2} \leq b_i \quad \forall i
\]

\[-y_{ij} \leq x_j - z_{ij} \leq y_{ij} \quad \forall i, j \in J_i
\]

\[y \geq 0\]
Robust Optimisation

\[ \sum_{j} a_{ij}x_j + z_i \Gamma_i + \sum_{j \in J_i} p_{ij} \leq b_i \quad \forall \ i \]

\[ z_i + p_{ij} \geq \breve{a}_{ij}y_j \quad \forall \ i, j \in J_i \]

\[ -y_j \leq x_j \leq y_j \quad \forall \ j \]

\[ l_j \leq x_j \leq u_j \]

\[ p_{ij} \geq 0 \quad \forall \ i, j \in J_i \]

\[ y_j \geq 0 \quad \forall \ j, z_i \geq 0 \quad \forall \ i \]

where \( \Gamma_i \in [0, |J_i|] \)
Robust Optimisation

- A reformulation without special constructs of the problem above implies the creation of:
  - Artificial variables \((y_j, z_{ij}, p_{ij}, \ldots)\)
  - Added constraints and bounds
  - The formulation of SOC constraints (for Ben-Tal and Nemirovski) must be recognizable by solvers
Robust Optimisation

- **SAMPL**: same as deterministic model but

```plaintext
param eprice{t in TIME, a in ASSETS};
param amplitude{t in TIME, a in ASSETS};
param Rob{t in TIME};

random param randomPrice{t in TIME, a in ASSETS}
  dist symmetric(eprice[t,a] - amplitude[t,a],
                  eprice[t,a] + amplitude[t,a]);

subject to
  cashbalance{t in TIME} suffix robustness Rob[t]:
    (1-Tcost)*(sum{a in ASSETS}randomprice[t,a]*sell[t,a]) +
    income[t] -
    (1+Tcost)*(sum{a in ASSETS}randomprice[t,a]*buy[t,a]) >=
    liab[t];

option RobustForm BenTal_Nemirovski;
```
Robust extensions

- Keyword *symmetric* to declare the distribution of the random parameter
- Option *RobustForm* to choose the desired robust form, between
  - *Soyster*
  - *BenTal_Nemirovski*
  - *Bertsimas_Sim*
- Suffix *robustness* to specify $\Omega_i$ or $\Gamma_i$ for each
Robust Optimisation

- SOC constraints are not easily recognised by a modelling system/solver

- i.e. Ben-Tal and Nemirovski formulation is:

\[ \sum_{j} a_{ij} x_j + \sum_{j \in J_i} \tilde{a}_{ij} y_{ij} + \Omega_i \sqrt{\sum_{j \in J_i} \tilde{a}_{ij}^2 z_{ij}^2} \leq b_i \]

whilst CPLEX recognizes SOC constraints as:

\[ \sum_{j} (a_{ij} x_j)^2 \leq (d u)^2 \]

where \( u \) is a variable and \( d \) is a constant
Robust Optimisation

• The constraint above is therefore converted by the system in a format acceptable by CPLEX as:

\[ \sum_{j} (a_{ij}x_j)^2 \leq \left( \frac{u_i}{\Omega_i} \right)^2 \]
Solution methods

• Scenario based problems are characterized by finite discrete distributions
• Deterministic equivalent is a large-scale LP with a specific structure
• Solution methods:
  – LP methods directly applied to deterministic equivalent: Simplex, IPM
  – Decomposition methods: Benders’, regularized, level decomposition, integer L-shaped
• Information about the model type and its structure is retained during modelling -> automatic selection of solution method
FortSP architecture

FortSP executable

FortSP library

API

plugin interface

AMPL Solvers

nl interface

CPLEX plugin
CPLEX library

GUROBI plugin
GUROBI library

FORTMP plugin
FORTMP library

Motivation
Architecture
SP
CCP
ICCP
Robust
Optimisation
Solver
Conclusions
Solution methods

- Two stage SP
- Multi stage SP
- Chance Constrained
- Integrated CCPs
- ROBUST Soyster’s
- ROBUST Bertsimas
- ROBUST Ben-Tal
- Benders
- Level
- Nested Benders
- ICCP cutting plane
- Deterministic Equivalent
- LP
- QP
- MIP
- SOCP
Solution methods

**Linear Programming**

\[
\min c^T x \\
\text{subject to } Ax \leq b \\
x \geq 0, x \in \mathbb{R}^n
\]

**Quadratic Programming**

\[
\min \frac{1}{2} x^T Q x + c^T x \\
\text{subject to } Ax \leq b \\
x \geq 0 \\
x \in \mathbb{R}^n
\]

**Mixed Integer Programming**

\[
\min c_1^T x_1 + c_2^T x_2 \\
\text{subject to } A_1 x_1 + A_2 x_2 \leq b \\
x \geq 0 \\
x_1 \in \mathbb{R}^{n_1}, x_2 \in \mathbb{Z}^{n_2}
\]

**Second Order Cone Programming**

\[
\min c^T x \\
\text{subject to } Ax \leq b \\
\|D_i x + e_i\| \leq f_i x + g_i \\
x \geq 0, x \in \mathbb{R}^n
\]
Conclusions

- The advantages of this AML tools approach are:
  - Simplification of the syntax to express these models (both SP and RO)
  - Easy to investigate and introduce a comprehensive modelling framework
  - Conveys the structure of the problem to the solver (permits the exploitation of appropriate solution methods)
References

