# Scenario generation

## Property matching with distribution functions

Michal Kaut[1] Arnt-Gunnar Lium[2]

[1]Molde University College, Molde, Norway
michal.kaut@himolde.no

[2]SINTEF Technology and Society, Trondheim, Norway
Arnt-Gunnar.Lium@sintef.no

Stochastic Programming School 2007
Bergamo, April 10–20, 2007

# Outline


Introduction
- Introduction to Scenario Generation
- Property-matching Methods

Property Matching using CDFs
- The core of the new method
- Improvements and extensions
  - Improving the convergence
  - Discrete distributions
- Other topics
  - Generating a starting point
  - Extensibility

Test case - stochastic network design
- The problem
- Numerical results

# Outline


## Introduction

Introduction to Scenario Generation

Property-matching Methods

## Property Matching using CDFs

The core of the new method

Improvements and extensions

Improving the convergence

Discrete distributions

Other topics

Generating a starting point

Extensibility

## Test case - stochastic network design

The problem

Numerical results

# Introduction to Scenario Generation

Covered by Ronald Hochreiter

# Outline


- Introduction
  - Introduction to Scenario Generation
  - Property-matching Methods
- Property Matching using CDFs
  - The core of the new method
  - Improvements and extensions
    - Improving the convergence
    - Discrete distributions
  - Other topics
    - Generating a starting point
    - Extensibility
- Test case - stochastic network design
  - The problem
  - Numerical results

# Property-Matching Scenario-Generation Methods

- A class of scenario-generation methods that *construct* a scenario tree to match a given set of *properties*. These can include:
  - Moments of the marginal distributions
  - Correlations of the margins
  - Percentiles of the marginal distributions
  - Extreme values of (some of) the margins
- Typically, the properties do not specify the distributions fully; the rest is left to the method.
  - Different methods produce very different results.
  - The issue is very significant for bigger trees, with many more degrees of freedom.

# Property-Matching Methods – Pros/Cons

## Pros

- Do not have to know/assume a distribution family, only to estimate values of the required properties.
- Can combine historical data with today's predictions.
- The marginal distributions can have very different shapes, so the vector does not follow any standard distribution.

## Cons

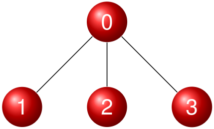
- No convergence to the true distribution.
- If we know the distribution, we can not utilize this information, i.e. we throw it away.
- Can be hard to find which properties to use.

# Example 1 – from Høyland and Wallace (2001)

- An optimization problem with values of the random variables and scenario probabilities as variables.
- The measured properties are expressed as function of these variables.
- The objective is to minimize a distance (usually $L_2$) of these properties from their target values.
- Leads to highly non-linear, non-convex problems.

- Works well for small trees, otherwise very slow.
- The optimization is often underspecified & no control what the solver does about the extra degrees of freedom.

# Example 1 – from Høyland and Wallace (2001) cont.




$\forall i : (x_i, y_i);\ p_i$ .

2 variables $x$, $y$ + node probabilities $p$
Specifications:

- $\mathbb{E}[x], \mathbb{E}[y];\ \mathbb{E}\left[x^2\right], \mathbb{E}\left[y^2\right];\ \mathrm{Cov}(x, y)$
- Possibly other functions of $x$, $y$, $p$.

$$\begin{aligned}\min_{x,y,p}\ & \Big(\sum_i p_i x_i - \mathbb{E}[x]\Big)^2 + \Big(\sum_i p_i y_i - \mathbb{E}[y]\Big)^2 \\ & + \Big(\sum_i p_i x_i^2 - \mathbb{E}\left[x^2\right]\Big)^2 + \Big(\sum_i p_i y_i^2 - \mathbb{E}\left[y^2\right]\Big)^2 \\ & + \Big(\sum_i p_i (x_i - \mathbb{E}[x])(y_i - \mathbb{E}[y]) - \mathrm{Cov}(x, y)\Big)^2\end{aligned}$$

$$\text{s.t.: } \sum_i p_i = 1 \quad \text{and} \quad p_i \geq 0,\ i = 1, \ldots, 3\,.$$

# Example 2 – from Høyland, Kaut, Wallace (2003)

- Developed as a fast approximation to the previous method, in the case of four marginal moments + correlations.
- Build around two transformations:
  1. Correcting correlations
     - Multiply the random vector by a Cholesky component
     - Changes also the marginal distributions (except normal)
  2. Correcting the marginal distributions
     - A cubic transformation of the margins, one margin at a time
     - Changes the correlation matrix
- The two transformations are repeated alternately.
- Starting point can be, for ex., a correlated normal vector.

- Works well for large trees (creates smooth distributions).
- Needs pre-specified probabilities (typically equiprobable).

# Example 2 – from Høyland, Kaut, Wallace (2003) cont.

## Correction of the correlations

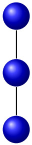
- The target correlation matrix is $R_* = L_* L_*^T$.
- The correlation matrix at step $k$ is $R_k = L_k L_k^T$.
- Then $Y = L_* L_k^{-1} X$ has a correlation matrix equal to $R_*$.

## The cubic transformation

- For each margin $i$: $Y_i = a + bX_i + cX_i^2 + dX_i^3$
- To find the coefficients $a, b, c, d$, we have to:
  - express the moments of $Y_i$ as a function of $a, b, c, d$ and the moments of $X$;
  - find the values of $a, b, c, d$ that minimize the $L_2$ distance of the moments from their target values.
- This is a non-linear, non-convex optimization problem fortunately with only four variables.

# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



$t-3$

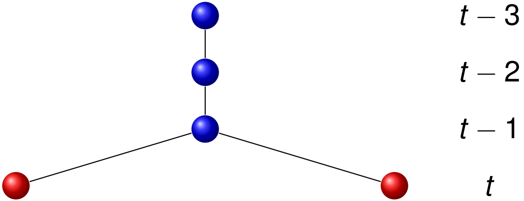
$t-2$

$t-1$

$t$

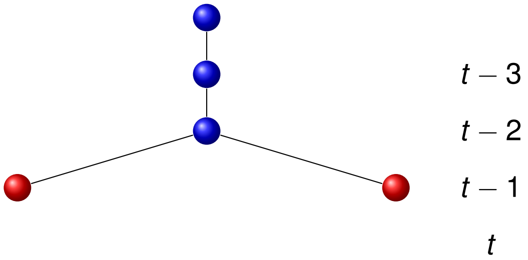
# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



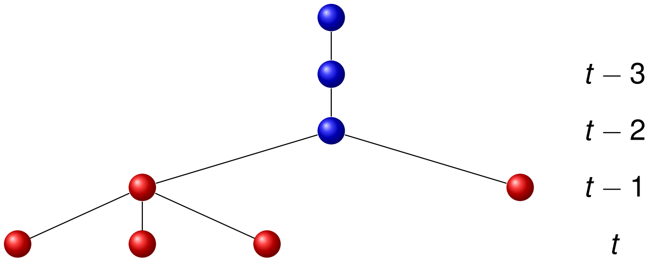
# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



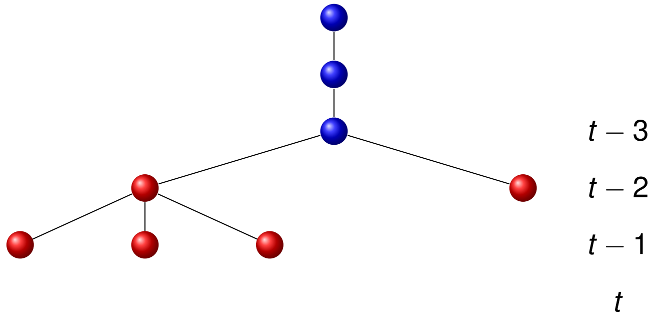
# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



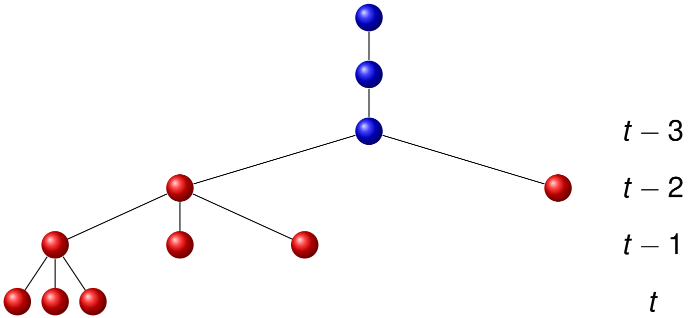
# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



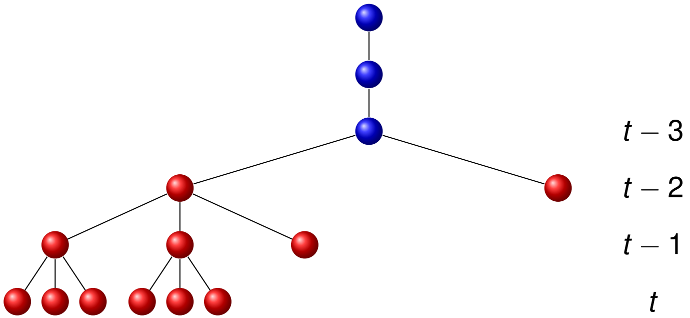
# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



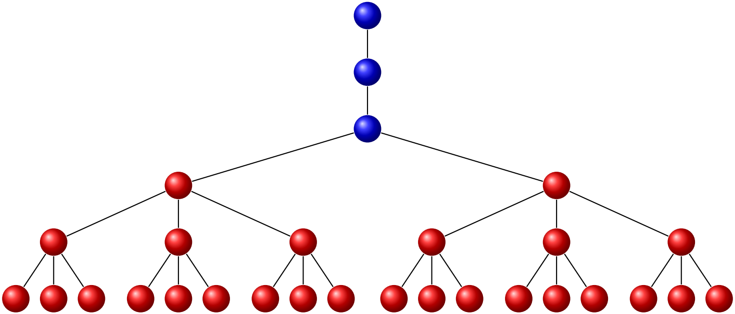
# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



# Example 2 – Handling multiple periods

- The method creates only one-period trees
- For multi-period trees, create the subtrees one by one
- If needed, update the conditional distributions:



# Outline

Introduction
- Introduction to Scenario Generation
- Property-matching Methods

Property Matching using CDFs
- The core of the new method
- Improvements and extensions
  - Improving the convergence
  - Discrete distributions
- Other topics
  - Generating a starting point
  - Extensibility

Test case - stochastic network design
- The problem
- Numerical results

# Property-Matching With Distribution Functions

- What if we *know* the marginal distributions, more precisely their cumulative distribution functions (CDFs)?
  - Using only moments would mean wasting a lot of information.
- Can we use CDFs directly inside an algorithm similar to the transformation method from Høyland et al. (2003)?
- Yes, if we combine them with the correction of correlations from above.
- So, instead of correcting the margins' moments with the cubic transformation, we want to correct the CDFs. For this we need:
  - Transformation to change distribution of one margin to make it "closer" to the distribution given by a CDF.
  - A measure of the distance of the current sample from the target distribution.

# Property-Matching With Distribution Functions

- What if we *know* the marginal distributions, more precisely their cumulative distribution functions (CDFs)?
  - Using only moments would mean wasting a lot of information.
- Can we use CDFs directly inside an algorithm similar to the transformation method from Høyland et al. (2003)?
- Yes, if we combine them with the correction of correlations from above.
- So, instead of correcting the margins' moments with the cubic transformation, we want to correct the CDFs. For this we need:
  - Transformation to change distribution of one margin to make it "closer" to the distribution given by a CDF.
  - A measure of the distance of the current sample from the target distribution.

# CDF-correcting transformation

1. Spread the margin $\boldsymbol{x} = (x_1, \dots, x_S)$ evenly on $(0, 1)$:

$$u_s = F^e_{\boldsymbol{x}}(x_s) := \frac{2\,\mathrm{rank}(x_s, \boldsymbol{x}) - 1}{2S}, \quad s = 1 \dots S\,, \tag{1}$$

2. Transform them to the target distribution:

$$\boldsymbol{x} = F^{-1}(\boldsymbol{u})\,. \tag{2}$$

Hence, the whole transformation can be written as

$$\boldsymbol{x} \leftarrow F^{-1}\left(F^e_{\boldsymbol{x}}(\boldsymbol{x})\right)\,, \tag{3}$$

or, in terms of ordered values $x_{(s)}$

$$x_{(s)} \leftarrow F^{-1}\left(\tfrac{2s-1}{2S}\right), \quad s = 1 \dots S\,.$$

# Measuring distance from the true distribution

- Closely related to the margin-correcting procedure:
- Compute the actual CDF $F(x_s)$.
- Define the error as its mean-square difference from the "discretized CDF" $F^e_{\boldsymbol{x}}(x_s)$:

$$\text{error of margin } \boldsymbol{x} = \sqrt{\frac{1}{S}\sum_{s=1}^{S}\left(F(x_s) - F^e_{\boldsymbol{x}}(x_s)\right)^2}\,.$$

- The error is zero after the correction, because

$$F(x_s) = F^e_{\boldsymbol{x}}(x_s) = u_s \text{ for all } s\,.$$

# What about Kolmogorov distance?

Why not use the standard Kolmogorov distance,

$$\sup_x |F(x) - F^e_{\boldsymbol{x}}(x)|$$

$$= \max_{1 \le s \le S} \max \left\{ \frac{\mathrm{rank}(x_s, \boldsymbol{x})}{S} - F(x_s), F(x_s) - \frac{\mathrm{rank}(x_s, \boldsymbol{x}) - 1}{S} \right\}.$$

- We use mean-square error for moments and wanted something similar/comparable for CDFs.
- For continuous distributions, the Kolmogorov distance will never be zero.
  - We want that for compatibility with moments.

# Connection to Kolmogorov distance

## Proposition

(i) *Transformation* (3) *minimizes the Kolmogorov distance from the true CDF, within a class of discrete distributions with support of cardinality* $S$*.*

(ii) *Kolmogorov distance of vector* $\boldsymbol{x}$ *from the true CDF after transformation* (3) *is equal to* $\frac{1}{2S}$*.*

## Proof.

(ii) Since $\boldsymbol{x} = F^{-1}(\boldsymbol{u})$, we get $F(x_s) = u_s = \frac{2\,\mathrm{rank}(x_s,\boldsymbol{x})-1}{2S}$. The result then follows from definition of Kolmogorov dist.

(i) The distance is a maximum of $2S$ non-negative values that sum up to one, so it can not be smaller than $\frac{1}{2S}$.

# Outline

Introduction
- Introduction to Scenario Generation
- Property-matching Methods

Property Matching using CDFs
- The core of the new method
- Improvements and extensions
  - Improving the convergence
  - Discrete distributions
- Other topics
  - Generating a starting point
  - Extensibility

Test case - stochastic network design
- The problem
- Numerical results

# Improving the convergence

- After the transformation, all margins have a fixed discrete distribution, with values $F^{-1}\left(\frac{2s-1}{2S}\right),\ s = 1 \ldots S$.
  - Values are fixed, only the order is random.
- Correlations can be improved only by pairing up these fixed values.
  - Too little freedom, i.e. a too constraint problem
  - Most likely impossible to get an exact match
  - *Negative impact on convergence*

Solution: replace $\boldsymbol{u} = F^e_{\boldsymbol{x}}(\boldsymbol{x})$ by

$$\boldsymbol{u} = \alpha F^e_{\boldsymbol{x}}(\boldsymbol{x}) + (1-\alpha) F(\boldsymbol{x})\,, \tag{1'}$$

where $\alpha \in [0,1]$ is a weight increasing during the algorithm.

# Improving the convergence, cont.

- The complete transf. $\boldsymbol{x} \leftarrow F^{-1}\left(F^e_{\boldsymbol{x}}(\boldsymbol{x})\right)$ becomes

$$\boldsymbol{x} \leftarrow F^{-1}\left(\alpha F^e_{\boldsymbol{x}}(\boldsymbol{x}) + (1-\alpha)F(\boldsymbol{x})\right), \tag{3'}$$

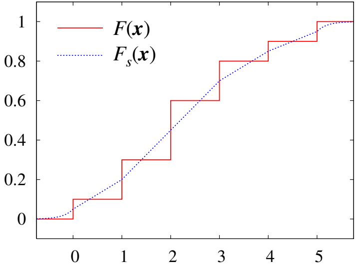
- Equals to:
  - identity for $\alpha = 0$
  - the original transf. for $\alpha = 1$.
- By increasing $\alpha$ slowly from zero to one, the correlations have a better chance to 'settle down'.
- If we can not get exact correlations with $\alpha = 1$:
  - Stop the algorithm with some $\alpha < 1$.
  - Trade-off between margins and correlations
  - Note that $\alpha < 1$ does *not* necessarily mean wrong marginal distributions, in the sense that the standard tests would not reject the null hypothesis.

# Discrete distributions

- The most important differences are:
  - CDFs are piece-wise constant.
  - Typically have many scenarios sharing the same value.
- Possibility of big changes in values while correcting the margins
- Can have a negative impact on the overall convergence.
- Margin-correcting transformations may cease to be rank-preserving.
  - Can be avoided by a slight change in the rank(., $\boldsymbol{x}$) function.

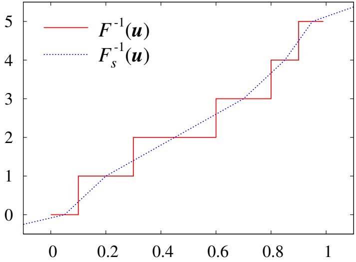
# Improving the convergence for discrete distrib.

- Use a smoothed CDF, denoted by $F_s$.
  - In out case, a linear interpolation of $F$, connecting the average of the left and right limits at each value of the discrete distribution.
  - Below the minimum and above the maximum it converges to 0 and 1, respectively, so $F_s(x_s)$ is always inside $(0, 1)$.



# Improving the convergence for discrete distrib. cont.

- Unfortunately, we can not use directly the inverse of $F_s$:
  - Defined only on interval $(0, 1)$.
  - Values outside $(0, 1)$ are possible at the early stages of the algorithm.
  - Hence, we extend $F_s^{-1}$ to the whole $\mathcal{R}$.
  - Will be denoted $F_s^{-1}$ anyway.



# Improving the convergence for discrete distrib. cont.

- The smoothed versions of the distribution functions are then used similarly to the 'empirical CDF':
  - $F(\boldsymbol{x})$ is replaced by $\beta F(\boldsymbol{x}) + (1-\beta) F_s(\boldsymbol{x})$
  - $\beta$ increases from zero to one.
- Combined with $\boldsymbol{u} = \alpha F^e_{\boldsymbol{x}}(\boldsymbol{x}) + (1-\alpha) F(\boldsymbol{x})$, we get

$$\boldsymbol{u} = \alpha F^e_{\boldsymbol{x}}(\boldsymbol{x}) + (1-\alpha)\big(\beta F(\boldsymbol{x}) + (1-\beta) F_s(\boldsymbol{x})\big)\,. \qquad (1")$$

  - The correction of correlations may cause some $x_s$ to be outside the support of its distribution.
  - In such a case, $F_s(x_s)$ will be outside $[0,1]$ and the same may be true for $u_s$, depending on the values of $\alpha$ and $\beta$.
  - This is why we have to extend the definition of $F_s^{-1}$ onto $\mathcal{R}$.
- The same for the inverse CDF, so $\boldsymbol{x} = F^{-1}(\boldsymbol{u})$ becomes:

$$\boldsymbol{x} = \beta F^{-1}(\boldsymbol{u}) + (1-\beta) F_s^{-1}(\boldsymbol{u})\,. \qquad (2")$$

# Outline

Introduction
- Introduction to Scenario Generation
- Property-matching Methods

Property Matching using CDFs
- The core of the new method
- Improvements and extensions
  - Improving the convergence
  - Discrete distributions
- Other topics
  - Generating a starting point
  - Extensibility

Test case - stochastic network design
- The problem
- Numerical results

# Starting point

- Sampling from data (if available)
  - simple to implement
  - guaranteed correct distribution (not only correlations)
  - The alg. can be seen as a *post-process* for sampling.
- Generation, starting with indep. $U(0,1)$ sample.
  - Start by correcting the margins.
    1. First spread the samples evenly on $(0,1)$.
    2. Then transform the margins to their target distributions.
    3. The result is correct margins that are approximately independent.
  - Start by correcting correlations, via normal distribution.
    1. Transform to standard normal.
    2. Set the correlations $\rightarrow$ margins remain normal.
    3. Spread the margins evenly in terms of percentiles.
    4. Correct the marginal distributions.
    5. The result is correct distributions with approximate correlations.

# Extensibility

- The alg. is suited for extensions by adding new type of corrections, both for the *marginal distributions* and for the *multivariate structure*.
- Hence, if we have one or more margins with distributions requiring a special treatment, they can be handled by adding a new type of correction.
- Ex.: If we start with data, we can use the empirical CDF from the data—possibly interpolated—as the CDFs in the algorithm.
- Ex.: if the only information we have about the marginal distribution is a set of percentile values, we can stretch the margins to match a given set of percentiles—see Okunev and White (2006).

# Outline

Introduction
- Introduction to Scenario Generation
- Property-matching Methods

Property Matching using CDFs
- The core of the new method
- Improvements and extensions
  - Improving the convergence
  - Discrete distributions
- Other topics
  - Generating a starting point
  - Extensibility

Test case - stochastic network design
- The problem
- Numerical results

# The problem

- Test case from stochastic network design for less-than-truckload trucking carrier.
  - two-stage stochastic integer program
  - the first stage corresponds to routing of the trucks
  - the second stage represents the flow of freight, satisfying stochastic demands
  - flow of freight is treated as continuous variables
    $\rightarrow$ integrality only in the first stage.
  - Costs are associated only with the first stage (depend only on the number of trucks and the distances)
  - $\rightarrow$ Objective function is fully determined by the first-stage solution (truck routing).
  - $\rightarrow$ Only the first-stage solutions are considered in the tests.
- More info in Lium and Kaut (2006).

# Details of the model

- Hard constraints in the second stage
  - Constructing routes so that the demands can be satisfied in all scenarios.
  - This increases the instability of the scenario-based solutions.
  - $\longrightarrow$ It magnifies the differences between the different scenario-generation techniques.
  - Note: Generally a bad idea, but useful here.
  - Note: Solutions obtained using one scenario tree will most likely be infeasible in a model using a different set of scenarios.

# Setup of the test

Testing using methodology from Kaut and Wallace (2007):

- In-sample stability
  - Comparing the *reported* performance of solutions.
- Out-of-sample stability
  - Ideally, we would want to the *true* performances of solutions.
  - We use a tree with 1000 scenarios as our representation of reality—the truth tree
  - In out model, the objective value is given by the first stage, so it is *known*—but the scenario solutions are most likely to be infeasible on the truth tree.
  - $\rightarrow$ compare the solutions by the amount of infeasibility.

## Setup of the test cont.

- 12 commodities, i.e. 12 random variables.
- the *truth tree* consists of 1000 scenarios for 12 variables, sampled independently from triangular distributions.
- We compare performance of the following scenario-generation methods:
  - Standard sampling from the truth tree.
  - Moment matching, with moments and correlations estimated from the truth tree.
  - The new method, using triangular distributions with parameters estimated from the truth tree; Started by $U(0, 1)$ random numbers.
  - The new method, using interpolated empirical distribution functions from the truth tree; Started by sampling from the truth tree.

# Setup of the test cont.

- For each method, we do
  - Scenario trees from 13 to 53 scenarios, with step of 8.
  - 20 different problems for each combination.
- $\rightarrow$ 480 problems in total.
- Using AMPL and CPLEX 9 on a 3 GHz PC with 1 GB RAM
- Solution times from a couple of minutes to a couple of days(!)

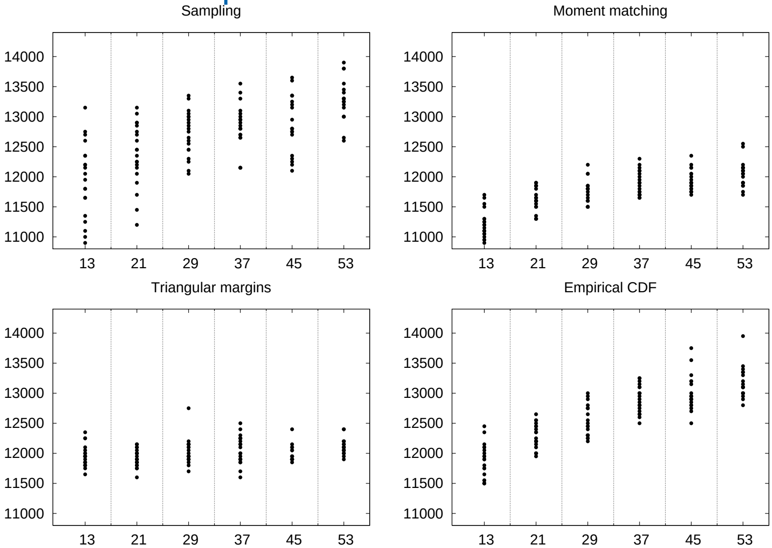
# Outline

Introduction
Introduction to Scenario Generation
Property-matching Methods

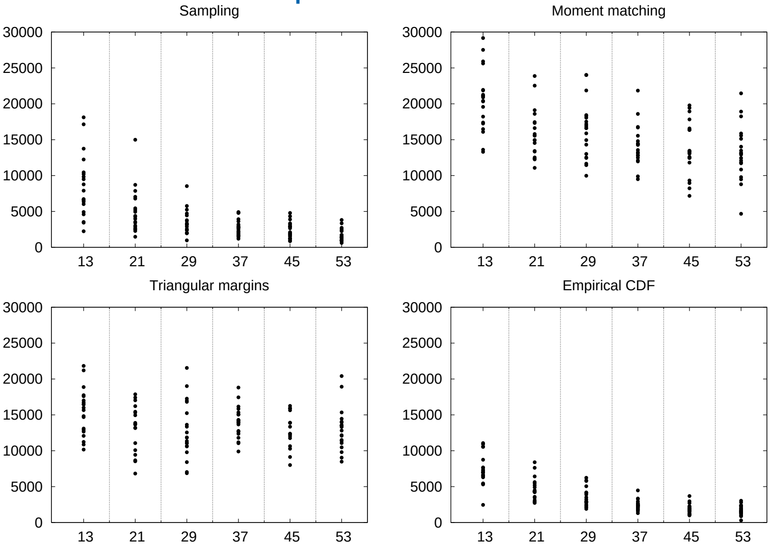
Property Matching using CDFs
The core of the new method
Improvements and extensions
Improving the convergence
Discrete distributions
Other topics
Generating a starting point
Extensibility

Test case - stochastic network design
The problem
Numerical results

# Results – In-sample tests



# Results – Out-of-sample tests



# Test – Conclusions

- The new method using empirical CDF from the truth tree and started by sampling is clearly the best of the four methods
  - the best out-of-sample stability
  - more in-sample stability than sampling
- The test shows the importance of using all the information available
  - The two methods that approximate CDFs have significantly worse out-of-sample performance.
  - They both find “cheaper” solutions that turn out to be worse in the “real world”.

# Outline


## Introduction

Introduction to Scenario Generation

Property-matching Methods

## Property Matching using CDFs

The core of the new method

Improvements and extensions

Improving the convergence

Discrete distributions

Other topics

Generating a starting point

Extensibility

## Test case - stochastic network design

The problem

Numerical results

## Summary

# Summary

- A new method for generating scenarios with specified correlations and marginal distributions given by CDFs.
- Improvement over matching only the moments of the distributions, in the terms of:
  - approximating the distribution
  - obtaining better solutions of stochastic programs
- Improvement illustrated on a case from service network design
  - The new method proved to be better than both sampling and the moment-matching approach.
- Open questions:
  - Improve the control of co-distribution (multi-variate structure of the distribution) – at the moment still only correlations.

# For Further Reading I


- K. Høyland and S. W. Wallace.
  Generating scenario trees for multistage decision problems.
  *Management Science*, 47(2):295–307, 2001.

- Kjetil Høyland, Michal Kaut, and Stein W. Wallace.
  A heuristic for moment-matching scenario generation.
  *Comput. Optim. Appl.*, 24(2-3):169–185, 2003.

- Michal Kaut and Stein W. Wallace.
  Evaluation of scenario-generation methods for stochastic programming.
  *Pacific Journal of Optimization*, to appear, 2007.

- Arnt-Gunnar Lium and Michal Kaut.
  Scenario generation for obtaining sound solutions.
  In *Stochastic Service Network Design*, chapter 4. Molde University College, 2006.
  PhD thesis by A.-G. Lium.

# For Further Reading II


W.K. Mak, D.P. Morton, and R.K. Wood.
Monte carlo bounding techniques for determining solution quality in stochastic programs.
*Oper. Res. Lett.*, 24:47–56, 1999.

John Okunev and Derek R. White.
Moment matching for the masses.
Available from `http://www.ssrn.com/`, 2006.

# The End