

# Optimisation under Uncertainty

## Session 2/4

I Workshop de Otimização sob Incerteza - UFSCar

Fabricio Oliveira

Systems Analysis Laboratory  
Department of Mathematics and Systems Analysis

Aalto University  
School of Science

August 17, 2023

# Outline of this lecture

Introduction

Scenario trees

Generating scenario trees

Scenario (tree) generation methods

Sample Average Approximation (SAA)

# Outline of this lecture

Introduction

Scenario trees

Generating scenario trees

Scenario (tree) generation methods

Sample Average Approximation (SAA)

# Stochastic programming models

Mathematical programming models in which some of the parameters are assumed to be **random variables**.

It comprises the following parts:

1. A mathematical programming model
2. **Deterministic** parameter values
3. Description of the **stochasticity**, e.g.,
  - a known probability distribution;
  - historical data;
  - distribution properties (average, standard deviation, i.e., moments)

# Stochastic programming models

Mathematical programming models in which some of the parameters are assumed to be **random variables**.

It comprises the following parts:

1. A mathematical programming model
2. **Deterministic** parameter values
3. Description of the **stochasticity**, e.g.,
  - a known probability distribution;
  - historical data;
  - distribution properties (average, standard deviation, i.e., moments)

The most widespread use of stochastic programs relies on **scenarios**:

- ▶ Lead to **tractable** deterministic equivalents;
- ▶ Are **approximations** of the original stochastic process

# Stochastic programming models

A scenario tree  $\xi$  comprises **sequentially observed realisations** of  $\xi^t$ , for  $t = 1, \dots, H$ :

- ▶  $\xi = (\xi^t)_{t \in [H]}$ , where  $(\cdot)$  denotes a **sequence** and  $\xi^t \in \Xi_t$ ;
- ▶ a **scenario** is denoted  $\xi_s = (\xi_s^t)_{t \in [H]}$  forming a “path” through  $\xi$ ;
- ▶ Thus,  $\xi = \{\xi_s\}_{s \in [S]}$ , where  $S$  is the number of scenarios.

# Stochastic programming models

A scenario tree  $\xi$  comprises **sequentially observed realisations** of  $\xi^t$ , for  $t = 1, \dots, H$ :

- ▶  $\xi = (\xi^t)_{t \in [H]}$ , where  $(\cdot)$  denotes a **sequence** and  $\xi^t \in \Xi_t$ ;
- ▶ a **scenario** is denoted  $\xi_s = (\xi_s^t)_{t \in [H]}$  forming a “path” through  $\xi$ ;
- ▶ Thus,  $\xi = \{\xi_s\}_{s \in [S]}$ , where  $S$  is the number of scenarios.

**Example:**

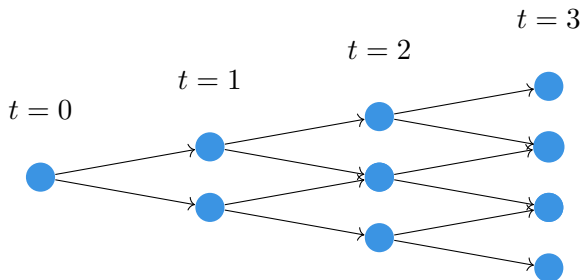


Figure: A 4-stage (**lattice**) scenario tree with 2 scenarios per stage.  $\xi = (\xi^1, \xi^2, \xi^3)$ ;

# Outline of this lecture

Introduction

Scenario trees

Generating scenario trees

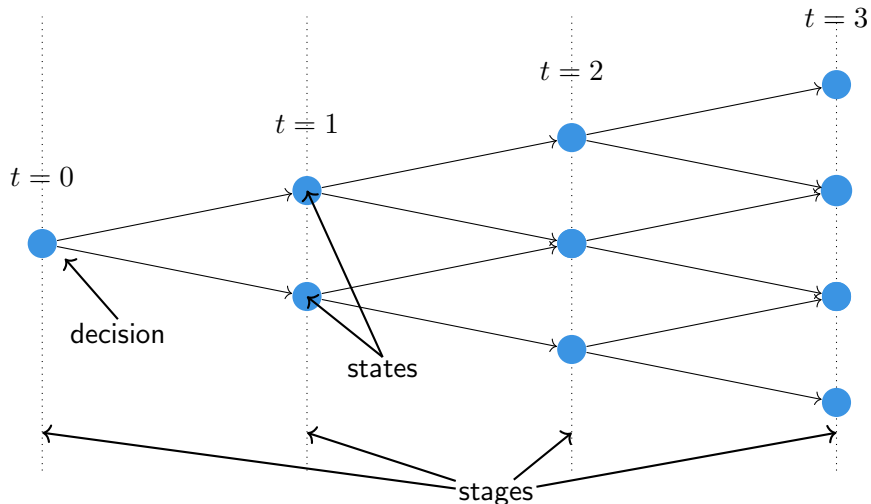
Scenario (tree) generation methods

Sample Average Approximation (SAA)

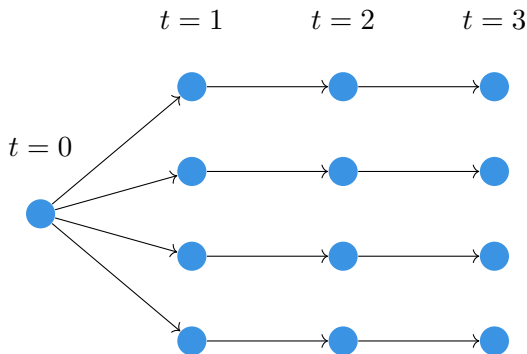


# Taxonomy of scenario trees

## Terminology



# Taxonomy of scenario trees



**Branching** indicates a decision upon arrival of **new information**

- ▶ No branching, no additional information;
- ▶ **Fan trees** represent **multi-period 2-stage problems**.

# Outline of this lecture

Introduction

Scenario trees

Generating scenario trees

Scenario (tree) generation methods

Sample Average Approximation (SAA)

## Trade-off approximation quality vs. tractability

Two parameters govern the geometry of a scenario tree:

- ▶ **Depth:** number of stages  $H$
- ▶ **Breadth (or width):** number of realisations per stage  $|\xi^t|$

## Trade-off approximation quality vs. tractability

Two parameters govern the geometry of a scenario tree:

- ▶ **Depth:** number of stages  $H$
- ▶ **Breadth (or width):** number of realisations per stage  $|\xi^t|$

The **total of scenarios** is  $O(N^H)$  (assuming  $|\xi_t| = N$  for  $t \in [H]$ )

- ▶ Larger  $H$  convey more **adaptability** to revealed information;
- ▶ Larger  $|S|$  convey a more **precise** description of the uncertainty;
- ▶ **Computational tractability** issues pressure them to be as small as possible.

Most scenario generation methods seek to find trees with **minimal**  $|\xi|$  such that **representation quality** requirements are observed.

# Data source

Typical **sources** for scenarios include:

1. **Historical data:** past observations as possible future observations;
2. **Simulation models:** Monte Carlo, systems dynamics, agent-based and discrete event simulation;
3. **Expert elicitation:** typically a small number of scenarios with no possible back-testing

# Data source

Typical **sources** for scenarios include:

1. **Historical data:** past observations as possible future observations;
2. **Simulation models:** Monte Carlo, systems dynamics, agent-based and discrete event simulation;
3. **Expert elicitation:** typically a small number of scenarios with no possible back-testing

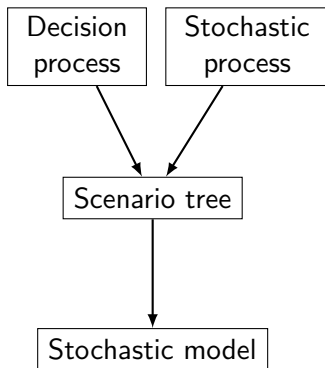
Often, a **combination** of the above is used:

1. Start from the **data**;
2. Define and fit a **parametric model**;
3. Generate **observations** from the model.

# Scenario generation and modelling

Scenario generation must be part of the **modelling process**

- ▶ Problem dependent;
- ▶ The method for generating scenarios is a **modelling decision**;
- ▶ Often overlooked in applications;
- ▶ Quality of scenarios **majorly** influences quality of solution (“garbage in = garbage out”)





# Quality measures for scenario trees

Apart from **epistemic error** questions, two measures must be considered when generating scenario trees:

## 1. Error

- Error introduced for using an **approximation** of the real stochastic process;
- Unlikely to be measurable, but possible to be approximated.

## 2. Stability

- Scenario-trees approximating the same stochastic process should yield the same solution;
- Likewise, objective function values should be stable.

Let  $\xi$  be a scenario tree representing the original stochastic process  $\eta$ , and  $\mathcal{F}(x, \xi) = \mathbb{E}_{\xi} [F(x, \xi)]$ . We are interested in understanding how well

$$\min_{x \in X} \mathcal{F}(x, \xi) \text{ approximates } \min_{x \in X} \mathcal{F}(x, \eta)$$

## Quality measures for scenario trees

Let  $\xi_k$  for  $k = 1, \dots, n$  be a collection of alternative scenario trees generated to represent  $\eta$ . We have that

$$x_k^* = \arg \min_{x \in X} \mathcal{F}(x, \xi_k).$$

The **approximation error** [Pflug, 2001] is defined as

$$\begin{aligned} e(\eta, \xi_k) &= \mathcal{F}(\arg \min_{x \in X} \mathcal{F}(x, \xi_k), \eta) - \mathcal{F} \arg \min_{x \in X} \mathcal{F}(x, \eta), \eta) \\ &= \mathcal{F}(x_k^*, \eta) - \min_{x \in X} \mathcal{F}(x, \eta). \end{aligned}$$

## Quality measures for scenario trees

Let  $\xi_k$  for  $k = 1, \dots, n$  be a collection of alternative scenario trees generated to represent  $\eta$ . We have that

$$x_k^* = \arg \min_{x \in X} \mathcal{F}(x, \xi_k).$$

The **approximation error** [Pflug, 2001] is defined as

$$\begin{aligned} e(\eta, \xi_k) &= \mathcal{F}(\arg \min_{x \in X} \mathcal{F}(x, \xi_k), \eta) - \mathcal{F}(\arg \min_{x \in X} \mathcal{F}(x, \eta), \eta) \\ &= \mathcal{F}(x_k^*, \eta) - \min_{x \in X} \mathcal{F}(x, \eta). \end{aligned}$$

- ▶ Calculating  $\mathcal{F}(x_k^*, \eta)$  requires evaluating the “true” objective function;
- ▶ Alternatively, **Monte Carlo simulation** is often employed to approximate  $\mathcal{F}(x_k^*, \eta)$ ;
- ▶ Clearly, there is no way to evaluate  $\min_{x \in X} \mathcal{F}(x, \eta)$ .

# Stability for scenario trees

## Out-of-sample stability

We often assume that we can **approximate**  $\mathcal{F}(x_k^*, \eta)$ . This allows us to

- ▶ compare solution  $x_1^*$  and  $x_2^*$ ;
- ▶ compare **alternative** scenario generation methods;
- ▶ perform **out-of-sample** stability test:
  1. Generate a set of scenario trees  $\{\xi_1, \dots, \xi_n\}$ ;
  2. Obtain solutions  $x_k$ ,  $k = 1, \dots, n$ ;
  3. Test whether  $\mathcal{F}(x_k^*, \eta) \approx \mathcal{F}(x_l^*, \eta)$ , for  $k, l = 1 \dots, n : k \neq l$ .

# Stability for scenario trees

## Out-of-sample stability

We often assume that we can **approximate**  $\mathcal{F}(x_k^*, \eta)$ . This allows us to

- ▶ compare solution  $x_1^*$  and  $x_2^*$ ;
- ▶ compare **alternative** scenario generation methods;
- ▶ perform **out-of-sample** stability test:
  1. Generate a set of scenario trees  $\{\xi_1, \dots, \xi_n\}$ ;
  2. Obtain solutions  $x_k$ ,  $k = 1, \dots, n$ ;
  3. Test whether  $\mathcal{F}(x_k^*, \eta) \approx \mathcal{F}(x_l^*, \eta)$ , for  $k, l = 1 \dots, n : k \neq l$ .

### Remarks:

- ▶  $e(\eta, \xi_k) \approx 0 \Rightarrow e(\eta, \xi_k) \approx e(\eta, \xi_l) \equiv \mathcal{F}(x_k^*, \eta) \approx \mathcal{F}(x_l^*, \eta)$ ;
- ▶ The procedure above can also be used to **assess scenario tree** width (scenarios per stage).

# Stability for scenario trees

## In-sample stability

**In-sample** stability is defined as

$$\mathcal{F}(x_k^*, \xi_k) \approx \mathcal{F}(x_l^*, \xi_l), \text{ for } k, l = 1, \dots, n : k \neq l.$$

# Stability for scenario trees

## In-sample stability

**In-sample** stability is defined as

$$\mathcal{F}(x_k^*, \xi_k) \approx \mathcal{F}(x_l^*, \xi_l), \text{ for } k, l = 1, \dots, n : k \neq l.$$

In some contexts, can also be defined as

$$\|x_k^* - x_l^*\|_p, \text{ for } k, l = 1, \dots, n : k \neq l.$$

where  $\|\cdot\|_p$  is a vector  $p$ -norm.

# Stability for scenario trees

## In-sample stability

**In-sample** stability is defined as

$$\mathcal{F}(x_k^*, \xi_k) \approx \mathcal{F}(x_l^*, \xi_l), \text{ for } k, l = 1, \dots, n : k \neq l.$$

In some contexts, can also be defined as

$$\|x_k^* - x_l^*\|_p, \text{ for } k, l = 1, \dots, n : k \neq l.$$

where  $\|\cdot\|_p$  is a vector  $p$ -norm.

- ▶ No direct connection to **out-of-sample** stability;
- ▶ Useful for assessing the **internal** stability of a random scenario generation method;
- ▶ Translates into **confidence** in the objective function value reported.



# Stability for scenario trees

## Final considerations

Some practical advice:

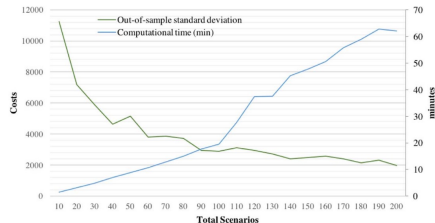
- ▶ No stability implies **dependence** on the scenario tree. To improve stability one can
  1. Consider alternative scenario generation methods
  2. Increase the number of scenarios
- ▶ In case approximating  $\mathcal{F}(x_k^*, \eta)$  is not feasible, cross-testing can be employed. Let

$$\overline{\mathcal{F}} = \{\mathcal{F}(x_k^*, \xi_l)\}_{k,l=1,\dots,n:k \neq l}.$$

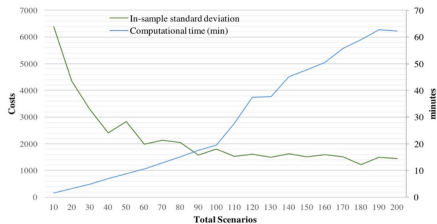
Out-of-sample stability implies that the standard deviation of  $\overline{\mathcal{F}}$  is close to 0.

- ▶ For a rigorous treatment of stability, check [Dupačová, 1990, Schultz, 2000, Heitsch et al., 2006].

# Stability for scenario trees [Dillon et al., 2017]



(a) Out-of-sample stability



(b) In-sample stability

# Outline of this lecture

Introduction

Scenario trees

Generating scenario trees

Scenario (tree) generation methods

Sample Average Approximation (SAA)

# Scenario generation methods

The main types of scenario-generation methods are:

1. **Sampling:** Monte-Carlo sampling, or quasi Monte-Carlo sampling using variance reduction techniques (e.g., Sobol sequences). Combined with Sample Average Approximation (SAA).

# Scenario generation methods

The main types of scenario-generation methods are:

1. **Sampling:** Monte-Carlo sampling, or quasi Monte-Carlo sampling using variance reduction techniques (e.g., Sobol sequences). Combined with Sample Average Approximation (SAA).
2. **Moment matching:** artificially generates a set of scenarios with the same (four plus correlation, usually) moments as the desired distribution;

# Scenario generation methods

The main types of scenario-generation methods are:

1. **Sampling:** Monte-Carlo sampling, or quasi Monte-Carlo sampling using variance reduction techniques (e.g., Sobol sequences). Combined with Sample Average Approximation (SAA).
2. **Moment matching:** artificially generates a set of scenarios with the same (four plus correlation, usually) moments as the desired distribution;
3. **Metric-based:** form smaller scenario sets whilst minimising some probabilistic distance metric. Includes clustering (k-means and related methods) and scenario reduction.

# Scenario generation methods

## Moment matching

Build a scenario tree  $(z_s, p_s)_{s \in \Xi}$  that has statistical moments  $f_m(z, p)$  matching target values  $M_m^{\text{VAL}}$ .

- ▶ Moments extracted from the original distribution, or data;
- ▶ The following problem must be solved ([Høyland and Wallace, 2001]):

$$\begin{aligned} \min_{z, p \geq 0} \quad & \sum_{m \in M} w_m (f_m(z, p) - M_m^{\text{VAL}})^2 \\ \text{s.t.:} \quad & \sum_{j=1}^{|\Xi|} p_j = 1, \end{aligned}$$

where  $w_m$  are weights.

**Remark:** [Høyland et al., 2003] show how the above problem can be heuristically solved.

# Scenario generation methods

## Metric-based methods

Probability-metric based methods use the following result [Pflug, 2001]

$$e(\eta, \xi_k) \leq Kd(\eta, \xi_k)$$

where  $K$  is a (Lipschitz-related) constant and  $d$  is a **Wasserstein distance** between  $\eta$  and  $\xi_k$ . Thus, the focus is on obtaining trees that **minimise**  $d$ .



# Scenario generation methods

## Metric-based methods

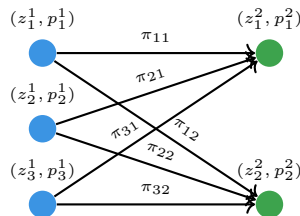
Probability-metric based methods use the following result [Pflug, 2001]

$$e(\eta, \xi_k) \leq Kd(\eta, \xi_k)$$

where  $K$  is a (Lipschitz-related) constant and  $d$  is a **Wasserstein distance** between  $\eta$  and  $\xi_k$ . Thus, the focus is on obtaining trees that **minimise**  $d$ .

Let  $\xi^l = (z^l, p^l) \in \Xi^l$ . The Wasserstein distance  $d(\xi^1, \xi^2)$  is given by:

$$\begin{aligned} \min_{\pi} \quad & \sum_{i \in \xi^1, j \in \xi^2} \|z_i^1 - z_j^2\|^r \pi_{ij} \\ \text{s.t.:} \quad & \sum_{j \in \xi^2} \pi_{ij} = p_i^1, \quad \forall i \in \xi_1 \\ & \sum_{i \in \xi^1} \pi_{ij} = p_j^2, \quad \forall j \in \xi_2. \end{aligned}$$



# Scenario generation methods

## Metric-based methods

### 1. “Clustering-like” methods:

- ▶  $k$ -means, and variants incorporating Wasserstein distance as the metric [Condeixa et al., 2020]
- ▶ Work well in case scenarios are generated from **data** [Kaut, 2021];
- ▶ [Löwhndorf, 2016]: Learning-based algorithms such as competitive learning and Voronoi cell sampling as alternatives to  $k$ -means.

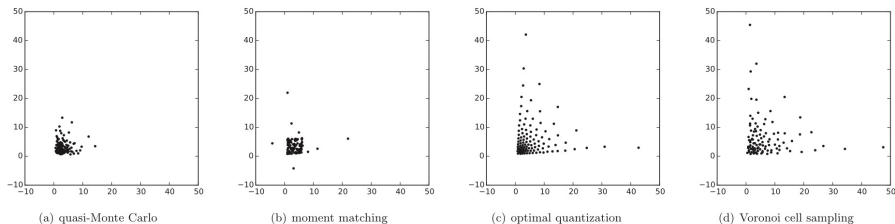


Figure: comparison of scenario generation methods ([Löwhndorf, 2016])

# Scenario generation methods

## Metric-based methods

2. **Scenario reduction methods:** Obtain  $\xi^2$  from  $\xi^1$  where  $|\xi^2| > |\xi^1|$ .
- ▶ Based on the theory of stability of stochastic programs [Römisch, 2003]
    - Changes in the solution can be approximated using a Forter-Mourier-type metric
    - Calculation leads to a Monge-Kantorovich mass transportation problem
  - ▶ “Historical” chronology:
    1. [Dupačová et al., 2003, Heitsch and Römisch, 2003]: first **backward reduction** and **forward selection** methods;
    2. [Heitsch and Römisch, 2007] improved versions of the heuristics;
    3. [Heitsch and Römisch, 2009] The above does not work for **multi-stage** problems. Provides a method that does.

# Scenario generation methods

## Scenario reduction

Types of reduction algorithms. Let  $K$  be a target value for  $|\xi^2|$

- ▶ Backward reduction: repeat until  $|\xi^2| = K$ . Start from  $\xi^1$ 
  1. Find the scenario whose removal causes the **smallest error increase**
  2. Remove the scenario and redistribute its probability
- ▶ Forward selection: repeat until  $|\xi^2| = K$ . Start from  $\xi^2 = \emptyset$ 
  1. Find the scenario whose inclusion causes the **largest error decrease**
  2. Add the scenario and redistribute its probability

# Scenario generation methods

## Scenario reduction

Types of reduction algorithms. Let  $K$  be a target value for  $|\xi^2|$

- ▶ Backward reduction: repeat until  $|\xi^2| = K$ . Start from  $\xi^1$ 
  1. Find the scenario whose removal causes the **smallest error increase**
  2. Remove the scenario and redistribute its probability
- ▶ Forward selection: repeat until  $|\xi^2| = K$ . Start from  $\xi^2 = \emptyset$ 
  1. Find the scenario whose inclusion causes the **largest error decrease**
  2. Add the scenario and redistribute its probability

Some final practical remarks:

- ▶ In [Heitsch and Römisch, 2003], their results indicate:
  - 50% of the scenarios gives 90% relative accuracy
  - 1% of the scenarios gives 50% accuracy
- ▶ **Forward selection** gives better results, but is slow for large  $|\xi^1|$  and  $K$ .
- ▶ **Scenred2** (GAMS) is an available implementation.

# Scenario generation methods

Some of my own experience

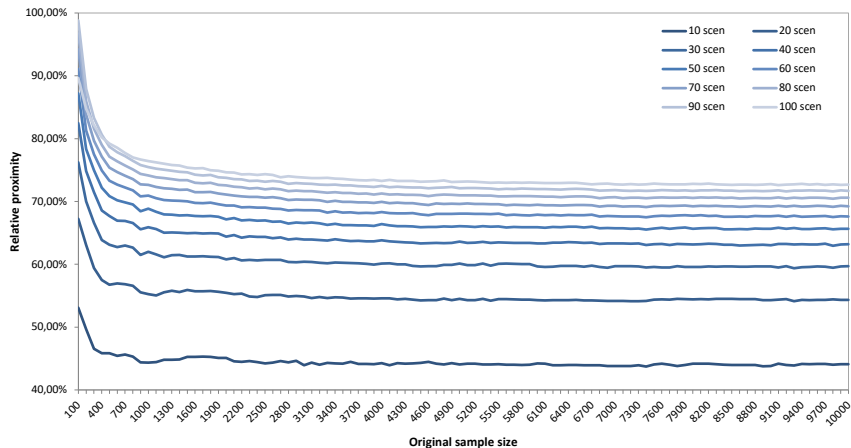
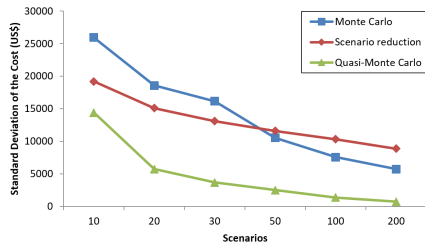


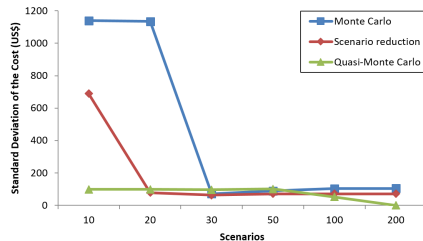
Figure: Relative accuracy for scenario reduction;  $x$ -axis is  $|\xi^1|$ , lines are different  $|\xi^2|$ .  
[Oliveira et al., 2016]

# Scenario generation methods

Some of my own experience



(a) In-sample



(b) Out-of-sample

**Figure:** Objective function standard deviation comparing 3 alternative scenario reduction methods. Original sample had 1000 scenarios [Fernández Pérez et al., 2018]

# Outline of this lecture

Introduction

Scenario trees

Generating scenario trees

Scenario (tree) generation methods

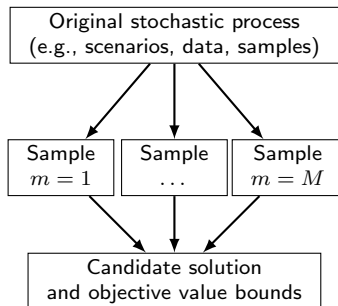
Sample Average Approximation (SAA)



# What is SAA

In the context of stochastic programming, SAA [Shapiro and Homem-de Mello, 1998] is an **alternative** to generating scenario trees

- ▶ Purely based on **sampling**;
- ▶ Monte Carlo simulation for estimating objective function bounds;
- ▶ Useful for handling **large** scenario sets;
- ▶ Typically, sample  $m$  size  $N \ll |\xi|$  or  $|\eta|$ ;
- ▶ Requires solving  $M$  problems.



## How SAA works

SAA is based on the **law of large numbers** (LLN) and the **central limit theorem** (CLT). As such, we can

- ▶ Estimate bounds using mean values;
- ▶ Estimate confidence intervals.

---

<sup>1</sup> $f(x)$  is a shorthand for  $\mathcal{F}(x, \xi)$ .

## How SAA works

SAA is based on the **law of large numbers** (LLN) and the **central limit theorem** (CLT). As such, we can

- ▶ Estimate bounds using mean values;
- ▶ Estimate confidence intervals.

First, let us define our notation for **2SSPs**

$$z = \min_x f(x),$$

where:

- ▶  $f(x) = \mathbb{E}_\xi [F(x, \xi)]^1$
- ▶  $F(x, \xi) = \{c^\top x + Q(x, \xi) : x \in X\};$
- ▶  $Q(x, \xi) = \min_y \{q(\xi)^\top y : W(\xi)y = h(\xi) - T(\xi)x, y \geq 0\};$
- ▶  $X = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}.$

---

<sup>1</sup> $f(x)$  is a shorthand for  $\mathcal{F}(x, \xi)$ .

# How SAA works

## Calculating a lower bounds for $z$

Let  $N$  be the number of samples we draw from our original stochastic process, forming the set scenario set  $S = \{\xi^1, \dots, \xi^N\}$ .

Then, we can solve the sample-based approximation problem

$$\hat{z}_N = \min_x \left\{ \tilde{f}_N(x) = \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right\}. \quad (1)$$

---

<sup>2</sup>LLN:  $\lim_{N \rightarrow \infty} \mathbb{E} \left[ \frac{\sum_{n=1}^N X_n}{N} \right] = \frac{N\bar{X}}{N} = \bar{X}$  for i.i.d. random variable  $X_n$  with mean value  $\bar{X}$ .

# How SAA works

## Calculating a lower bounds for $z$

Let  $N$  be the number of samples we draw from our original stochastic process, forming the set scenario set  $S = \{\xi^1, \dots, \xi^N\}$ .

Then, we can solve the sample-based approximation problem

$$\hat{z}_N = \min_x \left\{ \tilde{f}_N(x) = \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right\}. \quad (1)$$

First, notice that  $\tilde{f}_N(x)$  is an unbiased estimator<sup>2</sup> for  $f(x)$ :

$$\mathbb{E}_{\xi} \left[ \tilde{f}_N(x) \right] = \frac{1}{N} \mathbb{E}_{\xi} \left[ \sum_{n=1}^N F(x, \xi_n) \right] \xrightarrow{LLN} \frac{1}{N} (N f(x)) = f(x). \quad \square$$

---

<sup>2</sup>LLN:  $\lim_{N \rightarrow \infty} \mathbb{E} \left[ \frac{\sum_{n=1}^N X_n}{N} \right] = \frac{N\bar{X}}{N} = \bar{X}$  for i.i.d. random variable  $X_n$  with mean value  $\bar{X}$ .

# How SAA works

## Calculating lower bounds for $z$

We now show that  $\mathbb{E}[\hat{z}_N]$  is a **lower bound** on  $z$ :

$$\begin{aligned}\hat{z}_N &= \min_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right\} \leq \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \\ \mathbb{E}_\xi \left[ \min_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right\} \right] &\leq \mathbb{E}_\xi \left[ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right] \\ \mathbb{E}_\xi [\hat{z}_N] &\leq \mathbb{E}_\xi \left[ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right] \\ \mathbb{E}_\xi [\hat{z}_N] &\leq \min_x \left\{ \mathbb{E}_\xi \left[ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right] \right\} \xrightarrow{N \rightarrow \infty} \\ &\min_x \{ \mathbb{E}_\xi [F(x, \xi)] \} = \min_x f(x) = z. \quad \square\end{aligned}$$

# How SAA works

Calculating lower bounds for  $z$

In turn, we can approximate  $\mathbb{E}[\hat{z}_N]$  using a sample estimate.

1. For that, we sample  $M$  scenario trees of size  $N$ :

$$\{\xi_1^1, \dots, \xi_N^1\}, \dots, \{\xi_1^M, \dots, \xi_N^M\}.$$

---

<sup>3</sup>Again an unbiased estimator, see footnote 2.

# How SAA works

Calculating lower bounds for  $z$

In turn, we can approximate  $\mathbb{E}[\hat{z}_N]$  using a sample estimate.

1. For that, we sample  $M$  scenario trees of size  $N$ :

$$\{\xi_1^1, \dots, \xi_N^1\}, \dots, \{\xi_1^M, \dots, \xi_N^M\}.$$

2. For each scenario tree, we solve

$$\hat{z}_N^m = \min_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n^m) \right\}$$

---

<sup>3</sup>Again an unbiased estimator, see footnote 2.



# How SAA works

Calculating lower bounds for  $z$

In turn, we can approximate  $\mathbb{E}[\hat{z}_N]$  using a sample estimate.

1. For that, we sample  $M$  scenario trees of size  $N$ :

$$\{\xi_1^1, \dots, \xi_N^1\}, \dots, \{\xi_1^M, \dots, \xi_N^M\}.$$

2. For each scenario tree, we solve

$$\hat{z}_N^m = \min_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n^m) \right\}$$

3. We can then estimate<sup>3</sup>  $\mathbb{E}[\hat{z}_N]$  as

$$L_N^M = \frac{1}{M} \sum_{m=1}^M \hat{z}_N^m.$$

---

<sup>3</sup>Again an unbiased estimator, see footnote 2.

# How SAA works

Statistical bounds for  $L_N^M$

We can use the CLT to provide **confidence intervals** for  $L_N^M$ . A sample-estimate for  $\sigma_{L_N^M}^2$  can be obtained as

$$s_{L_N^M}^2 = \frac{1}{M-1} \sum_{m=1}^M (\hat{z}_N^m - L_N^M)^2.$$

# How SAA works

Statistical bounds for  $L_N^M$

We can use the CLT to provide **confidence intervals** for  $L_N^M$ . A sample-estimate for  $\sigma_{L_N^M}^2$  can be obtained as

$$s_{L_N^M}^2 = \frac{1}{M-1} \sum_{m=1}^M (\hat{z}_N^m - L_N^M)^2.$$

We can use  $s_{L_N^M}^2$  to obtain an  **$1-\alpha$  confidence interval** for  $L_N^M$ :

$$\left[ L_N^M - \frac{z_{\alpha/2} s_{L_N^M}}{\sqrt{M}}, L_N^M + \frac{z_{\alpha/2} s_{L_N^M}}{\sqrt{M}} \right]$$

where  $z_{\alpha/2}$  is the standard normal  $1 - \alpha/2$  quantile.

# How SAA works

Calculating upper bounds for  $z$

Let

$$\hat{x}_N^m = \operatorname{argmin}_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n^m) \right\}, \quad \forall m \in [M].$$

Notice that  $f(\hat{x}_N^m) \geq z, \forall m \in [M]$ .

# How SAA works

Calculating upper bounds for  $z$

Let

$$\hat{x}_N^m = \operatorname{argmin}_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n^m) \right\}, \quad \forall m \in [M].$$

Notice that  $f(\hat{x}_N^m) \geq z, \forall m \in [M]$ .

We can obtain an unbiased estimate for  $f(\hat{x}_N^m)$  by

1. **Choosing** one solution  $\hat{x}_N^{m'}, m' \in [M]$ ;
2. Sampling  $T$  scenario trees of size  $\bar{N}$

$$\{\xi_1^1, \dots, \xi_{\bar{N}}^1\}, \dots, \{\xi_1^T, \dots, \xi_{\bar{N}}^T\}$$

3. For each scenario tree  $t$ , we evaluate

$$\check{z}_N^t = \frac{1}{\bar{N}} \sum_{n=1}^{\bar{N}} F(\hat{x}_N^{m'}, \xi_n^t)$$

# How SAA works

Calculating upper bounds for  $z$

4. We can estimate  $f(\hat{x}_N^m)$  as

$$U_N^T = \frac{1}{T} \sum_{t=1}^T z_N^t.$$

Analogously, we can use the sample-estimate for  $\sigma_{U_N^T}^2$

$$s_{U_N^T}^2 = \frac{1}{T-1} \sum_{t=1}^T (z_N^t - U_N^T)^2$$

to calculate the **1- $\alpha$  confidence interval** for  $U_N^T$  as

$$\left[ U_N^T - \frac{z_{\alpha/2} s_{U_N^T}}{\sqrt{T}}, U_N^T + \frac{z_{\alpha/2} s_{U_N^T}}{\sqrt{T}} \right].$$

## On estimating optimality gaps

In this context, an **optimality gap** refers to the quantity

$$f(\hat{x}_N^{m'}) - z.$$

## On estimating optimality gaps

In this context, an **optimality gap** refers to the quantity

$$f(\hat{x}_N^{m'}) - z.$$

On the other hand, we know that

$$\mathbb{E}[\hat{z}_N] \leq z \leq f(\hat{x}_N^{m'}).$$

Since we have estimates for  $\mathbb{E}[\hat{z}_N]$  ( $L_N^M$ ) and  $f(\hat{x}_N^{m'})$  ( $U_N^T$ ), we can calculate the **optimality gap** estimate

$$gap(N, M, \bar{N}, T) = U_N^T - L_N^M.$$



## On estimating optimality gaps

In this context, an **optimality gap** refers to the quantity

$$f(\hat{x}_N^{m'}) - z.$$

On the other hand, we know that

$$\mathbb{E}[\hat{z}_N] \leq z \leq f(\hat{x}_N^{m'}).$$

Since we have estimates for  $\mathbb{E}[\hat{z}_N]$  ( $L_N^M$ ) and  $f(\hat{x}_N^{m'})$  ( $U_N^T$ ), we can calculate the **optimality gap** estimate

$$gap(N, M, \bar{N}, T) = U_{\bar{N}}^T - L_N^M.$$

Confidence intervals can also be obtained for  $gap(N, M, \bar{N}, T)$  using

$$\sigma_{gap(N, M, \bar{N}, T)}^2 = s_{L_N^M}^2 + s_{U_{\bar{N}}^T}^2.$$

## On estimating optimality gaps

Some remarks on  $gap(N, M, \overline{N}, T)$ :

- ▶  $gap(N, M, \overline{N}, T)$  is a **biased** estimator, since

$$f(\hat{x}_N^{m'}) - \mathbb{E}[\hat{z}_N] \geq f(\hat{x}_N^{m'}) - z;$$

- ▶ As it overestimates  $f(\hat{x}_N^{m'}) - z$ , it is still useful in practice;
- ▶ Confidence intervals for  $gap(N, M, \overline{N}, T)$  can be **improved** by reducing:
  1.  $s_{L_N^M}^2$ , via increasing  $N$  and  $M$ : larger  $N$  leads to larger problems, but they can be solved as  $M$  **parallel** problems;
  2.  $s_{U_N^T}^2$ , via increasing  $\overline{N}$  and  $T$ ; larger  $\overline{N}$  leads to more costly evaluation; solvable as  $T$  (as  $\overline{N} \times T$  for 2SSPs) parallel problems.

## Final practical remarks

Regarding choosing a solution  $\hat{x}_N^{m'}$ :

- ▶ If feasible, evaluate all distinct solutions  $\hat{x}_N^m$  for  $m \in \{M\}$  and choose that with **best**  $L_N^M$ ,  $U_N^T$  or  $gap(N, M, \bar{N}, T)$ ;
- ▶ Too many **distinct** solutions may indicate that  $N$  is too **small**. Perform stability analysis.
- ▶ SAA holds for **non-independent** sampling schemes (e.g., Latin hypercube sampling or quasi Monte Carlo). These help keep  $N$  small.

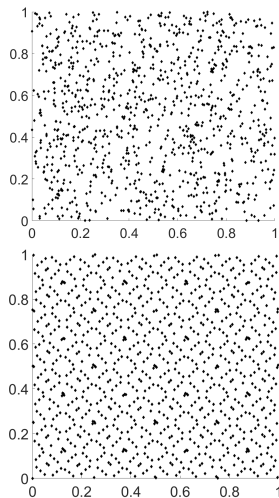


Figure: Monte Carlo (left) and quasi-Monte Carlo (right) sampling [Fernández Pérez et al., 2018]

## Final practical remarks

Regarding the choice of  $N$  [Oliveira and Hamacher, 2012]:

- ▶ Notice that  $\hat{z}_N$  is the expected value of the random variable

$$z_N(\xi) = F(\hat{x}_N, \xi), \text{ where } \hat{x}_N = \operatorname{argmin}_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right\}$$

## Final practical remarks

Regarding the choice of  $N$  [Oliveira and Hamacher, 2012]:

- ▶ Notice that  $\hat{z}_N$  is the expected value of the random variable

$$z_N(\xi) = F(\hat{x}_N, \xi), \text{ where } \hat{x}_N = \operatorname{argmin}_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right\}$$

- ▶ As such, we can estimate its sample-based variance and a  $1 - \alpha$  confidence interval, given by

$$s_N^2 = \frac{1}{N-1} \sum_{n=1}^N (\hat{z}_N - z_N(\xi_n))^2 \text{ and } \hat{z}_N \pm \frac{z_{\alpha/2} s_N}{\sqrt{N}}.$$

## Final practical remarks

Regarding the choice of  $N$  [Oliveira and Hamacher, 2012]:

- ▶ Notice that  $\hat{z}_N$  is the expected value of the **random variable**

$$z_N(\xi) = F(\hat{x}_N, \xi), \text{ where } \hat{x}_N = \operatorname{argmin}_x \left\{ \frac{1}{N} \sum_{n=1}^N F(x, \xi_n) \right\}$$

- ▶ As such, we can estimate its sample-based variance and a  $1 - \alpha$  confidence interval, given by

$$s_N^2 = \frac{1}{N-1} \sum_{n=1}^N (\hat{z}_N - z_N(\xi_n))^2 \text{ and } \hat{z}_N \pm \frac{z_{\alpha/2} s_N}{\sqrt{N}}.$$

- ▶ If we predefine a desired **relative width**  $\beta$  for the confidence interval, we can infer that

$$N \geq \frac{z_{\alpha/2} s_N}{(\beta/2) \hat{z}_N}.$$

# References I



Condeixa, L., Oliveira, F., and Siddiqui, A. S. (2020).

Wasserstein-distance-based temporal clustering for capacity-expansion planning in power systems.

*In 2020 International Conference on Smart Energy Systems and Technologies (SEST)*, pages 1–6. IEEE.



Dillon, M., Oliveira, F., and Abbasi, B. (2017).

A two-stage stochastic programming model for inventory management in the blood supply chain.

*International Journal of Production Economics*, 187:27–41.







Dupačová, J. (1990).

Stability and sensitivity-analysis for stochastic programming.

*Annals of operations research*, 27:115–142.

## References II

-  Dupačová, J., Gröwe-Kuska, N., and Römisch, W. (2003).  
Scenario reduction in stochastic programming.  
*Mathematical programming*, 95:493–511.
-  Fernández Pérez, M. A., Oliveira, F., and Hamacher, S. (2018).  
Optimizing workover rig fleet sizing and scheduling using deterministic and stochastic programming models.  
*Industrial & engineering chemistry research*, 57(22):7544–7554.
-  Heitsch, H. and Römisch, W. (2003).  
Scenario reduction algorithms in stochastic programming.  
*Computational optimization and applications*, 24:187–206.
-  Heitsch, H. and Römisch, W. (2007).  
A note on scenario reduction for two-stage stochastic programs.  
*Operations Research Letters*, 35(6):731–738.



## References III



Heitsch, H. and Römisch, W. (2009).

Scenario tree modeling for multistage stochastic programs.  
*Mathematical Programming*, 118:371–406.



Heitsch, H., Römisch, W., and Strugarek, C. (2006).

Stability of multistage stochastic programs.  
*SIAM Journal on Optimization*, 17(2):511–525.



Høyland, K., Kaut, M., and Wallace, S. W. (2003).

A heuristic for moment-matching scenario generation.  
*Computational optimization and applications*, 24:169–185.



Høyland, K. and Wallace, S. W. (2001).

Generating scenario trees for multistage decision problems.  
*Management science*, 47(2):295–307.

## References IV



Kaut, M. (2021).

Scenario generation by selection from historical data.  
*Computational Management Science*, 18(3):411–429.



Löhndorf, N. (2016).




An empirical analysis of scenario generation methods for stochastic optimization.  
*European Journal of Operational Research*, 255(1):121–132.



Oliveira, F. and Hamacher, S. (2012).

Optimization of the petroleum product supply chain under uncertainty: A case study in northern brazil.  
*Industrial & Engineering Chemistry Research*, 51(11):4279–4287.

## References V

-  Oliveira, F., Nunes, P. M., Blajberg, R., and Hamacher, S. (2016).  
A framework for crude oil scheduling in an integrated terminal-refinery system under supply uncertainty.  
*European Journal of Operational Research*, 252(2):635–645.
-  Pflug, G. C. (2001).  
Scenario tree generation for multiperiod financial optimization by optimal discretization.  
*Mathematical programming*, 89:251–271.
-  Römisch, W. (2003).  
Stability of stochastic programming problems.  
*Handbooks in operations research and management science*, 10:483–554.

## References VI



Schultz, R. (2000).

Some aspects of stability in stochastic programming.

*Annals of Operations Research*, 100:55–84.



Shapiro, A. and Homem-de Mello, T. (1998).

A simulation-based approach to two-stage stochastic programming with recourse.

*Mathematical Programming*, 81(3):301–325.