Scenario Tree Reduction via Wasserstein Barycenters

Daniel Mimouni[1,2*], Paul Malisani[2], Jiamin Zhu[3], Welington de Oliveira[1]

[1]Centre de Mathématiques Appliquées (CMA), Mines Paris PSL, Sophia Antipolis, 06560, France.
[2]Department of Applied Mathematics, IFP Energies nouvelles, Rueil-Malmaison, 92063, France.
[3]Department of Control Signals and Systems, IFP Energies nouvelles, Rueil-Malmaison, 92063, France.

*Corresponding author(s). E-mail(s): daniel.mimouni@ifpen.fr;
Contributing authors: paul.malisani@ifpen.fr; jiamin.zhu@ifpen.fr; welington.oliveira@minesparis.psl.eu;

Abstract

Scenario tree reduction techniques are essential for achieving a balance between an accurate representation of uncertainties and computational complexity when solving multistage stochastic programming problems. In the realm of available techniques, the Kovacevic and Pichler algorithm (Ann. Oper. Res., 2015 [1]) stands out for employing the nested distance, a metric for comparing multistage scenario trees. However, dealing with large-scale scenario trees can lead to a prohibitive computational burden due to the algorithm's requirement of solving several large-scale linear problems per iteration. This study concentrates on efficient approaches to solving such linear problems, recognizing that their solutions are Wasserstein barycenters of the tree nodes' probabilities on a given stage. We leverage advanced optimal transport techniques to compute Wasserstein barycenters and significantly improve the computational performance of the Kovacevic and Pichler algorithm. Our boosted variants of this algorithm are benchmarked on several multistage scenario trees. Our experiments show that compared to the original scenario tree reduction algorithm, our variants can be eight times faster for reducing scenario trees with 8 stages, 78 125 scenarios, and 97 656 nodes.

Keywords: Nested Wasserstein Distance , Scenario Tree Reduction, Wasserstein Barycenter, Optimal Transport, Stochastic Optimization, Scenario Selection.

MSC Classification: 46N10 , 90C08 , 49Q22 , 90C15.

1 Introduction

Stochastic optimization techniques have proved essential in solving optimization problems in the presence of uncertainties driven by variables such as fluctuating prices, unpredictable demand, supply variations, resource availability, and scheduling intricacies. The notion of stochastic programming is pioneered by Dantzig [2], and applications can be found in various sectors, including the financial industry [3, 4], supply chains [5], management science [6], energy economics [7, 8], electrical markets [9], hydro-thermal power systems [10] and maintenance of units [11].

Multistage stochastic programming, a class of stochastic optimization, often relies on scenario trees to represent the underlying stochastic process. In the quest for an accurate representation of uncertainties, large scenario trees are required. However, as the number of scenarios increases, so does the complexity of the problem and the computational effort regardless of the optimization method, whether it is based on scenario decomposition or stage decomposition [12]. Hence, finding a balance between an accurate representation of uncertainties and numerical tractability is of paramount importance in real-life applications modeled as multistage stochastic optimization problems.

In 2003, the influential work [9] introduced scenario reduction techniques based on the minimization of the Wasserstein distance between two scenario trees, pioneering the forward reduction and backward selection methodologies. Based on these ideas, [10] proposes to compute the Wasserstein distance at different nodes of a multistage scenario tree to design a scenario tree reduction algorithm. Differently, [13] formulates the scenario reduction problem as a mixed-integer linear programming problem. To decrease the computational effort when dealing with the Wasserstein distance, which can be formulated as a linear programming (LP) problem for finite sample of scenarios, subsequent works [14, 15] employ entropy-regularization schemes leveraged by the Sinkhorn–Knopp algorithm [16]. However, being based on the Wasserstein distance, these techniques ignore the filtration (structure) of multistage scenario trees. Neglecting the tree filtration potentially leads to deviations in solutions, because the solution of a multistage stochastic optimization problem is non-anticipative. As an attempt to cope with this shortcoming, the work [17] takes into consideration the so-called filtration distance, which relies on the tree structure. The stability results in that paper are the basis for the scenario tree reduction approach proposed in [18], which is a Wasserstein-based method that measures distances between nodes with the same parent, ultimately reducing pairs of nodes until a stopping criterion is met. However, this method encounters computational challenges, as it relies on solving NP-hard facility location problems. The work [19] proposes a scenario tree reduction algorithm that circumvents this difficulty by clustering tree nodes based on a new filtration distance and computes an approximation of the reduction problem. Scenario reduction strategies

based on clustering are numerous in the literature, but often lack stability properties; see, for instance, [20, 21] and references therein.

A milestone in the field of scenario tree reduction was the introduction of the *nested distance* in [22], subsequently studied in [23]. The nested distance, also called *process distance*, offers a valuable framework for comparing multistage scenario trees as it considers the underlying filtrations. Leveraging the nested distance to guide scenario tree reduction enables control over the reduced tree's statistical quality and its impact on the objective value of the underlying multistage stochastic optimization problem. While it is relatively easy to calculate the nested distance between two given trees [24], finding a tree (with given filtration) that minimizes the nested distance is a much more challenging task. The reason is that the approximating tree's probabilities and support (i.e., the scenarios or outcomes) must be chosen to minimize the nested distance. This leads to a challenging optimization problem, which is large-scale and nonconvex. To tackle such a problem, Kovacevic and Pichler [1] introduced an algorithm that directly targets the minimization of the nested distance by alternating between probability and support optimizations. While the last task has an explicit solution (should the Euclidean norm be employed as a metric in the nested distance), optimizing the probabilities amounts to solving several large-scale LPs per stage. This represents the main bottleneck of the scenario tree reduction algorithm of [24], which can be partially avoided in some special cases. For instance, the work [8] provides a variant of the algorithm capable of efficiently handling large-scale multistage scenario trees provided the stochastic process is *stage-wise independent*, a strong assumption we do not assume in this work. Hence, for general stochastic processes, there is a clear need for more practical scenario tree reduction approaches based on the nested distance. This work contributes in this direction by boosting the Kovacevic and Pichler (KP) algorithm [24].

One of our contributions stems from recognizing that the algorithm's most time-consuming task, the probability optimization step, amounts to computing Wasserstein barycenters of the tree nodes' probabilities. In addition, we leverage advanced optimal transport techniques to compute Wasserstein barycenters within the KP's algorithm by employing efficient and dedicated approaches such as the Iterative Bregmann Projection method (IBP) [25] and the Method of Averaged Marginals (MAM) [26], to attractively boost the computational performance for reducing scenario trees. As a third contribution, we benchmark our variants of the KP algorithm on real-life data and empirically show that they significantly outperform the baseline KP algorithm for large-scale multistage scenario trees. Our findings alleviate the algorithm's bottleneck, helping thus to elevate the KP approach among the top algorithmic choices for scenario tree reduction. We also emphasize the importance of the filtration initialization in the finding of a reduced tree.

The remainder of this work is organized as follows. Section 2 presents the Kovacevic and Pichler's approach and recalls the mathematical background for the nested distance. The barycentric approaches are introduced in Section 3, and the improved variant of the scenario tree reduction algorithm is developed. Some applications in Section 4 compare the new approach with the original algorithm by Kovacevic and Pichler, concluding with a speed-up method designed for real-life problems.

2 The Kovacevic and Pichler's approach

In [22], Pflug introduced the Nested Distance (ND), which is built on the Wasserstein distance, exploiting its structure and extending it to accommodate the specific characteristics of stochastic processes (see also [23]). It is a tailor-made measure for stochastic processes: it inherits the foundational properties of the Wasserstein distance, such as its sensitivity to local structure and robustness to outliers, while providing a more nuanced and specialized measure for comparing the similarities between different realizations of stochastic processes.

2.1 Scenario trees and notations

A T-period scenario tree ($\mathcal{N}$, A) is a discrete form of a random process - a family of random variables (see [1] for more). It is a composed of (a set of) nodes $\mathcal{N}$ and edges A.

A node $m \in \mathcal{N}$ is a direct predecessor or parent of the node $n \in \mathcal{N}$, if $(m, n) \in A$, where (m, n) denotes the edge between the nodes m and n. This relation is embodied by the notation $m = n-$. Reciprocally, n is a direct successor (or child) of m and this set is denoted $m+$, such that $n \in m+$ if and only if $m = n-$. In the same vein, we denote the predecessors (or ancestors) of $n \in \mathcal{N}$ as $\mathcal{A}(n)$, the set of nodes for which there exists a path to n: for example $m_1 = m_2-$ and $m_2 = n-$ (equivalently $n \in m_2+$ and $m_2 \in m_1+$), then $m_1, m_2 \in \mathcal{A}(n)$. We consider only trees with a single root, denoted by 0, i.e., $0- = \emptyset$, but the methods developed here can be generalized for multi-root trees (forests). Nodes $n_T \in \mathcal{N}$ without successor nodes (i.e., $n_T+ = \emptyset$) are called leaf nodes. For every leaf node n_T there is a sequence $\xi_{n_T} := (n_0, \ldots, n_t, \ldots, n_T)$ where $n_0 \in \mathcal{A}(n_1)$, $n_1 \in \mathcal{A}(n_2)$ etc, from the root to the leaf node n_T composed by $T + 1$ nodes. $\mathcal{F}_t$ is the filtration generated by the sigma-algebra of the family $(\xi_n)_{n \in \mathcal{N}_t}$ and we define $\mathcal{F} := (\mathcal{F}_t)_{t=1,\ldots,T}$. The nodes bear a value named *quantizer*: $\xi : \mathcal{N} \mapsto \Xi$, we call $\xi(n)$ the quantizer of node n, where Ξ is a finite dimension metric space. We denote the probability, assigned to node n, by $P(n)$, that satisfies: $P(n) = \sum_{\tilde{n} \in n+} P(\tilde{n})$ for $n \in \mathcal{N}$ and $\sum_{n \in \mathcal{N}_T} P(n) = 1$. We denote conditional probability between successors by $P(n|m) = P(n)/P(m)$ for $m = n-$. Furthermore, using a distance $\mathtt{d} : \Xi^{t+1} \times \Xi^{t+1} \to \mathbb{R}$, we denote the distance between two nodes n_1, n_2 at stage t, respectfully, as:

$$\mathtt{d}_{n_1,n_2} := \sum_{n \in \xi_{n_1}, \bar{n} \in \xi_{n_2}, n \text{ and } \bar{n} \text{ at stage } t} \tilde{\mathtt{d}}(\xi(n), \xi(\bar{n})) \tag{1}$$

Where $\tilde{\mathtt{d}}$ is a distance on Ξ.

2.2 Nested Distance for Trees

Using the tree notations introduced in Section 2.1, the process distance of order ι between two trees $\mathbf{P} := (\Xi^{T+1}, \mathcal{F}, P)$ and $\mathbf{P}' := (\Xi^{T+1}, \mathcal{F}', P')$ is the *discrete Nested Distance for Trees* (NDT). The transport mass between node $i \in \mathcal{N}_t$ and node $j \in \mathcal{N}'_t$ at stage $t \in \{1, \ldots, T\}$, is noted $\pi_{i,j}$ or $\pi(i, j)$.

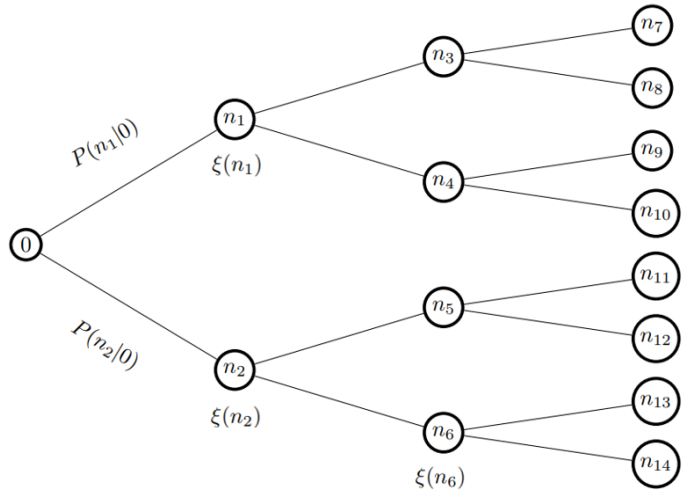


Fig. 1: Scenario tree notations.

Definition 1 (Nested Distance for Trees) For $\iota \in [1, \infty)$, the process distance of order ι between $\mathbf{P}$ and $\mathbf{P}'$ is the ιth root of the optimal value of the following LP:

$$\mathrm{ND}_\iota(\mathbf{P}, \mathbf{P}') := \begin{cases} \min\limits_{\pi} & \sum\limits_{i \in \mathcal{N}_T, j \in \mathcal{N}'_T} \pi(i,j) \mathtt{d}^\iota_{i,j} \\ \text{s.t.} & \sum_{\{j: n \in \mathcal{A}(j)\}} \pi(i,j|m,n) = P(i|m), \ \ (m \in \mathcal{A}(i), n) \\ & \sum_{\{i: m \in \mathcal{A}(i)\}} \pi(i,j|m,n) = P'(j|n), \ \ (n \in \mathcal{A}(j), m) \\ & \pi_{i,j} \geq 0 \text{ and } \sum_{i,j} \pi_{i,j} = 1. \end{cases} \tag{NDT}$$

Note that (NDT) is a generalization of the Wasserstein distance. Indeed, the transport plan π does not only respect the marginals imposed by P and P' but also respects the conditional marginals. These constraints embed the filtration in the definition of the distance between trees. Note that in the following, the term *ND* is used to refer to the value of (NDT).

2.3 The Kovacevic and Pichler algorithm for scenario tree reduction

The scenario tree reduction mechanism involves solving structure-like (NDT) problems, between the original tree and the smaller one. The latter has given filtration (same number of stages but considerably fewer number of nodes than the original tree), initialized probabilities and quantizer values.

The scenario reduction optimization problem is non-convex, due to the optimization of both quantizers and probabilities. The method in [1] operates a classical block coordinate optimization scheme. As illustrated in Figure 2, after a filtration is chosen, the first step is the optimization of the probabilities P' for fixed quantizers, and the second step is the optimization of the quantizer values $\{\xi'(n) \in \Xi : n \in \mathcal{N}'\}$ for fixed probability. The latter step has an exact analytical solution in the Euclidean case, i.e. $\iota = 2$, and when $\tilde{\mathtt{d}}$ in (1) is the Euclidean norm [1]. The probability optimization is more difficult because it requires solving multiple (potentially large-scale) LPs. In the remainder of this work, we will only develop the probability optimization, the quantizer optimization will only be recalled briefly in Algorithm 3.

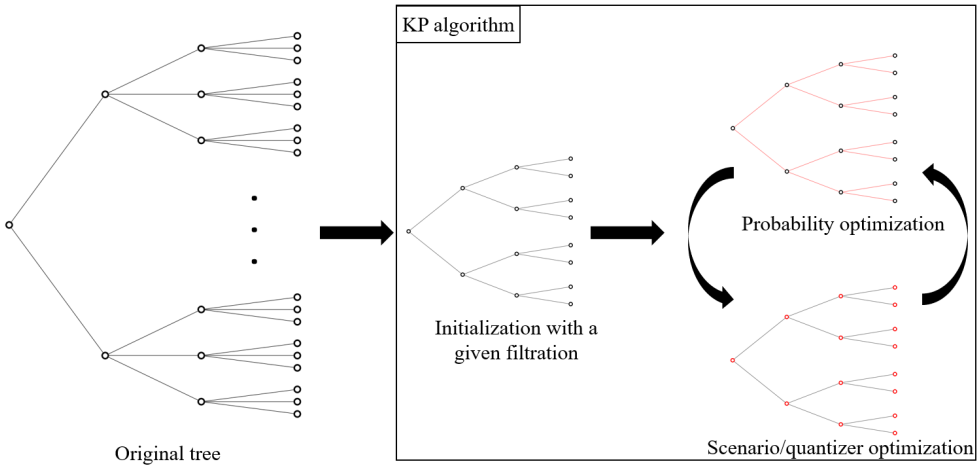


Fig. 2: A general scheme of the Kovacevic and Pichler algorithm: to approximate a tree, a smaller tree with a given filtration is improved in order to minimize the nested distance with the original tree. The probabilities and the quantizers are alternatively optimized until convergence.

The probability optimization steps can be stated as follows: given the stochastic process quantizers $\{\xi'(n) \in \Xi : n \in \mathcal{N}'\}$ and structure of $(\mathcal{N}', A')$, we are looking for the optimal probability measure P' to approximate $\mathbf{P} := (\Xi^{T+1}, \mathcal{F}, P)$, regarding the nested distance. Inspecting formulation (NDT), it turns out that P' can be computed by jointly optimizing with respect to $\pi_{i,j}$ and $P'(j|j-)$. This leads to the following

large non-convex optimization problem:

$$\begin{cases} \min\limits_{\pi, P'} \sum\limits_{i \in \mathcal{N}_T, j \in \mathcal{N}'_T} \pi(i,j) \mathtt{d}^{\iota}_{i,j} \\ \\ \text{s.t.} \ \sum_{j \in n+} \pi(i,j|m,n) = P(i|m), \ \ (\forall m \in \mathcal{A}(i), n) \\ \sum_{i \in m+} \pi(i,j|m,n) = P'(j|n), \ (\forall n \in \mathcal{A}(j), m) \\ \pi_{i,j} \geq 0 \text{ and } \sum_{i,j} \pi_{i,j} = 1 \\ P'(j|j-) \geq 0. \end{cases} \tag{2}$$

This is a bilinear problem, hence difficult to handle: there is a large number of decision variables and bilinear constraints (issued by the conditional probabilities in the second group of constraints, which involve the decision variables composing π and P'). Using the conditional probabilities $\pi(i,j) = \pi(i,j|m,n) \times \pi(m,n)$, we can derive the recursive formula (3a) below.
Let $\delta_\iota(m,n) := \sum_{i \in m+, j \in n+} \pi(i,j|m,n) \delta_\iota(i,j)$ for $m \in \mathcal{N}_t, n \in \mathcal{N}'_t$, and $\delta_\iota(i,j) = \mathtt{d}_\iota(\xi_i, \xi'_j)^\iota =: \mathtt{d}^\iota_{i,j}$ for the leaves i, j of the trees.

$$\sum_{i \in \mathcal{N}_T, j \in \mathcal{N}'_T} \pi(i,j) \mathtt{d}^\iota_{i,j} = \sum_{i \in \mathcal{N}_T, j \in \mathcal{N}'_T} \pi(i,j) \delta_\iota(i,j) \tag{3a}$$

$$= \sum_{i \in \mathcal{N}_T, j \in \mathcal{N}'_T} \sum_{m \in i-, n \in j-} \pi(i,j|m,n) \pi(m,n) \delta_\iota(i,j) \tag{3b}$$

$$= \sum_{n \in \mathcal{N}'_{T-1}} \sum_{m \in \mathcal{N}_{T-1}} \pi(m,n) \underbrace{\sum_{i \in m+, j \in n+} \pi(i,j|m,n) \delta_\iota(i,j)}_{\delta_\iota(m,n)} \tag{3c}$$

$$= \sum_{n \in \mathcal{N}'_{T-1}} \sum_{m \in \mathcal{N}_{T-1}} \pi(m,n) \delta_\iota(m,n). \tag{3d}$$

Note also $\delta_\iota(0,0) = \mathrm{ND}(\mathbf{P}, \mathbf{P}')$, recursively. Thanks to this recursive formula, the problem can be split into recursive smaller problems for $m \in \mathcal{N}_t$ and $n \in \mathcal{N}'_t$: the

conditional probability $\pi(\cdot,\cdot|m,n)$ is a solution to

$$\begin{cases} \min\limits_{\pi} \sum_{m\in\mathcal{N}_t} \pi(m,n) \sum_{i\in m+, j\in n+} \pi(i,j|m,n)\delta_\iota(i,j) \\ \\ \text{s.t.} \quad \sum_{j\in n+} \pi(i,j|m,n) = P(i|m), & (i \in m+) \\ \qquad \sum_{i\in m+} \pi(i,j|m,n) = \sum_{i\in \tilde{m}+} \pi(i,j|\tilde{m},n), & (j \in n+ \text{ and } m,\tilde{m} \in \mathcal{N}_t) \\ \qquad \pi(i,j|m,n) \geq 0. \end{cases} \tag{RP}$$

This reformulation of the problem is still bilinear, however, to overcome this difficulty, [1] proposes to fix $\pi(m,n)$ with the values computed from the previous iteration (or initialized at first iteration) giving rise to a LP approximation. The authors have empirically shown that after few iterations of their algorithm the values assigned to $\pi(m,n)$ stabilize.

Despite this approximation, the problem is still challenging due to its huge dimensions. Thus, the method's main limitation is its computational burden that becomes prohibitive for large-scale scenario trees, as exemplified in Section 4. The reason is that the method requires solving potentially large-scale LPs as in (RP) repeatedly. We address the challenge by noticing that (RP) with fixed $\pi(m,n)$ for $m \in \mathcal{N}_t$ for a given n is a *Wassertein barycenter problem* (WB), for which specialized and efficient algorithms exist.

3 The Probability Optimization Step (RP) is a Wasserstein Barycenters Problem

We start with the Wasserstein distance definition.

Definition 2 (Wasserstein Distance) Given two probability measures $\mu, \nu \in P(\mathbb{R}^d)$, their Wasserstein distance is the ι-th root of

$$W_\iota^\iota(\mu,\nu) := \min_{\pi\in U(\mu,\nu)} \langle D, \pi\rangle. \tag{WD}$$

Where $\langle D, \pi\rangle := \sum_{r,s} D_{r,s}\pi_{r,s}$. The cost matrix D is composed by the distance values $(\mathtt{d}_{r,s})_{(r\times s)\in[1,R]\times[1,S]}$, where R and S are the support sizes of μ and ν respectively; π is the transport matrix between the two probability densities; and $U(\mu,\nu)$ is the set of all transport plans having marginals μ and ν.

Definition 3 (Wasserstein Barycenter) Given M measures $\{\nu^1,\ldots,\nu^M\}$ in $P(\mathbb{R}^d)$, an ι-*Wasserstein barycenter* with weights $\alpha \in \Delta_M$ is a solution to the following optimization problem:

$$\min_{\mu\in P(\mathbb{R}^d)} \sum_{m=1}^{M} \alpha_m W_\iota^\iota(\mu,\nu^m)\,. \tag{4}$$

In what follows we present one of our contributions.

Proposition 1 *The recursive problem* (RP) *with fixed* $\pi(m,n)$ *for* $m \in \mathcal{N}_t$ *and a given* n *is a Wasserstein Barycenter problem.*

Proof Let M empirical (discrete) measures ν^m having finite support sets:

$$\mathtt{supp}(\nu^m) := \left\{\xi_1'^m, \ldots, \xi_{S^m}'^m\right\} \quad \text{and} \quad \nu^m = \sum_{s=1}^{S^m} q_s^m \underline{\delta}_{\xi_s'^m}, \tag{5}$$

with $\underline{\delta}_u$ the Dirac unit mass on $u \in \mathbb{R}^d$ and $q^m \in \Delta_{S^m}$, $m = 1, \ldots, M$.
A barycenter $\mu := \sum_{r=1}^{R} p_r \underline{\delta}_{\xi_r}$ with $p \in \Delta_R$ of the family $(\nu^m)_{m\in[1,M]}$ is a solution to (4).

Problem (4) can be equivalently formulated as a LP, where π^m for $m = 1, \ldots, M$ denote the transport plan between the μ and the probability measures, see Section 2 of [26]:

$$\begin{cases} \min\limits_{p,\pi} \ \alpha_1 \sum\limits_{r=1}^{R}\sum\limits_{s=1}^{S^1} \mathtt{d}_{rs}^1 \pi_{rs}^1 + \cdots + \alpha_M \sum\limits_{r=1}^{R}\sum\limits_{s=1}^{S^M} \mathtt{d}_{rs}^M \pi_{rs}^M \\ \text{s.t.} \ \sum_{r=1}^{R} \pi_{rs}^1 = q_s^1, \quad s = 1, \ldots, S^1 \\ \qquad \ddots \qquad \vdots \\ \qquad \sum_{r=1}^{R} \pi_{rs}^M = q_s^M, \quad s = 1, \ldots, S^M \\ \sum_{s=1}^{S^1} \pi_{rs}^1 = p_r, \quad r = 1, \ldots, R \\ \qquad \ddots \qquad \vdots \\ \qquad \sum_{s=1}^{S^M} \pi_{rs}^M = p_r, \quad r = 1, \ldots, R \\ p \in \Delta_R, \pi^1 \geq 0 \quad \cdots \quad \pi^M \geq 0, \end{cases} \tag{6}$$

Let us now show that (RP) fits into this structure.
Let $\alpha_{m_i}^n := \pi(m_i, n)$ for $i = 1, \ldots, M$, $M = |\mathcal{N}_t|$, and $(P'(j|n))_{j\in n+}$ an auxiliary vector of probabilities allowing to reformulate and couple the last group of constraints in (RP):$\sum_{i\in m+} \pi(i,j|m,n) = P'(j|n)$ for $j \in n+$, for all $m = m_1, \ldots, m_M$. Given

$t \in \{1, \dots, T\}$ and $n \in \mathcal{N}'_t$, problem (RP) reads as:

$$\begin{cases} \min\limits_{P',\pi} \ \alpha_1^n \sum\limits_{i \in m_1+, j \in n+} \pi(i,j|m_1,n)\delta_\iota(i,j) \quad + \cdots + \ \alpha_M^n \sum\limits_{i \in m_M+, j \in n+} \pi(i,j|m_M,n)\delta_\iota(i,j) \\ \text{s.t.} \ \sum_{j \in n+} \pi(i,j|m_1,n) \qquad\qquad\qquad\qquad = P(i|m_1), \quad (i \in m_1+) \\ \qquad\qquad \ddots \qquad\qquad\qquad \vdots \\ \qquad\qquad\qquad \sum_{j \in n+} \pi(i,j|m_M,n) \quad = P(i|m_M), \quad (i \in m_M+) \\ \quad \sum_{i \in m_1+} \pi(i,j|m_1,n) \qquad\qquad\qquad = P'(j|n), \quad (j \in n+) \\ \qquad\qquad \ddots \qquad\qquad\qquad \vdots \\ \qquad\qquad\qquad \sum_{i \in m_M+} \pi(i,j|m_M,n) = P'(j|n), \quad (j \in n+) \\ \quad \sum_{j \in n+} P'(j|n) = 1, \pi(i,j|m_1,n) \geq 0 \ \cdots \qquad \pi(i,j|m_M,n) \geq 0, \end{cases} \tag{WB}$$

which is equivalent to

$$\min_{P'(.|n) \geq 0} \sum_{m \in \mathcal{N}_t} \alpha_m^n W_\iota^\iota((P'(j|n))_{j \in n+}, (P(i|m))_{i \in m+}) \quad \text{s.t.} \quad \sum_{j \in n+} P'(j|n) = 1,$$

i.e., a Wasserstein Barycenter problem.

□

To further illustrate our interpretation of problem (RP) as a Wassertein Barycenter problem, see Figure 3. The boxed subtree of the approximated (smaller) tree forms a probability measure with support (n_7, n_8) and probability $(P(n_7|n_3), P(n_8|n_3))$; the probabilities of this subtree helping to minimize the ND is a barycenter of the (original tree's) subtrees with initial nodes m_3, m_4, m_5, m_6. The Wasserstein Barycenter $(P(n_7|n_3), P(n_8|n_3))$ is computed by solving (WB), the next (approximated tree's) subtree, that is the one issued by node n_4 is solved in the same manner etc. The probability $(P(n_9|n_4), P(n_{10}|n_4))$ are computed by solving another barycenter problem, but with different weights $\alpha_m^{n_4}$ (and costs) for $m \in \{m_3, m_4, m_5, m_6\}$ (see eq. (WB)).

It is thus clear that several Wasserstein Barycenter problems must be solved at every iteration of the scenario tree reduction algorithm [1]. The faster the computation of such barycenters, the faster the Kovacevic and Pichler's algorithm.

3.1 Wasserstein barycenters techniques for scenario reduction

Problem from (WB) is an LP and could, in principle, be solved by LP solvers such as Gurobi, Cplex, HiGHs, and others. However, in real life applications of scenario tree reduction, problem (WB) (equivalently problem from (RP)) can have huge dimensions and be intractable by LP solvers, thus hindering the whole scenario tree reduction process. Many methods can be employed to tackle problem (WB) (for e.g. [25, 27, 28]). Note that these methods are typically employed to retrieve a barycentric probability measure. However, in the context of this work, we instead focus on directly extracting the transport plans between the measures and the barycenter. We will further develop these concepts in the following sections. In the remainder of the paper, we use two different WB algorithms, namely the Method of Averaged Marginals (MAM) [26] and

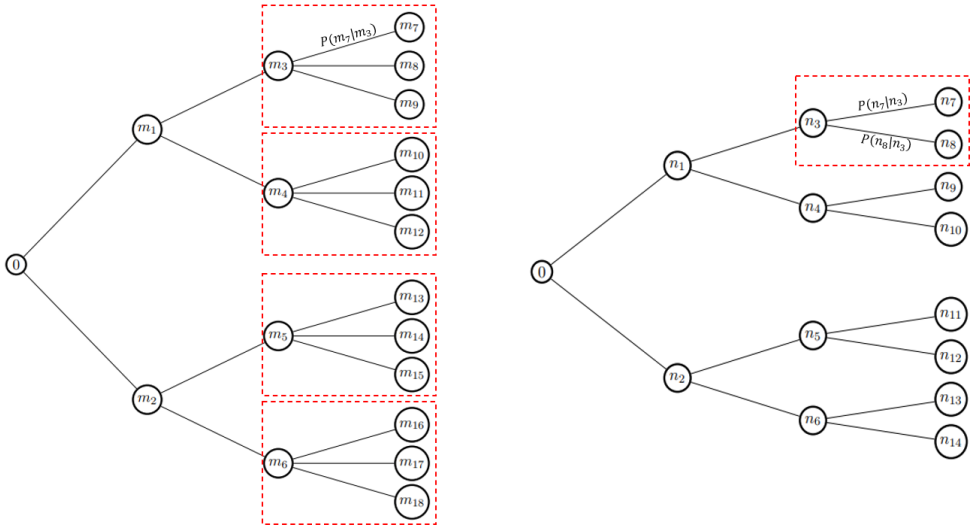


Fig. 3: *(left)* Original tree, *(right)* Approximated tree. The probabilities $(P'(n_7|n_3), P'(n_8|n_3))$ are computed as the Wasserstein Barycenter of the set of (known) probabilities associated to the boxed subtrees on the left.

the Iterative Bregmann Projection algorithm (IBP) [25]. In the following, we simplify the notation by denoting: $\pi(\cdot,\cdot|m_{\hat{m}}, n) := \pi^{\hat{m}} \in \mathbb{R}^{R\times S^{\hat{m}}}$ for all $\hat{m} = 1,\dots,M$, since n is fixed in (WB), and $P(\cdot|m)$ in (WB) is denoted by $q^{\hat{m}}$.

3.1.1 The Method of Averaged Marginals (MAM)

MAM's main idea [26] is to reformulate (WB) as the problem of finding a zero of the sum of two maximal monotone operators, which can be solved by the celebrated Douglas-Rachford algorithm [29, 30]. When specialized to (WB), the paper [26] shows that Douglas-Rachford algorithm boils down to Algorithm 1.

The projection step can be performed *exactly*, and in parallel, by using efficient specialized methods [31]. We employ the notation $\pi^m_{:s}$ to denote the sth column of the matrix π^m.

The *Method of Averaged Marginals*, depicted in Algorithm 1, leverages the transport plans from which one can extract marginals whose average is actually a barycenter approximation. The MAM algorithm asymptotically solves (WB) and produces both a barycenter and associate transport plans that are needed for the recursion problem (RP). Note that Step 2 and 3 can be computed in parallel over the M measures. We refer the interested reader to [26] for more details.

3.1.2 Computation of Transport Plans via Regularized Techniques

Utilizing efficient regularized methods enables us to quickly compute the transport plans required for the scenario tree reduction approach. For example, the *Iterative*

Algorithm 1 Method of Averaged Marginals - MAM

Input: Initial plan $\pi = (\pi^1, \dots, \pi^m)$ and parameter $\rho > 0$
Set $S^m \leftarrow |\ \texttt{supp}(q^m)\ |$, for $m = 1, \dots, M$
Define $a_m \leftarrow (\frac{1}{S^m})/(\sum_{j=1}^{M} \frac{1}{S^j})$ and set $p^m \leftarrow \sum_{s=1}^{S^m} \pi^m_{rs}$, $m = 1, \dots, M$
Set $D^m \leftarrow \alpha_m \left(\delta_\iota(i,j)\right)_{(i,j)\in m+\times n+}$ and set $q^m = (P(i|m))_{i\in m+}$

while not converged **do**

$p \leftarrow \sum_{m=1}^{M} a_m p^m$ ▷ Average the marginals

for $m = 1, \dots, M$ **do**
  for $s = 1, \dots, S^m$ **do**
    $\pi^m_{:s} \leftarrow \texttt{Proj}_{\Delta(q^m_s)}\left(\pi^m_{:s} + 2\frac{p-p^m}{S^m} - \frac{1}{\rho}D^m_{:s}\right) - \frac{p-p^m}{S^m}$
  end for
  $p^m \leftarrow \sum_{s=1}^{S^m} \pi^m_{rs}$ ▷ Update the mth marginal
end for

end while

Bregman Projection (IBP) [25] is a state-of-the-art technique, that applies regularization to the optimization problems presented in (WD) and (WB).

Consider the entropic function:

$$E(\pi) := \sum_{r,s} \pi_{r,s}(\log(\pi_{r,s}) - 1), \tag{7}$$

with the convention $0\log(0) = 0$. This is a strongly convex function which assures that $E(\pi) \geq 0$ and $E(\pi) = 0$ if and only if $\pi = 0$. This function is employed to regularize the LP (WD), leading to the following nonlinear optimization problem:

$$W_\lambda(\mu,\nu) := \min_{\pi\in U(\mu,\nu)} \langle D, \pi\rangle + \frac{1}{\lambda}E(\pi) \tag{8a}$$

$$= \min_{\pi\in U(\mu,\nu)} \frac{1}{\lambda}\sum_{r,s} \left(\lambda D_{r,s}\pi_{r,s} + \pi_{r,s}\log(\pi_{r,s}) - \pi_{r,s}\right) \tag{8b}$$

$$= \min_{\pi\in U(\mu,\nu)} KL(\pi|K) \tag{8c}$$

Note that KL is the Kullback-Leibler divergence between $\pi, K \in \mathbb{R}^{R\times S}$ and $K_{r,s} > 0$ for all (r,s): $KL(\pi|K) := \sum_{r,s} \pi_{r,s}\left(\log\left(\frac{\pi_{r,s}}{K_{r,s}}\right) - 1\right)$ (the precise definition of matrix K is given in the algorithm below).
By noticing that $p = \pi\mathbb{1}_R$, problem (WB) with W_λ introduced in (8), instead of the

classical Wasserstein distance, writes:

$$\min_{\substack{\pi^m \in U(\mu,\nu^m),\\ m=1,\dots,M}} \sum_{m=1}^{M} \alpha_m KL(\pi^m|K^m) \tag{9}$$

Since problem (9) is strongly convex, it has a unique optimal solution. Following [25], the optimal coupling $\pi^m, m = 1,\dots,M$ can be derived after iterative KL projections onto the right and left constraints, embodied by the sets $U(\mu,\nu^m), m = 1,\dots,M$.

Algorithm 2 IBP ALGORITHM

Input: Given α_m for $m = 1,\dots,M$, $\lambda > 0$, initialize v^0 and u^0 with an arbitrary positive vector, for example $\mathbb{1}_S$ Initialize p^0, for example $\mathbb{1}_R/R$
Set $D^m \leftarrow \alpha_m\,(\delta_\iota(i,j))_{(i,j)\in m+\times n+}$ and set $q^m = (P(i|m))_{i\in m+}$
Define $K^m = e^{-\lambda D^m}$ for all $m = 1,\dots,M$
while not converged **do**

▷ Projections onto the constraints

for m=1,...,M **do**
$v^{m,k+1} = \frac{q^m}{(K^m)^T u^{m,k}}$
$u^{m,k+1} = \frac{p^{k+1}}{K^m v^{m,k+1}}$
end for

▷ Approximation of the barycenter

$p^{k+1} = \prod_{m=1}^{M} (K^m v^{m,k+1})^{\alpha_m}$
end while

return $\pi^m = \mathrm{diag}(u^m) K^m \mathrm{diag}(v^m)$ for all $m = 1,\dots,M$

Observe that all the algorithm's steps consist of matrix-vector multiplication, and are thus simple to execute. Note that p^{k+1} is the current estimate of the barycenter in Algorithm 2. The algorithm's drawback is its accuracy, which strongly depends on $\lambda > 0$. The greater is λ the closer is the solution of (9) to an exact solution of (WB). However, if λ is too large, the values of K diverge, leading to computational issues such as double-precision overflow errors.

3.2 Boosted Kovacevic and Pichler's Algorithm

We rely on the previous subsections about Wasserstein barycenters computation methods to provide the following improved variant of the scenario tree reduction algorithm of [1]. In Algorithm 3, given a multistage scenario tree represented by $\mathbf{P} = (\Xi^{T+1}, \mathcal{F}, P)$ a smaller scenario tree $\mathbf{P}' = (\Xi^{T+1}, \mathcal{F}', P')$ is constructed by updating the quantizer values $\{\xi(n) \in \Xi : n \in \mathcal{N}\}$ and probability P', iteratively. The filtration $\mathcal{F}'$ is a

data given to the algorithm. In other words, the number of scenarios and the structure of the reduced tree is an input data, and the algorithm seeks for the familly $\{\xi'(n) \in \Xi : n \in \mathcal{N}'\}$ and P' that minimizes the nested distance between $\mathbf{P}$ and $\mathbf{P}'$.

Algorithm 3 SCENARIO TREE REDUCTION VIA NESTED DISTANCE AND WASSERSTEIN BARYCENTERS

▷ Step 0: input

Let the original T-stage scenario tree $\mathbf{P} = (\Xi^{T+1}, \mathcal{F}, P)$ and a smaller scenario tree $\mathbf{P}'^0 = (\Xi^{T+1}, \mathcal{F}', P'^0)$ be given.

Compute a transport probabilities $\pi^0(i,j)$ between scenarios $(\xi_i)_{i\in\mathcal{N}_T}$ and $\left(\xi_j'^0\right)_{j\in\mathcal{N}'_T}$

Set $k \leftarrow 0$ and choose a tolerance $\texttt{Tol} > 0$

for $k = 1, 2, \ldots$ **do**

▷ **Step 1:** Improve the scenario values (quantizers)

Set $\xi'^{k+1}(n_t) = \sum_{m\in\mathcal{N}_t} \frac{\pi^k(m,n_t)}{\sum_{i\in\mathcal{N}_t}\pi^k(i,n_t)}\xi_t(m)$ for all $n_t \in \mathcal{N}'_t$ for $t = 1, \ldots, T$

▷ **Step 2:** Improve the probabilities

Set $\delta_\iota^{k+1}(i,j) \leftarrow \mathtt{d}(\xi_i, \xi_j)^\iota$ for all $i, j \in \mathcal{N}_T$

for $t = T-1, \ldots, 0$ **do** ▷ Recursivity

for all $n \in \mathcal{N}'_t$ **do** ▷ Wasserstein barycenters

Set $\alpha_m^n \leftarrow \pi^k(m,n)$, $m \in \mathcal{N}_t$

Use IBP, or MAM to compute $\pi^{k+1}(\cdot,\cdot|\cdot,n)$ solving (WB)

Set $\delta_\iota^{k+1}(m,n) \leftarrow \sum_{i\in m+, j\in n+} \pi^{k+1}(i,j|m,n)\delta_\iota^{k+1}(i,j)$, $m \in \mathcal{N}_t$

end for

end for

▷ Build π^{k+1} the unconditional transport plan matrix

Set $\pi^{k+1}(0,0) \leftarrow 1$

for t=1,...,T **do**

Compute $\pi^{k+1}(i,j) = \pi^{k+1}(i,j|m,n) \times \pi^{k+1}(m,n)$ for $i \in \mathcal{N}_t, j \in \mathcal{N}'_t$ and $m = i-$, $n = j-$

end for

▷ **Step 3:** Stopping test

if $\delta_\iota^k(0,0) - \delta_\iota^{k+1}(0,0) \leq \texttt{Tol}$ **then**

Define $P'(n_T) = \sum_{m_T\in\mathcal{N}_T} \pi^{k+1}(m_T, n_T)$ for all $n_T \in \mathcal{N}'_T$ then $P'(n) = \sum_{j\in n+} P'(j)$ for all $n \in \mathcal{N}'_t, t \neq T$

Set $\mathrm{ND}_\iota(\mathbf{P}, \mathbf{P}') \leftarrow \delta_\iota^{k+1}(0,0)$

Stop and return with the reduced tree $\mathbf{P}' = (\Xi^{T+1}, \mathcal{F}', P')$ and nested distance $\mathrm{ND}_\iota(\mathbf{P}, \mathbf{P}')$

end if

end for

Some comments on Algorithm 3 are in order.

Initialization

The probability $P^{'}$ and the scenario values $\{\xi'(n) \in \Xi : n \in \mathcal{N}'\}$ will be updated by the algorithm but the tree structure is fixed.

We will provide some initialization methods in Section 4.3.

Note that line 2 is a straightforward operation since the initial transport plan only needs to respect the marginal constraints, therefore $\pi^0(i,j) = \frac{P(i|m)}{|n+|}$ for all $m, n, i \in m+, j \in n+$ is a sufficient initialization.

Quantizer Optimizations

Algorithm 3 only considers the Euclidean case ($\iota = 2$) in line 4, but if $\iota \neq 2$, the quantizer optimization step boils down to a gradient descent (see [1] for details) and the algorithm is still applicable although slower.

Hyperparameters

IBP relies on the use of a parameter $\lambda > 0$ chosen by the user [25, 32]. Such parameter has an impact on the result's accuracy. The MAM algorithm also requires setting a parameter, but it only impacts the convergence speed and can be determined with a sensitivity analysis [26].

Parallelization

MAM is a parallelizable (and randomizable) algorithm, such a feature can also be leveraged in Algorithm 3. Line 8 can also be treated in a parallel manner.

Stopping criteria

The given stopping criteria is a heuristic: the algorithm terminates when the improvement of the nested distance between the two trees is below a certain level of tolerance `Tol`.

Convergence

The algorithm leads to an improvement in each iteration. It should be kept in mind that the above algorithm is nothing but a heuristic, as it is a block-coordinate scheme seeking to minimize the nested distance by alternating minimization over scenarios and then with respect to probabilities.

4 Applications

This section illustrates the performance of Algorithm 3 and its behaviour when using different solvers to compute Wasserstein barycenters (i.e. solutions of (RP) and (WB)). All the following applications employ Algorithm 3 in the Euclidean case, though similar conclusions can be extended to other values of ι. If a standard LP solver is employed, then Algorithm 3 boils down to the setting of [1]. We ensure that the considered solvers attain the same level of precision when tackling the Wasserstein barycenter problems at

Step 2 of Algorithm 3. Numerical experiments were conducted using 20 cores (*Intel(R) Xeon(R) Gold 5120 CPU*) and *Python 3.9*, using a *MPI* based parallelization when possible. The test problems and solvers' codes are available for download at the link https://github.com/dan-mim/Nested_tree_reduction.

4.1 Impact of the Tree Size

In what follows, we consider scenarios trees composed by 4 to 8 stages and 5 to 6 children per nodes, offering numbers of scenarios and spanning from hundreds to ten thousands nodes (see Table 1 for details). The quantizers of both trees are set randomly within the range $[-10, 10]$. In our preliminary tests, the reduced tree is always binary.

We consider the following variants of the algorithm, by varying the solver used to compute the Wasserstein Barycenter in Step 2 of Algorithm 3:

- *LP*: Algorithm 3 employing the LP solver HiGHS;
- *MAM*: Algorithm 3 employing the *Method of Averaged Marginals* [26];
- *IBP*: Algorithm 3 employing IBP, inspired from the code of G. Peyré [33]. This algorithm relies on the tune of an hyperparameter that has been preset to guarantee its best efficiency and convergence.

All variants use $\texttt{Tol} = 0.1$ as a tolerance for the stopping test. It has been empirically verified that results do not improve significantly if we decrease this tolerance further.

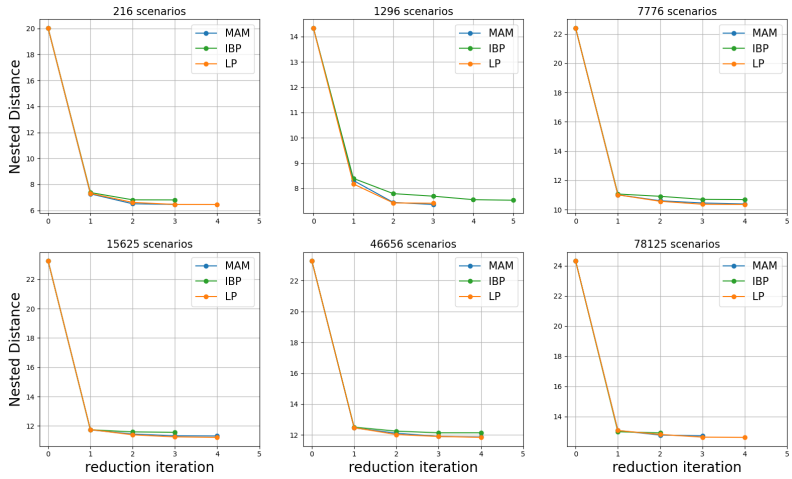


Fig. 4: Evolution of the Nested Distance along the reduction iterations for different initial tree sizes. The final reduced tree has always two children per node.

Figure 4 shows that the exact ND between the original tree and the approximate one, iteratively decreases, no matter the variant of Algorithm 3. All variants are initialized with the same tree, generated randomly (probabilities and scenarios). Note

that the initial ND is always at least halved after the reduction. This emphasizes how important is the use of a reduction method. Within this scale, one can see that every variant converges to approximately the same precision, although not necessarily to the same reduced scenario tree (because different solvers compute different optimal transportation plans, impacting the construction of scenarios composing the reduced tree).

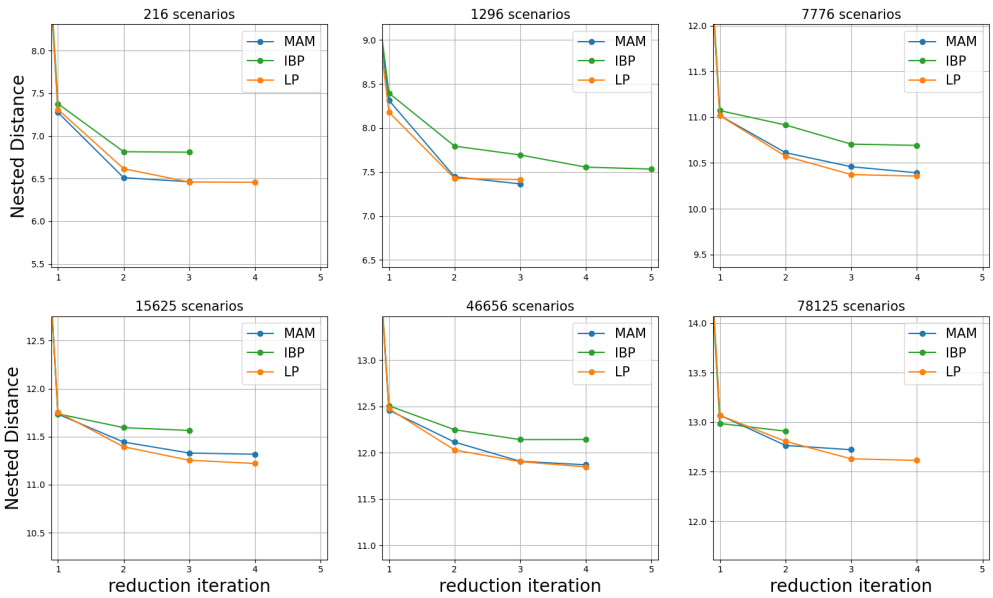


Fig. 5: Evolution of the Nested Distance along the reduction iterations for different initial tree sizes with a zoom.

Even though the ND decreases with all variants, it seems faster (in terms of number of iterations) when using LP or MAM. Figure 5 shows that the IBP algorithm tends to reach a less precise plateau. This is due to the core of the IBP method, which is an inexact algorithm. MAM being an exact algorithm for solving (WB), it naturally follows the lead of LP. The slight differences between the variants LP and MAM can be explained by the fact that the general problem is non-convex and different optimal transportation plans computed by different solvers can lead to different optimization paths that result in different reduced scenario trees. Therefore, depending on the employed solvers and the initialization, the approximated trees could be different while still having close ND with the original tree.

Table 1 shows that for small initial trees, up to 7776 scenarios, the LP variant is very efficient: it provides the lowest ND solution in the shortest time. But from 15625 scenarios and more, IBP is faster. As the number of scenarios increases, the relative performances of MAM get better and, in our case with more than 46656 scenarios, MAM is eventually twice faster than LP while reaching the same precision as depicted in Figure 4. As shown in the last graph, with even more nodes and scenarios (78125

Scenarios	Nodes	LP	IBP	MAM	MAM 4 processors
216	**259**	0.17	0.49	2.21	0.56
1296	**1555**	1.54	14.83	18.23	6.28
7776	**9331**	74.25	161.19	344.83	124.44
15625	**19531**	487.58	323.76	816.46	341.62
46656	**55987**	4905	2136	2541	1256
78125	**97656**	13797	4334	3458	1635

Table 1: Total time (in seconds) per method for the studied trees.

and 97656, respectively), MAM is the fastest variant. Note that IBP is a very robust method: not only is the precision reached more than reasonable, according to Figure 4, but the total time of execution is always in the ballpark of the fastest execution time of all algorithms. Leveraging that MAM is parallelizable, we ran results using 4 processors. We witnessed that in this configuration, the variant MAM is the most advantageous one when the initial tree has more than 20000 scenarios by far.

4.2 Impact of the tree structure

We observed when using the different approaches to tackle real-life examples that the *structure* of the initial large tree has an impact on the convergence speed. Real-life trees do not span homogeneously from the root to the last stage, and when heterogeneity occurs, switching from one method to another can make the global reduction algorithm significantly faster.

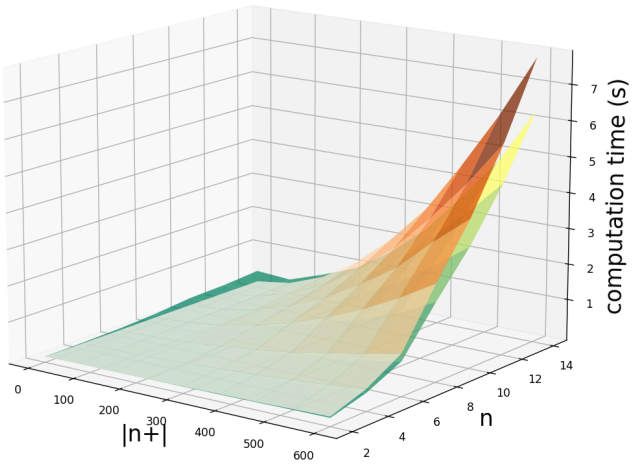


Fig. 6: Influence of the tree structure on the computation time of a stage, depending on the method in use: MAM in green and LP in orange.

Figure 6 illustrates the tree structure influence on the computation time of a stage, depending on the algorithm used. The original large tree is built as follows: from the root, two nodes are spanning (stage $t = 2$), then stage $t = 3$ counts n nodes. From

this configuration, we made the number $|n+|$ of children from these n nodes vary from 1 to 600 children (stage $t = T = 4$). To put it differently, we built a tree with n subtrees at *stage* 3 having dimension $|n+|$. We reduce this tree into a binary one and evaluate *stage* 3 reduction time[1]. We recall that the most expensive step of the algorithm consists in solving the greatest number of (WB) problems, thus stage 3.

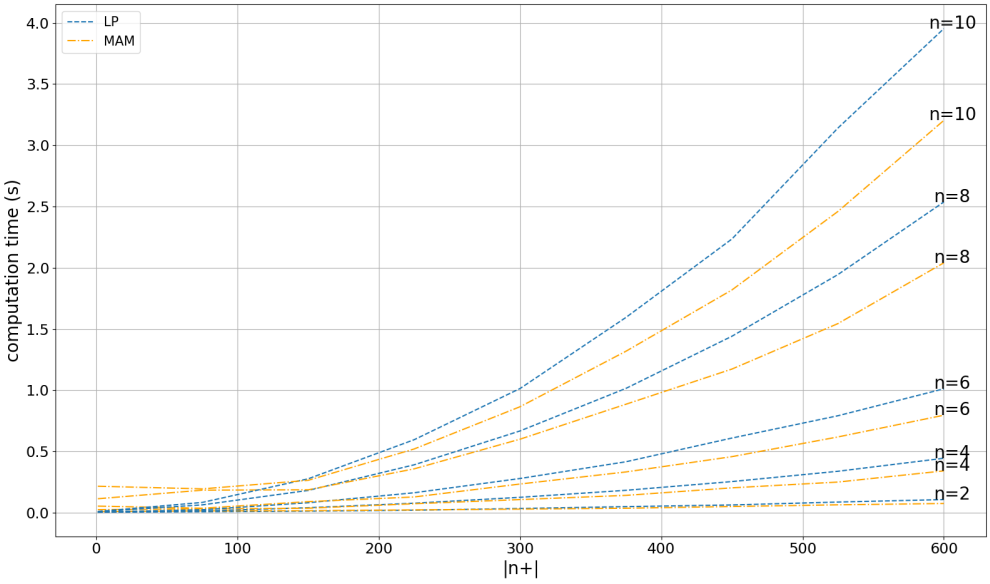


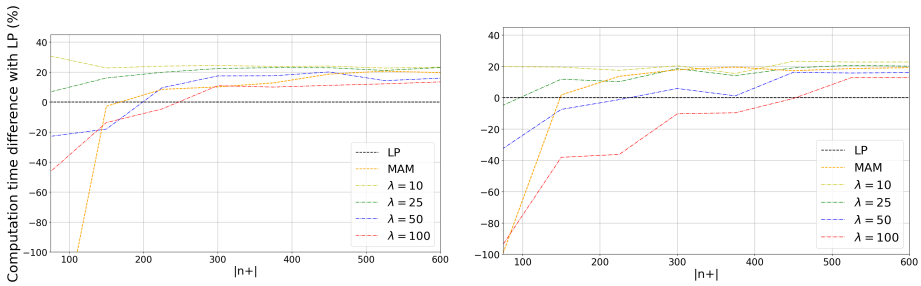
Fig. 7: Influence of the tree structure on the computation time for small n.

For any number n of subtrees, the LP solver is always better when $|n+|$ is low, but when increasing $|n+|$ MAM solver ends up being faster because it treats better larger problems. This discrepancy gains momentum when the number of subtrees rises. It is observed that the treshold is reached even faster when n is large. Therefore, at a stage with $n > 10$ (and $|n+| > 1$) where the reduction can take more than $4s$ we would advise to always use MAM. But, as illustrated in Figure 7, for smaller n we would have a closer look, and use MAM only if $|n+| > 150$, otherwise keep the LP method. Figure 6 and Figure 7 show that choosing the adequate method to tackle the reduction can speed up the computation time from 20% to 35%. In practice, it can be observed that this repartition ends up using half MAM (mostly for the deep stages where $t \gg 0$) and half LP (mostly for the stages close to the root and the heterogeneity in the structure) to reduce a real-life initial tree.

The IBP method is challenging to use in practice within the tree reduction algorithm because of the complexity involved in tuning the hyperparameter λ, which ideally needs to be carefully chosen for each barycenter computation to achieve acceptable precision. Fine-tuning is necessary to control its accuracy. If a broadly set

[1] The computation time is evaluated as the mean of the five iterations along the reduction - at stage 3

hyperparameter λ is used, the algorithm may either encounter double-precision overflow errors at certain stages or compute a solution that significantly deviates from the exact optimization, without any guarantee or control over the resulting precision.



(a) $n = 8$ (b) $n = 12$

Fig. 8: Speed comparison with IBP for different λ: A positive time difference means the method is faster than LP. Each curve is obtained by averaging the ND accuracy over $n \in \{2, 4, 6, 8, 10, 12, 14, 16\}$.

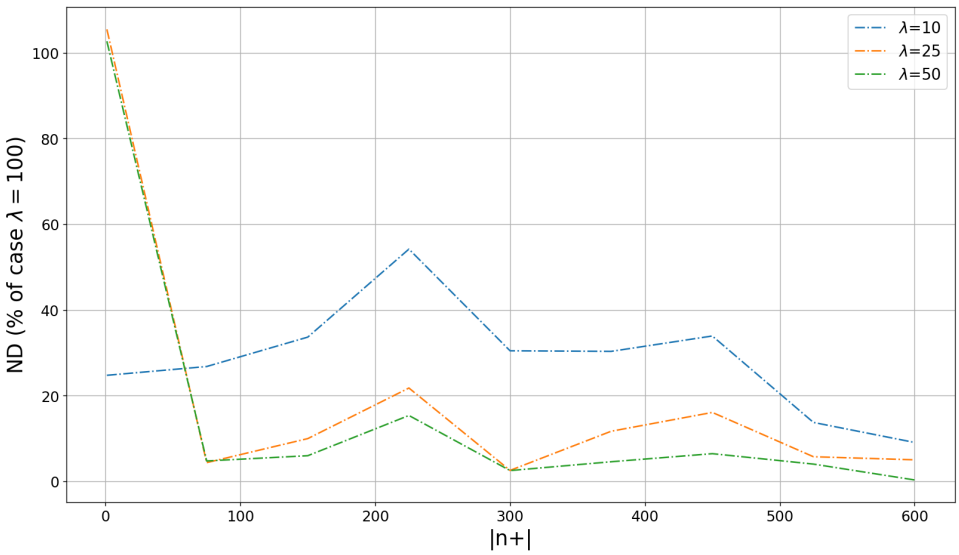


Fig. 9: Average influence of λ in the precision. Each curve is obtained by averaging the ND accuracy over $n \in \{2, 4, 6, 8, 10, 12, 14, 16\}$.

Figure 8 presents a study on the influence of a broad λ used for IBP in the tree reduction algorithm for the same datasets as earlier. It shows that the use of IBP enables the reduction tree algorithm to be fast for both small and large n and $n+$,

and stresses that the smaller the λ the faster the reduction. But Figure 9 underlines that this speed comes with a cost since the smaller the λ and the less accurate the computation of the barycenter - the greater the ND. Figure 9 has been obtained by averaging the ND accuracy over n [2], where the case $\lambda = 100$ is taken as reference.

4.3 Impact of the Initialization

In this section, we incorporate the reduction tree algorithm (utilizing either MAM or LP, depending on the results from Section 4.2) into a stochastic optimization pipeline. Data were collected from an industrial site in Solaize, France, where battery production and local consumption have been monitored over several years. Using this data, we generated scenarios for battery consumption and production throughout a single day, divided into 48 time steps, employing the methodology outlined in [34]. This approach allows us to create 100 $2D$ scenarios, each consisting of 48 stages with scenario values spanning from 0 to 25kW for the consumption and 0 to 40kW for the production. These scenarios can be modeled as a large tree. We reduce this tree using various initial filtrations to construct a smaller, approximate tree. To built these filtrations we leverage two scenario selection algorithms:

- *Kmeans method*, starting from 100 scenarios it creates 25 clusters using the Euclidean norm, and then computes the 25 corresponding barycenters;
- *Fast Forward Selection (FFS) method*, introduced by Heitsch and Römisch in [35]. The method iteratively selects scenarios that minimize the Wasserstein distance to the remaining scenarios. At each step, the scenario that best approximates the distribution is added to the reduced set until the desired number of 25 scenarios is reached, ensuring an efficient yet effective reduction.

From these reduced number of scenarios we generate the associate trees. They will initialize Algorithm 3. We make the experience multiple times by generating several sets of scenarios.

Table 2 compares the ND to the original large tree, both before and after tree reduction, highlighting the importance of the initial filtration. Note that the ND is consistently reduced after the tree reduction algorithm, as expected. When FFS is used to generate the initial filtration, the reduced approximated tree remains consistently closer to the original tree in terms of nested distance, even if the ND between the original tree and the FFS filtration is larger than that between the original tree and the K-means filtration at initialization. This is because FFS is a specialized algorithm designed to minimize the Wasserstein distance between the selected scenarios and the original ones, thereby preserving the maximum amount of information from the initial processes.

[2] $n \in \{2, 4, 6, 8, 10, 12, 14, 16\}$.

Scenario set	Filtrations	initial ND	reduced ND
1	**Kmeans**	2757	1219
	FFS	1384	658
2	**Kmeans**	1699	1092
	FFS	1936	896
3	**Kmeans**	1653	832
	FFS	1499	716
4	**Kmeans**	1858	963
	FFS	1161	566
5	**Kmeans**	1968	1054
	FFS	917	540

Table 2: Comparison of the ND to the original tree before and after tree reduction using different initialization techniques.

5 Compliance with Ethical Standards

Funding: None.
Conflict of Interest: None.
Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

References

[1] Kovacevic, R.M., Pichler, A.: Tree approximation for discrete time stochastic processes: a process distance approach. Annals of operations research **235**(1), 395–421 (2015)

[2] Dantzig, G.B.: Linear programming under uncertainty. Management science **1**(3-4), 197–206 (1955)

[3] Edirisinghe, N.: Multiperiod portfolio optimization with terminal liability: Bounds for the convex case. Computational Optimization and Applications **32**, 29–59 (2005)

[4] Brodt, A.I.: Min-mad life: A multi-period optimization model for life insurance company investment decisions. insurance: Mathematics and Economics **2**(2), 91–102 (1983)

[5] Paulo, H., Cardoso-Grilo, T., Relvas, S., Barbosa-Póvoa, A.P.: Designing integrated biorefineries supply chain: combining stochastic programming models with scenario reduction methods **40**, 901–906 (2017)

[6] Carpentier, P., Chancelier, J.-P., Cohen, G., Lara, M., Girardeau, P.: Dynamic consistency for stochastic optimal control problems. Annals of Operations

Research **200**, 247–263 (2012)

[7] Analui, B., Pflug, G.C.: On distributionally robust multiperiod stochastic optimization. Computational Management Science **11**, 197–220 (2014)

[8] Beltrán, F., de Oliveira, W., Finardi, E.C.: Application of scenario tree reduction via quadratic process to medium-term hydrothermal scheduling problem. IEEE Transactions on Power Systems **32**(6), 4351–4361 (2017)

[9] Dupačová, J., Gröwe-Kuska, N., Römisch, W.: Scenario reduction in stochastic programming. Mathematical programming **95**, 493–511 (2003)

[10] de Oliveira, W.L., Sagastizábal, C., Penna, D.D.J., Maceira, M.E.P., Damázio, J.M.: Optimal scenario tree reduction for stochastic streamflows in power generation planning problems. Optimisation Methods & Software **25**(6), 917–936 (2010)

[11] Qian, X., Tang, Q.: Scenario reduction method based on output performance for condition-based maintenance optimization. Mechanics **23**(5), 743–749 (2017)

[12] Beltrán, F., Finardi, E.C., de Oliveira, W.: Two-stage and multi-stage decompositions for the medium-term hydrothermal scheduling problem: A computational comparison of solution techniques. International Journal of Electrical Power & Energy Systems **127**, 106659 (2021) https://doi.org/10.1016/j.ijepes.2020.106659

[13] Li, Z., Floudas, C.A.: Optimal scenario reduction framework based on distance of uncertainty distribution and output performance: I. single reduction via mixed integer linear optimization. Computers & Chemical Engineering **70**, 50–66 (2014)

[14] Li, Z., Li, Z.: Linear programming-based scenario reduction using transportation distance. Computers & Chemical Engineering **88**, 50–58 (2016)

[15] Kammammettu, S., Li, Z.: Scenario reduction and scenario tree generation for stochastic programming using sinkhorn distance. Computers & Chemical Engineering **170**, 108122 (2023)

[16] Sinkhorn, R., Knopp, P.: Concerning nonnegative matrices and doubly stochastic matrices. Pacific Journal of Mathematics **21**(2), 343–348 (1967)

[17] Heitsch, H., Römisch, W., Strugarek, C.: Stability of multistage stochastic programs. SIAM Journal on Optimization **17**(2), 511–525 (2006)

[18] Heitsch, H., Römisch, W.: Scenario tree modeling for multistage stochastic programs. Mathematical Programming **118**, 371–406 (2009)

[19] Chen, Z., Yan, Z.: Scenario tree reduction methods through clustering nodes. Computers & Chemical Engineering **109**, 96–111 (2018)

[20] Xu, D., Chen, Z., Yang, L.: Scenario tree generation approaches using k-means and lp moment matching methods. Journal of Computational and Applied Mathematics **236**(17), 4561–4579 (2012)

[21] Horejšová, M., Vitali, S., Kopa, M., Moriggia, V.: Evaluation of scenario reduction algorithms with nested distance. Computational Management Science **17**(2), 241–275 (2020)

[22] Pflug, G.C.: Version-independence and nested distributions in multistage stochastic optimization. SIAM Journal on Optimization **20**(3), 1406–1420 (2010)

[23] Pflug, G.C., Pichler, A.: A distance for multistage stochastic optimization models. SIAM Journal on Optimization **22**(1), 1–23 (2012)

[24] Pichler, A., Weinhardt, M.: The nested sinkhorn divergence to learn the nested distance. Computational Management Science **19**(2), 269–293 (2022)

[25] Benamou, J.-D., Carlier, G., Cuturi, M., Nenna, L., Peyré, G.: Iterative bregman projections for regularized transportation problems. SIAM Journal on Scientific Computing **37**(2), 1111–1138 (2015)

[26] Mimouni, D.W., Malisani, P., Zhu, J., de Oliveira, W.: Computing wasserstein barycenters via operator splitting: The method of averaged marginals. SIAM Journal on Mathematics of Data Science **6**(4), 1000–1026 (2024)

[27] Cuturi, M., Doucet, A.: Fast computation of wasserstein barycenters. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 32, pp. 685–693. PMLR, Bejing, China (2014)

[28] Ye, J., Wu, P., Wang, J.Z., Li, J.: Fast discrete distribution clustering using wasserstein barycenter with sparse support. IEEE Transactions on Signal Processing **65**, 2317–2332 (2017) https://doi.org/10.1109/TSP.2017.2659647

[29] Bauschke, H.H., Combettes, P.L.: Convex Analysis and Monotone Operator Theory in Hilbert Spaces. CMS Books in Mathematics. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-48311-5

[30] Douglas, J., Rachford, H.H.: On the numerical solution of heat conduction problems in two and three space variables. Transactions of the American mathematical Society **82**(2), 421–439 (1956)

[31] Condat, L.: Fast projection onto the simplex and the $\boldsymbol{l_1}$ball. Mathematical Programming **158**(1), 575–585 (2016)

[32] Cuturi, M., Doucet, A.: Fast computation of wasserstein barycenters. International Conference on Machine Learning **32**(2), 685–693 (2014) https://doi.org/

10.1137/140951758

[33] Peyré, G.: BregmanOT. GitHub (2014). https://github.com/gpeyre/2014-SISC-BregmanOT

[34] Amabile, L., Bresch-Pietri, D., El Hajje, G., Labbé, S., Petit, N.: Optimizing the self-consumption of residential photovoltaic energy and quantification of the impact of production forecast uncertainties. Advances in Applied Energy **2**, 100020 (2021)

[35] Heitsch, H., Römisch, W.: Scenario reduction algorithms in stochastic programming. Computational Optimization and Applications **24**, 187–206 (2003)