A neurodynamic approach to solve rectangular programs with joint probabilistic constraints

Siham Tassouli, Abdel Lisser

A neurodynamic approach to solve rectangular programs with joint probabilistic constraints

Siham Tassouli*[a], Abdel Lisser[a]

[a]*Laboratoire des Signaux et Systèmes (L2S), CentraleSupélec, Université Paris Saclay, 3, rue Joliot Curie, 91192 Gif sur Yvette cedex, France*

Abstract

This paper considers a noncovex geometric problem with two-sided joint probabilistic inequalities constraints, namely rectangular constraints. We transform the stochastic problem into a deterministic one. Further, we use a logarithmic transformation combined with the arithmetic-geometric mean inequality to obtain a biconvex problem. Based on the biconvex structure of the obtained program and the correspondent partial KKT system, we propose a dynamical neural network to solve the initial rectangular problem. The main feature of our framework is to propose a converging method to solve rectangular joint chance-constrained optimization problems without the use of any convex approximation unlike the state-of-the-art solving methods. To verify the performances of our approach, we conducted several tests on a minimum transport cost problem and a shape optimization problem.

Keywords: Biconvex optimization, Joint probabilistic constraints, Rectangular programming, Dynamical neural network, Lyapunov theory, Partial KKT system

2010 MSC: 00-01, 99-00

1. Introduction

Chance constrained programming was first introduced in by Charnes & Cooper (1959) to solve optimization problems under various uncertainties. Since then, many studies introducing chance constraints have been done. In this paper, we are interested in two-sided joint geometric chance constraints called rectangular chance constraints.

Perlumutter (1967) introduced geometric programming in 1967. Over the last few decades, geometric programming has been used in several fields, e.g., aircraft design problems (Hoburg & Abbeel, 2012), communication systems (Chiang (2005)), power control (Chiang et al. (2007)), digital circuit optimization (Boyd et al. Boyd et al. (2005)), biochemical systems (Liu et al. (2014)), operational amplifiers design (Vanderhaegen & Brodersen (2004)), metal cutting optimization (Dupačová (2010)). To solve geometric programs with joint probabilistic constraints, Liu et al. (2016) approximate the problem using piecewise linear functions, which leads to a lower bound. In order to find an upper bound, they propose a sequential convex optimization algorithm. Liu et al. (2020) give an asymptotically tight approximation for rectangular programs with joint probabilistic constraints based on

*Corresponding author.

Email addresses: siham.tassouli@centralesupelec.fr (Siham Tassouli*), abdel.lisser@centralesupelec.fr (Abdel Lisser)

variable transformation and linear approximation methods. Xu (2014) gives a global optimization approach to solve signomial geometric programs using some convex transformation strategies.

In this paper, we use a dynamical neural network to solve a joint chance-constrained rectangular problem. Different methods using dynamical systems were used to solve optimization problems. Faybusovich (1991) proposes dynamical systems to solve optimization problems with linear constraints. Schropp & Singer (2000) use differential-algebraic equations to solve general smooth minimization problems. Inspired by quantum mechanics, Aluffi-Pentini et al. (1985) study the global minimizers by following the paths of a system of stochastic differential equations. Effati & Nazemi (2006) propose two recurrent neural network models for solving linear and quadratic programming problems.

In this paper, we study the following stochastic rectangular programming problem:

$$\min_{t\in\mathbb{R}^M_{++}} \quad \mathbb{E}\left[\sum_{i\in I_0} c_i \prod_{j=1}^{M} t_j^{a_{ij}}\right], \tag{1}$$

$$\text{s.t.} \quad \mathbb{P}\left(\alpha_k \leq \sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq \beta_k, k = 1, ..., K\right) \geq 1-\epsilon, \tag{2}$$

where $c_i, i \in I_k$ are uncorrelated normally distributed random variables , i.e., $c_i \sim \mathcal{N}(\bar{c}_i, \sigma_i^2)$, $\bar{c}_i \geq 0$,$0 < \alpha_k < \beta_k$ The coefficients $a_{ij}, i \in I_k, j = 1, ..., M$ are deterministic, and $1-\epsilon$ is a given probability level.

Liu et al. (2020) propose convex approximations based on the variable transformation to solve problem (1)-(2) with an elliptical distribution. They give upper and lower bounds for the optimal solution.

Main contributions

This paper studies a joint chance-constrained rectangular problem and proposes a recurrent neural network to solve it. The main contributions of this paper are listed as follows.

(i) We reformulate joint constrained rectangular problems as a nonlinear biconvex deterministic equivalent problem. To the best of our knowledge, this is the first time that rectangular problems with joint probabilistic constraints are reformulated using neurodynamic system.

(ii) Generally to solve stochastic problems with joint constraints, convex approximations of the biconvex functions and the stochastic gradient methods are used. In our paper, we converge directly to a good near-optimal solution of the studied problem.

(iii) The numerical experiments part shows the robustness of our neural network.

The rest of the paper is organized as follows. In Section 1, a deterministic biconvex equivalent problem is obtained using the arithmetic-geometric mean inequality combined with a logarithmic transformation is given. In Section 2, we study the optimality conditions of the obtained biconvex problem. In Section 3, we propose a dynamical neural network to solve problem (1)-(2) based on the partial KKT system, and we study the convergence and the stability of the neural network. Finally, we dedicate Section 4 to study the numerical performances of our neural network by solving a shape optimization problem.

2. Deterministic biconvex equivalent problem

Problem (1)-(2) is a joint constrained program. To transform the joint constraints into deterministic ones, we assume that the row vector constraints are mutually independent. Then, we introduce auxiliary variables y_k, $k = 1, .., K$ and we rewrite the constraint (2) equivalently as

$$\mathbb{P}\left(\alpha_k \leq \sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq \beta_k\right) \geq y_k, \quad k = 1, .., K, \tag{3}$$

$$\prod_{k=1}^{K} y_k \geq 1-\epsilon, 0 \leq y_k \leq 1, \quad k = 1, .., K. \tag{4}$$

The rectangular constraints (3) are equivalent to

$$\mathbb{P}\left(\sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \geq \alpha_k\right) + \mathbb{P}\left(\sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq \beta_k\right) - 1 \geq y_k, k = 1, .., K. \tag{5}$$

Then, we introduce two additional K-dimensional auxiliary variables $z^+, z^- \in \mathbb{R}_+^K$ Liu et al. (2020) such that (5) is equivalent to

$$\mathbb{P}\left(\sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \geq \alpha_k\right) \geq z_k^+, k = 1, .., K, \tag{6}$$

$$\mathbb{P}\left(\sum_{i\in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq \beta_k\right) \geq z_k^-, k = 1, .., K, \tag{7}$$

$$z_k^+ + z_k^- - 1 \geq y_k, 0 \leq z_k^-; z_k^+ \leq 1, k = 1, .., K, \tag{8}$$

$$\prod_{k=1}^{K} y_k \geq 1-\epsilon, 0 \leq y_k \leq 1, k = 1, .., K. \tag{9}$$

Deterministic reformulations of constraints (6) and (7) are given as follows. Cheng & Lisser (2012)

$$-\sum_{i\in I_k} \bar{c}_i \prod_{j=1}^{M} t_j^{a_{ij}} + \phi^{-1}(z_k^+)\sqrt{\sum_{i\in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq -\alpha_k, \tag{10}$$

$$\sum_{i\in I_k} \bar{c}_i \prod_{j=1}^{M} t_j^{a_{ij}} + \phi^{-1}(z_k^-)\sqrt{\sum_{i\in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq \beta_k, \tag{11}$$

Constraint (10) can be reformulated as follows

$$\phi^{-1}(z_k^+)^2 \sum_{i\in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}} \leq \left(\sum_{i\in I_k} \bar{c}_i \prod_{j=1}^{M} t_j^{a_{ij}} - \alpha_k\right)^2. \tag{12}$$

We write (12) equivalently as

$$2\alpha_k \sum_{i\in I_k} \bar{c}_i \prod_{j=1}^{M} t_j^{a_{ij}} - \sum_{i\in I_k}\sum_{p\in I_k} \bar{c}_i\bar{c}_p \prod_{j=1}^{M} t_j^{(a_{ij}+a_{pj})} + \phi^{-1}(z_k^+)^2 \sum_{i\in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}} \leq \alpha_k^2,$$

which can be reformulated as

$$\frac{2\alpha_k \sum\limits_{i\in I_k} \bar{c}_i \prod\limits_{j=1}^{M} t_j^{a_{ij}} + \phi^{-1}(z_k^+)^2 \sum\limits_{i\in I_k} \sigma_i^2 \prod\limits_{j=1}^{M} t_j^{2a_{ij}}}{\sum\limits_{i\in I_k}\sum\limits_{p\in I_k} \bar{c}_i\bar{c}_p \prod\limits_{j=1}^{M} t_j^{(a_{ij}+a_{pj})} + \alpha_k^2} \leq 1. \tag{13}$$

We propose to approximate the denominator of constraint (13) with a monomial function by applying the arithmetic-geometric mean inequality

$$\sum_{i\in I_k}\sum_{p\in I_k}\bar{c}_i\bar{c}_p\prod_{j=1}^{M}t_j^{(a_{ij}+a_{pj})}+\alpha_k^2\geq\prod_{i\in I_k}\prod_{p\in I_k}\left(\frac{\bar{c}_i\bar{c}_p\prod_{j=1}^{M}t_j^{(a_{ij}+a_{pj})}}{\delta_{ip}}\right)^{\delta_{ip}}\left(\frac{\alpha_k^2}{\delta_0}\right)^{\delta_0},$$

where δ_0 and δ_{ip} are nonnegative parameter $\forall i\in I_k$ and $\delta_0+\sum_{i,p\in I_k}\delta_{ip}=1$

We write then problem (1)-(2) as

$$\min_{t\in\mathbb{R}^M_{++}}\quad\sum_{i\in I_0}\bar{c}_i\prod_{j=1}^{M}t_j^{a_{ij}},\tag{14}$$

$$\text{s.t.}\quad\left(2\alpha_k\sum_{i\in I_k}\bar{c}_i\prod_{j=1}^{M}t_j^{a_{ij}}+\phi^{-1}(z_k^+)^2\sum_{i\in I_k}\sigma_i^2\prod_{j=1}^{M}t_j^{2a_{ij}}\right)$$

$$\prod_{i\in I_k}\prod_{p\in I_k}\left(\frac{\bar{c}_i\bar{c}_p\prod_{j=1}^{M}t_j^{(a_{ij}+a_{pj})}}{\delta_{ip}}\right)^{-\delta_{ip}}\left(\frac{\alpha_k^2}{\delta_0}\right)^{-\delta_0}\leq 1,k=1,...,K,\tag{15}$$

$$\sum_{i\in I_k}\bar{c}_i\prod_{j=1}^{M}t_j^{a_{ij}}\phi^{-1}(z_k^-)\sqrt{\sum_{i\in I_k}\sigma_i^2\sum_{j=1}^{M}t_j^{2a_{ij}}}-\beta_k\leq 0,k=1,...,K,\tag{16}$$

$$z_k^++z_k^--1\geq y_k,0\leq z_k^-;z_k^+\leq 1,k=1,..,K,\tag{17}$$

$$\prod_{k=1}^{K}y_k\geq 1-\epsilon,0\leq y_k\leq 1,k=1,..,K.\tag{18}$$

We apply a logarithmic transformation of the problem by introducing $r_k=log(t_k)$, $x_k=log(y_k)$, $k=1,..,K$. We obtain the following biconvex equivalent problem

$$\min_{r\in\mathbb{R}^M} \quad \sum_{i\in I_0}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}, \tag{19}$$

$$\text{s.t.} \quad \frac{1}{\prod_{i\in I_k}\prod_{p\in I_k}\delta_{ip}}\left(2\alpha_k\sum_{i\in I_k}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}+\phi^{-1}(z_k^+)^2\sum_{i\in I_k}\sigma_i^2\exp\left\{\sum_{j=1}^{M}2a_{ij}r_j\right\}\right)$$

$$\exp\left\{\sum_{i\in I_k}\sum_{p\in I_k}-\delta_{ip}(\ln(\bar{c}_i)+\ln(\bar{c}_p))\sum_{j=1}^{M}(a_{ij}+a_{pj})r_j\right\}\left(\frac{\alpha_k^2}{\delta_0}\right)^{-\delta_0}\leq 1, k=1,...,K, \tag{20}$$

$$\sum_{i\in I_k}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}+$$

$$\sqrt{\sum_{i\in I_k}\sigma_i^2\exp\left\{\sum_{j=1}^{M}2a_{ij}r_j+\log(\phi^{-1}(z_k^-)^2)\right\}}-\beta_k\leq 0, k=1,...,K, \tag{21}$$

$$\log(1-\epsilon)-\sum_{k=1}^{K}x_k\leq 0,\ x_k\leq 0, \qquad k=1,...,K,$$

$$\exp(x_k)-z_k^+-z_k^-+1\leq 0, k=1,...,K, \tag{22}$$

$$z_k^+-1\leq 0, k=1,...,K, \tag{23}$$

$$z_k^--1\leq 0, k=1,...,K, \tag{24}$$

$$-z_k^+\leq 0, k=1,...,K, \tag{25}$$

$$-z_k^-\leq 0, k=1,...,K. \tag{26}$$

We define $z=(z^+,z^-)^T$, $f(r)=\sum_{i\in I_0}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}$, $h(x)=(\log(1-\epsilon)-\sum_{k=1}^{K}x_k, x_1,\ldots,x_K)^T$, $l(x,z)=(\exp(x_1)-z_1^+-z_1^-+1,...,\exp(x_K)-z_K^+-z_K^-+1)^T$,

$w(z)=({z_1}^+-1,\ldots,{z_K}^+-1,{z_1}^--1,\ldots,{z_K}^--1,-{z_1}^+,\ldots,-{z_K}^+,-{z_1}^-,\ldots,-{z_K}^-)^T$ and

$$g(r,z)=\begin{pmatrix}
\left(2\alpha_1\sum_{i\in I_1}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}+\phi^{-1}(z_1^+)^2\sum_{i\in I_1}\sigma_i^2\exp\left\{\sum_{j=1}^{M}2a_{ij}r_j\right\}\right)\\
\prod_{i\in I_1}\prod_{p\in I_1}\left(\frac{\bar{c}_i\bar{c}_p\exp\left\{\sum_{j=1}^{M}(a_{ij}+a_{pj})r_j\right\}}{\delta_{ip}}\right)^{-\delta_{ip}}\left(\frac{\alpha_1^2}{\delta_0}\right)^{-\delta_0}-1\\
\vdots\\
\left(2\alpha_k\sum_{i\in I_k}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}+\phi^{-1}(z_k^+)^2\sum_{i\in I_k}\sigma_i^2\exp\left\{\sum_{j=1}^{M}2a_{ij}r_j\right\}\right)\\
\prod_{i\in I_k}\prod_{p\in I_k}\left(\frac{\bar{c}_i\bar{c}_p\exp\left\{\sum_{j=1}^{M}(a_{ij}+a_{pj})r_j\right\}}{\delta_{ip}}\right)^{-\delta_{ip}}\left(\frac{\alpha_k^2}{\delta_0}\right)^{-\delta_0}-1\\
\sum_{i\in I_1}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}+\phi^{-1}(z_1^-)\sqrt{\sum_{i\in I_1}\sigma_i^2\exp\left\{\sum_{j=1}^{M}2a_{ij}r_j+\right\}}-\beta_1\\
\vdots\\
\sum_{i\in I_K}\bar{c}_i\exp\left\{\sum_{j=1}^{M}a_{ij}r_j\right\}+\phi^{-1}(z_K^-)\sqrt{\sum_{i\in I_K}\sigma_i^2\exp\left\{\sum_{j=1}^{M}2a_{ij}r_j+\right\}}-\beta_K
\end{pmatrix},$$ we write (19)-

(26) as

$$\begin{aligned} \min_{r\in\mathbb{R}^M} \quad & f(r), \\ \text{s.t.} \quad & g(r,z) \leq 0, \\ & h(x) \leq 0, \\ & l(x,z) \leq 0, \\ & w(z) \leq 0. \end{aligned} \tag{27}$$

3. Optimality conditions

Now, we study the optimality conditions for problem (27). Since the problem is biconvex, we do not talk about a KKT system but rather a partial KKT system (Jiang et al., 2021).

Definition 1. Let $(r^*, z^*, x^*) \in \mathbb{R}^m \times \mathbb{R}^{2K} \times \mathbb{R}^K$, if there exists $\mu^{(1)}$, $\mu^{(2)}$, $\lambda^{(1)}$, $\lambda^{(2)}$, γ and ω such that

$$\nabla f(r^*) + {\mu^{(1)}}^T \nabla_r g(r^*, z^*) = 0, \tag{28}$$

$$\mu^{(1)} \geq 0, {\mu^{(1)}}^T g(r^*, z^*) = 0, \tag{29}$$

$${\mu^{(2)}}^T \nabla_z g(r^*, z^*) + {\lambda^{(1)}}^T \nabla_z l(x^*, z^*) + \gamma^T \nabla_z w(z^*) = 0, \tag{30}$$

$$\mu^{(2)} \geq 0, {\mu^{(2)}}^T g(r^*, z^*) = 0, \lambda^{(1)} \geq 0, {\lambda^{(1)}}^T l(x^*, z^*), \gamma \geq 0, \gamma^T w(z^*) = 0, \tag{31}$$

$${\lambda^{(2)}}^T \nabla_x l(x^*, z^*) + \omega^T \nabla_x h(x^*) = 0, \tag{32}$$

$$\lambda^{(2)} \geq 0, {\lambda^{(2)}}^T l(x^*, z^*) = 0, \omega \geq 0, \omega^T h(x^*) = 0 \tag{33}$$

Then (r^*, z^*, x^*) is called a partial KKT point of (27).

Remark 1. The vectors $\mu^{(1)}$, $\mu^{(2)}$, $\lambda^{(1)}$, $\lambda^{(2)}$, γ and ω in Definition 1 are equivalent to the Lagrange multipliers in a KKT system.

The following theorem is driven by the equivalence between a partial optimum and a partial KKT point of a biconvex program.

Theorem 2. Let $(r^*, z^*, x^*) \in \mathbb{R}^M \times \mathbb{R}^{2K} \times \mathbb{R}^K$ be a partial solution of (12), with respect to partial Slater constraints qualification (Jiang et al., 2021) at (r^*, x^*). Then (r^*, z^*, x^*) is a KKT point of (27) if and only if the partial KKT system (28)-(33) holds with $\mu^{(1)} = \mu^{(2)}$ and $\lambda^{(1)} = \lambda^{(2)}$. Furthermore, if $\mu^{(1)} = \mu^{(2)}$ and $\lambda^{(1)} = \lambda^{(2)}$ then (r^*, z^*, x^*) is a KKT point of (27).

Remark 3. The main idea of the proof of Theorem 2 can be found in Jiang et al. (2021).

4. A dynamical neural network approach

Based on the partial KKT system (28)-(33) obtained in the previous section, we construct a dynamical neural network system that converges to a partial KKT point of (13). The dynamical

neural network is driven by the following system, where r, z, x, μ, λ, γ, and ω are time-dependent variables

$$\frac{dr}{dt} = -(\nabla f(r) + \nabla_r g(r,z)^T(\mu + g(r,z))_+), \tag{34}$$
$$\frac{dz}{dt} = -(\nabla_z g(r,z)^T(\mu + g(r,z))_+ \nabla_z l(x,z)^T(\lambda + l(x,z))_+ + \nabla_z w(z)^T(\gamma + w(z))_+), \tag{35}$$
$$\frac{dx}{dt} = -(\nabla_x l(x,z)^T(\lambda + l(x,z))_+ \nabla_x h(x)^T(\omega + h(x))_+), \tag{36}$$
$$\frac{d\mu}{dt} = (\mu + g(r,z))_+ - \mu, \tag{37}$$
$$\frac{d\lambda}{dt} = (\lambda + l(x,z))_+ - \lambda, \tag{38}$$
$$\frac{d\gamma}{dt} = (\gamma + w(z))_+ - \gamma, \tag{39}$$
$$\frac{d\omega}{dt} = (\omega + h(x))_+ - \omega. \tag{40}$$

For the sake of simplicity, let $y = (r, z, x, \mu, \lambda, \gamma, \omega)$ we rewrite the dynamical system (34)-(40) equivalently as follows

$$\begin{cases} \frac{dy}{dt} = \Phi_(y) \\ y(t_0) = y_0 \end{cases}.$$

The hardware implementation of the neural network (34)-(40) is provided in Figure 1.

To study the stability and the convergence of the proposed neural network, we first show the equivalence between a partial KKT point (28)-(33) and an equilibrium point of (34)-(40).

Theorem 4. Let $y = (r, z, x, \mu, \lambda, \gamma, \omega) \in \mathbb{R}^M \times \mathbb{R}^{2K} \times \mathbb{R}^K \times \mathbb{R}^{2K} \times \mathbb{R}^{K+1} \times \mathbb{R}^K \times \mathbb{R}^{4K}$, y is an equilibrium point of (34)-(40) if and only if (r, z, x) is a KKT point of (27).

Proof. Let $(r, z, x, \mu, \lambda, \gamma, \omega)$ an equilibrium point of (34)-(40), then $\frac{dr}{dt} = 0$, $\frac{dz}{dt} = 0$, $\frac{dx}{dt} = 0$, $\frac{d\mu}{dt} = 0$, $\frac{d\lambda}{dt} = 0$, $\frac{d\gamma}{dt} = 0$ and $\frac{d\omega}{dt} = 0$.
$\frac{d\mu}{dt} = 0 \Leftrightarrow (\mu + g(r,z))_+ = \mu \Leftrightarrow \mu \geq 0$ and $g(r,z) \leq 0$ and $\mu^T g(r,z) = 0 \Leftrightarrow$ (29) . We use the same approach to obtain (31) and (33).
$\frac{dr}{dt} = 0 \Leftrightarrow \nabla f(r) + \nabla_r g(r,z)^T(\mu + g(r,z))_+ = 0 \Leftrightarrow f(r) + \nabla_r g(r,z)^T \mu = 0 \Leftrightarrow$ (28) . We obtain (30) and (32) following the same steps. We conclude that (r, z, x) is a partial KKT system of (27). It is easy to check the converse part of the theorem. □

Now, to show the stability and the convergence of our neural network, we need first to prove the negative semidefiniteness of the jacobian matrix $\nabla\Phi(y)$.

Theorem 5. The jacobian matrix $\nabla\Phi(y)$ is negative semidefinite.

Proof. Let $p, q, s, t \in \mathbb{N}$ such that
$(\mu + g)_+ = (\mu_1 + g_1(r,z), \mu_2 + g_2(r,z), ..., \mu_p + g_p(r,z), \underbrace{0, ..., 0}_{2K-p})$,
$(\lambda + l)_+ = (\lambda_1 + l_1(x,z), \lambda_2 + l_2(x,z), ..., \lambda_q + l_q(x,z), \underbrace{0, ..., 0}_{K-q})$,
$(\gamma + w)_+ = (\gamma_1 + w_1(z), \gamma_2 + w_2(z), ..., \gamma_s + w_s(z), \underbrace{0, ..., 0}_{4K-s})$,

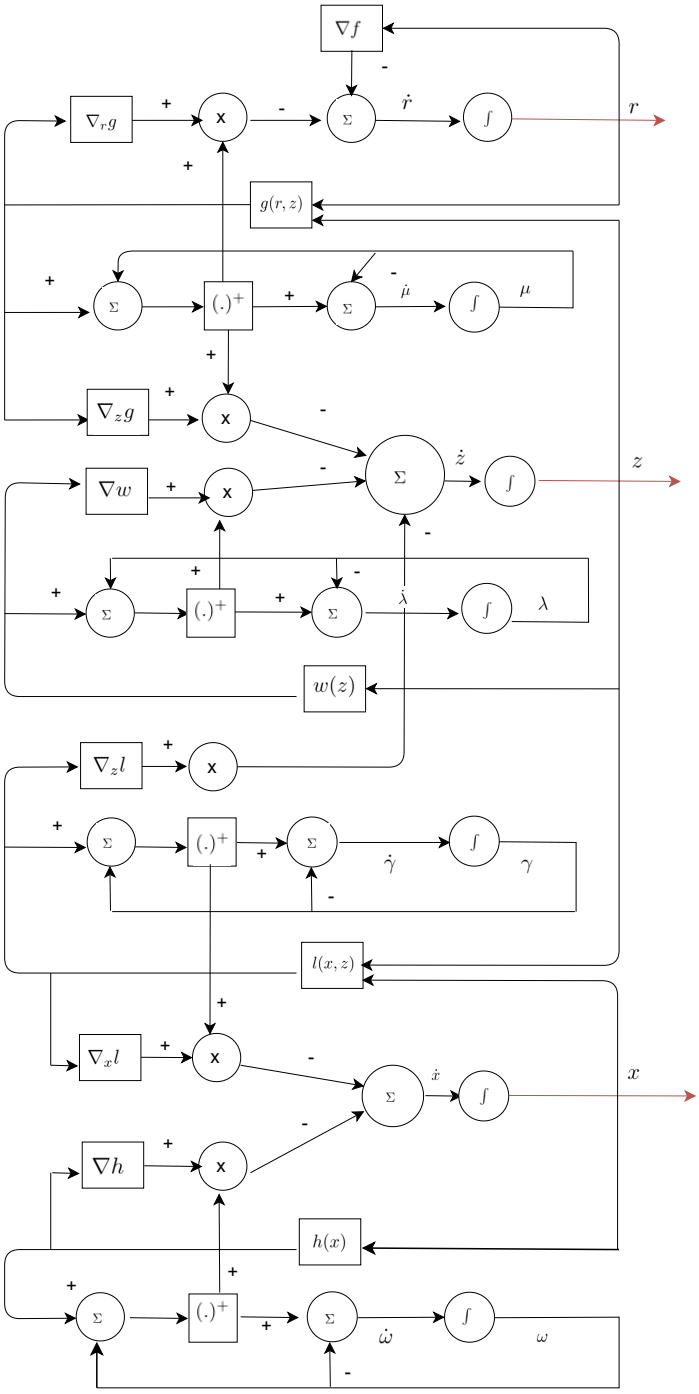

Figure 1: The architecture of the neural network (26)-(32)

$$(\omega + h)_+ = (\omega_1 + h_1(x), \omega_2 + h_2(x), ..., \omega_t + h_t(x), \underbrace{0, ..., 0}_{K+1-t}).$$

We write $\nabla\Phi(z) = \begin{bmatrix} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 & A_7 \\ B_1 & B_2 & B_3 & B_4 & B_5 & B_6 & B_7 \\ C_1 & C_2 & C_3 & C_4 & C_5 & C_6 & C_7 \\ D_1 & D_2 & D_3 & D_4 & D_5 & D_6 & D_7 \\ E_1 & E_2 & E_3 & E_4 & E_5 & E_6 & E_7 \\ F_1 & F_2 & F_3 & F_4 & F_5 & F_6 & F_7 \\ G_1 & G_2 & G_3 & G_4 & G_5 & G_6 & G_7 \end{bmatrix}$,

where,

$A_1 = -(\nabla^2 f(r) + \sum_{i=1}^{p}((\mu_i + g_i)\nabla_r^2 g_i^p(r,z)) + \nabla_r g^p(r,z)^T \nabla_r g^p(r,z))$,

$A_2 = -(\sum_{i=1}^{p}((\mu_i + g_i)\nabla_z \nabla_r g_i^p(r,z)) + \nabla_z g^p(r,z)^T \nabla_r g^p(r,z))$,

$A_4 = -\nabla_r g^p(r,z)^T$, $A_3 = 0$, $A_5 = 0$, $A_6 = 0$, $A_7 = 0$,

$B_1 = -(\sum_{i=1}^{p}((\mu_i + g_i)\nabla_r \nabla_z l_i^p(r,z)) + \nabla_z g^p(r,z)^T \nabla_r g^p(r,z))$,

$B_2 = -(\sum_{i=1}^{p}((\mu_i+g_i)\nabla_z^2 g_i^p(r,z)) + \nabla_r g^p(r,z)^T \nabla_z g^p(r,z) + \sum_{i=1}^{q}((\lambda_i+l_i)\nabla_z^2 l_i^q(x,z)) + \nabla_z l^q(x,z)^T \nabla_z l^q(x,z) + \sum_{i=1}^{s}((\gamma_i + w_i)\nabla_z^2 w_i^s(z)) + \nabla_z w^s(z)^T \nabla_z w^s(z))$,

$B_3 = -(\sum_{i=1}^{q}((\lambda_i + l_i)\nabla_x \nabla_z l_i^q(x,z)) + \nabla_z l^q(x,z)^T \nabla_x l^q(x,z))$,

$B_4 = -\nabla_z g^p(r,z)^T$, $B_5 = -\nabla_z l^q(x,z)^T$, $B_6 = -\nabla_z w^s(z)^T$, $B_7 = 0$, $C_1 = 0$,

$C_2 = -(\sum_{i=1}^{q}((\lambda_i + l_i)\nabla_z \nabla_x l_i^q(x,z)) + \nabla_x l^q(x,z)^T \nabla_z l^q(x,z))$,

$C_3 = -(\sum_{i=1}^{q}((\lambda_i+l_i)\nabla_x^2 l_i^q(x,z)) + \nabla_x l^q(x,z)^T \nabla_x l^q(x,z) + \sum_{i=1}^{t}((\omega_i+h_i)\nabla_x^2 h_i^t(x)) + \nabla_x h^t(x)^T \nabla_x h^t(x))$,

$C_4 = 0$, $C_6 = 0$, $C_5 = -\nabla_x l^q(x,z)^T$, $C_7 = -\nabla_x h^t(x)^T$,

$D_1 = \nabla_r g^p(r,z)^T$, $D_2 = \nabla_z g^p(r,z)^T$,$D_3 = 0$ $D_4 = S_p = -\begin{bmatrix} O_{p\times p} & O_{p\times(2K-p)} \\ O_{(2K-p)\times p} & I_{(2K-p)\times(2K-p)} \end{bmatrix}$,

$D_5 = 0$, $D_6 = 0$,$D_7 = 0$,

$E_1 = 0$, $E_2 = \nabla_z l^q(x,z)^T$, $E_3 = \nabla_x l^q(x,z)^T$, $E_4 = 0$, $E_5 = S_q = -\begin{bmatrix} O_{q\times q} & O_{q\times(K-q)} \\ O_{(K-q)\times q} & I_{(K-q)\times(K-q),} \end{bmatrix}$,

$E_6 = 0$, $E_7 = 0$,

$F_1 = 0$, $F_2 = \nabla_z w^s(z)^T$, $F_3 = 0$, $F_4 = 0$, $F_5 = 0$, $F_6 = S_s = -\begin{bmatrix} O_{s\times s} & O_{s\times(4K-s)} \\ O_{(4K-s)\times s} & I_{(4K-s)\times(4K-s),} \end{bmatrix}$,

$F_7 = 0$,

$G_1 = 0, G_2 = 0, G_3 = \nabla_x h^t(x)^T, G_4 = 0, G_5 = 0, G_6 = 0, G_7 = S_t = -\begin{bmatrix} O_{t\times t} & O_{t\times(K+1-t)} \\ O_{(K+1-t)\times t} & I_{(K+1-t)\times(K+1-t),} \end{bmatrix}$.

We rewrite the jacobian matrix $\nabla\Phi$ as follows, $\nabla\Phi(z) = \begin{bmatrix} A_1 & B_1^T & 0 & A_4 & 0 & 0 & 0 \\ B_1 & B_2 & B_3 & B_4 & B_5 & B_6 & 0 \\ 0 & B_3^T & C_3 & 0 & C_5 & 0 & C_7 \\ -A_4 & -B_4 & 0 & S_p & 0 & 0 & 0 \\ 0 & -B_5 & -C_5 & 0 & S_q & 0 & 0 \\ 0 & -B_6 & 0 & 0 & 0 & S_s & 0 \\ 0 & 0 & -C_7 & 0 & 0 & 0 & S_t \end{bmatrix}$,

We can represent $\nabla\Phi$ as $\nabla\Phi(z) = \begin{bmatrix} A & B \\ -B^T & C \end{bmatrix}$, where $A = \begin{bmatrix} A_1 & B_1^T & 0 \\ B_1 & B_2 & B_3 \\ 0 & B_3^T & C_3 \end{bmatrix}$, $B = \begin{bmatrix} A_4 & 0 & 0 & 0 \\ B_4 & B_5 & B_6 & 0 \\ 0 & C_5 & 0 & C_7 \end{bmatrix}$, and $C = \begin{bmatrix} S_p & 0 & 0 & 0 \\ 0 & S_q & 0 & 0 \\ 0 & 0 & S_s & 0 \\ 0 & 0 & 0 & S_t \end{bmatrix}$,

Since w and h are convex and twice differentiable, there follows that $\nabla_z^2 w_i^s(z)$ and $\nabla_x^2 h_i^t(x)$ are positive semidefinite. Furthermore, g and l are biconvex and twice differentiable, then we have $\nabla_z^2 g_i^p(r,z)$, $\nabla_z^2 l_i^q(x,z)$ $\nabla_x^2 l_i^q(x,z)$ are positive semidefinite (Gorski Jochen & Kathrin, 2007). It is easily shown that $\nabla_r g^p(r,z)^T \nabla_z g^p(r,z)$, $\nabla_z l^q(x,z)^T \nabla_z l^q(x,z)$ and $\nabla_x l^q(x,z)^T \nabla_x l^q(x,z)$ are positive semidefinite. We conclude that B_2 and C_3 are negative semidefinite and hence $\begin{bmatrix} B_2 & B_3 \\ B_3^T & C_3 \end{bmatrix}$ is negative semidefinite (Foias & Frazho, 1990). Following the same steps, we show that A_1 is negative semidefinite, and we conclude that A is negative semidefinite. We easily verify that C is negative semidefinite. We conclude that $\nabla\Phi$ is negative semidefinite. □

Theorem 6. The neural network (34)-(40) is stable and converges to $y^* = (r^*, z^*, x^*, \mu^*, \lambda^*, \gamma^*, \omega^*)$ where (r^*, z^*, x^*) is a KKT point of (27).

Before giving the proof of Theorem 6., we need to introduce the relationship between the monotocity of mapping and the semidefiniteness of its jacobian matrix.

Definition 2. Tyrrell et al. (1998) A mapping $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is said to be monotonic if

$$(x-y)^T(F(x)-F(y)) \geq 0, \qquad \forall x, y \in \mathbb{R}^n$$

Lemma 7. Tyrrell et al. (1998) A differentiable mapping $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is monotonic, if and only if the jacobian matrix $\nabla F(x)$, $\forall x \in \mathbb{R}^n$, is positive semidefinite.

Proof. of Theorem 6.

Let $y^* = (r^*, z^*, x^*, \mu^*, \lambda^*, \gamma^*, \omega^*)$ an equilibrium point of (34)-(40) and consider the Lyapunov function defined by $V_1(y) = ||\Phi(y)||^2 + \frac{1}{2}||y - y^*||^2$. We have that $\frac{dV_1(y)}{dt} \leq 0$. In fact, $\frac{dV_1(y)}{dt} = (\frac{d\Phi}{dt})^T\Phi + \Phi^T\frac{d\Phi}{dt} + (y-y^*)^T\frac{dy}{dt}$. Or since $\frac{d\Phi}{dt} = \nabla\Phi(y)\Phi(y)$, then $\frac{dV_1(y)}{dt} = \Phi^T(\nabla\Phi(y)^T + \nabla\Phi(y))\Phi + (y-y^*)^T\Phi(y)$. We use Theorem 5 and Lemma 7 to conclude.

There follows that the neural network (34)-(40) is stable in the sense of Lyapunov.

Notice that $V_1(y) \geq \frac{1}{2}\left\|y - y^*\right\|^2$, consequently there exists a convergent subsequence $(y(t_k)_{k\geq 0})$ such that $lim_{k\longrightarrow\infty} y(t_k) = \tilde{y}$ and $\frac{dV_1(\tilde{y})}{dt} = 0$.

Starting from a certain y_0, we have by LaSalle's invariance principle that the neural network converges to the largest invariant set contained in M which is defined by $M = \{y(t) | \frac{dV_1(y)}{dt} = 0\}$.

Observe that $\frac{dy}{dt} = 0 \Leftrightarrow \frac{dV_1(y)}{dt} = 0$, we have then that $\tilde{y}$ is an equilibrium point of (34)-(40).

Let show now that the neural network converges to $\tilde{y}$. For this, we consider the following Lyapunov function $V_2(y) = \left\|\Phi(y)\right\|^2 + \frac{1}{2}\left\|y - \tilde{y}\right\|^2$.

We have that V_2 is continuously differentiable, $V_2(\tilde{y}) = 0$ and $lim_{k\longrightarrow\infty} y(t_k) = \tilde{y}$, then $lim_{t\longrightarrow\infty} V_2(y(t)) = V_2(\tilde{y}) = 0$.

Additionally, since $\frac{1}{2}\|y - \tilde{y}\|^2 \leq V_2(y)$ then $lim_{t\longrightarrow\infty}\|y - \tilde{y}\| = 0$ and $lim_{t\longrightarrow\infty} y(t) = \tilde{y}$. There follows that the neural network converges to an equilibrium point $\tilde{y} = (\tilde{r}, \tilde{z}, \tilde{x}, \tilde{\mu}, \tilde{\lambda}, \tilde{\gamma}, \tilde{\omega})$ where $(\tilde{r}, \tilde{z}, \tilde{x})$ is a KKT point of (27). □

5. Numerical experiments

In order to test the performances of our proposed neural network, we study a first problem of minimizing the transportation cost. In a second subsection, we study a generalized shape optimization problem to analyze the behavior of the neural network for different sizes of problems. All the numerical experiments are done using Python. To compute the partial derivatives and the jacobians, we use the package autograd. To generate the random instances, we use the package numpy.random. The ODEs of the recurrent dynamical neural networks are solved using the function solve_ivp of scipy.integrate library. We run our algorithms on Intel(R) Core(TM) i7-10610U CPU @ 1.80GHz.

5.1. Minimizing transport cost problem

In order to shift Vm^3 grains from a warehouse to a factory, we can use an open rectangular box of length x_1 meters, of width x_2 meters, and of height x_3 meters Figure 3. The bottom costs c_1, each side costs c_2 and each end costs c_3. Each round trip of the box costs c_4. We aim to find the minimum cost of transporting Vm^3 of grain.

We use a transporter to carry the box into the truck. The floor area of the box x_1x_2 must be less then $\beta_{floor}A_{floor}$, where β_{floor} is the maximum occupancy rate and A_{floor} is the floor area and bigger then $\alpha_{floor}A_{floor}$ to avoid wasting in capacity. The same thing applies for the wall area $2x_1x_3 + 2x_2x_3$ that must be less than $\beta_{wall}A_{wall}$ and larger then $\alpha_{wall}A_{wall}$. We assume that the floor and the wall areas of the transporter are random.

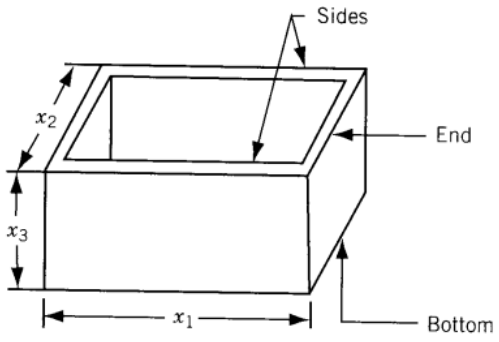

Figure 2: The shape of the box Rao (2009)

We reformulate then our minimization problem as

$$\begin{aligned} \min_{x\in\mathbb{R}^3_{++}} \quad & c_1x_1x_2 + 2c_2x_1x_3 + 2c_3x_2x_3 + c_4\frac{V}{x_1x_2x_3}, \\ \text{s.t.} \quad & \mathbb{P}(\alpha_{wall}A_{wall} \leq 2x_1x_3 + 2x_2x_3 \leq \beta_{wall}A_{wall}, \\ & \alpha_{floor}A_{floor} \leq x_1x_2 \leq \beta_{floor}A_{floor}) \geq 1-\epsilon. \end{aligned} \tag{41}$$

To solve problem (41) using our proposed neural network, we set $\alpha_{wall} = \alpha_{floor} = 50\%$, $\beta_{wall} = \beta_{floor} = 95\%$, $c_1 = 80$, $c_2 = 20$, $c_3 = 30$, $c_4 = 1$, $V = 80m^3$, $\frac{1}{A_{wall}} \sim \mathcal{N}(1.0/6.0, 0.01)$ and $\frac{1}{A_{floor}} \sim \mathcal{N}(3.0, 0.01)$.

The neural network converges to a minimum of 260.81 at $x_1 = 0.68m$, $x_2 = 0.46m$ and $x_3 = 2.01m$. We follow the convergence of x_1, x_2 and x_3 in Figure 4.

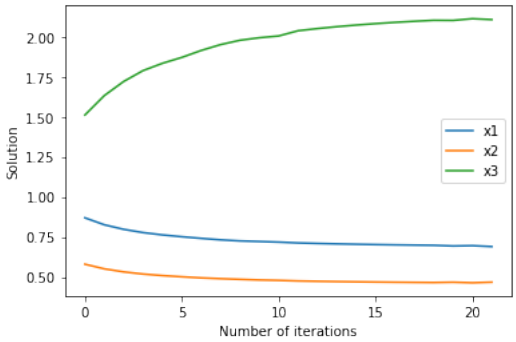

Figure 3: The convergence of the neural network of problem (33)

5.2. Stochastic shape optimization problem

In order to evaluate the performances of the proposed dynamical network, we introduce the following shape optimization problem taken from Lisser et al. (2020). We remind that $A_{wall\,j}$ and A_{floor} are random and defined as in the previous subsection. The generalized problem is defined as follows.

$$\begin{aligned} \min_{x\in\mathbb{R}^M_{++}} \quad & \prod_{i=1}^{m} x_i^{-1}, \\ \text{s.t.} \quad & \mathbb{P}(\alpha_{wall} \leq \sum_{j=1}^{m-1}(\frac{m-1}{A_{wall\,j}}x_1 \prod_{i=2,i\neq j}^{m} x_i) \leq \beta_{wall}, \\ & \alpha_{floor} \leq \frac{1}{A_{floor}}\prod_{j=2}^{m} x_j \leq \beta_{floor}) \geq 1-\epsilon. \end{aligned} \tag{42}$$

For comparison, we additionally solve the problem with individual constraints.

$$\begin{aligned} \min_{x\in\mathbb{R}^M_{++}} \quad & \prod_{i=1}^{m} x_i^{-1}, \\ \text{s.t.} \quad & \mathbb{P}\left(\alpha_{wall} \leq \sum_{j=1}^{m-1}(\frac{m-1}{A_{wall\,j}}x_1 \prod_{i=2,i\neq j}^{m} x_i) \leq \beta_{wall}\right) \geq 1-\epsilon, \\ & \mathbb{P}\left(\alpha_{floor} \leq \frac{1}{A_{floor}}\prod_{j=2}^{m} x_j \leq \beta_{floor}\right) \geq 1-\epsilon. \end{aligned} \tag{43}$$

We solve problem (43) using the same neural network where the value of y_k in (3)-(4) is $1-\epsilon$ for all $k = 1, .., K$.

For the numerical experiments, we set $\epsilon = 0.05$, $\frac{1}{A_{floor}} \sim \mathcal{N}(1.0/20.0, 0.01)$, $\frac{1}{A_{wall_j}} \sim \mathcal{N}(1.0/60.0, 0.001)$, $\alpha_{wall} = 0.5$, $\beta_{wall} = 1.0$, $\alpha_{floor} = 0.5$ and $\beta_{floor} = 1.0$.

In order to check the robustness of our approach, we generate a set of 100 scenarios of the stochastic constraints, and we visualize the number of violated scenarios (VS) for each problem.

The numerical results are represented in Table 1. Column one gives the number of variables m. Columns two, three, and four give the objective value of problem (43), the number of VS, and the corresponding CPU time, respectively. Columns five, six, and seven give the objective value of problem (42), the number of VS, and the corresponding CPU time, respectively.

m	Individual constraints			Joint constraints		
	Obj value	VS	CPU Time	Obj value	VS	CPU Time
3	0.039	3	35.27	0.042	0	25.82
5	0.117	6	61.60	0.120	3	92.20
7	0.230	4	86.36	0.236	2	80.32
10	0.440	3	203.84	0.456	1	169.57
15	0.909	5	444.26	0.907	5	531.83
20	1.384	4	1111.72	1.402	2	874.75

Table 1: Results of the generalized transportation problem for different values of m

We observe that the objective values of the two problems are relatively close. Nevertheless, the problem (42) covers better the risk region. In fact, we remark that the number of violated scenarios for $m = 5$ for problem (42) is equal to 3, whereas the number of violated scenarios for problem (43) is equal to 6 as shown in Figures 4 and 5.

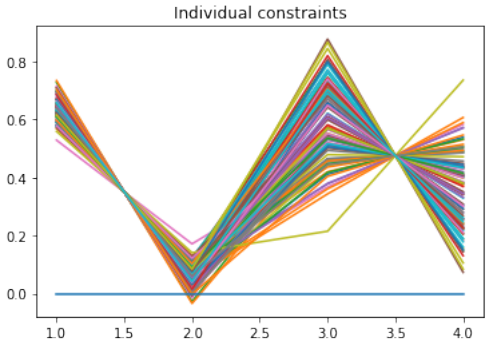

Figure 4: Out of 100 scenarios, the constraints were violated 6 times

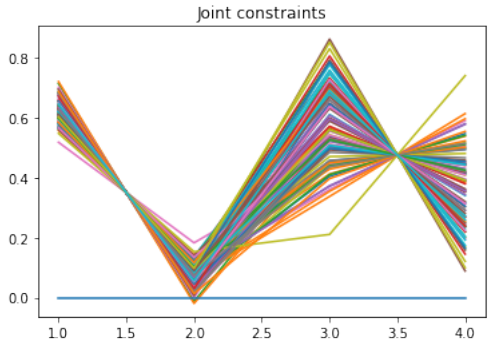

Figure 5: Out of 100 scenarios, the constraints were violated 3 times

Finally, we test the impact of the confidence level $1-\epsilon$. We solve then problems (42) and (43) for different value of ϵ and we fix $m = 5$. The obtained results are recapitulated in Table 2. We observe that as the value of ϵ increases, we obtain better minimal solutions. Although, the number of

VS increases. In fact, a higher value of means a larger risk area and a less restrictive minimization problem.

ϵ	Individual constraints			Joint constraints		
	Obj value	VS	CPU Time	Obj value	VS	CPU Time
0.1	0.115	8	39.34	0.121	1	66.67
0.15	0.113	7	39.44	0.115	5	38.02
0.2	0.112	28	74.80	0.114	15	44.32
0.3	0.109	31	49.87	0.111	24	49.84
0.4	0.107	48	44.21	0.109	39	52.20

Table 2: Results of the generalized transportation problem for different values of ϵ

To the best of our knowledge, the only work dealing with rectangular programs with joint probabilistic constraints is Lisser et al. (2020). They propose new convex approximations based on the variable transformation and piecewise linear approximation methods to come up with lower and upper bounds for the optimal solutions. Since in the worst case, our neural network converges to a partial optimum of the minimization problem, we converge then at worst to an upper bound of the optimal solution. The advantage of our approach is that we don't use any convex approximations to approximate the optimal solution. Although, the neurodynamic approach takes more time to solve the stochastic rectangular programs as the size of the problem increases compared to the convex approximations. Therefore, we must note that our approach doesn't replace the existed convex approximations but gives some promising results and opens the way for a new vision of the joint probabilistic problems.

6. Conclusion

This paper studied a particular case of rectangular problems with joint probabilistic constraints. Using a log variable transformation, we transform the probabilistic model into a deterministic one. We introduce a dynamical neural network to solve our program based on the partial KKT system of the obtained deterministic biconvex problem. We show the neural network's stability and convergence to a partial KKT point of the initial problem. To show the performance of the approach, we study a minimum-cost transportation problem and a generalized shape optimization problem. Our numerical experiments show that our dynamical neural network-based joint probabilistic model is more robust than the individual probabilistic model counterpart.

References

Aluffi-Pentini, F., Parisi, V., & Zirilli, F. (1985). Global optimization and stochastic differential equations. *Journal of Optimization Theory and Applications*, *47*, 1–16.

Boyd, S. P., Kim, S., Patil, D., & Horowitz, M. (2005). Digital circuit optimization via geometric programming. *Oper. Res.*, *53*, 899–932. URL: https://doi.org/10.1287/opre.1050.0254. doi:10.1287/opre.1050.0254.

Charnes, A., & Cooper, W. W. (1959). Chance constrained programs with normal deviates and linear decision rules.

Cheng, J., & Lisser, A. (2012). A second-order cone programming approach for linear programs with joint probabilistic constraints. *Operations Research Letters*, *40*, 325–328. doi:https://doi.org/10.1016/j.orl.2012.06.008.

Chiang, M. (2005). Geometric programming for communication systems. *Foundations and Trends® in Communications and Information Theory*, *2*, 1–154. URL: http://dx.doi.org/10.1561/0100000005. doi:10.1561/0100000005.

Chiang, M., Tan, C. W., Palomar, D. P., O'neill, D., & Julian, D. (2007). Power control by geometric programming. *IEEE Transactions on Wireless Communications*, *6*, 2640–2651. doi:10.1109/TWC.2007.05960.

Dupačová, J. (2010). Stochastic geometric programming with an application. *Kybernetika*, *46*, 374–386. URL: http://eudml.org/doc/196973.

Effati, S., & Nazemi, A. (2006). Neural network models and its application for solving linear and quadratic programming problems. *Applied Mathematics and Computation*, *172*, 305–331. URL: https://www.sciencedirect.com/science/article/pii/S0096300305002055. doi:https://doi.org/10.1016/j.amc.2005.02.005.

Faybusovich, L. (1991). Dynamical Systems which Solve Optimization Problems with Linear Constraints. *IMA Journal of Mathematical Control and Information*, *8*, 135–149. URL: https://doi.org/10.1093/imamci/8.2.135. doi:10.1093/imamci/8.2.135.

Foias, C., & Frazho, A. E. (1990). Positive definite block matrices. In *The Commutant Lifting Approach to Interpolation Problems* (pp. 547–586). Basel: Birkhäuser Basel. URL: https://doi.org/10.1007/978-3-0348-7712-1_16. doi:10.1007/978-3-0348-7712-1_16.

Gorski Jochen, P. F., & Kathrin, K. (2007). Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research*, (p. 373–467). doi:10.1007/s00186-007-0161-1.

Hoburg, W., & Abbeel, P. (2012). Geometric programming for aircraft design optimization. volume 52. doi:10.2514/6.2012-1680.

Jiang, M., Meng, Z., & Shen, R. (2021). Partial exactness for the penalty function of biconvex programming. *Entropy*, *23*. doi:10.3390/e23020132.

Lisser, A., Liu, J., & Peng, S. (2020). Rectangular chance constrained geometric optimization. *Optimization and Engineering*, *21*, 1573–2924. doi:https://doi.org/10.1007/s11081-019-09460-3.

Liu, C.-S., Xu, G., , & Wang, L. (2014). An improved geometric programming approach for optimization of biochemical systems. *Journal of Applied Mathematics*, *2014*. doi:10.1155/2014/719496.

Liu, J., Lisser, A., & Chen, Z. (2016). Stochastic geometric optimization with joint probabilistic constraints. *Operations Research Letters*, *44*, 687–691. doi:https://doi.org/10.1016/j.orl.2016.08.002.

Liu, J., Peng, S., Lisser, A., & Chen, Z. (2020). Rectangular chance constrained geometric optimization. *Optimization and Engineering*, *21*, 537–566. URL: https://hal-centralesupelec.archives-ouvertes.fr/hal-02950874. doi:10.1007/s11081-019-09460-3.

Perlumutter, D. D. (1967). Geometric programming–theory and application. *AIChE Journal*, *13*, 829–830. URL: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690130408. doi:https://doi.org/10.1002/aic.690130408.

Rao, S. S. (2009). Geometric programming. In *Engineering Optimization* chapter 8. (pp. 492–543). John Wiley Sons, Ltd. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470549124.ch8. doi:https://doi.org/10.1002/9780470549124.ch8. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470549124.ch8.

Schropp, J., & Singer, I. (2000). A dynamical systems approach to constrained minimization. *Numerical Functional Analysis and Optimization*, *21*, 537–551. URL: https://doi.org/10.1080/01630560008816971. doi:10.1080/01630560008816971.

Tyrrell, R., Roger, J., & Wets, B. (1998). *Variational Analysis*. Springer, Berlin, Heidelberg. doi:10.1007/978-3-642-02431-3.

Vanderhaegen, J. P., & Brodersen, R. W. (2004). Automated design of operational transconductance amplifiers using reversed geometric programming. In *Proceedings of the 41st Annual Design Automation Conference* DAC '04 (p. 133–138). New York, NY, USA: Association for Computing Machinery. URL: https://doi.org/10.1145/996566.996608. doi:10.1145/996566.996608.

Xu, G. (2014). Global optimization of signomial geometric programming problems. *European Journal of Operational Research*, *233*, 500–510. URL: https://www.sciencedirect.com/science/article/pii/S0377221713008394. doi:https://doi.org/10.1016/j.ejor.2013.10.016.