Computers and Chemical Engineering 198 (2025) 109112

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Analytical identification of process design spaces using R-functions

S. Kucherenko [a,*], N. Shah [a], O.V. Klymenko [b]

[a] Imperial College London, London, SWT 2AZ, UK
[b] School of Chemistry and Chemical Engineering, University of Surrey, Guildford GU2 7XH, UK

ARTICLE INFO

Keywords:
Design space
R-functions
Multivariate polynomial model
Pharmaceutical manufacturing

ABSTRACT

A process design space (DS) is defined as the combination of process design and operational conditions that guarantees the assurance of product quality. This principle ensures that, as long as a process operates within its DS, it consistently yields a product that meets specifications. A novel DS identification method called the R-DS identifier has been developed in this work. It makes no assumptions about the underlying model - the only requirement is that each model constraint (e.g., defining product Critical Quality Attributes or process Key Performance Indicators) should be approximated by a closed-form function, e.g., a multivariate polynomial model. The method utilizes the methodology of V.L. Rvachev's R-functions and allows for explicit analytical representation of the DS with only a limited number of model runs. R-functions provide a framework for representing complex geometric shapes and performing operations on them through implicit functions and inequalities defining the regions. The theory of R-functions enables the solution of geometric problem such as identification of DS through algebraic manipulation. It is more practical than traditional sampling or optimization-based methods. The method is illustrated using a batch reactor model.

1. Introduction

The pharmaceutical manufacturing industry is subject to strict requirements to ensure drug quality and safety. The modern approach to regulating pharmaceutical product development utilizes the concept of Quality by Design (QbD), introduced by the US Food and Drug Administration and the International Council for Harmonisation (ICH, 2009). QbD is based on the principle that quality should be built directly into the design of products and manufacturing processes. This approach ensures that the physical or chemical properties of a product, known as critical quality attributes (CQAs), remain within specified limits. CQAs can vary during processing due to inevitable variations in critical process parameters (CPPs), which are closely monitored and controlled. To manage variability in process operational parameters, the QbD principle includes the use of the design space (DS), defined as "the multidimensional combination and interaction of input variables and process parameters that have been demonstrated to provide assurance of quality" (ICH, 2009). Within the regulator-approved DS, operators can adjust (or tolerate fluctuations of) process parameter values without violating CQAs, providing manufacturers with operational flexibility that does not require further approvals or product quality testing. Importantly, the QbD paradigm involves the use of model-based approaches to identify acceptable operating conditions (i.e., the DS) at the process design as opposed to Quality-by-Testing that relies on exhaustive experimentation which may be costly, time-consuming and therefore potentially infeasible.

A probability-based concept of the DS was introduced by Castagnoli et al., 2010, defining DS as a set of process conditions where the probability of meeting Critical Quality Attributes (CQA) exceeds business risk, accounting for parameter uncertainty. There are three primary methods for identifying DS: optimization techniques, Bayesian inference and sampling of the knowledge space.

Optimization-based methods, rooted in flexibility analysis (Halemane and Grossmann, 1983; Grossmann and Morari, 1983), involve solving a problem under CQA constraints. The primary objective of flexibility analysis is to identify and characterise the flexibility region, which defines the operational conditions that ensure feasible and safe process performance. Swaney and Grossmann (1985a,b) introduced the flexibility index as the largest deviation in uncertain parameters (within a specified range) for which the system remains feasible.

Flexibility index and design centering are two complementary approaches for evaluating candidate DSs. From a geometric viewpoint, the flexibility index seeks to identify the largest hyperrectangle (or other geometric shape) that can be inscribed within the feasible region, while

* Corresponding author.
E-mail address: s.kucherenko@imperial.ac.uk (S. Kucherenko).

https://doi.org/10.1016/j.compchemeng.2025.109112
Received 27 November 2024; Received in revised form 22 March 2025; Accepted 24 March 2025
Available online 26 March 2025
0098-1354/© 2025 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

design centering aims to locate the center of that region, maximizing robustness to variations. Zhao et al. (2022) proposed a novel flexibility index formulation using a direction search method, capable of handling design spaces of any shape. However, its application was demonstrated only for rectangular and elliptical cases, highlighting the method's complexity. Although optimization methods are computationally efficient, they may lead to non-unique solutions and complex multi-level problems. Additionally, these methods are typically limited to processes with closed-form and differentiable constraints. Tian et al. (2024) presented an overview of key concepts and computational methods for evaluating feasibility and flexibility, including recent advances and applications in optimizing process design and operations.

Zhao and Chen (2018) proposed an operational flexibility analysis method to explicitly describe the flexibility region and establish functional relationships between uncertain parameters and control variables. The problem is framed as an existential quantifier problem, solved using the cylindrical algebraic decomposition method to eliminate quantifiers and derive triangular quantifier-free formulas. A logical combination of these formulas defines the flexibility region and provides a control strategy for feasible operations. Although it is limited to polynomial functions, the method can handle non-polynomial systems using local polynomial approximations. Due to the computational intensity of symbolic computation needed to apply the cylindrical algebraic decomposition method, the computational burden increases exponentially with system size, hence the method is only applicable to relatively small-scale problems.

Bayesian approaches (Kusumo et al., 2019) offer flexible DS estimation by incorporating uncertainty and prior knowledge, making them particularly useful when data is limited. Sampling-based methods assess dense grids of parameter space points, with deterministic DS derived from simulations and probabilistic DS obtained using Monte Carlo simulations (Figueroa et al., 2013; García-Muñoz et al., 2015). While accurate, this can be computationally expensive. Metamodels (Rogers and Ierapetritou, 2015; Wang and Ierapetritou, 2018) improve efficiency in high-dimensional models.

Original sampling-based methods for defining the Design Space (DS) involved creating a dense grid of sample points across the process parameter space and performing simulations at each point to assess whether the CQAs are met, resulting in a set of feasible points defining a deterministic DS. When uncertainties are present in the process model, a probabilistic DS can be constructed via a Monte Carlo simulation over uncertain process parameters at each sampling point in the space of deterministic operational parameters. The probability is estimated as the fraction of times the Monte Carlo simulations produce results that comply with all CQAs. Although computationally demanding, this method provides a straightforward way to identify the DS (García-Muñoz et al., 2015). Identification of probabilistic DS is a demanding task and for a typical practical problem the traditional approach based on exhaustive sampling requires costly computations. A novel theoretical and numerical framework for determining probabilistic DS using metamodelling and adaptive sampling was proposed in Kucherenko et al. (2020). It is based on the multi-step adaptive technique using a metamodel for a probability map as an acceptance-rejection criterion to optimize sampling for identification of the DS. This methodology was further developed for the case of multiple unit operations in Kucherenko et al. (2024).

Wang and Ierapetritou (2018) provided an overview of recent advancements in the use of global sensitivity analysis (GSA) and flexibility analysis techniques, aimed at enhancing the understanding of pharmaceutical processes. Global Sensitivity Analysis (GSA) attributes the uncertainty in model outputs to the uncertainty in input factors, characterized by probability distribution functions (Saltelli et al., 2004; Sobol and Kucherenko, 2005). It is widely applied in decision-making, risk assessment, model simplification, and other fields. Unlike local sensitivity analysis, GSA accounts for interactions between variables by evaluating the effect of varying a specific input (or set of inputs) while allowing all other inputs to vary as well, making it independent of any "nominal" point. GSA is suitable for non-linear, static, and time-dependent models. It is commonly used to rank variables by importance or to identify non-essential variables that can be fixed at nominal values to reduce model complexity. Among GSA methods, Sobol's variance-based global sensitivity analysis method has become the preferred choice among practitioners because of its efficiency and straightforward interpretability. Kucherenko et al. (2025) demonstrated that application of variance based GSA can significantly reduce model complexity and reduce computational time for identifying and quantifying a probabilistic DS by screening out nonimportant uncertain parameters.

A challenge remains in the visualization of DS needed to communicate high-dimensional probability maps in formal documentation required for the approval of the DS by a government agency. For more than three process parameters visualization of a DS is a challenging task. Guidance for Industry, Q8(R2) Pharmaceutical Development (2009) has the following recommendations: "When multiple parameters are involved, the design space can be presented for two parameters, at different values (e.g., high, middle, low) within the range of the third parameter, the fourth parameter, and so on." Considering DS in n-dimensional process parameters space, a graphical presentation of DS via 2D plots prescribed in Guidance for Industry (2009) would result in $N_P = \frac{n(n-1)}{2}\, 3^{n-2}$ two-dimensional plots.

In this paper, we introduce the R-DS method, which utilizes V.L. Rvachev's R-function methodology (Rvachev, 1982) to analytically represent the DS. By leveraging R-functions, complex geometric shapes can be represented and manipulated through implicit functions, enabling the identification of the DS via algebraic operations. This approach significantly reduces the number of required model runs compared to traditional methods. Unlike previous studies where the DS was identified by simultaneously considering all CQAs, our approach involves a two-stage process. First, we identify feasible subspaces of the process operational parameter space for each constraint separately, deriving implicit functions to analytically describe their boundaries. Next, we use the theory of R-functions to define the DS as the intersection of these feasible subspaces. This method yields the DS given by a single analytical expression, which also addresses the issue of communicating high-dimensional DS to regulators in formal documentation. Moreover, the analytical definition of the DS using R-functions offers additional advantages, such as streamlining operations like "boxing" (determining a tight axis-aligned bounding box), identifying points within the feasible domain, and calculating the domain's volume. In addition, this method also allows the handling non-convex and disjoint DS shapes.

This paper is organized as follows: Section 2 provides the mathematical formulation of the DS problem. Section 3 provides an overview of the theory of R-functions and demonstrates their utility for the description of geometrical domains using 2D and 3D examples (expanded upon in the Appendix). Section 4 introduces the methodology for identifying the DS of a process model using R-functions and illustrates its application on the model of a batch reactor system. Conclusions are presented in Section 5.

2. Definition of design spaces

Consider vector functions $\boldsymbol{f}(\boldsymbol{u};\boldsymbol{\theta})$ and $\mathbf{g}(\boldsymbol{u};\boldsymbol{\theta})$ representing a process model and a set of inequality constraints (product CQAs), respectively:

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{u};\boldsymbol{\theta}), \tag{1}$$

$$\mathbf{g}(\boldsymbol{u};\boldsymbol{\theta}) \geq \mathbf{g}^{*}, \tag{2}$$

where $\boldsymbol{y}$ is a vector of model outputs (e.g., state variables). Here $\boldsymbol{u}$ is a vector of process operational parameters defined in the n-dimensional real space $\mathbb{R}^n$, $\boldsymbol{\theta}$ is a vector of uncertain parameters defined in the m-

dimensional real space $\mathbb{R}^m$ with a given probability density function (PDF) $\varphi(\boldsymbol{\theta})$, $\mathbf{g}^*$ is a vector of L thresholds for the process CQAs. Function $\mathbf{g}(\boldsymbol{u};\boldsymbol{\theta})$ can (and often does) depend on model outputs $\boldsymbol{y}$ but this dependence is not indicated explicitly here for brevity. The DS is then defined as:

$$DS = DS(\boldsymbol{u} \mid p(\boldsymbol{u}) \geq p^*), \tag{3}$$

$$p(\boldsymbol{u}) = E_\theta[I(\mathbf{g}(\boldsymbol{u};\boldsymbol{\theta}) \geq \mathbf{g}^*)]. \tag{4}$$

Eqs. (3) and (4) define the DS as a set of values of process parameters $\boldsymbol{u}$ such that the probability of the process meeting the CQA constraints is greater than the critical probability threshold p^*. Here $I(\cdot)$ represents an indicator function taking the value of either 1 (satisfying constraints) or 0 (not satisfying constraints).

Consider a case when a vector of uncertain model parameters $\boldsymbol{\theta}$ has a fixed value $\boldsymbol{\theta} = \boldsymbol{\theta}^*$ and p^*=1. It is formally equivalent to the probability density function $\varphi(\boldsymbol{\theta})$ being equal to the delta function: $\varphi(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$. Then DS is defined as

$$DS = DS(\boldsymbol{u}|p(\boldsymbol{u}) = 1), \tag{5}$$

or equivalently

$$DS = DS(\boldsymbol{u}|\mathbf{g}(\boldsymbol{u};\boldsymbol{\theta}^*) \geq \mathbf{g}^*). \tag{6}$$

This setting can also be seen as deterministic and as a subset of the general formulation (3), (4). In this work, we consider primarily the deterministic setting, but the developed methodology can be easily extended for the case of probabilistic setting. In presenting the results, we include the probabilistic setting for comparison purposes. Further we omit the dependence of $\mathbf{g}(\boldsymbol{u};\boldsymbol{\theta}^*)$ on the constant vector of parameters $\boldsymbol{\theta}^*$ for brevity.

We highlight the following important equality for the proposed methodology:

$$DS(\boldsymbol{u}|\mathbf{g}(\boldsymbol{u}) \geq \mathbf{g}^*) = DS\left(\boldsymbol{u}|\bigcap_{i=1}^{L}\left(\boldsymbol{u}|g_i(\boldsymbol{u}) \geq g_i^*\right)\right) = \bigcap_{i=1}^{L} DS_i\left(\boldsymbol{u}|g_i(\boldsymbol{u}) \geq g_i^*\right). \tag{7}$$

which is valid for any finite number of constraints L.

3. Introduction to R-Functions

René Descartes introduced the coordinate method, linking geometric objects such as points, lines, and shapes with analytical representations like numbers, equations, and inequalities. Historically, much attention has been devoted to the direct problem of analytic geometry: analyzing curves and surfaces defined by equations. The inverse problem- representing geometric objects with equations- has been solved for basic shapes like lines, circles, conic sections, and second-degree surfaces, typically described by polynomial equations. While polynomials are relatively simple, solving these equations often proves challenging, limiting their application to more complex shapes. However, by incorporating additional operations, the descriptive capabilities of analytical methods can be significantly expanded. This expansion enables the creation of equations for geometric objects of almost any shape, effectively solving the inverse problem of analytic geometry in principle.

Rvachev's R-functions, named after the Ukrainian mathematician Vladimir Rvachev (Rvachev, 1982; Shapiro, 1991), are a versatile mathematical tool widely used in fields such as computer graphics, computational geometry, and solid modeling. They offer a formal framework for representing complex geometric shapes and performing operations on them using implicit functions. This introduction highlights the core concepts, applications, and advantages of R-functions.

R-functions are based on the use of implicit functions to define geometric objects. An implicit function $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^n$ represents a shape in $\mathbb{R}^n$ such that the points inside the shape satisfy $f(\boldsymbol{x}) > 0$, points on the boundary satisfy $f(\boldsymbol{x}) = 0$, and points outside satisfy $f(\boldsymbol{x}) < 0$. R-functions extend this principle by enabling set-theoretic operations, such as union, intersection, and difference, through algebraic combinations of these implicit functions. These combinations implement logical operations like conjunction (AND), disjunction (OR), and negation (NOT), making R-functions a powerful tool for manipulating and combining geometric shapes.

3.1. Applications of R-functions

R-functions find applications across a wide range of fields. In computer graphics, they facilitate procedural modeling, enabling the creation of complex shapes using simple algebraic expressions. In computational geometry, R-functions are instrumental in representing and manipulating geometric objects, making them essential for tasks such as collision detection, mesh generation, and object recognition. In solid modeling, they offer robust tools for constructing and editing solid models in CAD systems, supporting Boolean operations and freeform shape design. Additionally, in engineering analysis, R-functions are used in finite element analysis and other simulations to precisely define domains and boundary conditions.

The use of R-functions offers several advantages:

1. Compact representation: They allow for expressing complex shapes using simple algebraic expressions, reducing storage and computational needs.
2. Smooth transitions: R-functions ensure smooth transitions between combined shapes, which is beneficial for applications requiring high-quality surface continuity.
3. Versatility: The framework allows for easy modification and combination of shapes, making it highly adaptable for various design and analysis tasks.
4. Mathematical rigor: R-functions provide a mathematically rigorous way to perform set-theoretic operations on geometric objects, ensuring consistency and accuracy.

3.2. Theory of R-functions

Function $y = f(x_1, \ldots, x_n)$ is called an R-function if there exists a Boolean function $Y = F(X_1, \ldots, X_n)$ such that the following equality holds for any $(x_1, \ldots, x_n) \in \mathbb{R}^n$ (Shapiro, 1991):

$$S[(f(x_1, \ldots, x_n)] = F[S(x_1), \ldots, S(x_n)]. \tag{8}$$

Here $S(x)$ is a step function:

$$S(x) = \begin{cases} 0 & \text{if } x \leq -0, \\ 1 & \text{if } x \geq +0. \end{cases} \tag{9}$$

$X_1, \ldots, X_n$ are Boolean variables which are obtained by applying the step function $S(x)$ to each real variable: $X_i = S(x_i)$ for $i = 1, 2, \ldots, n$. We recall that a Boolean function is a mathematical function that takes inputs from a set of binary values (usually 0 and 1, or *false* and *true*) and produces a binary output. The Boolean function F is referred to as the companion function of a given R-function.

A real-valued function qualifies as an R-function if its sign changes only when one or more of its arguments also change sign. Basic R-functions can be defined using logical operations on implicit functions:

- Union (OR): The union of two implicit functions f_1 and f_2 is defined by $f_\vee \geq 0$ if $f_1 \geq 0$ or $f_2 \geq 0$. It is also called R-disjunction.
- Intersection (AND): The intersection of two implicit functions f_1 and f_2 is defined by $f_\wedge \geq 0$ if $f_1 \geq 0$ and $f_2 \geq 0$. It is also called R-conjunction.
- Complement (NOT): The negation of f describes the complement of the set defined by this function: $f_\neg \geq 0$ if $f \leq 0$. It is also called R-negation.

In all the definitions above, the functions depend on the input $(x_1,\ldots,x_n)$. These operations use specific algebraic combinations of implicit functions to preserve geometric properties.

According to the above definition, the following real-valued functions qualify as R-functions, with their respective Boolean companion functions indicated in parentheses (Shapiro, 1991):

$$\begin{aligned} x_1 \wedge_1 x_2 &= \frac{1}{2}\left(x_1 + x_2 - \left(x_1^2 + x_2^2 - 2x_1x_2\right)^{\frac{1}{2}}\right), \\ x_1 \vee_1 x_2 &= \frac{1}{2}\left(x_1 + x_2 + \left(x_1^2 + x_2^2 - 2x_1x_2\right)^{\frac{1}{2}}\right). \end{aligned} \tag{16}$$

These expressions can also be rewritten in the following form:

$$C \equiv \text{const}; \quad \text{(logical 1)}$$
$$\overline{x} \equiv -x; \text{ (logical negation } \neg\text{)}\ x_1 \wedge_1 x_2 \equiv \min(x_1,x_2); \text{ (logical conjunction } \wedge\text{)}\ x_1 \vee_1 x_2 \equiv \max(x_1,x_2); \text{ (logical disjunction } \vee\text{)}$$

R-functions are infinite in variety, but only a sufficiently complete system is needed for practical applications. For instance, conjunction ($\wedge$) and negation ($\neg$) form a complete system. In this work we consider the following general set R_α of R-functions:

$$\begin{aligned} x_1 \wedge_\alpha x_2 &\equiv \frac{1}{1+\alpha}\left(x_1 + x_2 - \sqrt{x_1^2 + x_2^2 - 2\alpha x_1x_2}\right); \\ x_1 \vee_\alpha x_2 &\equiv \frac{1}{1+\alpha}\left(x_1 + x_2 + \sqrt{x_1^2 + x_2^2 - 2\alpha x_1x_2}\right), \end{aligned} \tag{12}$$

where $\alpha(x_1,x_2)$ is an arbitrary symmetric function such that $-1 < \alpha(x_1, x_2) \leq 1$. Value of α defines smoothness of R-functions. The first function in (12) defines R-conjunction, while the second function defines R-disjunction. Negation is defined as

$$\overline{x} \equiv -x. \tag{13}$$

3.3. Describing geometrical objects with real functions

This section explores how geometric objects can be represented using real functions in Euclidean space $\mathbb{R}^n$ and a theory of R-function.

Drawings and Regions: A *drawing* is defined as the set of points where a function $\phi(x_1,\ldots,x_n) = 0$ holds. A *region* is defined by points satisfying $\phi(x_1,\ldots,x_n) \geq 0$. Many well-known geometric shapes (curves and surfaces) are algebraic drawings. These are often defined by polynomial functions, serving as "building blocks" for more complex shapes.

Composite Geometric Regions: A composite geometric region $D \subseteq \mathbb{R}^n$ can be described as:

$$D = F[(\phi_1 \geq 0),\ldots,(\phi_m \geq 0)], \tag{14}$$

where $\phi_i(x_1,\ldots,x_n) \geq 0$ define primitive geometric regions. F is a set function constructed using standard set operations $\cap, \cup, \setminus$ on these primitives.

Logical Representation: F can also be viewed as a Boolean function where set operations are replaced with logical functions $(\Lambda, \vee, \neg)$.

For a composite object D defined as a closed subset of $\mathbb{R}^n$, it can be proven (Rvachev, 1982; Shapiro, 1991) that a real function defining the region D can be obtained by constructing an appropriate R-function and substituting its arguments with the real functions ϕ_i defining the primitive regions:

$$f(\phi_1,\ldots,\phi_m) \geq 0. \tag{15}$$

This formulation provides a mathematical framework for representing complex geometric shapes using real-valued functions. In the next Section we show how it works in practice.

3.4. Illustrative examples

In the following, we will be using the branch of R-functions R_1 (i.e., an instance of system R_α with $\alpha = 1$). Then from (12) we obtain:

$$\begin{aligned} x_1 \wedge_1 x_2 &= \frac{1}{2}(x_1 + x_2 - |x_1 - x_2|), \\ x_1 \vee_1 x_2 &= \frac{1}{2}(x_1 + x_2 + |x_1 - x_2|). \end{aligned} \tag{17}$$

We assume that we are given a set of implicit functions $(\phi_1,\ldots,\phi_m)$ describing individual elementary geometric regions through inequalities $\phi_k \geq 0,\ k = 1,\ldots,m$, which define a more complex geometric region of interest. Following the theory of R-functions, if all the inequalities must be simultaneously satisfied, the resulting region is defined by applying the continuous R-conjunction to all functions ϕ_k:

$$f_\wedge(\phi_1,\ldots,\phi_m) = (\phi_1 \wedge_1 \ldots \wedge_1 \phi_m) \geq 0. \tag{18}$$

Another possible alternative is a "disjunctive region" consisting of points for which at least one of the inequalities is satisfied. It can be defined using R-disjunction in the following way:

$$f_\vee(\phi_1,\ldots,\phi_m) = (\phi_1 \vee_1 \ldots \vee_1 \phi_m) \geq 0. \tag{19}$$

Shapes of any complexity can be constructed in a similar way by applying the three R-functions in an appropriate way as illustrated below in 2D and with additional examples in 3D presented in Appendix.

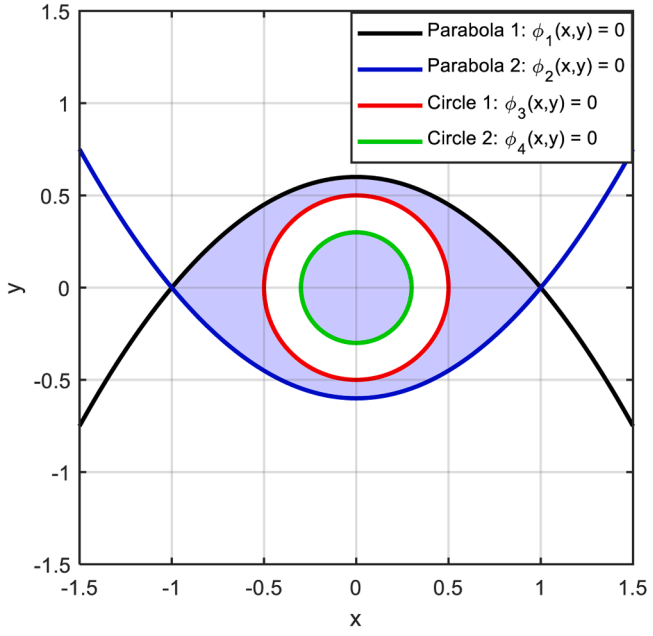


Fig. 1. The eye-shaped domain generated using the R-function (22) produced from the primitives in (20).

3.5. A disjoint and non-simply connected region

In this Section we construct a complex region using all three R-operations (R-conjunction, R-disjunction and negation) to illustrate how they can be used to generate shapes of any complexity including those with holes or consisting of multiple disjoint domains. For this example, we define the following four constraints (or primitives):

$$\begin{array}{rcl}\phi_1(x,y) & = & -y + 0.6(1 - x^2) \geq 0,\\ \phi_2(x,y) & = & y + 0.6(1 - x^2) \geq 0,\\ \phi_3(x,y) & = & 0.5^2 - x^2 - y^2 \geq 0,\\ \phi_4(x,y) & = & 0.3^2 - x^2 - y^2 \geq 0.\end{array} \tag{20}$$

These constraints represent the area under the parabola (i) $y = 0.6(1 - x^2)$, (ii) area above the parabola $y = 0.6(x^2 - 1)$, (iii) inside of the circle of radius 0.5 centred at the origin, and (iv) inside of the circle of radius 0.3 centred at the origin.

The shaded area in Fig. 1 is constructed through a series of logical operations applied to the primitives in (21), as described by the following R-function:

$$f = \varphi_1 \wedge_1 \varphi_2 \wedge_1 (\overline{\varphi_3} \vee_1 \varphi_4). \tag{21}$$

The process begins by forming the intersection of the regions beneath parabola 1 and above parabola 2 ($\varphi_1 \wedge_1 \varphi_2$), resulting in an "eye"-shaped region. To remove the "iris" from the eye, we negate the inequality $\varphi_3 \geq 0$, which describes the interior of the larger circle, making its interior infeasible. This negated region is then combined using R-disjunction with the "pupil" of the eye, defined by $\varphi_4 \geq 0$ ($\overline{\varphi_3} \vee_1 \varphi_4$). Finally, this result is intersected with the previously formed "eye"-shaped region, producing the expression given in (22).

$$\begin{array}{c} f = \dfrac{|A|}{4} - \dfrac{|25|A| + 50|y| + 30x^2 - 34|}{100} - \dfrac{|y|}{2} - \dfrac{3x^2}{10} + \dfrac{13}{50},\\ A = 2x^2 + 2y^2 - \dfrac{17}{50}. \end{array} \tag{22}$$

This example shows the flexibility of the R-function method for generating geometrical regions of any level of complexity.

4. Design space description using R-functions

4.1. R-DS identification method

The definition of a DS in (7) readily lends itself to the application of *R*-functions to find its representation as a single implicit algebraic inequality, provided the individual constraints $g_i(\boldsymbol{u}),\ i = 1, \ldots, L$ can be defined in an explicit form. When modelling realistic processes, this is not always feasible as the constraints can either be data-driven or defined implicitly (e.g., when the value of the constraint depends on the solution of the process model $f(\boldsymbol{u}; \boldsymbol{\theta}^*)$, in which case the whole model must be solved to establish whether a given set of process parameter values $\boldsymbol{u}$ belongs to the DS). Therefore, we propose a novel method of finding an explicit analytical form of the whole DS which involves three key stages:

1. approximation of implicitly defined constraints $g_i(\boldsymbol{u}) \geq 0$ with Multivariate Polynomial Regression metamodels $g_i^M(\boldsymbol{u}),\ i = 1, \ldots, L$;
2. definition of approximate feasible regions corresponding to individual constraints $DS_i \approx DS_i\left(\boldsymbol{u} | g_i^M(\boldsymbol{u}) \geq g_i^*\right)$ via implicit functions $\phi_i(\boldsymbol{u}) = g_i^M(\boldsymbol{u}) - g_i^*$:

$$DS_i \approx DS_i(\boldsymbol{u} | \phi_i(\boldsymbol{u}) \geq 0), i = 1, \ldots, L \tag{23}$$

3. construction of the joint *DS* using R-conjunction (see Sections 3.3-3.4):

$$DS = \bigwedge_{i=1}^{L} DS_i \text{ as } DS = DS(\boldsymbol{u} |\ f_\wedge(\phi_1(\boldsymbol{u}), \ldots, \phi_L(\boldsymbol{u})) \geq 0). \tag{24}$$

A detailed illustration of the methodology is provided in the flowchart (Fig. 2), which outlines the process of constructing metamodels for individual constraints using a sample of model runs within an appropriate subspace of the parameter space $\boldsymbol{u} \in \mathbb{R}^n$.

This algorithm is more efficient and practical than the existing sampling-based methods in that:

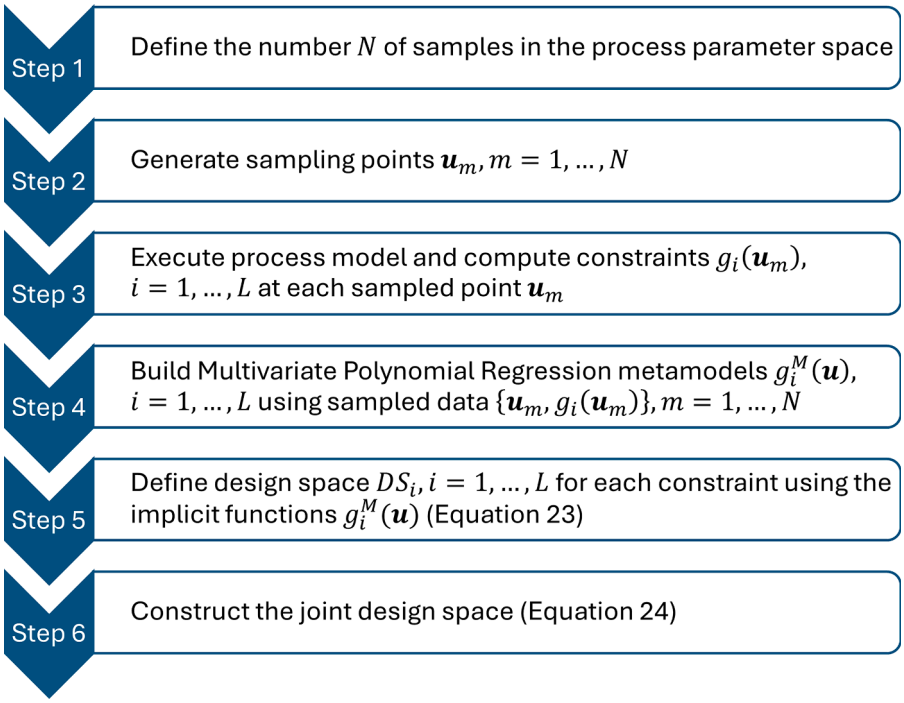


Fig. 2. Flowchart detailing the steps of the R-DS methodology.

Table 1
Mean values of the kinetic parameters.

Parameter	Value
E_1 (J/mol)	2500.2
E_2 (J/mol)	5000.1
k^*_1 (min^{-1})	0.0666
k^*_2 (min^{-1})	10,334.5

1. metamodels are built for individual constraints and not for the joint DS. In this case the number of required model runs N is smaller due to the individual constraints generally being smoother (both in terms of the geometrical shapes and the respective algebraic functions describing them) than more complex shapes of the joint DS for which metamodels were built previously;
2. the identified joint DS is represented in an explicit analytical form as a single algebraic inequality.

We note that other types of functional approximations, such as those based on Fourier series or other types of function expansions, can also be used instead of Multivariate Polynomial Regression on stage (i).

4.2. Batch reactor system

To illustrate the R-DS methodology on a practical example, we consider a batch reactor problem adapted from Samsatli et al. (1999), where the following two-stage chemical reaction takes place:

$$2A \overset{k_1}{\rightarrow} B \overset{k_2}{\rightarrow} C.$$

The reactor is a fixed-volume vessel with the volume $V = 1\ \mathrm{m}^3$ that initially contains a solution of reactant A at a concentration 2000 mol $/m^3$ B represents the desired product, while C denotes an undesired byproduct.

The mathematical model of the reactor is formulated as the following system of differential-algebraic equations (DAEs):

$$\frac{dC_A}{d\tau} = t\cdot\left(-2k_1\cdot {C_A}^2\right),$$

$$\frac{dC_B}{d\tau} = t\cdot\left(k_1\cdot {C_A}^2 - k_2\cdot C_B\right),$$

$$\frac{dC_C}{d\tau} = t\cdot(k_2\cdot C_B),$$

$$k_j = k^*_j e^{-E_j/RT},\ j = 1, 2 \tag{25}$$

with the initial conditions $C_A(\tau = 0) = C_{A0} = 2000\frac{\mathrm{mol}}{m^3}$, $C_B(\tau = 0)$, $C_C(\tau = 0) = 0$, where τ is the normalized time, $\tau \in [0, 1]$, t is the processing time (batch duration), T is the reactor temperature (assumed to be constant as reactor operation is isothermal), and k_j are the reaction rate constants.

The mean values of reaction kinetic parameters (activation energies E_j and pre-exponential factors k^*_j) are given in Table 1.

For the model with uncertain parameters, the reaction kinetic parameters $\boldsymbol{\theta}$ follow a normal distribution with mean values specified in Table 1 and a coefficient of variation (CV). In studies by Kucherenko et al. (2020, 2025), a CV value of 1 % was used. Kucherenko et al. (2024) demonstrated through GSA that only the parameter k^*_1 significantly influences the model output. Consequently, to define the DS, the three non-influential parameters (E_1, E_2, k^*_2) can be fixed at their nominal values. It was shown to reduce model complexity and computation time without affecting the results. To get a better agreement between the deterministic and probabilistic DSs, we set CV to 0.1 %. This low CV value indicates minimal variability of the uncertain parameter k^*_1. Value of p^* in Eq. (3) was taken to be equal 0.9.

The two controlled variables (temperature T and processing time t) can be adjusted independently to satisfy two CQAs: the product's $Purity = g_1(\boldsymbol{u})$ and operational $Profit = g_2(\boldsymbol{u})$ with $\boldsymbol{u} = (t, T)$ should be above their lower thresholds at $\tau = 1$: $Purity(\tau = 1) \geq P^* = 80\%$, $Profit(\tau = 1) \geq Pr^* = \$128/min$, where *Purity* and *Profit* are defined as

$$Purity\ (\tau) = \frac{C_B(\tau)}{C_A(\tau) + C_B(\tau) + C_C(\tau)}, \tag{26}$$

$$Profit(\tau)\ = \frac{(100\cdot C_B(\tau) - 20\cdot C_{A0})\cdot V}{t + 30}. \tag{27}$$

Further we will omit their dependence on τ by assuming their final values at $\tau = 1$. The ranges of the controlled variables are $T \in [250, 300]$ K, and $t \in [250, 350]$ *mins*.

To construct an analytical description of the process DS, where both constraints (Eq. (26)) and (Eqs. (26) and (27)) are simultaneously satisfied, we apply the methodology outlined in Section 4.1. First, we need to define separately the DSs for *Purity* and *Profit*:

$$DS_{(\boldsymbol{u}:Purity(\boldsymbol{u}))} = (\boldsymbol{u}|\ Purity \geq\ P^*\ ), \tag{28}$$

$$DS_{(\boldsymbol{u}:Profit(\boldsymbol{u}))} = (\boldsymbol{u}|\ Profit\ \geq\ Pr^*\ ). \tag{29}$$

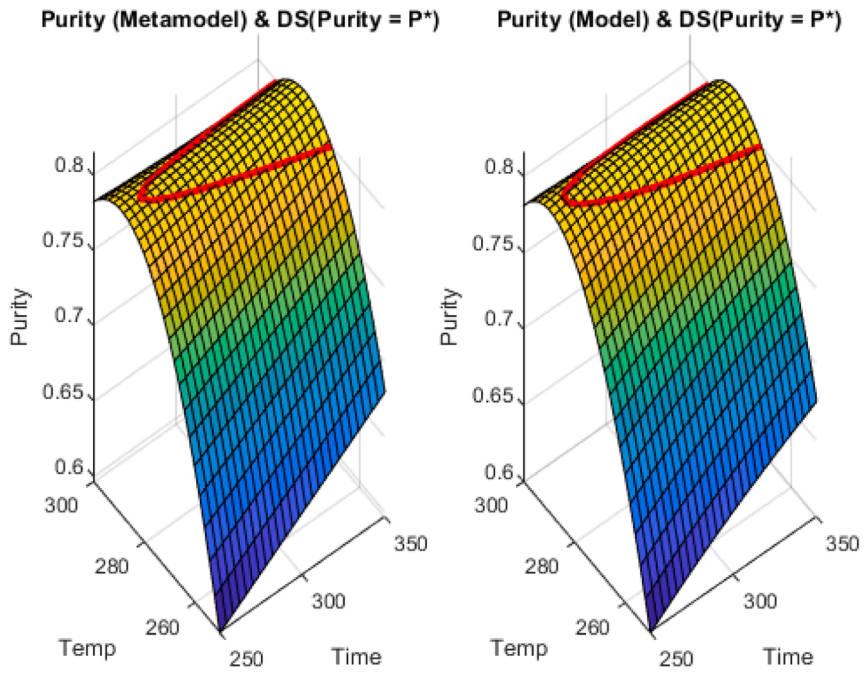
For this purpose, we build Multivariate Polynomial Regression metamodels $g^M_i,\ i = 1,\ 2$ for each of the two constraints $g_i(\boldsymbol{u}),\ i = 1,\ 2$. N points are sampled in the space of the controlled variables $\boldsymbol{u} = (t, T)$. For each sampled point the DAE system (24) is solved, and the resulting input-output table is used to build metamodels.

In this work we used the SobolGSA software (Kucherenko and Zaccheus, 2024) to build Multivariate Polynomial Models. There are two different methods implemented in SobolGSA: the Random Sampling-High Dimensional Model Representation (RS-HDMR) (Li et al., 2002; Zuniga et al., 2013) and the Bayesian Sparce Polynomial Chaos Expansion (BSPCE) method (Shao et al., 2017). These methods belong to a class of methods known as polynomial chaos expansion (PCE). PCE is a metamodeling technique in which the output of a model is presented as a series of polynomial functions of the input variables, with each polynomial term weighted by coefficients that reflect the contribution of the corresponding input variable to the output variability. Usually only a few terms are really relevant in the PCE structure. The BSPCE method makes use of sparse PCE, in which the selection of the proposed PCE structure is based on a Bayesian approach using the Kashyap information criterion for model selection (Shao et al., 2017). This enables the construction of accurate metamodels using only a limited number of model runs. Sampling was performed using QMC sampling based on Sobol' sequences (Sobol et al., 2011). QMC sampling gives a much better way of arranging N points in n–dimensions and a faster convergence rate than random sampling (Kucherenko et al., 2015).

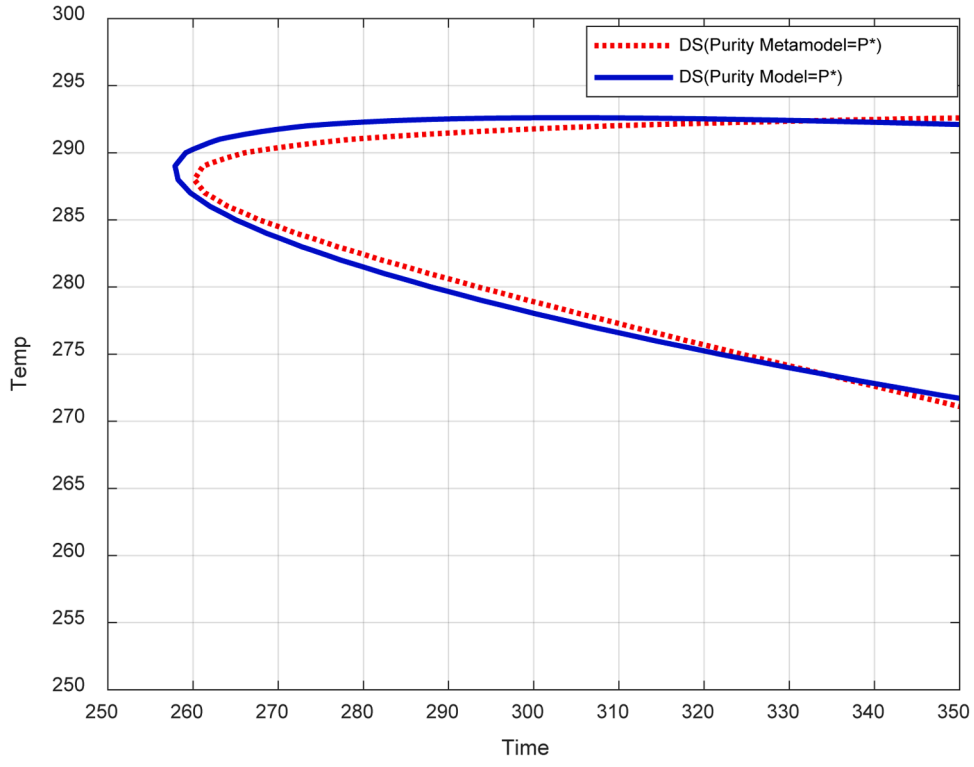
The application of BSPCE showed that, for both constraints, the optimal PCE approximation is a second-order polynomial in T, a first-order polynomial in t, and a first-order cross term between T and t:

$$g^M_i = a^{[i]}_{00} + a^{[i]}_{10}T + a^{[i]}_{20}T^2 + a^{[i]}_{01}t + a^{[i]}_{11}t\ T. \tag{30}$$

The coefficients $\{a^{[i]}_{kl}\}$ are found by fitting the polynomial function (29) to the input-output training data. Tests showed that $N = 32$ model runs were sufficient to build metamodels for $Purity(t, T)$ and $Profit(t, T)$ with high accuracy: in both cases the coefficient of determination R^2 was very close to 1. However, we opted to use $N = 64$ in all tests, as this value provided a DS volume closer to the average across different model run



(a) (b)



(c)

Fig. 3. Comparison of model constraints and their metamodels for CQA *Purity*(t,T): (a) metamodel; (b) exact model. Curves in red show DS boundaries corresponding to the *Purity* threshold value P^*. (c) DS boundaries *Purity*$(t,T) = P^*$ for metamodel (red broken line) and exact model (blue solid line).

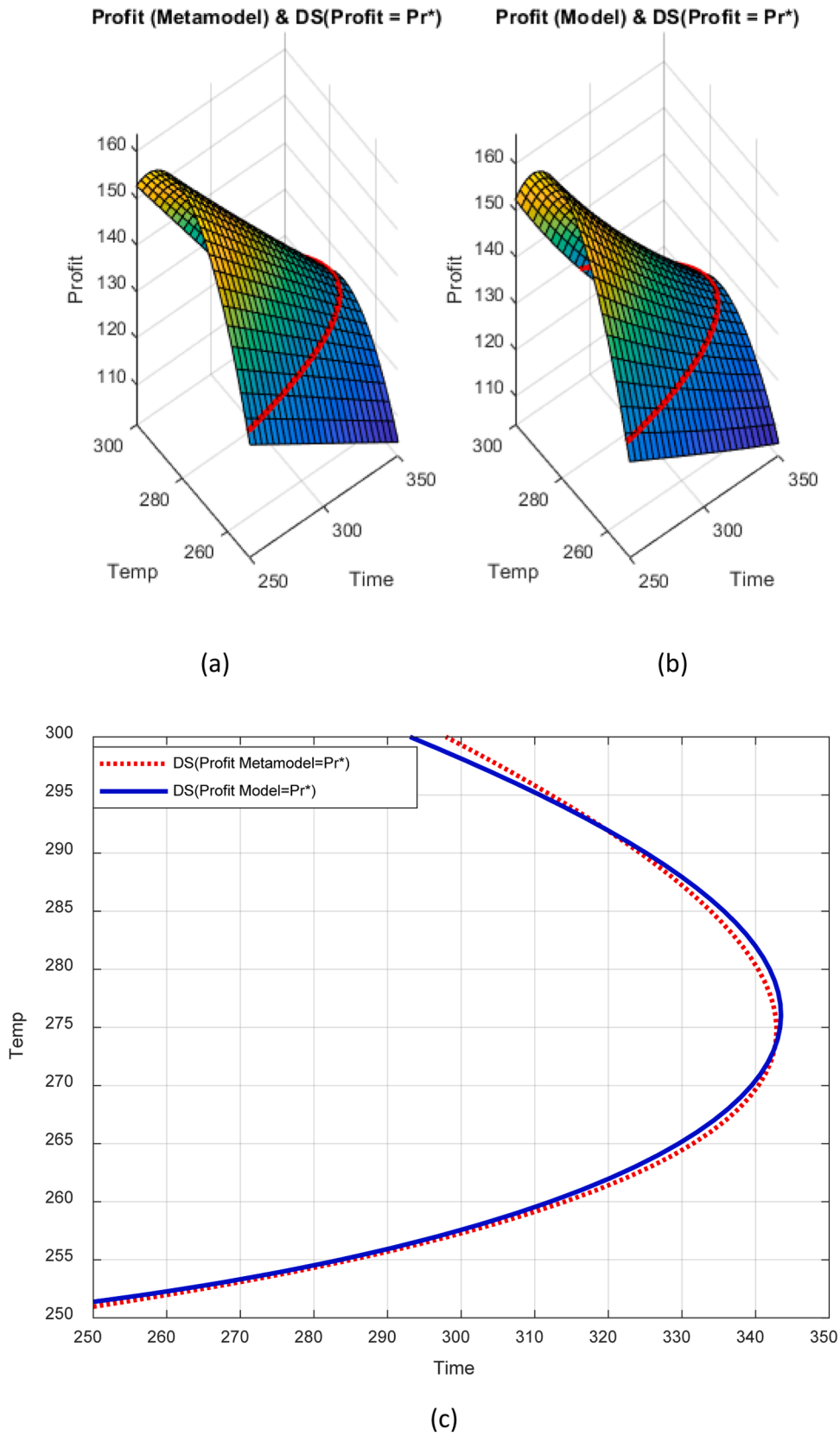


Fig. 4. Comparison of model constraints and their metamodels for CQA *Profit*(t,T): (a) metamodel; (b) exact model. Curves in red show DS boundaries corresponding to the *Profit* threshold value *Pr**. (c) DS boundaries *Profit*$(t,T) = Pr^*$ for metamodel (red broken line) and exact model (blue solid line).

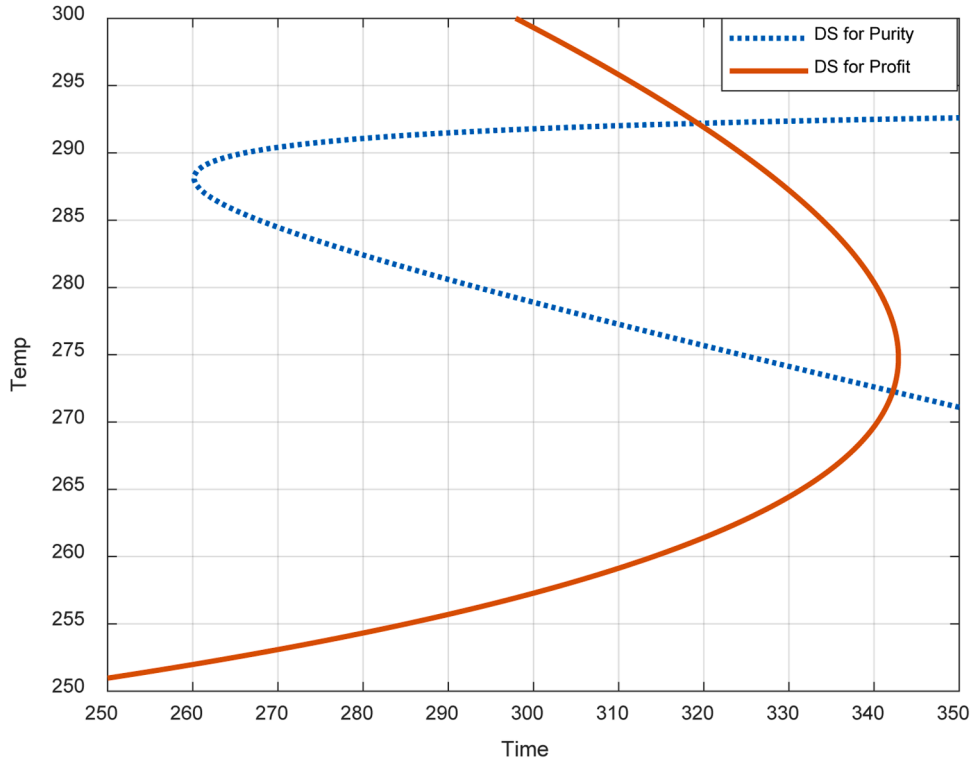


Fig. 5. DS boundaries for *Purity* (light blue) and *Profit* (orange).

counts. This is discussed further in Section 4.3.

Comparisons of metamodels for $Purity(t,T)$ and $\text{Profit}(t,T)$ with the full model solutions show a very good agreement between surfaces (Figs. 3 and 4). DS boundaries for each of the two constraints (red curves) defined by $DS_{(\boldsymbol{u}:Purity(\boldsymbol{u}))} = (\ \boldsymbol{u},\ Purity = P^*)$, $DS_{(\boldsymbol{u}:Profit(\boldsymbol{u}))} = (\boldsymbol{u},\ Profit = Pr^*)$ also agree very well (Figs. 3c and 4c).

The fact that the metamodels of process CQAs are given explicitly in the analytical forms (30) enables the application of the theory of *R*-functions for the identification of the joint DS from the equations for individual DSs. Following the framework presented in Section 4.1, we define implicit functions

$$\phi_1(T,t) = \ -P^* + \ a_{00}^{[1]} + a_{10}^{[1]}T + a_{20}^{[1]}T^2 + a_{01}^{[1]}t + a_{11}^{[1]}t\ T, \tag{31}$$

$$\phi_2(T,t) = \ -Pr^* + \ a_{00}^{[2]} + a_{10}^{[2]}T + a_{20}^{[2]}T^2 + a_{01}^{[2]}t + a_{11}^{[2]}t\ T, \tag{32}$$

which change sign at the boundaries of the corresponding individual DSs.

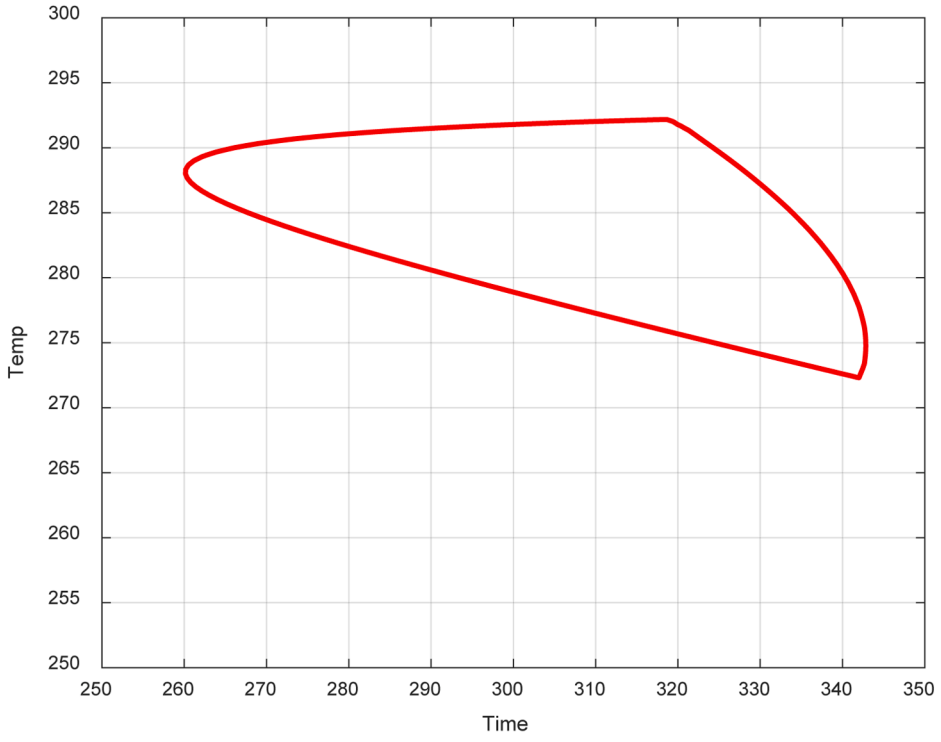
The best-fit metamodels are shown in Eqs. (32)-(33) with exact polynomial coefficients. Fig. 5 shows the DS curves for $Purity(t,T)$ and $Profit(t,T)$ defined by (33), (34).

$$\phi_1(T,t) = -P^* + \frac{1492155457490699 * T}{18014398509481984} + \frac{6353038647553177 * t}{1152921504606846976} - \frac{5532151810052315 * T * t}{295147905179352825856} - \frac{2496177142651283 * T^2}{18446744073709551616} - \frac{8909117195501237}{703687441776640}. \tag{33}$$

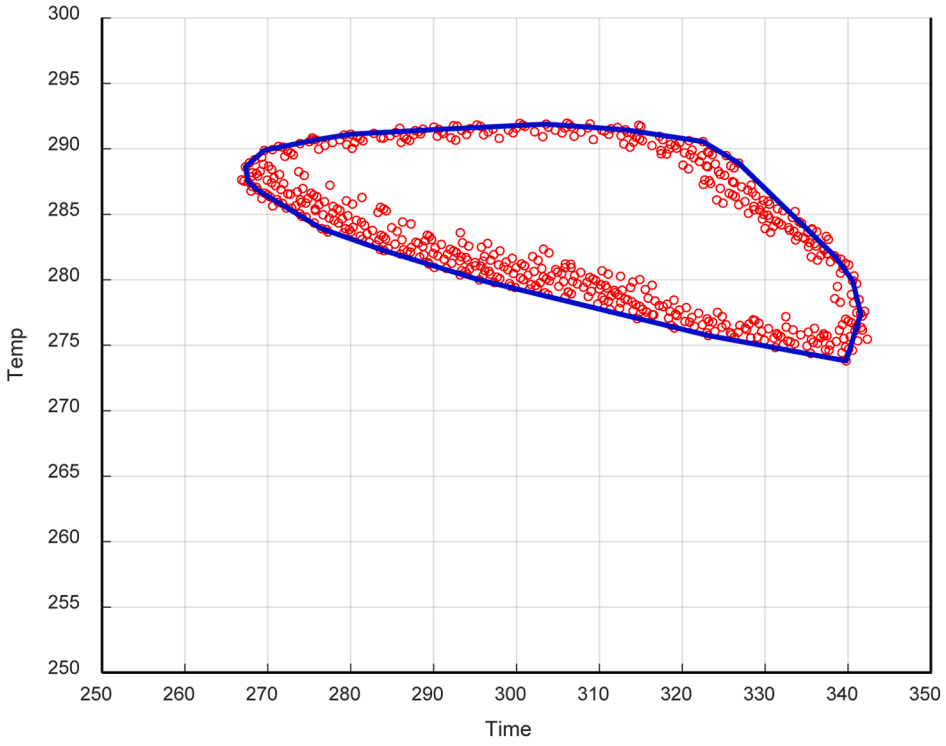
$$\phi_2(T,t) = -Pr^* + \frac{6181541458134125 * \mathrm{T}}{281474976710656} + \frac{1444357484257337 * t}{1125899906842624} - \frac{6903876884140963 * \mathrm{T} * t}{1152921504606846976} \frac{(5220857310952963 * T^2)}{144115188075855872} - \frac{6981627641092835}{2199023255552}. \tag{34}$$

The critical values of P^*, Pr^* are given above. Applying R-conjunction $f_\wedge(\phi_1,\phi_2)$ we obtain the joint DS in an analytical form:

$$f_\wedge(T,t) = \frac{39711080778074699 * T}{36028797018963968} + \frac{1485375102527066265 * t}{2305843009213693952} - \frac{1772924634150138843 T * T}{59029581035870565 1712} - (8809301652650171065 * T * t - 1885016352296908686080 * t - 3228684270494460321 7920 * T + 5326188469274623 8480 * T^2 + \frac{(46666072257026189437 17376)^2 \left(\frac{1}{2}\right)}{295147905179352825 8560} - \frac{(670765912944630547 * T^2)}{36893488147419103232} - \frac{2243029962345208437}{1407374883553280}. \tag{35}$$



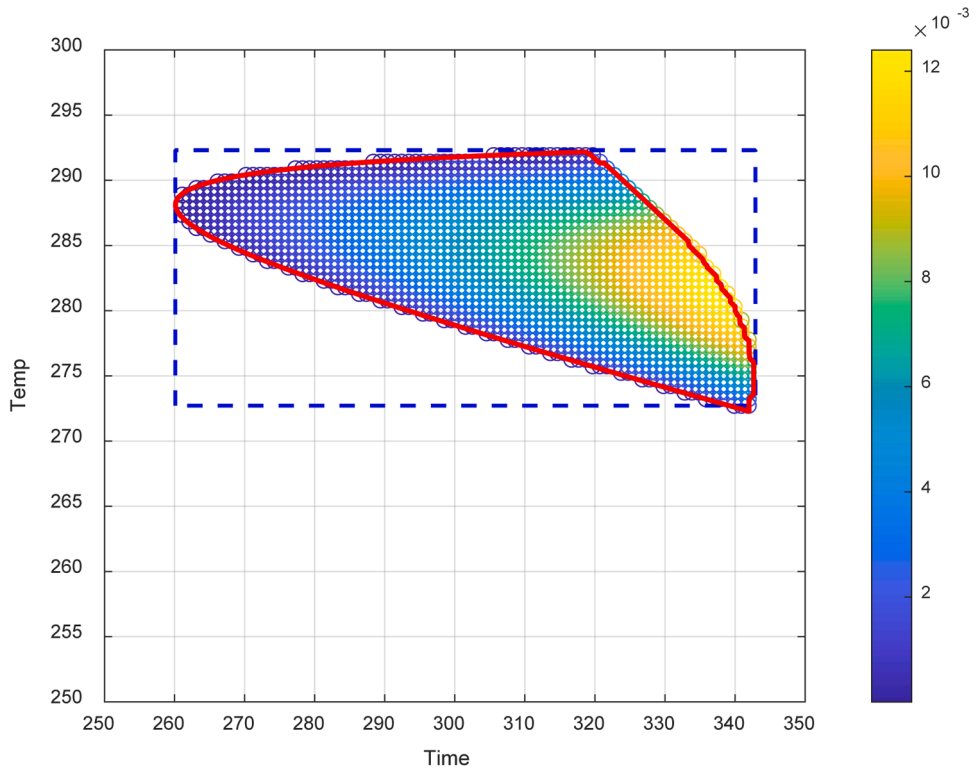
(a)



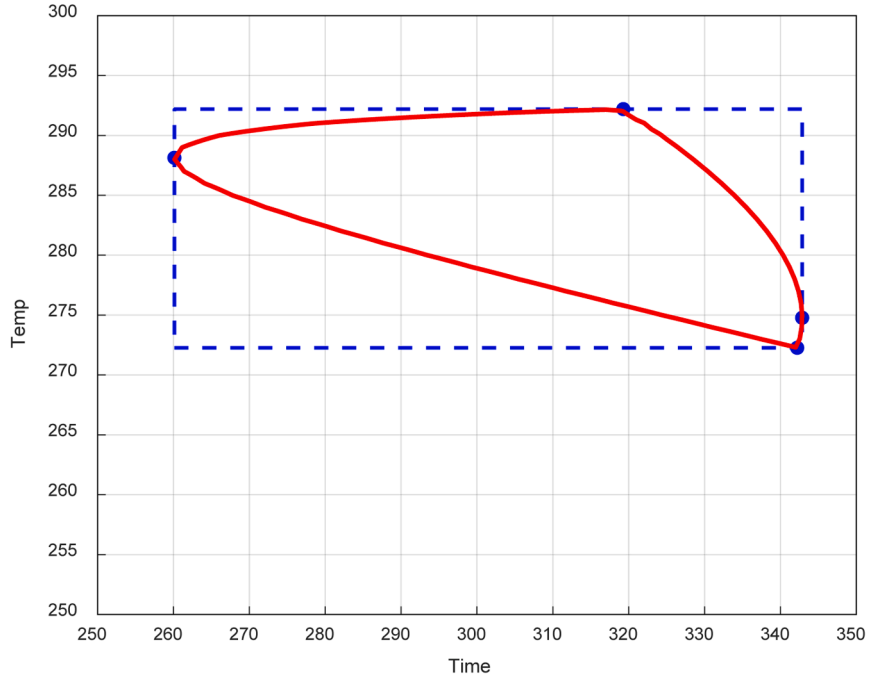
(b)

Fig. 6. (a) Joint DS analytically defined as $f_{\wedge}(T,t) = 0$ (Eq. (34)) for the deterministic model of the batch reactor system; (b) probabilistic DS identified using the acceptance–rejection sampling approach (Kucherenko et al. 2019).

The boundary of the joint DS defined by Eq. (34) is presented in Fig. 6a. Comparing the DS derived using the R-function for the deterministic model (Fig. 6a) with the DS obtained through the acceptance-rejection sampling method (Kucherenko et al., 2020) for the probabilistic model (Fig. 6b) shows a qualitative similarity in the shapes of both DSs. However, differences remain, as the DS volumes V_{DS} (see Section 4.3 for details) differ and the exact locations of the two DSs do not completely align due to the differences in deterministic and



(a)



(b)

Fig. 7. Axis-aligned bounding box (blue dashed line) and the enclosed DS domain (red curve) obtained using: (a) the Monte Carlo sampling approach, and (b) solutions of the constrained optimization problem (35) (blue points). In (a), a heatmap illustrates the function values $f_{\wedge}(T,t) \geq 0$ within the DS.

probabilistic problem settings.

The R-DS method required approximately 1 s of the CPU time on a strandard PC, whereas the acceptance-rejection sampling approach took 49 s. However, The probabilistic problem setup includes an additional loop over the space of uncertain parameters, with the loop running for $j = 1 : N_\theta$, where $N_\theta = 32$ in this case. When adjusting the CPU time for the acceptance-rejection sampling approach to $N_\theta = 1$, the estimated time reduces to 29 s. Consequently, the R-DS method is approximately 30 times faster than the acceptance-rejection sampling approach.

We evaluated the accuracy of the R-DS identifier method for identifying the DS using its volume, $V_{DS,}$ as the accuracy criterion. A series of tests were conducted with the number of model runs N ranging from 32 to 256. The results showed that the average V_{DS} across different runs was 880 units (see Section 4.3 for details), with a coefficient of variation CV of 1%. For all subsequent tests, $N = 64$ model runs were used. In comparison, the acceptance-rejection method (Kucherenko et al., 2019) applied to a probabilistic model required $N = 1730$ model runs to achieve converged results.

In the acceptance–rejection method the solution for the DS is given as a set of "accepted points". The MATLAB (The MathWorks, 2024) "convhull" function was used to compute the convex hull of the accepted points and plot the DS boundary (Fig. 6b). However, there are no comparable easy to apply tools for accurately representing the DS shape when the points form a non-convex set in a multidimensional case. In contrast, the R-DS identifier method can handle any shape, including non-convex and disjoint regions.

The proposed method is straightforward to implement using symbolic computation tools available in software platforms such as MATLAB, Maple, Mathematica, or Python. These tools facilitate the derivation of implicit functions $\phi_i(u)$ for individual DSs and the joint DS $f_\wedge(u)$. In this study, MATLAB was used for all symbolic computations and plotting. However, the R-DS methodology can also be applied using standard numerical methods without symbolic computations. In this case, the implicit functions $\phi_i(u)$ and the joint DS $f_\wedge(u)$ can be obtained through grid-based or sampling approaches.

We observe that, although other methods—such as the one introduced by Zhao and Chen (2018), which relies on the cylindrical algebraic decomposition method (referred to as the CAD method for brevity)—can analytically represent DSs or flexibility regions, the R-DS method provides several significant advantages:

1. It represents the DS using a single implicit function, unlike the CAD method, which defines the DS or flexibility regions as multiple disjoint feasible regions through quantifier elimination.
2. The R-DS method is not restricted to polynomial functions, whereas the CAD method is inherently limited to polynomial functions. Although CAD can handle non-polynomial systems, it does so by employing local polynomial approximations, which introduce approximation errors.
3. The application of R-functions is straightforward, computationally efficient, and scalable, as it can be performed with or without symbolic computation and easily handles high-dimensional problems. In contrast, the CAD method involves computationally intensive symbolic operations, with the computational burden growing exponentially with the system's size, making it suitable primarily for small-scale problems.

4.3. Exploration of the identified DS

Defining the DS analytically using R-functions has additional benefits, such as simplifying operations like "boxing" (determining a tight axis-aligned bounding box), identifying points within the feasible domain, and calculating the domain's volume. Monte Carlo sampling is one of the simplest approaches for achieving these tasks.

Steps to determine an axis-aligned bounding box:

1. Define the implicit function $f_\wedge$ that characterizes the DS, such as the one given by Eq. (35).
2. Generate a set of points within a broader domain that is guaranteed to encompass the entire DS.
3. Calculate $f_\wedge$ at each sample point and retain those where $f_\wedge \geq 0$.
4. Identify the minimum and maximum values of each parameter u_k among the feasible points. These values approximate the lower and upper bounds of a tightly fitting hyperrectangular box.

Fig. 7a illustrates the application of this algorithm. The volume of the DS, V_{DS} is calculated by multiplying the fraction of points accepted in Step 3 by the total volume V of the sampled parameter domain (see Step 2): $V_{DS} = 880$ (K * mins), while $V = 50$ (K) * 100 (mins) = 5000 (K * mins). For comparison, in the acceptance–rejection method, $V_{DS} = 790$ (K * mins).

For low dimensions, Step 2 can be executed using sampling on a regular grid. In higher dimensions, Monte Carlo or Quasi Monte Carlo sampling is preferred. Note that the density of sample points influences the accuracy of determining the bounding box.

Another approach to define an axis-aligned bounding box is to solve a constrained optimization problem with nonlinear constraints:

$$\begin{aligned} &\min_{\boldsymbol{x}} \boldsymbol{\alpha}^\top \boldsymbol{x}, \\ &\text{s.t. } \| \boldsymbol{\alpha} \| = 1, \\ &f_\wedge(\boldsymbol{x}) = 0. \end{aligned} \tag{36}$$

Minimizing $\boldsymbol{\alpha}^\top \boldsymbol{x}$ represents a linear projection of $\boldsymbol{x}$ along the direction of $\boldsymbol{\alpha}$. By choosing different vectors $\boldsymbol{\alpha}$ and initial guesses for the optimization problem (36), one can determine the minimum value of $\boldsymbol{x}$ on the surface defined by $f_\wedge(\boldsymbol{x}) = 0$ along direction $\boldsymbol{\alpha}$.

For the test case considered, where $x = (t, T)$, the method of Lagrange multipliers was applied to identify all critical points (the minimum and maximum t and T) on the DS boundary. The critical points were identified using two direction vectors $\boldsymbol{\alpha}_1 = (0, 1)$ and $\boldsymbol{\alpha}_2 = (1, 0)$, and were subsequently used to construct the axis-aligned bounding box shown in Fig. 7b.

5. Conclusions

V.L. Rvachev's R-functions are incredibly useful for representing geometric shapes and operations. Their ability to combine implicit functions to represent complex geometric shapes through algebraic expressions makes them invaluable in fields ranging from computer graphics to engineering analysis. We showed that they also enable the solution of geometric problems such as the identification of Design Spaces (DS) through algebraic manipulation.

Many manufacturing processes, especially in pharmaceutical production, are governed by highly complex mathematical models, making it computationally challenging to define the DS using existing methods. This challenge is particularly pronounced in multidimensional cases where the DS shape can be non-convex, as no efficient methods currently exist to accurately represent such shapes.

We have developed a new methodology for efficient DS identification by using R-functions to describe the DS in the form of analytical inequalities involving algebraic functions over polynomials. The R-DS identification method can accommodate any shape, including non-convex and disjoint regions. Importantly, the R-DS methodology is not limited to constraint functions that are defined as or approximated by polynomials.

A major strength of the R-DS methodology is its broad applicability, enabling it to address "implicit" constraints. In practical applications, like those demonstrated in this work, evaluating such constraints typically involves solving the process model, as explicit analytical expressions may be unavailable for a given set of parameter values. In these scenarios, approximating constraints using polynomial models offers a

practical and efficient solution.

This approach can help effectively convey high-dimensional DS shapes and incorporate this information into formal documentation required for the approval of process DSs by regulatory authorities. Indeed, the application of the R-DS methodology results in a single algebraic inequality representing the whole DS as opposed to $N_P = \frac{n(n-1)}{2} 3^{n-2}$ two-dimensional cross-section plots (the number of which grows exponentially with the number of process parameters) that are typically conveyed to regulatory authorities under the current approval process.

CRediT authorship contribution statement

S. Kucherenko: Writing – original draft, Visualization, Supervision, Software, Methodology, Conceptualization. **N. Shah:** Writing – review & editing, Resources, Project administration, Funding acquisition, Conceptualization. **O.V. Klymenko:** Writing – original draft, Software, Methodology, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no competing financial interests or personal relationships that could have influenced the work reported in this manuscript.

Acknowledgements

S.K and N.S. acknowledge the financial support of Eli Lilly and Company, the EPSRC Grant EP/T005556/1 and EPSRC grant EP/V042432/1. The authors would like to thank two anonymous reviewers for their insightful and constructive comments.

Appendix

Test cases in 3D Paraboloids and cylinders
Consider the following inequalities that define:

1. the area under the paraboloid $z = 0.6\left(1 - x^2 - y^2\right)$:

$$f_1(x,y,z) = -z + 0.6\left(1 - x^2 - y^2\right) \geq 0 \tag{A.1}$$

2. the area above the paraboloid $z = -0.6\left(1 - x^2 - y^2\right)$:

$$f_2(x,y,z) = z + 0.6\left(1 - x^2 - y^2\right) \geq 0 \tag{A.2}$$

3. the area inside of the cylinder of radius 0.5 centered around the z-axis:

$$f_3(x,y,z) = 0.5^2 - x^2 - y^2 \geq 0 \tag{A.3}$$

4. the area inside of the cylinder of radius 0.3 centered around the z-axis:

$$f_4(x,y,z) = 0.3^2 - x^2 - y^2 \geq 0. \tag{A.4}$$

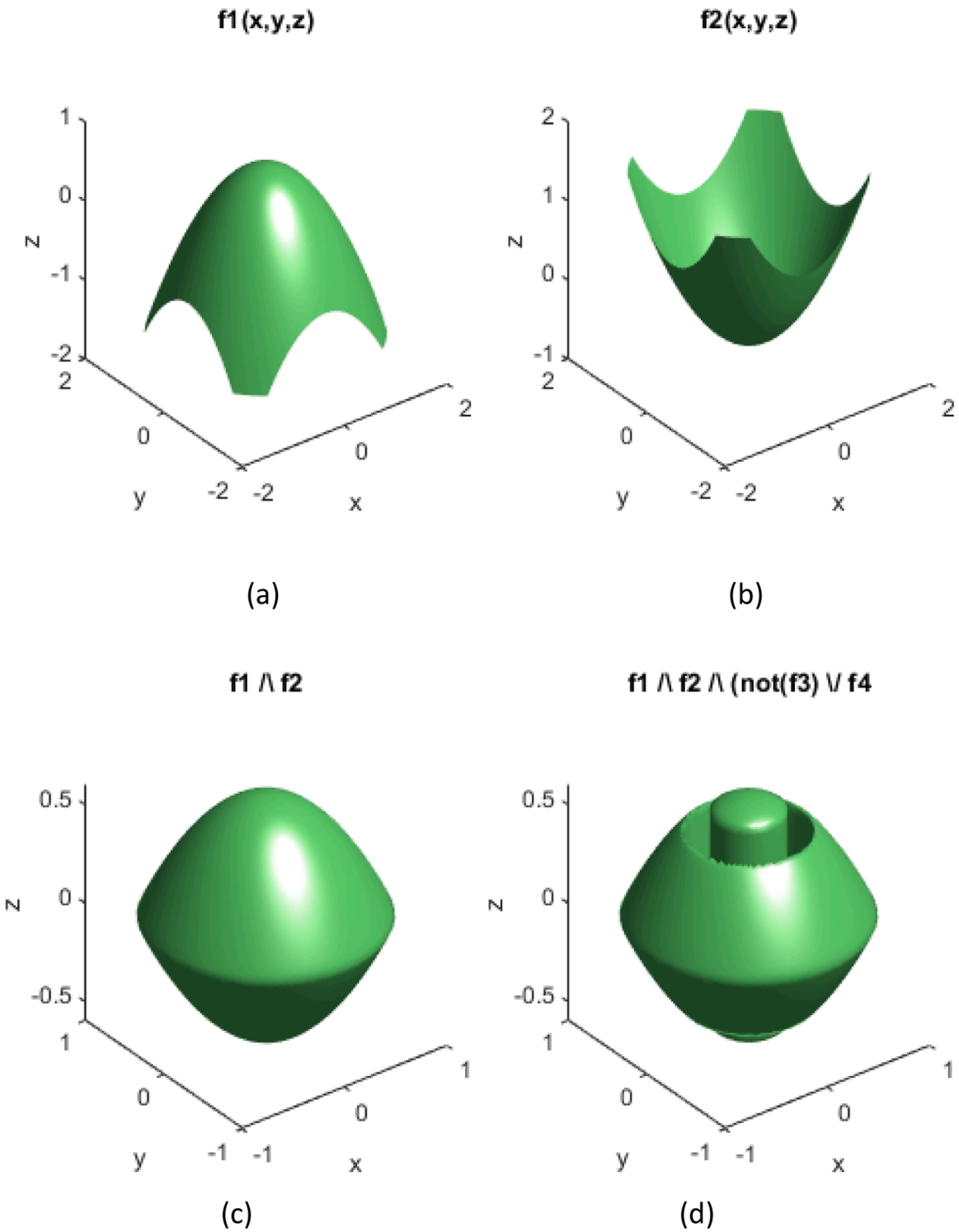


Fig. A.1. $f_1(x,y,z) = 0$ (a); $f_2(x,y,z) = 0$ (b); R-conjunction of $f_1(x,y,z)$ and $f_2(x,y,z)$: $f_1 \wedge_1 f_2 = 0$ (c); $f_1 \wedge_1 f_2 \wedge_1 (\overline{f_3} \vee_1 f_4) = 0$ (d).

Using these geometrical primitives, we can construct domains either using simple operations as above, such as an R-conjunction of $f_1(x,y,z)$ and $f_2(x,y,z)$ (Fig. A.1c), as well as their combinations to describe complex non-simply connected domains like the one shown in Fig. A.1d The latter is described by the R-function $f_1 \wedge_1 f_2 \wedge_1 (\overline{f_3} \vee_1 f_4) \geq 0$ and represents the domain shown in Fig. A.1c with an annular cylindrical cut-out.

Data availability

No data was used for the research described in the article.

References

Castagnoli, C., Yahyah, M., Cimarosti, Z., Peterson, J.J., 2010. Application of quality by design principles for the definition of a robust crystallization process for casopitant mesylate. Org. Process Res. Dev. 14, 1407–1419.

Figueroa, I., Vaidyaraman, S., Viswanath, S., 2013. Model-based scale-up and design space determination for a batch reactive distillation with a Dean–Stark trap. Org. Process Res. Dev. 17 (10), 1300–1310.

García-Muñoz, S., Luciani, C.V., Vaidyaraman, S., Seibert, K.D., 2015. Definition of design spaces using mechanistic models and geometric projections of probability maps. Org. Process Res. Dev. 19 (8), 1012–1023.

Grossmann, I.E., Morari, M., 1983. Operability, Resiliency, and Flexibility: Process Design Objectives for a Changing World. Carnegie Mellon University.

Guidance for Industry, Q8(R2) Pharmaceutical Development, U.S. Department of Health and Human Services, 2009.

Halemane, K.P., Grossmann, I.E., 1983. Optimal process design under uncertainty. AlChE J. 29, 425–433.

ICH, 2009. Q8(R2) Pharmaceutical Development. The International Council for Harmonisation of Technical Requirements for Pharmaceuticals for Human Use. Available at https://admin.ich.org/home.

Kucherenko, S., Albrecht, D., Saltelli, A., 2015. Exploring multi-dimensional spaces: a comparison of Latin Hypercube and quasi Monte Carlo sampling techniques. arXiv: 1505.02350.

Kucherenko, S., Giamalakis, D., Shah, N., García-Muñoz, S., 2020. Computationally efficient identification of probabilistic design spaces through application of metamodeling and adaptive sampling. Comput. Chem. Eng. 132, 106608.

Kucherenko, S., Sopittakamol, N., Shah, N., 2024. Design space identification of a coupled two-stage batch reactor system. Comput. Chem. Eng. 187, 108727.

Kucherenko, S., Giamalakis, D., Shah, N., 2025. Application of global sensitivity analysis for identification of probabilistic design spaces. Int J Uncertain Quantif 15 (1).

Kucherenko, S., Zaccheus, O., 2024. SobolGSA Software. Available at: https://www.imperial.ac.uk/process-systems-engineering/research/free-software/sobolgsa-software/.

Kusumo, K.P., Gomoescu, L., Paulen, R., García Muñoz, S., Pantelides, C.C., Shah, N., Chachuat, B., 2019. Bayesian approach to probabilistic design space characterization: a nested sampling strategy. Ind. Eng. Chem. Res. 59, 2396–2408.

Li, G., Wang, S.W., Rabitz, H., 2002. Practical approaches to construct RS-HDMR component functions. J. Phys. Chem. A 106 (37), 8721–8733.

Rogers, A., Ierapetritou, M., 2015. Feasibility and flexibility analysis of black-box processes part 1: surrogate-based feasibility analysis. Chem. Eng. Sci. 137, 986–1004.

Rvachev, V.L., 1982. Theory of R-functions and some applications. Naukova Dumka (In Russian).

Saltelli, A., Tarantola, S., Campolongo, F., Ratto, M., 2004. Sensitivity Analysis in Practice: a Guide to Assessing Scientific Models. Wiley, New York.

Samsatli, N.J., Papageorgiou, L.G., Shah, N., 1999. Batch process design and operation using operational envelopes. Comput. Chem. Eng. 23, 887–890.

Shao, Q., Younes, A., Fahs, M., Mara, T.A., 2017. Bayesian sparse polynomial chaos expansion for global sensitivity analysis. Comput. Methods Appl. Mech. Eng. 318, 474–496.

Shapiro, V., 1991. Theory of R-functions and applications: A primer. Cornell University.

Sobol, I.M., Kucherenko, S.S., 2005. Global sensitivity indices for nonlinear mathematical models. Wilmott 1, 56–61.

Sobol, I.M., Asotsky, D., Kreinin, A., Kucherenko, S., 2011. Construction and Comparison of High-Dimensional Sobol' Generators. Wilmott, pp. 64–79.

Swaney, R.E., Grossmann, I.E., 1985a. An index for operational flexibility in chemical process design. Part I: formulation and theory. AlChE J. 31 (4), 621–630.

Swaney, R.E., Grossmann, I.E., 1985b. An index for operational flexibility in chemical process design. Part II: computational algorithms. AlChE J. 31 (4), 631–641.

Tian, H., Jagana, J.S., Zhang, Q., Ierapetritou, M., 2024. Feasibility/flexibility-based optimization for process design and operations. Comput. Chem. Eng. 180, 108461.

The MathWorks, Inc., MATLAB Release R 2024. Natick, Massachusetts, United States. Available at: https://uk.mathworks.com/products/matlab.html.

Wang, Z., Ierapetritou, M., 2018. Global sensitivity, feasibility, and flexibility analysis of continuous pharmaceutical manufacturing processes. Computer-Aided Chemical Engineering 41, 189–213.

Zhao, F., Chen, X., 2018. Analytical and triangular solutions to operational flexibility analysis using quantifier elimination. AlChE J. 64 (11), 3894–3911.

Zhao, F., Ochoa, M.P., Grossmann, I.E., García-Muñoz, S., Stamatis, S.D., 2022. Novel formulations of flexibility index and design centering for design space definition. Comput. Chem. Eng. 166, 107969.

Zuniga, M.M., Kucherenko, S., Shah, N., 2013. Metamodelling with independent and dependent inputs. Comput. Phys. Commun. 184, 1570–1580.