Mixed Integer Nonlinear Optimisation for A Portfolio of Products with Heuristics Algorithm

Muhammad Robith Hadhromi (02529728)

Supervisor: Dr. Antonio Del Rio Chanona

Submitted to the Faculty of Engineering

Department of Chemical Engineering

Imperial College London

September 2024

Mixed Integer Nonlinear Optimisation for A Portfolio of Products with Heuristics Algorithm

Muhammad Robith Hadhromi

Department of Chemical Engineering, Imperial College London, London, SW7 2AZ, United Kingdom

Abstract

This paper explores the development and application of a Mixed Integer Nonlinear Programming (MINLP) model to optimise a product portfolio, specifically focusing on detergent laundry products. The study addresses the complexity of product formulation and production, involving multiple ingredients and processes, by leveraging heuristic algorithms and open-source software tools such as Pyomo. The research is grounded in the need for efficient resource allocation in product development, which is crucial for maintaining competitive advantage in the fast-moving consumer goods (FMCG) industry. The methodology integrates statistical approaches with mathematical modelling to solve the pooling and blending problems inherent in the product formulation process. The results demonstrate that the proposed optimisation is able to solve the complexity of the problem that covers various models and constraints, such as the Stain Removal Index (SRI). The study also suggests further refinement of the model by incorporating advanced algorithmic strategies to overcome challenges like local minima and tightened constraints, potentially enhancing the robustness of the optimisation outcomes.

Keywords: MINLP, Optimisation, Product Portfolio, Laundry Products, P&G

1. Introduction

The fastened effect on incremental economic factors pushes companies to keep updated with accommodating customer needs. This includes the requirement for product development or new product research that is always well-related to considering cost, complexity, and market sales [1]. For companies, expanding their product portfolio requires testing on resource optimisation and market analysis [2]. As part of the critical points, the research and development (R&D) stage should be considered for new product release decisions. Nevertheless, the budget for the R&D stage is particularly significant for the initial step of product portfolio development [3].

Product portfolio development is necessary for Procter & Gamble (P&G) as a top-tier manufacturer in the fast-moving customer goods (FMCG) industry. P&G has established extensive product portfolios including ten product categories in five operating sectors [4]. Joining the FMCG industry market competition, P&G encounters not only technological updates but also product portfolio development. With these circumstances, P&G should continuously expand its products [5]. To support this action, the R&D process is a starting point with a pack of resource allocations.

The traditional approaches for product portfolio development involve problem simplification, solution hypothesis generation, physical testing and prototyping, result analysis, and reiterating the process until the objectives are found. These manual techniques are time-consuming and cost-weighted in resulting the final objective. To anticipate this recursive problem, P&G has adapted the statistical approaches and mathematical modelling optimisation to reserve the resources. One of the successful approaches is inventory occupancy, which can reduce the occupancy by up to 7% [6]. In the laundry product, P&G has developed the optimisation algorithm to fasten the product development iteration [7]. Applying this approach, the P&G Laundry team can step on the hindering aspect of the complexity in product formulation. This allows them to explore more possible product portfolios to be implemented.

With the benefit of a 10% reduction in total cost for product portfolio assessment, this approach is utterly worth further exploration with appropriate application [8]. Therefore, the product portfolio optimisation method is developing with the enhancement of the process. It includes research on the utilisation of flexible sources and the advanced methods used. In addition, P&G has opened to collaboration research which also covers technology transfer, joint R&D, and open-source software applications [9].

Problem formulation is the beginning step in adapting the mathematical optimisation algorithm for product portfolio development. The formulation evaluation will affect how the to develop the product. Most of the time, the product producer is focusing only on the product formulation. The article by Siew [10] is detailing about parameters to be considered to achieve optimised product formulation, including utilising modelling and simulation methods. By applying this approach, time and cost can be reduced significantly compared to the conventional experiment methodology.

General pharmaceutical products already have a well-designed optimisation approach as explained by [11]. The stages include problem definition and variable selection, model formulation design, result analysis, and model validation and optimisation. For generic formulated products which are composed of various ingredients mixed, [12] described a more explained framework to optimise the formulation design. It compromised the classic mathematical formulation such as the objective function, applicable constraints, and process analysis.

Specific to laundry products, [13] showed the process of synthesis and formulation development. This work explained various selections of raw materials, additives, and heuristics for product development. Due to its extensive ingredients composition and multi-stage process, the mathematical model for laundry products is composed as part of a mixed integer nonlinear program (MINLP). To solve this MINLP formulation, a bunch of researchers explore the solution method [7], [14],[15]. Tackling this optimisation problem allows companies to manage resource allocation efficiently and increase their margin and customer satisfaction [16], [17], [18].

This report aims to explore the possibility of adapting the MINLP optimisation problem for product portfolios with open-source software to get a robust and flexible solution. This report consists of several sections as details: In Section 2 background and problem definition are introduced to elaborate on the issue to tackle and describe possibilities on methodologies as well. Section 3 explains the detailed methodology used in this paper including mathematical and optimisation formulation. The result of the current research is explained in Section 4 and discussed in detail in Section 5 with suggested improvements that may be helpful for further research. This paper is concluded in Section 6 which performs the brief of the research action and recommendation.

2. Background and Problem Definition

The optimisation problem for laundry products consists of multi-layer components that influence the behaviour of optimal solution search. To untangle those issues, problem definition and scope are set for this research. Other than that, a literature review is conducted to obtain several methods that can be adapted to solve the problem.

The project scope is defined to limit the research conduct so it can focus on the specific target. This research aims to develop and construct an optimisation algorithm based on the heuristics model to solve the mixed integer nonlinear problem for product portfolios. The research scope is specifically based on P&G's current optimisation approach and statistical model, which have already been established by the P&G team.

2.1. Laundry Products Process Production

In P&G, the fabric and home care category is the leading segment with over 35% of net sales in 2023 [4]. The laundry variants are included in this category, such as liquid, powder, pods, tablets, and bars. P&G as part of FMCG producers needs to actively improve and evaluate their laundry product to come up with the other competitors [19]. To enhance the market share and stabilise customer demand, the product portfolio may help, even opening the possibility of conquering a new market segment [20].

The multi-stage process production with various materials introduced at each stage contributes to the complexity of the laundry product development. The main ingredients in laundry products are surfactant as a dirt-binding agent, filler as a release agent, and additive components such as perfumery, whiteners, enzymes, and so on [18]. Some staple process stages involved in detergent manufacturing are mixing, drying, and spraying. Combining the components and process stages for product portfolio development is challenging [21]. The basic parameters to be achieved for detergent performance during product formulation are stain removal and whiteness [22].

For this initial research, the liquid detergent is evaluated to develop the optimisation program. This is because the liquid detergent process is simpler in the process stages which is not required for the drying process. The powder and liquid detergent process scheme is depicted in Figure 1 and 2 with the difference in the drying process.

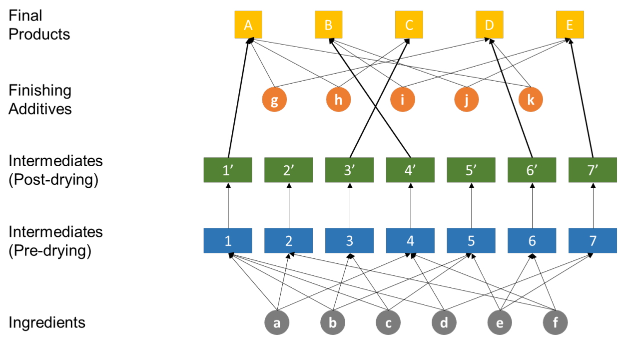

Figure 1. Powder detergent process scheme.

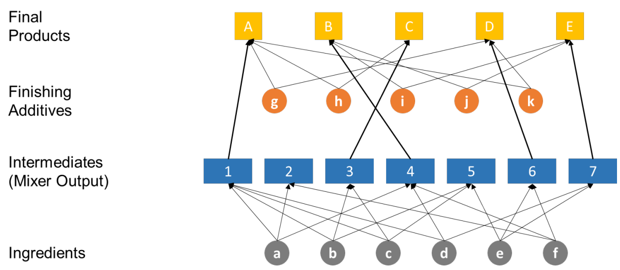

Figure 2. Liquid detergent process scheme.

2.2. *MINLP in Laundry Detergent Product Portfolio Optimisation*

The product portfolio optimisation in laundry detergent is highly related to the mixing process of each stage. This process is similar to the pooling problem that is often found to optimise mixing products with different quality to achieve final results within acceptance criteria. The early pooling problem is developed for linear programs with the condensate mixing case [23]. However, the advanced pooling problem can lead to nonlinearities and nonconvexities for the process model [24]. In the advanced stage of the pooling problem usually it is connected with the blending problem. The pooling and blending problem leads to multiple local optima and puts an effort to obtain the global solution [25]. The general structure of the pooling and blending problem is well-captured in Figure 3.

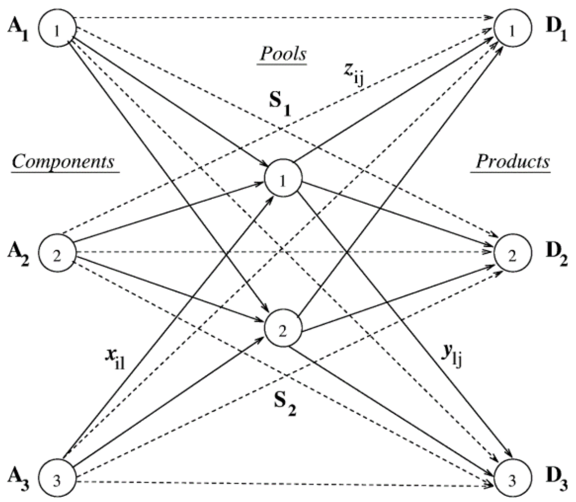

Figure 3. A common pooling and blending problem [26].

The complicated pooling problem is uneasy to solve with the impregnated nonconvexity. This problem formulation is widely explored by [27], [28]. To obtain the solution to this problem, some familiar methods are utilised, namely domain partitioning, discretisation methods, and piecewise linear relaxation [29]. The nonlinearity and nonconvexity may appear due to the problem size or bilinear constraints from the blending systems [30]. Generally, the pooling and blending are in the form of a non-linear program (NLP), however, it also can be in the form mixed integer linear program (MILP) [31]. Research by [32] elaborated on different pooling classes, including standard pooling, generalised pooling, extended pooling, and nonlinear pooling. The research explained several methods such as successive linear programming, global optimisation algorithms, and piecewise-affine underestimation can be utilised for pooling problem-solving. Nevertheless, the real industrial challenges with larger sizes and more physical-relevant problems require adjustment and tuning to apply the optimisation algorithm.

For the product portfolio optimisation, the pooling and blending are considered as a MINLP class because of the various nonlinear constraints that also include the integer and discrete variables [24]. Solving this problem can be done by applying a local or global optimisation algorithm [26]. Applying local optimisation approaches often leads to a poor convergence rate and a large-time solution required, therefore this result is less likely to be accepted. The global optimisation approaches are commonly categorised as either a decomposition-based or branch-and-bound algorithm with the guarantee of a global solution.

The common method utilised to untangle the MINLP problem is correlated with the relaxation approach on the integer variable. The well-known approach is the branch-and-bound method that was popular to solve the MILP initially [33]. Other alternative approaches to tackle the MINLP problem such as Lagrange relaxation, LP relaxation, Decomposition-based, Discretisation with MILP, and Column Generation also perform good results. For the laundry product portfolio optimisation, the column generation approach is a good way to obtain the result [7].

2.3. *Open Source Software Package*

To solve the optimisation problem, various software can be utilised. Previous research done by P&G utilised the SAS Optimisation library [7]. However, the package requires a subscription and does not apply to wider applications [34]. Another option for the public yet robust package is to employ a package with Python library codes. Some notable libraries are Gekko, SciPy, PuLP, and Pyomo [35]. More information about these libraries is shown in Table 1.

Table 1. Information of notable optimisation libraries.

Library Name	License Required	Developer Team	Programming Language
SAS [36]	Proprietary	SAS Inc.	SAS
Pyomo [37]	Freeware	Pyomo	Python
SciPy [38]	Freeware	BSD	Python
PuLP [39]	Freeware	PuLP	Python
Gekko [40]	Freeware	MIT	Python

Pyomo or Python Optimisation Modelling Objects, as one of the notorious libraries, supports the mathematical modelling formulation and analysis with wide-range and complex optimisation implementations [41]. Pyomo offers a resilient and rigorous program with the Python codes application [42]. It is also coupled with diverse optimisation solvers, tightly coupled with Python interface libraries such as IPOPT, GUROBI, and GLPK, and loosely coupled with an external interface, BONMIN for example [43].

With Pyomo, users are granted to construct blocks for each optimisation and can be correlated to others by creating tree-shaped structures [44]. In addition, Pyomo also provides specified solvers namely Mixed Integer Nonlinear Decomposition Toolbox in Pyomo (MindtPy) and Pyomo Robust Optimization Solver (PyROS). MindtPy is able to decompose the MINLP problem into MILP and NLP to find a solution [43]. MindtPy is equipped with notable algorithms to be used for MINLP approximation for example, Outer-Approximation (OA), LP/NLP Branch-and-Bound (LP/NLP BB), and Extended Cutting Plane (ECP). Whereas PyROS is capable of tackling non-convex with two-stage adjustable optimisation problems [45]. PyROS applies the Generalized Robust Cutting-Set (GRC) algorithm to solve the problem [46].

Current research is getting familiarised with Pyomo because it offers an extendable version to create another system. For example, Pyomo.dae is utilised for discretisation frameworks in Model Predictive Control (MPC) and has the capability for multigrid applications [47]. Pyomo's structure is also easy to follow and has noticeable readability clearance. This adds points for Pyomo as a high-level programming library that offers flexible and robust systems.

Apart from the optimisation package, the solver to be applied also has various options. The notable solver packages that applied to Pyomo are described in detail in Table 2 [41]. BONMIN (Basic Open-source Nonlinear Mixed Integer programming)is a wise choice to be utilised as the solver for this research problem since it is able to give the global solution for the MINLP problem without further reformulation [33]. BONMIN applies six different algorithms for solving the MINLP problem as detailed in Table 3.

Table 2. Comparison of different solvers applied in Pyomo [41].

Solver Package	License Required	Directly Solve MINLP
BONMIN	Freeware	Yes
IPOPT	Freeware	No
GLPK	Freeware	No
GUROBI	Proprietary	No
BARON	Proprietary	Yes

Table 3. Description of algorithms applied in BONMIN solver [48].

Algorithm	Description	Ref
B-BB	A simple branch-and-bound algorithm to solve a continuous nonlinear program	[49]
B-OA	An outer-approximation based decomposition algorithm	[50]
B-QG	An outer-approximation based branch-and-cut algorithm	[51]
B-Hyb	A hybrid outer-approximation / nonlinear programming based branch-and-cut algorithm	[52]
B-Ecp	Another outer-approximation based branch-and-cut	[53]
B-iFP	An iterated feasibility pump algorithm	[54]

3. Methods

These research methods highly rely on the previous research conducted by P&G's team in North America with adjustments to handle the compatibility of Pyomo and solver packages. Also, simplification is attributed in this initial research to capture any inconvenience of the adaptation and to be evaluated in the forthcoming research.

3.1. Case Study of P&G Optimisation Problem

In this research, the product portfolio to be examined is referred to as P&G North America problem [7]. The models and equations are based on statistical research that has been conducted based on their manufacturing processes. To limit the research, 23 raw materials (RM) which contain 27 substances (SUB) are assigned to be processed and mixed in 2 mixers (MIXER) to obtain 5 final products (PROD). Of 23 raw materials, some of them are strictly mixed in the mixer and part of them are only added at the final stage. The simple process diagram is captured in Figure 4.

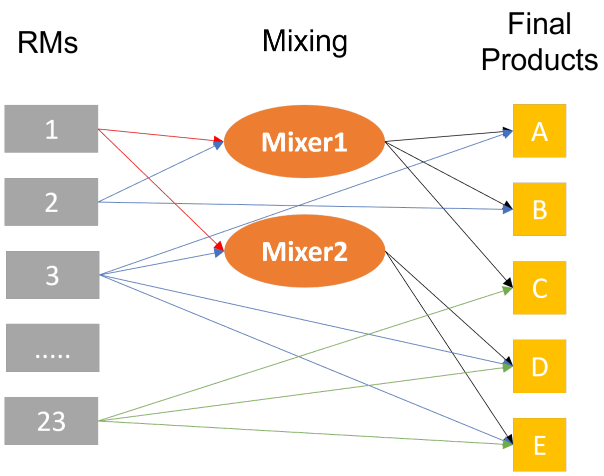

Figure 4. Simple process diagram of laundry product portfolio.

From Figure 4, RM 1 is only assigned for the mixing process (red lines), RM 2 and 3 can be used both in the mixing or final stage (blue lines), and RM 23 is only added in the final stage (green lines). This is related to the characteristics of each RM. For example, surfactants are added in the mixing process, while perfumery and colourant are introduced in the last process [18]. Each final product is supplied only from one mixer (black lines) and all mixers should be assigned to at least one final product. Some equations are elaborated for the comprehensive examination [7], however, in this preliminary research only mixer-related equations and Stain Removal Index (SRI)-related equations will be considered.

SRI-related equations are detailed in Equation (1) to (4):

$$Model_1 = C_1 + C_2\mathrm{SUB}_2 + C_3\mathrm{SUB}_1\mathrm{SUB}_2 + \dots \qquad (1)$$

$$Model_2 = C_4 + C_5 \mathrm{SUB}_4 \mathrm{SUB}_4 + \dots \tag{2}$$

$$Model_3 = C_6 \mathrm{SUB}_{12} - C_7 \mathrm{SUB}_{13} \mathrm{SUB}_{13} + \dots \tag{3}$$

$$Model_4 = 0.3 Model_1 + 0.2 Model_2 + 0.5 Model_3 \tag{4}$$

C_k indicates coefficients and SUB_j indicates substance concentrations in the final product (mg/L). The complete forms and coefficients can be found in the Appendix section.

The mixer-related equation is defined in Equation (5):

$$\begin{aligned} Model_5 = 1 - \frac{\mathrm{SUB}_2}{C_1} \times \mathrm{SUB}_3 + C_2 \times \frac{\mathrm{SUB}_7}{C_3} \times C_4 \mathrm{RM}_6 \\ + C_5 \times \mathrm{SUB}_3^2 - C_6 \times \mathrm{SUB}_7^2 + \dots \end{aligned} \tag{5}$$

with C_k indicates coefficients, SUB_j indicates substance concentrations in the final product (mg/L), and RM_i indicates raw material concentrations in the mixer output (%). Refer to the Appendix section for the complete form and coefficients.

To optimise the mixer composition, the cost of RM in the mixer is evaluated with Equation (6):

$$Model_6 = \sum_i c_{\mathrm{RM}i} \left(\sum_{i,m} y_m \times \mathrm{RM}_{i,m} \right) \tag{6}$$

Additionally, the cost of RM after-mixer composition for each final product is formulated in Equation (7):

$$Model_7 = \sum_{i,p} c_{\mathrm{RM}i} \times \mathrm{RM}_{i,p} \tag{7}$$

The $c_{\mathrm{RM}i}$ indicates RM cost, y_m indicates the integer variable of mixer utilisation (2 mixers), and set p is for final products (5 final products).

Combining $Model_6$ and $Model_7$, the objective function is evaluated as Equation (8):

$$\begin{aligned} &Model_8 \text{ (Objective Function)} = \\ &0.0004 \times \left(\sum_p \mathrm{ML}_p \times Model_6 + Model_7 \right) \end{aligned} \tag{8}$$

Special constraints for each mixer and final product are applied to this optimisation problem. The detail of the constraints is displayed detail in the Appendix section. Furthermore, there is a systemic constraint that applies to the total of all final products. This systemic constraint is related to the total contaminant from all final products that should be controlled under certain values. This systemic constraint is written in Equation (9).

$$Model_9 = 400 \times \sum_p SUB_{25,\,p} \tag{9}$$

From the main equations (1) - (9) above, the product portfolio for liquid laundry products is constructed.

3.2. Constructing Optimisation Problem

Before developing the complete set of an optimisation problem, several general requirements are noticed:

1. All ingredient mass fractions must be added up to 100% in all mixtures, RM in the mixer and final products (linear constraint).
2. Each ingredient mass fraction must fulfil the given bounds (linear constraint).
3. Substance quantity is defined as the percentage of RM multiplied by SUB contained in RM. (linear constraint)
4. SRI of final products must be on the predefined bounds (nonlinear constraint).
5. All mixer outputs must be used (binary variable).
6. Mixer outputs can only be introduced to one final product (binary variable).

With all the models, constraints, and requirements applied, the optimisation model is constructed as pictured in Figure 5.

$$\begin{array}{ll}
\underset{RM_{i,m}, RM_{i,p}, y_m, ML_p}{\min} & Model_8 \\
s.t. & c_p \leq Model_{4,p}, \ \forall p \quad c_p = [52,47,44,53,48] \\
& 9 \leq Model_{5,m}, \ \forall m \\
& Model_9 \leq 5.25 \\
& 0.25 \leq RM_{5,m}, \ \forall m \\
& 0.008 \leq RM_{12,p}, \forall p \\
& SUB_{3,p} \leq 0.09, \ \forall p \\
& \Sigma_i RM_{i,m} = 1 \quad \forall m \\
& \Sigma_m y_m = 1 \\
& \Sigma_i RM_{i,p} = 1 \quad \forall p \in P \\
& 0 \leq RM_{i,m} \leq 1 \text{ (Raw Material Mass Fraction in Mixer)} \\
& 0 \leq ML_p \leq 1 \text{ (Mixer Level Fraction)} \\
& 0 \leq RM_{i,p} \leq 1 \text{ (Raw Material Mass Fraction in Product)} \\
& y_m = \{0,1\} \quad \text{(1 is when mixer is assigned for the product, elsewhere)} \\
& i \in [1,23], \ j \in [1,27], m \in [1,2], \ p \in [ProdA, ProdE]
\end{array}$$

Figure 5. Optimisation model for laundry product portfolio.

More applied constraints related to the upper and lower bound items are available in the Appendix.

To solve that optimisation model, the combination of the decomposition method and column generation approach is assigned referring to [7]. In detail, the below algorithm is applied to that paper:

1. Singleton: This step is to assign each final product with its dedicated mixer outputs.
2. Grouping: Each singleton output is then grouped to create pools for the next stage.
3. Configuration: This step allows each group to find its local minimum cost.
4. Selection: At this stage, each configuration is evaluated to make an end decision.
5. Iteration: It is a step to guarantee the convergence. This is to repeat the algorithm from step 2 until all criterion is fulfilled.

The proposed algorithm is developed by adapting part of the previous approach. It includes the one-time running algorithm to evaluate preliminary issues that may appear by transitioning from the SAS model to a Python-based package. Figure 6 shows the proposed algorithm flowchart.

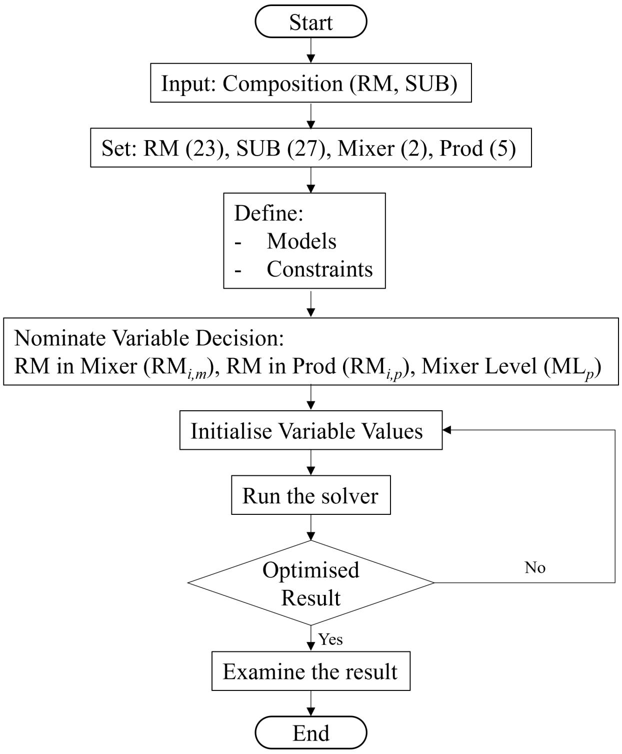

Figure 6. Optimisation model for laundry product portfolio.

3.3. Solving Optimisation Problem

In this research, Pyomo is utilised for constructing the codes to accommodate the proposed algorithm captured in Figure 6. The code is developed with Spyder version 5.5.5. and Pyomo version 6.8.0. The operating system used for code development is Windows 64-bit with Intel 11th Gen Core i5.

In this initial research, the binary variables for mixer selection (y_m) are relaxed such that:

$$y_1 = 1 \text{ and } y_2 = 0$$

Hence, the binary variable can be neglected at the moment and the problem becomes NLP. Therefore, the IPOPT solver package can be utilised because it can handle NLP problems with distinguished results. Moreover, IPOPT installation is more managed for the Pyomo package because it is directly coupled with Pyomo than BONMIN which requires third-party AMPL Solver Library (ASL) installation as a prerequisite. IPOPT version 1.0.3. is assigned as a solver package in this paper.

For this product portfolio, global optima is not the main goal. If the algorithm is able to perform the feasible solution, it will considered as the optimum solution even if it is just the local optimum. This is because the global optimum composition may not be feasible to be produced. Hence, the local minimum is also acceptable to enrich the product portfolio options.

After all, the codes are built and examined. The optimisation algorithm's procedural architecture and final product composition outcomes are visually delineated in Figure 7.

Algorithm 1: Laundry Product Portfolio Optimisation

Input: Raw Materials and Substances
Params: RM, SUB
Var: RM_{Mixer}, $RM_{Prod\_A\ to\ E}$, $Mixer\_Level_{Prod\_A\ to\ E}$
Init: Mixer_Level = 0, iter =0
while iter < max_iter **do**
    Evaluate: Model 1 to 7 and 9
    Evaluate: Bound constraints
    Minimise: Model 8
    Solve: Solver IPOPT
    iter: iter + 1
end while

Figure 7. Codes algorithm for laundry product portfolio.

4. Results

The constructed codes are running well and take about 4-10 seconds for each running. Here are the results for some running scenarios. The codes can be further reviewed via this GitHub path: `5Prods 1Mixer Optimisation.py`

Basic Model

These codes are based on putting all the models and constraints into the optimisation package. The detailed results are presented in Figure 8 - 15.

From the results of the basic codes, it is known that the mixer level of each product has a similar percentage. To anticipate this reiterate, the small number difference is introduced, epsilon. This is to push the optimisation solver to search for another potential solution.

Model with Mixer Level Difference Formula

In this code, the additional formula is induced to search for an alternative for mixer level values. The mixer level is not more different than the basic code. The detailed result is captured in Figure 16 - 23.

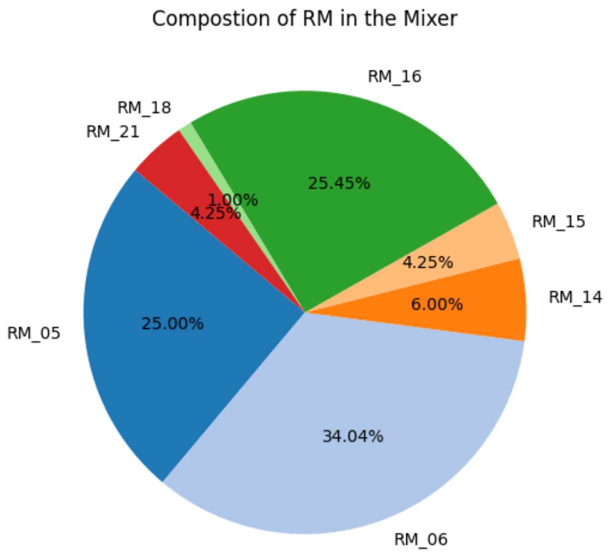

Figure 8. Composition of RM in the mixer.

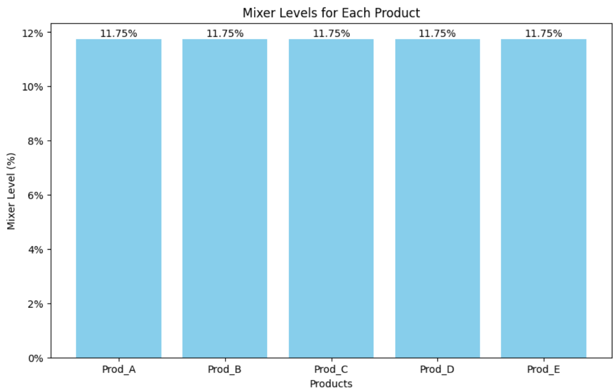

Figure 9. Mixer level of each final product.

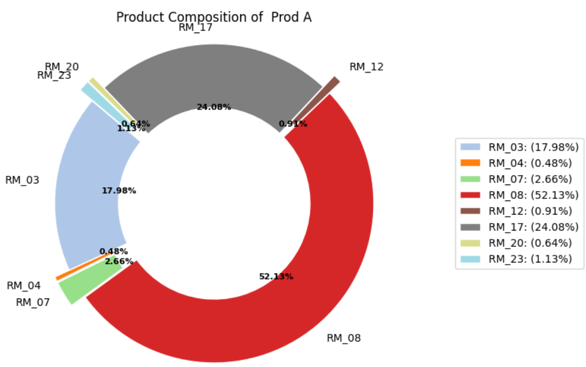

Figure 10. Composition of RM for Prod A.

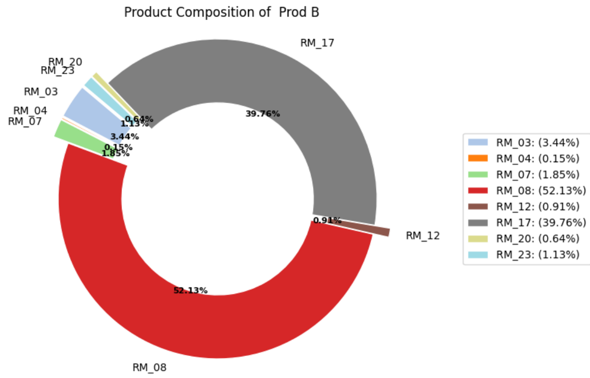

Figure 11. Composition of RM for Prod B.

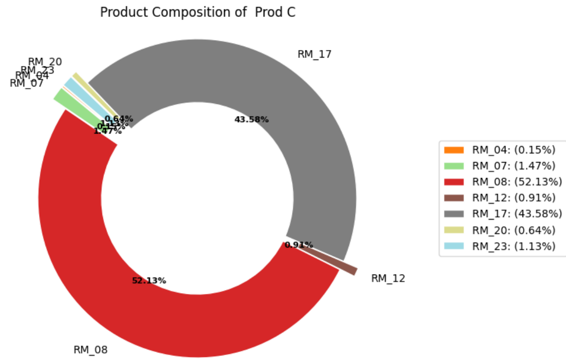

Figure 12. Composition of RM for Prod C.

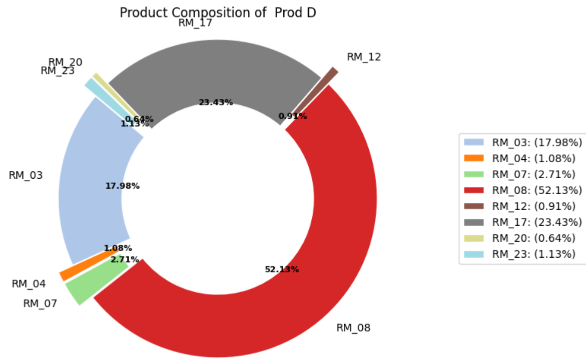

Figure 13. Composition of RM for Prod D.

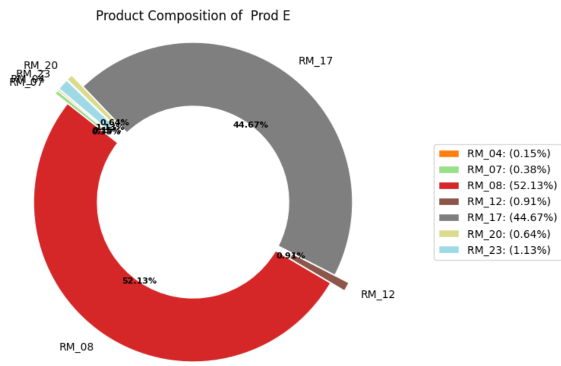

Figure 14. Composition of RM for Prod E.

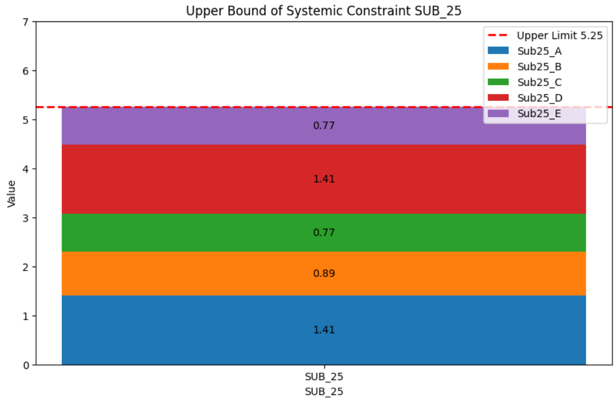

Figure 15. Systemic Constraint of SUB_25.

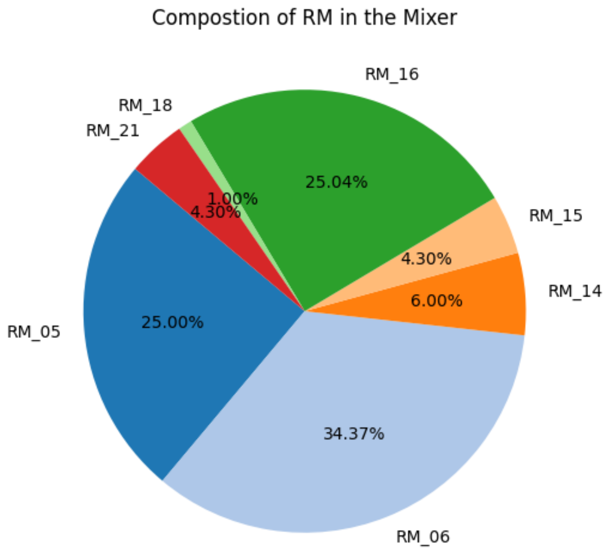

Figure 16. Updated composition of RM in the mixer.

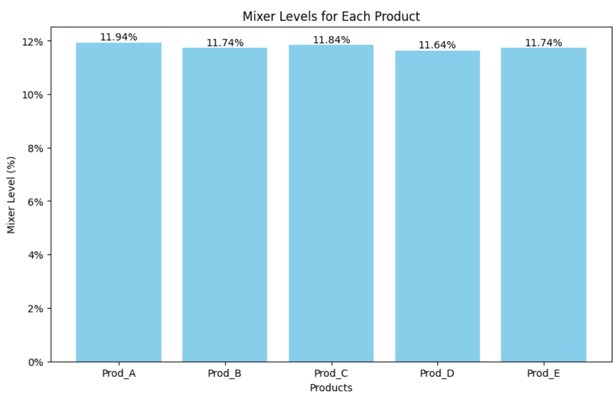

Figure 17. Updated mixer level of each final product.

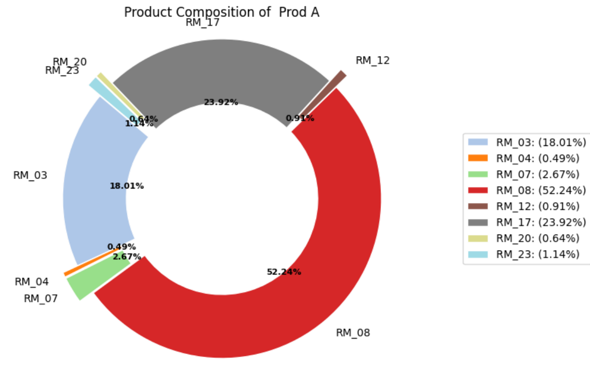

Figure 18. Updated composition of RM for Prod A.

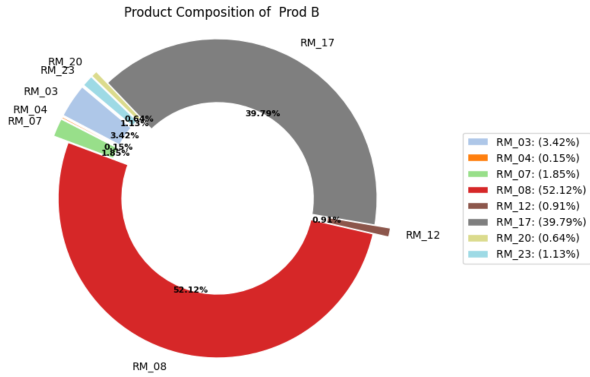

Figure 19. Updated composition of RM for Prod B.

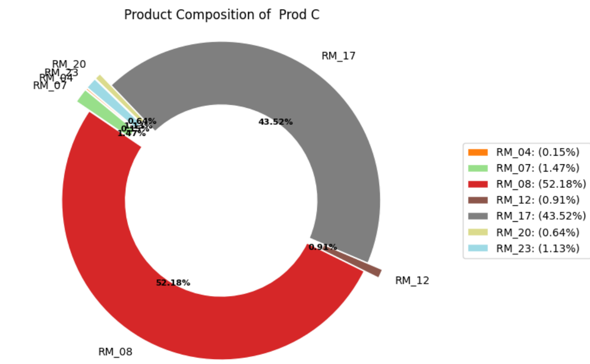

Figure 20. Updated composition of RM for Prod C.

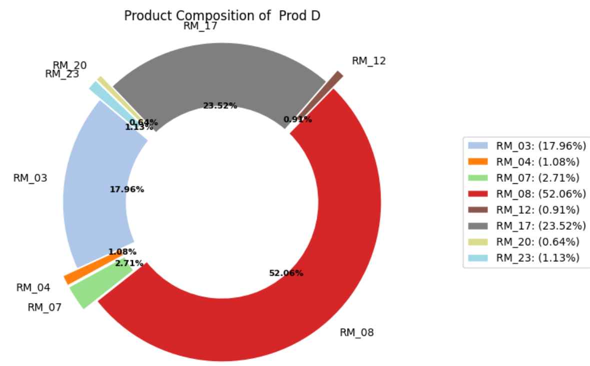

Figure 21. Updated composition of RM for Prod D.

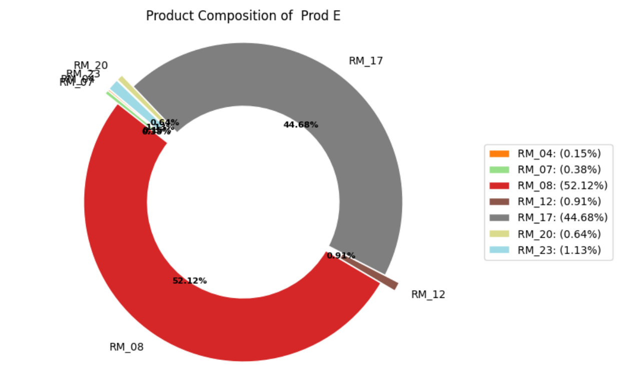

Figure 22. Updated composition of RM for Prod E.

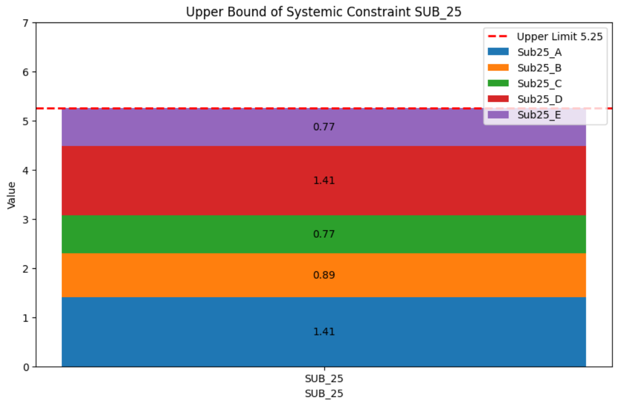

Figure 23. Updated Systemic Constraint of SUB_25.

The second code's result gives similar behaviour to the basic codes. The mixer level of each product is still in the range of 11% with the difference near the epsilon set. Hence, no impactful changes in the result for mixer or product compositions. Further discussion is elaborated in the next section.

5. Discussion

After getting the result of two designed algorithms, analysis is conducted to see if any improvement can be applied to improve the result.

Both of the codes give a similar range for the value of the decision variables. Some indication of this issue is identified and requires further exploration to prove the analysis.

1. The Local Minima Stuck
   Getting stuck in local minima is a common challenge in nonlinear programming, particularly in problems with non-convex landscapes [55]. One of the solutions that can be applied to avoid this stuckness is by introducing random initialisation to expand the search area [56]. Other than that, embedding the integer cut scenario may be useful in eliminating the current results obtained. Integer cuts are constraints added to the problem to exclude certain suboptimal solutions, allowing the algorithm to escape local minima and continue searching for better global solutions. By systematically eliminating non-promising regions of the solution space, integer cuts can direct the search process toward more optimal solutions, enhancing both the accuracy and efficiency of the optimisation [57]. The utility of integer cuts in overcoming local minima is well-documented in the literature, including in the research of [58], who discusses application in improving the global optimality of MINLP problems.
2. Tightened Bounds Applied
   From the five final products, the only difference is on the ($Model_4$) lower bounds and the lower bounds of each product only differ in a range of 1-9 points. The detail about the ($Model_4$) lower bounds are displayed in Table 5.

Table 5. Product details and their corresponding LB values

Product Name	Model	LB
PROD_A	MODEL_4	52
PROD_B	MODEL_4	47
PROD_C	MODEL_4	44
PROD_D	MODEL_4	53
PROD_E	MODEL_4	48

Relaxing bounds in optimisation problems can significantly enhance the performance of algorithms, particularly in complex, non-linear problems. By relaxing the bounds, the feasible region of the solution space is expanded, allowing the algorithm to explore areas that might contain better or alternative solutions that were previously inaccessible.

This broader exploration can lead to the identification of solutions that are not only feasible but also closer to the global optimum. For instance, in process optimisation where constraints are often non-linear, relaxed bounds can help escape local minima and explore more diverse regions of the solution space, improving the overall robustness and effectiveness of the optimisation process. The work of [59] supports this approach, and emphasises that relaxed bounds can be beneficial in the convergence of large-scale nonlinear programming problems, ultimately aiding in finding more globally optimal solutions more efficiently.

From the possible issues that appear above, several solutions may be useful to solve the optimisation problems. The application of these proposed solutions requires advanced algorithm development and further analysis. Some possible efforts to apply are:

1. Introducing Grouping and Selection Mechanism
   One effective strategy to mitigate this issue is the use of grouping and selection mechanisms. These

mechanisms involve dividing the variables or potential solutions into different groups based on certain characteristics, such as their influence on the objective function or constraints. By grouping variables, the algorithm can focus on solving smaller, more manageable subproblems, reducing the complexity of the search process [58].

Within each group, a selection mechanism is applied to choose the most promising candidates for further exploration. This could involve selecting variables or solutions that contribute most significantly to improving the objective function or that demonstrate the greatest potential for leading the search out of a local minimum. By iterative grouping and selecting variables or solutions, the algorithm can effectively narrow down the search space, avoiding areas that are likely to lead to suboptimal local solutions. This approach also facilitates a more systematic exploration of the solution space, increasing the likelihood of escaping local minima and progressing toward a global optimum.

This mechanism is already applied by P&G for the current algorithm [7]. However, adjusting the mathematical model to the Pyomo package is one other task to prepare. By successfully adapting the mathematical model to Pyomo, the algorithm can further enhance the optimisation capabilities, allowing for more effective and scalable solutions to complex engineering challenges.

2. Adapting Metaheuristics Approach
To enhance the robustness of the algorithm used for product portfolio optimisation, adapting a metaheuristics approach can be highly effective. Metaheuristics offer the flexibility to escape local minima by using probabilistic rules to explore the solution space, making them well-suited for the dynamic and often nonlinear nature of product portfolio optimisation [60]. The implementation of these techniques can significantly improve the robustness and overall performance of the optimisation algorithm, leading to more effective and resilient portfolio decisions [61].

A set of metaheuristics algorithms is commonly coupled to solve the optimisation problem, for example, Simulated Annealing (SA), Genetic Algorithm (GA), Particle Swarm Optimisation (PSA), and Differential Evolution (DE) [62], [63]. By incorporating these approaches, the algorithm can better avoid being trapped in local optima and instead converge towards a global solution that optimally balances the various trade-offs involved in product portfolio decisions.

3. Apply Advanced Algorithm: Generalised Disjunction Programming Generalised Disjunction Programming (GDP) is a powerful extension of traditional mathematical programming that allows the modelling of problems with discrete and continuous decisions through logical constraints, often represented as disjunctions [58]. These disjunctions enable the representation of complex decision structures, such as ”either-or” conditions, which are common in many industrial optimisation scenarios. By utilising GDP, the optimisation package can model these scenarios more naturally and accurately, capturing the intricacies of real-world problems that involve both binary choices and continuous variables (e.g., the quantity of that product) [64].

One of the key advantages of GDP is its ability to provide more compact and expressive formulations compared to traditional mixed-integer programming (MIP). This can lead to significant computational savings, as the GDP formulations often result in smaller and more structured models, which are easier for solvers to handle [58]. Furthermore, GDP allows for more efficient branching strategies during the optimisation process, which can reduce the time required to find optimal or near-optimal solutions. For complex problems like product portfolio optimisation or process network design, where multiple conflicting objectives and constraints must be balanced, GDP offers a robust framework that can accommodate the inherent complexities and lead to more optimal and practical solutions [64]. The application of GDP in pooling and blending is extensively explored nowadays. the general pooling and blending application on the process network were applied in the [65] as displayed in Figure 24. Study by [64] included the research on the water treatment process scheme optimisation by applying the GDP method as captured by Figure 25. [66] applied the GDP method in the tank blending systems to obtain minimum cost. This is detailed and captured in Figure 26.

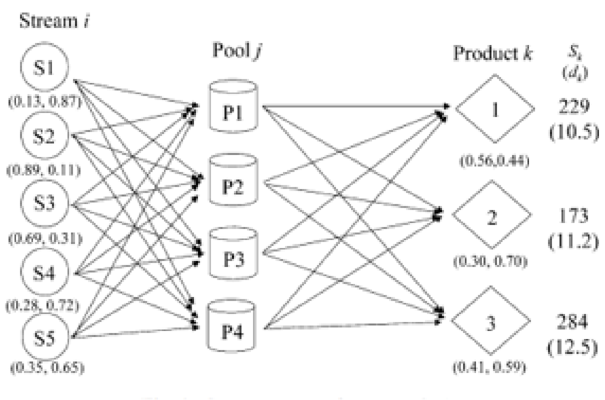

Figure 24. GDP for general blending and pooling [65].

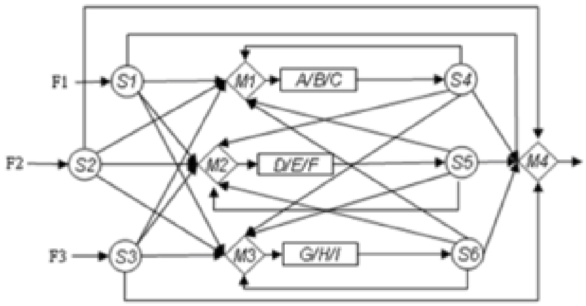

Figure 25. GDP for wastewater process optimisation [64].

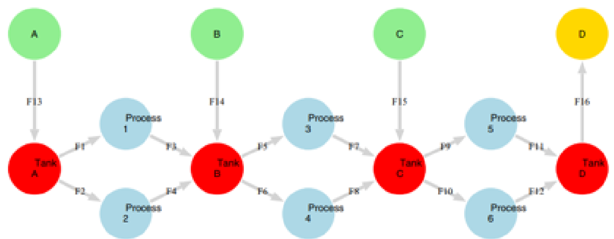

Figure 26. GDP for wastewater process optimisation [64].

6. Conclusion

The research successfully demonstrates the potential of using a MINLP approach with heuristic algorithms to optimise laundry product portfolios. By adapting open-source software tools like Pyomo and IPOPT as the solvers, the study presents a viable solution for managing the complex formulation and production processes involved in product development. The findings highlight local minima stuckness and possibilities that can lead to the issues. However, the study also identifies areas for future work, including the refinement of algorithmic strategies to address challenges such as local minima and constraint management. These advancements could further enhance the robustness and applicability of the model, making it a valuable tool for the FMCG industry.

Acknowledgement

The completion of this paper is deeply indebted to the invaluable guidance and expertise of supervision from Dr Antonio Del Rio Chanona and Emma Pajak from OPTIML PSE Research Group as well as Nats Esquejo from the P&G Team. Furthermore, my gratitude to LPDP as the Indonesian Scholarship Endowment allows me to obtain as much knowledge at Imperial College London.

Appendix

The Indices

i = Raw Material Quantity (23)
j = Substance Quantity (27)
m = Mixer Quantity (2)
p = Final Product Quantity (5)

Supplementary Documents

Supplementary documents consist of complete model equations, raw material compositions based on substances, complete bounds applied, and reference values from the current P&G products. The supplementary documents are available in `Supplementary Documents`

References

[1] D. Distel, E. Hannon, M. Krause, A. Krieg, The data-analytics solution for simplifying complex product portfolios | McKinsey (Dec. 2020).
URL `https://www.mckinsey.com/capabilities/operations/our-insights/finding-the-spot-in-product-portfolio-management`

[2] T. Brzeczek, Optimisation of product portfolio sales and their risk subject to product width and diversity, Review of Managerial Science 14 (5) (2020) 1009–1027. `doi:10.1007/s11846-018-0315-y`.
URL `http://link.springer.com/10.1007/s11846-018-0315-y`

[3] R. G. Cooper, S. J. Edgett, E. J. Kleinschmidt, New Product Portfolio Management: Practices and Performance, Journal of Product Innovation Management 16 (4) (1999) 333–351. `doi:10.1111/1540-5885.1640333`.
URL `https://onlinelibrary.wiley.com/doi/10.1111/1540-5885.1640333`

[4] P&G, P&G Annual Report 2023, Tech. rep. (2024).
URL `https://us.pg.com/annualreport2023/`

[5] X. Han, Research Report on Product Marketing of Procter & Gamble, Advances in Economics, Management and Political Sciences 7 (1) (2023) 414–419. `doi:10.54254/2754-1169/7/20230263`.
URL `https://aemps.ewapublishing.org/article.html?pk=10a8f42a97c14ea28e7632cf93daf398`

[6] I. Farasyn, S. Humair, J. I. Kahn, J. J. Neale, O. Rosen, J. Ruark, W. Tarlton, W. Van De Velde, G. Wegryn, S. P. Willems, Inventory Optimization at Procter & Gamble: Achieving Real Benefits Through User Adoption of Inventory Tools, Interfaces 41 (1) (2011) 66–78. `doi:10.1287/inte.1100.0546`.
URL `https://pubsonline.informs.org/doi/10.1287/inte.1100.0546`

[7] N. Esquejo, K. Miller, K. Norwood, I. Oliveira, R. Pratt, M. Zhao, Statistical and Optimization Techniques for Laundry Portfolio Optimization at Procter & Gamble., Interfaces 45 (5) (2015) 444–461.
URL `http://www.jstor.org/stable/43699521`

[8] C. Adams, M. Borens, E. Burroughes, I. Marcos, Mastering complexity with the consumer-first product portfolio | McKinsey (Oct. 2020).
URL `https://www.mckinsey.com/operations/our-insights/mastering-complexity-with-the-product-portfolio`

[9] N. N. Ozkan, An Example of Open Innovation: P&G, Procedia - Social and Behavioral Sciences 195 (2015) 1496–1502. `doi:10.1016/j.sbspro.2015.06.450`.
URL `https://linkinghub.elsevier.com/retrieve/pii/S1877042815039294`

[10] A. Siew, Designing Optimized Formulations, Pharmaceutical Technology 41 (4) (2017) 16–21.
URL https://www.pharmtech.com/view/designing-optimized-formulations

[11] R. K. Garg, I. Singhvi, OPTIMIZATION TECHNIQUES: AN OVERVIEW FOR FORMULATION DEVELOPMENT 5 (3).

[12] L. Zhang, K. Y. Fung, X. Zhang, H. K. Fung, K. M. Ng, An integrated framework for designing formulated products, Computers & Chemical Engineering 107 (2017) 61–76. doi:10.1016/j.compchemeng.2017.05.014.
URL https://linkinghub.elsevier.com/retrieve/pii/S0098135417302132

[13] H.-k. Fung, C. Wibowo, K. M. Ng, Product-centered Process Synthesis and Development: Detergents, in: Computer Aided Chemical Engineering, Vol. 23, Elsevier, 2007, pp. 239–274. doi:10.1016/S1570-7946(07)80011-3.
URL https://linkinghub.elsevier.com/retrieve/pii/S1570794607800113

[14] K. C. Cheng, Z. S. Khoo, N. W. Lo, W. J. Tan, N. G. Chemmangattuvalappil, Design and performance optimisation of detergent product containing binary mixture of anionic-nonionic surfactants, Heliyon 6 (5) (2020) e03861. doi:10.1016/j.heliyon.2020.e03861.
URL https://linkinghub.elsevier.com/retrieve/pii/S2405844020307064

[15] Y. Yu, J. Zhao, A. E. Bayly, Development of Surfactants and Builders in Detergent Formulations, Chinese Journal of Chemical Engineering 16 (4) (2008) 517–527. doi:10.1016/S1004-9541(08)60115-9.
URL https://linkinghub.elsevier.com/retrieve/pii/S1004954108601159

[16] B. V. Smith, M. G. Ierapepritou, Integrative chemical product design strategies: Reflecting industry trends and challenges, Computers & Chemical Engineering 34 (6) (2010) 857–865. doi:10.1016/j.compchemeng.2010.02.039.
URL https://linkinghub.elsevier.com/retrieve/pii/S0098135410000906

[17] R. Gani, Chemical product design: challenges and opportunities, Computers & Chemical Engineering 28 (12) (2004) 2441–2457. doi:10.1016/j.compchemeng.2004.08.010.
URL https://linkinghub.elsevier.com/retrieve/pii/S0098135404002297

[18] E. Smulders, W. Rähse, G. Jakobi, Laundry detergents, Wiley-VCH, Weinheim, 2002, oCLC: ocm48571942.

[19] F. Marzia, T. Giorgio, Empirical Analysis of the Nature of Competition in a Laundry-Cleaning Additive Market, WSEAS Transactions on Business and Economics 11 (2014) 209–218.
URL https://www.researchgate.net/publication/280740179_Empirical_Analysis_of_the_Nature_of_Competition_in_a_Laundry-Cleaning_Additive_Market

[20] A. N. Ashokan, K. P. Satish, SWOT Analysis of A Multi Product Detergent Manufacturing Industry- A Case Study, International Research Journal of Engineering and Technology (IRJET) 09 (07) (2022) 2387–2392.
URL https://www.irjet.net/archives/V9/i7/IRJET-V9I7438.pdf

[21] G. Ding, A. Teiwes, Process Technologies to Optimize Detergent Manufacturing, SOFW Journal - Home & Personal Care Ingredients & Formulation 146 (10/20) (2020) 16–19.
URL https://foodfeedfinechemicals.glatt.com/wp-content/uploads/2020/12/104_Process-Technologies-Detergent-Manufacturing_en_SOFW_2020-10.pdf

[22] A.I.S.E, Laundry Detergent Testing Guidelines, Tech. rep., International Association for Soaps, Detergents and Maintenance Products (Oct. 2020).

[23] C. A. Haverly, Studies of the behavior of recursion for the pooling problem, ACM SIGMAP Bulletin (25) (1978) 19–28. doi:10.1145/1111237.1111238.
URL https://dl.acm.org/doi/10.1145/1111237.1111238

[24] C. A. Floudas, Mixed Integer Nonlinear Programming: MINLP, in: C. A. Floudas, P. M. Pardalos (Eds.), Encyclopedia of Optimization, Springer US, Boston, MA, 2008, pp. 2234–2247. doi:10.1007/978-0-387-74759-0_394.
URL https://link.springer.com/10.1007/978-0-387-74759-0_394

[25] C. Audet, J. Brimberg, P. Hansen, S. L. Digabel, N. Mladenović, Pooling Problem: Alternate Formulations and Solution Methods, Management Science 50 (6) (2004) 761–776. doi:10.1287/mnsc.1030.0207.
URL https://pubsonline.informs.org/doi/10.1287/mnsc.1030.0207

[26] V. Visweswaran, MINLP: Applications in Blending and Pooling Problems, in: C. A. Floudas, P. M. Pardalos (Eds.), Encyclopedia of Optimization, Springer US, Boston, MA, 2008, pp. 2114–2121. doi:10.1007/978-0-387-74759-0_375.
URL https://link.springer.com/10.1007/978-0-387-74759-0_375

[27] M. Alfaki, D. Haugland, Strong formulations for the pooling problem, Journal of Global Optimization 56 (3) (2013) 897–916. doi:10.1007/s10898-012-9875-6.
URL http://link.springer.com/10.1007/s10898-012-9875-6

[28] N. Boland, T. Kalinowski, F. Rigterink, New multi-commodity flow formulations for the pooling problem, Journal of Global Optimization 66 (4) (2016) 669–710. doi:10.1007/s10898-016-0404-x.
URL http://link.springer.com/10.1007/s10898-016-0404-x

[29] Y. Chen, C. T. Maravelias, X. Zhang, Tightening Discretization-based MILP Models for the Pooling Problem using Upper Bounds on Bilinear Terms, arXiv:2207.03699 [cs, math] (Jan. 2023).
URL http://arxiv.org/abs/2207.03699

[30] Z. Haonan, M. Samavati, A. J. Hill, Heuristics for integrated blending optimisation in a mining supply chain, Omega 102 (2021) 102373. doi:10.1016/j.omega.2020.102373.
URL https://linkinghub.elsevier.com/retrieve/pii/S0305048320307271

[31] A. Gupte, S. Ahmed, S. S. Dey, M. S. Cheon, Relaxations and discretizations for the pooling problem, Journal of Global Optimization 67 (3) (2017) 631–669. doi:10.1007/s10898-016-0434-4.
URL http://link.springer.com/10.1007/s10898-016-0434-4

[32] R. Misener, C. A. Floudas, Advances for the pooling problem: Modeling, global optimization, and computational studies survey, Applied and Computational Mathematics 8 (Jan. 2009).

[33] J. Kronqvist, D. E. Bernal, A. Lundell, I. E. Grossmann, A review and comparison of solvers for convex MINLP, Optimization and Engineering 20 (2) (2019) 397–455. doi:10.1007/s11081-018-9411-8.
URL http://link.springer.com/10.1007/s11081-018-9411-8

[34] How to buy SAS Viya.
URL https://www.sas.com/en_gb/software/viya/how-to-buy-sas-viya.html

[35] Solver Max - Optimization modelling in Python.
URL https://www.solvermax.com/resources/links/optimization-modelling-in-python#:~:text=Python%2DMIP-,SciPy,many%20other%20classes%20of%20problems.

[36] SAS Help Center.
URL https://go.documentation.sas.com/doc/en/pgmsascdc/9.4_3.3/casmopt/titlepage.htm

[37] Pyomo Documentation 6.8.1.dev0 — Pyomo 6.8.1.dev0 documentation.
URL https://pyomo.readthedocs.io/en/latest/index.html

[38] SciPy -.
URL https://scipy.org/

[39] Optimization with PuLP — PuLP 2.9.0 documentation.
URL https://coin-or.github.io/pulp/

[40] Gekko Python.
URL https://machinelearning.byu.edu/

[41] M. L. Bynum, G. A. Hackebeil, W. E. Hart, C. D. Laird, B. L. Nicholson, J. D. Siirola, J.-P. Watson, D. L. Woodruff, Pyomo — Optimization Modeling in Python, Vol. 67 of Springer Optimization and Its Applications, Springer International Publishing, Cham, 2021. doi:10.1007/978-3-030-68928-5.
URL http://link.springer.com/10.1007/978-3-030-68928-5

[42] S. N. L. , Pyomo Tutorial, Tech. rep., Sandia Corporation (2016).

[43] W. E. Hart, J.-P. Watson, D. L. Woodruff, Pyomo: modeling and solving mathematical programs in Python, Mathematical Programming Computation 3 (3) (2011) 219–260. doi:10.1007/s12532-011-0026-8.
URL http://link.springer.com/10.1007/s12532-011-0026-8

[44] F. Ceccon, R. Misener, Solving the pooling problem at scale with extensible solver GALINI, arXiv:2105.01687 [math] (May 2021).
URL http://arxiv.org/abs/2105.01687

[45] PyROS Solver — Pyomo 6.8.0 documentation.
URL https://pyomo.readthedocs.io/en/stable/contributed_packages/pyros.html

[46] N. M. Isenberg, P. Akula, J. C. Eslick, D. Bhattacharyya, D. C. Miller, C. E. Gounaris, A generalized cutting-set approach for nonlinear robust optimization in process systems engineering, AIChE Journal 67 (5) (2021) e17175. doi:10.1002/aic.17175.
URL https://aiche.onlinelibrary.wiley.com/doi/10.1002/aic.17175

[47] B. Nicholson, J. D. Siirola, J.-P. Watson, V. M. Zavala, L. T. Biegler, pyomo.dae: a modeling and automatic discretization framework for optimization with differential and algebraic equations, Mathematical Programming Computation 10 (2) (2018) 187–223. doi:10.1007/s12532-017-0127-0.
URL http://link.springer.com/10.1007/s12532-017-0127-0

[48] P. Bonami, M. Kilinç, J. Linderoth, Algorithms and Software for Convex Mixed Integer Nonlinear Programs (Oct. 2009).
URL https://hal.science/hal-00423416

[49] O. K. Gupta, A. Ravindran, Branch and Bound Experiments in Convex Nonlinear Integer Programming, Management Science 31 (12) (1985) 1533–1546. doi:10.1287/mnsc.31.12.1533.
URL https://pubsonline.informs.org/doi/10.1287/mnsc.31.12.1533

[50] M. A. Duran, I. E. Grossmann, An outer-approximation algorithm for a class of mixed-integer nonlinear programs, Mathematical Programming 36 (3) (1986) 307–339. doi:10.1007/BF02592064.
URL http://link.springer.com/10.1007/BF02592064

[51] I. Quesada, I. Grossmann, An LP/NLP based branch and bound algorithm for convex MINLP optimization problems, Computers & Chemical Engineering 16 (10-11) (1992) 937–947. doi:10.1016/0098-1354(92)80028-8.
URL https://linkinghub.elsevier.com/retrieve/pii/0098135492800288

[52] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, A. Wächter, An algorithmic framework for convex mixed integer nonlinear programs, Discrete Optimization 5 (2) (2008) 186–204. doi:10.1016/j.disopt.2006.10.011.
URL https://linkinghub.elsevier.com/retrieve/pii/S1572528607000448

[53] K. Abhishek, S. Leyffer, J. Linderoth, FilMINT: An Outer Approximation-Based Solver for Convex Mixed-Integer Nonlinear Programs, INFORMS Journal on Computing 22 (4) (2010) 555–567. doi:10.1287/ijoc.1090.0373.
URL https://pubsonline.informs.org/doi/10.1287/ijoc.1090.0373

[54] P. Bonami, G. Cornuéjols, A. Lodi, F. Margot, A Feasibility Pump for mixed integer nonlinear programs, Mathematical Programming 119 (2) (2009) 331–352. doi:10.1007/s10107-008-0212-2.
URL http://link.springer.com/10.1007/s10107-008-0212-2

[55] S. A. Vavasis, Nonlinear optimization: complexity issues, no. 8 in The International series of monographs on computer science, Oxford University Press, New York, 1991.

[56] B. Choi, J.-H. Lee, D.-H. Kim, Solving local minima problem with large number of hidden nodes on two-layered feed-forward artificial neural networks, Neurocomputing 71 (16-18) (2008) 3640–3643. doi:10.1016/j.neucom.2008.04.004.
URL https://linkinghub.elsevier.com/retrieve/pii/S0925231208002002

[57] Z. Huang, K. Wang, F. Liu, H.-L. Zhen, W. Zhang, M. Yuan, J. Hao, Y. Yu, J. Wang, Learning to select cuts for efficient mixed-integer programming, Pattern Recognition 123 (2022) 108353. doi:10.1016/j.patcog.2021.108353.
URL https://linkinghub.elsevier.com/retrieve/pii/S0031320321005331

[58] I. E. Grossmann, Review of Nonlinear Mixed-Integer and Disjunctive Programming Techniques, Optimization and Engineering 3 (3) (2002) 227–252. doi:10.1023/A:1021039126272.
URL http://link.springer.com/10.1023/A:1021039126272

[59] L. T. Biegler, Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes, Society for Industrial and Applied Mathematics, 2010. doi:10.1137/1.9780898719383.
URL http://epubs.siam.org/doi/book/10.1137/1.9780898719383

[60] M. A. Boschetti, V. Maniezzo, M. Roffilli, A. Bolufé Röhler, Matheuristics: Optimization, Simulation and Control, in: M. J. Blesa, C. Blum, L. Di Gaspero, A. Roli, M. Sampels, A. Schaerf (Eds.), Hybrid Metaheuristics, Vol. 5818, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 171–177, series Title: Lecture Notes in Computer Science. doi:10.1007/978-3-642-04918-7_13.
URL http://link.springer.com/10.1007/978-3-642-04918-7_13

[61] V. Maniezzo, M. A. Boschetti, T. G. Stützle, Matheuristics: algorithms and implementations, EURO advanced tutorials on operational research, Springer, Cham, 2021.

[62] M. Gendreau, J.-Y. Potvin (Eds.), Handbook of Metaheuristics, 3rd Edition, no. 272 in International Series in Operations Research & Management Science, Springer International Publishing : Imprint: Springer, Cham, 2019. doi:10.1007/978-3-319-91086-4.

[63] Metaheuristics for hard optimization: methods and case studies, Springer, Berlin, 2006.

[64] I. E. Grossmann, J. P. Ruiz, Generalized Disjunctive Programming: A Framework for Formulation and Alternative Algorithms for MINLP Optimization, in: J. Lee, S. Leyffer (Eds.), Mixed Integer Nonlinear Programming, Vol. 154, Springer New York, New York, NY, 2012, pp. 93–115. doi:10.1007/978-1-4614-1927-3_4.
URL http://link.springer.com/10.1007/978-1-4614-1927-3_4

[65] S. Lee, I. E. Grossmann, Global optimization of nonlinear generalized disjunctive programming with bilinear equality constraints: applications to process networks, Computers & Chemical Engineering 27 (11) (2003) 1557–1575. doi:10.1016/S0098-1354(03)00098-X.
URL https://linkinghub.elsevier.com/retrieve/pii/S009813540300098X

[66] H. D. Perez, I. E. Grossmann, Extensions to generalized disjunctive programming: hierarchical structures and first-order logic, Optimization and Engineering 25 (2) (2024) 959–998. doi:10.1007/s11081-023-09831-x.
URL https://link.springer.com/10.1007/s11081-023-09831-x