

**proposte titolo**

1. Transition from decision science to data science in logistics and operations design and control;
2. Data-driven analytics to support decision-making in supply chain systems;
3. Data-driven analytics in decision making for logistics and operations;
4. Design and control of logistics and operations. The role of data-driven analytics;
5. Design and control of a supply chain system. The role of data-driven analytics;
6. Data-driven analytics for supply chain system design and management;
7. The new role of data-driven analytics in logistics and operations management;
8. Analytics and methods for data-driven logistics.
9. Analytics and methods for logistics and operations;
10. Data-driven methods for logistics and operations.

Draft Version 1.0

Draft Version 1.0

PhD Thesis

Eng. Alessandro Tufano

September 2020

Draft Version 1.0

## Preface

*How do we know what we know?*

---

Thousands of years ago, the Greek philosopher Plato introduced the “theory of *Forms*”, i.e. a philosophical viewpoint where an ideal world called *Hyperuranion* contains the purest and most accurate realisation of the knowledge. This unreachable knowledge is called the *Form*. The reality surrounding us in the real world is an imitation of the *Form*, and it is called the *Substance*. Then, according to Plato, knowledge is a deductive process, from the steady and perfect *Form* to its “dirty” realisation in the *Substance*.

Differently, the Greek Philosopher Aristotle considers the real world as the only source of knowledge. In his viewpoint, the empirical process of observing the world is the only path to get knowledge. This process is, then, inductive and implies that anything can only exist if a living being can observe it.

Aristotle’s philosophy is the background of this work that will approach the logistics and operations phenomena with an inductive approach. The data-driven methodology creating knowledge by observing and classifying **data** is an implementation on lifeless machines of one of the most beautiful philosophical intuition in the story of our world.



Figure 1: “The school of Athens”, Apostolic Palace, Vatican City. In the centre of the fresco, Plato and Aristotle.

## Contents

<b>I DATA-DRIVEN MODELLING</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Taxonomy . . . . .	3
1.2 Decision Science Topology . . . . .	4
1.2.1 Centralised information with centralised decision-makers (CC) . . . . .	5
1.2.2 Centralised information with distributed decision-makers (CD) . . . . .	6
1.2.3 Decentralised information with distributed decision-makers (DD) . . . . .	7
1.2.4 Decentralised information with centralised decision-makers (DC) . . . . .	7
1.3 Data Science Topology . . . . .	7
1.3.1 Fundamentals of data science . . . . .	10
1.4 History of Logistics and Operations Research . . . . .	10
1.5 A change in the perspective . . . . .	12
1.6 Towards Predictive-Prescriptive Optimisation . . . . .	16
1.7 The quality of data . . . . .	17
<b>2 Research Background</b>	<b>21</b>
2.1 Academic background . . . . .	22
2.2 Industrial background . . . . .	24
<b>3 The Information Framework</b>	<b>31</b>
3.1 Ontology . . . . .	31

3.2	The physics of a supply chain . . . . .	33
3.3	The information of a supply chain . . . . .	36
3.3.1	Information framework . . . . .	37
3.3.2	Implementation and utilisation of the framework . . . . .	40
<b>4</b>	<b>Data-driven decision making</b>	<b>43</b>
4.1	Data Glossary . . . . .	44
4.2	Decision patterns . . . . .	44
4.3	Decision trees . . . . .	47
4.4	Main contributions of this book . . . . .	56
<b>II</b>	<b>LET'S MATH</b>	<b>59</b>
<b>5</b>	<b>Logical Modelling</b>	<b>61</b>
5.1	Business Process Model and Notation . . . . .	62
<b>6</b>	<b>Elements of probability and statistics</b>	<b>65</b>
6.1	Probability Theory . . . . .	65
6.1.1	Statistical Moments . . . . .	66
6.1.2	Covariance and Correlation . . . . .	68
6.1.3	Distance between random variables . . . . .	70
6.2	Statistics . . . . .	73
6.2.1	Estimators . . . . .	73
6.2.2	Maximum Likelihood Inference . . . . .	76
6.2.3	Kernel Density Estimation . . . . .	76
6.2.4	Bootstrapping method . . . . .	77
6.2.5	Montecarlo method . . . . .	78
6.2.6	Data collection and Measurement systems . . . . .	79
6.3	Statistical Distributions . . . . .	80
6.3.1	Normal distribution . . . . .	81
6.3.2	Central limit theorem . . . . .	82
6.3.3	Multivariate normal distribution . . . . .	82
6.3.4	Poisson Distribution . . . . .	82
6.3.5	Triangular Distribution . . . . .	83
6.4	Statistical tests . . . . .	84
6.4.1	Z-test (normal distribution) . . . . .	85
6.4.2	t-test . . . . .	87
6.4.3	$\chi^2$ -test . . . . .	87
6.4.4	F-test . . . . .	89
6.5	Time series analysis . . . . .	90
6.5.1	Time series decomposition . . . . .	91
6.5.2	ARIMA models . . . . .	94
6.5.3	Fourier transform . . . . .	97

6.6 Bayesian Statistics . . . . .	99
<b>7 Dimensionality reduction</b>	<b>103</b>
7.1 Feature extraction . . . . .	104
7.1.1 Singular Value Decomposition (SVD) . . . . .	104
7.1.2 Principal Component Analysis (PCA) . . . . .	105
7.1.3 Multi-dimensional scaling . . . . .	108
7.1.4 t-SNE . . . . .	108
7.2 Feature selection . . . . .	108
7.2.1 Selection by correlation . . . . .	108
7.2.2 Selection by variance . . . . .	109
7.2.3 Selection by Lasso coefficients . . . . .	111
7.2.4 Selection by using a decision tree . . . . .	111
7.2.5 Forward Stepwise selection . . . . .	112
<b>8 Unsupervised learning</b>	<b>115</b>
8.1 Association rules . . . . .	115
8.2 Clustering . . . . .	116
8.2.1 K-means . . . . .	117
8.2.2 Hierarchical clustering . . . . .	117
8.2.3 Mixture models . . . . .	120
8.2.4 Bag of words . . . . .	122
<b>9 Linear Methods for Regression</b>	<b>123</b>
9.1 Supervised learning . . . . .	123
9.2 Linear regression (OLS) . . . . .	129
9.2.1 Geometrical representation of the linear regression . . . . .	130
9.3 Shrinkage methods . . . . .	133
9.3.1 Ridge regression (L2-regularisation) . . . . .	133
9.3.2 Lasso regression (L1-regularisation) . . . . .	135
9.3.3 Elastic-net regression . . . . .	135
9.3.4 Least angle regression . . . . .	135
9.4 Derived input methods . . . . .	136
<b>10 Linear Methods for Classification</b>	<b>139</b>
10.1 Model selection . . . . .	141
10.1.1 Accuracy . . . . .	141
10.1.2 Confusion matrix . . . . .	141
10.1.3 Log-loss . . . . .	142
10.1.4 AUC . . . . .	142
10.1.5 Precision and recall . . . . .	143
10.2 Linear Discriminant Analysis . . . . .	144
10.3 Logistic Regression . . . . .	145

<b>11 Non-linear methods</b>	<b>147</b>
11.1 Evolutionaries' methods . . . . .	148
11.1.1 Genetic search . . . . .	149
11.2 Symbolists' methods . . . . .	150
11.2.1 Decision trees . . . . .	150
11.3 Bayesians' methods . . . . .	151
11.3.1 Naïve Bayes . . . . .	152
11.3.2 Markov Chains . . . . .	153
11.3.3 Hidden Markov Models and Kalman filter . . . . .	153
11.3.4 Bayesian Networks and Montecarlo Markov Chains . . . . .	154
11.4 Analogisers' methods . . . . .	156
11.4.1 Support vector machines . . . . .	156
11.5 Connectionists' methods . . . . .	157
11.5.1 Neural Networks . . . . .	158
<b>12 Ensemble methods</b>	<b>161</b>
12.1 Bagging . . . . .	161
12.2 Random Forests . . . . .	162
12.3 AdaBoost . . . . .	163
12.4 Gradient boosting . . . . .	164
12.5 Mixture of Experts . . . . .	165
<b>13 Prescriptive analysis</b>	<b>167</b>
13.1 Prescriptive models . . . . .	167
13.2 Heuristics and metaheuristics . . . . .	168
13.3 Linear Optimisation . . . . .	169
13.3.1 Duality . . . . .	169
13.4 Discrete event simulation . . . . .	173
13.5 Multiple decisions or decision-makers . . . . .	173
<b>III DISTRIBUTION SYSTEMS</b>	<b>175</b>
<b>14 Diagnostic Models</b>	<b>177</b>
14.1 Ontology . . . . .	178
14.2 Data Structure . . . . .	179
14.2.1 A relational model for distribution networks . . . . .	180
14.2.2 A non-relational model for distribution networks . . . . .	182
14.3 Decision patterns . . . . .	186
<b>15 Distribution System Control</b>	<b>189</b>
15.1 Introduction . . . . .	189
15.1.1 The actors of a supply chain network . . . . .	189
15.1.2 Logistics platforms . . . . .	192

15.1.3 The day-by-day of a distribution network . . . . .	196
15.2 Performance assessment (P8) . . . . .	199
15.2.1 Model-driven methods (D4) . . . . .	199
15.2.2 Data-driven methods (D1, D2) . . . . .	201
15.3 Workload prediction (P9) . . . . .	217
15.3.1 Model-driven methods (PD2) . . . . .	217
15.3.2 Data-driven methods (PD1) . . . . .	218
15.4 Vehicle choice & synchromodality (P2) . . . . .	220
15.4.1 Data-driven methods (PS2) . . . . .	220
15.5 Vehicle routing (P10) . . . . .	223
15.5.1 Model-driven methods (PS1) . . . . .	223
15.5.2 Data-driven methods (PS4) . . . . .	233
<b>16 Distribution System Design</b>	<b>239</b>
16.1 Location – allocation problem (P6) . . . . .	240
16.1.1 Model-driven methods (PS3) . . . . .	241
16.1.2 Application . . . . .	242
16.2 Network topology design (P2) . . . . .	242
16.2.1 Model-driven methods (PS1) . . . . .	243
16.2.2 Data-driven methods (PS2) . . . . .	244
16.2.3 Application . . . . .	252
16.3 Route frequency design (P5) . . . . .	255
16.3.1 Model-driven methods (PS4) . . . . .	255
16.4 Service time windows design (P5) . . . . .	256
16.4.1 Model-driven methods (PS4) . . . . .	256
16.5 Shipping priority definition (P7) . . . . .	261
<b>IV STORAGE SYSTEMS</b>	<b>263</b>
<b>17 Diagnostic Models</b>	<b>265</b>
17.1 Ontology . . . . .	266
17.2 Data Structure . . . . .	268
17.2.1 A relational model for warehousing systems . . . . .	268
17.2.2 A non-relational model for warehousing systems . . . . .	271
17.3 Decision Patterns . . . . .	276
<b>18 Storage System Control</b>	<b>281</b>
18.1 Performance assessment (P8) . . . . .	281
18.1.1 Model-driven methods (D4) . . . . .	281
18.1.2 Data-driven methods (D2, D4) . . . . .	283
18.2 Workload prediction (P9) . . . . .	293
18.2.1 Model-driven methods (PD2) . . . . .	293
18.2.2 Data-driven methods (D1, PD1) . . . . .	293

18.3 Vehicle routing (P10) . . . . .	297
<b>19 Storage System Design</b>	<b>299</b>
19.1 Weight and size estimation (P1) . . . . .	300
19.1.1 Model-driven methods (D2) . . . . .	300
19.1.2 Data-driven methods (D1) . . . . .	301
19.2 Definition of the inventory level (P5) . . . . .	302
19.2.1 Data-driven methods (D2, D3) . . . . .	302
19.3 Choice of the storage technology (P2) . . . . .	304
19.3.1 Model-driven methods (D4) . . . . .	309
19.4 Storage allocation (P5) . . . . .	310
19.4.1 Model-driven methods (PS4) . . . . .	310
19.5 Storage assignment/slotting (P2) . . . . .	312
19.5.1 Model-driven methods (PS3) . . . . .	312
19.5.2 Data-driven methods (PS2) . . . . .	313
19.6 Layout design (P6) . . . . .	314
19.6.1 Model-driven methods (PS3) . . . . .	314
19.6.2 Data-driven methods (D3) . . . . .	315
19.7 Picking policy design (P7) . . . . .	317
19.7.1 Model-driven methods (D4) . . . . .	318
19.8 Route design (P3) . . . . .	318
19.9 Inbound and outbound area design (P5) . . . . .	318
19.9.1 Model-driven methods (D4) . . . . .	318
<b>V PRODUCTION SYSTEMS</b>	<b>331</b>
<b>20 Diagnostic Models</b>	<b>333</b>
20.1 Ontology . . . . .	333
20.2 Data Structure . . . . .	335
20.2.1 A relational model for production environments . . . . .	336
20.2.2 A Non-relational model for production environments . . . . .	342
20.3 Decision Patterns . . . . .	343
<b>21 Production Node Control</b>	<b>347</b>
21.1 Performance assessment (P8) . . . . .	347
21.1.1 Model-driven methods (D4) . . . . .	347
21.1.2 Data-driven methods (D1, D2, D3, D4) . . . . .	348
21.2 Workload prediction (P9) . . . . .	349
21.2.1 Model-driven methods (D1) . . . . .	349
21.2.2 Data-driven methods (PD1, PD2) . . . . .	351
21.3 Job scheduling (P10) . . . . .	351
21.3.1 Model-driven methods (PS1) . . . . .	351
21.4 Applications . . . . .	352

21.4.1 Control of a food catering plant . . . . .	352
21.4.2 Control of a 3PL packaging plant for automotive spare parts . . . . .	357
<b>22 Plant Design</b>	<b>363</b>
22.1 Clustering parts into product families (P1) . . . . .	364
22.1.1 Model-driven methods (D2) . . . . .	365
22.1.2 Data-driven methods (D1) . . . . .	366
22.2 Facility location (P6) . . . . .	368
22.2.1 Model-driven methods (PS4) . . . . .	368
22.3 Auxiliary systems design (P4) . . . . .	370
22.4 Technology and asset choice (P2) . . . . .	370
22.4.1 Model-driven methods (PS3) . . . . .	372
22.4.2 Data-driven methods (PS2) . . . . .	374
22.5 Definition of the number of assets (P5) . . . . .	375
22.5.1 Model-driven methods (PS4) . . . . .	375
22.6 Layout design (P6) . . . . .	376
22.6.1 Model-driven methods (PS1) . . . . .	376
22.7 Applications . . . . .	377
22.7.1 Plant Design of a food catering plant . . . . .	378
22.7.2 Plant Design of a 3PL packaging plant for automotive spare parts . . . . .	382
<b>23 Process Design</b>	<b>391</b>
23.1 Inventory policy design (P7) . . . . .	391
23.1.1 Model-driven methods (PS4) . . . . .	391
23.1.2 Data-driven methods . . . . .	396
23.2 Handling design (P3) . . . . .	397
23.3 Workstation design (P4) . . . . .	399
23.3.1 Model-driven methods (PS4) . . . . .	399
23.4 Applications . . . . .	400
23.4.1 Process Design of a food catering plant . . . . .	400

Draft Version 1.0

Draft Version 1.0

# Part I

## DATA-DRIVEN MODELLING



# 1

## Introduction

*How do you know if what you believe is really true?*

This chapter introduces the background of this work identifying its relevance, originality and the placement within the existing literature.

The background of this work is engineering. Engineering is composed of models based on math (approximations) that describe a physical phenomenon. The electricity current passing through a wire, the lift force of the air which supports the weight of an aircraft, the exchange of heat in an air conditioning system are physical phenomenon described by engineering models. On-field experiments and observations lead to the definition of the models, i.e. the mathematical relationships between the entities involved in the phenomenon.

### 1.1 Taxonomy

Logistics research is the research domain of this work, which is borderline between engineering and economics. The word *taxonomy* comes from the ancient Greek τάξις (order) and νόμος (law). A taxonomy is used to classify the disciplines of a research domain systematically. Figure 1.1 proposes a taxonomy of the research domain regarding logistics and production systems. Table 1.1 introduces the glossary with the keywords of this taxonomy.

This work considers the relationship between decision science and data science proposing a new role of data analytics as a prescriptive tool (together with operations management). The *System Analysis* is the field of applica-

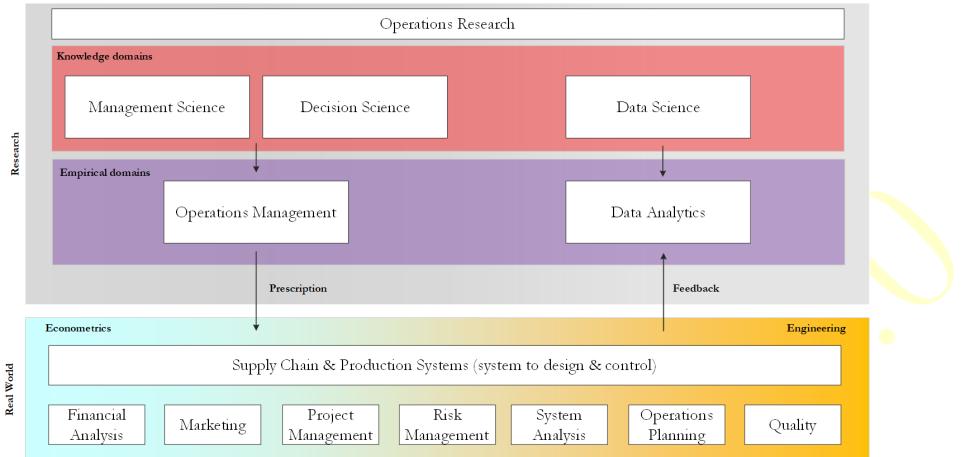


Figure 1.1: Taxonomy of logistics and operations research

Table 1.1: Glossary of operations research.

Keyword	Description
Research domain	contributions of academic research on a specific topic.
Real World	a physical system.
Operations Research	a research discipline studying the operations (e.g., storage, distribution and production) in the supply chain.
Management Science	a research discipline studying the relationship between people and resources in an operational context.
Decision Science	a research discipline studying the outcome of people's decisions or expert systems in an operational context.
Data Science	a research discipline studying the information content of data.
Operations Management	the outcome of management and decision science, i.e. a set of models to design and control a supply chain.
Data Analytics	a set of models to understand and analyse the information content of a dataset.
Supply Chain & Production systems	the set of physical and virtual entities implementing technology for production, storage and transportation of goods.
Financial Analysis	application of a set of methods to identify the financial position of a physical (e.g. an asset) or virtual (e.g. a stock) entity in the supply chain.
Marketing	application of a set of methods to connect the demand and offer of a product/service.
Project Management	application of a set of methods to manage resource and time for the realisation of a project.
Risk Management	application of a set of methods to manage and control the risk connected to the entities of the supply chain.
System Analysis	application of a set of methods to manage and control the behaviour of a set of entities in the supply chain.
Operations Planning	application of a set of methods to manage resource and time for the realisation of the product/service.
Quality	application of a set of methods to keep the operations under statistical control.

tion of this work in the supply chain. The following paragraphs introduce the topology for decision 1.2 and data science 1.3.

## 1.2 Decision Science Topology

The word *topology* comes from the ancient Greek τόπος and λόγος, literally place and study. Differently from taxonomy, the topology studies the “space” of a discipline. In this case, we want to explore which variables influence the nature of the tools used in decision and data science.

To define the object of study of this work, we consider the literature [1] classifying a decision-making process according to:

- the number of decision-makers;
- their knowledge about the information relevant for a decision.

Usually, a decision-maker (DM) makes a decision (i.e., setting the value of decision variables) based on the knowledge of some information (i.e., decision parameters).

A decision process can have single or multiple DMs. We define, accordingly, concentrated and distributed decision making. The information for the decision-making process can be centralised within a single entity/actor or decentralised and owned by several sources. Depending on the topology of information and decision-makers, there are different tools to support the decision process. Figure 1.2 illustrates this decision topology with classic decision science tools. We analyse four different cases (CC, CD, DC and DD).

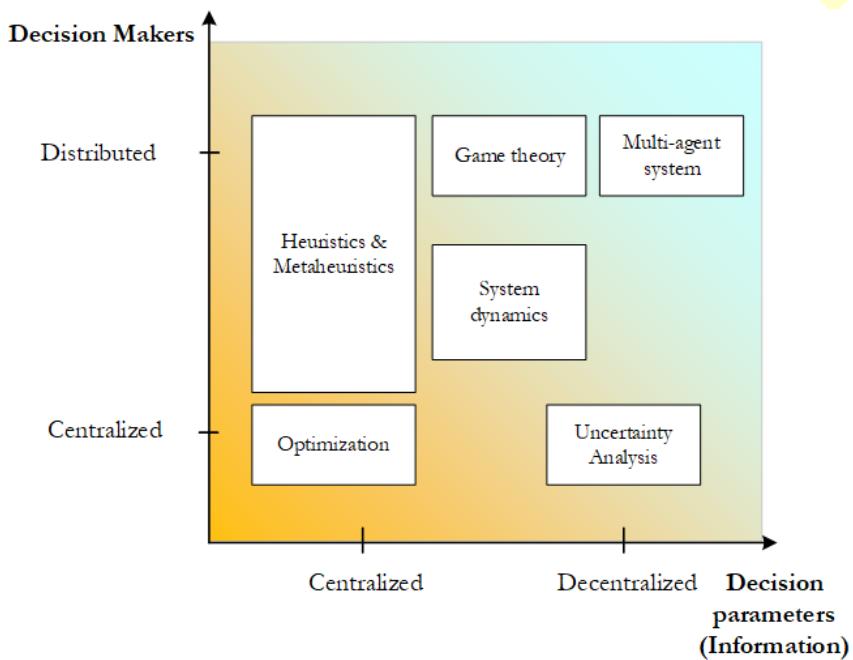


Figure 1.2: The Taxonomy of deision support methods.

### 1.2.1 Centralised information with centralised decision-makers (CC)

When a single DM knows all the decision parameters, the optimisation (e.g. integer linear programming, nonlinear programming, robust optimisation) is the most used decision support tool. When the complexity of an instance

arises, the optimisation runtime increases exponentially. Under these conditions, suboptimal heuristics or metaheuristics are good alternatives to get proper results within a brief time. A vast number of operational decisions belong to this group since a single person (i.e. the manager) has the responsibility for the decision and the ownership of the information. Some examples are:

- the definition of the storage assignment in an industrial storage system;
- the job scheduling on a production machine;
- the loading sequence of the containers on a vessel;
- the definition of the route for a fleet of trucks.

An intermediate decision support tool between CC and DC is the “system dynamics”, which defines connection and cause-effect relationships between the decision variables (i.e. the entities of a system) and the DMs. The relationships are usually nonlinear, and the system dynamic evolves at different time steps. The discrete event simulation (DES) is an implementation of system dynamics which evaluates the states of a system at discrete time lags.

### 1.2.2 Centralised information with distributed decision-makers (CD)

When the output of a single DM depends on the information owned by other DMs, it is necessary to identify different decision scenarios guessing the behaviour of the other actors. Usually, two tools go in this direction:

- game theory;
- problem-oriented heuristics or metaheuristics algorithms.

The game theory assesses the probability of the outcomes of a decision process evaluating all the alternatives in a tree structure. Game theory is mostly used to evaluate the payback of many actors cooperating or not cooperating in a supply chain. All the possibilities of cooperation or not cooperation are evaluated, finding the payback for each actor. Different approaches exist to identify the strategy which provides a higher benefit to a single actor or the entire system.

Heuristics and metaheuristics result suitable to provide sub-optimal solutions since they can be tailored to a specific instance of the problem.

The definition of the route of a truck is an example of this problem: the planner has all the information, but some decisions (e.g. route replanning) may happen due to exogenous facts as disruptions.

### 1.2.3 Decentralised information with distributed decision-makers (DD)

When the information and the ownership of a decision are decentralised, the outcome of the decision process depends on several independent decisions for each actor of the system. An important support tool is the agent-based modelling where each actor (i.e. an agent) has its own set of (partial) information to base its decision. Multi-agent simulation is similar to discrete event simulation. However, each actor implements its specific heuristics to make decisions based on the current state of the system from his point of view (i.e. based on its information set).

The participation of a company to a logistic tender for the outsourcing of the shipping or storage service (i.e. third-party logistics) is an example of DD; each company makes an offer based on its information. The outcome of the process depends on all the offers from all the companies participating in the tender.

### 1.2.4 Decentralised information with centralised decision-makers (DC)

When a single DM has the ownership of the decision but not all the information needed, the analysis of the uncertainty is a good choice to deal with the decision process. There are statistical models and methods to evaluate the risk behind a decision (e.g. Montecarlo simulation), and it is possible to infer information finding integrity rules among data.

Any CC case with incomplete knowledge on the decision parameters is a DC decision process. The incompleteness can be approached using statistical approaches as Montecarlo simulation or Markov chain to identify confidence intervals on a target variable. It is the case of cost-benefit analysis, where some cost distributions are skewed or hard to estimate.

This book focuses on centralised decision making in the supply chain, i.e. CC and DC cases.

## 1.3 Data Science Topology

The previous paragraph introduces a topology of the most used decision support methods without paying attention to the type of data they need. First of all, it is necessary to classify the input data depending on their structure:

- structured data. These data usually are from a relational database (e.g., SQL-based) designed using an entity-relationship (ER) model;

- Semi-structured data. These data come with a precise but non-relational structure (e.g. CSV, NO-SQL, XML and JSON<sup>1</sup> data);
- unstructured data. These data have no structure (e.g. text, pictures).

In the majority of cases, the literature on logistics and operations works with structured data. Usually, scholars and researchers first approach the methodology to solve a problem and then look for data to validate the approach (model-driven approach). These data are rarely available, and a considerable part of the literature on decision sciences relies on theoretical models validated by randomly generated instances. This research approach supports the development of robust and efficient algorithms, but it does not address real problems. Besides, there is a risk that the problem identified by the researchers does not exist in practice.

In this work, we use a different perspective, by introducing the data-driven approach, that is based on the classification in [2] (see Figure 1.3). Data-driven means “let data speak for itself”. The major effort is on understanding the meaning of data and developing a decision support method only after translating data into information.

During the last decade, scholars pay attention to data science because of the advent of big data. Big data is the collection, storage and manipulation of data that are [3]:

- Huge in storage volume;
- High in velocity (e.g., real time);
- Diverse in variety (e.g., structured/semi-structured/unstructured);
- Exhaustive in scope (i.e., getting the information of a whole population)
- Fine-grained in resolution and uniquely indexical;
- Relational (i.e., with attributes lining different datasets);
- Flexible (i.e., can easily add new attributes);
- Scalable (i.e., can easily add new records).

Under the perspective of big data, data science opens a new horizon for science. Traditionally, statistical techniques are used to extract knowledge from scarce and static data with weak relations between datasets. This analysis served to answer specific questions (generated under restrictive assumptions) by a researcher with a clear research question in mind [4]. This perspective is changed by big data whose characteristics provides a

---

<sup>1</sup> JavaScript Object Notation.

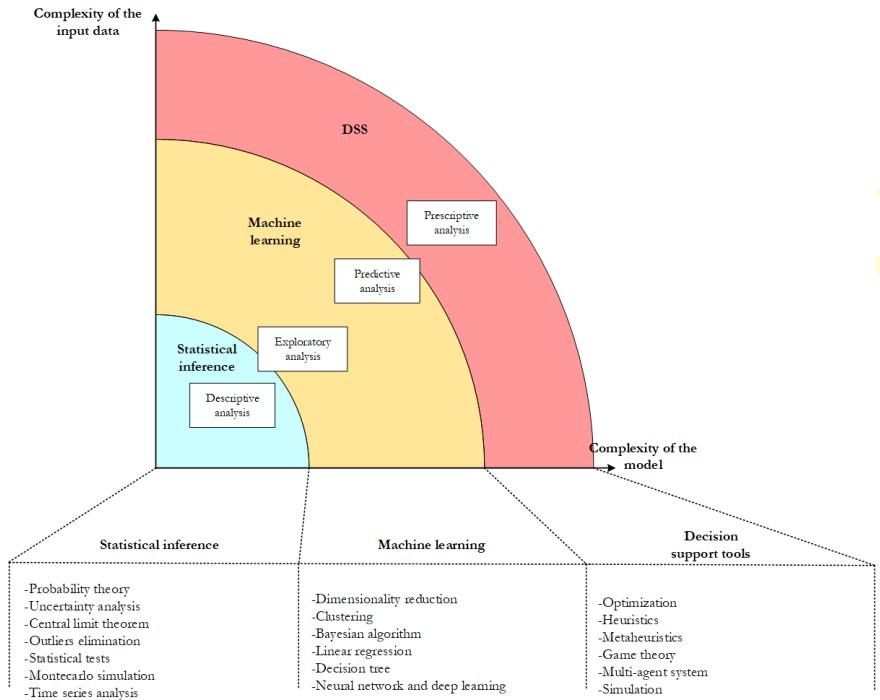


Figure 1.3: The relationship between data and analytical methods.

level of information to explore phenomena without a specific question in mind. Data science introduces a new research paradigm (see Table 1.2): Exploratory Science [5].

Paradigm	Nature	Form	When
First	Experimental science	Empiricism; describing natural phenomena	Pre-Renaissance
Second	Theoretical science	Modelling and generalisation	Pre-computers
Third	Computational science	Simulation of complex phenomena	Pre-Big Data
Fourth	Exploratory science	Data-intensive; statistical exploration and data mining	Nowadays

Table 1.2: The four research paradigms and their features.

The following chapters of this book aim at evaluating when data-driven

methods are suitable to address strategic decisions in the field of logistics and operations.

### 1.3.1 Fundamentals of data science

We start, first, from the fundamental theoretical concepts of data science [6].

1. Extracting useful knowledge from data to solve business problems can be treated systematically by following a process with reasonably well-defined stages.
2. Evaluating data-science results requires careful consideration of the context in which they will be used.
3. The relationship between the business problem and the analytics solution often can be decomposed into tractable subproblems via the framework of analysing expected value.
4. Information technology can be used to find informative data items from within a large body of data.
5. Entities that are similar with respect to known features or attributes are similar with respect to unknown features or attributes.
6. If you look too hard at a set of data, you will find something—but it might not generalise beyond the data you’re observing.
7. To draw causal conclusions, one must pay very close attention to the presence of confounding factors, possibly unseen ones.

The following paragraphs comes from the (2) and (3) proposing a framework for logistic and operational data. The other points are implemented in the following sections of this work that are dedicated to storage systems, distribution networks and production plants.

## 1.4 History of Logistics and Operations Research

Section 1.1 introduced the research field of this work, while sections 1.2 and 1.3 review the organisation of the decision support tools belonging to the fields of Decision Science and Data Science. The careful readers already will have noticed some overlaps between these two disciplines. Nevertheless, to understand these overlaps and to point out the direction of this work, it is necessary to introduce some history of the operations research (that, in our taxonomy, embeds both Decision Science and Data Science).

Logistics research was born in 1910 with the term “scientific management”, and it is possible to identify eight different periods where technologies, decision making and quantitative methods embed different roles within the same research domain [7]:

1. Scientific Management (1910-1945)
2. Scientific Method (1945-1965)
3. Management Information System (1965-1970)
4. Decision Support System (1970-1990)
5. Business Intelligence (1990-2005)
6. Analytics (2005-2015)
7. Artificial Intelligence (2015-today)
8. Non-Human Intelligence (future)

The scientific management of the work (1) starts together with the first assembly line to produce the Ford Model T. For the first time there is a scientific organisation of the work and the time and motion analysis are used to measure and control the productivity of the line. It is the beginning of the application of scientific tools to operations, the era of logistics research began.

In the second period (2), after World War II, the Von Neumann's architecture defines how to connect the component of a computer. This architecture changed the world, implementing the separation of the processor and the storage unit of a computer. This architecture is still used today in almost all IT applications, and it opened for the implementation of many decision support tools we still use nowadays.

The third period (3) has seen the development of microchips, which make affordable for companies to build information technology (IT) systems. Companies started to collect and store data on their operations into their system.

In the fourth period (4), these data are used to improve the decision process through:

- decision support systems (DSS), providing the decision-makers with a suggestion to their problem;
- expert system (ES), guiding the DM step by step to get a suggestion to his/her problem.

During the period of business intelligence (5), it becomes obvious that there is a value behind the data collected by the IT systems. In addition to companies, the world wide web started providing tons of data and information to a broad public.

The period of analytics (6) saw the realisation of the deductive process where information is obtained from tons of data and used to understand and improve the operations. Mathematics is used for understanding the data, but human intuition is still necessary to choose and implement the right decision. According to [7], period (6) was the current one. Nevertheless, the technological developments of the last few years move us to add two additional periods.

The period of artificial intelligence (7) starts in 2015 when a team of data scientist from Google develops a neural network able to play the game of Go better than the world champion of this game [8]. It is the advent of artificial intelligence. IT systems have already overcome the speed of the human brain (during period (6)), but now they are also able to think and react similarly to a real human brain. The following step is theoretical, but natural (8): when the artificial systems will be able to program themselves, they will be able to understand a problem autonomously and to get a solution faster and better than a real human brain. However, for the sake of our knowledge, this step in the history of IT is yet to come.

This work focuses on step (6) in the field of logistics and operations. We want to approach analytics to deeper investigate how human intuition and the value of logistics information match for proper management of a supply chain [9].

Accordingly to the periods defined above, step (6) should already be out-of-date. Nevertheless, the literature (see chapter 2) shows that few papers study analytics from a research point of view and their application in the logistics practice is still rare. The majority of supply chain systems stopped at the stage (4) where complex ES and DSS are seen as the final weapon against the inefficiencies of the supply chain.

Big companies use analytics to profile customers, price products and make financial analysis (the green shade in Figure 1). Nevertheless, they rarely studied analytics for engineering applications whose potential is even higher due to big data available from production, tracking and other activities in a supply chain.

## 1.5 A change in the perspective

Figure 1.1 illustrates the topology of logistics research and the traditional role of decision science and data science. Here we want to investigate their role from a control perspective. Traditionally, decision science is used to model the system to control (e.g. a supply chain or a plant), and data

science is used to get relevant data and keep the variables of the system under control. Figure 1.4 illustrates the differences between a data-driven and a model-driven approach [10], which will now be investigated, from a system control perspective. It is important to remark that a model is never the reality, but a useful approximation used to get information on it [11].

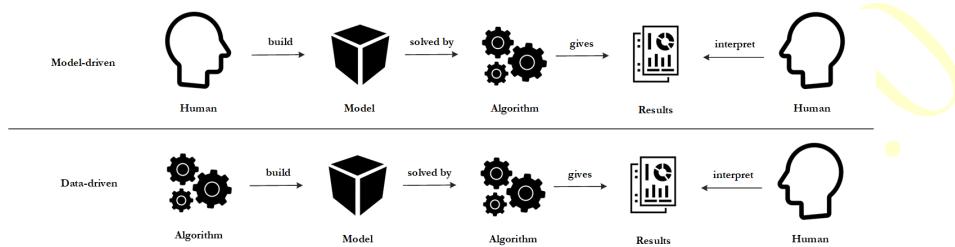


Figure 1.4: Model-driven vs. data-driven learning process.

The traditional model-driven approach can be seen as a system to control through a feedback loop:

1. the real system is modelled through decision science, defining decision variables based on the human understanding of the system;
2. the historical data of the system is the input parameter of the model;
3. the output of the model is analysed and used again to control the system.

Here we have a traditional model-driven approach (see Figure 1.5) where the “system model” is an ES or a DSS.

In this work, we introduce a different approach (see Figure 1.6). Data science is not only the base for the control of the system but also the design of the model. Figure 1.4 already illustrated the differences between model-driven to data-driven perspectives.

By using this approach, data science is used to train a model which is based on the real behaviour of the system and not on apriori human intuition. The outcome of the model which highlight patterns, correlations and relationships among data are discussed and interpreted by human and used as input for decision-making. The model is, then, used to predict the behaviour of the system in the future. The model is trained and updated each time new historical data from the real system are available, making the data-driven approach flexible and up-to-date. We can think of this fact as the change from a world where there was the need to design from scratch to a world where it is necessary to understand, control and improve the extant

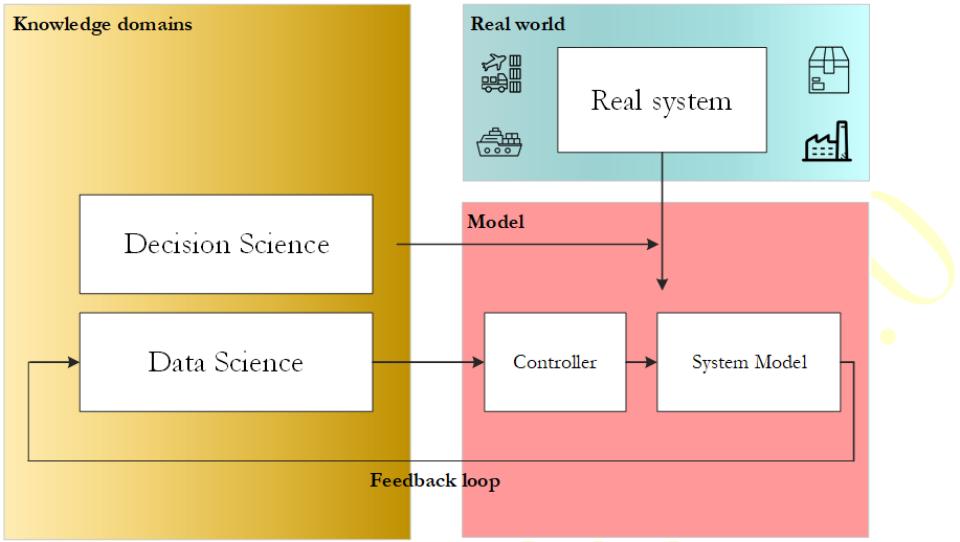


Figure 1.5: Traditional model-driven approach, based on feedback control.

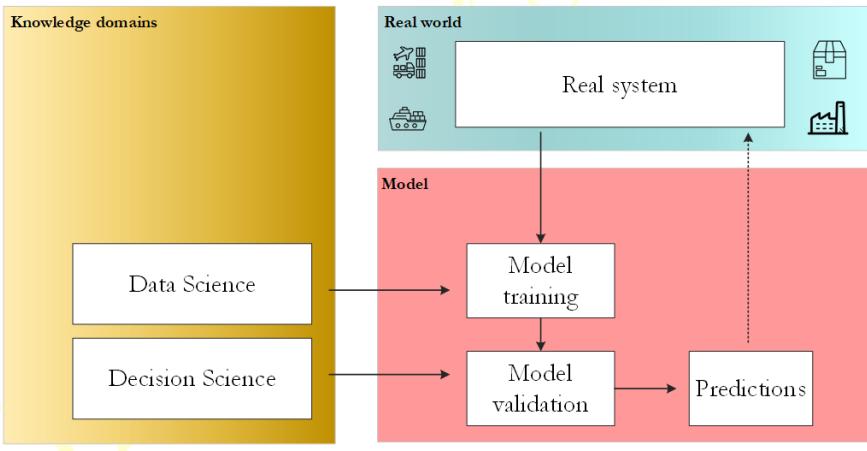


Figure 1.6: Novel data-driven approach, based on predictions.

processes. This is the reason why the following sections of this book analyse the control of a logistic system first.

This approach does not question the relevance of decision science, but it considers a new relationship between data science and decision science. We can exemplify this new relationship thinking of the design of a DSS. We

identify the four fundamental stages in the design of a DSS from a data-driven perspective (whose implementation will be the object of Parts III, IV and V of this book):

1. diagnosis of the real system (business-as-usual);
2. training of the model;
3. validation of the model;
4. deployment of the DSS.

The diagnosing of the business-as-usual scenario (1) is the phase where the DM assesses the problem, its environment, its criticalities, identify the managerial question to answer and the KPIs to measure the goodness of possible answers. In this phase, the DM takes a “snapshot” of the real system using statistical inference and data visualisation to:

- understand the business-as-usual process;
- identify the level of information at his/her disposal.

The training of the model (2) is the phase where the decision variables are identified and the model to predict their value is built (based on data science methods). The model is trained upon the available data from the real world. Differently from the model-driven approach, we do not assume any relationship between the input variables. In some case, assumptions are made on the value of each column to clean the data but, in general, we are interested in augmenting the knowledge we have on the data by only observing the data. For this reason, other assumptions are discouraged at this stage. The model will highlight the relationships between the input data and the decision variables.

Human intuition is then required to understand the output of the model and the link between the input data that is the validation of the model (3). Decision science methods can be implemented at this stage when the scenario is complex, and it requires additional information to be useful for the DM (e.g. it is necessary to merge the results of two models trained on different decision variables).

When the output of the model is considered confident enough for the DM (and from a statistical point of view), the model can be deployed into a DSS (e.g., a software) able to get input, train the model and present the output aiding the DM in his/her work.

This work proposes a novel approach where data-driven models are used together with the traditional scholars' model-driven approach. In other words, instead of modelling a logistic/operations process based on human

understanding, the modelling phase is based on the data available from the measurement of that process.

A human effort is required in the organisation of the measurement data and the understanding of the outcome of the model. In other words, we switch from the control of a logistic/operation process to the prediction of the outcome of that process. In this work, we use the terms “data-driven”, “logistics 4.0”, “smart logistics”, supply-chain 4.0” and “predictive logistics” to indicate this change of perspective.

## 1.6 Towards Predictive-Prescriptive Optimisation

Model-driven and data-driven models can co-exist. For this reason, we introduce the mathematical formulation of a problem addressed together by them [12]). We introduce here the general formulation of:

- data-driven optimisation;
- supervised learning;
- predictive-prescriptive problem.

Let introduce the following variables. Let  $y_1, \dots, y_N \in Y$  a quantity of logistics/operational interest (e.g. the demand);  $x_1, \dots, x_N \in X$  be an associated covariates  $X$  (e.g. search engine attention);  $z \in Z$  the decision variables.

In data-driven optimisation, we are interested in setting decision variables  $z$  to minimise an uncertain cost  $c(z; Y)$ . We only consider the prescriptive approach (i.e. defining the values of  $z$ ) and the historical data  $Y$ . The problem can be written as:

$$\min_{z \in Z} [c(z, Y)] \quad (1.1)$$

The solution to the problem can be found by sample average approximation solving the problem with different replication of the input variable  $Y$ :

$$\hat{z}_N^{SAA} \in \arg \min_{z \in Z} \frac{1}{N} \sum_{i=1}^N c(z; y^i) \quad (1.2)$$

The “pure” supervised learning approach, on the other side, does not involve prescription but focuses on the prediction (i.e. the forecast) of the future values of  $Y$  given the historical values of  $Y$  and  $X$ . In other words, we

are looking for a prediction  $\hat{m}_N(x)$  for the future value of  $Y$  after observing  $X = x$  and  $Y = y$ . The problem can be written as:

$$E[Y|X = x] \quad (1.3)$$

The problems defined by (1) and (3) matches together in the predictive-prescriptive problem i.e. fitting a machine learning (supervised) model  $\hat{m}_N \approx E[Y|X = x]$  and then solving a deterministic problem:

$$\hat{z}_N^{point-pred}(x) \in \arg \min_{z \in Z} c(z; \hat{m}_N(x)) \quad (1.4)$$

An optimal solution to the problem can be found as:

$$z^*(x) \in \operatorname{argmin}_{z \in Z} E[c(z; Y)|X = x] \quad (1.5)$$

Roughly speaking, predictive-prescriptive optimisation works:

1. finding a relationship (i.e. a predictive model) between the set of variables  $X$  and  $Y$ ;
2. exploring the cost  $c$  of different configurations of  $Y$ , predicted by different settings of  $X$ ;
3. Averaging over different replicates finding the set of decision variables  $z$  connected to the minimum of  $c$ .

## 1.7 The quality of data

Data-driven methods offer the technology to substitute some knowledge-based roles of a human by a machine. Nevertheless, we would like to know how much accurate can a machine be. In other words, if a company needs a logistic expert, they hire (or train) a person with a logistic background. How can we measure the degree of competence of a machine?

The outcomes of a data-driven process always depend on the quality of the input data [11]. We can think of the data production process as a manufacturing process with three stages:

1. raw data;
2. data processing;
3. data product.

The *data product* is the final product of a data collection process and, as for physical products, it is possible to measure its quality. The higher the quality of the data product, the more robust the results of a data-driven

Metric	Description
Accuracy	Do the data contain errors?
Timeliness	Are the data up-to-date?
Consistency	Do data have the same structure and format?
Completeness	Are there missing data?

Table 1.3: Metrics to measure the quality of data.

model fed by the data product. We identify four metrics to measure the quality of a data product (see Table 1.3).

The accuracy identifies if data corresponds to the real values. Timeliness describes if data records are up-to-date (currency) and their update frequency (volatility). Consistency regards the robustness of data format and data structure. Completeness measures the fraction of missing data over the total information content. It is important to check the quality of the input data before starting with data-driven modelling since poor data quality is the most common cause of bad predictions.

## Bibliography

- [1] G. Stecca, G. Lanza, and S. Peters, “Optimization in Manufacturing,” in *CIRP Encyclopedia of Production Engineering*, 2019.
- [2] L. Irv, D. Brenda, J. Christer, and D. Christopher, “The analytics journey,” *Analytics Magazine*, vol. November/D, pp. 11–13, 2010.
- [3] R. Kitchin, “Big Data, new epistemologies and paradigm shifts,” *Big Data & Society*, vol. 1, no. 1, p. 205395171452848, 2014.
- [4] Miller, “The data avalanche is here. Shouldn’t we be digging?,” *Journal of Regional Science*, vol. 50, no. 1, pp. 181–201, 2010.
- [5] T. Hey, S. Tansley, and K. Tolle, *The Fourth Paradigm*. 2009.
- [6] F. Provost and T. Fawcett, “Data Science and its Relationship to Big Data and Data-Driven Decision Making,” *Big Data*, vol. 1, no. 1, pp. 51–59, 2013.
- [7] M. J. Mortenson, N. F. Doherty, and S. Robinson, “Operational research from Taylorism to Terabytes: A research agenda for the analytics age,” *European Journal of Operational Research*, vol. 241, no. 3, pp. 583–595, 2015.

- [8] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, “Mastering the game of Go with deep neural networks and tree search.,” *Nature*, vol. 529, no. 7587, pp. 484–9, 2016.
- [9] M. Arvan, B. Fahimnia, M. Reisi, and E. Siemsen, “Integrating human judgement into quantitative forecasting methods: A review,” *Omega (United Kingdom)*, vol. 86, pp. 237–252, 2019.
- [10] D. Hedgebeth, “Data-driven decision making for the enterprise: An overview of business intelligence applications,” *Vine*, vol. 37, no. 4, pp. 414–420, 2007.
- [11] B. T. Hazen, C. A. Boone, J. D. Ezell, and L. A. Jones-Farmer, “Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications,” *International Journal of Production Economics*, vol. 154, pp. 72–80, 2014.
- [12] D. Bertsimas and N. Kallus, “From Predictive to Prescriptive Analytics,” 2014.

Draft Version 1.0

## Research Background

*The line between statistics, computer science and engineering is getting thinner.*

This chapter reviews the relevant literature identifying the academic position of this work. As introduced, this work provides advances in the field of engineering applied to the “system analysis” (see Figure 1.1) of a supply chain system by using data-driven technologies.

Before starting with this data-driven journey, it is necessary to introduce a brief glossary with the keywords used to gather academic papers by scholars and practitioners (see Table 2.1).

Glossary	Description
Data-science	The branch of science developing models to get information from data.
Data-driven	An approach to model the reality based on patterns within data. Here it is used to indicate both machine learning and deep learning.
Machine Learning	Data processing algorithms based on the theory of statistical models and the power of modern computers to build predictive models.
Deep learning	Complex data processing algorithms to build models on multiple (hidden) layers extracting data feature and patterns.
Learning algorithm	Any machine learning or deep learning algorithm.
Industry 4.0	The organisation of operations and logistics based on data, connectivity and new technologies (e.g., robotics).
Smart Factory	Production plant with huge data collection via sensor connected to the internet.

Figure 2.1: Glossary of data science terms.

## 2.1 Academic background

The data-driven approach is gathering an increasing interest in literature during the last decade. In particular, the number of research paper approaching logistics research issues from a data-driven perspective is increasing exponentially [1, 2]. Figure 2.2 illustrates research trends on Scopus reporting the number of research papers published in international journals resulting from four research queries. The world “machine learning”, “deep learning” and “data-driven” are used together with the word “industry” to check how literature evolves approaching this topic from an industrial perspective. The last trend is focused on the field of supply chain management. This trend is similar, but with a lower absolute number of papers.

This analysis shows that, in the last few years, scholars are paying more attention to data-driven approaches applied to industrial fields. Nevertheless, the amount of contributions in the field of industrial engineering and supply chain is still limited. A closer investigation of the industrial engineering field confirms the need for a broader study of data-driven methods in this field [3]. Figure 2.3 categorises the topics of industrial engineering where data-driven approach is used within a sample of 23 journal articles.

Industrial engineering shares engineering and decision science features. Figure 2.4 (defined on a sample of 52 journal articles) shows that both of them are areas of research where analytics and data-driven modelling are established methodologies [4].

The literature lacks of a comprehensive approach of data-driven models in the field of logistics research as tools for system analysis [5, 6]. Many papers show frameworks for the application of analytics and data-driven methods to sub-topics as safety engineering, industrial database design data structures and real-time data collection with internet-of-things (IoT) industrial devices [7, 8]. Nevertheless, none of them proposes a holistic point of view on logistics and operations.

On the other side, they clearly recognise a value behind data [9, 10] and the importance of smart logistics and smart factories whose capability is to adapt decision making dynamically to predict future scenarios.

This new approach involves a large mass of data exchanged between several actors [11, 12]. Literature suggests robust methodologies to manage the information flow of data [13]. The outcome of these data structures supports the development of the smart factory, i.e. a manufacturing plants where information is used to control the process and to make decisions on the future processes (i.e. the design) [14].

In [15], they identify three main research issues to improve the operations in a smart factory using data-driven methods:

1. modelling the theory and method of smart factory;

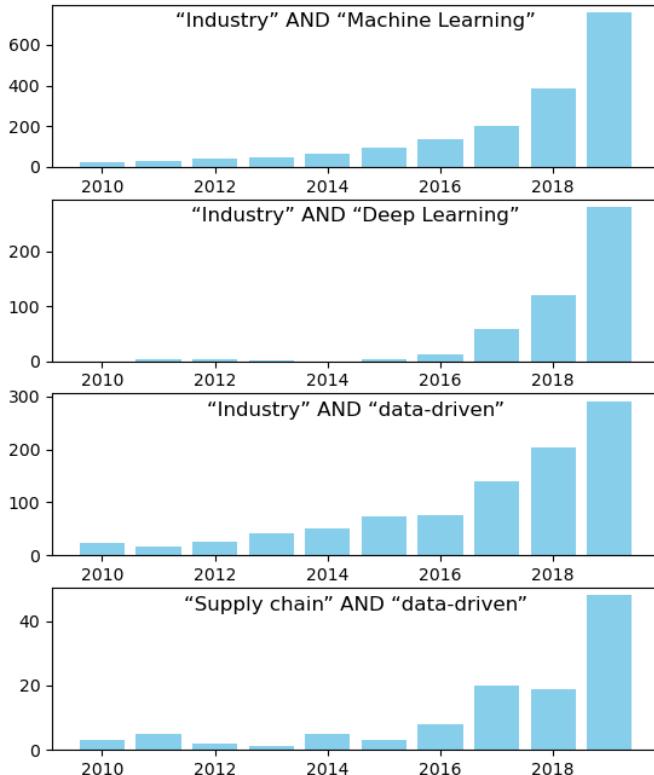


Figure 2.2: Literature trends corresponding to different research queries on Scopus.

2. knowledge discovery and knowledge management based on industrial big data analysis;
3. adaptive scheduling and optimisation of the smart factory.

This book focuses on the key issues (2) and (3). Often, literature focuses only on theoretical frameworks for the application of data-driven methods [16]. In this work, we aim at a comprehensive approach for logistics and operations being practice-ready for a company proposing areas of application of data-driven methods.

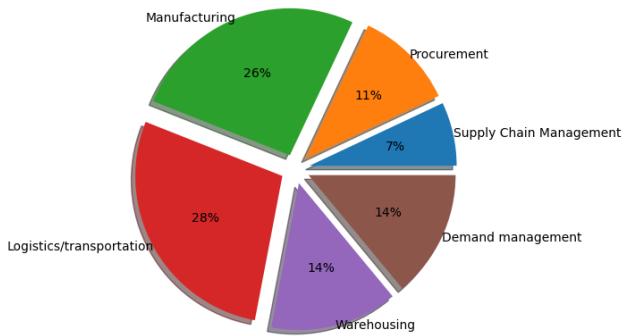


Figure 2.3: The fields of application of data-driven algorithms in the industrial engineering sector.

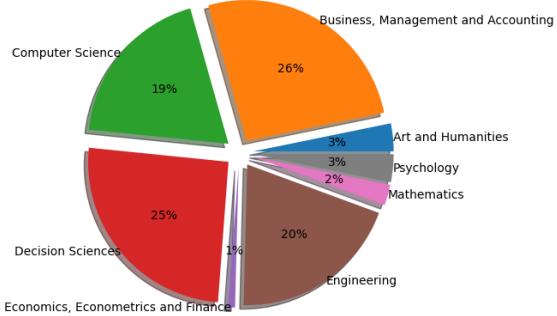


Figure 2.4: The fields of application of data-driven algorithms.

## 2.2 Industrial background

In addition to the academic relevance, we want to highlight the rationale of this work considering the structure of the external industrial sector: data science is the present, not the future. In the last decade, companies started creating a new professional role called “data scientist” able to generate value from industrial data. From this perspective, the statistic is the his-

tory, machine learning is the past, and deep learning is the present [17]. These methodologies have already been established and implemented in many business areas (e.g. marketing, finance) with or without the contribution of the research community since a number of companies recognised their data could be used to improve the efficiency, reliability and sustainability of their business [18, 19].

Looking at the industrial practice, we notice that there is an enormous potential for data-driven application in the field of logistics. The industry is reacting to this new trend with significant investments in data-driven projects [20, 21]. Managers and directors from logistics and operations identify in the big data analytics the main tool to handle decision and processes in the future [22]. Nevertheless, a significant gap exists between large companies, able to train data scientist themselves, and small and medium enterprises (SME) that cannot afford this kind of investment [23].

It is the case of large third-party providers that develop machine learning and artificial intelligence tools to support their business [24]. They identify a precise workflow to check if a data-driven approach can be used to generate value, improving a logistic process (see Figure 2.5) with two main activities:

1. creating new knowledge;
2. reducing costs.

These two objectives are the reason why companies started collecting data on their processes and hiring data scientists. The flowcharts in Figure 2.5 identify the necessary ingredients to get success from a data-driven approach. The quantity and quality of the data is, obviously, a crucial issue to work with a data-driven approach. Besides, the data collected must be relevant to solve a decision problem. Finally, if an algorithm can find patterns better than a human does, the data-driven approach is a good choice to create knowledge and reduce costs.

Both these objectives are reachable when data scientists have both analytic skills and profound knowledge of the industry-specific domain [24]. When these two characteristics come together, data-driven application leads these companies to success while some small logistics companies still strive to work with pen and paper.

Logistics companies identify the need to anticipate the behaviour of the market with predictions on:

- lead times;
- transport capacity;
- customer orders.

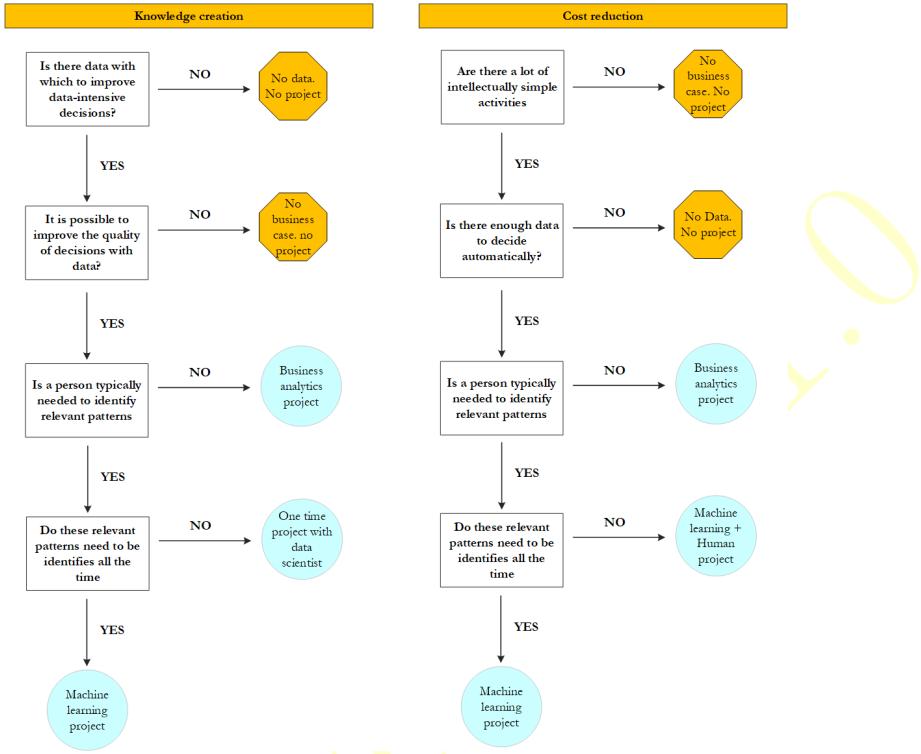


Figure 2.5: Checklist for the implementation of a data-driven project adapted from [24].

Predictive logistics can forecast the value of these metrics anticipating market demand. Traditional logistics is way less efficient and unable to anticipate market behaviour [25]. Data-driven tools are not limited to transportation activities. Recent applications in the manufacturing and storage industry have seen several predictive models to support the operations [26, 27, 28].

While a small number of big companies recognise the value of its operational data, researchers strive to get data to test their assumptions enriching the literature and public knowledge. Besides, the more these companies see a profit in their data-driven approaches, the less they are willing to share data with the research community. This happens because they get a competitive advantage from the utilisation of their data and sharing their valuable knowledge with the public would not be safe for their business.

In this sense, our work is even harder since it is difficult not only to design new methods but also to get data to test them and keep them useful in the real world. For this reason, we notice a gap between practice and

academic research, especially when internal research units of multinational companies hold the data and autonomously lead the research activities.

Also, the application of data-driven models in industrial practice is still limited in the field of logistics and operations [29]. We think that academic research has the responsibility to explore the role and the potential of analytics and data-driven methods in the logistics and operations fields.

## Bibliography

- [1] M. A. Moktadir, S. M. Ali, S. K. Paul, and N. Shukla, "Barriers to big data analytics in manufacturing supply chains: A case study from Bangladesh," *Computers and Industrial Engineering*, vol. 128, no. April 2018, pp. 1063–1075, 2019.
- [2] K. Spanaki, Z. Gürgüç, R. Adams, and C. Mulligan, "Data supply chain (DSC): research synthesis and future directions," *International Journal of Production Research*, vol. 56, no. 13, pp. 4447–4466, 2018.
- [3] T. Nguyen, L. ZHOU, V. Spiegler, P. Ieromonachou, and Y. Lin, "Big data analytics in supply chain management: A state-of-the-art literature review," *Computers and Operations Research*, vol. 98, pp. 254–264, 2018.
- [4] S. Gupta, S. Modgil, and A. Gunasekaran, "Big data in lean six sigma: a review and further research directions," *International Journal of Production Research*, vol. 0, no. 0, pp. 1–23, 2019.
- [5] G. Wang, A. Gunasekaran, E. W. Ngai, and T. Papadopoulos, "Big data analytics in logistics and supply chain management: Certain investigations for research and applications," *International Journal of Production Economics*, vol. 176, pp. 98–110, 2016.
- [6] K. Lamba and S. P. Singh, "Modeling big data enablers for operations and supply chain management," *International Journal of Logistics Management*, vol. 29, no. 2, pp. 629–658, 2018.
- [7] L. Huang, C. Wu, B. Wang, and Q. Ouyang, "Big-data-driven safety decision-making: A conceptual framework and its influencing factors," *Safety Science*, vol. 109, no. May, pp. 46–56, 2018.
- [8] Y. Zhang, Z. Guo, J. Lv, and Y. Liu, "A Framework for Smart Production-Logistics Systems Based on CPS and Industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 9, pp. 4019–4032, 2018.

- [9] S. Balandin, S. Andreev, and Y. Koucheryavy, “Big Data Governance for Smart Logistics: A Value-Added Perspective,” in *15th international conference, NEW2AN 2015 and 8th conference, ruSMART 2015 St. Petersburg, Russia*, 2015.
- [10] D. Uckelmann, “A definition approach to smart logistics,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5174 LNCS, pp. 273–284, 2008.
- [11] A. Kawa, “SMART Logistics Chain,” in *ACIIDS*, pp. 432–438, 2012.
- [12] P. M. Singh, M. J. Van Sinderen, and R. J. Wieringa, “Smart logistics: An enterprise architecture perspective,” *CEUR Workshop Proceedings*, vol. 1848, pp. 9–16, 2017.
- [13] H. Tran-Dang and D. S. Kim, “An Information Framework for Internet of Things Services in Physical Internet,” *IEEE Access*, vol. 6, pp. 43967–43977, 2018.
- [14] M. Hajdul and M. Cudzilo, “Information Technologies in Environmental Engineering,” *Information Technologies in Environmental Engineering*, vol. 3, pp. 501–513, 2011.
- [15] Y. TAN, F. QIAO, S. YANG, J. WANG, and L. SHI, “Engineering management for high-end equipment intelligent manufacturing,” *Frontiers of Engineering Management*, vol. 5, no. 4, p. 420, 2018.
- [16] V. L. da Silva, J. L. Kovaleski, and R. N. Pagani, “Technology transfer in the supply chain oriented to industry 4.0: a literature review,” *Technology Analysis and Strategic Management*, vol. 31, no. 5, pp. 546–562, 2019.
- [17] HubSpot, “Artificial Intelligence Is here - People Just Don’t Realize It,” 2016.
- [18] D. L. M. Nascimento, V. Alencastro, O. L. G. Quelhas, R. G. G. Caiado, J. A. Garza-Reyes, L. R. Lona, and G. Tortorella, “Exploring Industry 4.0 technologies to enable circular economy practices in a manufacturing context: A business model proposal,” *Journal of Manufacturing Technology Management*, vol. 30, no. 3, pp. 607–627, 2019.
- [19] P. Trkman, K. McCormack, M. P. V. De Oliveira, and M. B. Ladeira, “The impact of business analytics on supply chain performance,” *Decision Support Systems*, vol. 49, no. 3, pp. 318–327, 2010.
- [20] W. economic forum, “These charts will change how you see the rise of artificial intelligence,” 2017.

- [21] R. Y. Zhong, S. T. Newman, G. Q. Huang, and S. Lan, "Big Data for supply chain management in the service and manufacturing sectors: Challenges, opportunities, and future perspectives," *Computers and Industrial Engineering*, vol. 101, pp. 572–591, 2016.
- [22] B. Roßmann, A. Canzaniello, H. von der Gracht, and E. Hartmann, "The future and social impact of Big Data Analytics in Supply Chain Management: Results from a Delphi study," *Technological Forecasting and Social Change*, vol. 130, no. November 2017, pp. 135–149, 2018.
- [23] R. Dubey, A. Gunasekaran, S. J. Childe, C. Blome, and T. Papadopoulos, "Big Data and Predictive Analytics and Manufacturing Performance: Integrating Institutional Theory, Resource-Based View and Big Data Culture," *British Journal of Management*, vol. 30, no. 2, pp. 341–361, 2019.
- [24] Kü, M. Ckelhaus, G. Chung, B. Gesing, G. Steinhauer, M. Heck, and K. Dierckx, *Artificial Intelligence in Logistics*. DHL Customer Solution & Innovation, 2018.
- [25] LOGISTICA EFFICIENTE, "Come Logistica 4.0 rivoluzionerà la SCM (e come conviverci)," 2019.
- [26] Logisticofthings, "HOW AI WILL REVOLUTIONIZE LOGISTICS," 2017.
- [27] Package.ai, "Back Office Automation For Last Mile Logistics," 2017.
- [28] W. industrial Reporter, "John Deere and IBM Pilot to Create a Futuristic Model for Manufacturing," 2016.
- [29] M. S. Garver, "Threats to the Validity of Logistics and Supply Chain Management Research," *Journal of Business Logistics*, vol. 40, no. 1, pp. 30–43, 2019.

Draft Version 1.0

## The Information Framework

*Pure mathematics is, in its way,  
the poetry of logical ideas.*

Albert Einstein

Supply chains systems involve a large number of connected entities, resources, actors and flows. In this chapter, we introduce a framework to model the entities of storage systems, distribution networks and production plants. In particular, this work investigates the room for the application of data science in the field of logistics and operations. For this reason, we are interested in mapping the information relationships between these entities.

Here we introduce an ontology of entities and metrics. This ontology meets the *fractal manufacturing system* philosophy (see [1]), where each element of a supply chain can be seen as a black box with inputs and outputs. The elements can be aggregated or disaggregated into other black boxes. In the following parts of this book, we will re-define this ontology by applying it to smaller blocks of the supply chain, with specific references to storage systems, distribution networks and production plants.

### 3.1 Ontology

We base our ontology on [2], a milestone book in logistics and operations science. It was the first providing a scientific framework to model a factory.

#### Entities

We identify the following entities.

**Part (*i*):** it is a piece of raw material, component, subassembly or assembly. Where:

- Raw material is a part purchased out of the system.
- Component is an individual piece assembled into more complex products.
- Subassembly is an assembled unit further assembled into more complex products.
- Assembly/final assembly/finished product is the fully assembled product.

**Processing node (*j*):** we define the processing node using the concept of fractal manufacturing. From this perspective, a processing node is any entity that can be modelled as a black box with input and output. A production machine, a production line, a packing machine, a manufacturing robot, a production plant, a storage system, a port, a logistic platform, a train terminal are all examples of entities working as a processing node. A processing node usually performs value-added activities on a part.

**Edge (*j, k*):** it is a physical connection between processing nodes. Aisles and conveyors are edges in a production plant; while railways, rivers and roads are the edges of a distribution system.

**Vehicle (*v*):** a vehicle is a handling unit able to transport one or more parts from a processing node to another. Operators, forklifts, AGVs, trucks, trains, vessels are all examples of a vehicle.

**Consumable (*s*):** a consumable is a material that is used by a processing node or a vehicle to perform its work. A consumable is generally responsible for the variable costs of the processing node or a vehicle (e.g. energy and fuel).

**Route (*e*):** is the sequence (ordered set) of processing nodes  $j = 1, \dots, m$  visited by a vehicle to add value on a part.

**Order (*o*):** is a processing request on a part sent by a customer.

**Job (*b*):** is the response to the market demand from the operations side. Production batches and transportation loads are examples of a job.

**System network  $G(V, A)$ :** the system network is the set of processing nodes  $j \in V$  and edges  $(j, k) \in A$  that connect all the entities involved in the supply chain system.

## Metrics

We identify the following metrics to assess the performances of a processing node  $j$ .

**Throughput ( $TH_j$ ):** the throughput of a processing node is the average output per unit of time (e.g., parts per hour).

**Work in process ( $WIP_j$ ):** is the number of parts (i.e., the level of inventory) being processed/waiting for processing by a processing node.

**Work in process ( $WIP_{jk}$ ):** is the number of parts (i.e. the inventory position) being transported on edge  $(j, k)$  by a vehicle  $v$ .

**Capacity ( $C_j$ ):** is the upper bound of the throughput.

**Capacity ( $C_v$ ):** is the maximum capacity of a vehicle  $v$ .

**Utilisation ( $U_j$ ):** it is the fraction of time that a processing node is not idle for lack of demand.

**Utilisation ( $U_v$ ):** is the fraction of transportation space that a vehicle uses due to the variability of the transportation demand.

**Lead time ( $LT_e$ ):** is the time allocated for a given route.

**Cycle time ( $CT_e$ ):** is the average time from the release of a job to the end of its route.

**Service level ( $SL_e$ ):**  $\text{Prob}\{\text{cycle time} \leq \text{lead time}\}$

Little's law (see equation (3.1)) defines a basic relationship linking the three main metrics, i.e.  $WIP$ ,  $TH$  and  $CT$ .

$$WIP_j[pz] = TH_j \left[ \frac{pz}{h} \right] \times CT_{e:e \in \{j\}}[h] \quad (3.1)$$

Usually, managers have data on the entities at their disposal, but they need to get information on the metrics. This data can be dirty or incomplete. For this reason, we build upon this ontology to show how to infer the properties of entities and metrics when data are incomplete or missing.

Our approach is highly generalisable and based on a few standard rules of a supply chain, with minimum bias. For this reason, the theoretical effort of this work stands in the definition of a structure for industrial data, the understanding of the role of data, and the consistency rules among different features that can be used to increase the value of an incomplete dataset.

The modelling of a system accordingly with a robust data structure allows using only logistics-relevant data and base the model on robust relationships. The following sections define this data structure for storage, distribution and production systems. We first introduce the law of physics to analyse the material flows of a supply chain. Then, a theorem regulating the consistency of the information flow is introduced and demonstrated.

## 3.2 The physics of a supply chain

The laws of physics can be used to model the material flows between the entities of a supply chain. We can think of a processing node  $j$  as a physical system with an inventory  $WIP_j$  subjected to “forces” modifying its value. It is not difficult to imagine that the value of  $WIP_j$  can change when:

- some forces *pull* a number of items from  $j$ ;

- some forces *push* a number of items into  $j$ .

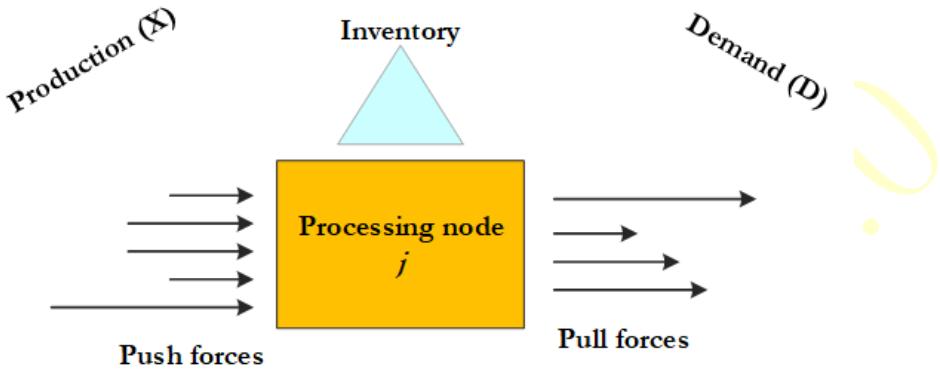


Figure 3.1: Physical model of a processing node.

Figure 3.1 presents a scheme of the physical model of a processing node where the forces generated by the production and the demand modify the value of the inventory.

The concept of a *force* modifying the inventory matches very well with the definition of pull and push production philosophy. The analysis of push and pull systems is out of the scope of this book. We will limit to describe the dynamics and the rules of these systems using the idea of a *force* associated with the production and the demand. To model  $j$ , we are just interested in defining how these forces are linked to the value of  $WIP_j$ .

The purpose of a supply chain is to provide a physical connection between demand and offer. A supply chain connects several processing nodes to develop a product or a service to the final consumer. Processing nodes work at a variable rate, and the only way to keep them synchronised is by using buffers. There are three types of buffers with different nature:

- **inventory buffer** is a number of goods stored between the production and the demand of a processing node;
- **time buffer** is an amount of queuing time between the time instant when the order of a customer occurred and the time instant when it is finally satisfied;
- **capacity buffer** is an amount of production capacity calculated as the difference between the capacity of a working station and its average demand.

These types of buffers are interrelated and react to the demand and production forces. Time and capacity buffers are generally defined in the design phase of a network since they are linked with the type of physical assets of the supply chain. Inventory buffers are generally more flexible and more used since they are cheaper than the two others.

We use an approach based on dynamic and stochastic system equations to generalise the Little's law, and understand the relationships between the variables describing the physics of a supply chain.

We can model a production-inventory model as follows. Let consider the inventory of a processing node measured in a number of parts. We are interested in knowing the value of the inventory function  $q$ . The inventory function depends on the demand function  $d$ , and the production function  $x$ . Figure 3.2 identifies samples demand and production functions.<sup>1</sup> The demand curve is generally more volatile, while the production curve is more stable since the output of a processing node is hardly flexible as related to its assets.

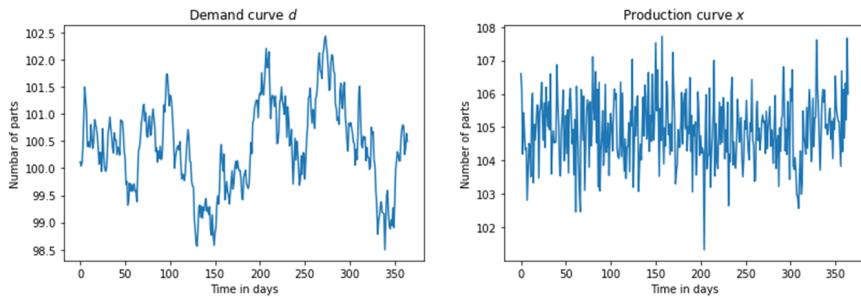


Figure 3.2: Examples of demand and production functions.

Given  $x$ ,  $d$ , and an initial inventory  $q_0$ , it is possible to define the inventory function  $q$ .

$$q_t = q_{t-1} + p_t - d_t \quad (3.2)$$

The difference between the demand and production functions define the *momentum*  $p$  of the production node. The momentum indicates the direction in which the inventory function  $q$  moves, and it is measured as the number of parts per unit of time.

$$p = x - d = \dot{q} \quad (3.3)$$

We identified an approach to model a supply chain using the law of physics. In particular, there is a relationship between the flows (demand  $d$

---

<sup>1</sup>The source code of Figure 3.2 is available [here](#).

and production  $p$ ) and the inventory  $q$ . We can assume that some forces (may them be "push" or "pull" forces) play a role in changing the value of the inventory function  $q$ . Table 3.3 illustrates the parallelism between classical mechanics and the physics of a supply chain.

Parameter	Physics of a Supply Chain	Classical Mechanics
$q$	Inventory	Coordinates in the space $(x, y, z)[m]$
$p$	Productivity	Speed $\left[\frac{m}{s}\right]$ or momentum $\left[Kg \times \frac{m}{s}\right]$
$\dot{p}$	Movements	Acceleration $\left[\frac{m}{s^2}\right]$

Figure 3.3: Examples of demand and production functions.

[3] demonstrated that the energy conservation law is applicable to the physics of a supply chain to control the behaviour of the inventory curve  $q$ . The Lagrangian is used to express the equation of the force  $\dot{p}$ , representing the "push" and "pull" forces of a supply chain, i.e. the movements of the supply chain.

$$\dot{p} = \frac{\partial L(q, \dot{q})}{\partial q} \quad (3.4)$$

Where  $L(q, \dot{q}) = \frac{1}{2}p^2 - V(q)$ . The Lagrangian express a difference between kinetic and potential energy. The potential energy is defined using a linear potential  $V(q) = -F_0q$ . This means that, as it works with the force of gravity, the perturbation of the value of the inventory depends on the value of the inventory  $q$  with a linear law. We can assume conservation of energy using the Hamiltonian principle (the Principle of Least Action). This principle states that nature always chooses the lowest energy path (i.e. the difference between kinetic and potential energy) between all the feasible ones defined by forces. The inventory function  $q$ , is then, defined from the Lagrangian as:

$$S[q] = \int_{t_1}^{t_2} L(q, \dot{q}, t) dt = 0 \quad (3.5)$$

[3] shows how choosing the value of  $F_0$  to control the value of inventory  $q$ .

### 3.3 The information of a supply chain

Section 3.2 demonstrated that a physical relation exists between three variables of a supply chain: the inventory  $q$ , the productivity  $p$ , and the movements  $\dot{p}$ . Physics is the core of science, and we can always rely on physical

laws. Nevertheless, in practice, we may not have the possibility to apply these laws since data are collected with different granularities (i.e. referred to different processing nodes) or with inconsistencies. For this reason, this section introduces an original information framework to build  $q$ ,  $p$ , and  $\dot{p}$  using the available data. We move the focus from the physical relationship to the information relationship existing between these three functions.

### 3.3.1 Information framework

While defining data structures to support logistics, databases are usually designed using an entity-relationship (ER) model [4]. The focus of ER models is on the entities, i.e. the ones we defined in section 3.1. Here we propose a different perspective, focusing on the physical logistic phenomenon, i.e. the variation in the demand  $d$ , production  $x$  and inventory  $q$  due to the forces  $\dot{p}$ , regardless of the entities generating these forces.

This change of perspective enhances the flexibility of this modelling approach since it is often difficult to collect data from all the entities involved in a supply chain and to connect them into an ER model. On the other side, the effect of the forces generated by these entities is clearly measurable on the production and demand functions.

For the sake of clarity, we abandon the dot notation, and we define:

1. a movement function  $M_i(t)$  to represent the forces  $\dot{p}$ , applied to a part  $i$ ;
2. a productivity function  $P_j(t)$  to represent the speed  $p$  at which the forces change the value of the inventory  $q$ ;
3. an inventory function  $I_i(t)$  representing the inventory  $q$  of part  $i$ .

The  $M$  function has different granularity (e.g. order, vehicle, terminal, network) depending on the measurement system and the data collection system used. The most granular is a movement called by a single order  $o$  of a part  $i$  involving a processing node  $j$  and a vehicle  $v$ . The function uses:

- a positive value to describe the physical movement of a quantity  $q$  from a processing node to a vehicle (load movement:  $j \rightarrow v$ );
- a negative value to describe a physical movement of a quantity  $q$  from a vehicle to a processing node (offload movement:  $v \rightarrow j$ ).

$$M_o^{j,v}(t) = \begin{cases} q & \text{if } t = t_{IN}, \\ -q & \text{if } t = t_{OUT}, \\ 0 & \text{otherwise.} \end{cases} \quad (3.6)$$

Where  $t$  measures the time;  $q$  is the quantity (e.g. the number of parts) involved in the physical movement;  $t_{IN}$  is the timestamp when the part  $i$  is loaded on a vehicle  $v$ ;  $t_{OUT}$  is the timestamp when it is unloaded from it.

The inventory function  $I$  describes the inventory position (e.g. the number of parts) of a vehicle  $v$  or a processing node  $j$ . The function  $I$  is linked to  $M_o^{j,v}(t)$  by aggregating the movements of single orders.

$$I_j(t) = I_j(t - \epsilon) - \sum_i \sum_v M_o^{j,v}(t) \quad (3.7)$$

$$I_v(t) = I_v(t - \epsilon) + \sum_i \sum_j M_o^{j,v}(t) \quad (3.8)$$

The parameter  $\epsilon$  is a sufficiently small time sampling unit (e.g. a minute). It is essential to consider the initial inventory  $I_j(t = 0)$  and  $I_v(t = 0)$  to define the inventory functions correctly. Figure 3.4 represents an example of the movements and inventory functions for a part  $i$ .

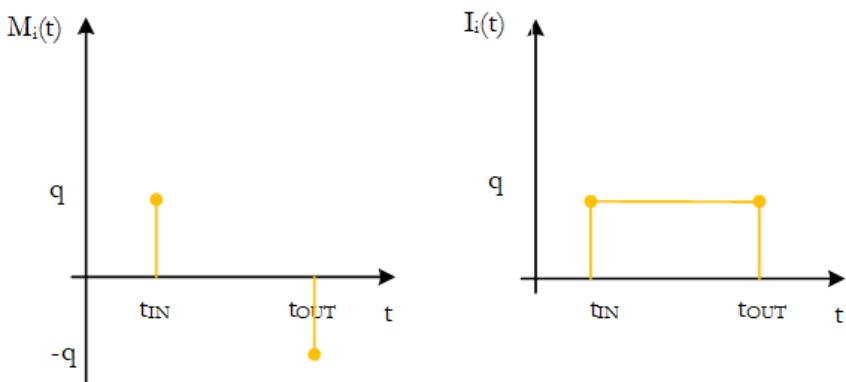


Figure 3.4: Example of movements and inventory function.

The productivity function  $P$  defines the motion equation of a processing node  $j$ , i.e. the speed of the changes in its inventory position. Two distinct  $P$  functions exist to describe the loads (IN), and the offloads (OUT) speed of a processing node. The productivity function  $P$  is defined, starting from the cumulative function of the movements  $Q$ .

$$Q_j^{IN}(\tau) = \sum_i \sum_v \{M_i^{j,v} | M_i^{j,v} > 0, t \leq \tau\} \quad (3.9)$$

$$Q_j^{OUT}(\tau) = \sum_i \sum_v \{M_i^{j,v} | M_i^{j,v} < 0, t \leq \tau\} \quad (3.10)$$

The productivity  $P$  at a time instant  $t^*$  is given by:

$$P_j^{IN}(t^*) = \frac{dQ_j^{IN}(\tau)}{d\tau}|_{t^*} \quad (3.11)$$

$$P_j^{OUT}(t^*) = \frac{dQ_j^{OUT}(\tau)}{d\tau}|_{t^*} \quad (3.12)$$

Movements usually have redundant information on vehicles ad processing nodes. We overcome this redundancy by splitting this information into two separate functions of inventory  $I$  and productivity  $P$ . Productivity function, in fact, does not suffer redundancies, and it converges to a value determined by the physical assets of a terminal.

At this stage, we want to demonstrate the relationship between these three functions. For this reason, we consider the movement function at the granularity of a processing node  $M_j(t) = \sum_v \sum_o M_o^{j,v}(t)$  and we prove the following theorem that introduces consistency rules between the functions  $M_j$ ,  $I_j$  and  $P_j$  calculated with the granularity of a processing node  $j$ .

Let us consider the supply chain system as a network  $G(V, A)$  where  $V$  is the set of the processing nodes connected by arcs  $(j, k) \in A$ . Let us define a set  $B$  containing the vehicles  $v \in B$ , travelling on the arcs  $(j, k) \in A$ . We define a state function  $\Lambda(\tau)$  to describe the state of the network  $G$  at the time instant  $\tau$ .

$$\Lambda(\tau) = \left\{ I_j(\tau), j \in V \cup I_v(\tau), v \in B \right\} \quad (3.13)$$

We demonstrate that:

**Theorem 1.** *One of the following set of equations*

- (i)  $M_j(t)$
- (ii)  $I_j(t)$
- (iii)  $P_j^{IN} \cup P_j^{OUT}$

*has enough information to define the state  $\Lambda$  of a logistic network.*

*Proof.* We demonstrate Theorem 1 by showing that the knowledge of (i), (ii), or (iii) is enough to define all the three  $M$ ,  $I$ , and  $P$ . Statement (i) has already been demonstrated by the 3.7, 3.11 and 3.12. Statement (ii) can be proved considering the definition of  $I$ , since:

$$M_j(t) = I_j(t) - I(t - \epsilon) \quad (3.14)$$

Given  $M_j(t)$  the functions  $P_j^{IN}$  and  $P_j^{OUT}$  are calculated similarly to 3.11 and 3.12.

Statement (iii) can be proved considering:

$$Q_j^{IN}(\tau) = \int_0^\tau P_j^{IN}(t)dt \quad (3.15)$$

$$Q_j^{OUT}(\tau) = \int_0^\tau P_j^{OUT}(t)dt \quad (3.16)$$

By sampling the functions  $Q$  with a sampling frequency  $\epsilon$  (e.g. a minute), it is possible to get the movement function.

$$M_j(t) = \frac{dQ_j^{IN}(t)}{dt} - \frac{dQ_j^{OUT}(t)}{dt} \quad (3.17)$$

□

The proof of Theorem 1 shows that the functions  $M$ ,  $I$  and  $P$  are interconnected, and retain enough information to define the state  $\Lambda$  of the network when they are measured with terminal granularity.

### 3.3.2 Implementation and utilisation of the framework

The following chapters of this work start from the application of this information framework to warehouses, production and distribution system. The consistency rules between the metrics of movements, inventory and productivity, are the starting point to collect, store and analyse data leading, and to generate knowledge from this data.

We start with the definition of a data structure able to host movements and inventory data of a specific type of processing node. Then data-driven methodologies are proposed to build prediction models addressing node-specific problems. ER structures are designed to store both planning or actual data. In particular:

- planning data, (i.e., recorded before it happens what they describe) describes how operations are planned to be performed;
- actual data (i.e., recorded when it happens what they describe), describes how activities were performed.

Table 3.1 qualitatively introduces different types of data recorded on-field and used in the following sections to populate the data structure, according to the  $M, I, P$  framework.

Function	Production system	Storage system	Distribution network
M	Production cycles	Putaway/picking records	Truck load/unload records
I	Buffering between working stations (parts)	Inventory level (dm3)	Truck saturation
P	Throughput of a production line (parts/h)	Warehouse productivity (lines/h)	Loading/unloading dock throughput (pallet/h)

Table 3.1: Examples of datasets connected to the functions.

Draft

## Bibliography

- [1] T. Sprock, “Self-similar architectures for smart manufacturing and logistics systems,” *Manufacturing Letters*, vol. 15, pp. 101–103, 2018.
- [2] W. J. Hopp and M. L. Spearman, *Factory Physics*. Waveland Press. Long Grove, IL, 2011.
- [3] M. L. Spearman, “Of physics and factory physics,” *Production and Operations Management*, vol. 23, no. 11, pp. 1875–1885, 2014.
- [4] P. Lake and P. Crowther, *Concise Guide to Databases*. 2013.

# 4

## Data-driven decision making

*A data scientist must deeply know his knowledge domain.*

Chapter 3 introduced the entities and the metrics of a supply chain system. This book deals with these entities from a system analysis perspective (see section 1.1), i.e. considering the design and control alternatives to manage the operations of a distribution network, a storage system, or a production plant. These problems have been mostly addressed by decision science. The following chapters aim at identifying boundaries between decision science, and data science in the system analysis of a supply chain system. It will be remarked, as well, where data science is ready to compete with decision science to address a decision problem.

It is crucial to consider the knowledge domain of supply chain systems to do this job [1, 2]. While decision science has the ability to model each problem precisely, with instance-tailored boundaries and objective functions, we aim at designing a high-level classification of supply chain system problems. We identify patterns among similar decision problems encountered in different supply chain systems whose entities are defined by the ontology in 3.1. Each decision pattern is identified by an id used in the following chapter as reference.

The following section introduces a glossary of data and information to understand the meaning of these words correctly; the classification of the decision patterns; a method to choose the right analytical technique to address a supply chain system problem, given its data, and its decision pattern; a summary of the main contributions of this book.

## 4.1 Data Glossary

Since this book deals with data, it is important to remark some crucial aspects. The traditional engineering modelling approach is based on intuition. The researcher observes a logistic phenomenon, makes assumptions, build a model, and fit the empirical data to the model. The data-driven approach is profoundly different since it collects data and different models fit the data, identifying the model with the best fit.

Our data collection activities involve multiple entities since we use a top-view perspective, looking at the system analysis. For this reason, we may have multiple data sources with redundant information, different synchronism, different granularities, different data protocols. Table 4.1 introduces the glossary explaining all these elements.

Glossary	Description
Data	Measure of a physical variable on a digital support
Data granularity	Level of aggregation of the data collected. (e.g. data collected for a single product has a smaller granularity than data collected for all the products of a lot together)
Data redundancy	Presence of the same piece of information repeated within the same data set.
Data synchronism	Synchronisation of the recording timestamp of different data sets.
Data exchange protocol	Method to exchange or share data between different sources.
Data attributes	Variables of a physical phenomenon observed and recorded simultaneously (e.g. the id of a product and its description).
Information	High level knowledge obtained by processing raw data.

Table 4.1: Glossary of data-related terms.

## 4.2 Decision patterns

Entities are connected with relevant strategic and control issues that must be addressed to properly organise the operations of a warehouse, production plant, or distribution system. The literature defines these issues as problems, using mathematical models. We organise these problems by introducing an original classification, with ten classes. Each class is addressable by the same branch of analytics (descriptive, explorative, predictive, or prescriptive), depending on the type of problem.

**Family problems (P1)** This class of problems aims at defining homogeneous clusters of parts, to simplify the organisation of operations. Issues belonging to this class concern definitions of product families with the same production cycle (i.e. the route), and definitions of classes of stock-keeping units (SKUs) with similar characteristics (e.g. weight and volume). Descriptive and explorative analytics offer tools to assess the features of parts and to produce clusters.

**(Technology) Assignment problems (P2)** This class of problems solves the assignment of parts to resources and vehicles from a high-level strategic perspective. Some examples are the assignment of SKUs to storage locations, the assignment of points of demand to trucks or the identification of the families of product to be processed by a resource (e.g. a production line or an FMS). The best configuration is found by the evaluation of every single alternative (prescriptive methodologies) or the definition of homogeneous clusters by an explorative technique.

Problems involving the definition of adequate technology for production nodes belong to this class and can be addressed using the same rationale. The definition of the storage system technology (storage rack with/without forward-reserve, floor stack, automated storage technology) and the identification of the level of automation in a production plant are examples of these problems. The technological choice in a distribution network (i.e. truck/rail/water/air) is neglected since it is imposed by the existent infrastructure or by political choices falling outside the domain of this research. The choice of the vehicle is solved in an operational environment with synchromodality i.e. when multiple transportation options exist for the same transportation unit (e.g. a pallet on a barge or a truck).

**Flow problems (P3)** This class of problems identifies how processing nodes are connected. They only exist for warehouse and production nodes since the design of the distribution infrastructure belongs to transportation science, and it is outside the domain of this research. The definition of the rationale to travel within storage racks (i.e. return or traversal policy) and the identification of paths for conveyors and forklifts in a production plant are examples of these problems. As for technology assignment problem, flow problems need prescriptive tools to identify efficient handling solutions.

**Mechanical plant and equipment design (P4)** This class of problems addresses engineering issues, such as the design of power plants, thermal plants, mechanical plants, lighting systems, air conditioning systems, and/or the physical designs of workbenches, material slots, and ergonomics. All these activities usually have poor historical data, and it is difficult to structure a series of previous observations of data addressing the same prob-

lem. For this reason, prescriptive models often address these problems by relying on an engineering model describing the dynamics or thermodynamics of the system.

**Power problems (P5)** This class of problems identifies the amount of power required for a specific resource  $j$ , and defines its capacity  $C_j$ . The storage allocation and design of areas for inbound/outbound operations are examples of power problems in a storage system. The design of the frequency of a route and service time windows at a terminal are power problems in a distribution network; the definition of the number of machines of the same type is a version of this problem in a production node. Prescriptive tools allow for solving these problems.

**Placement problems (P6)** A placement problem defines the identification of the proper disposition of entities (e.g. resources or set of parts) on the plant layout of a production system or storage system. Examples include the definition of a plant layout or location of a facility, the assignment and placement of SKUs to warehouse zones, the definition of the location of the facilities of a distribution network (location-allocation problem). Prescriptive and explorative techniques address these problems by clustering parts with similar behaviour (e.g. placing the same resources close to each other).

**Dispatching rules (P7)** This class of problems provides rules for organising operations among the many possibilities and uncertainties that may occur in practice. The definitions of the shipping priority for transportation units and the picking policy (e.g. batching, sorting, pick and pack, single-order picking, cross-dock) for SKUs are examples of dispatching rules. In a production environment, the choice between a pull or a push policy is another dispatching rule. Dispatching rules affect the level of the work in process, size of the lots, and rationale for satisfying the market demand (i.e. pull/push). The definition of these rules requires a prescriptive model.

**Performance assessment (P8)** This class of problem is deliberately descriptive. It aims to describe the system  $G$ , its entities (i.e. parts, resources and vehicles), and its processes (i.e. jobs and routes). Data science offers tools to approach this type of problem, as the results are obtained exclusively using descriptive and explorative tools for evaluating the log data from a storage system, production system, or distribution network.

**Workload predictions (P9)** This class of problems aims at forecasting the values of relevant variables regarding a product or a process in the future. The predictions are influenced by market demand. Hence, predictions

consider the number of parts (e.g. SKUs, containers, or products) which will be processed by a system  $G$  in the near future. The estimate of the number of parts leads to the prediction of other relevant process variables (e.g. the speed of a machine, or number of operators required to perform activities).

**Operations management (P10)** This class of problems prescribes how to act within given circumstances to perform operations. This problem always involves a prescription, e.g. the definition of a sequence of products to process on a machine (job scheduling), or the sequence of locations to visit from a truck in a distribution system or a forklift in a warehouse system. Prescription tools can approach these problems when the input data are consistent and compliant with their hypotheses. In particular situations, predictive and explorative tools may be used to address the problem as well. For example, when processing a single part to assign to a resource in an online version of the problem (e.g. a last-minute order to assign to a production machine), a prediction or clustering model based on robust data can solve the assignment, given the current state of the system  $G$ .

## 4.3 Decision trees

Once the class of analytics (i.e. descriptive, explorative, predictive, or prescriptive) for solving a problem has been defined, it is necessary to identify the technique for obtaining a solution. Different methodological paths exist to solve a problem  $P$ , depending on:

1. the need to set up decision variables  $D$  to solve the problem (e.g. in prescriptive analytics);
2. the availability of measurements on previous realisations of the solution;
3. the completeness and the accuracy of the realisation dataset; and
4. the knowledge on the boundary conditions of  $X$ .

We introduce three original decision trees to identify these paths, and to guide the decision-maker in the definition of the proper technique to address a problem. Decision trees focus on descriptive, predictive, and prescriptive analytics, whereas exploratory analytics appears in some branches of the other trees.

### Descriptive decision tree

This tree aims at the description of a variable  $y$  as a random variable;  $y$  is usually a KPI metric, and there are no decision variables  $D$  to set up. Sometimes, it is impossible to directly measure a variable on-field (e.g. often when  $y$  is a cost). In these cases, it is necessary to link the value of  $y$  with a kinematic model  $y = f(X)$ , based on a set of measurable variables  $X$  that are inputs of a motion equation  $f$ . The variables  $X$  usually describe the movements of a logistical process (e.g. the path travelled by a truck on a road or by a forklift within a plant). Once  $f$  is defined, a numerical simulation (e.g. Monte Carlo simulation) allows for investigating the behaviour of  $y$ , depending on the distribution of  $X$  and on the tuning parameters of the kinematic model. When  $y$  is measurable but the dataset presents different data sources with different levels of accuracy, Bayesian statistics can be used to infer the properties of  $y$  by leveraging its estimate and considering the reliability of each input data source. Otherwise, the inferential statistic is appropriate when the data are from a single source, and there are no measurements  $Y$  of other variables linked to  $y$ . When a dataset  $Y$  containing observations of a set of variables connected to  $y$  is available, explorative analytics (i.e. clustering) is suitable for investigating the correlations between the variables and improving the knowledge of the decision maker regarding the important variables affecting the process. Figure 4.1 introduces the descriptive decision tree, and illustrates the decision steps identified above.

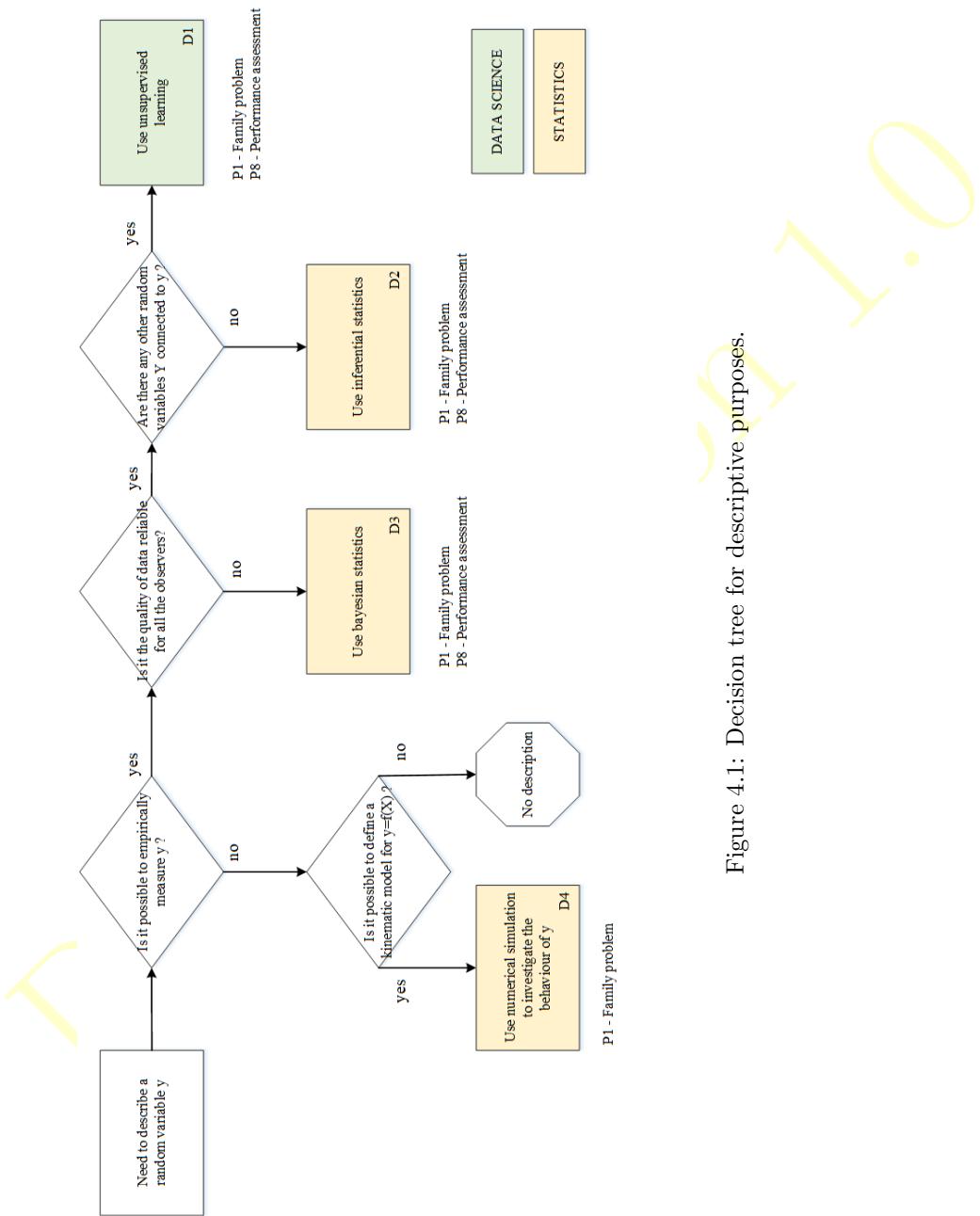


Figure 4.1: Decision tree for descriptive purposes.

### Predictive decision tree

This tree aims at forecasting future realisations of a variable  $y$ , given its description as a random variable. For this reason, there are no variables  $D$  to set up (except for the hyperparameters of some forecasting models). The ability to empirically measure the variable  $y$  is a key branch of the decision tree. If the variable cannot be measured directly (e.g. the inventory position of a truck while travelling), the only other option is to define a kinematic model to estimate its value (e.g. estimating the inventory using loading and unloading records). Without a kinematic model and/or a direct measurement, it is impossible to make predictions. When the variable can be measured but the measures are incomplete (e.g. a small subset of the dataset), explorative clustering techniques can be used to extend the properties measured for the small sample to the entire population  $\tilde{y}$  (e.g. from a subset of parts to an entire product family). In this case, it is necessary to have a validation dataset  $\hat{y}$  associated with the values of  $\tilde{y}$ . If a validation dataset is not available, the kinematic model remains the sole alternative for obtaining an estimate. Otherwise, it is possible to set a prediction model. When there are no other variables  $Y$  associated with  $y$ , time-series forecasting (e.g. decomposition, Fourier analysis, autoregressive integrated moving average (ARIMA) models) can be used to make predictions. When a dataset of variables  $Y$  associated with  $y$  is available, supervised machine learning models can lead to more accurate predictions. Figure 4.2 illustrates the decision tree, along with the decision steps identified above.

Draft

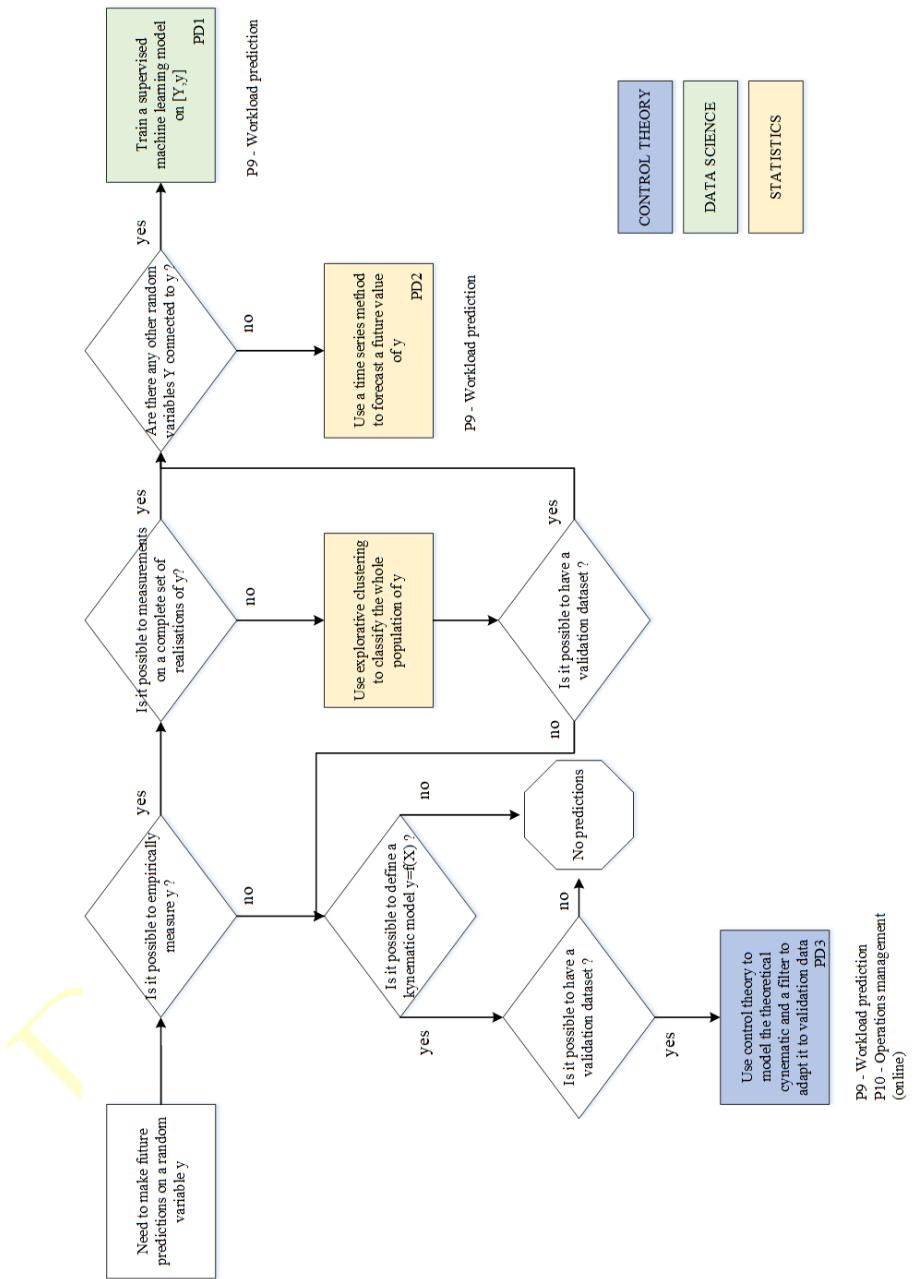


Figure 4.2: Decision tree for predictive purpose.

### Prescriptive decision tree

When addressing prescription problems, it is necessary to identify a set of decision variables  $D$ , such that an objective on  $y$  can be reached (e.g. a cost can be minimised). As for the previous trees, when  $y$  cannot be directly measured, it is necessary to define a kinematic model  $y = f(D)$ . If no validation dataset  $\hat{y}$  is available, there is no scientific way to set  $D$ . Otherwise, control theory can help to adapt the kinematic model to realisations in the real world. The optimal solution is obtained by using mathematical minimisation (e.g. derivatives) on the motion equations of the system. When measurements are available on a subset of realisations of  $y$ , unsupervised clustering is used to extend the properties of the observed dataset with a clustering function  $y = g(D)$ . If the problem is an assignment problem, the clustering boundaries  $g$  may be sufficient to solve the problem (e.g. assignment of parts to processing nodes); otherwise, it is necessary to enumerate all of the alternatives and to select the best one. When a dataset with observations for all of the entities is available and it is possible to define a feasibility region for any value of  $y$ , optimisation should be used. Figure 4.3 presents the decision tree, along with the decision steps identified above.

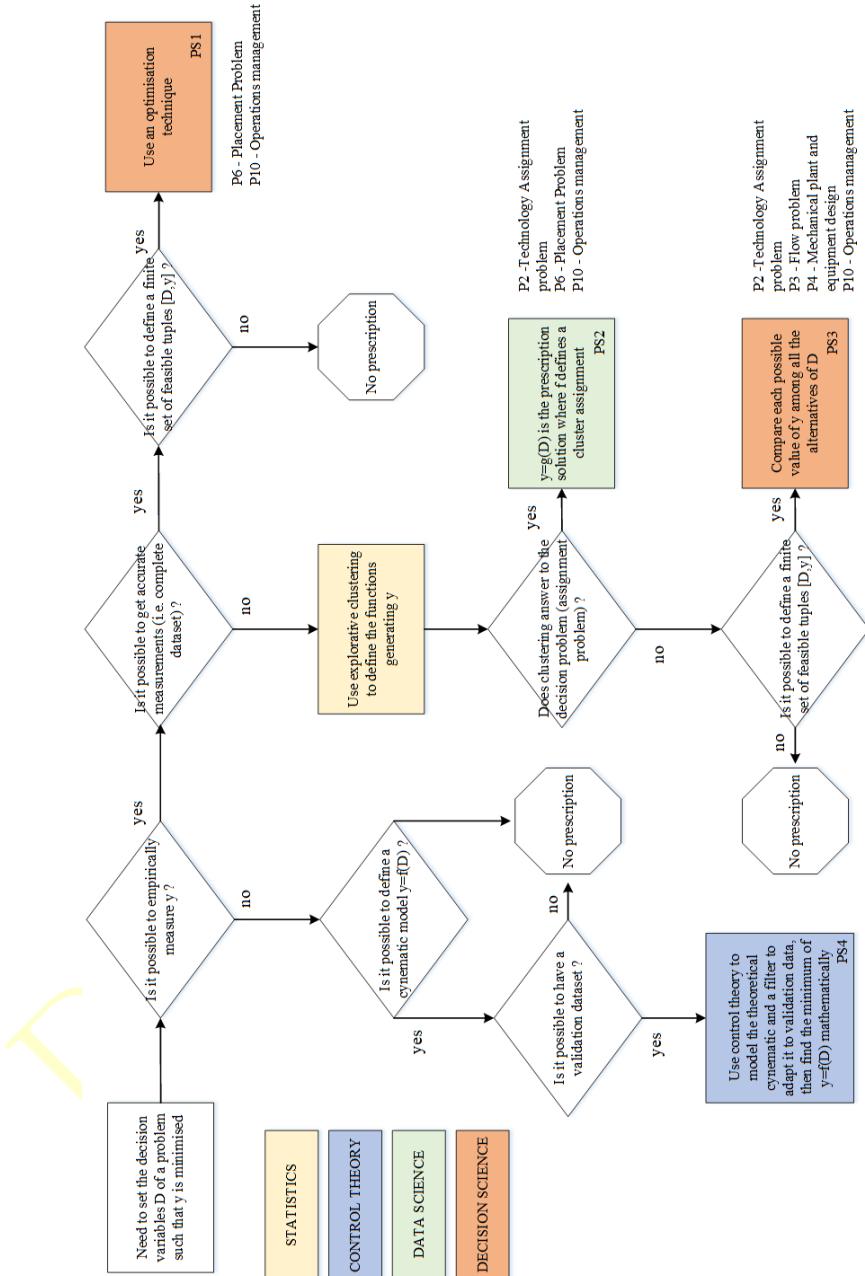


Figure 4.3: Decision tree for prescriptive purpose.

The resulting classification of problems, analytics, and solving techniques is introduced in Table 4.2. Each row of the table refers to a problem within its system domain (i.e. warehouse/production plant/distribution system). It identifies the type of decision (design or control) and the entities involved (e.g. part, vehicle). The right side of the table describes the classes of analytics that address the problem, and the methodologies for obtaining a solution (according to the labels used in the decision trees). For example, the first row of the table addresses the family problem (P1) in warehousing systems. This is a design decision involving the definitions of clusters of SKUs. SKUs are, then, the entities involved in being modelled as parts in the ontology. The problem is addressable by using descriptive or explorative methodologies, labelled as D1, D2, and D3 in the decision trees.

Draft Version

System	Class of the problem	Decision	Task	Entity	Ontology	Descriptive	Exploratory	Predictive	Prescriptive	Decision Science	Methodologies
Warehouse node P1 - Family problem	Design	Volume, weight, size estimation	Stock Keeping Unit	Part	•	•	○	○	○	D1, D2, D3	
Warehouse node P5 - Power problem	Design	Definition of the inventory level	Stock Keeping Unit	Part	○	○	○	●	●	PS4	
Warehouse node P2 - Technology Assignment problem	Design	Choice of the storage technology	Storage technology	Part, Vehicle	○	●	●	●	●	PS2, PS3	
Warehouse node P5 - Power problem	Design	Storage allocation slotting	Stock Keeping Unit	Part	○	●	○	○	○	PS2, PS3	
Warehouse node P2 - Assignment problem	Design	Storage assignment (Random ABC Dedicated)	Stock Keeping Unit	Part, processing node	○	●	●	●	●	PS2, PS3	
Warehouse node P6 - Placement Problem	Design	Layout design (Zones)	Stock Keeping Unit	Part, processing node	○	●	●	●	●	PS1, PS2	
Warehouse node P7 - Dispatching rates	Design	Picking policy design (Single order/Batching)	Stock Keeping Unit	Part	○	○	○	●	●	PS4	
Warehouse node P3 - Flow problem	Design	Route design (Return/Traversal)	Aisles	Edges	○	○	○	●	●	PS3	
Warehouse node P5 - Power problem	Design	Inbound & Outbound area design	Buffers	Processing node	○	○	○	●	●	PS4	
Warehouse node P8 - Performance assessment	Control	Performance assessment	Storage system	System Network, Parts, Resources, Vehicles, Jobs, Routes	●	●	○	○	○	D1, D2, D3, D4	
Warehouse node P9 - Workload prediction	Control	Workload forecast	Stock Keeping Unit	Part	○	○	●	●	●	PD1, PD2, PD3	
Warehouse node P10 - Operations management	Control	Vehicle routing	Vehicle	Vehicle, Part	○	●	●	●	●	PS1, PS2, PS3	
Distribution Network	P6 - Placement Problem	Design	Locality-allocation problem	Points of demand	Processing node	●	●	○	○	○	D1, D2, D3
Distribution Network	P2 - Assignment problem	Design	Network topology design	Point of demand, routes	Part, Vehicle	●	●	○	○	○	PS2, PS3
Distribution Network	P5 - Power problem	Design	Route frequency design	Air/Rail/Road/Water connections	Edges	○	○	●	●	●	PS4
Distribution Network	P5 - Power problem	Design	Service time windows design	Terminal	Processing node	○	○	●	●	●	PS4
Distribution Network	P7 - Dispatching rates	Design	Shipping priority definition	Transportation unit (Carton/pallet/containers)	Part	○	○	●	●	●	PS4
Distribution Network	P8 - Performance assessment	Control	Performance assessment	Distribution network	System Network, Parts, Resources, Vehicles, Jobs, Routes	●	●	○	○	○	D1, D2, D3, D4
Distribution Network	P9 - Workload prediction	Control	Workload forecast	Transportation unit (Carton/pallet/containers)	Part	○	○	●	●	●	PD1, PD2, PD3
Distribution Network	P2 - Technology Assignment problem	Control	Vehicle choice (Synchromodality)	Vehicle	○	●	●	●	●	PS2, PS3	
Distribution Network	P10 - Operations management	Control	Vehicle routing	Vehicle	Vehicle, Part	○	●	●	●	●	PS1, PS2, PS3
Production node P6 - Family problem	Plant Design	Plant Design	Clustering of parts/families	Stock Keeping Unit	Part	●	●	○	○	○	D1, D2, D3
Production node P4 - Mechanical plant and equipment design	Plant Design	Plant Design	Facility location	Production system	○	○	○	○	○	PS4	
Production node P2 - Technology Assignment problem	Plant Design	Technology and asset choice	Auxiliary systems design	Energy, thermal plant and services	○	●	●	●	●	PS2, PS3	
Production node P5 - Power problem	Plant Design	Plant Design	Definition of the number of assets	Machines, workstations and resources	Processing node	○	●	●	●	●	PS4
Production node P6 - Placement Problem	Plant Design	Plant Design	Layout design	Workstations	Processing node	○	○	○	○	○	PS1
Production node P7 - Dispatching rates	Process Design	Process Design	Inventory policy design	Production system, resources	Processing node	○	●	●	●	●	PS4
Production node P3 - Flow problem	Process Design	Handling design	Conveyer/fieldlift	Vehicle	○	○	○	○	○	PS3, PS4	
Production node P4 - Mechanical plant and equipment design	Process Design	Workstation design	Workstations	Processing node	○	○	○	○	○	PS3, PS4	
Production node P8 - Performance assessment	Production Control	Performance assessment	Storage system	System Network, Parts, Resources, Vehicles, Jobs, Routes	●	●	●	●	●	D1, D2, D3, D4	
Production node P9 - Workload prediction	Production Control	Workload forecast	Stock Keeping Unit	Part	○	○	●	●	●	PD1, PD2, PD3	
Production node P10 - Operations management	Production Control	Job scheduling	Machine/resource	Processing node, Part	○	●	●	●	●	PS1, PS2, PS3	

Table 4.2: Definition of the problems for warehouse nodes, production nodes and distribution networks.

## 4.4 Main contributions of this book

Chapter 3, and the previous paragraph of this chapter, introduced a new method to structure data, and precise pattern to identify the analytical approach to solve a problem. The remainder of this book shows how to apply these elements to distribution networks, storage systems, and production plants.

In particular, we focus on the data-driven models presented in part II to show how to learn information from a dataset; while part III, IV, and V shows what information can be learnt from logistics and operational data.

We explore the role of data in logistics and operations research. The research, according to the first three paradigms, involves the design of a model and the optimisation of the modelled system using the models' parameters. Here we use the fourth science paradigm; for this reason, the solely modelling activity involves the definition of consistency rules between data (i.e. the information framework introduced in chapter 3). For this reason, each of the following sections proposes relational, and non-relational data structures to host data from a specific industrial domain (i.e. warehousing, transportation and production). We will show that, if data are stored correctly, it is possible to define consistency rules to get the highest information even when the input data is incomplete.

Another important contribution regards the way we do research. Researchers must do experiments. Doing experiments at a system level is hard since it is impossible to build a lab containing a globally distributed supply chain. For this reason, we create virtual environments using digital twins of the entities of a supply chain. Then, we do experiments on these virtual entities. Data define the entities while scripts of code do the experiments on these entities. Our research becomes reproducible since the scripts can be run many times with different input data supporting the generalisation of our research. These scripts implement the analytics, that is the technology we want to explore and test in this research. There are two groups of scripts implemented within the virtual laboratory: general-purpose, and problem-oriented scripts.

General-purpose scripts implement general-purpose methods, i.e. pieces of code based on data-driven approaches that are not necessarily linked to the field of logistics and operations. All the methods illustrated in part II belong to this class of scripts.

Problem-oriented scripts find the solution of a problem by using a decision-pattern illustrated in 4.2, and may combine many general-purpose scripts. These methods are presented in part III, IV, V.

Another type of script manages the flow of data from the industrial data sources to the other type of scripts. Figure 4.4 illustrates the flow of data.

To support the research community in the field of supply chain systems with robust methods, and data structure, problem-oriented and general-

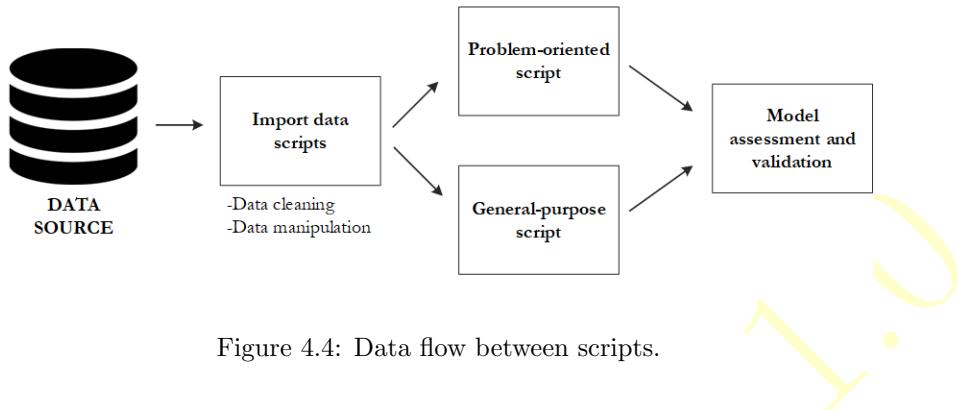


Figure 4.4: Data flow between scripts.

purpose scripts are written using python [3], the most used programming language nowadays, and distributed with an open-source licence, according with the open-science mission indicated by the European Union [4].<sup>1</sup>

From a research perspective, we can state several research questions addressed by the contents of this book.

*RQ1: Which data are needed to solve a problem in the field of logistics and operations?*

*RQ2: how to collect, organise, preprocess and manipulate this data?*

*RQ3: which method should be used to address an issue in the field of logistics and operations?*

*RQ4: when the data-driven approach is recommendable to address a problem in the field of logistics and operations?*

The remainder of this book is organised as follows. Part II introduces and clarifies all the math, statistics and the basic models which will be applied in this book. Part III, IV and V are dedicated to a logistic system, i.e. storage nodes, distribution networks, and production plants. Each part has a similar structure illustrating:

- The design of a diagnostic model to assess the entities and the metrics of a logistics system;
- The design of a relational data structure to store planned and actual logistics/operations data;
- The design of a non-relational data structure to store planned and actual logistics/operations data;

<sup>1</sup>The package logproj contains general-purpose, and problem-oriented scripts [here](#).

- Model-driven methods to address control issues of the logistic system;
- Data-driven methods to address control issues of the logistic system;
- Model-driven methods to address design issues of the logistic system;
- Data-driven methods to address design issues of the logistic system.

## Bibliography

- [1] H. Stadtler and C. Kilger, *Supply chain management and advanced planning (Fourth edition): Concepts, models, software, and case studies*. 2008.
- [2] J. Hull, M. Meixell, P. Luoma, Deloitte, J. Macaulay, L. Buckalew, G. Chung, R. Aboutalebi, G. Vijayan, N. H. Kamarulzaman, A. Mukherjee, S. K. N. Vaiappuri, N. E. W. Trends, C. In, A. Industry, L. Operations, B. Schmidt, C. M. Wallenburg, S. Rutkowsky, L. Einmahl, I. Petersen, F. Klötzke, K. Kasemsap, D. Knoll, M. Pr??glmeier, G. Reinhart, R. Handfield, F. Straube, H.-C. Pfohl, A. Wieland, R. Caballero, T. Go, D. Data, A.-W. Scheer, A. Maurer, S. Wieland, C. M. Wallenburg, M. Springinklee, F. Rousseau, F. Montaville, F. Videlaine, C. Logistics, G. U. Martin Belvisi, Riccardo Pianeti, R. O. Large, N. Kramer, R. K. Hartmann, U. Pradesh, A. Cox, H. Gleissner, J. C. Femerling, G. Neubert, P. Bartoli, Göran, Svensson, F. P. Buffa, K. T. Yeo, J. H. Ning, L. Kroll, P. Blau, M. Wabner, U. Friess, J. Eulitz, M. Klärner, R. Jothi Basu, N. Subramanian, N. Cheikhrouhou, K. Lee, H. Cho, M. Jung, M. Florian, J. Kemper, W. Sihn, B. Hellingrath, E. Commission, E. U. F. Rance, G. Ermany, I. Taly, J. Apan, and VDMA, *The future of German mechanical engineering Operating successfully in a dynamic environment*, vol. 35. 2016.
- [3] K. D. Lee and S. Hubbard, *Undergraduate Topics in Computer Science Data Structures and Algorithms with Python*. 2015.
- [4] E. Parliament, “DIRECTIVE (EU) 2019/770 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL,” *Official Journal of the European Union*, vol. 2019, no. March, pp. 1–27, 2019.

# Draft Version 1.0

## Part II

### LET'S MATH



# 5

## Logical Modelling

*All models are wrong, but some are useful.*

---

George E. P. Box

This book part is about math. Nevertheless, before deepening into math, it is necessary to mention the Logic, another important branch of science deeply connected with math. The Logic was the main character of the philosophical debate in ancient Greece. Aristotle was the first to set the rules of the Logic as we study nowadays [1].

Today we study the Logic by using the truth table and the logical operator (i.e. *AND*, *OR*, *NOT*). These simple operations on zero and ones are performed on any CPU allowing from simple calculations to the moon landing, to machine learning and artificial intelligence. Every input or output of a computer is processed by using the rules of the logic.

Logical modelling is crucial for all the STEM disciplines (science, technology, engineering and mathematics) to identify a deterministic connection between the input and the output of a phenomenon. Logic permits to STEM researchers to build models based on their intuitions. These models can be validated or not by using math and statistics. Nevertheless, it is necessary to remember that all models are wrong. Models approximate reality, but they are not reality. For this reason, they are wrong.

Models help to understand how reality works. If you can understand reality, you can control, and change it. This fact is actual in the field of logistics and operations, whose environment involves thousands of resources, assets, goods and tasks. In this chapter, we use the logic to model and connect these entities aiming at the understanding of complex operational

environments.

## 5.1 Business Process Model and Notation

Literature introduces many notations for modelling a business process. In this work, we introduce the Business Process Model and Notation (BPMN) that results adequate, when applied to the modelling of logistics and operational processes [2]. The BPMN uses a number of predefined symbols to model entities, flows, resources and activities of any business process. The full notation requires hundred of pages of details, but the main elements (see Figure 5.1) are three [3]:

- Event: it is represented by a circle and is something that “happens” during a business process. These Events affect the flow of the process and usually have a cause (trigger) or an impact (result).
- Activity: it is represented by a rounded-corner and is a generic term for work that company performs.
- Gateway: it is represented by a diamond and is used to control the divergence and convergence of flows as a logic gate.

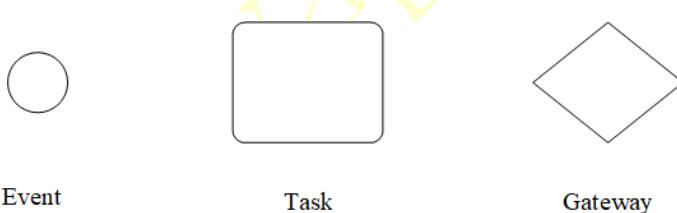


Figure 5.1: Main elements of a BPMN.

The following sections of this work identify the resources, asset, tasks and goods to associate with each of these elements. Figure 5.2 illustrates an example of a BPMN to describe the operations of a port terminal.

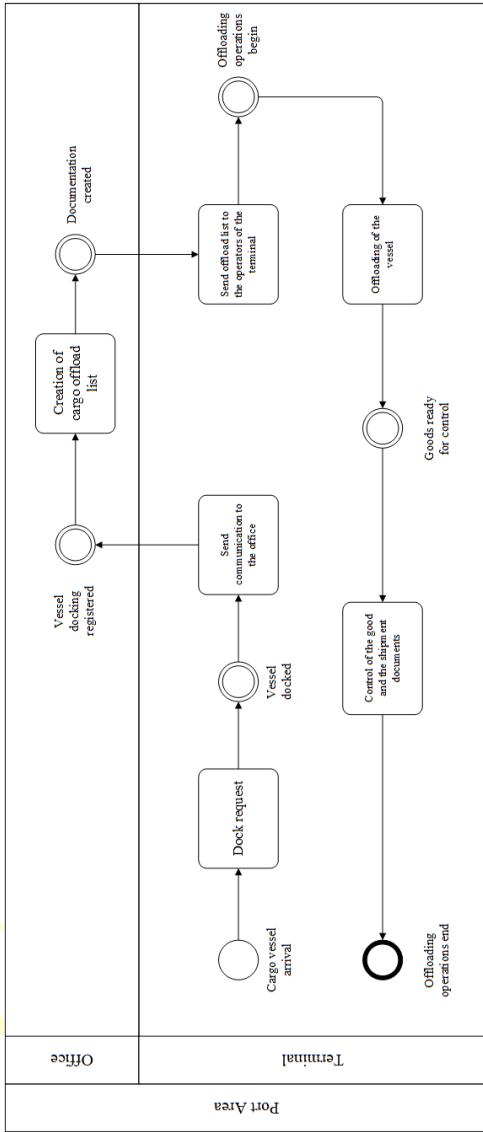


Figure 5.2: an example of a BPMN applied to the logistics of a port terminal.

An additional logistic feature of the BPMN is represented by the possibility of georeferencing tasks and events. Using this method, the physical location of each activity is identified. In addition, it is possible to annotate the responsible of the activity. The BMN allows tracking the physical and information flows of logistics operations qualitatively, by introducing logical rules on the events and gateways leading to the tasks.

## Bibliography

- [1] Aristotle, H. P. Cooke, and H. Tredennick, *the Organon*. 1938.
- [2] OMG, “Business Process Model and Notation (BPMN),” tech. rep., 1998.
- [3] S. A. White, “Introduction to BPMN,” *BPTrends*, 2004.

# 6

## Elements of probability and statistics

*Let math do its work.*

This work comes with the idea that probability and statistics can solve the majority of the problem from logistics and operations. Unfortunately, many logistics and operations managers forgot about the superpowers of statistics. Software developers do the same while deploying warehouse management systems (WMS), transportation management system (TMS) and manufacturing execution system (MES). For this reason, this chapter reviews the most essential elements of statistics and probability upon which are the base of the methods implemented in the following chapters.

### 6.1 Probability Theory

Probability theory aims at defining the behaviour of a variable (let us call it random variable) whose value is not deterministic. A random variable describes the realisations of an event whose outcomes are not static. Any phenomenon measured on-field is describable by a random variable. Flipping a coin is an event having two outcomes (heads or tails); a random variable can be used to describe the outcome (e.g. the expected number of heads over many flips). Operations and logistics management are based on many quantities measured on-field (e.g., time and motion, efficiency, productivity, as already introduced in Chapter 3). Random variables can be used to describe all these variables.

Switching to math, we can define a series of  $n$  observations recording the values of  $n$  different realisations of the event described by the random

variable  $X$ . In practice, we use the  $n$  empirical observations to infer the properties of the random variable  $X$ . A random variable  $X$  is fully defined by:

- a probability density function (PDF)  $f_X(x)$ , or
- a cumulative distribution function (CDF)  $F_X(x)$  defined as follows.

$$f_X(x) = \text{prob}\{X = x\} \quad (6.1)$$

$$F_X(x) = \text{prob}\{X \leq x\} \quad (6.2)$$

The probability theory is based on the knowledge of the CDF or PDF of a random variable  $X$ . As previously stated, one out of the two functions is enough to fully define  $X$  since  $f$  and  $F$  are linked by the following equation.

$$F_X(x) = \int_{-\infty}^{+\infty} f_X(t) dt \quad (6.3)$$

Probability theory defines many “famous” probability functions where  $f$  and  $F$  are defined on a continuous domain by using closed-form equations. The Gaussian, exponential, uniform, beta, Weibull distributions are examples of continuous probability distributions. Discrete distributions as the binomial and Poisson are used to define a discrete realisation of an event (e.g. heads or tails, ‘0’ or ‘1’). In practice, a researcher tries to find the best fit between a series of  $n$  realisations collected on-field and a “famous” probability distribution with known  $f$  and  $F$ . When he/she finds an adequate fit, the probability distribution models the behaviour of the random variable and can be used to infer relevant information of the observed event.

In practice, we interpret the as a frequency analysis (i.e. the histogram) of the random variable  $X$ . On the other side, the CDF measures the relative importance of every single observed value cumulated to all the previous. Figure 6.1 shows the empirical and the best-fit PDF and CDF of a sample with  $n = 200$  observations.<sup>1</sup>

### 6.1.1 Statistical Moments

As stated before, it is of our interest to infer properties on the random variable to understand the realisations of the related event. Statistical moments are properties that aim at parametrising the shape of a PDF. Let us define the moment of order  $m$  as:

$$E[(x)^m] = \int_{-\infty}^{+\infty} f(x^m) dx \quad (6.4)$$

---

<sup>1</sup>The source code of Figure 6.1 is available [here](#).

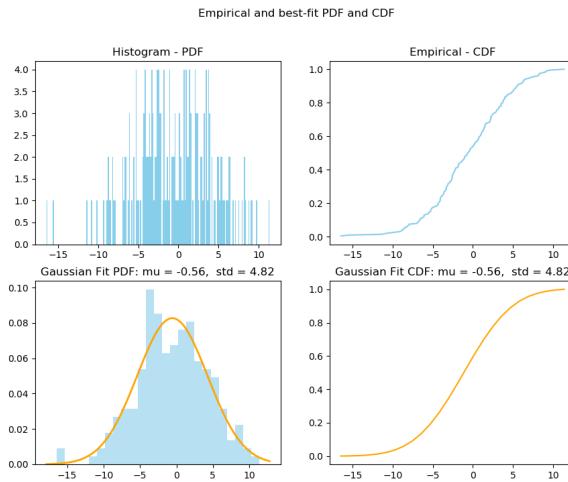


Figure 6.1: Empirical and best-fit probability distribution

and the central moment of order  $m$ , as:

$$M_m = E[(x - \mu)^m] \quad (6.5)$$

The moment of the first order is called *mean* (or expectation) of the probability distribution, generally indicated using the Greek letter  $\mu = E[(x)]$ .

The central moment of the second order is called variance, and it quantifies how much the observations are far from the mean value  $\mu$  of the distribution. The variance is represented by  $\sigma^2$ . We can express the value of  $\sigma^2$  by using equation 6.4.

$$\begin{aligned} \sigma^2 &= E[(x - \mu_x)^2] \\ &= \int_{-\infty}^{+\infty} (t - \mu)^2 f_x dt = E[x^2 + \mu^2 - 2x\mu] \\ &= E(x^2) + \mu^2 - 2\mu^2 \\ &= E(X^2) - \mu^2 \end{aligned} \quad (6.6)$$

Usually, we take care of  $\sigma = \sqrt{\sigma^2}$  since it has the same unit of measure of  $\mu$ .

Other important central moments (order 3 and 4) are used to describe the shape of a PDF:

- $\frac{M_3}{\sigma^3}$  is called *skewness*;

- $\frac{M_4}{\sigma^4}$  is called *kurtosis* or *flatness*.

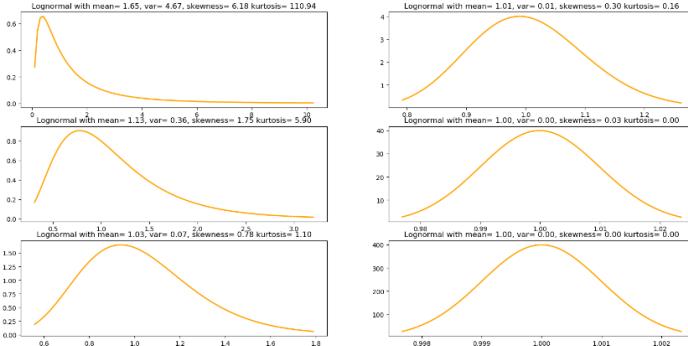


Figure 6.2: Skewness and kurtosis of a lognormal distribution.

Figure 6.2 presents an example of skewness and kurtosis of different distributions (here we use the lognormal). Skewness describes how the mode  $M$  of the distribution is far from the mean  $\mu$  (positive skewness when  $M < \mu$  and negative skewness when  $M > \mu$ ). Kurtosis defines the tailedness of the distribution, higher the kurtosis, higher the relevance of the tails of the distribution.<sup>2</sup>

### 6.1.2 Covariance and Correlation

Often, it is necessary to compare the behaviour of two random variables to understand if their related events are somehow correlated. The covariance function  $cov(X, Y)$  measures how much two random variables vary together.

$$cov(X, Y) = E[(X - E[X])(Y - E[Y])] = E[XY] - E[X]E[Y] \quad (6.7)$$

The correlation between two random variables is a scalar number defining a measure of their statistical association. The correlation between two random variables is measured normalising the covariance to  $\rho_{X,Y}$ .

$$\rho_{X,Y} = \frac{cov(X, Y)}{\sigma_X \sigma_Y} \quad (6.8)$$

---

<sup>2</sup>The source code of Figure 6.2 is available [here](#).

Measuring the correlation between variables is extremely important to evaluate their information content. If two variables are completely correlated (or uncorrelated), their information content is entirely defined by a single of them. Scatterplots are used to visualise correlations. Figure 6.3 presents an example from the famous *iris dataset*. This dataset is largely used in machine learning examples, and it contains 50 samples for each of the three species of the iris flower (*iris setosa*, *iris virginica* and *iris versicolor*). For each sample, the dataset maps the sepal and petal length and width. A high positive correlation can be easily identified between the variables *petal\_len* and *petal\_wid*.<sup>3</sup>

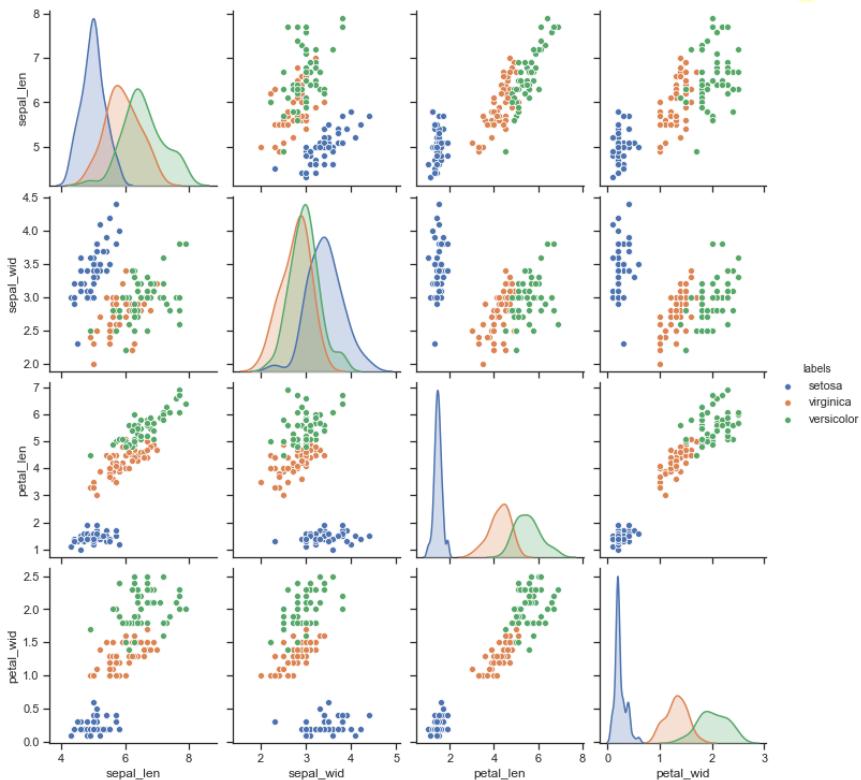


Figure 6.3: Scatterplot of the iris sample dataset.

Sometimes, it may be interesting to evaluate how much a single random variable varies with itself. This is the case of a time series that may have some seasonal components. The autocovariance is introduced to meet this

<sup>3</sup>The source code of Figure 6.3 is available [here](#).

goal. It measures how much a random variable varies with itself after some lag  $k$  (e.g. the sampling period of a time series).

$$\gamma_k = \text{cov}(X_t, X_{t-k}) = E[(X_t - \mu_t)(X_{t-k} - \mu_{t-k})] - \mu_t\mu_{t-k} \quad (6.9)$$

Similarly to the variance, it is possible to define a global autocorrelation function (ACF)  $\rho_k$  measuring the correlation of a variable  $X$  with itself after a time lag  $k$ . This function expresses the linear dependence between the random variable observed at time  $t$  and itself observed at time  $t - k$ .

$$\rho_k = \text{corr}(X_t, X_{t-k}) = \frac{E[(X_t - \mu_t)(X_{t-k} - \mu_{t-k})]}{\sqrt{\text{Var}(X_t)\text{Var}(X_{t-k})}} \quad (6.10)$$

A partial autocorrelation function (PACF)  $\phi_{kk}$  is introduced to measure the linear dependence between the random variable observed at time  $t$  and itself observed at time  $t - k$  without taking into account the intermediate correlations (i.e.  $\phi_{kk}$  does not consider the dependence between  $X_t$  and  $X_{t-1}$ ;  $X_t$  and  $X_{t-2}$ ; ...;  $X_t$  and  $X_{t-k+1}$ ).

$$\phi_{kk} = \text{Corr}(X_t, X_{t-k}|X_{t-1}, X_{t-2}, \dots, X_{t-k+1}) \quad (6.11)$$

Figure 6.4 illustrates a seasonal time series with its ACF and PACF. ACF and PACF of the series evidence that autocorrelation of the realisations exists after about five time lags.<sup>4</sup> Additional details on the use of this information for time series analysis are introduced in Section 6.5.

### 6.1.3 Distance between random variables

In many applications, it is interesting to have a measure of the distance between two random variables as, for example, the distance between two points on a line, chosen with a law of probability.

The distance function is a random variable  $Z$  estimated as  $Z = |x - y|$ . We need to estimate its PDF or CDF to get knowledge about  $Z$ . By the definition of  $F$  we have:

$$F_z(z) = \text{Prob}\{Z \leq z\} = \text{Prob}\{|x - y| \leq z\} \quad (6.12)$$

It is necessary to integrate the density function  $f_Z$  in the domains  $D_X$ , and  $D_Y$ , to calculate  $F_Z$ . In order to define the domains  $D_X$  and  $D_Y$ , it is necessary to consider the function  $Z = |X - Y|$  on the plan x,y (see Figure 6.5).

The value of  $F_Z$  can be determined from the PDF  $f_{XY}$ .

$$F_z(z) = \int_{D_X} \int_{D_Y} f_{xy}(x, y) dx dy \quad (6.13)$$

---

<sup>4</sup>The source code of Figure 6.4 is available [here](#).

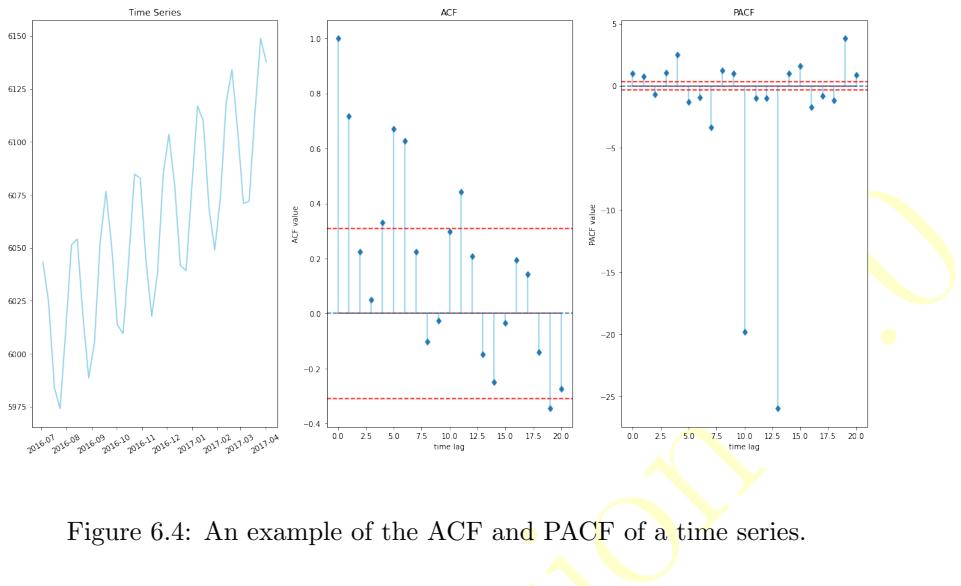
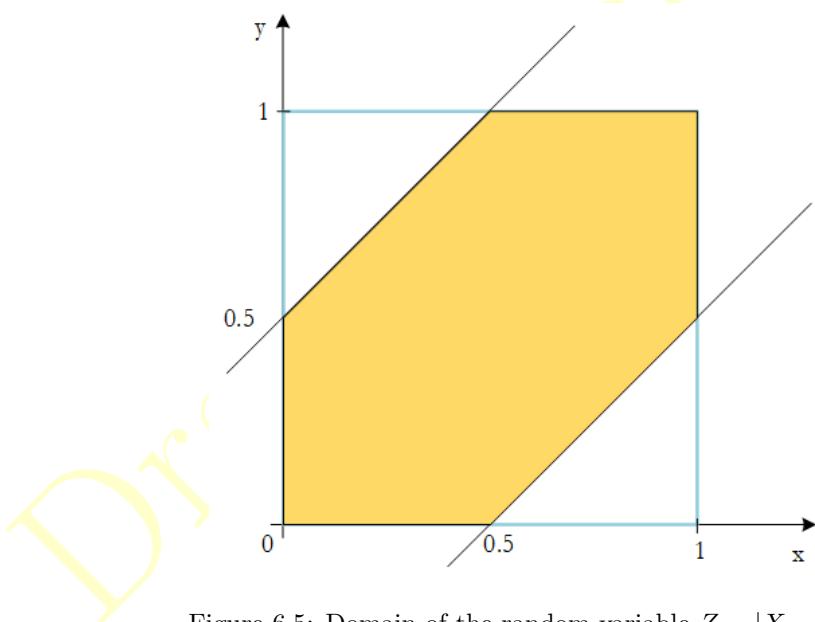


Figure 6.4: An example of the ACF and PACF of a time series.

Figure 6.5: Domain of the random variable  $Z = |X - Y|$ 

Let assume  $X$  and  $Y$  being independent<sup>5</sup> and uniformly distributed on

<sup>5</sup>With the independence hypothesis,  $f_{XY} = f_X \bullet f_Y$

$[0, p]$ .

$$\begin{aligned} X & \sim U[0, p]; f_x = \frac{1}{p}; F_x = \frac{x}{p} \\ Y & \sim U[0, p]; f_y = \frac{1}{p}; F_y = \frac{y}{p} \end{aligned} \quad (6.14)$$

The value of  $F_Z$  is consequently defined as:

$$F_Z(z) = \int_{D_X} \int_{D_Y} f_x(x) f_y(y) dx dy \quad (6.15)$$

The domain of the function is one out of the three regions of plan defined by the corresponding equalities  $Y = X + Z$ ,  $Y = X - Z$ . In Figure 6.5,  $Z = 0.5$ . The region is the one between the two lines. It is, then possible to obtain  $F_Z$ .

$$\begin{aligned} F_Z(z) &= \int_{x=0}^{x=p-z} \int_{y=0}^{y=z+x} \frac{1}{p} \times \frac{1}{p} dy dx + \int_{x=p-z}^{x=z} \int_{y=0}^{y=p} \frac{1}{p} \times \frac{1}{p} dy dx + \\ &\quad + \int_{x=z}^{x=p} \int_{y=x-z}^{y=p} \frac{1}{p} \times \frac{1}{p} dy dx = \\ &= \frac{1}{p^2} \left\{ \int_{x=0}^{x=p-y} (x+z) dx + \int_{x=p-z}^{x=z} pdx + \int_{x=z}^{x=p} (p-x+z) dx \right\} \\ &= \frac{z(2p-z)}{p^2} \end{aligned} \quad (6.16)$$

The PDF  $f_Z$  is defined from equation 6.3 as follows.

$$f_Z(z) = \frac{dF(z)}{dz} = \frac{2(p-z)}{p^2} \quad (6.17)$$

We can prove that  $f_z$  is a PDF since by the definition of  $X, Y$  its domain is  $[0, p]$  and its integral equals 1.

$$f_{Z(z)} = \int_0^p \frac{2(p-z)}{p^2} dz = \left[ \frac{2z^2}{2p^2} \right]_0^p = 1 \quad (6.18)$$

At this stage, all the properties of  $Z$  are defined by  $f$  and  $F$ . For example, it is possible to calculate the mean value corresponding to the

average distance between  $X$  and  $Y$ .

$$\begin{aligned}
 E[Z] &= \int_0^p \frac{2(p-z)}{p^2} zdz = \frac{1}{p^2} \int_0^p (2pz - 2z^2) dz = \\
 &= \frac{1}{p^2} \left[ \frac{2pz^2}{2} - \frac{2z^3}{3} \right]_0^p = \frac{1}{p^2} \left[ p^3 - \frac{2p^3}{3} \right] = \\
 &= \frac{p}{3}
 \end{aligned} \tag{6.19}$$

The procedure above can be applied to any probability distribution of  $X$  and  $Y$  under the independence hypothesis.

## 6.2 Statistics

The statistic is an application of the probability theory to infer the properties of a population of elements working on a small subset of it (also known as "sample"). The statistic was born to solve the trade-off between the time necessary to collect data on-field and the accuracy of the information obtained by these data. In fact, it is always impossible to collect all the information available since a population may count thousand or millions of different elements. Statistics provides models to get robust results even when we have few observations of a physical phenomenon.

### 6.2.1 Estimators

Statistics usually follows a precise workflow:

1. Collect data;
2. Sample data;
3. Infer properties from samples to the whole population.

The last step is the one we are interested in the most: we need to estimate the parameters of the probability distribution of the population. Estimators are used to calculating the value of a parameter of a population (e.g., the mean or the variance) based on the observed values given by the sample. Estimators are classified as biased or unbiased. We call "unbiased" an estimator  $\hat{\theta}$  of a parameter  $\theta$  when  $E[\hat{\theta}] = \theta$ .

The most common estimators are needed for the estimation of  $\mu$  and  $\sigma$  of the population. The sample mean  $\bar{X}$  is an unbiased estimator of  $\mu$  (see equation 6.20). While the sample variance  $S^2$  is an unbiased estimator for  $\sigma^2$  (see equation 6.21).

$$\bar{X} = \frac{X_1 + \dots + X_N}{N} \tag{6.20}$$

$$S^2 = \frac{1}{N-1} \sum_i^N (X_i - \bar{X})^2 \quad (6.21)$$

Estimators are evaluated according to their accuracy (i.e. their closeness to the random variable they estimate). In general, we can find two sources of inaccuracy of an estimator: the bias and the variance. Let assume having a model  $f$  producing an estimator  $\hat{\theta}$  for the random variable  $\theta$  with an error  $\epsilon$ .

$$\theta \simeq \hat{\theta} = f(X) + \epsilon \quad (6.22)$$

A vector  $x_0$  defines a set of realisations of  $X$  and we want to define the error of the estimator  $\hat{\theta}$ .

$$\begin{aligned} \epsilon(x_0) &= E \left[ (\hat{\theta} - \hat{f}(X))^2 | X = x_0 \right] = \\ &= E \left[ (\hat{\theta} - E[\hat{f}(x_0)] + E[\hat{f}(x_0)] - \hat{f}(x_0))^2 \right] = \\ &= E[(\hat{\theta} - E[\hat{f}(x_0)])^2 + (E[\hat{f}(x_0)] - \hat{f}(x_0))^2 + \\ &\quad 2(\hat{\theta} - E[\hat{f}(x_0)]) \times (E[\hat{f}(x_0)] - \hat{f}(x_0))] = \\ &= E \left[ (\hat{\theta} - E[\hat{f}(x_0)])^2 \right] + E \left[ (E[\hat{f}(x_0)] - \hat{f}(x_0))^2 \right] + \\ &\quad + E[2(\hat{\theta} - E[\hat{f}(x_0)])(E[\hat{f}(x_0)] - \hat{f}(x_0))] = \\ &\quad Var(\hat{f}(x_0)) + (E[\hat{f}(x_0)] - \hat{f}(x_0))^2 \\ &= Var(\hat{f}(x_0)) + Bias^2(\hat{f}(x_0)) \end{aligned} \quad (6.23)$$

The error of an estimator can be defined according to 6.23 using:

- $Bias^2(\hat{f}(x_0))$  is the squared bias in the estimation of the mean. It describes how much the mean of the estimator is far from the true mean.
- $Var(\hat{f}(x_0))$  is the deviation of the estimator around its mean.

The role of the variance and bias can be easily interpreted graphically in Figure 6.6.

Complex prediction models developed using data-driven methods, have to take into account the variance and the bias of the predictions they produce. In particular, there is a bias-variance trade-off. Typically, the more

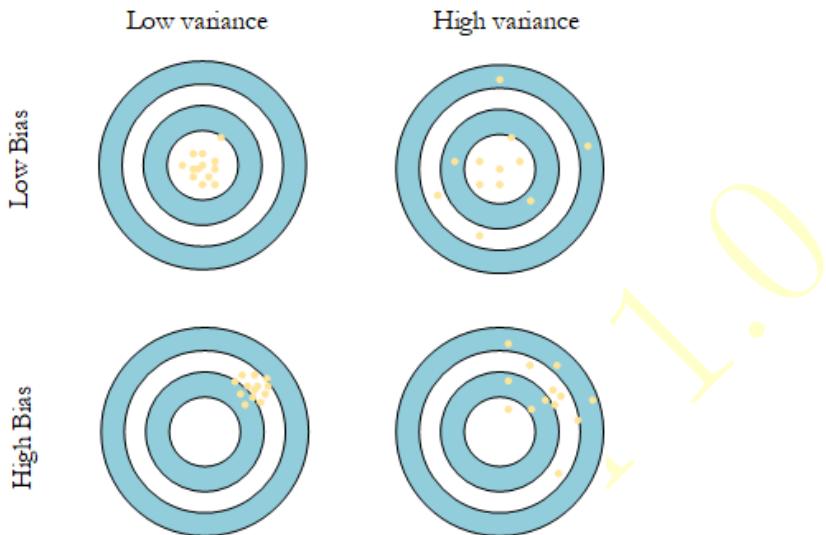


Figure 6.6: The bias and variance of an estimator

complex the model, the lowest the variance and the highest the bias produced by predictions of the models. On the opposite, a simple model leads to low bias but high variance. Practically speaking, it is always necessary to take into account the bias and the variance of the response of a model to check if it fits with its purpose. Developing a complex model to solve a simple problem is just a way to add additional bias to the responses of the model. In general, we keep a model as simpler as possible (according to the Ockham's razor principle).

We introduce the Gauss-Markov theorem to show an important property of the linear regression estimator, one of the simplest prediction models (see chapter 9). Let  $\hat{\theta} = c^T y$  be an unbiased estimator of:

$$\alpha^T \beta = \alpha^T (X^T X)^{-1} X^T y \quad (6.24)$$

Then:

$$E [c^T y] = \alpha^T \beta \quad (6.25)$$

$$Var(\alpha^T \hat{\beta}) \leq Var(c^T y) \quad (6.26)$$

The 6.24 is the expression of a linear regression of  $\theta = \alpha^T \beta$ . In other words, a linear regression provides the lowest variance estimator possible. The lowest variance does not imply a lower error in the prediction since

we have no information about the other error component (i.e. the bias). Nevertheless, we should prefer linear regression, that is a very simple model, when we are sure there is no bias, i.e.  $E\left[E\left[\hat{\theta}\right] - \theta\right]^2 = 0$ . In other words, when the world behaves linearly, use a linear model.

### 6.2.2 Maximum Likelihood Inference

In many practical cases, we need to get a good estimate of a parameter  $\theta$  of a PDF, given a sample of the population. Maximum likelihood estimation (MLE) is the tool to do that. We define a function  $g_\theta(z)$ ; where  $g$  is the PDF (e.g., normal distribution) of  $z_i$  and  $\theta$  are the unknown parameters to estimate (e.g. the mean and the variance  $\mu, \sigma^2$ ). We need to find values of  $\theta$  such that they properly represent the statistical sample. This equals to imply the maximisation of a likelihood function  $L$ .

$$L(\theta, Z) = \prod_{i=1}^N g_\theta(z_i) \quad (6.27)$$

The maximisation is done by considering the logarithm of  $L$ . Maximising  $L$  will maximise  $\log(L)$  too, due to the monotony of the logarithm. To get a maximum likelihood estimation, we need to maximise:

$$l(\theta, Z) = \sum_{i=1}^N l(\theta, z_i) = \sum_{i=1}^N \log(g_\theta(z_i)) \quad (6.28)$$

This is done by looking for  $\theta$  maximising the function:

$$\dot{l}(\theta, Z) = \sum_{i=1}^N \dot{l}(\theta, z_i) = \sum_{i=1}^N \frac{\partial l(\theta, z_i)}{\partial \theta} = 0 \quad (6.29)$$

In practical cases, it may be difficult to express the formula of the PDF  $g$  and to calculate its derivative to get an MLE. Computerised algorithms have been implemented to solve this problem by approximation where it is not possible to solve it analytically. Bootstrap and Montecarlo simulation are common examples.

### 6.2.3 Kernel Density Estimation

The estimation of a PDF, can be obtained using a graphic methodology, instead of mathematically define its parameters. Having a set of empirical observations, it is always possible to use a histogram to represent its frequency analysis. The width of each bin of the histogram defines the shape

of the curve. Figure 6.7 shows different histograms of the same empirical sample by using different widths of the histogram bins.<sup>6</sup>

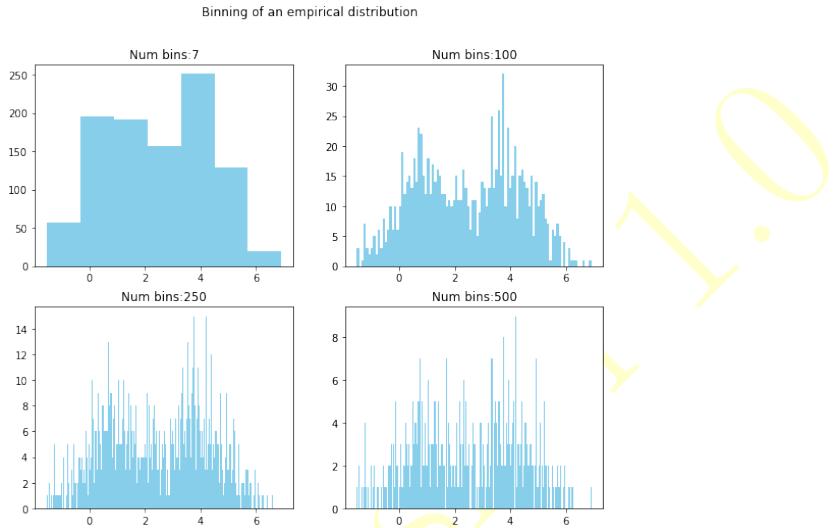


Figure 6.7: Definition of histograms with different bin size

Kernel Density Estimation (KDE) is a procedure to estimate the PDF of a random variable based on its observations. The idea is to define the shape of the PDF based on the empirical values smoothed around a local region  $b$  called bandwidth. This is similar to the choice of the number of bins to define a histogram. KDE can be expressed as:

$$\hat{f} = \frac{1}{n} \sum_i^n K\left(\frac{x - x(i)}{b}\right) \quad (6.30)$$

Where  $K$  is a kernel function with a peak on 0 (it is common to use a gaussian function). Figure 6.8 shows the effect of different KDEs with several values of  $b$ .<sup>7</sup>

#### 6.2.4 Bootstrapping method

Bootstrapping is an algorithm used to estimate the value of a parameter  $\alpha$  (e.g. the mean or variance) from a population where its PDF is unknown or too difficult to estimate analytically. Let  $X$  be a set of observations with

<sup>6</sup>The source code of Figure 6.7 is available [here](#).

<sup>7</sup>The source code of Figure 6.8 is available [here](#).

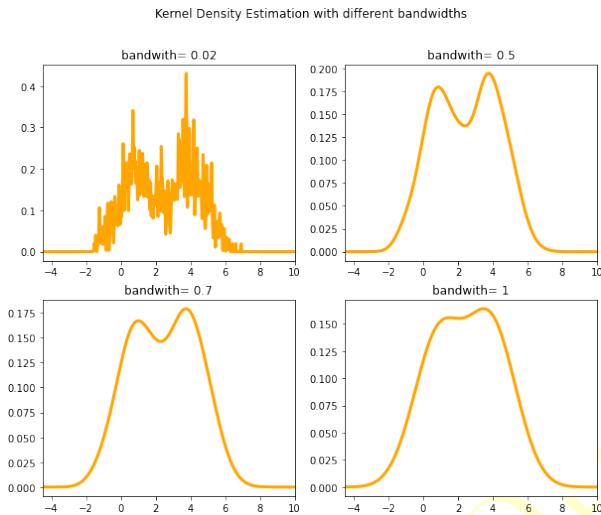


Figure 6.8: KDE with different bandwidths to estimate the PDF of an empirical sample.

cardinality  $n$ . Algorithm 1 illustrates the Bootstrapping method.

---

#### Algorithm 1: Bootstrapping algorithm

---

Set the number of iterations  $B$   
**for**  $i = 1 : B$  **do**  
  | Set  $\beta = n$  points randomly picked from  $X$ .  
  | Use  $\beta$  to estimate the parameter  $\alpha$ .  
**end**

Define the confidence interval of  $\alpha$  using the statistic of the  $B$  iterations;  $\bar{\alpha} = \frac{1}{B} \sum_{i=1}^B \alpha_i$

---

### 6.2.5 Montecarlo method

Montecarlo simulation is a valid alternative to measure the outcome of a process where many random variables with given PDF are involved, but their joint distribution is hard to compute.

Let consider the  $M$  random variables  $X_i$   $i = 1, \dots, M$  whose distribution are given and  $\alpha = f(X_i)$ . We need to infer properties on the distribution of  $\alpha$ . Algorithm 2 shows the Montecarlo simulation to estimate the distri-

bution of  $\alpha$ .

---

**Algorithm 2:** Montecarlo algorithm

---

Set the number of iterations M  
**for**  $i = 1 : M$  **do**  
  | Sample the value for each  $X_i$ ,  $i \in q$  according to their PDFs  
  | Evaluate  $\alpha = f(X_i)$   
**end**  
Define the confidence interval of  $\alpha$  using the statistic of the  $M$  iterations;  $\bar{\alpha} = \frac{1}{M} \sum_{i=1}^M \alpha_i$

---

### 6.2.6 Data collection and Measurement systems

Dealing with empirical data, it is always necessary to define a measurement system to pick accurate data on-field. If the measurement system is not accurate or not precise, all the following analyses will keep an underlying error. A measurement system determines the measured value  $X_1$  of a realisation  $X_{true}$ . Having:

$$X_1 = X_{true} + \beta + \epsilon \quad (6.31)$$

Where  $X_{true}$  is the real value of the variable;  $\beta$  is the bias (accuracy or systematic error), i.e. how much far the average value of the measure is from  $X_{true}$ ;  $\epsilon$  are random errors depending on the precision of the measurement system. A good measurement procedure has:

$$\mu = \lim_{N \rightarrow +\infty} \frac{1}{N} \sum_{i=1}^N X_i = X_{true} + \beta \quad (6.32)$$

$$\lim_{N \rightarrow +\infty} \frac{1}{N} \sum_{i=1}^N \epsilon_i = 0 \quad (6.33)$$

The analysis of uncertainty aims at the definition of  $\beta$  and  $\epsilon$  of data collected on-field, providing methods to handle and process empirical data correctly.

#### Systematic errors

Systematic error (or bias errors) is determined by a measure of the accuracy of the measurement instrument. Systematic errors occur when an instrument has not an appropriate level of accuracy compared to the variable one wants to measure. For example, it is inadequate to measure the length of a warehouse rack using a ruler. A measurement instrument always requires calibration to be accurate. For example, using a calliper, the systematic error is often linked to a wrong calibration.

## Uncertainty errors

The uncertainty error (or random errors) is a measure of precision linked with the random nature of the measurement process. This is unavoidable and must be normally distributed in any empirical data collection (e.g., the processing time of a part on a workbench should be normally distributed when the process is under control).

## Mistakes

Mistakes are data points with wrong values. They may be error storing the results of an experiment or outliers which must be deleted when there are solid arguments against the value of these data points.

## Data cleaning

The process of cleaning data implies deleting data points whose value is outside the limit of the analysis one wants to perform. In particular, it often happens to have outliers whose measure is due to errors, having no connection with the real measure. We introduce two methodologies to deal with outliers.

The Chauvenet's criterion provides a simple method to deal with outliers assuming that data have a Gaussian distribution with mean  $\mu$  and standard deviation  $\sigma$ . Let consider a point  $i$  with value  $x_i$  to be suspect of being an outlier. Its  $t$ -value is defined as  $t_i = \frac{|x_i - \mu|}{\sigma}$ . Chauvenets' criterion considers the probability that  $i$  is found inside or outside a probability band defined as  $P = 1 - \frac{1}{2N}$ ; where  $N$  is the number of samples. Chauvenet's criterion defines  $z = \text{Prob}(t_i \sigma \notin P) \times N$ . If  $z < 0.5$  the point should be rejected. In other words, if a point  $i$  is too far from the mean of the normal distribution associated with the sample, it should be rejected.<sup>8</sup>

The second methodology we introduce to deal with outliers is the interquartile range (IQR). The IQR method considers the range between the 25<sup>th</sup> and the 75<sup>th</sup> percentile of the data points to detect outliers. In particular, being  $IQR = Q_3 - Q_1$ , where  $Q_1$ , and  $Q_3$  are the first and the third quartile (equivalent to the 25<sup>th</sup>, and the 75<sup>th</sup> percentiles), outliers are found below  $Q_1 - (1.5 \times IQR)$ , and above  $Q_3 + (1.5 \times IQR)$ .<sup>9</sup>

## 6.3 Statistical Distributions

This section introduces the relevant statistical distribution for the applications in the field of logistics and operations.

<sup>8</sup>The source code of the Chauvenet's method is available [here](#).

<sup>9</sup>The source code of the IQR method is available [here](#).

### 6.3.1 Normal distribution

The most important probability distribution is called normal (or Gaussian) distribution, and its PDF is defined as:

$$f(X) = \frac{e^{-\frac{(X-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{(2\pi)}} \quad (6.34)$$

Having mean  $\mu = \sum_{i=1}^N \frac{x_i}{N}$ , and standard deviation  $\sigma = \left[ \frac{1}{N} \sum_i^N (X_i - \mu)^2 \right]^{\frac{1}{2}}$ . When the distribution has a large standard deviation, its peak tends to be lower.

The Normal distribution is characterized by a concentration of the observation around the mean. It is possible to control the density of the function with reference to the distance from the mean  $\mu$  expressed in terms of the number of standard deviations  $\sigma$  (Figure 6.9).

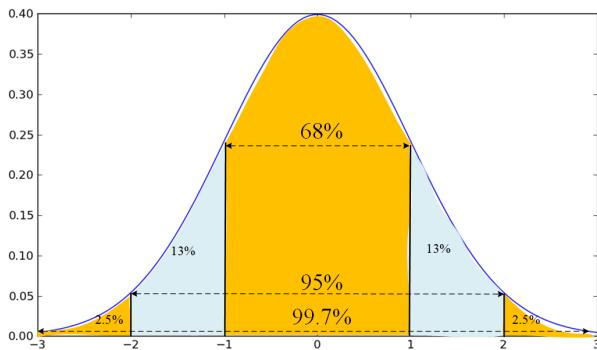


Figure 6.9: Normal distribution probability distribution function (PDF).

When a random variable  $X$  is normally distributed, we can prove that there is a probability equal to 0.95 that its mean value is found within a confidence interval of  $1.96\sigma$ .

$$\text{Prob} \left\{ -1.96 \leq \frac{X_i - \mu}{\sigma} \leq 1.96 \right\} = 0.95 \quad (6.35)$$

$$\text{Prob} \{ X_i - 1.96\sigma \leq \mu \leq X_i + 1.96\sigma \} = 0.95 \quad (6.36)$$

A normal distribution has many useful properties. In practice, it is necessary to prove that a sample is normally distributed (e.g. using statistical

tests see 6.4) to apply all the magical properties of the normal distribution. This way, by estimating the  $\sigma$  of the population with the estimator  $s = \frac{1}{N-1} \sum_{i=1}^N (X_i - \bar{X})^2$ , the mean value  $\mu$  of the population will have a confidence interval of 95% within the range  $\pm 1.96\sigma$ . The central limit theorem generalises these properties of the normal distribution to any statistical distribution.

### 6.3.2 Central limit theorem

The central limit theorem states that describing an event with a sufficiently large number  $N$  (with  $N \geq 30$ ) of random variables  $X_i$ , the arithmetic mean of these variables is normally distributed.

$$\bar{x} = \frac{X_1 + X_2 + \dots + X_N}{N} \sim N(\mu, \sigma) \quad (6.37)$$

In other words, no matter the distribution of the random variables  $X_i$ , increasing the number of experiments measuring  $X_i$ , there is a random variable describing the mean value of the experiments, and it is normally distributed.

### 6.3.3 Multivariate normal distribution

The multivariate normal distribution generalizes the univariate normal distribution to a multidimensional space. Assume  $X = (X_1, \dots, X_p)^T$  be a  $p$ -dimensional random vector.  $X$  is distributed as a multivariate normal distribution when its density function is as follows.

$$X \sim N_p(\mu, \Sigma) = f(x_1, \dots, x_p) = \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \quad (6.38)$$

Where  $\mu$  is the mean vector  $\mu = E[X] = [E[X_1], E[X_2], \dots, E[X_p]]^T$  and  $\Sigma$  is a  $p \times p$  covariance matrix  $\Sigma_{i,j} = E[(X_i - \mu_i)(X_j - \mu_j)] = Cov[X_i, X_j]$ .

### 6.3.4 Poisson Distribution

The Poisson distribution is a discrete probability distribution useful to describe the realisations of events when the average time between the event is given, but the interarrival time between the events is random. This situation is common when dealing with queues (the average throughput of the resource is given, but the interarrival time of the workload is random) or maintenance (the mean time to failure is given, but the exact failure time is unknown). The Poisson distribution has PDF:

$$Prob(X = k) = \frac{\lambda^k e^{-\lambda}}{k!} \quad (6.39)$$

The Poisson distribution identifies the probability of realisation of  $k$  events within a time interval  $\tau$  having an average number of event per time unit  $d$ , where  $\lambda = d\tau$ . Figure 6.10 identifies the shape of the PDF of the Poisson distribution using different values of  $\lambda$ .<sup>10</sup>

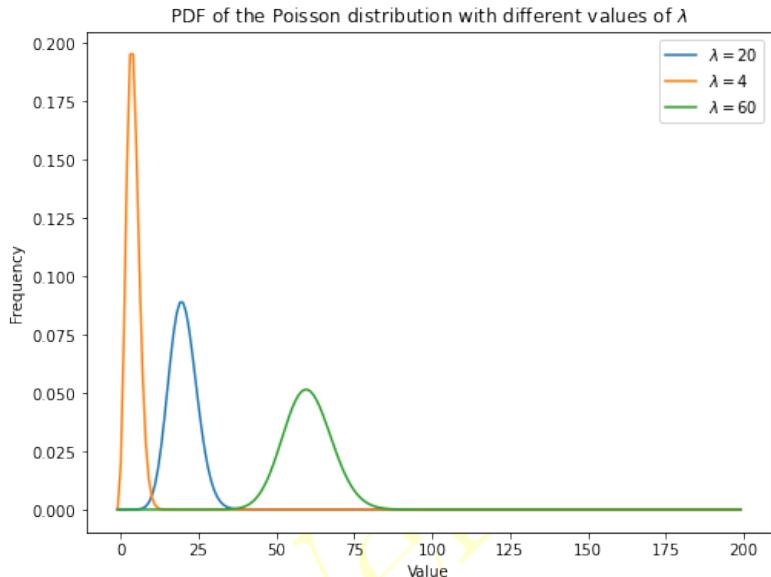


Figure 6.10: Poisson distribution probability distribution function (PDF).

### 6.3.5 Triangular Distribution

Some times empirical measurements are done on the minimum, maximum and average value of a random variable  $X$ . In these situations, we have only three values to infer the properties of the random variable  $X$ . Triangular distribution assumes  $X$  having a density with a triangular shape with its mode in correspondence of the mean value, and the vertices at the minimum and maximum values. The triangular distribution has PDF:

$$f(X) = \begin{cases} 0 & \text{if } X < a, \\ \frac{2(X-a)}{(b-a)(c-a)} & \text{if } a \leq X \leq c, \\ \frac{2}{b-a} & \text{if } X = c, \\ \frac{2(b-X)}{(b-a)(b-c)} & \text{if } c < X \leq b, \\ 0 & \text{if } b < X \end{cases} \quad (6.40)$$

<sup>10</sup>The source code of Figure 6.10 is available [here](#).

Where  $a$  is the minimum value,  $b$  is the maximum value, and  $c$  is the mean value. Figure 6.11 shows the shape of a triangular distribution.<sup>11</sup>

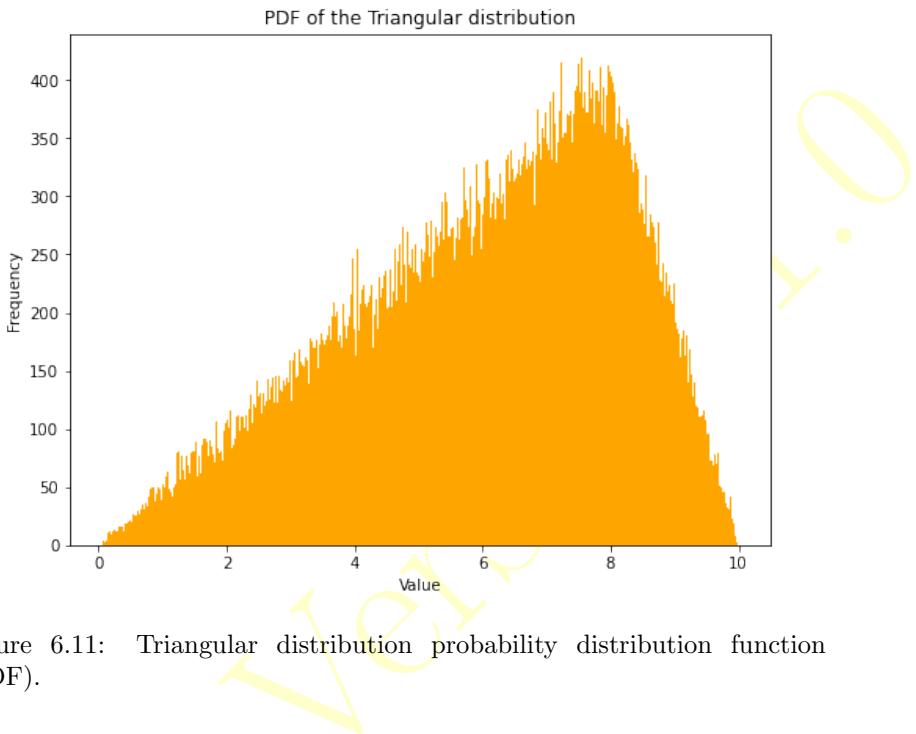


Figure 6.11: Triangular distribution probability distribution function (PDF).

## 6.4 Statistical tests

All the statistical tools introduced so far have been used with a descriptive purpose, i.e. to describe the behaviour of an experiment. At some point, it may be necessary to use statistical tools to get answers about the validity of a theory (e.g. a research intuition). Statistical tests are used for this reason. A statistical test checks if a “test distribution” (e.g., Gaussian,  $t$ ,  $\chi^2$ ) fits with the empirical data. The workflow of a statistical test is as follows.

1. Identify the problem and the parameters of interests;
2. State the null hypothesis  $H_0$ ;
3. State the alternative hypothesis  $H_1$ ;
4. Identify a level of significance  $\alpha$ ;

---

<sup>11</sup>The source code of Figure 6.11 is available [here](#).

5. Choose an appropriate statistical test;
6. Define the rejection region for the null hypothesis;
7. Compute the test and check whether  $H_0$  should be rejected or not.

$H_0$  (or, more often  $H_1$ ) is the hypothesis (e.g. the research intuition one wants to test). Engineeringly speaking,  $H_0$  is often formulated as the opposite of what one wants to test. Such that, rejecting  $H_0$  is a successful test since it supports the initial thesis.

A typical null hypothesis is that there are no differences between two parameters (e.g., the means  $\mu_1$  and  $\mu_2$ ) observed from two populations (i.e. they have the same distribution and the differences between them are only due to the chance). The hypothesis is tested within a certain level of significance  $\alpha$ . Any statistical test works calculating a probability  $p$  called *p-value*.

The *p-value* is the probability of obtaining an empirical result more extreme than the observed ones due to the sample variability, assuming  $H_0$  being true. In other words, the *p-value* is an estimation of the probability that rejecting the null hypothesis is only due to the chance. A high *p-value* may be due to a bad selection of the sample (e.g., too small) while a low *p-value* suggests the significance of the test.

For a random variable  $X$  with unknown distribution, we observe a value  $x$ . The calculation of *p-value* is:

$$p = \text{Prob} \{X \leq x | H_0\} \quad (6.41)$$

Figure 6.12 shows an example where  $X$  is an unknown distribution that is tested to have the same mean  $\mu$  of the represented normal distribution. The null hypothesis  $H_0$  is that “*the empirical data are normally distributed*”. In the figure on the left, the sample with mean value  $X$  is too far from the mean of the distribution and the *p-value* is low; for this reason,  $H_0$  is rejected. Otherwise, in the figure on the right, the sample with mean  $X$  behaves as the distribution (within the confidence interval of the test  $\alpha = 0.05$ ), the *p-value* is higher than  $\alpha$ , and  $H_0$  is not rejected.

The significance of the test depends on the value of  $\alpha$  chosen for the test. Table 6.1 shows some common values of  $\alpha$  used to accept or discard hypothesis. Please note that  $\alpha$  has to be chosen before the test depending on the context and the level of significance expected from the decision-maker. It is a bad practice to make an experiment and afterwards evaluate its results depending on the *p-value*.

### 6.4.1 Z-test (normal distribution)

Z-test is used to check if the mean value of a sample is equal to a reference value. In other words, given the value of  $\sigma$ , one wants to check how far is the

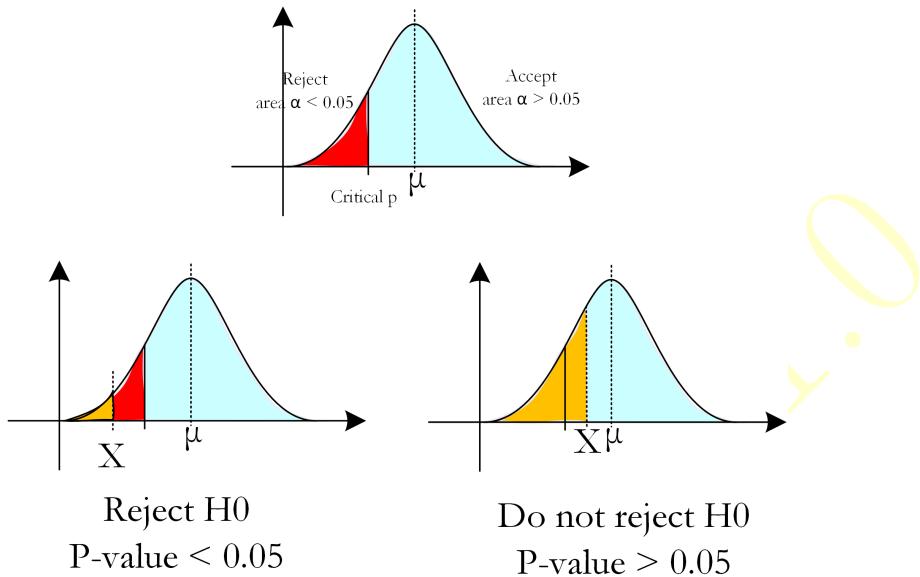


Figure 6.12: Graphical representation of a statistical test.

P-value	Diagnosis	Hypothesis
$p \geq 0.05$	Not significant	Don't reject $H_0$
$0.01 \leq p \leq 0.05$	Statistically significant	Reject $H_0$
$0.001 \leq p \leq 0.01$	Very significant	Reject $H_0$
$p \leq 0.001$	Extremely significant	Reject $H_0$

Table 6.1:  $p$ -value and levels of significance.

sample mean  $\bar{X}$  from the population mean  $\mu$ . Table 6.2 lists the summary of the  $Z$ -test.

Test summary	
Hypothesis	Data are normally distributed
Hypothesis	Sample observations are independent
Hypothesis	$\sigma^2$ is known
$H_0$	$\bar{X} = \mu$
$H_1$	$\bar{X} \neq \mu$
Application	Compare the empirical sample mean and the population mean when the empirical sample is large ( $n > 30$ )

Table 6.2: Z-test summary.

$Z$ -test assumes a normal distribution and evaluates the  $Z$ -score of the

distribution. The statistic of the test is as follows.

$$Z = \frac{\bar{X} - \mu}{\sigma / \sqrt{n}} \quad (6.42)$$

where  $n$  is the number of samples. If  $Z < -1.96$  OR  $Z > 1.96$   $H_0$  is rejected. The  $p$ -value is higher than 0.05 and the value of  $\bar{X}$  is too far from  $\mu$ . If  $-1.96 \leq Z \leq 1.96$   $H_0$  is accepted since  $\bar{X}$  falls within the acceptance region. When the value of  $\sigma$  is unknown, or the sample size is too small ( $n < 30$ ), a similar test can be performed using a  $t$ -test.

### 6.4.2 t-test

The  $t$ -test aims at checking if a sample behaves like a normal distribution when the sample size  $n$  is small. In other words, dealing with a small sample it is important to check if it is possible to apply normal distribution statistics or if the sample belongs to a different distribution. In practice, it is necessary to compare the sample mean  $\bar{X}$  and the population mean  $\mu$ . Since the population variance is unknown, it is assumed  $\sigma^2 = s^2$  i.e., the variance of the population equals the variance of the sample. The  $t$ -student distribution measures the difference between the sample mean and the population mean. The statistic of the test follows this distribution.

$$t = \frac{(\bar{X} - \mu)}{s / \sqrt{n}} \quad (6.43)$$

Where  $n$  indicates the number of samples i.e., the degree of freedom of the distribution. Consequently, the  $t$ -distribution has a different shape for different degrees of freedom (see Figure 6.13).<sup>12</sup> When the number of samples  $n$  (i.e. the degrees of freedom) approaches 30, the  $t$ -distribution has the same shape of the normal distribution. Table 6.3 illustrates the summary of this test.

### 6.4.3 $\chi^2$ -test

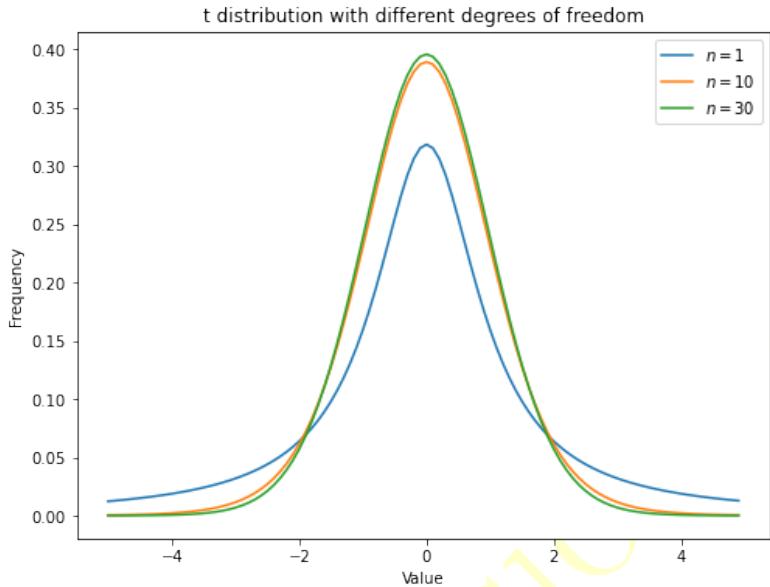
This test is used to check if a sample is distributed according to a given statistical distribution. The distribution of the test is a  $\chi^2$  distribution defined as:

$$\chi_k^2 = \sum_{i=1}^k x_i^2 = x_1^2 + \dots + x_k^2 \quad (6.44)$$

$x_1, \dots, x_k$  are random variables normally distributed and  $k$  is the number of degrees of freedom. To check if a sample fits a statistical distribution (e.g.,

---

<sup>12</sup>The source code of Figure 6.13 is available [here](#).

Figure 6.13: PDF of the  $t$ -distribution with different degrees of freedom.

Test summary	
Hypothesis	Data are normally distributed
Hypothesis	Sample observations are independent
Hypothesis	$\sigma^2 = s^2$
$H_0$	$\bar{X} = \mu$ (using a double-tail test)
$H_0$	$\bar{X} \geq \mu$ (using a single-tail test)
Application	Compare the empirical sample mean, and the population mean.
Application	Compare a single point, and a sample mean to check if it belongs to the same population.
Application	Compare the mean of two dependent empirical samples with an expected difference.
Application	Compare the mean of two independent empirical samples.

Table 6.3:  $t$ -test summary.

a normal distribution), the events are divided into  $k$  subset each one having an observed frequency  $o_k$  (sample) and an expected frequency  $e_k$  (from the distribution). The definition of the  $k$  classes is very important. The more the classes, the more one can check the fit with a statistical distribution. A good rule is that % of the classes has at least 5 items and no class is empty. The statistic of the test is calculated as follows:

$$s = \sum_i^k \frac{|o_i - e_i|^2}{e_i} \quad (6.45)$$

The value  $s$  is, then, compared to the value of a random variable  $\chi^2$  distributed with  $n - 1$  degrees of freedom. The greater the value of  $s$ , the greater the gap with the theoretical distribution. To quickly compare the obtained value of  $s$ , one defines a confidence interval  $\alpha$ , i.e. the maximum  $p$ -value accepted and the value of the degree of freedoms (i.e. the number of independent variables). The hypothesis is discarded, from the definition of the  $p$ -value, when  $s \geq \chi_{dof}^2$ . Note that the value  $\alpha = 0.1$  is the most restrictive test since it considers a smaller region of acceptance than the other (the non-acceptance region is 10% of the whole data distribution). The hypothesis about the data and  $H_0$  of the test are summarised in Table 6.4.

Test summary	
$H_0$	The sample is distributed as a given probability distribution
Application	Compare the theoretical classes from a distribution with the classes defined from an empirical sample

Table 6.4:  $\chi^2$  test summary

#### 6.4.4 F-test

This test aims at checking if two samples are normally distributed with the same variance. The statistic of this test is distributed as a Fisher-Snedecor distribution.

$$F = \frac{\frac{N_1}{m}}{\frac{N_2}{n}} \quad (6.46)$$

Where  $N_1$  and  $N_2$  are independent random variable  $\chi^2$  distributed with  $m$  and  $n$  degrees of freedom. The test assumes  $X$  and  $Y$  are normally distributed.  $X$  has  $n$  samples and  $Y$  has  $m$  samples. The null hypothesis is  $H_0 = \sigma_X^2 = \sigma_Y^2$ . The statistic follows a Fisher distribution with  $n - 1$  and  $m - 1$  degrees of freedom:

$$F = \frac{S_X^2}{S_Y^2} \quad (6.47)$$

The value of  $F$  can be easily calculated having a dimensional space a number of  $p$  parameters considering the sum of the squared residuals.

$$F = \frac{\left( \frac{SSR_X - SSR_Y}{p_X - p_Y} \right)}{\frac{SSR_Y}{n - p_Y}} \quad (6.48)$$

The summary of the  $F$ -test is presented in Table 6.5.

Test summary	
Hypotheses	$X, Y$ normally distributed
$H_0$	$\sigma_X^2 = \sigma_Y^2$
Application	Compare the variance of the empirical sample and the population.

Table 6.5: F-test summary.

## 6.5 Time series analysis

A time series (TS) is a series of the realisation of a random variable measured at constant time intervals. Time series can be used to forecast future values or to classify the realisations according to the properties of the TS (e.g., the seasonality). Even if both these applications sound intuitive, there is a lot of theory and math behind TSs. The entire theory of TS analysis is based on statistics.

The aim of TS analysis is the definition of a PDF describing the realisation of the events over time. Sometimes observations are some way linked to the others. This concept can be easily recognised thinking about the “continuity” of the nature around us (this is somehow related to the Principle of Least Action, already introduced in chapter 3.2). Besides, it may exist a seasonality involving observations distant in the time (e.g. every summer is warmer than every winter). All these aspects drop the hypothesis of the independence of the observed variable that is commonly used in statistics to built simpler models. Under the independence hypothesis, it is possible to assume that the joint PDF of  $X_1, \dots, X_n$  random variables equals  $f(x_1, \dots, x_n) = \prod_{i=1}^n f(x_i)$ . This is not true for TSs, and our work will get harder.

Since it is easy that each observation of a TS may depend on the previous: there is a sort of “influence” between them (in general, we can say that a TS has *memory*). This is good news since it suggests that we could check historical values to make forecasts, that is one of our purposes.

TSs are modelled as stochastic processes; for this reason, we introduce the notation indicating  $X(\omega, t)$ , where  $X$  is the stochastic process. A TS behaves as a set of events  $\omega$ , one for each  $t$  step, generated by  $X(\omega, t)$ . The random variable  $X_t$  models the event at each time period  $t$ . Figure 6.14 shows the inputs and outputs of a stochastic process. We aim at describing the generating process  $X(\omega, t)$  modelling the behaviour of the process to make forecasts.

When modelling a TS as a stochastic process, there is a limit. A TS is composed of a single observation for each couple  $(\omega, t)$ . This is equal to have each event at a time  $t$  represented by a single record (as having a single TS of the  $n$  represented in Figure 6.4). In practice, a single observation  $X(t)$  is

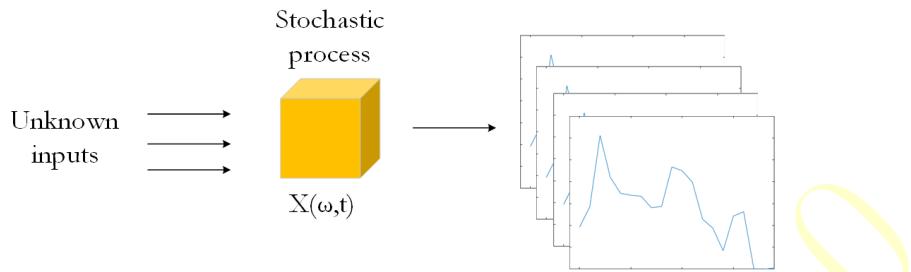


Figure 6.14: Representation of a stochastic process. 

assumed to be representative of the entire event  $\omega$  at time  $t$ . <sup>13</sup>

### 6.5.1 Time series decomposition

TS decomposition is one of the approaches used to estimate the parameters of a stochastic process. The first assumption is made on how the stochastic process works. It is assumed it generates the TS based on three independent components.

- A trend component  $T(t)$ ;
- A seasonal component  $S(t)$ ;
- A residual (random) component  $R(t)$ .

The literature proposes two models to mix these components. An additive (see equation 6.49) and a multiplicative model (see equation 6.50). Figure 6.15 illustrates two realisations of an additive a multiplicative model.<sup>14</sup>

$$X(t) = T(t) + S(t) + R(t) \quad (6.49)$$

$$X(t) = T(t) \times S(t) \times R(t) \quad (6.50)$$

For the sake of brevity, the following paragraphs consider a TS modelled through additive model (see equation 6.49) given that a multiplicative series can be transformed into an additive one using a logarithm transformation.

$$\begin{aligned} \log(X(t)) &= \log(T(t) \times S(t) \times R(t)) = \\ &= \log(T(t)) + \log(S(t)) + \log R(t) \end{aligned} \quad (6.51)$$

<sup>13</sup>The package logproj provides methods to deal with time series [here](#).

<sup>14</sup>The source code of Figure 6.15 is available [here](#).

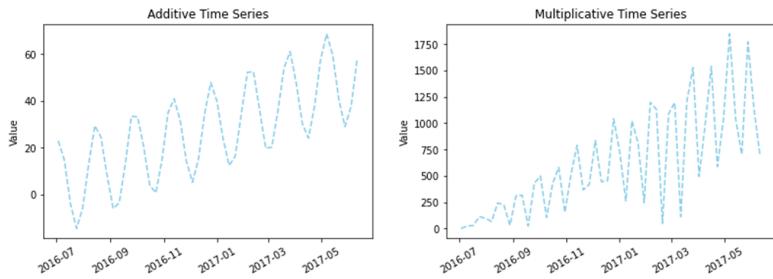


Figure 6.15: Comparison between additive and multiplicative time series

For this reason, all the techniques here presented are applicable to multiplicative models (see equation 6.50) too. In practice, one can fit both of them and measure their goodness of fit choosing the model which better describe the empirical measurements.

An option to fit an additive model is to get the trend component of the series as a linear regression model fitted by ordinary least square (OLS<sup>15</sup>). The result is a function  $T(t) = mt + q$ , where  $m$  is the angular coefficient and  $q$  is the intercept of the straight line best approximating the trend of  $X(t)$ . Alternatively, the trend can be estimated using a smoothing function, for example, using a Moving Average (MA) with a time window equal to the seasonality of  $X(t)$ <sup>16</sup>. Equation 6.52 illustrates the formula of the MA; the time window is equal to  $2h+1$  (e.g.,  $2h+1 = 7$  for a weekly seasonality of a time series having one sample per day).

$$T_t = \frac{1}{2h+1} \sum_{i=t-h}^{t+h} X_i \quad (6.52)$$

Once the trend component  $T(t)$  has been extracted, the residual part (see Figure 6.16) of the TS equals<sup>17</sup>:

$$S(t) + R(t) = X(t) - T(t) \quad (6.53)$$

To estimate the seasonal component  $S(t)$  averaging is performed on the residual series  $S(t) + R(t)$ . Averaging works by grouping all the observation of the same seasonal period and applying average on it. The resulting value defines  $S_t$ . It is necessary to know the seasonality period to perform averaging (see Figure 6.17). This can be done by the visualisation of the

<sup>15</sup>Section 9.2 provides additional details of the OLS method.

<sup>16</sup>Sometimes, the seasonality of  $X(t)$  is unknown. Section 6.5.3 shows how to detect the seasonality of a TS.

<sup>17</sup>The source code of Figure 6.16 is available [here](#).

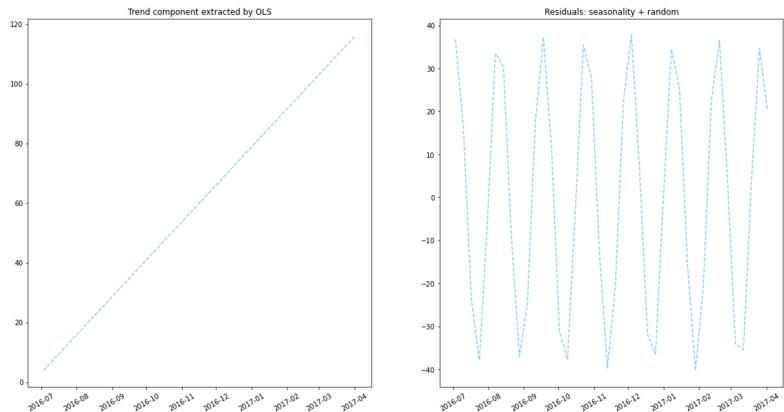


Figure 6.16: Extraction of the trend component from the TS.

graphs or analytically using the Fourier transform analysing the frequency domain of the TS (see Section 6.5.3).

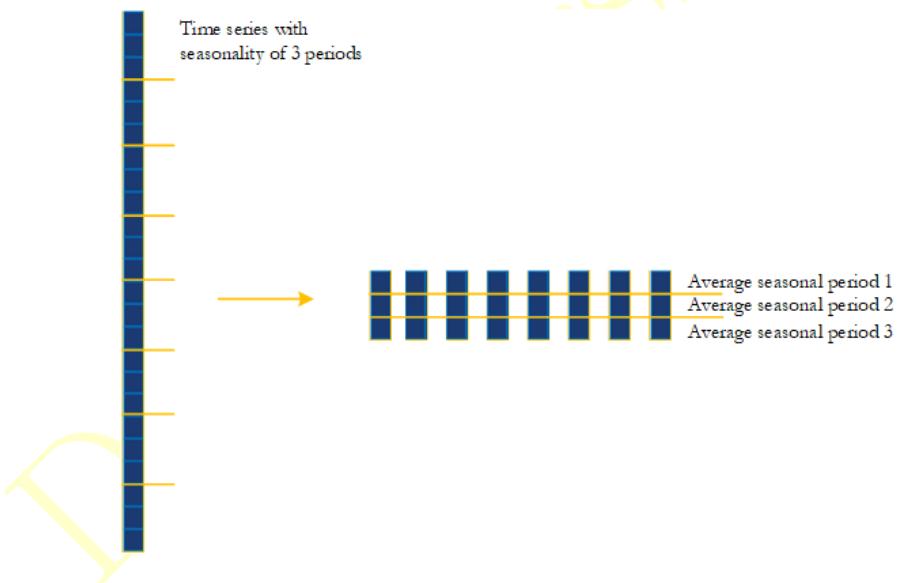


Figure 6.17: Visualisation of the averaging method to estimate  $S(t)$

The residuals  $R_t$  are obtained, again, by subtraction. If the residuals are randomly distributed, our model is interpreting the behaviour of the TS correctly. Otherwise, if the residuals show a pattern, the estimations of  $T(t)$  and  $S(t)$  need more accuracy, or the choice of additive or multiplicative

model is wrong. Figure 6.18 shows the seasonal component and residuals. The residuals are randomly distributed, and it is possible to conclude that the decomposition process worked properly. Nevertheless, their magnitude is significant; collecting a higher number of samples would help in practice to reduce their magnitude and better detect the seasonality<sup>18</sup>.

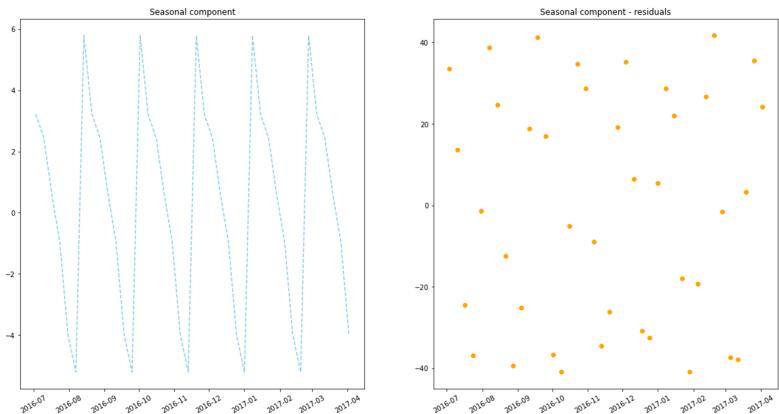


Figure 6.18: Extraction of the seasonal component from the TS.

### 6.5.2 ARIMA models

A more complex way to model TSs comes from the autoregressive and moving average (ARIMA) models. These models are based on the Wold's decomposition theorem stating that “Every covariance-stationary time series  $X(t)$  can be written as the sum of two time series, one deterministic and one stochastic. In other words, given a stationary stochastic process  $X_t$  with mean value  $\mu$  it is always possible to decompose the process into  $X_t = Z_t + V_t$  such that  $\text{cov}(Z_t, X_t) = 0$ . In particular:

$$V_t = \mu + \sum_{j=1}^{+\infty} [\alpha_j \sin(\omega_j t) + \beta_j (\omega_j t)], \quad 0 \leq \omega_j \leq \pi \quad (6.54)$$

$$Z_t = \sum_{j=0}^{+\infty} \psi_j a_{t-j} \quad (6.55)$$

Equation 6.54 models the autoregressive component where  $V_t$  is the deterministic part of the model;  $\omega_j$  is a fixed frequency and  $\alpha_j$  and  $\beta_j$  are un-

---

<sup>18</sup>The source code of Figure 6.18 is available [here](#).

correlated white noise <sup>19</sup> processes.  $Z_t$  (see equation 6.55) is the stochastic process and it is a moving average of infinity order where  $a_t$  is the prediction error. In practice, a consequence of the Wold's theorem is that once a TS has been transformed into a stationary TS it can be modelled as a linear function of a white noise process. A stochastic process is stationary when it has constant mean and variance and its autocovariance only depends on the time lag  $k$  (not by the time  $t$ ). Using equations:

$$\begin{aligned} E(X_t) &= \mu < \infty, \forall t \in T \\ Var(X_t) &= \sigma_X^2 < \infty, \forall t \in T \\ Cov(X_t, X_{t-k}) &= \gamma_k < \infty, \forall k \in T \end{aligned} \quad (6.56)$$

If these conditions are met, it is possible to apply an ARIMA model to a TS to model it as a sum of an autoregressive process (AR) (derived from  $V_t$ ) and a moving average process (MA) (derived from  $Z_t$ ). In general, a TS is not stationary, and it is not possible to directly apply ARIMA models. A trend in the series, for example, violates equations 6.56. For this reason, detrending is almost always necessary to get a series meeting the stationarity condition. Box and Jenkins [1] introduced some transformation to deal with non-stationary TS (see Table 6.6).

Relationship between $\sigma$ and $\mu$	Transformation
$\sigma = k\mu$	$Z(t) = \log(X(t))$
$\sigma = \mu$	$Z(t) = X(t)$
$\sigma = k\sqrt{\mu}$	$Z(t) = 2(\sqrt{X(t)} - 1)$
$\sigma = k\mu^{\frac{2}{3}}$	$Z(t) = 3(X(t)^{\frac{1}{3}} - 1)$
$\sigma = k\mu^2$	$Z(t) = -(\frac{1}{X(t)} - 1)$

Table 6.6: Transformations to obtain a stationary TS

Once the TS has been transformed into a stationary one, it is possible to fit an ARIMA model choosing adequate parameters  $p$  and  $q$ , indicating the autoregressive (AR) order and the moving average (MA) order. At this purpose, the autocorrelation functions PACF, and ACF are studied. The echo phenomenon exemplifies the effect of the autocorrelation. After a certain amount of time units (i.e. time lags), the echo overlaps original message altering the sound waveform. The echo of the signal of a TS is the seasonality (i.e. certain weeks or months of the year where the TS is amplified). To detect this phenomenon, we use ACF and PACF as defined

<sup>19</sup>A white noise process  $a_t \sim WN(0, \sigma_a)$  is such that:  $E(a_t) = 0$ ;  $Var(a_t) = \sigma_a^2$ ;  $cov(a_t, a_{t-k}) = 0 \forall t \in T, k \neq 0$ .

in Section 6.1.2. Figure 6.19 illustrates a TS with its ACF and PACF, while Figure 6.20 illustrates the same series with ACF and PACF after detrend using OLS.<sup>20</sup>.

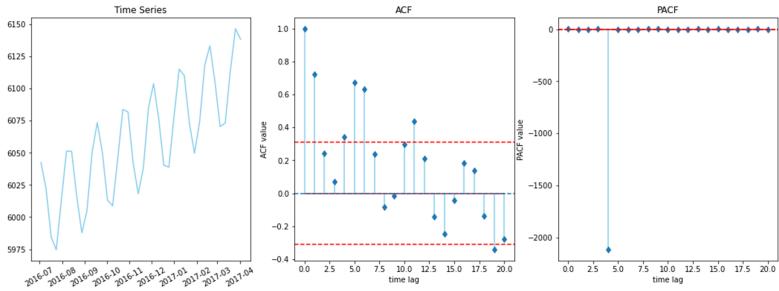


Figure 6.19: ACF and PACF of the original TS.

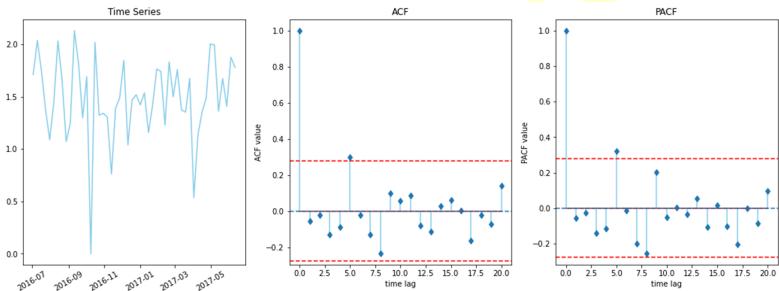


Figure 6.20: ACF and PACF of the detrended TS.

Stationarity can be tested using the Dickey-Fuller test for stationarity. Alternatively, a series is stationary if its ACF and PACF decrease, i.e. the ACF, and PACF correlograms tends to zero asymptotically. The last significant value of the PACF is used for the parameter  $p$  (the order of the AR model) while the last significant lag value of the ACF is used for the parameter  $q$  (the order of the MA model). In this case, we would try to fit a model ARIMA(1,1). Note that if the PACF goes to zero immediately, the most important value is '1' (since a TS is always autocorrelated with itself at the same time lag). In this case,  $T_t$  is constant and no waveform (i.e. seasonality) exists. When the ACF suggests no values, instead, '0' should be the order used for  $q$ .

---

<sup>20</sup>The source code of Figure 6.19, and Figure 6.20 is available [here](#).

### 6.5.3 Fourier transform

All the models introduced in the previous paragraphs assume that a seasonality exists and its lag is defined (or can be defined analysing the graph of the time series). Nevertheless, in some cases, it may be necessary to have analytical methods to study the seasonality of a series. For this reason, we introduce the spectrum analysis and the Fourier transform. Spectrum analysis is a methodology widely used in telecommunication for the analysis of signals. Signals (analogue or digital) are usually periodic and characterised by a period  $\tau$ , due to their sinusoidal behaviour. A periodic (i.e. sinusoidal) signal (see Figure 6.21) can be modelled as<sup>21</sup>:

$$y = S(t) = A \sin(\omega t + \phi) \quad (6.57)$$

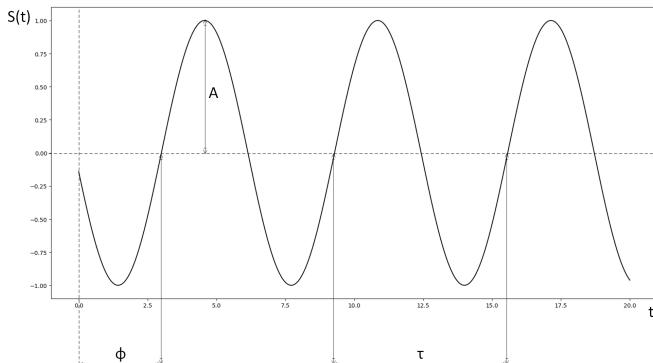


Figure 6.21: Model of a periodic signal.

Where  $A$  is the amplitude of the signal,  $\phi$  is the phase (translation on the time axis),  $\omega$  is the angular velocity, i.e. the number of periods within a time interval of  $2\pi$ . The frequency  $f$  is linked to the period  $\tau$  and to the angular velocity  $\omega$  by the following.

$$f = \frac{1}{\tau} = \frac{\omega}{2\pi} \quad (6.58)$$

Given the equation 6.58, the  $S(t)$  can be expressed in terms of the frequency  $f$ .

$$y = A \sin(f 2\pi t + \phi) \quad (6.59)$$

Telecommunication uses different strategies to transmit signals avoiding losses during the transmission. It often happens to transmit signals digitally.

---

<sup>21</sup>The source code of Figure 6.21 is available [here](#).

When the source is analogue (i.e. continuous by nature) the signal has to be sampled. Sampling means removing part of the signal, but if it is performed correctly, it does not remove any information. Sampling is performed at a fixed frequency  $f_s$  i.e., each sample has a distance in time from the previous one equal to  $T$ . To properly maintain the level of information of a signal,  $f_s$  must be chosen adequately. The Nyquist-Shannon sampling theorem demonstrates that given a periodic signal  $g(t)$  with maximum frequency  $f_M$  (i.e. a bandwidth  $[0, f_M]$ ), can be completely defined (i.e. without loss of information) using a sampling frequency  $f_s \geq 2f_M$  i.e. a sampling step  $t_s \leq \frac{1}{2f_M}$ . Figure 6.22 shows the samples of a signal generated by a continuous source.<sup>22</sup>

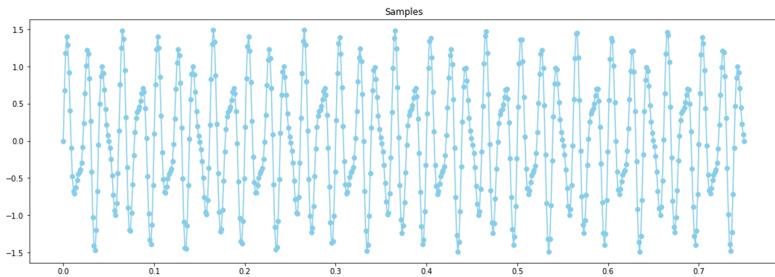


Figure 6.22: signal with  $N = 600$  and  $f_s = 800$  Hz.

The original signal can be recreated using the Fourier transform of the sampled signal, and investigating its behaviour in the frequency domain. The signal is expected to have a maximum original frequency of 400 Hz (i.e. it has been sampled correctly, accordingly with the Nyquist-Shannon theorem).

The Fourier theorem states that any periodic function  $x(t)$  may be expressed as a sum of infinite terms of sine and cosine terms (called Fourier series), each of them with a specific amplitude and phase complex coefficient  $c_n$  called the Fourier coefficient.

$$c_n = \frac{1}{T_0} \int_{T_0} x(t) e^{-2\pi n f_0 t} dt \quad (6.60)$$

Fourier transform is used to calculate the values of  $c_n$ .

$$X(f) = F\{x(t)\} = \int_{-\infty}^{+\infty} x(t) e^{-j2\pi f t} dt \quad (6.61)$$

The representation of these frequencies is obtained through amplitude and phase spectra which represent the absolute value and the argument of

---

<sup>22</sup>The source code of Figure 6.22 is available [here](#).

the complex coefficients  $c_n$ . The amplitude spectrum of the sampled signal is shown in Figure 6.23.<sup>23</sup>

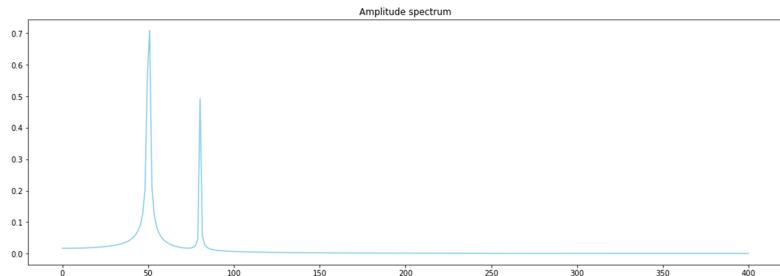


Figure 6.23: Amplitude spectrum of the signal.

The amplitude spectrum shows that two sinusoids with frequency 50 and 80 Hz generate the original signal; the harmonics are showed on the chart. Looking at the source code, the generating function of the samples was  $y = \sin(50 \times 2\pi x) + 0.5 \sin(80 \times 2\pi x)$  which confirms the results of the transform.

Dealing with time series, we can use the Fourier transform to investigate if a periodical signal can be used to model the seasonal component of the series. In particular, we are looking for  $f$  of the model in equation 6.58. Where  $f$  can be reasonably be expressed in  $\text{week}^{-1}$ , i.e.  $f^{-1}$  defines the length of a period, i.e. the length of the seasonality.

Time series are usually extracted from a database which is digitalised and already sampled (e.g. daily/monthly/yearly) so it is difficult to define a priori the number of samples. This can be defined by different grouping strategy (e.g., daily/weekly). It is, anyway, essential to verify if the number of samples allows for resilient inference on the seasonality.

To perform Fourier analysis on a time series, it is necessary to detrend it first, as shown in 6.5.1. At this stage, the time series fluctuates around its mean with an unknown frequency and can be modelled as a periodic signal using the Fourier transform.

## 6.6 Bayesian Statistics

The statistics illustrated so far is entirely based on the observation of physical phenomena. The events are observed, measured, and the outcomes of these experiments are analysed in a *frequency* fancy. The values occurring the most are the most probable. This statistics is based on the frequentist

---

<sup>23</sup>The source code of Figure 6.23 is available [here](#).

approach. Sometimes we do not have the possibility of measure the variable of our interest; nevertheless, we have *beliefs* on the behaviour of this variable, and we may be interested in expressing these believes in terms of probability.

For example, when measuring partial data of a total quantity, we have the belief that the total quantity is higher than the measured one. We can measure a variable using two different measurement systems, having the belief that one outperforms the other under certain circumstances. We may start an experiment having prior beliefs on the expected outcome. Bayesian statistics helps in all these situations. The main theorem of Bayesian statistic is the Bayes' theorem:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A)}{P(B)} \quad (6.62)$$

This theorem is also known as the theorem of conditional probability. While frequentist statistics assume  $A$ , and  $B$  being events; the Bayesian approach defines  $A$  as the prior, and  $B$  the posterior. The prior  $A$  defines the belief we have before an experiment starts, i.e. before starting collecting measurement. The information of the measurements is, then, contained in the posterior  $B$ . Bayesian statistics aims at matching the prior and the posterior by assuming that the prior  $A$  is true. In practice, we have observations, we expect they behave as  $A$  describes, but we observe a behaviour  $B$ , and we need to correct it. Bayesian statistics is the perfect tool to match prior models with posterior empirical observations to predict their behaviour. Bayesian tools work similarly to machine learning models since all machine learning models aim at the definition of a joint probability distribution between a prior and a posterior. For this reason, these methods are presented in 11.3.

## Further reading

Supplementary reading materials can be found in [2], [3], [4], [5].

## Bibliography

- [1] G. Box and G. Jenkins, *Time Series Analysis: Forecasting and Control*. 1970.
- [2] R. M. Sauter, *Introduction to Statistics and Data Analysis*, vol. 44. 2002.
- [3] D. Ruppert and M. D. S., *Statistics and Data Analysis for Financial Engineering*. Springer, 2015.

- [4] S. Brandt, *Data Analysis: Statistical and Computational Methods for Scientists and Engineers*. Springer, 2014.
- [5] R. M. Heibergerer and H. Burt, *Statistical Analysis and Data Display*. Springer, 2015.

Draft Version 1.0

Draft Version 1.0

## Dimensionality reduction

We need data, but maybe not all of them.

Databases, Internet of Things, big data provide tons of data per second. There is an obvious obstacle in processing all of them since a large computational power and a lot of storage memory are needed. Besides, a portion of this data does not provide useful information since it may have a fixed value or an extremely high correlation with the other data. We introduce dimensionality reduction strategies to smooth the process of data processing and to get information fastly and efficiently. These strategies aim at getting the highest level of information possible using the lowest amount of input data.

Let assume that data from our sources is organised in a dataset  $X_{N \times P}$  having  $N$  observation (rows) and  $P$  features (columns). Each observation describes the realisation of a phenomenon characterised by the  $P$  features. Dimensionality reduction aims at explaining the information of each row in  $X$  using only  $K$  features, with  $K < P$ . To meet this goal, features can be:

- extracted (i.e. the  $P$  features are transformed to explain the majority of the information in  $X$ );
- selected (i.e. a subset  $K$  of  $P$  explains the majority of the information in  $X$ ).

The following paragraphs explore techniques belonging to these two methodologies.<sup>1</sup>

<sup>1</sup>The package logproj provides methods to deal with dimensionality reduction [here](#).

## 7.1 Feature extraction

The most common feature extraction strategy is the principal component analysis (PCA); this section introduces it using the singular value decomposition (SVD) to decompose a dataset  $X$ . SVD and PCA can be used when a learning table is composed of many columns (e.g. with image recognition or dummy columns from an initial dataset containing categorical variables converted into binary features). SVD and PCA use eigenvalues and eigenvectors to transform the features space reducing overfitting. Eigenvectors are defined as vectors whose direction does not change when a linear transformation is applied to them. Eigenvalues are the scalar used to transform the eigenvectors. Given a matrix  $A$ , we can write:

$$Ax - \lambda x = 0 \quad (7.1)$$

Where  $x$  is the eigenvector of  $A$  and  $\lambda$  are the eigenvalues of  $A$ .

### 7.1.1 Singular Value Decomposition (SVD)

In general, any matrix  $X_{N,P}$  can be decomposed into a product of a mixing matrix  $U_{N,K}$  and a dictionary matrix  $V_{P,K}$  such that:

$$X = UV^T \quad (7.2)$$

A row  $x_i \in X$  is a linear combination (according to an entry  $u_i \in U$ ) of the linearly independent elements  $v_i \in V$ . Singular value decomposition (SVD) is a matrix factorisation technique which decomposes a matrix  $X$  into:

$$X_{N,P} = U_{N,K} D_{K,K} V_{P,K}^T \quad (7.3)$$

Table 7.1 shows the information content of the three matrices produced by the SVD.

Matrix	$U$ , left singular-vectors	$D$ , singular values	$V$ , right singular-vectors
Dimension	$N \times K$	$K \times K$	$P \times K$
Object	Observation	Principal components	Variables
Content	Mixing coefficients	Square roots of the eigenvalues of $X^T X$	Dictionary of patterns
Properties	Orthogonal matrix	Diagonal matrix	Orthogonal matrix
Linear transformation	Perform rotations	Perform stretches	Perform rotations

Table 7.1: elements of the SVD.

### 7.1.2 Principal Component Analysis (PCA)

PCA aims at reducing  $X_{N,P}$  to  $C_{N,K}$  with  $K < P$  where  $K$  is the number of orthogonal vectors used to explain the variability of  $X$ . The PCA projects the elements of  $X$  in the  $K$  directions defined by  $V_K$  such that the variance of the  $N$  observation is maximised along this direction. In practice, PCA projects the reference system of the  $P$  variables onto a new  $K$ -dimensional coordinate system  $V$ . All the entries  $x_i$  of the matrix  $X_{(N,P)}$  are converted into the new reference system at  $x_{i(1,P)}^T v_{(P,1)}$ . PCA defines:

$$v_{(P,1)} = \arg \max_{v: \|v\|=1} \frac{1}{N} \sum_i (x_i^T v)^2 \quad (7.4)$$

It is necessary to remember that, before applying PCA:

1. Data has to be centred; i.e.  $X = X - \bar{x}^T$ , since the equation 7.4 maximises the variance of the sample data.
2. Data has to be standardised; i.e. having the same variance of all the variables  $P$ . Otherwise, variables with higher (absolute) variances will overcome the others.

In addition to these, the maximisation objective is constrained to  $\|v\| = 1$  for two reasons:

1. We are only interested in the direction of the projection, not in its magnitude;
2. Without this constraint, the maximisation objective would be unbounded.

Let, now, express the maximisation function at:

$$\arg \max_{v: \|v\|=1} \frac{1}{N} v^T X^T X v \quad (7.5)$$

Please note that the covariance matrix  $S_{xx}$  equals  $\frac{1}{N} X^T X$ , then:

$$\arg \max_{v: \|v\|=1} v^T S_{xx} v \quad (7.6)$$

By using Lagrangian multipliers  $\lambda$ , it is possible to express the maximisation function at:

$$\max v^T S_{xx} v - \lambda(v^T v - 1) \quad (7.7)$$

It is then, possible proceed to calculate the stationary points, i.e. where the derivative of the function (regards to  $v$ ) is equal to zero.

$$\frac{d}{dv} \{ v^T S_{xx} v - \lambda(v^T v - 1) \} = v^T S_{xx} - \lambda v = 0 \quad (7.8)$$

$$S_{xx}v = \lambda v \quad (7.9)$$

We conclude that:

- $v$  is the eigenvector of  $S_{xx}$ ;
- $\lambda$  is the eigenvalue of  $S_{xx}$ .

It is, then, possible to conclude that the variance is maximised when  $v$  is an eigenvector of  $S_{xx}$  (i.e. the covariance matrix) corresponding to the largest eigenvalue  $\lambda$ . In practice, the PCA is performed using the sample covariance matrix  $S_{xx} = \frac{1}{N-1}X^T X$  and the SDV of  $X^T X$ .

$$X^T X = (UDV)^T (UDV^T) = VD^T U^T UDV^T \quad (7.10)$$

Since  $U$  is orthogonal,  $U^T U = I$ , then:

$$X^T X = VD^T DV^T \quad (7.11)$$

Since  $D$  is a square matrix,  $D^T D = D^2$ , then:

$$\begin{aligned} X^T X &= VD^2V^T \\ V^T X^T X V &= D^2 \\ \frac{1}{N-1} V^T X^T X V &= \frac{1}{N-1} D^2 \\ V^T S_{xx} V &= \frac{1}{N-1} D^2 \end{aligned} \quad (7.12)$$

Then:

- The eigenvectors of  $S_{xx}$  are the right-singular vectors  $V$ ;
- The eigenvalues  $\lambda_k$  (which are the variance of the components) are equal to  $\frac{1}{N-1}d_k$ , where  $d_k$  are the squared singular values.

In conclusion, the PCA performs the SVD on the data covariance matrix  $S_{xx}$ , and it produces three outputs:

1. The principal component directions, i.e. the right singular vectors  $V_{K,P}$  of an SVD which are the eigenvectors of  $X^T X$ .
2. The principal components, i.e. a matrix  $C_{N,K}$ , obtained projecting  $X_{N,P}$  onto the principal components directions  $V_{K,P}$  the left singular vectors  $U_{N,K}$ :

$$C_{N,K} = X_{N,P} V_{P,K} = UDV_{N,P}^T V_{P,K} = UD_{N,K}^T \quad (7.13)$$

Accordingly,  $U$  (the left-singular value matrix) is the matrix of  $u_j$  with the projections of the row vectors of  $X$  in the new reference system (direction  $v_j$ ) scaled by  $d_j$ . In practice, the PCA produces  $k$  principal components ( $k = 1, \dots, K$ ) which are a linear combination of the original variables:

$$c_k = x_1 u_1 + x_2 u_2 + \dots + x_P u_P \quad (7.14)$$

3. The variance of each component, given by the eigenvalues  $\lambda_k = 1, \dots, K$ . This is obtained from the singular values in  $D_{K,K}$ .

$$\text{var}(c_k) = \frac{1}{N-1} d_k^2 \quad (7.15)$$

The theory does not explain how to choose the number of PCs. This information can be obtained by building a curve showing the information content of each principal component. Figure 7.2 presents this curve based on the data of the sample *wine dataset*. The curve illustrates the percentage of the variance of the dataset  $X$  given a certain number of components  $K$ . In this case, the dataset  $X$  count 13 features, but the first six components are enough to explain 80% of the variance of  $X$ .<sup>2</sup>

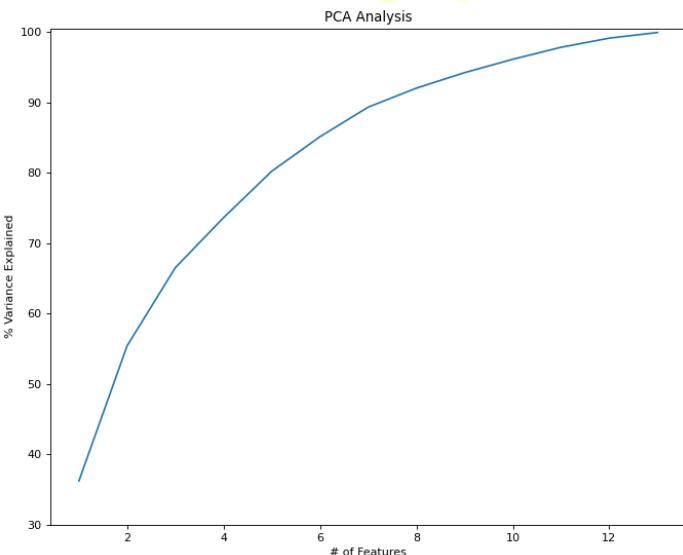


Table 7.2: Cumulative curve of the information content of the principal components.

---

<sup>2</sup>The source code of Figure 7.2 is available [here](#).

### 7.1.3 Multi-dimensional scaling

In some cases, we do not have a matrix  $X_{N,P}$  with observations and features but a distance matrix  $D_{N,N}$  expressing a pairwise distance between each observation, for a given feature. In this case, it is recommendable to turn the  $D$  matrix into a  $X$  one, but this involves the approximation of the distance values into a  $K$ -dimensional space.

Multidimensional scaling aims at this goal, considering a  $D_{N,N}$  and finding a low-dimensional projection of the data such that a stress function is minimised.

$$\min \text{stress}(X) = \sum_{i \neq j} (d_{ij} - \|x_i - x_j\|)^2 \quad (7.16)$$

### 7.1.4 t-SNE

t-SNE is a common technique when the number of features  $P$  is high. This technique tries to separate the variables rather than combining their effect (as in PCA); also, it provides effective visualisation in low dimensional space (e.g.,  $K = 2$ ).

t-SNE algorithm proceeds step by step, determining the similarity between each observation of the dataset  $X$  according to the values of its features  $P$ . The similarity is measured as the distance between each observation and a Gaussian curve which is, then normalised to 1. Once all similarity values are calculated, a similarity matrix  $D$  is defined.

All the observations are randomly scattered on the  $K$ -dimensional space, and their distance is measured as the distance between the observation and a  $t$ -distribution populating the matrix  $D_t$ . At this stage, points are re-organised in the  $K$ -dimensional space one by one to make  $D_t$  similar to  $D$  defining compact clusters.

## 7.2 Feature selection

Feature selection strategies implement heuristics to define a subset of the  $P$  features to train learning algorithms. The following paragraphs illustrate these strategies.

### 7.2.1 Selection by correlation

The correlations between the features of the input dataset  $X$  are values to check carefully before training a learning algorithm. If two features are highly correlated, it may be necessary to exclude one of them since the other already describes the variability of the dataset. The correlation matrix is used for this purpose to identify the correlation coefficients of all the possible

couples of variables. Figure 7.3 shows an example of a correlation matrix from the wine dataset using different colour gradients to highlight positive and negative correlations.<sup>3</sup>

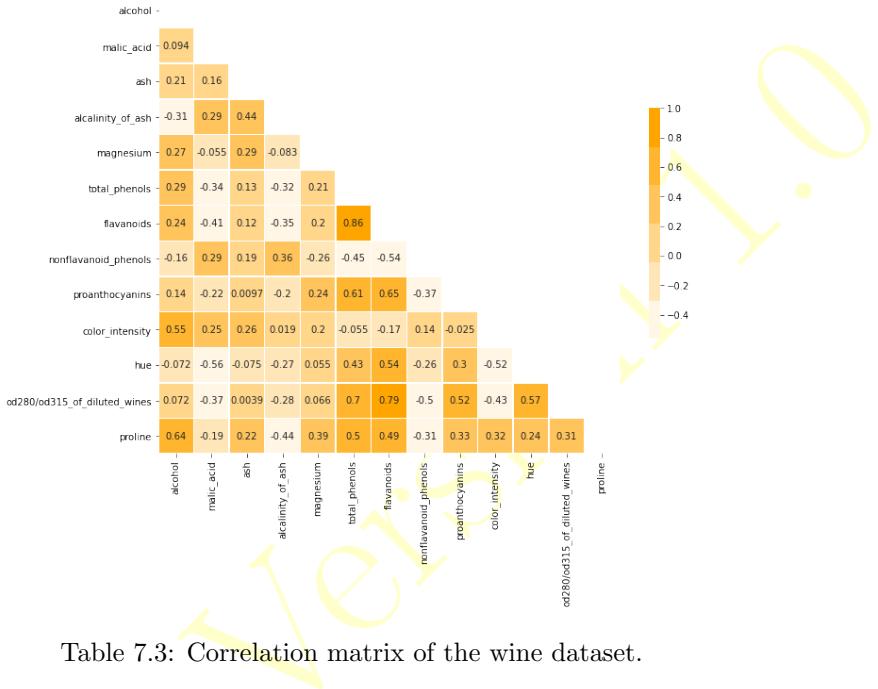


Table 7.3: Correlation matrix of the wine dataset.

Another strategy based on the correlation consists of training an algorithm only with a subset of variables having a minimum value of correlation with the target variable. Figure 7.4 shows the correlation behaviour of the features of the *wine dataset* with the target variable. The plot shows the number of features (y-axis) having a minimum correlation value (x-axis) with the target variable. One may decide to set a threshold of minimum correlation to work with a subset of variables resulting significantly correlated to the target variable (e.g. at least 30% of correlation).<sup>4</sup>

### 7.2.2 Selection by variance

Another strategy is to select a subset of variables whose variance is above a certain level. The idea is the following: if a feature has a low variance, it does not add too much information to the dataset. All the features with variance equal to zero should be removed since they do not add any information to

<sup>3</sup>The source code of Figure 7.3 is available [here](#).

<sup>4</sup>The source code of Figure 7.4 is available [here](#).

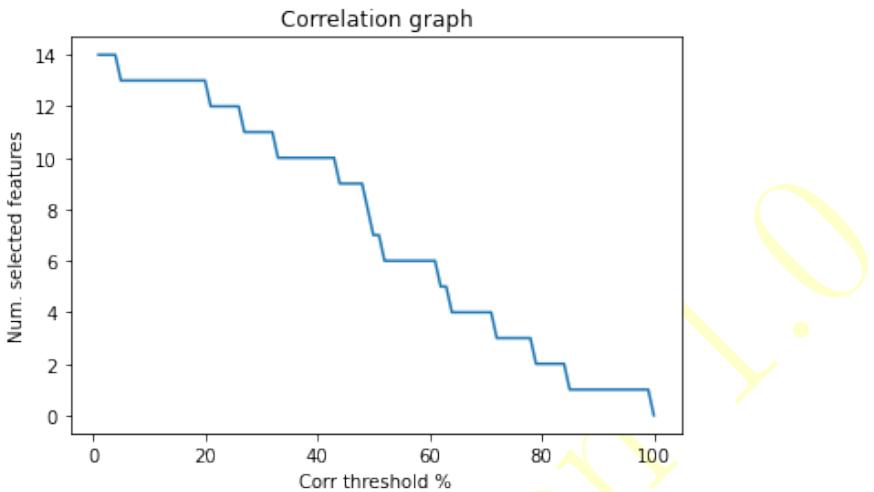


Table 7.4: Number of features with a minimum correlation threshold with the target variable (wine dataset).

the dataset. Figure 7.5 illustrates the variance of the features of the *wine dataset*. The majority of the features has a variance lower than 60%.<sup>5</sup>

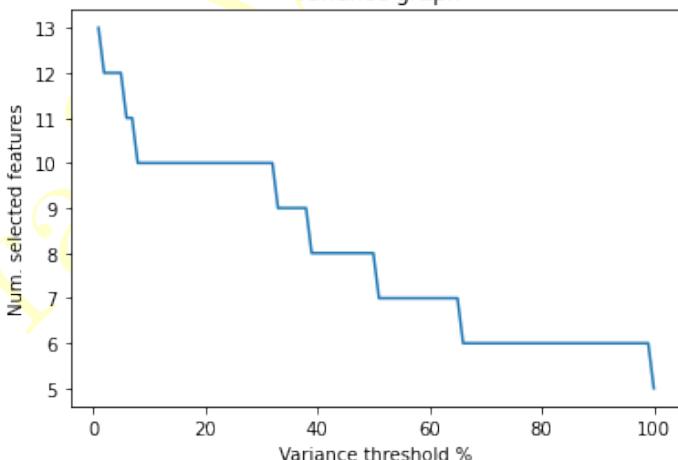


Table 7.5: Number of features above a minimum variance threshold (wine dataset).

<sup>5</sup>The source code of Figure 7.5 is available [here](#).

### 7.2.3 Selection by Lasso coefficients

Lasso regression (further details in Section 9.3.2) is a prediction model that extends the linear regression which embeds a feature selection strategy. It automatically identifies a coefficient for each feature, shrinking the feature according to its relative importance. The coefficients of a Lasso regression can be used to select only the important features. Figure 7.6 shows the graph with the feature coefficients of the *wine dataset* (on the y-axis) depending on the tuning hyperparameter  $\alpha$  of the Lasso on the x-axis and the value of the coefficients.<sup>6</sup>

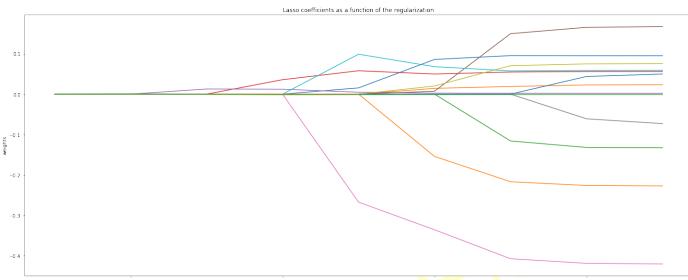


Table 7.6: Lasso shrinkage coefficients graph depending on the value of the hyperparameter  $\alpha$  (wine dataset).

The graph shows that many coefficients are kept to zero up to some values of  $\alpha$ . Features can be selected using Lasso identifying a minimum threshold on the value of the coefficients, pinpointing relative importance of the underlying attributes. Figure 7.7 illustrated the number of features selected from the *wine dataset* by using different thresholds on the value of the coefficients.<sup>7</sup>

### 7.2.4 Selection by using a decision tree

A decision tree trains a predicting model branching on the value of a variable defining a tree structure (further details in Section 11.2.1). The most important features can be selected, evaluating which of them appears the most as a branching variable. Identifying a minimum threshold on the number of times a feature appears as a branching variable works as a feature selection strategy.

<sup>6</sup>The source code of Figure 7.6 is available [here](#).

<sup>7</sup>The source code of Figure 7.7 is available [here](#).

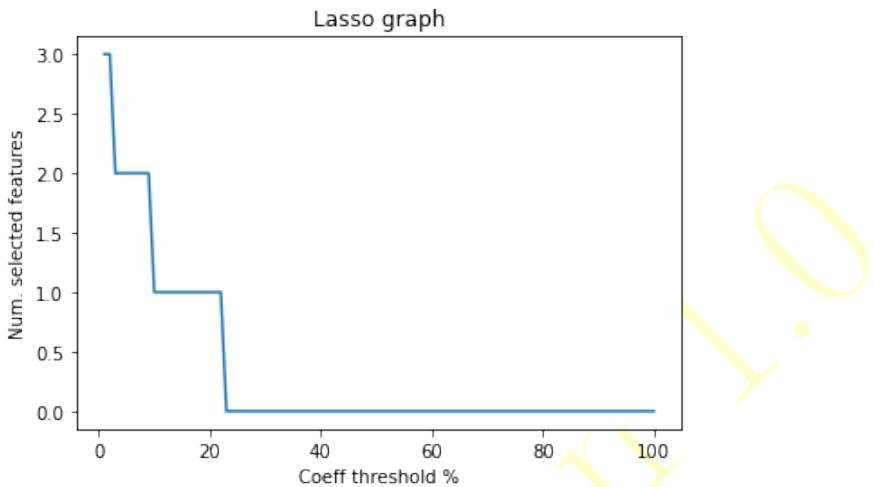


Table 7.7: Number of features above a minimum lasso coefficient threshold (wine dataset).

### 7.2.5 Forward Stepwise selection

Forward stepwise selection uses the principles of the linear regression (see section 9.2) to identify the most relevant features. There are other algorithms based on the linear regression to select features (e.g. best subset selection, forward stagewise selection) but forward stepwise has been chosen since it can be efficiently implemented compared to the others. The residual sum-of-squares (RSS) of linear regression is always minimised when the number of predictors is maximum. Nevertheless, this does not imply a low bias and variance (affecting the prediction error). For this reason, we want to select a subset of the initial features to reduce the probability of overfitting. Forward stepwise selection works as follows.

---

#### Algorithm 3: Forward Stepwise algorithm

---

```

Identify the intercept of the linear regression
for i=1:number of features of the dataset do
    Select one feature improving the most the fit of the model
    Add the feature to the model
end

```

---

Figure 7.8 shows the outcome of the forward stepwise selection of the wine dataset. Increasing the number of features, the RSS decreases and the  $r^2$  of the model increases. Besides, a relatively small number of feature (i.e. the first five features) is enough to fit the linear model obtaining a relatively

small error.<sup>8</sup>

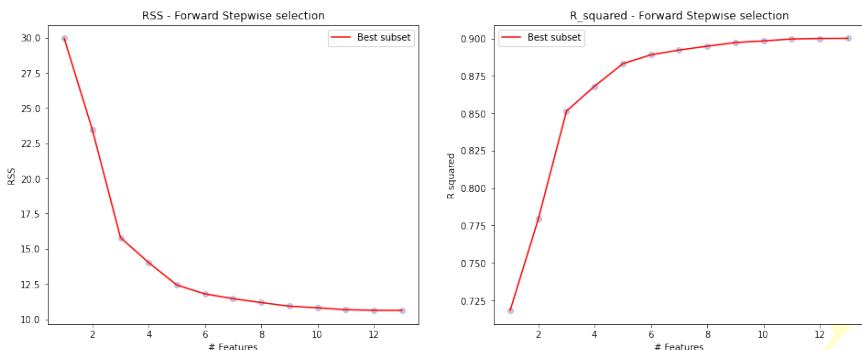


Table 7.8: Forward stepwise selection graph applied to the wine dataset.

## Further reading

Supplementary reading materials can be found in [1].

## Bibliography

- [1] I. D. Dinov, *Data Science and Predictive Analytics*. 2018.

<sup>8</sup>The source code of Figure 7.8 is available [here](#).

Draft Version 1.0

# 8

## Unsupervised learning

Ἐν οἷςα ὅτι οὐδὲν οἶδα.

Socrates

Sometimes we observe phenomena without a precise idea in mind of what we want to investigate. Phenomena may retain information and hidden data patterns that we have never considered. Unsupervised learning use algorithms to uncover these patterns and create knowledge from the data.<sup>1</sup>

### 8.1 Association rules

Association rules are a powerful data mining set of algorithm aiming at investigating patterns in the co-occurrences of items in a series of independent observations. Usually, they are used to mine a commercial database investigating patterns in the buyers' attitude to set promotions, discounts or the shelf allocation. The most used algorithm is the *apriori* algorithm which aims at the definition of association rules between the  $p$  features of a dataset  $X$ .

The definition of association rules is based on the evaluation of the following metrics for each association rule:

- Support,  $T(p_1 \rightarrow p_2)$ . It indicates the probability that an observation contains a group of features (e.g.,  $(p_1, p_2)$ );
- Confidence  $C(p_1 \rightarrow p_2) = \frac{T(p_1 \rightarrow p_2)}{T(p_1)}$ , it indicates the probability a feature  $p_2$  is in a transaction containing  $p_1$  (conditional probability).

<sup>1</sup>The package logproj provides methods to deal with unsupervised learning [here](#).

It is calculated as the support of the rule divided by the support of the antecedent;

- Lift  $L(p_1 \rightarrow p_2) = \frac{C(p_1 \rightarrow p_2)}{T(p_2)}$ , defines the increase in the observation of  $p_2$  when  $p_1$  is observed.

The outcome of the *apriori* algorithm is a set of rules  $(p_1 \rightarrow p_2)$  with support and confidence above a predetermined threshold.

## 8.2 Clustering

Association rules produce a list of causal relations between the features. Differently, clustering produces a label for each of the  $N$  observations identifying “homogeneous” groups. Clustering, in fact, involves a set of unsupervised learning algorithms aiming at grouping the observations into subsets such that the observations in the same subset are close to each other.

Clustering algorithms work using proximity matrices, defining the pairwise distance between observations. For this reason, it is necessary to convert qualitative, ordinal and categorical variables such that a measure of distance is defined.

Hard clustering algorithm creates clusters and assigns observations to one of them; on the other side, soft clustering defines a probability for each observation to belong to each cluster. Clustering approaches are divided into:

1. combinatorial algorithms, which directly works on the observed data;
2. mixture models, which makes assumptions on the probability distributions generating the observations.

Combinatorial algorithms (e.g., k-means) are hard-clustering algorithms minimising a loss function describing the distance between the observations. Let  $k = 1, \dots, K$  be the number of clusters and  $k = C(i)$  the assignment of observations  $i$  to the cluster  $k$ . Then:

$$W(C) = \frac{1}{2} \sum_{k=1}^K \sum_{C(i)=k} \sum_{C(i')=k} d(x_i, x_{i'}) \quad (8.1)$$

$$B(C) = \frac{1}{2} \sum_{k=1}^K \sum_{C(i)=k} \sum_{C(i') \neq k} d(x_i, x_{i'}) \quad (8.2)$$

Where  $d(x_i, x_{i'})$  is the distance between data points  $x_i$ , and  $x_{i'}$ .  $W(C)$  defines the distance of points within a cluster, while  $B(C)$  defines the distance of points between different clusters. Combinatorial algorithms aim at maximising  $W(C)$  or minimising  $B(C)$ . These two objectives are exactly the same.

### 8.2.1 K-means

K-means algorithm is used to cluster a set of  $N$  observation with  $p$  features (i.e., placed in a  $R^p$ ). This space is assumed to be Euclidean, and the algorithm produces  $k$  clusters. Algorithm 4 illustrates the procedure to generate the clusters.

---

**Algorithm 4:** K-means algorithm

---

```

 $k =$ number of centroids (clusters)
 $r =$ number of iterations of the algorithm
 $i = 1, \dots, N \in V$  set of points
 $j = 1, \dots, k \in C$  set of centroids
 $D_i \in R^p$  set of coordinates of point i
 $S = \emptyset$ 
for  $l = 1 : r$  do
    | randomly assign  $D_j, j \in C$ 
    |  $t = 0$ 
    |  $converge =$ false
    | while (not converge) do
        |   |  $t = t + 1$ 
        |   |  $z_{i,t} = argmin_{j \in C} [dist(D_j, D_i)]$ 
        |   |  $D_j = \frac{1}{|D_i|} \sum_{z_i=j} D_i$ 
        |   | if ( $z_{i,t} == z_{i,t-1}$ ) then
        |   |   |  $converge =$ true
        |   | end
    | end
    |  $z = \sum_{i=1}^N \sum_{j=1}^k dist[(D_j, D_i)]$ 
    |  $S = S \cup z$ 
end
Select  $\min(z) \in S$ 
```

---

We use the *digits dataset* containing images of a digit to show the power of unsupervised learning techniques. The dataset contains images of digits, from zero to nine with their label. We use unsupervised learning to cluster the observations, and we project the input dataset into two components to visually compare the results of the clustering with the true label. Figure 8.1 illustrates that k-means is able to detect patterns similar to the true labels.<sup>2</sup>

### 8.2.2 Hierarchical clustering

Hierarchical clustering defines clusters based on a proximity metric between the observations. Similarly to Multi-Dimensional scaling (see Section 7.1.3)

---

<sup>2</sup>The source code of Figure 8.1 is available [here](#).

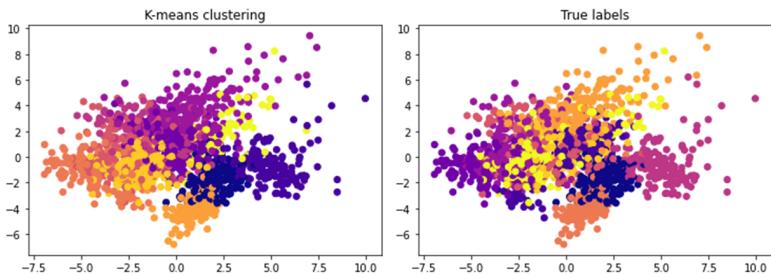


Figure 8.1: Comparison between k-means clustering and true labels of the digits dataset. Different colours identify different labels. There is no specific assignment between colours and labels.

hierarchical clustering does not work with a matrix  $X_{N,P}$  with a number of observations  $N$  and  $P$  features (as the k-means algorithm does). Hierarchical clustering relies on a proximity matrix  $D_{N,N}$  expressing a pairwise distance between each observation (according to a single feature expressing a distance). It is common to work using similarity values  $s_{ij}$  as entries of  $D_{N,N}$ . Once the pairwise distance  $d_{i,j}$  is calculated, the similarity can be calculated as  $s_{i,j} = 1 - \frac{d_{i,j}}{\max_{i,j} d_{i,j}}$ . Table 8.1 illustrates an example of the proximity matrix  $D_{N,N}$ .

	Observation 1	Observation 2	Observation 3	Observation 4
Observation 1	1	0.8	0.6	0.01
Observation 2		1	0.3	0.5
Observation 3			1	0
Observation 4				1

Table 8.1: Example of a similarity matrix.

The matrix in Table 8.1 is symmetric, this is not strictly required, but it can simplify the structure of the data without adding too much bias. If a similarity matrix is not symmetric, it can be converted into a symmetric one by setting  $s_{i,j} = s_{j,i} = \frac{s_{i,j} + s_{j,i}}{2}$ . Once  $D_{N,N}$  is defined, it is possible to apply hierarchical clustering to group the  $N$  observations into clusters. The number of clusters is not defined in advance. Algorithm 5 presents an

algorithm for hierarchical clustering.

---

**Algorithm 5:** Hierarchical clustering algorithm

---

```

 $i = 1, \dots, N \in V$  set of observations
 $s_{i,j}$  similarity between observation  $i$  and  $j$ 
 $S = \emptyset$ 
for  $k \leftarrow 1 : (N - 1)$  do
     $v = \max_{(i,j) \in S} (s_{i,j})$ 
     $(h, l) = \arg(v)$ 
     $S = S \cup (h, l)$ 
    for  $r \leftarrow 1 : m$  do
        if  $CLINK$  then
             $s_{r,h} = \min(s_{r,h}, s_{r,l})$ 
             $s_{h,r} = \min(s_{h,r}, s_{l,r})$ 
        end
        if  $SLINK$  then
             $s_{r,h} = \max(s_{r,h}, s_{r,l})$ 
             $s_{h,r} = \max(s_{h,r}, s_{l,r})$ 
        end
        if  $UPGMA$  then
             $s_{r,h} = \text{mean}(s_{r,h}, s_{r,l})$ 
             $s_{h,r} = \text{mean}(s_{h,r}, s_{l,r})$ 
        end
         $s_{r,l} = -1$ 
         $s_{l,r} = -1$ 
    end
end

```

---

The algorithm iteratively selects two observations and aggregate them into a single cluster until all the observations belong to one big cluster. The value of similarity  $s_{i,j}$  of an observation  $i$  (aggregated with an observation  $k$  at an iteration) and all the others,  $j$  is selected according to the tuning of the algorithm which can consider the minimum, the maximum or the average (complete linkage, single linkage or average linkage) among  $s_{i,j}$  and  $s_{k,j}$ .

Since each observation/cluster is aggregated at a value of similarity, at the end of the procedure, a similarity threshold is selected to identify a number of clusters and the cluster each observation belongs. This procedure can be visually interpreted by a dendrogram which maps the aggregations of the algorithms with a threshold of similarity identifying the clusters. Figure 8.2 illustrates the dendrogram obtained clustering the digits dataset using single, complete and average linkages having a Euclidean distance between the observations.<sup>3</sup>

---

<sup>3</sup>The source code of Figure 8.2 is available [here](#).

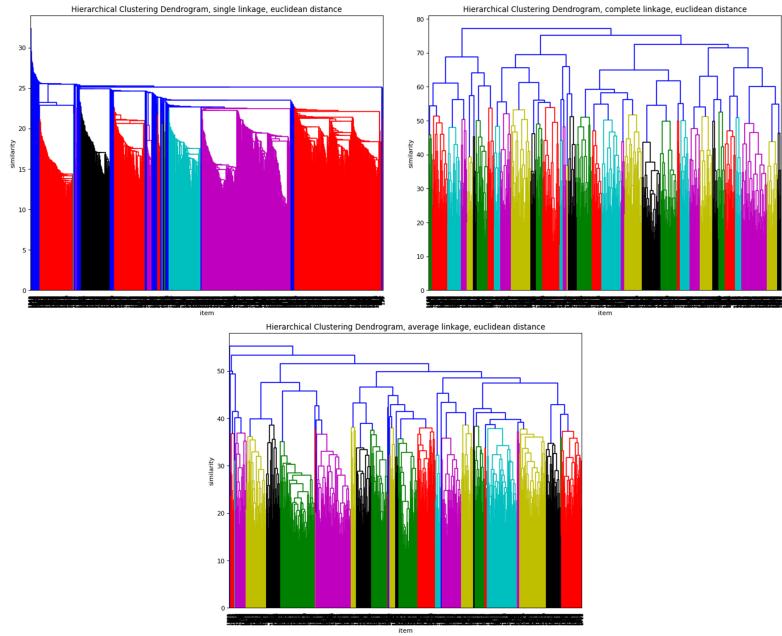


Figure 8.2: Similarity dendrograms of the digits dataset.

Different similarity thresholds identify a different number of clusters. Assuming ten clusters, ad the number of labels of the digits dataset, Figure 8.3 illustrates the comparison between the clusters obtained with the different linkages and the true labels.<sup>4</sup>

### 8.2.3 Mixture models

In general, the observations may be generated by an unknown number  $K$  of PDF with unknown parameters (i.e. mean and variance). Mixture models are soft clustering techniques used to investigate the probability that a point is generated by one of the  $K$  generating PDF. It is called soft because it defines a probability for each point and each distribution, without a direct binary (i.e. true or false) assignment to a cluster. The generating function of a Gaussian mixture model can be seen as:

$$f(x) = \sum_{m=1}^K \alpha_m \phi(x; \mu_m; \Sigma_m) \quad (8.3)$$

Where  $\phi$  is a multidimensional Gaussian PDF with parameters  $\mu_m$  and

---

<sup>4</sup>The source code of Figure 8.3 is available [here](#).

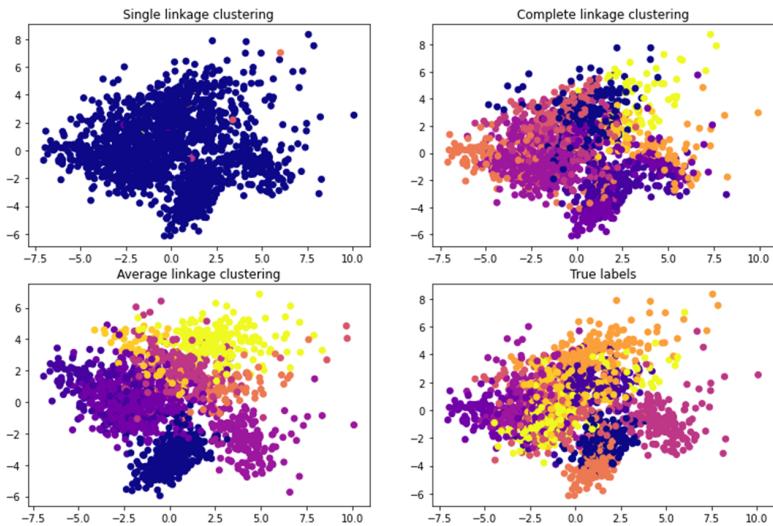


Figure 8.3: Comparison between hierarchical clustering and true labels of the digits dataset. Different colours identify different labels. There is no specific assignment between colours and labels.

$\Sigma_m$ , and  $\alpha_m \in [0, 1]$  is the probability of the  $m$ -th generating function. The problem of fitting a mixture model to data is to define the value of  $\alpha_m$ ,  $\mu_m$ ,  $\Sigma_m$  that best represent the real distribution of the data. This is a likelihood maximisation problem with  $\theta = \alpha_m, \mu_m, \Sigma_m$ . To efficiently get the result the so-called EM-algorithm (Expectation-Maximization) is used. This algorithm can be used when it is difficult to maximise a likelihood but it is made simpler by enlarging the sample using unobserved data. Considering  $K = 2$ , the EM algorithm can be exemplified as follows:

---

**Algorithm 6:** Expectation Maximization (EM) algorithm

---

1. Randomly select  $\mu_a, \sigma_a, \mu_b, \sigma_b$
  2. Calculate the posterior probability  $a_i = \text{Prob}(a|x_i)$  and  $b_i = \text{Prob}(b|x_i)$
  3. Redefine  $\mu_a, \sigma_a, \mu_b, \sigma_b$  as the weighted average of mean and variance of  $x_i$  in  $a$  and  $b$
  4. Repeat from 2. until  $\mu_a, \sigma_a, \mu_b, \sigma_b$  converges
- 

Figure 8.4 illustrates the output of a Gaussian mixture model applied to the digits dataset.<sup>5</sup>

---

<sup>5</sup>The source code of Figure 8.4 is available [here](#).

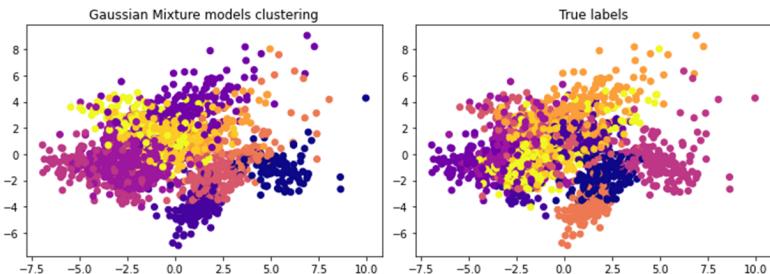


Figure 8.4: Comparison between Gaussian mixture model and true labels of the digits dataset. Different colours identify different labels. There is no specific assignment between colours and labels.

### 8.2.4 Bag of words

A Bag of words is a text-mining method which works as an unsupervised model for strings. It is a frequency analysis for strings of text. Given a dataset composed of  $N$  strings (e.g., a paragraph) the bag of word model counts the number of occurrences of each string. Each string can be interpreted as a feature of the dataset with different relative importance. The words occurring the most retain the highest level of information of the dataset and can be used as predictors (e.g., depending on the content, it is possible to classify an email into spam/not spam).

## Further reading

Supplementary reading materials can be found in [1], [2].

## Bibliography

- [1] C. C. Aggarwal, *Data Mining*. Springer, 2015.
- [2] R. C. Blattberg, B.-D. Kim, and S. A. Neslin, *Database Marketing*. Springer, 2008.

# 9

## Linear Methods for Regression

*The Earth is not flat, but the world behaves linearly, sometimes.*

This chapter, together with chapters, 10, 11, 12, introduces the so-called “supervised learning”. Differently from unsupervised learning, these algorithms train models on data to predict the value of a given feature  $y$ . This section addresses regression models, i.e. models targeting a real number.<sup>1</sup>

### 9.1 Supervised learning

Supervised learning (predictive algorithms) are used to predict the value of an unknown variable  $y$ , from a training set  $X$  of observations where  $y$  is given for each row of  $X$ . This technique is useful when it is necessary to build a prediction model of the future value of  $y$ . If the observations contains only the feature  $y$ , then time series analysis (see Section 6.5) applies. When a number of features  $P$  is available for each observation, together with  $y$ , then an option is to build a supervised learning model.

The dataset of a learning model is composed of (see Figure 9.1):

- A matrix  $X_{N,P-1}$  with  $N$  observations of the  $P - 1$  predictors;
- A vector  $y_{N,1}$  with  $N$  observations of the target variable.

<sup>1</sup>The package logproj provides methods to deal with linear regression [here](#).

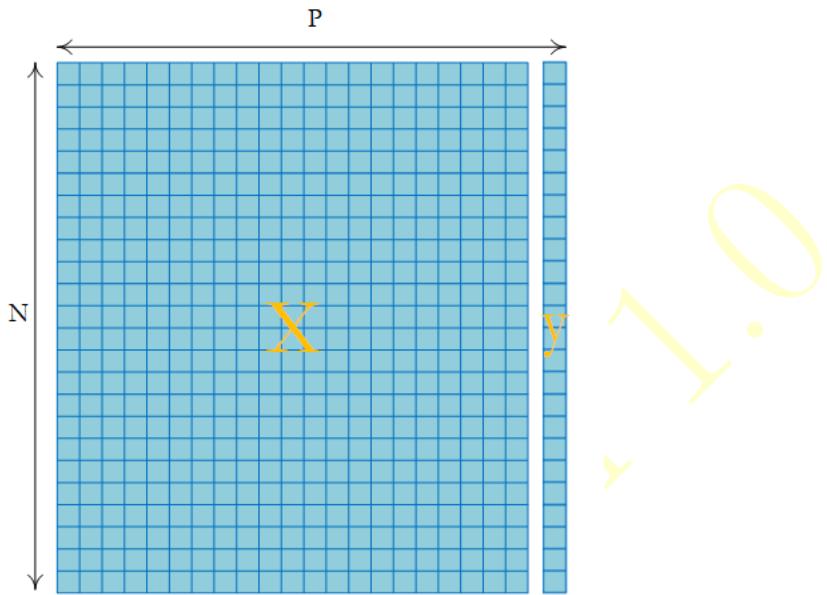


Figure 9.1: Scheme of the input dataset of a predictive model.

Our goal is to train a model to link  $X$  and  $y$  efficiently. This link is the approximation of the joint probability distribution function  $y = f(X)$ . Predictive models are effective when they correctly estimate  $f$ . For validation reasons, the dataset  $X$  is always split into two separate datasets:

1. the training set: used to train the model;
2. the testing set: used to test the performance of the model by measuring its error.

The training set is needed to train the model by setting a number of parameters to maximise the fitting of the function  $f$  to the data. The tuning parameters are specific for each family of models, and their value is set during the training phase. Models may have other parameters (called hyperparameters) whose values are set before the beginning of the training phase.

The testing set is used to compare the predictions obtained by the model with true values selected from the input dataset. If a model succeeded in this testing phase, it is ready for the implementation, i.e. to make predictions based on new data. Figure 9.2 illustrates the dataflow to build a predictive model.

The following chapters illustrate tens of machine learning models. It is necessary to understand how to choose the most proficient in practice. The

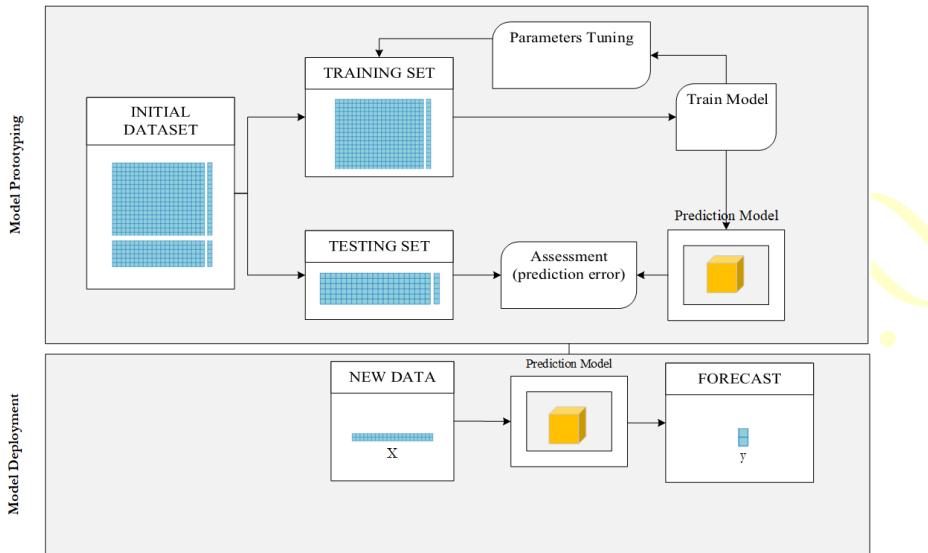


Figure 9.2: Flow of the development and deployment of a prediction model.

main idea is to have an error metric and to choose the model that minimises the most this error metric. The prediction error can be calculated for the training set or the testing set. Given a training set  $T$ , we can define a prediction error on the training set  $\overline{err}$  and a prediction error  $Err$  on the independent testing set as follows.

$$\overline{err} = \frac{1}{N} \sum_{i=1}^N L(y_i, f(x_i)) \quad (9.1)$$

$$Err = E_T [E_{X^0, Y^0} [L(Y^0, \hat{f}(X^0)) | T]] \quad (9.2)$$

Where  $L$  is our error metric, called loss function (e.g., the mean squared error (MSE) or the absolute error). The definition of  $\overline{err}$  and  $Err$  shows that an error metric can always be computed for both the training and the testing set. Unfortunately, the error  $\overline{err}$  measured on the training set is not a good estimate for the error  $Err$  in the testing set. We want to minimise the error  $Err$  on the testing set since it is the best estimate of the error that the model will have while working with new data. Stressing the minimisation of  $\overline{err}$  leads to a phenomenon called *overfitting*; the training error is minimised, while the training error raises.

A model adapts itself to best fit to the training set, but this does not imply the same good fit happens with the testing set. For this reason, it is

not a good idea striving to reduce the error in the training set. In general,  $\overline{err} < Err$  and the training error tends to zero increasing the complexity of the model (e.g., the number of features involved) but this fact does not guarantee good results of the test set.

Model selection involves different metrics to measure the performance of different predictive models in order to choose the best one. For this reason, the in-sample error  $Err_{IN}$  is introduced to describe the error having  $N$  new response values at each of the training points<sup>2</sup>.

$$Err_{IN} = \frac{1}{N} \sum_{i=1}^N E_T \left[ E_{Y^0} \left[ L \left( Y_i^0, \hat{f}(X_i) \right) | T \right] \right] \quad (9.3)$$

A metrics called optimism is defined as  $op = Err_{IN} - \overline{err}$ . We usually consider  $\omega = E_y[op]$  which can be estimated as:

$$\omega = \frac{2}{N} \sum_i cov(\hat{y}_i, y_i) \quad (9.4)$$

The underestimation by  $\overline{err}$  in the true error depends on how much  $y_i$  affects its own prediction.  $\omega$  is a metrics used as a basis for the evaluation of the performance of machine learning algorithms.

### MSE

The most commonly used error metric to select a regression model is the mean squared error. It is calculated as:

$$MSE = \frac{1}{N} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (9.5)$$

A main limitation of the MSE is that it suffers significant prediction errors on the outliers. For this reason, it is possible to introduce quantiles of errors which evaluates the error of a model within a given percentile. The mean absolute percentage error (MAPE) is used at this purpose:

$$MAPE = p \left( \frac{|y_i - \hat{y}_i|}{y_i} \right) \quad (9.6)$$

Where  $p$  is the chosen percentile. In practice, we can evaluate the percentage of estimates that differs from the true value no more than a given percentage (e.g. 10%).

---

<sup>2</sup>This is a sampling error, i.e. a measure of how much the sample chosen to train the model represents the entire population.

### AIC and BIC

We consider  $\widehat{Err}_{IN} = \overline{err} + \hat{\omega}$ . Akaike information criterion (AIC) and Bayesian information criterion (BIC) are two very common metrics used to assess the performance of a model. They are defined as follows.

$$AIC = -\frac{2}{N} E[\loglik] + \frac{2d}{N} \quad (9.7)$$

Where  $\loglik = \sum_{i=1}^N \log(\Pr_{\hat{\theta}} y_i)$ ,  $N$  is the number of samples, and  $d$  defines the number of features. BIC is defined similarly.

$$BIC = -2\loglik + (\log N)d \quad (9.8)$$

Both these metrics can be used to identify the best tuning parameter of a model or to compare different models. The model with the minimum AIC or BIC value should be chosen. To choose among AIC or BIC, it is important to remember that BIC is asymptotically consistent, which is equivalent to say that BIC selects the correct model with a probability approaching 1 as  $N \rightarrow \infty$ . Otherwise, increasing the number of samples AIC tends to choose more complex models. In other words, BIC tends to prefer simpler models than AIC, that chooses more complex models.

### Cross-validation

The metrics proposed in the previous paragraphs allows investigating the reliability of the predictions. Sometimes, when having few data, split into training, and the testing dataset may lead to very small datasets. For this reason, cross-validation (CV) or bootstrapping are used.

The selection of the train-test split of the data may bias these measures of the error. The CV is used to evaluate the extra-sample error  $Err = E[L(Y, \hat{f}(X))]$ .  $K$ -fold cross-validation splits the set into  $K$  subsets and performs all the different permutations choosing one of them as a validation set and using the others to train the algorithms. Figure 9.3 shows an example of 5-folds CV.

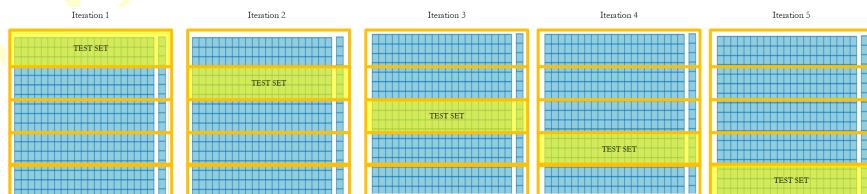


Figure 9.3: Schema of a 5-folds cross-validation.

The CV produces an estimate of  $Err$  as:

$$CV(\hat{f}) = \frac{1}{N} \sum_{i=1}^N L(y_i, \hat{f}^{-k(i)}(x_i)) \quad (9.9)$$

Where  $\hat{f}^{-k(i)}$  is the function fitted without the  $k$ -th fold. A CV can also be used to choose the value of a tuning hyperparameter  $\alpha$  of a model.

$$CV(\hat{f}, \alpha) = \frac{1}{N} \sum_{i=1}^N L(y_i, \hat{f}^{-k(i)}(x_i, \alpha)) \quad (9.10)$$

Besides, CV can be used together with bootstrap methods (see Section 6.2.4). The idea of the bootstrap is to estimate the value of the loss function. Bootstrapping samples from the empirical distribution of the data (i.e. the input dataset, since we do not know the real distribution). Bootstrap samples with replacement since a point can be added to the sampled distribution multiple times to reflect the behaviour of the empirical distribution (when sampling without replacement, we have a jackknife sampling). The error  $Err$  can be estimated as:

$$\widehat{Err} = \frac{1}{N} \sum_{i=1}^N \frac{1}{|C^{-1}|} \sum_{b \in C^{-1}} L(y_i, \hat{f}^{*b}(x_i)) \quad (9.11)$$

Where  $C^{-1}$  is the set of the bootstrap samples  $b$  that does not contain observation  $i$ .

### Hyperparameters tuning

While the training phase set the values of the parameters of the model, there is no precise way to set the hyperparameters of the model. Nevertheless, hyperparameters deeply affect the prediction performance of a model. This can be done by several iterations, trying different hyperparameters values for each model. There are different strategies to tune the model<sup>3</sup>. Two strategies are:

- Grid search: i.e. testing all the parameters of a given set, evaluate the performance of the model and choose the best one.
- Random search: random select a subsample of the grid and select the best hyperparameter among this subset.

Smart algorithms (e.g. gradient-based) exist to identify the best direction to search good values of a hyperparameter, but they are usually time-consuming and affect the total training time of the model significantly.

---

<sup>3</sup>The package `logproj` provides grid search methods to train model, identifying the best hyperparameter [here](#).

## 9.2 Linear regression (OLS)

Linear regression is a predictive model assuming a linear relationship between the input  $X$  and the output  $y$ . Let assume  $X_{N,P}$  being the input matrix of  $P$  features and  $N$  observation, we are interested in predicting the value of the output vector  $y_{N,1}$  using a linear relationship. In other terms, the linear regression works in a  $P + 1$ -dimensional space aiming at predicting the value of  $y \in \mathbb{R}$  as a linear combination of the variables  $x \in \mathbb{R}^P$ . In practice, we are looking for the function  $f$ .

$$f(X) = \beta_0 + \sum_{j=1}^P X_j \beta_j \quad (9.12)$$

Since  $X$  is given, our problem is to define a vector  $\beta$  of scalar values such that the residual sum of squares (RSS) between the values of  $y$  and  $\hat{y} = f(X)$  is minimized.

$$\begin{aligned} RSS(\beta) &= \sum_{i=1}^N (y_i - f(x_i))^2 = \\ &= \sum_{i=1}^N \left( y_i - \beta_0 - \sum_j x_{ij} \beta_j \right)^2 = \end{aligned} \quad (9.13)$$

By the definition, the sum of squares in matrix notation is the product of the transpose of a vector with the vector itself:  $A^2 = A^T A$ ; for this reason,

$$\begin{aligned} RSS(\beta) &= y^T y - y^T X \beta - \beta^T X^T y + \beta^T X^T (X \beta) = \\ &= y^T y - y^T X \beta - (X \beta)^T y + (X \beta)^T (X \beta) = \end{aligned} \quad (9.14)$$

By the definition,  $(AB)^T = B^T A^T$ ; then,

$$RSS(\beta) = y^T y - y^T X \beta - \beta^T X^T y + \beta^T X^T (X \beta) = \quad (9.15)$$

Considering that  $y_{1N}^T X_{NP} \beta_{P1}$  is a scalar number as well as  $\beta_{1P}^T X_{PN}^T y_{N1}$ ,

$$RSS(\beta) = y^T y - 2\beta^T X^T y + \beta^T X^T (X \beta) \quad (9.16)$$

The partial derivatives with respect to each  $\beta_i (i = 1, \dots, P)$  are considered. They are all set equal to 0 to find a minimum of  $RSS(\beta)$ . Let define a  $P \times 1$  vector  $e_i$  with 1 in the  $i$ -th position and 0 elsewhere.

$$\begin{aligned} \frac{\partial RSS(\beta)}{\partial \beta_i} &= -2e_i^T X^T y + e_i^T X^T X \beta + \beta^T X^T X e_i^T \\ &= -2e_i^T X^T y + 2e_i^T X^T X \beta \end{aligned} \quad (9.17)$$

It is a good idea learning for a minimum since the second derivative with respect to  $\beta$  is as follows.

$$\frac{\partial^2 \text{RSS}(\beta)}{\partial \beta} = 2X^T X \quad (9.18)$$

We can assume, by definition, that  $X^T X$  is always positive if  $X$  has full column rank (i.e., all its columns are linearly independent). In general, this is not true, but it is always possible to apply the PCA (see section 7.1.2) to obtain an  $X$  with full column rank. Given this hypothesis,  $\frac{\partial \text{RSS}(\beta)}{\partial \beta}$  is set equal to 0 (for all the values of  $i$ ) looking for a minimum.

$$\begin{aligned} -2e_i^T X^T y + 2e_i^T X^T X \beta &= X^T (y - X\beta) = 0 \\ X^T X \beta &= X^T y \\ \hat{\beta} &= (X^T X)^{-1} X^T y \end{aligned} \quad (9.19)$$

The equation (9.19) defines the value of  $\hat{\beta}$  which best fits the training data.

### 9.2.1 Geometrical representation of the linear regression

The linear nature of this model allows us to interpret the results of the previous paragraph geometrically. Equation (9.19) proves that:

$$\hat{y} = X\hat{\beta} = (X^T X)^{-1} X^T y \quad (9.20)$$

The value of  $\hat{\beta}$  is found such that  $X^T (y - X\hat{\beta}) = X^T (y - \hat{y}) = 0$ . This fact is due to the result of the derivative of the RSS ( $\beta$ ) = 0 but, by the definition of orthogonal vector, it implies the vector  $y - \hat{y}$  is orthogonal to the subspace generated by the  $P$  columns of  $X$  (remember of the hypothesis of full column rank of  $X$ ). In practice  $(X^T X)^{-1} X^T = H$  is the function generating  $\hat{y}$  as the orthogonal projection of  $y$  onto the subspace of  $\mathbb{R}^P$  generated by the  $P$  columns of  $X$  (see Figure 9.4).

$$\hat{y} = (X^T X)^{-1} X^T y = Hy \quad (9.21)$$

It is possible to take a step further considering the univariate (i.e., single variable) linear regression.

$$\begin{aligned} Y &= X\beta + \epsilon \\ \hat{\beta} &= \frac{\sum_{i=1}^N x_i y_i}{\sum_{i=1}^N x_i^2} \end{aligned} \quad (9.22)$$

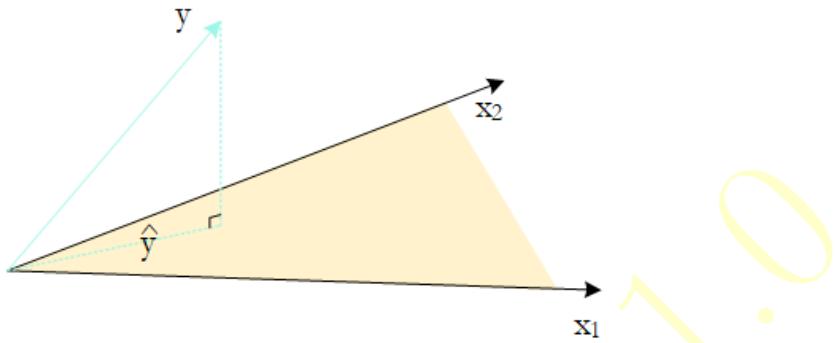


Figure 9.4: Linear regression as a projection of  $y$  on the space generated by  $X$ .

With residuals:  $r_i = y_i - x_i \hat{\beta}$ . These formulae can be written as scalar products<sup>4</sup> with vector notation.

$$\begin{aligned}\hat{\beta} &= \frac{\langle x, y \rangle}{\langle x, x \rangle} \\ r &= y - x \hat{\beta}\end{aligned}\tag{9.23}$$

For this reason, *compute a linear regression of  $b$  on  $a$*  means *orthogonalize  $b$  on  $a$*  by:

1. Producing the coefficient  $\hat{\gamma} = \frac{\langle a, b \rangle}{\langle a, a \rangle}$ ;
2. Producing the residuals  $z = b - \hat{\gamma}a$ .

When dealing with multivariate linear regression, it is always possible to use this approach, applying regression by successive orthogonalization (see Algorithm 7).

---

**Algorithm 7:** Multivariate linear regression

---

1. Set  $z_0 = x_0 = 1$
  2. **for**  $j = 1 : p$  **do**

Regress  $x_j$  on  $z_0, z_1, \dots, z_{j-1}$  to produce  $\hat{\gamma}_{l,j} = \frac{\langle z_l, x_j \rangle}{\langle z_l, z_l \rangle}$  with  
 $l = 0, \dots, j-1$  and  $z_l = x_j - \sum_{k=0}^{j-1} \hat{\gamma}_{l,j} z_k$

3. Regress  $y$  on the residual  $z_p$  to get  $\hat{\beta}_p$ .
  - end
- 

<sup>4</sup>  $\langle x, y \rangle = \sum_i^N x_i y_i = x^T y$

It is clear that we are writing  $x_j$  as a linear combination of  $z_k$  (with  $k < j$ ) where each  $z_k$  is the additive contribution of the  $j$ -th parameter. Theoretically,  $z_k$  are orthogonal. In case a  $x_j$  is highly correlated with any of the  $z_k$  (with  $k < j$ ) the additive contribution of the residual vector  $z_k$  will be close to zero (i.e. the information given by  $j$ -th parameter is already described by the previous  $z_k$  with  $k < j$ ).

There is still an open question: to identify the confidence interval of the linear coefficients  $\beta_j$ . To answer this question, the observations  $y_i$  are assumed to be uncorrelated with constant variance  $\sigma^2$ . In addition, the deviation of  $y$  around its mean is assumed being additive and Gaussian (i.e. additive white gaussian noise). These hypotheses allow applying some statistical test to check which of the input parameters  $p$  is significant for the prediction model.

The variance-covariance matrix<sup>5</sup> is obtained as:

$$\text{Var}(\hat{\beta}) = (X^T X)^{-1} \sigma^2 \quad (9.24)$$

The value of the variance  $\sigma^2$  can be estimated by an unbiased estimator (see section 6.2.1) as follows.

$$\hat{\sigma}^2 = \frac{1}{N - p - 1} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (9.25)$$

Given the these hypotheses,  $\beta$  is distributed as a multivariate normal distribution.

$$\hat{\beta} \sim N(\beta, (X^T X)^{-1} \sigma^2) \quad (9.26)$$

The variance can be described as a  $\chi^2$  distribution with  $N - p - 1$  degrees of freedom  $\sigma^2 \chi^2_{N-p-1}$ . To test the hypothesis  $H_0$  that a coefficient  $\beta_j = 0$  the  $Z$ -score of its coefficient is calculated as follows.

$$z_j = \frac{\hat{\beta}_j}{\hat{\sigma} \sqrt{v_j}} \quad (9.27)$$

With  $v_j$  the  $j$ -th diagonal element of  $(X^T X)^{-1}$ .  $z_j$  is distributed as  $t_{N-p-1}$  and the  $t$ -test is used to assess the null hypothesis  $H_0 : \beta_j = 0$ . When the number of samples increases, a  $Z$ -test can be used as well. A large (absolute) values of  $z_j$  (connected to low  $p$ -values) suggest rejecting  $H_0$  i.e., the  $\beta_j$  coefficient is relevant in the prediction model.

---

<sup>5</sup>The variance-covariance matrix is the generalization of the concept of covariance applied to a space with  $n$  variables.

To simultaneously compare the effect of groups of input parameters, the F-test is used. The value of RSS is calculated for each group of parameters (group 0 and group 1).

$$F = \frac{\left( \frac{RSS_0 - RSS_1}{p_0 - p_1} \right)}{\frac{RSS_1}{N - p_1 - 1}} \quad (9.28)$$

## 9.3 Shrinkage methods

As demonstrated by the Gauss-Markov theorem (see 6.2.1), the OLS provides the estimator of  $\beta$  with the smallest variance among all linear unbiased estimates. Nevertheless, this does not imply the lowest prediction error at all; especially while fitting a few data points having a high (e.g. more than 10) number of features. These characteristics lead to a high risk of overfitting. The shrinkage methods are introduced to avoid overfitting. The main idea of shrinking methods is adding some bias in the predictive model (i.e. to reduce the learning accuracy on the training set) to reduce the prediction error in the testing set.

### 9.3.1 Ridge regression (L2-regularisation)

Ridge regression adds some bias in the prediction model shrinking the regression coefficients adding a penalty on their value.

$$\hat{\beta}_{ridge} = \operatorname{argmin}_{\beta} \left\{ \sum_{i=1}^N \left( y_i - \beta_0 - \sum_{j=1}^N x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^P \beta_j^2 \right\} \quad (9.29)$$

Predictions with ridge regressions are less sensitive to variations in the independent variables compared to simple linear regression. Increasing the value of  $\lambda$ , the values of  $\hat{\beta}$  tend asymptotically to 0 (i.e. a constant line in  $\mathbb{R}^2$ ). Since the ridge coefficients are not equivariant, it is necessary to standardise the inputs before applying the ridge regression (see Figure 9.5).<sup>6</sup> In addition, it is better to centre the input (as already seen for the PCA in 7.1.2) by setting  $x_{ij} = x_{ij} - \bar{x}_j$ , and apply ridge regression without intercept.

$$\beta_0 = \bar{y} = \frac{1}{N} \sum_{i=1}^N y_i \quad (9.30)$$

---

<sup>6</sup>The source code of Figure 9.5 is available [here](#).

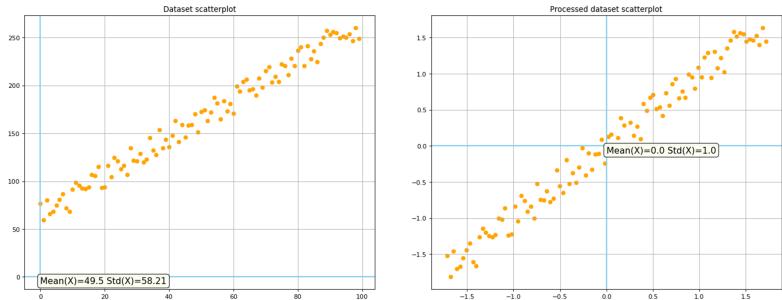


Figure 9.5: Comparison between the original dataset and the centered and scaled one.

By switching the equation (9.29) to matrix form we have:

$$RSS(\lambda) = (y - X\beta)^T (y - X\beta) + \lambda\beta^T\beta \quad (9.31)$$

The values of the ridge coefficients are calculated as follows.

$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{-1} X^T y \quad (9.32)$$

Compared to the  $\hat{\beta}$  coefficients of the linear regression, the ridge coefficients add  $\lambda$  to the diagonal of  $X^T X$ .  $I$  is a  $P \times P$  identity matrix. Using the singular value decomposition (see section 7.1.1) it is possible to express the least squared fitted vector and the solution of the ridge regression. In practice,  $U$  spans the columns space of  $X$ , while  $V$  spans the rows space of  $X$  and  $D$  is a diagonal matrix.

$$X\hat{\beta}^{ls} = UU^T y \quad (9.33)$$

$$X\hat{\beta}^{ridge} = UD(D^2 + \lambda I)^{-1} DU^T y \quad (9.34)$$

In practice, it can be proved that the matrix  $X^T X$  (which is similar to the sample covariance matrix  $S = \frac{1}{N} X^T X$ ) has been written using singular value decomposition matrix.

$$X^T X = VD^2V^T \quad (9.35)$$

As already introduced in section 7.1.1, the eigenvector  $V$  describes the directions of the principal components of  $X$ . In practice, ridge regression shrinks the most the predictors having a small variance protecting against a potentially high variance estimated in the short directions (coefficients close to zero). The value of  $\hat{\beta}$  and the goodness of fit  $r^2$  changes with different values of  $\lambda$ . Cross-validation can be used to identify the best value for  $\lambda$ .

### 9.3.2 Lasso regression (L1-regularisation)

Lasso regression works similarly to Ridge regression, but it considers the minimisation of a different penalty function.

$$\hat{\beta}_{lasso} = \operatorname{argmin}_{\beta} \left\{ \frac{1}{2} \sum_{i=1}^N \left( y_i - \beta_0 - \sum_{j=1}^N x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^p |\beta_j| \right\} \quad (9.36)$$

Differently from ridge regression, the value of  $\hat{\beta}_{ridge}$  cannot be computed directly since its computation is non-linear. Anyway, it can be efficiently computed through efficient algorithms to get a solution in a short time.

The lambda of a lasso regression can be determined using CV similarly to the lambda of ridge regression. Lasso regression penalty contains the  $\hat{\beta}$  as well as ridge regression, but they shrink parameters differently.

Lasso regression can shrink a coefficient slope to 0. Increasing lambda, the bad-prediction parameters can go to zero (and the linked features are excluded from the model). Figure 9.6 shows the effect of L1 and L2 regularisation on a 3-dimensional dataset.

### 9.3.3 Elastic-net regression

When the number of parameters increases dramatically, elastic-net regression results adequate since it mixes the power of the Ridge and the Lasso. It is useful when a correlation between features exist since lasso can discard useless features, and ridge shrinks the correlated features together.

With elastic net regression, the parameters associated with the correlated variables are shrunk or removed all at once. The elastic-net penalty has the formula

$$\lambda \sum_{j=1}^p (\alpha \beta_j^2 + (1 - \alpha) |\beta_j|) \quad (9.37)$$

### 9.3.4 Least angle regression

This shrinkage method works as a forward stepwise regression (see section 3) but it only enters “as much” of a predictor as it deserves. Its algorithm starts from a standardisation of the predictor. At each stage of the algorithm, it finds the predictor most correlated with the residuals, and it moves its  $\hat{\beta}_j$  from 0 to its least-square coefficient until some other predictor has as much correlation as  $j$ . It continues until all the  $j$  predictors have entered.

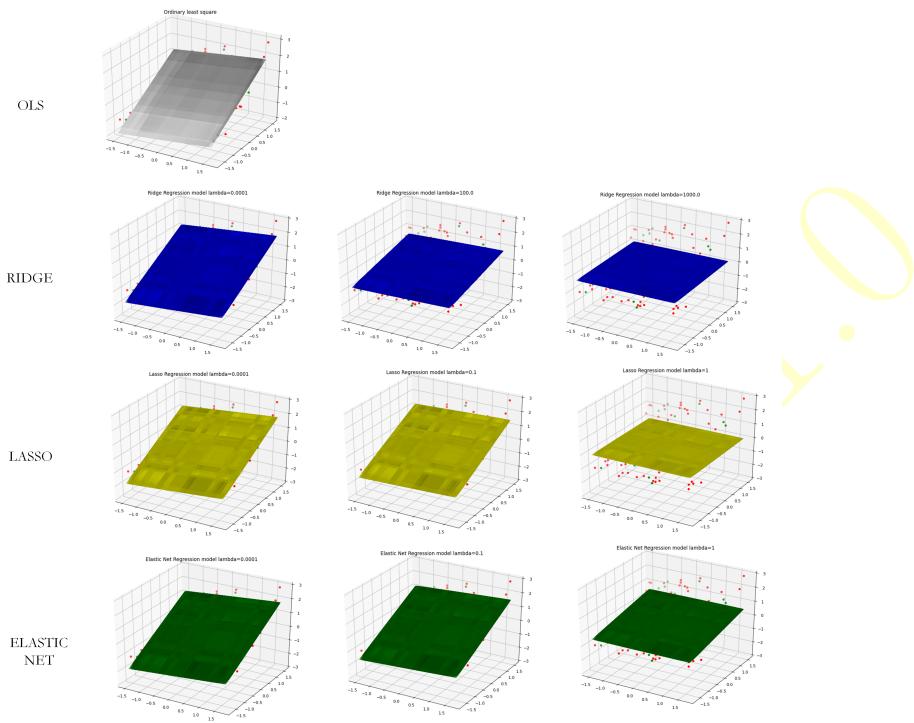


Figure 9.6: Effects of the regularisation algorithms on the linear regression.

## 9.4 Derived input methods

When the input is highly correlated, it could result convenient to preprocess the input first and then to apply a linear regression model

### Principal component regression

This method applies the PCA first, to define a subset of  $M < p$  orthogonal predictors, and then it performs the linear regression. It is important to standardise the input before applying PCA since it depends on its scaling. This procedure is similar to the Ridge regression, but it works discretely (it provides entire predictors) while ridge regression shrinks each coefficient.

### Partial Least Squares

Partial Least square works similarly to the principal component regression. Nevertheless, while the principal component regression is based on the PCA and it gives higher importance to the directions having a higher variance,

the partial least square seeks for the directions having both high variance and high correlation with the response.

### Transformation for linearity

The assumption of a linear model (i.e., the function  $f$  is linear in  $y = f(X)$ ) often produces approximations on real predictions. It is always possible to transform the input data  $X$  using a function  $h$  before applying a linear model. In particular, the prediction model will be in the form:

$$y = f(X) = \sum_{m=1}^M \beta_m h_m(X) \quad (9.38)$$

Common functions for  $h_m$  are:

- $h_m(X) = X_m$ , no transformation on the initial data;
- $h_m(X) = X_j^2$  or  $X_j X_k$ ;
- $h_m(X) = \log(X_j)$  or  $\sqrt{X_j}$  ;
- $h_m(X) = I(L_m \leq X_m < U_m)$ , this case applies spline function to get a local polynomial approximation of the initial data.

Finding a function  $h$  that linearise the relationship between  $X$  and  $y$  extends the field of application of the linear models.

### Further reading

Supplementary reading materials can be found in [1], [2].

### Bibliography

- [1] S. S. Skiena, *The data science design manual*. 2017.
- [2] L. Igual and S. Seguí, *Introduction to Data Science: A Python Approach to Concepts, Techniques and Applications*. 2017.

Draft Version 1.0

# 10

## Linear Methods for Classification

People label people; algorithms too.

Differently from regression models, classification models aim at predicting a categorical target variable  $k \in K$ .<sup>1</sup> The input space where the predictors  $X$  lies can always be divided into a set of regions divided by decision boundaries according to the values of  $G$ . A statistical model with a discrete target variables aims at describing a function  $F$  as:

$$\Pr(G = k) = F(x^T \beta) \quad (10.1)$$

Where  $G$  is a discrete value assuming values  $k \in K$  and  $F$  is a function of an input vector  $x$  and a vector of unknown parameters  $\beta$ . In general, the OLS model is not a good model since its domain is continuous. In this case, it is necessary to provide a different probability model with the following characteristics:

$$f(X) = \begin{cases} \lim_{x^T \beta \rightarrow -\infty} F(x^T \beta) = 0 \\ \lim_{x^T \beta \rightarrow +\infty} F(x^T \beta) = 1 \end{cases} \quad (10.2)$$

A cumulative distribution function (CDF) has both these features and it is chosen as a model, for this reason. Two very common CDF functions are used.

1. Probit (probability unit) function (CDF of the gaussian distribution

<sup>1</sup>The package logproj provides methods to deal with linear classification [here](#).

function)

$$\Pr(G = 1) = F_p(x^T \beta) = \int_{-\infty}^{x^T \beta / \sigma} \frac{1}{2\pi} e^{-\frac{z^2}{2}} dz \quad (10.3)$$

2. Logit (logistic unit) function (CDF of the logistic distribution function)

$$\Pr(G = 1) = F_l(x^T \beta) = \int_{-\infty}^{x^T \beta / \sigma} \frac{e^z}{(1 + e^z)^2} dz = \frac{e^{x^T \beta}}{1 + e^{x^T \beta}} \quad (10.4)$$

In particular, the logit function can be linearized to build a linear classification model (see section 10.3). To compare two classes (i.e., to define a boundary between them) the odds ratio  $\frac{p}{1-p}$  are considered. A decision boundary is defined where an odd ratio is equal to zero. In general, the  $\Pr(G = k|X = x)$  is defined according to different statistical distributions. Let assume:

- $f_{k(x)}$  is the class density of  $X$  in class  $G = k$ ;
- $\pi_k$  is the prior probability of class  $k$  (with  $\sum_{k=1}^K \pi_k = 1$ ).

By applying the Bayes theorem (see equation (6.62)), we have:

$$\Pr(G = k|X = x) = \frac{f_k(x) \pi_k}{\sum_{l=1}^K f_l(x) \pi_l} \quad (10.5)$$

Each classification model assumes or uses a different way to define  $f_{k(x)}$ . When it exists a monotone transformation of  $\Pr(G = k|X = x)$  which make it linear in  $X$ , then the prediction model is linear. All these models use the  $\Pr(G = k|X = x)$  to define discriminant functions  $\delta_k(x)$  such that an entry  $x$  is classified by the model into one of the  $k$  class maximising  $\delta_k(x)$ .

In general, classification aims at dividing the hyperplanes into a number of subspaces equal to the classes of the target variable. This can be done by minimising the distance of misclassified points to the decision boundaries. A simple algorithm explaining this logic is Rosenblatt's Perceptron Algorithm. A perceptron is a classifier which computes a linear combination of the input feature and returns the sign. If  $y_i = -1$ , then the point  $i$  is misclassified, otherwise  $y_i = 1$ . The objective function of the algorithm is to minimise:

$$\min(D(\beta, \beta_0)) = - \sum_{i \in M} y_i (x_i^T \beta + \beta_0) \quad (10.6)$$

Where  $M$  is the set of misclassified points. The gradient of this function is:

$$\frac{\partial D(\beta, \beta_0)}{\partial \beta} = - \sum_{i \in M} y_i x_i \quad (10.7)$$

$$\frac{\partial D(\beta, \beta_0)}{\partial \beta_0} = - \sum_{i \in M} y_i \quad (10.8)$$

The algorithm visits in a sequence all the misclassified points and updates the parameter  $\beta$ .

$$\begin{pmatrix} \beta \\ \beta_0 \end{pmatrix} \leftarrow \begin{pmatrix} \beta \\ \beta_0 \end{pmatrix} + \rho \begin{pmatrix} y_i x_i \\ y_i \end{pmatrix} \quad (10.9)$$

$\rho$  is called “learning rate”, and it can be proved that this algorithm converges when the classes are linearly separable.

## 10.1 Model selection

Similarly to the error metrics evaluating regression models, classification models have specific error metrics to compare the outcome of each model and choose the one with the best predictive performance.

### 10.1.1 Accuracy

The accuracy is the simplest indicator since it measures the number of good predictions over the total number of predictions. We consider the case of binary classification (i.e., two classes '1', and '-1') and the number of the:

- true positives  $TP$ : the items with true label '1', classified correctly;
- true negatives  $TN$ , the items with true label '-1', classified correctly;
- false positives  $FP$ , the items with true label '-1', classified incorrectly;
- false negatives  $FN$ , the items with true label '1', classified incorrectly.

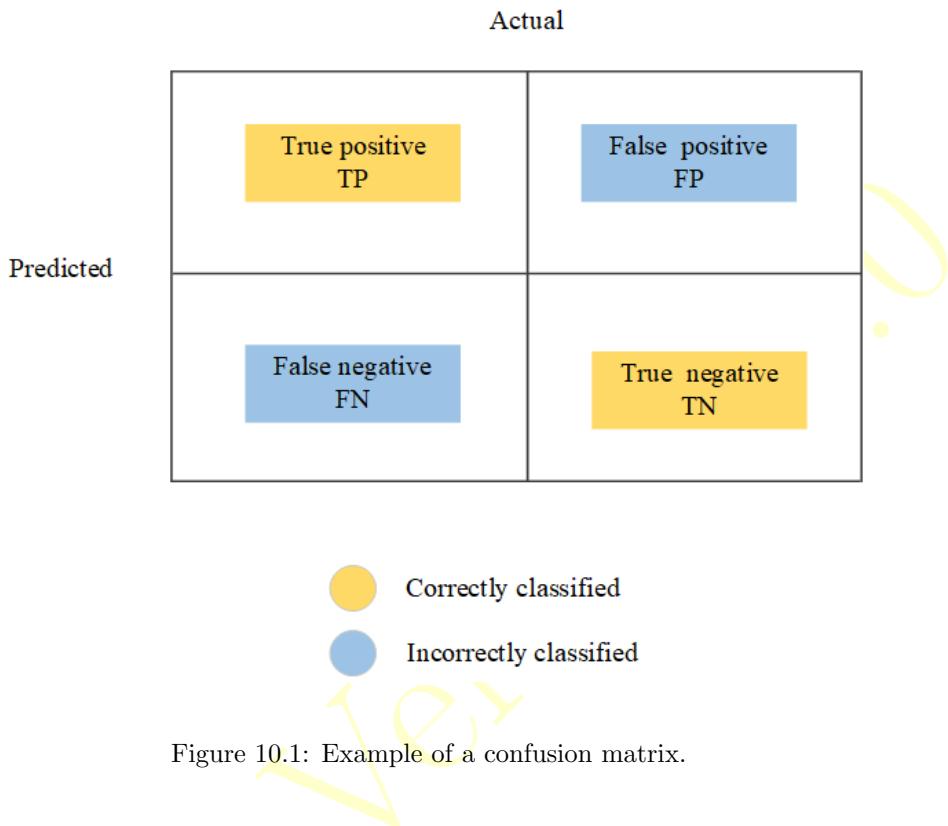
Accuracy is calculated as follows:

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \quad (10.10)$$

Accuracy can also be defined for every single class (per-class accuracy) to avoid class skewness (i.e. an imbalanced number of samples among classes in the training dataset).

### 10.1.2 Confusion matrix

A confusion matrix (see Figure 10.1) is a visualization tool used to identify the number of correctly or incorrectly classified points. A confusion matrix is useful when the error generated by a false positive and the one generated by a false negative have different relevance in practice which accuracy is not able to detect (since it averages all positives and negatives together).



### 10.1.3 Log-loss

Log-loss can be used as a “soft” (see section 8.2) measure of accuracy when the classifier outputs a probability that an observation belongs to a class (or another). Log-loss is calculated as:

$$\log\text{-}loss = -\frac{1}{N} \sum_{i=1}^N y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \quad (10.11)$$

This definition of log-loss is similar to the cross-entropy in the information theory, which measures the unpredictability of something. In practice, by minimising the cross-entropy, the accuracy is maximised.

### 10.1.4 AUC

The area under the curve (AUC) is a scalar indicator calculated as the area of a Receiver Operating Characteristic (ROC) curve. The ROC curve shows

the sensitivity of the classifier by plotting the rate of true positives to the rate of false positives. The sensitivity of a classifier is defined as follows.

$$\text{True positive rate} = \text{specificity} = \frac{TN}{FP + TN} \quad (10.12)$$

$$\text{False positive rate} = 1 - \text{specificity} = \frac{FP}{FP + TN} \quad (10.13)$$

Specificity, together with specificity, is used to define the ROC curve. A ROC curve defines the performance of a classifier by plotting the false positive rate on the x-axis, and the true positive rate on the y-axis. An ideal classifier (classifying observations 100% correctly) would go to a 100% rate of true positives immediately. This is unlike to happen in practice, but the faster the curve to go closer to 100%, the better the classifier. To quickly compare the performance of different models, the area under the ROC curve is calculated, and the higher the area, the better the performance of the classifier.

### 10.1.5 Precision and recall

Based on the confusion matrix, it is possible to identify other metrics to evaluate the classification model. There is no optimal evaluation metric. The choice of the metric depends on the problem instance and the risk of misclassification into false positives or false negatives.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (10.14)$$

$$\text{Recall} = \text{Sensitivity} = \frac{TP}{TP + FN} \quad (10.15)$$

Precision gives importance to the cost of false-positive. It answers the question “*Out of the items that the classifier predicted to be relevant, how many are truly relevant?*” On the other side, sensitivity (or recall) gives importance to the cost of a false negative. Answering the question: “*Out of all the items that are truly relevant, how many are detected by the classifier?*”. Precision and recall can be matched together by plotting a precision versus recall curve (similar to a ROC curve) or calculating the so-called F1 score.

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (10.16)$$

A low F1 score indicates that either precision or recall is small.

## 10.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) assumes the decision boundaries being linear. In particular, each class  $k$  is assumed having Gaussian density (defined by a multivariate Gaussian distribution), consequently:

$$f_k(x) = \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)} \quad (10.17)$$

LDA assumes all classes  $k$  has the same covariance matrix  $\Sigma$ . Under this hypothesis, the equation describing the decision boundaries remains a linear function of  $x$ . To compare two classes  $k$  and  $l$  defining decision boundaries, the log ratio is considered.

$$\begin{aligned} \log \left( \frac{Pr(G=k|X=x)}{Pr(G=l|X=x)} \right) &= \log \left( \frac{f_k(x)}{f_l(x)} \right) + \log \left( \frac{\pi_k}{\pi_l} \right) = \\ &= \log \left( \frac{\pi_k}{\pi_l} \right) - \frac{1}{2} (\mu_k + \mu_l)^T \Sigma^{-1} (\mu_k - \mu_l) + x^T \Sigma^{-1} (\mu_k - \mu_l) \end{aligned} \quad (10.18)$$

Decision boundaries are found where the equation (10.18) equals zero. Assuming that all classes have an equal covariance matrix  $\Sigma_k = \Sigma \forall k$  implies that the decision boundaries are linear in  $x$ . In other words, the  $p$ -dimensional space  $\mathbb{R}^p$ , where the points in  $X$  lie, is divided by hyperplanes (for example, when  $X \in \mathbb{R}^2$  i.e.,  $p = 2$  the plane is divided by a number of straight lines). The resulting discriminant function is as follows.

$$\delta_k(x) = x^T \Sigma^{-1} \mu_K - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k \quad (10.19)$$

In practice, the model assumes a certain distribution of the data (the multivariate Gaussian) estimating the parameters as follows. Given the number  $N_k$  of the observations for each class  $k$ , having

$$\hat{\pi}_k = \frac{N_k}{N} \quad (10.20)$$

$$\hat{\mu}_k = \sum_{g_i=k} \frac{x_i}{N_k} \quad (10.21)$$

$$\hat{\Sigma} = \sum_{k=1}^K \sum_{g_i=k} \frac{(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T}{N - K} \quad (10.22)$$

In general, when the covariance matrix  $\Sigma_k$  are different, the discriminant functions are quadratic (QDA).

$$\delta_k(x) = -\frac{1}{2} \log |\Sigma_k| - \frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \log \pi_k \quad (10.23)$$

It is possible to apply LDA as well, by shrinking the covariance matrices, using an hyperparameter  $\alpha$  and then applying LDA.

$$\hat{\Sigma}(\alpha) = \alpha \hat{\Sigma}_k + (1 - \alpha) \hat{\Sigma} \quad (10.24)$$

## 10.3 Logistic Regression

Similarly to LDA, Logistic regression is a linear model used to predict the value of a discrete variable. The model has the form:

$$\begin{aligned} \log \left( \frac{\Pr(G=1|X=x)}{\Pr(G=K|X=x)} \right) &= \beta_{10} + \beta_1^T x \\ \log \left( \frac{\Pr(G=2|X=x)}{\Pr(G=K|X=x)} \right) &= \beta_{20} + \beta_2^T x \\ &\dots \\ \log \left( \frac{\Pr(G=K-1|X=x)}{\Pr(G=K|X=x)} \right) &= \beta_{(K-1)0} + \beta_{(K-1)}^T x \end{aligned} \quad (10.25)$$

The model uses  $K - 1$  classes while the  $K$ -th class can be arbitrarily chosen and is always found in the denominator of the model. Differently from LDA (where linearity is given by the assumption of Gaussian parameter), logistic regression does not assume the distribution of  $X$ , but it is linear by construction. To get a simplified notation we introduce  $\theta = \beta_{10}, \dots, \beta_{(K-1)0}; \beta_1^T, \dots, \beta_{K-1}^T$ , and

$$\Pr(G=k|X=x) = p_k(x, \theta) \quad (10.26)$$

Since the probability distribution of  $x$  is not given apriori, it is not possible to develop the equation and get a closed-form formula to fit the model. A maximum likelihood estimator (MLE) is, then, used to fit the model and get  $\beta$ . A multinomial distribution is appropriate to define the distribution of  $\Pr(G|X)$ . Here the case of  $K = 2$  is discussed for brevity. Using the multinomial distribution, the formula of the log-likelihood<sup>2</sup> of  $\beta$  is:

$$l(\beta) = \sum_{i=1}^N y_i \log(p(x_i; \beta)) + (1 - y_i) \log(1 - p(x_i; \beta)) \quad (10.27)$$

---

<sup>2</sup>Log-likelihood is used since the logarithm is an increasing function and it is equivalent to maximize the likelihood or the log-likelihood. In this application and many other cases, maximizing a logarithm results much more simple.

To get an MLE the  $\frac{\partial l(\beta)}{\partial \beta}$  is computed, set to zero and solved in  $\beta$ . This implies solving  $p + 1$  non-linear equations in  $\beta$ . The calculus can be done using the Newton-Raphson algorithm which usually converges since log-likelihood is concave.

## Further reading

Supplementary reading materials can be found in [1].

## Bibliography

- [1] M. Bramer, *Principles of Data Mining*. 2016.

# 11

## Non-linear methods

*The world does not behave linearly. Models do.*

The previous chapters introduce regression and classification, presenting numerical example from linear models. Often, linear models cannot be applied in practice since they simplify too much the behaviour of reality. For this reason, non-linear models are implemented to build supervised models both for regression, and classification. Well-established mathematical models define non-linear models. Differently from linear models that the solving approach is not exact, but approximated by efficient heuristics.

This chapter presents non-linear models according to the classification of machine learning methods into the five tribes [1]. Each tribe is related to a specific branch of science whose researchers are used to learn and make discoveries with a precise technique. Each tribe has a precise way of working:

1. Evolutionaries create knowledge by evolving structures;
2. Connectionists create knowledge by learning parameters;
3. Symbolists create knowledge by composing element on the fly;
4. Bayesians create knowledge by weighting evidence;
5. Analogisers create knowledge by mapping to new situations.

All these tribes have a precise way of thinking, a specific optimisation algorithm, an evaluation metric and a representation tool (i.e. the formal language of each tribe). Figure 11.1, introduced in [1] illustrates all these elements and suppose the existence of a master algorithm able to merge the five approaches, aiming at learning anything.

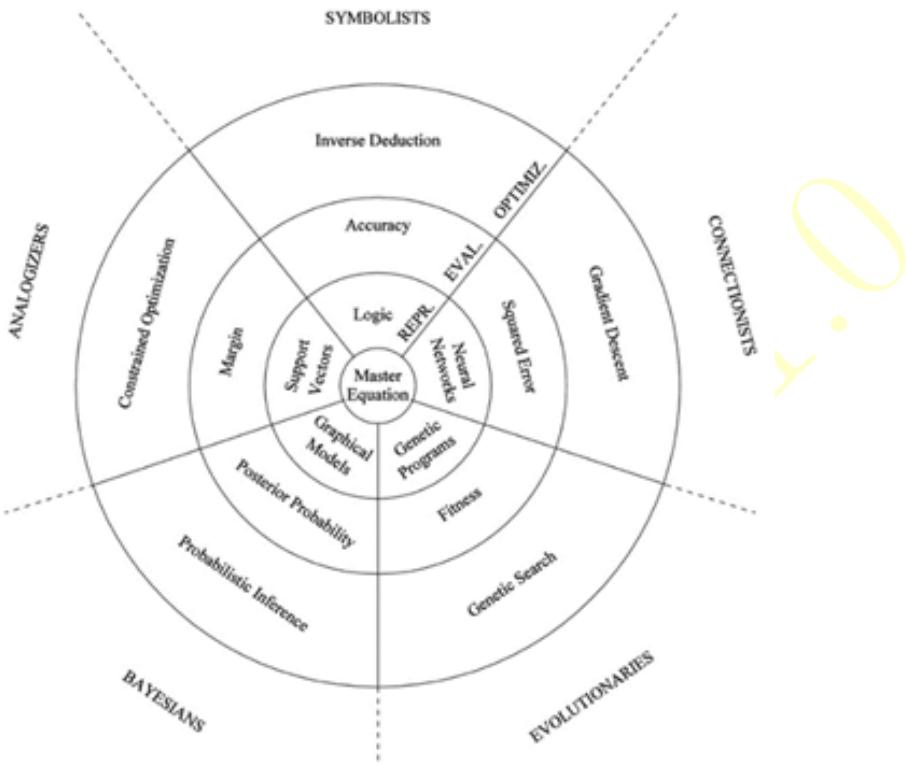


Figure 11.1: The five tribes optimisation, evaluation, and representation tools, from [1].

## 11.1 Evolutionaries' methods

The name evolutionaries comes from the theory of evolution of Charles Darwin [2]. With this tribe, we refer to researchers and methods born in the field of biology to model and understand how nature evolves. The machine learning tribe called evolutionaries mimics the natural behaviour to learn information.

The formal language of evolutionaries is the genetic programming and they think that it leads to classifying systems, as the classification of the species. Their evaluation metric is the goodness of fit that measures the distance between the outcome of a model and its true value. Evolutionaries use search algorithms that move on the input data hyperplane to find solutions. These algorithms are called genetic search algorithms and belong to the family of metaheuristics.

### 11.1.1 Genetic search

Genetic search algorithms are metaheuristic algorithms that consider an initial feasible solution of a prescriptive problem and make it evolve by mimics of the natural evolution mechanisms as mutations and cross-over. They aim at finding a new feasible solution with a better performing solution value.

Figure 11.2 identifies the mutation of a string of data, where a bit of the original input has been randomly changed. The impact of this mutation on the solution value is evaluated by the goodness of fit of the model.

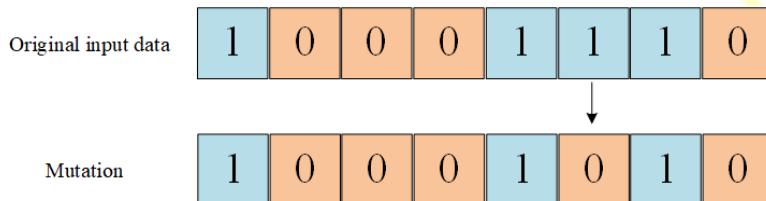


Figure 11.2: Example of mutation of a string of data.

Figure 11.3 identifies an example of a cross-over of two strings of data. The values of the string are pivoted around a splitting point and combined to generate two new strings. The fitness function evaluates if the cross-over leads to an improvement of the solution value.

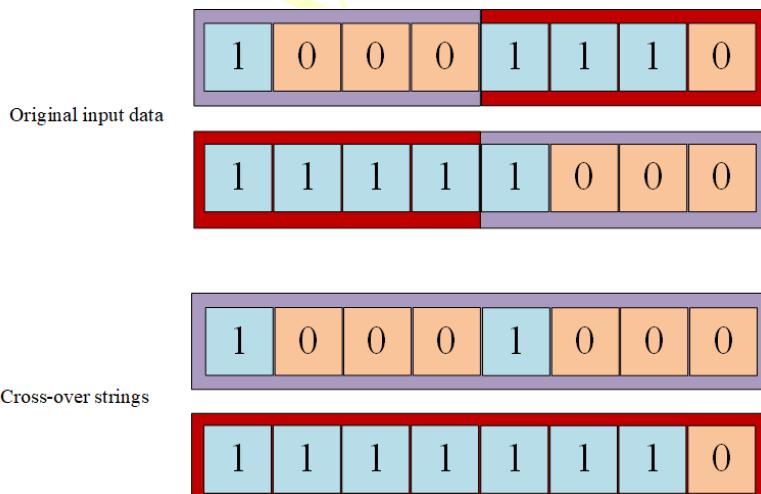


Figure 11.3: Example of cross-over of a string of data.

## 11.2 Symbolists' methods

The symbolists use the formal language of the logic to learn and generate information. We already introduced the logic in Chapter 5, and we have already seen the application of logic to unsupervised learning models in the definition of association rules (see Section 8.1). A similar “if-else” statement can be used to learn information in a supervised fashion by using decision trees. Decision trees are the application of the formal language of symbolists to supervised learning. Their evaluation metric is measured through the accuracy and the information gain.<sup>1</sup>

### 11.2.1 Decision trees

Decision trees can be used both for regression and for classification. The idea behind a regression tree is to split the space of the input features into  $M$  partitions such that the response of the model is:

$$f(x) = \sum_{m=1}^M c_m I(x \in R_m) \quad (11.1)$$

In other words, the model assigns a constant response value  $c_m$  for each partition  $m$  of the feature space. The model can be represented as a tree where the observations are split based on the value of a splitting variable  $j$  and a splitting point  $s$ . Each branch, consequently, divides the current feature region into two regions  $R_1$  and  $R_2$  such that  $R_1(j, s) = \{X | X_j \leq s\}$  and  $R_2(j, s) = \{X | X_j > s\}$ . The branching variable  $j$  and its splitting point  $s$  are chosen to minimize the error function:

$$\min_{j,s} \left[ \min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right] \quad (11.2)$$

At each stage of the tree, all the splitting points  $s$  are tested for each variable  $j$  very quickly defining two new partitions of the feature space. The responses  $c_1$  and  $c_2$  are defined as:

$$\begin{aligned} c_1 &= E[(y_i | x_i \in R_1(j, s))] \\ c_2 &= E[(y_i | x_i \in R_2(j, s))] \end{aligned} \quad (11.3)$$

A minimum node size can be predetermined to avoid an exponential growth of the number of nodes of the tree. Alternatively, a tree node can

---

<sup>1</sup>The package logproj provides methods to deal with symbolists' methods [here](#).

be split only if the decrease in the sum-of-squares exceeds a threshold. This can be expressed using a cost-complexity criterion:

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T| \quad (11.4)$$

Where  $N_m = \text{card}(x_i \in R_m)$ ,  $c_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i$  and  $Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - c_m)^2$ .  $Q_m(T)$  is a measure of impurity in each partitioned region, the equation (11.4) aim at defining a subtree that controls the tradeoff between impurity and the growth of the tree.

Classification trees work as regression trees with a different definition of the impurity function. Classification trees offer different impurity functions  $Q_m(T)$ :

- The misclassification error  $Q_m(T) = 1 - p_{mk}$ ;
- The Gini index  $Q_m(T) = \sum_{k=1}^K p_{mk}(1 - p_{mk})$ ;
- The cross-entropy  $Q_m(T) = -\sum_{k=1}^K p_{mk} \log p_{mk}$ .

All these impurity functions are based on the value of  $p_{mk}$ , i.e. the proportion of class  $k$  observation in node  $m$ .

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \quad (11.5)$$

Where  $N_m = \text{card}(x_i \in R_m)$ .

Similarly to regression trees, a new branch of a decision tree is opened if it reduces the value of the impurity function at that node  $m$  compared to the weighted impurity of the nodes of the new sub-tree.

When implementing a decision tree, it is possible to measure the relative importance of an input feature  $l$  by using:

$$I_l^2(T) = \sum_{t=1}^{J-1} \hat{i}_t^2 I(v(t) = l) \quad (11.6)$$

Where  $J-1$  are the internal nodes of the regression tree  $T$  and  $\hat{i}_t^2$  is the maximal estimated improvement in the squared error risk obtained by choosing  $l$  as splitting variable at the node  $t$ .

## 11.3 Bayesians' methods

Bayesian methods rely on the well known Bayes' theorem that we already introduced in section 6.6. By using prior and posterior probability, we

have the possibility to update the belief we have on an event when new observation (e.g. new data or data from another data source) are available. The formal language of the bayesians is the graphical modelling as Bayesian and Markov networks. The posterior probability given from the empirical observation is the evaluation metric of these models. The search method is the averaging between initial beliefs given by the prior (i.e. the initial assumptions) and the posterior (i.e. the empirical evidence).<sup>2</sup>

### 11.3.1 Naïve Bayes

The prior probability of the Bayes theorem reflects the initial belief we have on an event. The posterior probability indicates all the other data we add to support or discard our initial belief. When applying the Bayes theorem to machine learning, we obtain a relationship between causes (i.e. the input dataset  $X$ ) and effect (the target variable  $y$ ):

$$P(\text{cause}|\text{effect}) = \frac{P(\text{cause}) \times P(\text{effect}|\text{cause})}{P(\text{effect})} \quad (11.7)$$

Unluckily, phenomena have many causes leading to and effect.

We have already seen that any machine learning model can be interpreted as a method to estimate the joint probability distribution between the input features, and the target (i.e. the cause, and the effects). Naïve Bayes approaches this problem simply, by assuming all the causes, i.e. the input features are independent. Under this assumption, the joint density distribution is obtained as:

$$f_j(X) = \prod_{k=1}^p f_{jk}(X_k) \quad (11.8)$$

Each marginal density distribution can be estimated using one-dimensional kernel density estimation (see section 6.2.3). Naïve Bayes calculates the product of these kernels, and the logit transform is, then, used to get a model in an additive form (similar to the perceptron algorithm illustrated

---

<sup>2</sup>The package `logproj` provides methods to deal with bayesians' methods [here](#).

in chapter 10).

$$\begin{aligned}
 \log \left( \frac{\text{Prob}\{G = l|X\}}{\text{Prob}\{G = J|X\}} \right) &= \log \left( \frac{\pi_l f_l(X)}{\pi_J f_J(X)} \right) = \\
 &= \log \left( \frac{\pi_l \prod_{k=1}^P f_{lk}(X_k)}{\pi_J \prod_{k=1}^P f_{Jk}(X_k)} \right) = \\
 &= \log \left( \frac{\pi_l}{\pi_J} \right) + \sum_{k=1}^P \log \left( \frac{f_{lk}(X_k)}{f_{Jk}(X_k)} \right) = \\
 &= \alpha_l + \sum_{k=1}^P g_{lk}(X_k)
 \end{aligned} \tag{11.9}$$

### 11.3.2 Markov Chains

The hypothesis of the independence between the input variables is hardly often valid. On the other side, the definition of an accurate joint probability distribution requires a huge amount of data and a significant computational time. To overcome these obstacles, Markov chains assume that the probability of an event only depends on the current state of a system.

The  $p$  attributes of the input dataset  $X$  define the state of a system, and they are linked together by a transition probability. This way, even if the number of features  $p$  is high, the number of transition probabilities  $t$  calculate is still limited. Figure 11.4 introduce the graphical model of a Markov chain, whose information content is described by a transition matrix  $T$  with states  $i$  and  $j$  as rows and columns labels. The entries  $t_{ij} \in T$  identifies the probabilities from  $i$  to  $j$ .

### 11.3.3 Hidden Markov Models and Kalman filter

Sometimes, state variables are not observable, but the state depends on a subset of measurable variables that affect the state variables, and a model (i.e. a prior) of how these variables affect the state is given (see Figure 11.5).

Kalman filter implements this rationale when the states and the observations are continuous variables [3]. The filter works by considering a prior knowledge given by a model describing the state of a system [4, 5]. The state of the system is estimated and updated online by considering empirical measurements of the transitions (representing a posterior knowledge). Let  $\mu$  and  $\sigma^2$  be the mean value and standard deviation of the prior and  $\nu$  and  $r^2$  the ones of the posterior. The Kalman filter works correctly when the measurements have a higher information content than the model, i.e. their

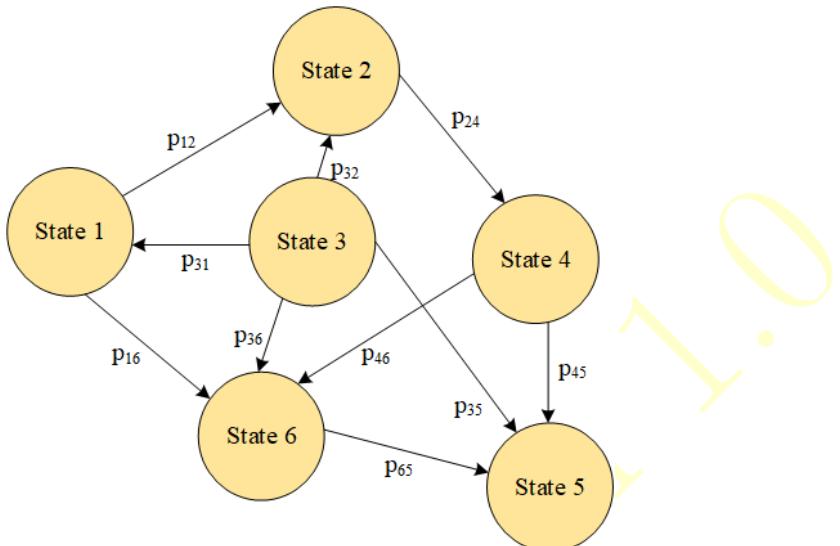


Figure 11.4: Network of a Markov chain.

accuracy is higher  $r^2 < \sigma^2$ . This hypothesis is often verified since the measurements depend on the accuracy of the instrument on a single transition (see 6.2.6), while models tend to be more general and inaccurate. Kalman filter calculates a new mean  $\mu'$  and  $\sigma'^2$  of the system state as:

$$\begin{aligned}\mu' &= \frac{r^2\mu + \sigma^2\nu}{r^2 + \sigma^2} \\ \sigma'^2 &= \frac{1}{\frac{1}{r^2} + \frac{1}{\sigma^2}}\end{aligned}\tag{11.10}$$

At this stage, the state is updated by considering the transition measurement  $u$ :

$$\begin{aligned}\mu' &= \mu + u \\ \sigma'^2 &= \sigma^2 + r^2\end{aligned}\tag{11.11}$$

### 11.3.4 Bayesian Networks and Montecarlo Markov Chains

When multiple prior and posteriors are involved, Bayesian networks can be used to describe a system. In practice, a Bayesian network defines a number of variables with a specific probability distribution and a connection between them. This way, it is possible to model multiple interconnected events

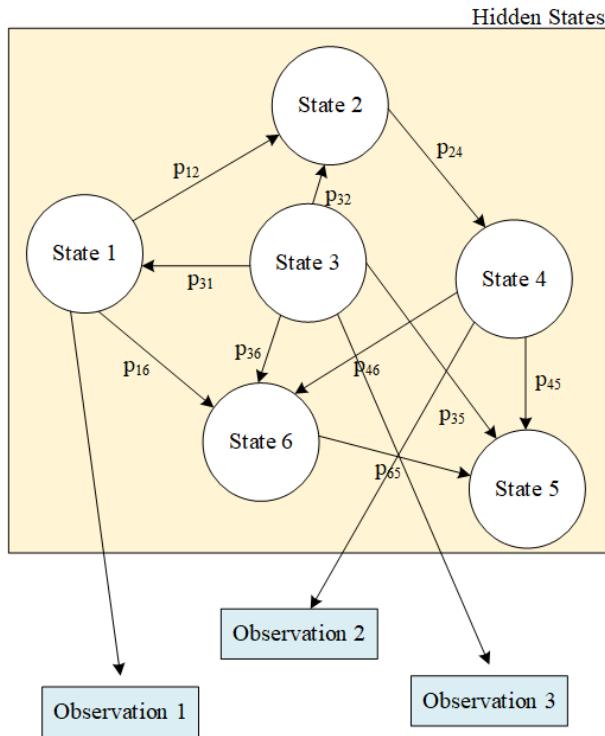


Figure 11.5: Network of a Hidden Markov Model

with different probability distributions. The parameters of a probability distribution can be variables as well.

Let us consider the example where we know that if it is sunny, we will go to the beach. We are interested in knowing the probability of going to the beach, without any empirical observations on it. Nevertheless, we know that it is sunny when it is not raining:  $\text{Prob}\{\text{sun}\} = 1 - \text{Prob}\{\text{rain}\}$ . We assume that rain probability is distributed as a Poisson probability distribution whose parameter  $\lambda$  follows a normal distribution. All these elements are the priors of our model (see Figure 11.6).

We are interested in sampling the posterior, i.e. to measure the probability of going to the beach, given all these parameters. A technique called Montecarlo Markov Chain (MCMC) permits to sample from the posterior, given the connection between all the random variables of a Bayesian network, and given observations of some of these variables. For example, we have past observations of the probability of rain, from the weather forecasts. MCMC can fit these empirical observations to the Poisson probability distribution and compute all the other parameters of the network consequently.

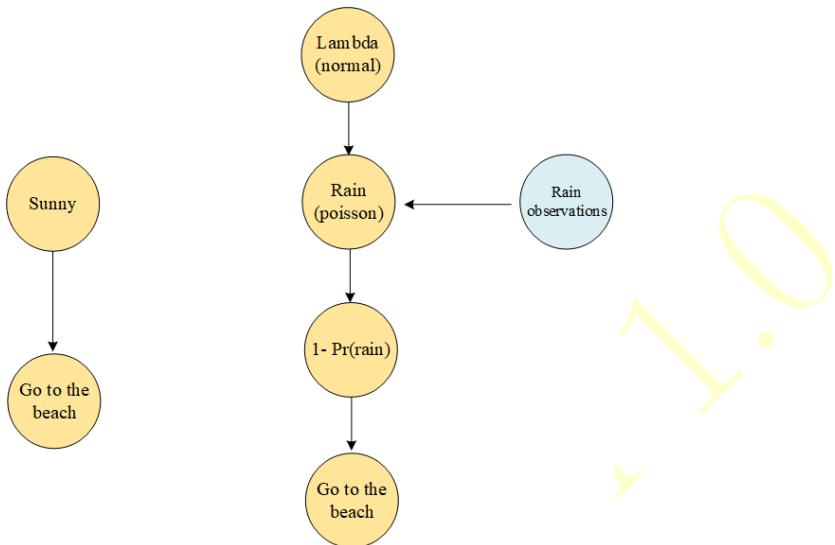


Figure 11.6: Example of a Bayesian network.

## 11.4 Analogisers' methods

Analogisers proceeds by labelling items depending on their similarity to a set of given items. The simplest approach is the  $k$ -nearest neighbour, an algorithm computing the distance of a new observation from all the given observations and assigning to it the label of the closes. The formal language of analogisers is the weighting of a specific instance as in the Support Vector Machine whose evaluation metric is the margin. Their search method is the constrained optimization that moves within the boundaries, looking for the best values.<sup>3</sup>

### 11.4.1 Support vector machines

Support vector machines (SVM) are based on the concept of support vector classifier (SVC); they can be interpreted as a weighted  $k$ -nearest neighbour. An SVC identifies thresholds are defined to identify regions of the hyperplane containing all the input data. Different regions have different labels. New observations are classified depending on the region of the hyperplane where they fall. The shortest distance between the observation and the threshold dividing the hyperplane's regions is called margin.

By setting thresholds that gives the largest margin, there is a high risk of misclassification if the input dataset contains outliers. For this reason, it

---

<sup>3</sup>The package logproj provides methods to deal with analogisers' methods [here](#).

is necessary to allow the SVC to misclassify the input data. This way, we have data points falling within a hyperplane's region with a different label. This is called soft margin classifier or SVC since the observation on the edge of a region, and within the soft margin are called support vectors. The support vectors define the regions' boundaries. The support vectors always have a  $p - 1$  dimension, where  $p$  is the dimension of the input dataset.

A SVC defines hyperplanes  $f(x) = x^T \beta + \beta_0$  such that  $y_i f(x_i) > 0 \forall i$ . The hyperplane which best separates the space is defined by the optimisation problem:

$$\begin{aligned} & \max_{\beta, \beta_0, |\beta|=1} M \\ & y_i (x_i^T \beta + \beta_0) \geq M, \quad i = 1, \dots, N \end{aligned} \tag{11.12}$$

In case the classes overlap, it is still possible to maximise  $M$  allowing for some points to be on the other side of the margin. Let  $\xi$  be a slack variable  $\xi = \xi_1, \dots, \xi_N$ , we have:

$$y_i (x_i^T \beta + \beta_0) \geq M - \xi_i \tag{11.13}$$

Where the value of  $\xi$  indicates a proportional amount by which the prediction is on the wrong side of the margin.

SVC have a problem when a region is enclosed by another, i.e. there is no possibility to define a support vector  $p-1$  dimensional to separate the two regions correctly. Kernel transformations are used to overcome this problem. The input dataset is projected into a hyperplane with a higher dimension, and then an SVC is used. This procedure is called SVM. Common kernels are polynomial kernel with dimension  $d$ , that projects the point using a power elevation of their values, or the non-linear radial basis function.

## 11.5 Connectionists' methods

Connectionists aim at learning by mimic the connection of the human brain. The human brain is composed of interconnected neurons activated by electrical impulses. The formal language of the connectionists is the neural network. A neural network has an input similar to the input from the five senses of a human. Depending on the input, different groups of neurons are activated, producing the output. The evaluation metric of connectionists is a continuous error metric, as the mean squared error (see section 9.1) that measures the difference between the know target label, and the output of the neural network. The search algorithm to define the behaviour of the neurons is the gradient descent.<sup>4</sup>

---

<sup>4</sup>The package logproj provides methods to deal with connectionists' methods [here](#).

### 11.5.1 Neural Networks

Neural networks work by mimic a network of neurons. Figure 11.7 illustrate how a mathematical neuron is defined. A neuron has some inputs, a bias input quantity  $b$ . All these inputs are summed up and processed by an activation function. This is usually a sigmoid function, as a logistic function, a hyperbolic tangent, a rectified linear (RELU) or a fixed threshold. If the sum of the input is enough to activate the function, a positive output is obtained; otherwise, the energy provided by the inputs is not enough to activate the neuron, that produces a zero.

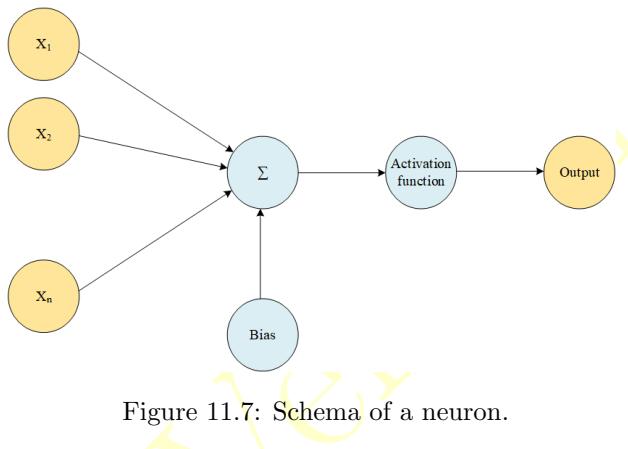


Figure 11.7: Schema of a neuron.

In practice, each neuron works similarly to a logistic regressor. Putting together multiple neurons into deep layers is what they called deep learning. The central idea of a neural network is to extract linear combinations of the input features  $X$  to create hidden features  $Z$  and model the target as a non-linear function of these features.

There are neural networks with complex mechanisms and topologies. For the sake of brevity, we analyse how a single layer neural network works. This model is also known as “single-layer perceptron Let consider the case of classification into  $K$  different classes. The model involves:

- $p$  input features  $x_1, \dots, x_p$ ;
- $M$  derived features  $Z_1, \dots, Z_m, \dots, Z_M$ ;
- $K$  target  $Y_1, \dots, Y_k, \dots, Y_K$ .

The main idea is modelling the target  $Y$  as a combination of  $Z$ .

$$f_k(X) = g_k(T) \quad (11.14)$$

Where:

- $T_k = \beta_0 + \beta_k^T Z$ ,  $k = 1, \dots, K$  is a linear combination of the hidden layer features  $Z$ , with
- $Z_m = \sigma(\alpha_{0m} + \alpha_m^T X)$ ,  $m = 1, \dots, M$ .

The function  $\sigma$  is the activation function, usually the sigmoid function  $\sigma(v) = \frac{1}{1+e^{-v}}$ . This function is a non-linear transformation of the input  $X$ , which is the core of neural networks. If  $\sigma$  equals an identity function, all the model collapses to a linear model.  $g_k$  is a transformation function to get the final input. For regression the identity function  $g_k(T) = T_k$  is used while the softmax function is used for  $k$ -classification problems  $g_k(T) = \frac{e^{T_k}}{\sum_{l=1}^K e^{T_l}}$ .

Fitting a neural network means to identify the unknown weights  $\alpha_{0m}, \alpha_m$ ;  $m = 1, \dots, M$  and  $\beta_0, \beta_k; k = 1, \dots, K$ . A neural network use the gradient descent algorithm to find the value of these parameters minimising an error metric as the sum-of-squared errors  $\sum_{k=1}^K \sum_{i=1}^N (y_{ik} - f_k(x_i))^2$  for regression, or the cross-entropy  $\sum_{i=1}^N \sum_{k=1}^K y_{ik} \log f_k(x_i)$  for classification.

## Bibliography

- [1] P. Domingos, “The Master Algorithm,” in *Basic Books*, 2015.
- [2] C. Darwin, *On the Origin of the Species*. 1859.
- [3] G. Anandalingam and L. Chen, “Bayesian forecast combination and kalman filtering,” *International Journal of Systems Science*, vol. 20, no. 8, pp. 1499–1507, 1989.
- [4] M. LAARAIEDH, “Implementation of kalman filter in python Language.”
- [5] H. Fang, N. Tian, Y. Wang, M. Zhou, and M. A. Haile, “Nonlinear Bayesian estimation: From Kalman filtering to a broader horizon,” *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 2, pp. 401–417, 2018.

Draft Version 1.0

# 12

## Ensemble methods

*A choir sings better than a single voice.*

In the chapter 11 we introduced machine learning methods born from different research approaches. Each of them has peculiarity and limits, and there is not the best method that outperforms the others. Ensemble learning aims at combining the power of these different learners to obtain a more effective result. The idea behind ensemble learning is to combine different prediction models and consider their output together to make predictions on the target variable. Ensemble methods are mainly based on bagging and boosting (see Figure 12.1). Bagging combines the effect of multiple independent learners while boosting creates a pipeline of learners in sequence.<sup>1</sup>

### 12.1 Bagging

The word *bagging* means *bootstrap aggregation*. This method uses the bootstrap sampling technique (see section 6.2.4) to get  $B$  different samples of the original dataset and train  $B$  different models on them. The type of model to train can be chosen arbitrarily. The final prediction will be the average of the prediction  $f^b$  of the  $B$  models.

$$f(x) = \frac{1}{B} \sum_{b=1}^B f^b(x) \quad (12.1)$$

<sup>1</sup>The package logproj provides methods to deal with ensemble methods [here](#).

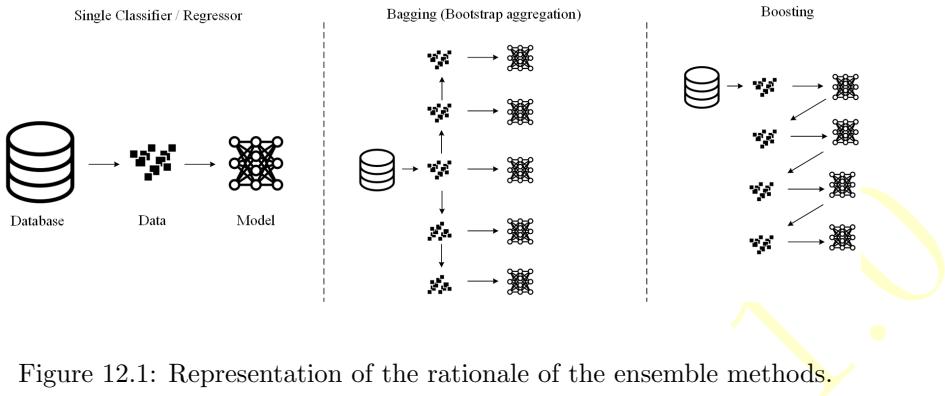


Figure 12.1: Representation of the rationale of the ensemble methods.

## 12.2 Random Forests

A Random Forest is a bagging method whose model is a decision tree or a regression tree. The underlying idea is to average many unbiased trees to reduce the variance in the predictions. Random forests work as follows.

---

### Algorithm 8: Random forests algorithm

---

Import the input dataset with  $N$  observations, and  $p$  features

Set the number of iterations  $B$

**for**  $b = 1 : B$  **do**

    Get a bootstrap sample  $Z^*$  of size  $N$

    Set a number  $m$  of features to extract

    Grow a tree tree  $T_b$  on  $Z^*$  doing the following

**while** a minimum node size  $n_{min}$  is not reached. **do**

            Select  $m$  variables out of  $p$  at random

            Choose the best variable split on  $m$

            Split the node into two daughter nodes

**end**

**end**

Output the ensemble of trees  $T_1, \dots, T_B$

---

Predictions are obtained differently depending on regression or classification. The predictions for a regression tree are obtained by averaging all the output predictions:

$$f(x) = \frac{1}{B} \sum_{b=1}^B T_b(x) \quad (12.2)$$

For a classification tree the function of majority vote is considered:

$$f(x) = \text{majority vote } \{C_1(x), \dots, C_B(x)\} \quad (12.3)$$

where  $C_b(x)$  is the class predicted by the  $b$  tree of the random forest.

The hyperparameters of a random forest are the number  $m$  of features to extract and the minimum number of nodes  $n_{min}$ . Valid tuning parameter for regression random forests are  $m = \sqrt{p}$ , and  $n_{min} = 1$ ; while for classification random forests are used  $m = p/3$ , and  $n_{min} = 5$ .

The performance of a random forest can be assessed through a traditional error metric (e.g. the MSE) calculated on out-of-bag (OOB) samples. OOB samples are the samples of the original dataset which does not appear in the bootstrapped dataset. This measure of error is similar to the cross-validation.

## 12.3 AdaBoost

*AdaBoost* means adaptive boosting, and it has been one of the firsts boosting methods. The main idea behind these methods is to combine the outputs of many *weak* classifiers to get a robust prediction. A weak classifier is any classification function with an error rate slightly better than random classification (i.e., flipping a coin). These *weak* classifiers are called *stumps*, and can be modelled as decision trees with a single split. Each of the classifiers votes to produce a different weighted contribution on the final prediction. AdaBoost born as a classification method on two classes  $y \in \{-1; 1\}$ . The final prediction is obtained as:

$$G(x) = \text{sign} \left( \sum_{m=1}^M \alpha_m G_m(x) \right) \quad (12.4)$$

Where  $G_m(x)$  is the contribution of each weak predictor and  $\alpha_1, \dots, \alpha_m$  are the relative importance of each predictor. The AdaBoost algorithm works as follows:

---

**Algorithm 9:** AdaBoost algorithm

---

Set a weight  $w_i = \frac{1}{N}$  for each observation  $i = 1, \dots, N$

**for**  $m = 1 : M$  **do**

    Normalise weights  $w_i$   $i = 1, \dots, N$

    Fit a stump classifier  $G_m(x)$  to the training data

    Compute the error  $err_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}$

    Compute  $\alpha_m = \log \left( \frac{1 - err_m}{err_m} \right)$

    Set  $w_i = w_i e^{\alpha_m I(y_i \neq G_m(x_i))}$   $i = 1, \dots, N$

**end**

Predict the output  $G(x) = \text{sign} \left( \sum_{m=1}^M \alpha_m G_m(x) \right)$

---

At each iteration, the algorithm increases the value of  $\alpha_m$  if the classifier  $G_m$  gives a robust prediction. In addition, a higher weight  $w_i$  is associated

with misclassified observation to give them higher importance in the forthcoming iterations. From this perspective, AdaBoost connects a series of stump classifiers as a chain such that the following classifiers produce good predictions where the previous failed.

## 12.4 Gradient boosting

Gradient boosting is a boosting model similar to AdaBoost with a main difference; it uses decision trees instead of stumps. Gradient boosting minimises a loss function  $L(f) = \sum_{i=1}^N L(y_i, f(x_i))$ , where  $f(x)$  is a sum of trees. Algorithm 10 illustrates how gradient boosting works.

---

**Algorithm 10:** AdaBoost algorithm

---

```

Initialise  $f_0(x) = \arg \min_{\gamma} \sum_{i=1}^N L(y_i, \gamma)$ 
for  $m = 1 : M$  do
    for observation  $i = 1 : N$  do
         $r_{im} = - \left[ \left( \frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right) \right]_{f=f_{m-1}}$ 
    end
    Fit a tree with target  $r_{im}$  and terminal regions  $R_{jm}$ ,
     $j = 1, \dots, J_m$ 
    for  $j = 1 : J_m$  do
         $\gamma_{jm} = \arg \min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$ 
    end
    Set  $f_m(x) = f_{m-1}(x) + \rho \sum_{j=1}^{J_m} \gamma_{jm} (x \in R_{jm})$ 
end
Predict the output  $\hat{f}(x) = f_M(x)$ 
```

---

In particular, working with a regression problem, the initial value is set to the average of the target variable  $f_0(x) = \bar{y}$ . The loss function  $L = \frac{1}{2}(y_i - f(x_i))^2$  is set to the sum of squared residual multiplied by a coefficient  $\frac{1}{2}$  which will be simplified after calculating the gradient of  $L$ .

The algorithm builds  $M$  different trees to get the final prediction. Each tree defines  $J_m$  terminal regions (i.e. leaves) to target the value of  $r_{im}$ . The value of  $r_{im}$  is calculated at each step, based on the value of the loss function  $L$  generated at the previous tree  $f_{m-1}$ . The values of  $f_m$ , i.e. the prediction on  $y$  at the step  $m$  of the algorithm, is generated summing the prediction from the previous step  $f_{m-1}$  plus a value  $\gamma_{jm}$ .  $\gamma_{jm}$  is the output value at the terminal region  $R_{jm}$  chosen to minimise the sum of squared residuals generated by the multiple observations falling into the same leave  $R_{jm}$ .  $\rho \in [0, 1]$  is a learning rate used to reduce the variance produced by the model on the test set. The experience suggests setting the hyperparameters choosing  $\rho = 0.1$  and building trees with 8 to 32 leaves.

Gradient boosting can be used for classification problem as well, by setting the initial value equal to the log of the odds of the target variable. The algorithm updates the values of the  $\gamma_{jm}$  with respect to the residuals of the predicted values calculated on a logistic function.

## 12.5 Mixture of Experts

The *mixture of experts* models combine the predictions of multiple trees in a soft way (see Section 8.2). As the word *mixture* suggests, this is not a hard model, but it works by calculating the probability that a point belongs to a split branch or another. The terminal nodes of this tree are called *experts*, while the non-terminal nodes are *gating networks*. Gating networks have an output in the form:

$$g_j(x, \gamma_j) = \frac{e^{\gamma_j^T x}}{\sum_{k=1}^K e^{\gamma_k^T x}} \quad j = 1, \dots, K \quad (12.5)$$

Where  $g_j(x, \gamma_j)$  represents the probability of assigning a vector  $x$  of the input dataset to the branch  $j$ .  $\gamma_j$  is a vector of unknown parameters representing the soft  $k$ -way split (while dealing with regression tree  $k$  is always equal to 2). At each terminal node (expert), the response variable is in the form:

$$Y \sim \Pr(y|x, \theta_{jl}) \quad (12.6)$$

Where  $Y$  is a linear regression model  $\theta_{jl} = (\beta_{jl}, \sigma_{jl}^2)$ .  $Y = \beta_{jl}^T x + \epsilon \sim N(0, \sigma_{jl}^2)$ . In the case of classification,  $Y$  is the logistic regression.

Draft

Draft Version 1.0

# 13

## Prescriptive analysis

*Aren't we doctor, after all?*

All the previous chapters of this mathematical section, introduced methods to describe and understand a dataset, and even to create knowledge by discovering hidden patterns within data.

When the decision-maker has a clear description of the processes, it is necessary to find decision-support methods to leave a footprint in the real world. This should be the last stage of any serious data-driven process since starting building a decision-support tool without a clear idea of the process where the decision-support tool has to be embedded may lead to catastrophic results.

Besides, it is important to remember that statistics and learning algorithms provide much information. Sometimes, this information is enough for making a decision without additional biases introduced by complex decision-support tools.

We know that prescriptive models are highly complex and incredibly problem-oriented. This fact suggests that they usually embed the highest bias possible and a limited generalisation of their mechanism, which also prevents their reproducibility and their value for the scientific community.

### 13.1 Prescriptive models

Prescriptive models aim at setting the value of decision variables that mimic the decision alternatives in a real context. These models are defined using the following notation.

- $P$  is the decision problem;
- $x$  is the matrix of the decision variables;
- $z$  is the objective function explaining the goodness of  $x$ ;
- $\xi$  is the domain of  $x$ .

Prescriptive models are algorithms minimising (or maximising)  $z$  by changing the entries of  $x$  within the domain  $\xi$ . Prescriptive models outcome are the solution  $\hat{x}$  and the solution value  $\hat{z}$ . Let assume  $P$  being a minimisation problem, we define  $\hat{x}$ :

- optimal solution (indicated with  $x^*$ ) if the value of  $\hat{z}$  is the minimum among all the possible values of  $\hat{z}$ ;
- feasible solution if  $x \in \xi$ ;
- unfeasible solution if  $x \notin \xi$ .

## 13.2 Heuristics and metaheuristics

Considering a minimisation problem  $P$ , a heuristic procedure is any algorithm that produces a feasible solution  $\hat{x}$  performing different step; each step aims at reducing the value of  $z$ . Each step of a heuristic algorithm is called *move*. Usually, a move of a heuristics produces an improvement (i.e., it reduces the value of  $z$ ), when no improvement occurs the algorithm stops returning the value of  $\hat{x}$  and  $\hat{z}$ . Differently, a metaheuristic algorithm allows accepting bad values of  $z$  at the intermediate moves of the algorithm aiming at a lower value of  $z$  in the last move. Figure 13.1 presents the typical behaviour of the moves of heuristics and metaheuristics algorithms.

Heuristics can be:

- greedy algorithms, i.e. they perform a number of steps to build a feasible solution;
- local search, i.e. they perform a number of steps to improve an initial feasible solution.

Metaheuristics usually start from a feasible solution (e.g. produced by a greedy algorithm), and they perform a number of moves according to the type of the algorithm. Examples of metaheuristics algorithms are simulated annealing, tabu search, ant colony algorithm, genetic algorithm (see section 11.1.1). Each of them has a precise logic to generate the solution at each step, it checks (or recovers) the feasibility of the incumbent solution within  $\xi$  improving  $z$ .

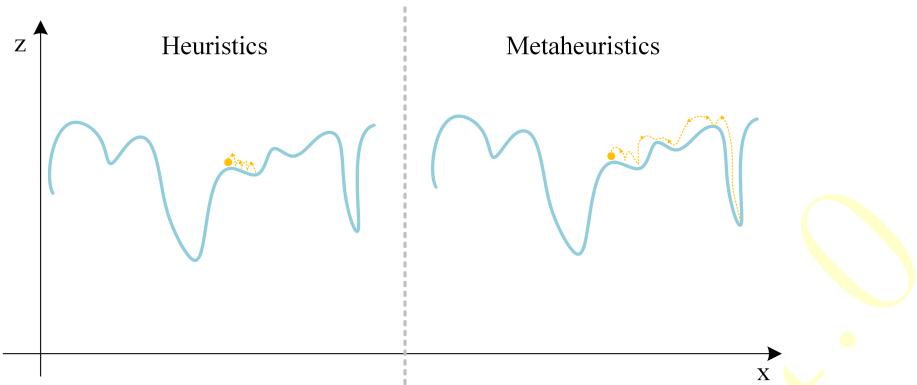


Figure 13.1: Representation of the moves of heuristics and metaheuristics algorithms.

### 13.3 Linear Optimisation

Heuristics and metaheuristics hardly guarantee to find the optimal  $x^*$  and  $z^*$ . Linear optimisation uses predefined algorithms to find the optimal solution to a linear problem  $P$  having  $\xi$  defined by linear constraints. An optimisation problem can be written in the form:

$$\begin{aligned} z &= \min c'x \\ Ax &\leq b \\ x &\geq 0 \end{aligned} \tag{13.1}$$

If  $\xi$  is a continuous domain,  $P$  is a linear problem (LP) and can be solved in polynomial time using the simplex algorithm. If  $\xi$  is made of integer values, the problem is an integer linear problem (ILP), and it is solvable in exponential time using the branch & bound algorithm. Exponential time may take forever to solve an instance; for this reason, ILP optimisation is not suitable for big instance of problems requiring real-time response. In the next paragraph, we introduce some algorithms (still having an exponential complexity) which can improve the running time to get an optimal solution.

#### 13.3.1 Duality

We can find a smart way to compute the optimal solution of an optimisation problem  $P$ , by considering its dual problem  $D$ .

Let consider a primal problem  $P$  defined as:

$$\begin{aligned} z &= \min c'x \\ Ax &\leq b \\ x &\geq 0 \end{aligned} \tag{13.2}$$

The optimal solution value  $z$  of the problem  $P$  is obtained at  $c'x^*$ .

Let consider the Lagrangian relaxation  $L$  of the problem  $P$ , where the set of constraints  $Ax \leq b$  is replaced by a penalty addendum  $p'(b - Ax)$  in the objective function.

$$\begin{aligned} \min c'x - p'(b - Ax) \\ x \geq 0 \end{aligned} \tag{13.3}$$

Let  $g(p)$  be the solution value of the relaxed problem  $L$ .  $g(p)$  is a lower bound of  $c'x$ .

$$g(p) \leq c'x \tag{13.4}$$

Equation (13.4) implies that it is possible to find a vector  $p'$  such that  $p'(b - Ax) = 0$  obtaining a solution value of the relaxed problem equal to the optimal solution value of the non-relaxed problem. The problem to find  $p'$  is called dual problem  $D$ , and it is formally defined as:

$$\begin{aligned} v &= \max p'b \\ p'A &\leq c \\ p &\geq 0 \end{aligned} \tag{13.5}$$

A set of mathematical rules from the theorem of duality allows obtaining the definition of the dual problem of any primal problem. Table 13.1 illustrates these rules.

Primal variables	$x_1 \geq 0$	$x_2 \geq 0$	...	$x_n \geq 0$	Primal relation	$\text{Min } v$
Dual variables						
$y_1 \geq 0$	$a_{11}$	$a_{12}$	...	$a_{1n}$	$\leq$	$b_1$
$y_2 \geq 0$	$a_{21}$	$a_{22}$	...	$a_{2n}$	$\leq$	$b_2$
.	.	.	.	.	.	.
$y_m \geq 0$	$a_{m1}$	$a_{m2}$	...	$a_{mn}$	$\leq$	$b_m$
Dual relation	IV	IV	IV	IV		
Max z	$c_1$	$c_2$	...	$c_n$		

Table 13.1: Table of the rules to obtain the dual problem.

Sometimes, solving the dual problem is easier than solving the primal problem. In particular, the dual problem can be used to:

1. find a lower bound of the solution value of a problem.  $g(p) \leq c'x$  (e.g. to start a branch & bound procedure);
2. generate variables (i.e. columns) for a problem with an exponential number of variables  $p'(b - Ax)$  (Branch & Price algorithm).

### Branch & price algorithm

Branch & price uses the duality theorem to expedite the search of an optimal solution for an ILP problem  $P$  having an exponential number of variables. It is a common technique when we express a problem in a set-covering approach, since set covering formulation has an exponential number of variables.

Let consider, for example, a facility location problem in its set covering formulation where  $w_i$  is the demand of node  $i$ , and  $W$  is the service capacity of a facility. We define the set  $S$  as a set of vectors  $s$ , each one containing a feasible solution of the facility location problem.

$$S = \left\{ s \subseteq \{i = 1, \dots, m\} : \sum_{i \in s} w_i \leq W \right\} \quad (13.6)$$

We formulate the facility location problem in the form of a set covering problem using the variable  $x_s$ , and the parameter  $c_s$ :

$$x_s = \begin{cases} 1 & \text{if configuration } s \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \quad (13.7)$$

$$c_s \text{ cost of serving set } s \quad (13.8)$$

The primal problem  $P$  is defined as:

$$\begin{aligned} & \min c_s x_s \\ & \sum_{s \in S: i \in s} x_s \geq 1 \quad \forall i = 1, \dots, m \\ & x_s \in \{0, 1\} \end{aligned} \quad (13.9)$$

If we relax the integrality of  $P$  we obtain the dual problem D.

$$\begin{aligned} & \max \pi_i \\ & \sum_{i \in S} \pi_i \leq c_s \quad \forall s \in S \\ & \pi_i \geq 0 \end{aligned} \quad (13.10)$$

We defined  $P$  as an ILP problem with an exponential number of variables  $x_s$ . The dual problem, instead, has an exponential number of constraints  $\sum_{i \in S} \pi_i \leq c_s \forall s \in S$ . To optimally solve  $D$  we use the separation procedure (also known as *branch & cut*). Let solve  $P$  and  $D$  and obtain  $x^*, cx^*, \pi^*, g(\pi^*)$ . Let assume  $x^*$  being feasible with no guarantee of optimality for  $P$ :

- if  $\pi^*$  is feasible for  $D$ , then  $x^*$  is optimal for  $P$  (weak duality theorem);
- if  $\pi^*$  is unfeasible for  $D$ , at least a dual constraint is violated, and a column (a variable of the primal problem  $P$ ) must be added to  $x$  to proceed towards optimality.

Since constraints of  $D$  are in the form:  $\sum_{i \in S} \pi_i - c_s \leq 0 \forall s \in S$ , a violated constraint of  $D$  equals to a vector  $s$  respecting the following constraints.

$$\begin{aligned} s : \sum_{i \in s} \pi_i - c_s &> 0 \\ s : \sum_{i \in s} \pi_i - \sum_{i \in s} c_i &> 0 \\ s : \sum_{i \in s} (\pi_i - c_i) &> 0 \end{aligned} \tag{13.11}$$

$\sum_{i \in s} (\pi_i - c_i)$  is the reduced cost of variable  $s$ . To identify the best column to add, an optimisation problem called *column generation problem*  $CG$  is defined.

$$\mu_i = \begin{cases} 1 & \text{if } i \text{ belongs to the column} \\ 0 & \text{otherwise} \end{cases} \tag{13.12}$$

$$\begin{aligned} \max \sum_{i=1}^m \mu_i (\pi_i - c_i) \\ \sum_{i=1}^m \pi_i w_i \leq W \\ \pi_i \in 0, 1 \end{aligned} \tag{13.13}$$

If the reduced cost  $\sum_{i=1}^m \mu_i (\pi_i - c_i)$  is positive, it is worth to add this column to reduce the value of the primal objective function. The procedure of adding columns is repeated until the objective function of equations (13.13) has a positive value (i.e., a reduced cost exists). When there is no reduced cost, all the necessary columns have been added to  $S$  in the primal

problem  $P$ . At this stage, the integrality constraints of  $P$  are restored, and  $P$  is solved to optimality.

A column generation procedure needs that  $S$  contains an initial feasible solution. For this reason, the initial value of  $S$  is initialised to a  $s_{ij} = 1$  if  $i = j$ , '0' otherwise. Figure 13.2 illustrates the evolution of the set  $S$  from an initial feasible solution to the optimal solution obtained by a column generation algorithm; the x-axis identifies the iterations of the algorithm while the y-axis the value of the added column (i.e. yellow for ones and blue for zeros).

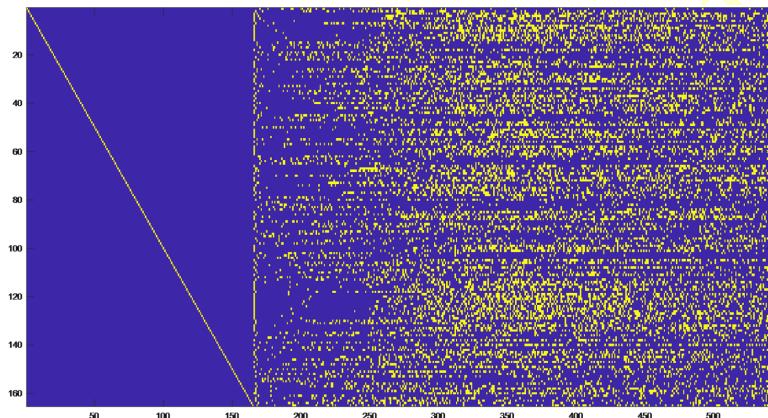


Figure 13.2: Evolution of a column generation algorithm.

## 13.4 Discrete event simulation

Discrete event simulation (DES) is a problem-oriented technique which virtualises an entire system (products, resources, material handling system) and simulates the effect of the production to measure the efficiency of the entire system and get an estimate of some parameters as queues and takt-time dynamically. A DES model, as well as its outcomes, is hard to generalise to multiple scenarios.

## 13.5 Multiple decisions or decision-makers

All the methods presented in this chapter are linked to a single objective and a single decision-maker. They can be easily generalised into multiple decision-making techniques using, for example, multi-objective optimisation. Nevertheless, when the information on the decision or the decision

itself is distributed among different actors, there are different techniques (as game theory agent-based theory) which results adequate for this type of problems. The following section will focus on the decision problem where the final decision is in charge of a single decision-maker. For this reason, we will not use these techniques.

Draft Version 1.0

Draft Version 1.0

## Part III

# DISTRIBUTION SYSTEMS



# 14

## Diagnostic Models

This section analyses distribution networks (also known as distribution systems). Distribution networks connect the nodes of a supply chain, that can have a worldwide extent. They involve the network infrastructure, i.e. roads, rails, water connections, air connection, pipelines and any other infrastructure needed to move goods from a point to another. Production network must be effective, by providing a fast and safe connection between customers and clients, and efficient since they add no value to the product transported.

Special attention must be paid in the design of the infrastructure of these networks. The capacity and the velocity of the connection make a country powerful and competitive in the international market. The design and control of the distribution on the network is a crucial issue as well. Decisions as the type of vehicle, or the frequency of a connection heavily affect the efficiency of the network.

Distribution systems are modelled using the graph theory which is well explored from a mathematical and computational point of view; for this reason, there is room for the application of graph-based data-driven models in the field of distributions science.

This chapter focuses on the definition of the keywords and key entities extending the ontology of chapter 3 to distribution systems. After that, it introduces the diagnostic framework for production nodes with a relational data structure. A non-relational data structure is introduced to overcome some limits of the relational data structure, enhancing data collection from multiple sources and more effective analysis to support the design and control of a distribution network.

Chapter 15 focuses on model-driven and data-driven approaches to control a distribution network. In contrast, chapter 16 does the same to support

the decisions on the design of a distribution network.

## 14.1 Ontology

Here we go deeper into the details of the general ontology introduced in chapter 3. This chapter applies that approach to a distribution network that, in general, can be considered the biggest logistic system possible, since it is globally distributed.

### Entities

We identify the following entities.

**Part (i):** A part is a handling unit (HU), i.e. the smallest part that is loaded and unloaded from a vehicle. The HU is usually the object of digital tracking in the supply chain. Depending on the type, and the extent of the network, a HU can be a carton (e.g. parcels supply chain), a pallet (e.g. supply chains with distribution centres), or even a container (e.g. globally distributed supply chain).

**Processing node (j):** The resources in charge of loading and unloading parts from vehicles. These entities are terminals, platform or bays, depending on the type of vehicle to load/unload (aircraft, vessels, trains, trucks, vans).

**Edge (j, k):** Any path connecting two processing nodes is an edge. Edges of a distribution network have a type, defining the type of vehicle able to travel on the edge (e.g. air, rail, water, road) and a capacity.

**Vehicle (v):** A vehicle is an element travelling on edges to transport HUs from a processing node to another.

**Consumable (s):** It models the energy or the fuel to operate vehicles.

**Route (e):** It is a predefined visiting sequence of different processing nodes.

**Order (o):** It is a transportation order received from a customer or generated by a freight forwarder.

**Job (b):** It is the schedule of the processing nodes to visit, including the expected service time windows and the detail of the HUs to load/unload.

**System network** The graph  $G(V, A)$  of nodes  $j \in V$  and edges  $(j, k) \in A$  composing the distribution network.

### Metrics

We identify the following metrics to assess the performances of a processing node  $j$ .

**Throughput ( $TH_j$ ):** The productivity of a resource, i.e. the average number of HUs loaded and unloaded per unit of time (e.g., containers per hour).

**Work in process ( $WIP_j$ ):** It is the number of parts stored in a location  $j$ .

**Work in process ( $WIP_{jk}$ ):** It is the number HUs (i.e., the level of inventory) waiting to be loaded on a vehicle  $v$ .

**Capacity ( $C_j$ ):** It is the upper bound of the throughput of a resource.

**Capacity ( $C_v$ ):** It is the maximum number of HUs transportable by a vehicle  $v$  at the same time.

**Utilisation ( $U_j$ ):** It is the average fraction of time that a resource is not idle for lack of vehicles to load/unload.

**Utilisation ( $U_v$ ):** It is the average fraction of non-empty space on a vehicle.

**Lead time ( $LT_e$ ):** It is the time allocated to serve a given route (i.e. from the beginning to the end).

**Cycle time ( $CT_e$ ):** It is the average time to serve a given route.

**Service level ( $SL_e$ ):**  $\text{Prob}\{\text{cycle time} \leq \text{lead time}\}$

### Information functions

Finally, we define the three information functions: Movements  $M$  are referred to load/unload of HUs on vehicles  $v$ ; inventories  $I$  are referred to the work in process on a terminal  $j$ , or a vehicle  $v$ . The productivity  $P$  refers to the inbound and outbound absorption rates of the terminal. Table 14.1 summarises the definition of the three functions in a distribution network.

Information function	Description
Movements $M_i^{j,v}(t)$	This function defines the loading and unloading of transportation units $i$ , on a vehicle $v$ , leaving from a terminal $j$ .
Inventory $I_j(t)$	This function defines the inventory level on a terminal $j$ .
Inventory $I_v(t)$	This function defines the work in process on a vehicle $v$ .
Productivity $P_j^{IN}(t^*)$	This function defines the inbound productivity (e.g. part per hour) of a terminal $j$ .
Productivity $P_j^{OUT}(t^*)$	This function defines the outbound productivity (e.g. part per hour) of a terminal $j$ .

Table 14.1: Definition of the information functions of a distribution network.

## 14.2 Data Structure

In the supply chain field, data collection developed together with the concept of traceability. Robust relational data structures rose to control the shipping processes and identify the status of HUs. For this reason, all the data collected in a distribution network revolves around the shipping information.

The bias of the data structure is shallow compared to other supply chain entities (e.g. production systems). HUs are similar, share similar vehicles, and they travel on a graph  $G$  defined by nodes and edges where often the arcs refer to a finite and well-known set of elements (roads, rail tracks, air tracks or waterways). We present a relational data structure to study many aspects of a distribution network from a research point of view, keeping in mind that the ER structure is the most widely used in the shipping companies. Together with that, an original non-relational structure is introduced to overcome the rigidity of the relational structure and to improve the level of information from different players on the supply chain.

### 14.2.1 A relational model for distribution networks

Organisational and industrial practice choose the entity-relationship (ER) model as the most used method to store and organise data. The outcome of this organisation is the well-known table structure of a SQL database. ER models use tables to describe the attributes of entities and define relationships to link entities that share the same attributes. Entities can be a part  $i$ , a vehicle  $v$  or resource  $j$  connected by relationships which describing a route  $e$ , or a job  $b$ . A SQL database provides many benefits in a production environment as:

- Low replication of data since tables are structured in the so-called normal form;
- Easy access to data. The SQL language allows programmers to get data just thinking of which data do they need and not how to fetch that data from the database;
- High data consistency; since relationships impose relational integrity which prevents from adding sparse data.

The ER model we propose is organised into three macro-areas of a distribution network aiming at providing a comprehensive framework to model a distribution network [1]:

1. A geographical level;
2. A logistic level;
3. An on-field level.

The entities and relationships belong to each of these areas.

## Geographical level

The geographical level aims at defining the characteristic of macro-areas of the world called *Geoareas* from a geopolitical perspective. A table *SocioEconomicProfile* identifies the statistics on the GDP, inflation and other similar indicators. The table *AgroProfile* tracks the characteristics of the soil while the table *ClimateProfile* tracks the weather and climate trends. The table *InfrastructureProfile* maps the transport infrastructure of a *Geoarea* while the table *LogisticProfile* contains aggregated logistics indicators as the import and export volumes. The tables *InfrastructureCostProfile* and *EnergyCostProfile* identify the average costs of the infrastructures (e.g. buildings, land, licences) and the price for consumables (e.g. electricity). A table *DemandGeoArea* has aggregated values of demand for clusters of similar products (e.g. fruits, vegetables, potatoes) within the *Geoarea*. Table 14.2 illustrates the details of the attributes for each table of the geographical level.

Table	Attributes
GeoArea	Country, Region, Province, Area, Latitude, Longitude, SocioEconomicProfile, AgroProfile, LogisticProfile, InfrastructureProfile, ClimateProfile, EnvironmentalProfile, InfrastructureCostProfile, EnergyCostProfile, LaborCostProfile
SocioEconomicProfile	GeoArea, period, AgricultureValueAdded, CasSurplus, CurrentAccountalance, CentralGovernmentDebt, ExportsOfGoodsAndServices, GDPperCapita, GDPGrowth, GrossCapitalFormation, ImportOfGoodAndServices, IndustryValueAdded, Inflation, PortFolioEquity, ServicesValueAdded, TradeInServices
AgroProfile	GeoArea, period, %AgriculturalRrigatedLand, %AgriculturalLand, AgriculturalValueAdded, AgriculturalValueAddedForWorker, ArableLand, EmploymentInAgriculture, ForestArea, ImprovedWaterSource, LandArea, PermanentCropland, LandUnderCereal, RuralPopulation, CerealYield, LivestockProductionIndex
LogisticProfile	GeoArea, period, FoodExport, FuelExport, TransportServices, TravelServices, AgriculturalRawMaterialImports, TariffRates, MerchandiseTrade, AbilityToTrackAndTraceConsignments, CompetenceAndQualityOfLogisticsServices, EaseOfArrangingCompetitivelyPricedShipment, TimeToExport, TimeToImport, EfficiencyOfCustomsClearanceProcess, FrequencyConsigneeWithExpectedTime
InfrastructureProfile	GeoArea, Period, AirTransportRegisteredCarriers, AnnualFreshWaterAgriculture, AnnualFreshWaterTotal, ContainerPortTraffic, ElectricPowerConsumption, ICTGoodsExport, ICTGoodsImport, InternetUsers, PassengerCars, RailLines, RenewableInternalFreshWater, RoadSectorFuelConsumption, RoadSectorEnergyConsumption, RoadsPaved, Vehicles
ClimateProfile	GeoArea, Code, Period, CO2Emissions, EnergyUse, MethaneEmissions, MortalityRateUnder5, NitrousOxideEmissions, OtherGreenhouseGasEmissions, PopulationGrowth, PopulationPrimaryCompletionRate, RatioOfGirlsToBoysInSchool, UranPopulation, AccessToElectricity, PovertyHeadcountRatio, PopulationInUrbanAgglomerationOfMoreThanOneMillion, LandAreaWhereElevationIsBelow5m
EnvironmentalProfile	GeoArea, Code, Period, AgriculturalMethaneEmissions, AgriculturalNitrousOxideEmissions, FishSpecies, MammalSpecies, CO2Emissions, ForestArea, MarineProtectedAreas, OrganicWaterPollutant, PlantSpecies, MethaneEmissions, NitrousOxideEmissions, OtherGreenhouseGasEmissions, WaterPollutionFoodIndustry, WaterPollutionMetalIndustry, WaterPollutionPaperAndPulpIndustry, WaterPollutionTextileIndustry, WaterPollutionOtherIndustry
InfrastructureCostProfile	GeoArea, Code, AgricultureCost, LicenseCost, ProductionBuildingCost, DistributionBuildingCost, EnergyInfrastructureCost, WoodlandCost
EnergyCostProfile	GeoArea, Code, KWhElectricityCostDomestic, KWhElectricityCostIndustrial, KWhGasCostIndustrial, KwhSolarCost, KwhWindCost, KwhCarbonCost, KwhGasCost, FuelCost
LaborCostProfile	GeoArea, Code, OperatorCost, FarmerCost, WorkerCost, WarehouseWorkerCost, EmployeeCost, DriverCost
DemandGeoArea	GeoArea, Period, Product, Demand, ChannelGDO, ChannelGroceryMarket, ChannelGroceryShop, ChannelGroceryMarket
Expense	GeoArea, Period, Product, ExpensePerFamily
ImportRules	GeoAreaImporter, GeoAreaExporter, NationalTax, Import rules
ExportRules	GeoAreaImporter, GeoAreaExporter, Export rules

Table 14.2: Tables and attributes of the geographical level.

## Logistic level

The logistic level contains all the entities to populate the ontology presented in 14.1. A table *node* identifies all the processing nodes of the network connected by the edges whose material flow is mapped in the table *Flow*. Each row of the table *Node* identifies a processing node of a specific type. Depending on the node type, the tables *Crop*, *ProcessingPlant*, *StorageFacility*,

*Market*, *Port*, *Terminal*, *MultimodalHub*, and *RailConnection* identify the characteristics of the node. Table 14.3 illustrates the details of the attributes for each table of the logistic level.

Table	Attributes
ImportExport	GeoAreaCode, ImportExportProfile, ProductCode, Period, TypeProduct, ImportFlow, InternalDemand, InternalProduction, ExportFlow
Supply	NodeCode, ProductCode, Period, NodeType, Country, ProductionQty, ProductionCost, PackagingCost, StorageCapacity, ProductionArea, LandfillCost, ProductionEmissions, SupplierEmissions, LandfillEmissions
Flow	NodeCodeFrom, NodeCodeTo, FlowCount, GeoAreaFrom, GeoAreaTo, WeeklyConnections, Lines, Product, NominalCapacity, MaxCapacity, OverallFlow, LiquidBulk, DryBulk, AgroLiquidBulk, AgroDryBulk, GeneralCargo
Node	NodeCode, NodeType, NodeName, Enterprise, ZipCode, Address, City, Province, Nation, Region, Continent, roadAccess, RailAccess, SeaAccess, AirAccess, StorageCapacity, HandlingCapacity, ThroughputCapacity, Power, EnergyRequired, WaterRequired, CO2, Turnover, WHProfile, LayoutProfile, PlantProfile
Port	NodeCode, NodeName, NodeType, City, Province, Country, Region, Continent, TotalArea, RailLength, QuayLength, Cranes, Employees, Vessels, Capacity, Throughput, MaxDraught, StorageArea, TotalTerminals, Cruises, DryBulk, LiquidBulk, GeneralCargo, Oilpipe, DryAgroBulk, LiquidAgroBulk, TransshipmentFraction, EUNorthernArea, EUSouthernArea, MediterraneanArea, ConnectionRail, ConnectionRoad, ConnectionMaritime
Terminal	NodeCode, PortCode, TerminalCode, Type, Name, ProductCategory, Throughput, Period, Cranes, StraddleCarriers, Forklift, PortTruck, PipePlugs, feerePlugs, VesselBerths, RailTerminal, ReachStackers, VesselCapacity, Area, StorageCapacity, throughputRate, LoadingRate, UnloadingRate, TemperatureStage, CoolStorage, FrozenStorage
Multimodal Hub	NodeCode, NodeName, NodeType, City, TotalArea, RailLength, Employees, StorageArea, StorageAreaTemp, TotalTerminals, DryBulk, LiquidBulk, GeneralCargo, OilPipe, DryAgroBulk, EUNorthernArea, EUSouthernArea, MediterraneanArea, ConnectionRail, ConnectionRoad, ConnectionMaritime, FixedCranes, MobileCranes, FloatingCranes
Rail Connection	ConnectionCode, Destination, enterprise, Frequency, TravellHours, NodeCode, NodeName

Table 14.3: Tables and attributes of the logistic level.

### On-field level

The on-field level deals with the track&trace world collecting the shipping logs and all the related information. The table order contains all the shipping orders (i.e. provisional data) from a processing node  $j$  to a processing node  $k$ . The table shipment contains all the shipping jobs (i.e. execution data) of the shipping orders from  $j$  to  $k$ . The table product map is the item master file containing all the details (e.g. code, size, volume, weight) of the items. A product mapped in this table is equal to the definition of a part in a production node (see section 20.2). A table shelflife identifies the quality decay rates of each item. The tables package and unitLoad maps all the features of the handling units (HU). A table *ClimateProfileMonitoring* contains the temperature logs of shipping. Table 14.4 illustrates the details of the attributes for each table of the on-field level.

Figure 14.1 presents the resulting ER structure with the tables belonging to the three levels.

#### 14.2.2 A non-relational model for distribution networks

This section presents a non-relational data structure able to embed the same information content of the ER model introduced in 14.2.1 but able to host any other additional attribute. Besides, the model has a minimum number of required attributes that enables define a minimum viable model (MVM). The MVM contains a number of mandatory attributes for each class of the model allowing analysis of a supply chain network. The classes

Table	Attributes
Order	OrderCode, Category, ShipmentCode, NodeCode, NodeCodeTo, DeliveryDate, PdkQuantity, PdkWeight, PalletQuantity, Parts, TransportationModeCode, Load, Description, ShipmentCode, UnitLoadCode
Product	ProductCode, ScientificName, FoodCode, Category, Name, Variety, Brand, Enterprise, DOP, IGP, IGT, DOC, DOCG, Energy, Edibility, Water, Protein, Lipid, Cholesterol, Carbohydrates, Amid, Sugar, Fibre, Alcohol, Sodium, Potassium, Iron, Calcium, Selenium, Tiamin, Riboflavin, VitaminA, VitaminB, VitaminC, VitaminE, EthyleneProduction, EthyleneSensitivity, ColdSensitivity, HarvestPeriod, Certification, OptimalStorageTemperature, OptimalStorageRH, MaximumShelflife, PackageCode
DemandNode	ProductCode, NodeCode, Period, Quantity, Udm
Shelflife	ProductCode, TempMin, TempMax, Name, Type, RhMin, RhMax, ShelfLifeMin, ShelfLifeMax
Price	ProductCode, Node, Period, Type, ProductName, Variety, Country, region, City, PriceCrossMarket, Udm, Week, BeginDate, EndDate, Month, Market, ShopVariance, DiscountVariance, LargeRetailerVariance, PublicMarketVariance, RetailerVariance
Shipment	TransportationCode, ShipmentCode, NodeName, NodeCode, NodeCodeTo, ProducerCode, ShipperCode, CarrierCode, ImporterCode, CustomerCode, ContainerType, ContainerSize, ProductCode, CartonQty, UnitLoadQty, OriginPort, DestinationPort, DepartureDate, ArrivalDate
TransportationMode	TransportationCode, TransportationName, TransportationType, VehicleName, Description, Class, LicenceRequired, VehicleLength, VehicleWidth, LoadHeight, WeightCapacity, GrossTonnage, Co, He, Nox, Pm, Ch4, N2O, Nh3, SO2, CO2, CO2Eq, FuelConsumption, EnergyConsumption, TransportationCost, ContainerQuantity, VehicleQty, AvgSpeed, MaxDistance
Package	PackageCode, ProductCode, PackageName, PackageType, PackageLevel, Firm, Shape, ProductWeight, PackageLength, PackageWidth, PackageHeight, Udm, Cork, Sleever, Material, color, Weight, Barcode, PackagePrimary
UnitLoad	UnitLoadCode, UnitLoadLength, UnitLoadWidth, UnitLoadHeight, unitLoadWeight, Material
Enterprise	EnterpriseCode, NodeName, Group, Enterprise, NodeType, Importer, Exporter, Grower, Distributor, Address, Zipcode, City, Province, Region, Nation, Continent
ShipmentProfile	ShipmentCode, Ibutton, DatePeriod, Position, Temperature, ThermalKit, ContainerCode, Sampling, ShipmentCode, NodeName, NodeCode, NodeCodeTo, TransportationCode, EnterpriseCode
Analysis	SimulationCode, ShipmentCode, Product, TotalSulphurousDioxide, FreeSulphurousDioxide, OpticDensity420nm, TotalAcidity, VolatileAcidity, PhenolicAcid, SO2Free, SO2Total, ColorIntensity, SmellIntensity, TasteIntensity, Shipment, Ibutton, DatePeriod
Simulation	SimulationCode, ShipmentCode, Product, SimStart, SimEnd, SamplingPeriod, SamplingTime, TempOffset, RhOffset, TempCurrent, RhCurrent, Shipment, Ibutton, DatePeriod

Table 14.4: Tables and attributes of the on-field level.

of the non-relational model are the natural implementation of the MIP model introduced in Section 3.3.1.

A non-relational data-structure is usually stored in a computer using a JSON or XML notation. Non-relational structures are widely used in web application with databases as MongoDB since they allow to store tons of data in an (apparently) unstructured way without the prior definition of any table or relationship. Each record of the database is a *document* of a *collection*. These characteristics lead to many benefits as:

- the high flexibility of the data structure, since it is possible to load data with or without attribute already stored in a collection;
- the fast data fetching due to a leaner structure compared to SQL databases;
- the scalability of the data structure since the performance of the database is not directly related to its size.

These benefits come with some important limitations:

- many data may be replicated, leading to data storage inefficiencies. Due to the lack of relationships, information can be replicated in many documents with inefficiencies in the use of the storage space;
- join operations are not allowed. When a join operation is needed, it is necessary to perform it externally and without query optimisation;

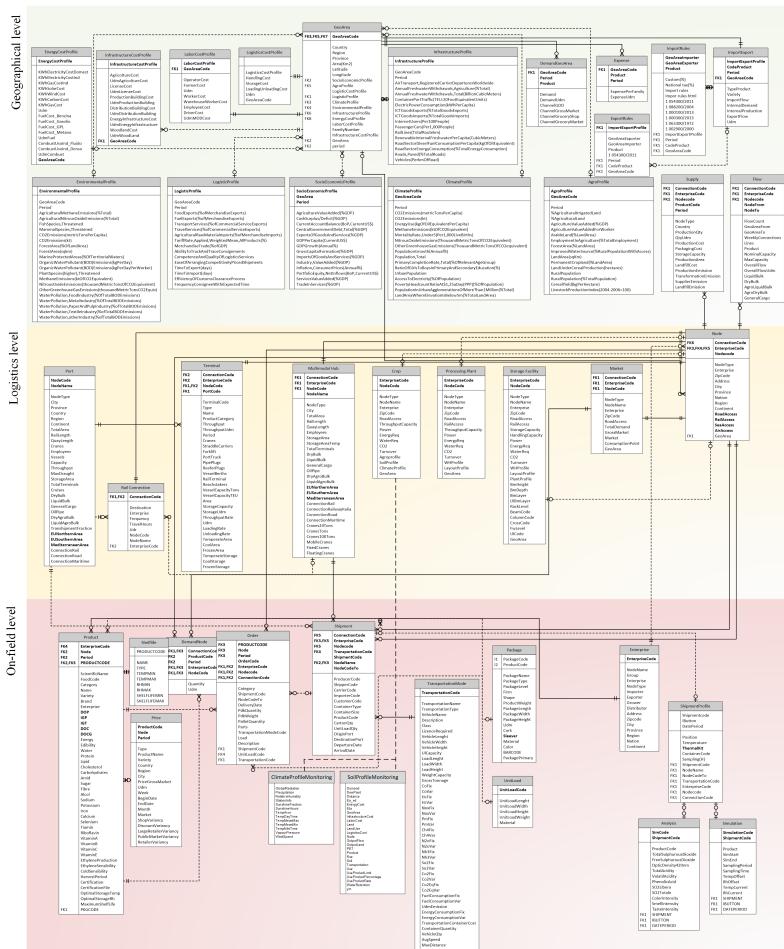


Figure 14.1: ER diagram of the relational data struvture for distribution networks.

- data are not as consistent as with an ER model; Since relations does not exist, it is impossible to store consistent data. Data consistency check must be performed outside of the database.

The model consists of three collections whose data are recorded by the transportation management system (TMS).

A collection *handling units* identifies all the parts  $i$  transported on the network. The code of the handling unit is the only attribute necessary to define the MVM. Other attributes can be stored as well, like:

- the description
- the size,
- the weight,
- the dimension,
- the type of package (e.g. primary, secondary, tertiary).

A collection *movement* defines the movement function of the MIP model, storing information on when a part  $i$  is loaded or discharged by a vehicle  $v$ . The timestamp and the quantity are the attributes needed for the definition of the MVM. Other features can be recorded, as for example:

- the id of the loading node,
- the id of the discharging node,
- the latitude and longitude of the loading node,
- the latitude and longitude of the discharging node,
- the provisional loading time window,
- the actual loading time window,
- the provisional discharging time window,
- the actual discharging time window,
- the id of the vehicle,
- the id of the voyage.

A collection *node* stored the information of all the suppliers and customers of the supply chain network. The id of the node is enough to define the MVM. It is possible to record a attributes as:

- the description,
- the latitude and longitude of a node,
- the address of a node,
- the type of the node (i.e. supplier/customer),
- the list of the material flows exchanges every year with a given node (e.g. the production plant serving the distribution network).

This non-relational structure is used in the rest of this section to support model and analysis on production systems. Figure 14.2 uses the unified modelling language (UML) to represent the MVM of the non-relational data structure of a distribution network.

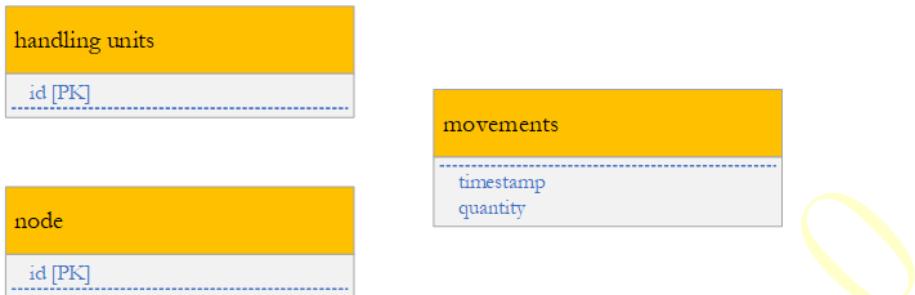


Figure 14.2: ER diagram of the relational data structure for distribution networks.

### 14.3 Decision patterns

This section aims at defining the set of decision patterns for the design and control of a distribution system, according to the definitions of section 4.2. The problems of a distribution system are classified into:

1. Network design problems, dealing with the design of the network, given the existent physical infrastructure;
2. Network control problems, dealing with the assessment and improvement of the performance of an existing distribution network.

Different methodologies allow getting feasible solutions to these problems. Table 14.5 illustrates the entities and their definition according to the ontology in Paragraph 3.1.

Draft

System	Class of the problem	Decision	Task	Entity	Data Science		Decision Science
					Descriptive	Predictive	
Distribution Network	P6 - Placement Problem	Design	Location-allocation problem	Points of demand	Processing node	•	○
Distribution Network	P2 - Assignment problem	Design	Network topology design	Part, vehicle	○	●	○
Distribution Network	P4 - Frequency design	Design	Route frequency design	Air Rail Road Water connections	Edges	○	●
Distribution Network	P5 - Power problem	Design	Service-time windows design	Terminal	Processing node	○	●
Distribution Network	P7 - Dispatching rules	Design	Shipping priority definition	Transportation unit (carton/pallet/containers)	Part	○	●
Distribution Network	P8 - Performance assessment	Control	Performance assessment	Distribution network	System Network, Parts, Resources, Vehicles, Jobs, Routes	●	○
					Transportation unit (carton/pallet/containers)	Part	●
Distribution Network	P9 - Workload prediction	Control	Workload forecast	Vehicle	Vehicle	○	○
Distribution Network	P2 - Technology Assignment problem	Control	Vehicle choice (Synchronicity)	Vehicle	Vehicle, Part	●	●
Distribution Network	P10 - Operations management	Control	Vehicle routing	Vehicle	Vehicle, Part	○	●

Table 14.5: Decision problems classification in a distribution system.

Seven decision problems are identified in the design and control of a storage node:

1. Network design; it is a covering problem where each point node of the network needs at least one route serving it.
2. Route frequency design; it involved the definition of the frequency of the service on a given route.
3. Service time windows design; it is the definition of the placement and the time span of the time windows to serve each node  $j$ .
4. Performance assessment (control); it involves the measurement of the performance of a distribution system.
5. Workload forecast (control); it involves the prediction of the workload and workforce needed to perform the operation in the short-, mid- or long-term.
6. Vehicle choice; it is the choice of the right vehicle (or the right type of vehicle in case of synchromodality) to perform a route.
7. Vehicle routing (control); it involves the definition of the scheduled routes and their assignment to vehicles.

While descriptive and prescriptive techniques are preferred for control problems, explorative and prescriptive techniques result adequate when dealing with design problems. Chapters 15 and 16 illustrate these techniques in details.

## Bibliography

- [1] R. Accorsi, S. Cholette, R. Manzini, and A. Tufano, “A hierarchical data architecture for sustainable food supply chain management and planning,” *Journal of Cleaner Production*, 2018.

# 15

## Distribution System Control

This chapter focuses on the problems and decisions involving the control of a distribution network. All the problems defined in this chapter deal with the presence of different actors and stakeholders using different systems and with a low willingness to cooperate. For this reason, the methods must take into account the incompleteness, inaccuracy and disharmony of the data collected from different systems.

### 15.1 Introduction

#### 15.1.1 The actors of a supply chain network

Supply chains connect stakeholders and resources providing services for the production, storage and transportation of goods. In the case of global supply chains, the stakeholders own resources with worldwide extent. The global logistics market has an estimated value of 1171 billions \$, which yearly increases by about 7% [1]. The demand for logistics connection increases as well as the number of companies working in this market.

A supply chain is complex since it involves actors, entities and procedures that are not synchronised or coordinated. When the demand increases and the scale is global, supply chains become even more complex, and the lack of coordination among the actors leads to inefficiencies in the management of the logistics flows. Supply chain stakeholders get revenues by managing three types of value flows [2]:

1. The physical flow of goods (e.g., between buyers and vendors);
2. The information flow needed to track the shipping state and the position of goods;

3. The flow of money that pays the activities performed by each stakeholder.

While thinking about these flows, it is easy to understand the complexity of managing all of them in a globally distributed supply chain. Carriers, shippers and freight forwarders make use of their practices for handling the physical flows, tracking shipping information and may have different currencies and protocols for the management of the flow of money.

Market trends show the increasing awareness of companies on the value of their data [3]. They started using new technologies like analytics, big data and internet of things devices [4] to improve the knowledge of the processes and increase their efficiency [5]. The information flow contains data whose content is not only intended to track a physical flow. Data embeds the value for making decisions about the design of the supply chain itself [6]. In the era of big data, the number of records available from a logistics information flow increases dramatically with obvious obstacles and costs to maintain the IT infrastructures. There is an unexplored potential of learning from historical data to design efficient operations in the future. The stakeholders usually involved in a generic distribution supply chain are:

- **producers**, are any production or storage system connected to a processing node  $j$  (i.e. a terminal) by road, rail or inland waterways;
- **freight forwarders**, are the operators in charge of organising door-to-door shipping. They get transportation orders from the producers, and they place them on one or many routes performed by one or more vehicles  $v$  to reach the destination;
- **operators (carriers)**, get transportation orders of handling units from the freight forwarders, and they organise the single transportation routes from terminals to terminals.
- **conveyance operators**, perform transportation by driving vehicles according to its route schedule;
- **terminal operators**, perform loading and unloading of handling units at the processing nodes  $j$ . They use physical equipment as loading bays, cranes, and forklifts;
- **customers**, receive the goods at the end of the transport operations.

Figure 15.1 introduces these stakeholders, their decision-making activities, and the logistic flows.

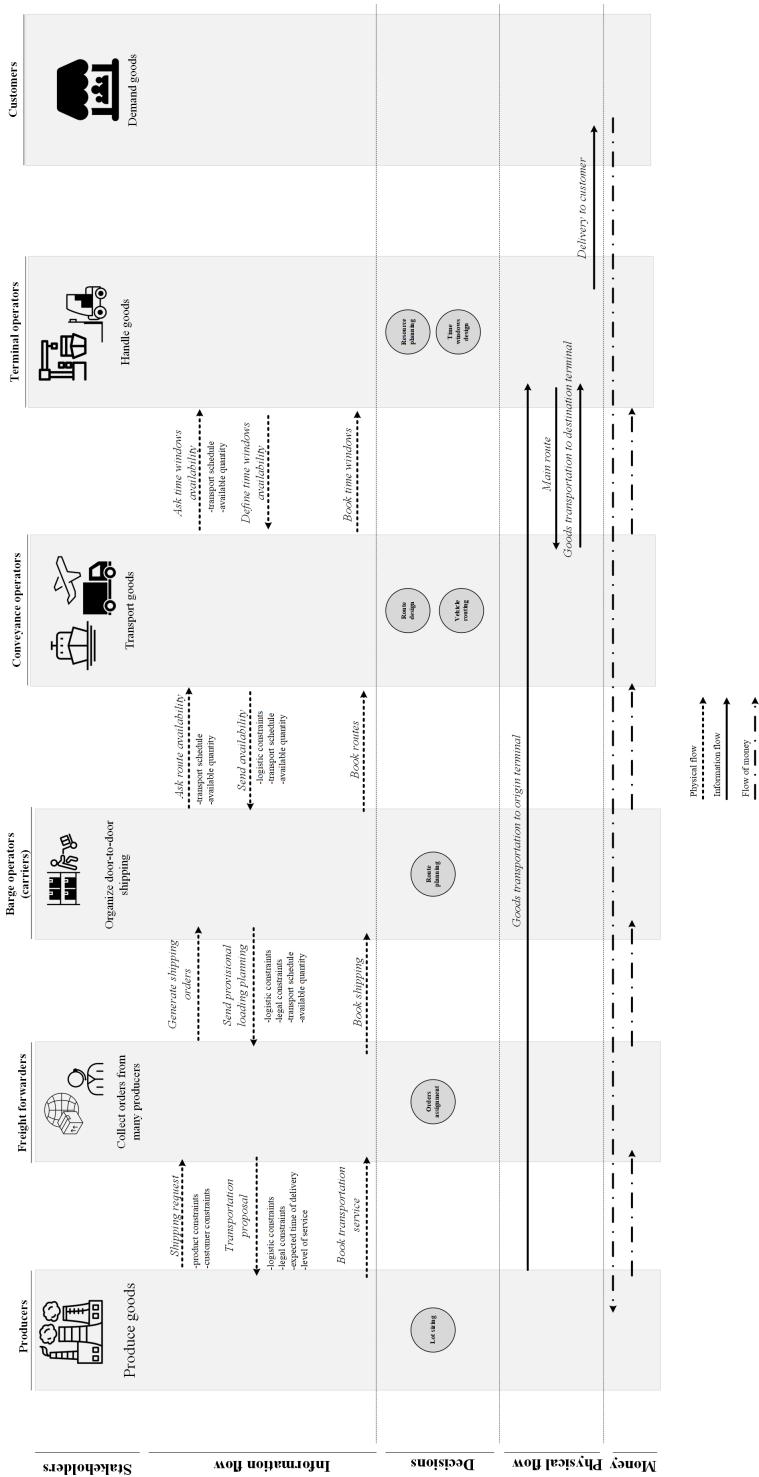


Figure 15.1: Stakeholders, logistic flows and related decisions.

As Figure 15.1 shows, coordination among different actors is a crucial issue to control the operations of a distribution network. The figure remarks the constraints imposed by each actor to the following one depicting the complexity of the system. In practice, a supply chain network is such constrained that can be thought as a system in unstable equilibrium where each actor found its equilibrium within the constraints imposed by the other. Unless a big player has the authority to change these constraints, it is difficult to improve the performance of the supply chain, and it is even challenging to collect the data to control it. During the last few years, the platform economy opened for new opportunities also in this field.

### 15.1.2 Logistics platforms

The advent of new technologies as the internet of things, artificial intelligence and fast internet connections paved the way for the so-called “platform revolution”, i.e. the establishment of new business models based on the idea of a platform connecting stakeholders [7]. Often, the economic edge of a platform is based on the concept of arbitrage, since platforms can link stakeholders in different ways compared to the established companies in the market [8, 9]. Platforms started by connecting the end-user to the supplier of a service. It is the case of platforms offering booking services for hotels, taxi or flight that started by offering a link between the supplier and the end-user and consequently started developing complex algorithms based on the collected data to improve their services and their edges [10, 11]. The value of a platform on the market depends on the amount of data it can collect [11]. Recently, platforms started to operate in the business-to-business (B2B) market, too [12]. In the B2B market, platforms connect companies using different data protocols and complex system by using interfaces and data exchange protocols [13].

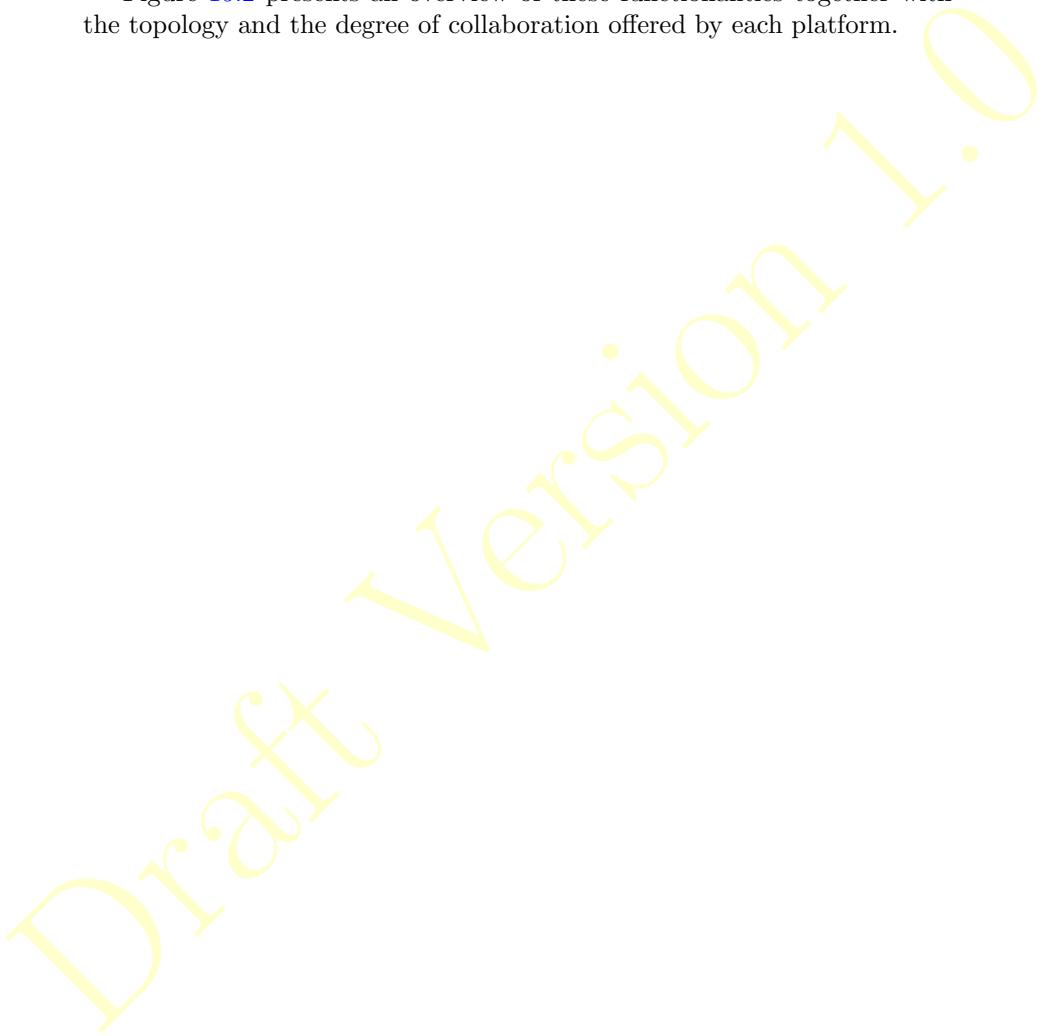
The market of logistics has seen a prolific increase in the development and use of logistic platforms during the last decade. A logistic platform is an IT infrastructure to store data and support storage, handling and transport operations across multiple stakeholders.

Literature classifies logistic platforms into four classes depending on the functionality they offer [14]:

1. Electronic Data Processing (EDP) platforms; they track the logistic processes storing data into relational databases.
2. Enterprise Resource Planning (ERP); they integrate all the activities of a stakeholder providing insights and algorithms to improve its performance.
3. Enterprise Application Integration (EAI); they integrate the activities of a subset of stakeholders enhancing collaborative planning.

4. Smart Contracts (SC); they fully integrate the tracking of goods processed by the stakeholders of a supply chain providing full visibility on the data and allowing for automatic execution of contracts (e.g., replenishment orders).

Figure 15.2 presents an overview of these functionalities together with the topology and the degree of collaboration offered by each platform.



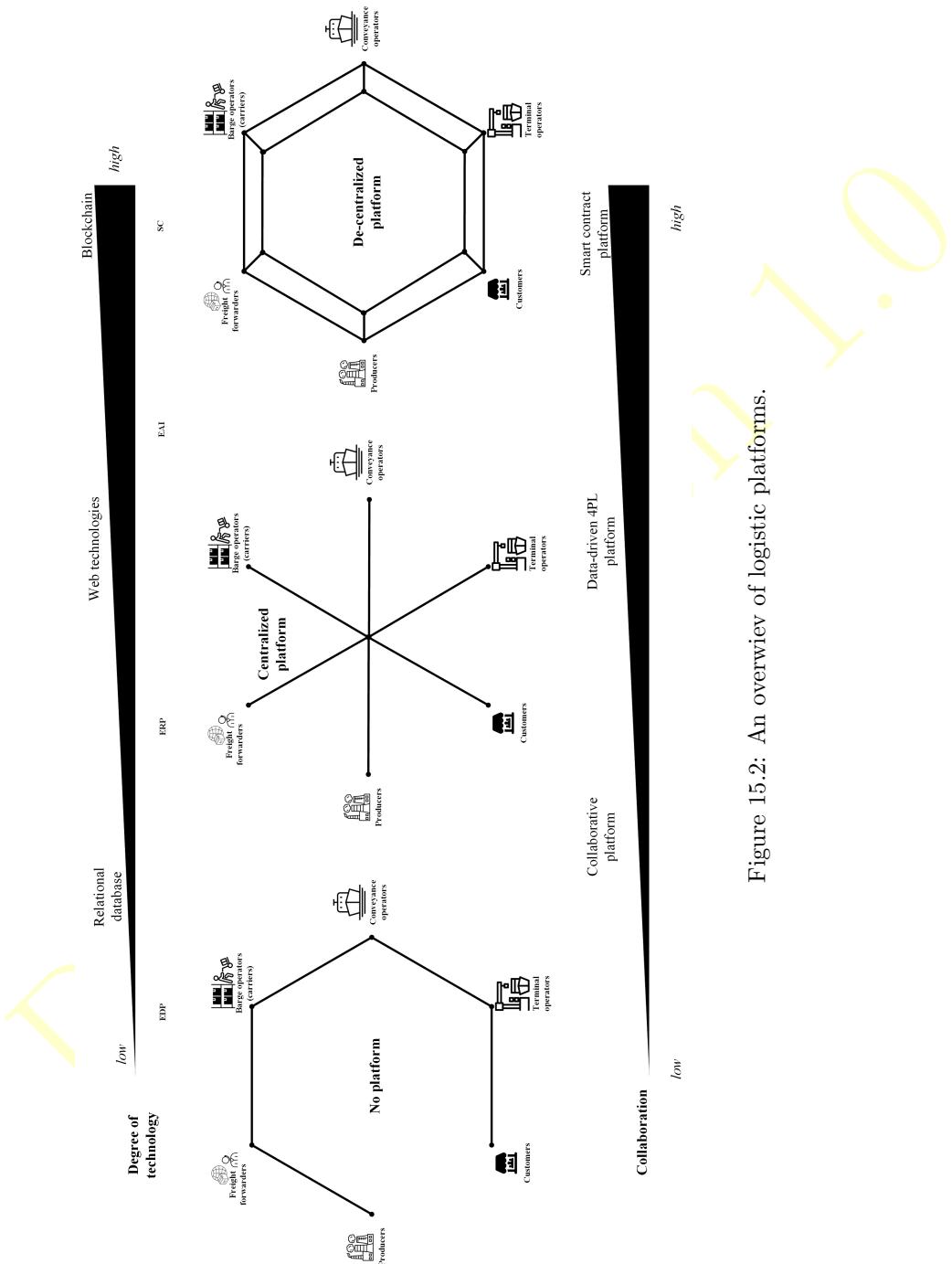


Figure 15.2: An overview of logistic platforms.

The topology of logistic platforms changes from a no-platform topology, where data is manually exchanged by the use of telephone, emails and fax [15, 16], to centralised platforms where a central entity guarantees the validity of data and transactions between stakeholders. The evolution of this topology is the de-centralised platform where no entity has full control on the platform, but it is distributed among all the stakeholders.

De-centralised platforms allow full collaborations between stakeholders using smart contract: each of them has full visibility on the data exchanged by the others, and this provides the highest amount of information for decision making.

When stakeholders are not willing to completely share data among them, centralised platforms using incomplete information result adequate. It is the case of collaborative platforms where a low number of stakeholders (typically two) decide to exchange some data to improve their performance. The data-driven 4PL platform is a more complex solution that collects data from many stakeholders providing them with functionalities and services based on big data (impossible to deliver basing only on the data of a few of them). These platforms maintain confidentiality on the data since they are not shared with the other users of the platform.

The interest of the scholars on the design and development of logistic platforms increased exponentially during the last decades (see Figure 15.3).

The role of a logistics platform stands in its ability to collect and merge data from different actors and use them to functionalities to them as:

- New pricing models [17, 18, 19];
- Synchromodality [20];
- Process integration [21];
- Estimated time of arrival forecast [22];
- Collaborative forecasting and replenishment [23];
- Possibility of last-minute planning [24];
- Real-time booking (arbitrage) by prediction of the available capacity (this paper);
- Digitisation of the processes;
- More visibility on the market.

All these methods are data-driven and rely on an incomplete but broad data structure. All the data-driven methods presented in the following paragraphs are intended for a logistic platform using a non-relational structure to merge data from different actors of the supply chain.

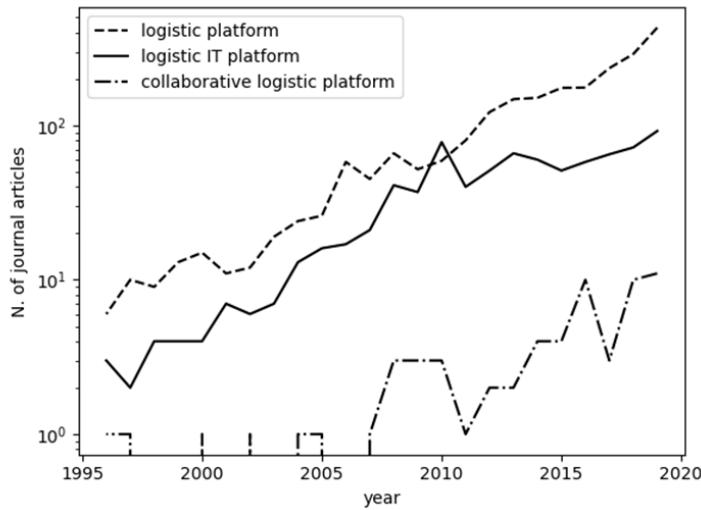


Figure 15.3: Number of journal articles in logarithmic scale obtained using “logistic platform”, “logistic IT platform”, and “collaborative logistic platform” as research queries on Scopus.

### 15.1.3 The day-by-day of a distribution network

Before diving into the math of problems and methods, it is important to understand the operations of a distribution network. For this reason, we introduce some additional details keeping, for example, a port supply chain, one of the most complex types of supply chains. This choice is due to the fact that a port connects thousands of destinations globally distributed using any type of equipment and vehicle to transport any type of good.

A port receives physical flows of many types. It always has road and rail connection, at least. Commonly, a port collects goods from inland waterways, and it may be close to an airport collecting air flows. Terminals  $j$  process all these flows using storage areas and handling units. Usually, inland terminals pre-process many flows from road and air by consolidating them on barges. Barges, then, reach the deepsea terminal where cargo vessels are loaded and unloaded. Cargo vessels operate:

- liner shipping, when routes are predetermined, and the schedule is fixed;
- tramp shipping, when shipping is operated on-demand (similarly to charter flights).

The complex information flows for booking and tracking is managed by freight forwarders, shipping company and shipbrokers. Figure 15.4 illustrates the logistic and the information flow of a port supply chain.

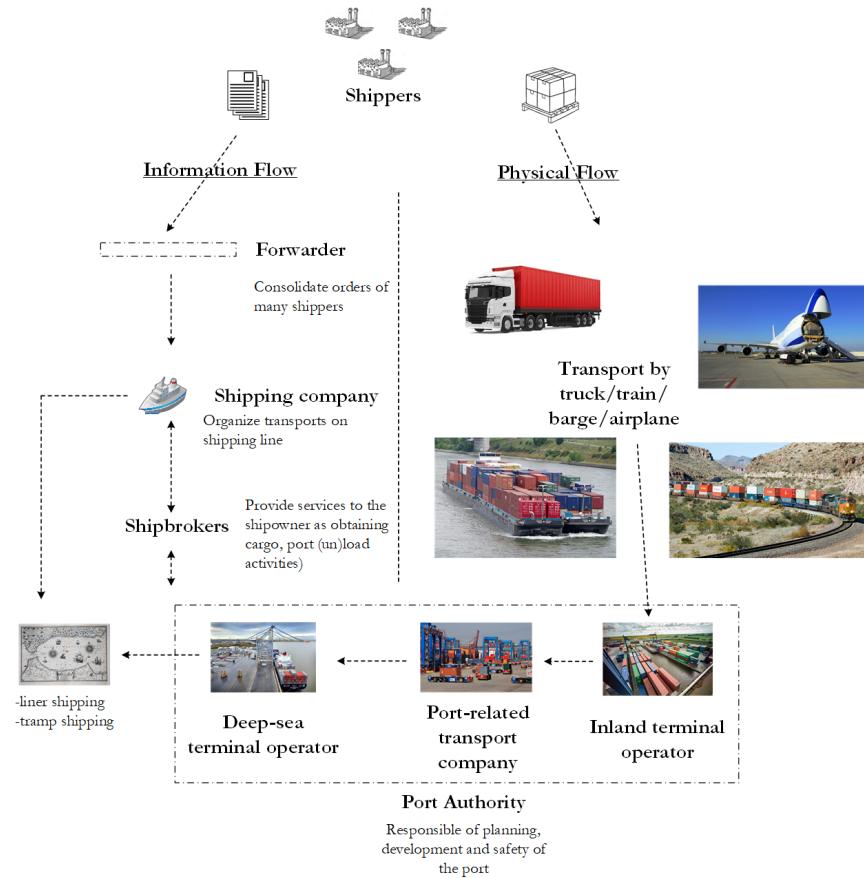


Figure 15.4: Physical and information flow of a port supply chain.

Different types of cargo vessels exist, depending on the type of handling unit they transport. Bulk cargo transport dry or liquid materials as coal, grain or crude oil. Breakbulk cargos are used for steel, wood or other voluminous goods that cannot fit a container. Container cargo vessels load TEU and FEU containers as transport units. Roll-on/off cargo vessels transport vehicles (car and trucks) that drive to enter or exit the vessel. Finally, project cargos are organised for oversize loads. Figure 15.5 illustrates images of the vessels used with different types of products.

### Dry Bulk Cargo



- coal
- iron
- ore
- grain

### Liquid Bulk Cargo



- crude oil
- refined oil

### Break Bulk (general cargo)



- steel
- metals
- papers
- wood
- fruit

### Container Cargo



- TEU
- FEU

### Roll-on/roll-off Cargo



- cars
- trucks
- trailers

### Project Cargo



- special transport (Yachts, ship's engines, wind turbines...)

Figure 15.5: Different types of cargo vessels.

Loading and unloading different types of cargo vessels need a different type of equipment. For this reason, port terminals are specialised on specific types of cargo vessels and specific type of products. Figure 15.6 illustrates the typical equipment of a terminal working with container cargo vessels.

Other supply chain nodes as airport and multimodal hubs are organised similarly. Generally, a logistic hub consists of multiple terminals (generally belonging to different companies) specialised in handling a specific type of product. Terminal companies own the equipment to load and unload cargos and connect a multitude of actors by using freight forwarders and brokers to manage the flow of money and information.

### Vessel loading/un-loading



Ship-to-shore gantry crane

Conventional shore crane (quay crane)

Mobile crane

### Container yard to berth



Truck chassis

Shuttle carrier

Straddle carrier

Automated guided vehicle

### Container yard operations



Top loader / reach stacker

Straddle carrier

Rubber tire gantry

Rail-mounted gantry

Automated stacking crane

Figure 15.6: Examples of the equipment and physical assets of a port terminal.

## 15.2 Performance assessment (P8)

This paragraph introduces models to assess the performance of a supply chain network from a model- and data-driven perspectives.

### 15.2.1 Model-driven methods (D4)

When the number of actors and flows is high and complex, qualitative business process mapping is always a good starting point to understand what is going on. A common technique is the Business Process Model and Notation (BPMN) (see section 5.1). It remarks the responsibility of each actor and their tasks (e.g. send an order, send confirmation, load container) remarking the activities triggered by each task. Applying the BPMN notation in a distribution network:

- activities are tasks necessary for the storage, handling or transportation of a handling unit;

- events identify when an activity is terminated (e.g. delivery at the point of consumption, loading completed at a terminal);
- gateways describe a different variant of the process (e.g. store in cross-dock or storage area of the terminal depending on the delivery due date).
- pools identify the actor of the supply chain in charge of an activity.

BPMN defines a qualitative map of the production processes useful for manager and practitioners to identify the way their processes are realised. More difficult is the assessment of these processes from a quantitative point of view. For this reason, a dashboard of KPIs is introduced, coherently with the ontology of 14.1. The KPIs used in these chapters refers to the problems defined in section 4.2. KPIs are organised according to four classes [25]:

1. Logistic KPIs, evaluate the logistic impact of a certain solution. They use metrics like time, distance and the performance parameters introduced in section 14.1.
2. Cost KPIs, evaluate the economic sustainability of a given solution. They are expressed in € or other currency.
3. Energy KPIs, evaluate the energy needed to feed a given solution. They use metrics as kW and kWh.
4. Environmental KPIs, evaluate the environmental impact of a given solution. They are expressing the equivalent  $CO_2$  produced per year.

Table 15.1 identifies which KPI is relevant to each problem. In general, each problem can be assessed from multiple perspectives.

System	Class of the problem	Decision	Task	Logistic KPIs	Cost KPIs	Energy KPIs	Environmental KPIs
Distribution Network	P1 - Family problem	Design	Network design (location-allocation problem)	WIPj, LTe	○	○	○
Distribution Network	P2 - Assignment problem	Design	Assignment of points of demand to routes	LTe, THj	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO2/year)
Distribution Network	P5 - Power problem	Design	Route frequency design	Uj, C, LOSe	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO2/year)
Distribution Network	P5 - Power problem	Design	Service time windows design	Uj, C, LOSe	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO2/year)
Distribution Network	P7 - Dispatching rules	Design	Shipping priority definition	WIPj, LTe, LOSe	Storage cost (€/year) Production costs (€/year)	○	○
Distribution Network	P8 - Performance assessment	Control	Performance assessment	CT <sub>b</sub> , U <sub>b</sub> , LOS <sub>b</sub>	Maintenance costs (€/year) Storage costs (€/year)	Required energy (kWh/year)	Environmental impact (CO2/year)
Distribution Network	P9 - Workload prediction	Control	Workload forecast	CT <sub>b</sub> , U <sub>b</sub> , LOS <sub>b</sub>	Direct labour cost (€/year)	○	○
Distribution Network	P2 - Technology Assignment problem	Control	Vehicle choice (Synchronomodality)	LTe, THj	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO2/year)
Distribution Network	P10 - Operations management	Control	Vehicle routing	CT <sub>b</sub> , U <sub>b</sub> , LOS <sub>b</sub>	○	○	○

Table 15.1: KPIs to evaluate the solutions to problems in a distribution network.

We show an application of a model-driven method to evaluate the impact of a food supply chain [26, 27]. The method relies on the definition of a relational data-structure containing information on:

- customer orders, defining the product code and the quantity to transport;
- products, including description, volumes, and weights;
- transportation units, defining size, capacity concerning the volumes and weights of the products;
- vehicles, defining size and capacity concerning the volumes and weights of the transportation units.
- impact, defining for each vehicle the cost, and the environmental impact KPIs (e.g.  $\frac{CO_2}{ton \times km}$ ).

In practice, the ER structure is organised with a Chinese boxes structure by defining for each vehicle, and for each transportation unit, how many products can be loaded. The model is based on customer orders, and it calculates:

1. The number of transportation units necessary to load all the products in the customer orders;
2. The number of vehicles necessary to transport the transportation units found at 1);
3. The distance travelled by each vehicle;
4. The overall cost and impact;
5. The cost and the impact of each vehicle;
6. The cost and the impact of each transport unit;
7. The cost and the impact of each product.

Figure 15.2 illustrates the entities and the KPIs involved in this model.

### 15.2.2 Data-driven methods (D1, D2)

The aforementioned model-driven approach is punctual and precise. Nevertheless, it relies on a massive amount of static data. All of them are necessary to perform the calculation of the KPIs. In addition, when a parameter is unknown, many hypotheses must be made to run the model, adding bias to the results. To avoid biased results and to expedite the data collection, we introduce a data-driven approach based only on the available data. This method assesses a distribution network from different points of view by considering only the available data, without additional assumptions. There are three macro-areas of analyses:

Entities	Customer Order	Shipping Order	Truck	Shipping	Product
Features		<ul style="list-style-type: none"> <li>- same destination</li> <li>- same due date</li> <li>- same shipping code</li> </ul>			
KPIs				<p>KPI per shipping: Environmental impact KPIs: co<sub>2</sub>, hc, nox, pm, ch4, n2o, nh3, so<sub>2</sub>, co2l, co2eq Energy KPIs: fuel, energy (kW) Logistics KPIs: km</p>	<p>KPI per product: Environmental impact KPIs: co<sub>2</sub>, hc, nox, pm, ch4, n2o, nh3, so<sub>2</sub>, co2, co2eq Energy KPIs: fuel, energy (kW) Logistics KPIs: km</p>

Table 15.2: Entities and KPIs of the model-driven approach.

1. the profiling of the actors of the supply chain;
2. the profiling of the operations of the supply chain;
3. the profiling of the geographical network of the supply chain.

The presence in the dataset of certain attributes allows the realisation of some analyses. We can think of attributes as keys and analysis as doors; the right keys unlock the right doors. Figure 15.7 illustrates the links between attributes and analysis (keys and doors), illustrated in details in the following paragraphs.

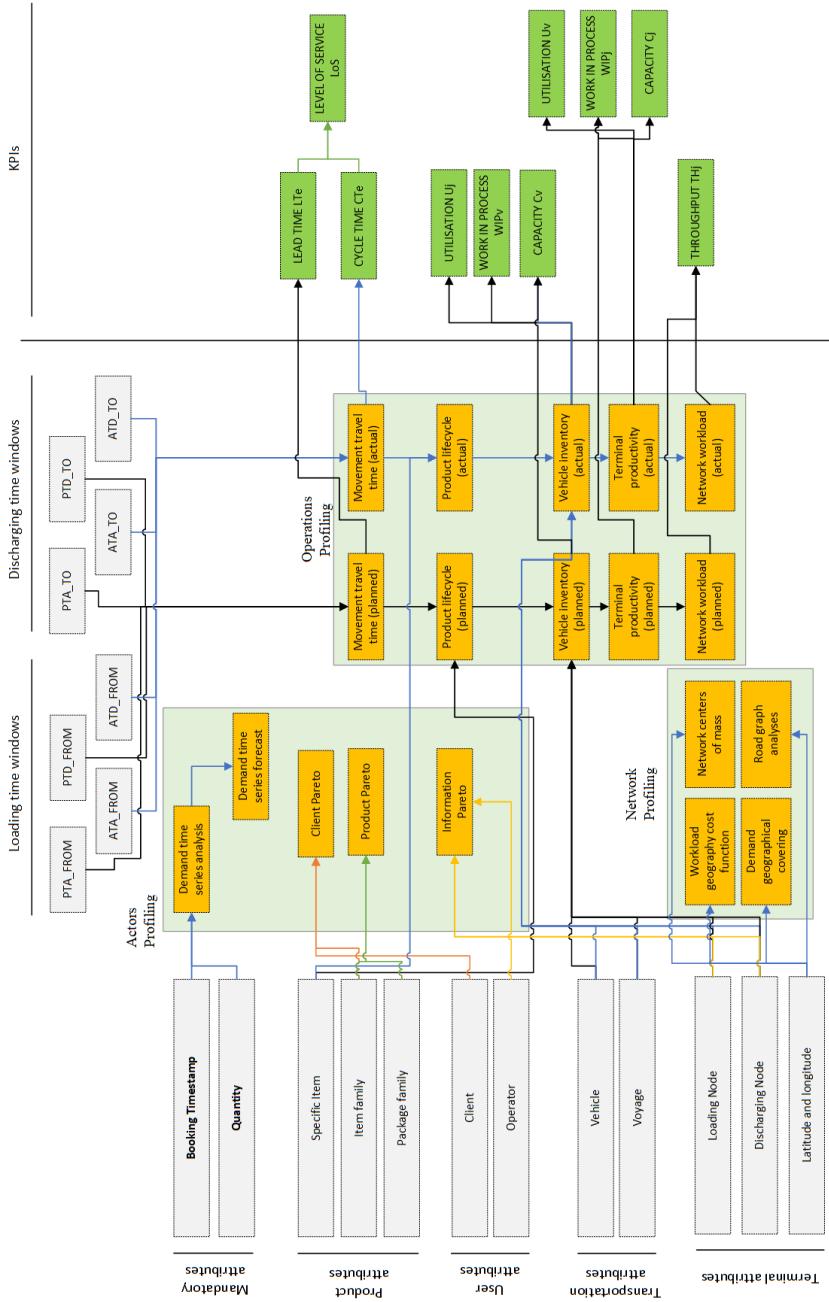


Figure 15.7: Connections between attributes, analyses and KPIs.

## Actors Profiling

**Demand time series analysis and forecast** To support our data-driven approach, we base on the definition of the MVM of the collection movement, defined in 14.1. The MVM prescribe two mandatory attributes:

- a timestamp describing when the movement is created (i.e. when an order arrives from a client);
- the quantity involved.

By using these two attributes, it is possible to analyse the demand time series and use this information to make forecasts (see section 15.3). Analysis should be performed on:

- the number of lines, i.e. the count of the timestamps;
- the quantities, i.e. the sum of the quantity processed at the same timestamps.

These two analyses are always relevant since the business of the companies can be line-oriented (e.g. it is the case of third party logistics) or quantity-oriented (e.g. for production industry). Timeseries must always be resampled using an aggregation function. The sampling interval depends on the amount of data collected and on the relevance of the analysis. Frequent sampling intervals are the day, the week or the month. Figure 15.8 illustrate the daily, weekly and monthly trends of the number of movement with their probability distributions. The bottom of Figure 15.8 shows the comparison between the lines and quantity trends, and a histogram of the number of movements per day of the week.<sup>1</sup>

These time series can be analysed, and decomposed to uncover trend and seasonality patterns. Figure 15.9 illustrates the decomposition of the weekly aggregated lines and quantities time series with the Fourier analysis.<sup>2</sup>

By considering the time series and all the available attributes of the dataset, it is possible to define a correlation matrix to uncover hidden patterns and behaviours of the network (see Figure 15.10).<sup>3</sup>

---

<sup>1</sup>The source code of Figure 15.8 is available [here](#).

<sup>2</sup>The source code of Figure 15.9 is available [here](#).

<sup>3</sup>The source code of Figure 15.10 is available [here](#).

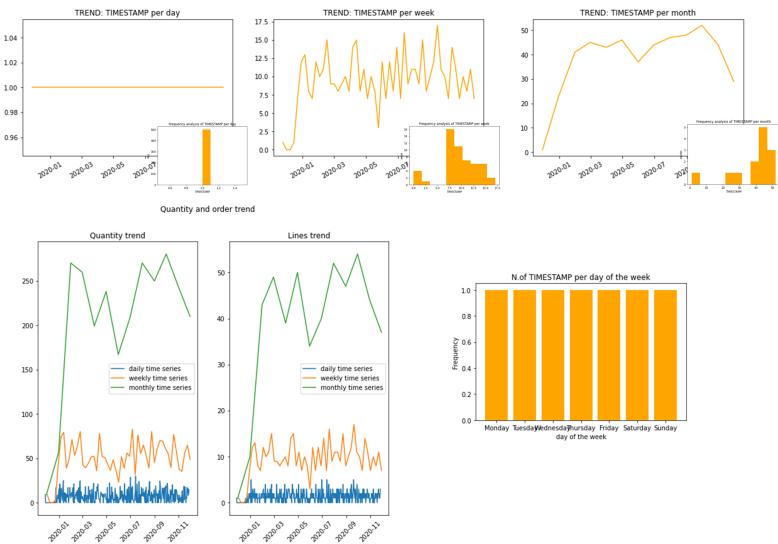


Figure 15.8: Time series and probability distributions of the movements of a distribution network, using different aggregation levels.

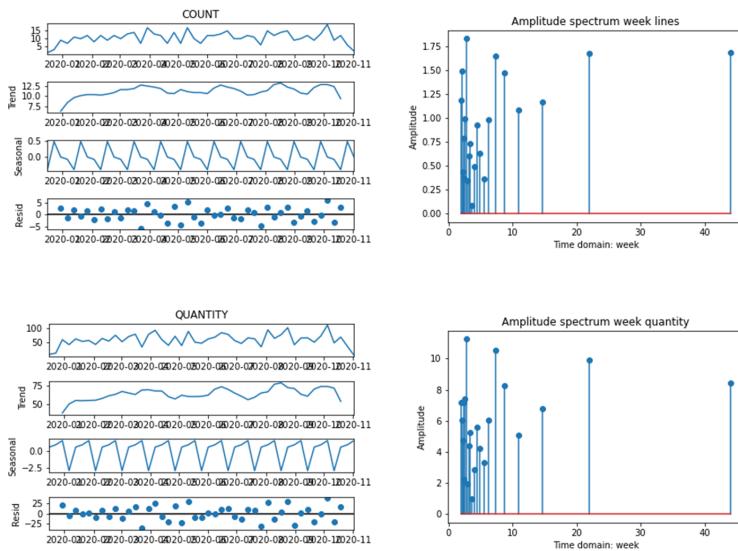


Figure 15.9: Time series decomposition, and Fourier analysis of the movements.

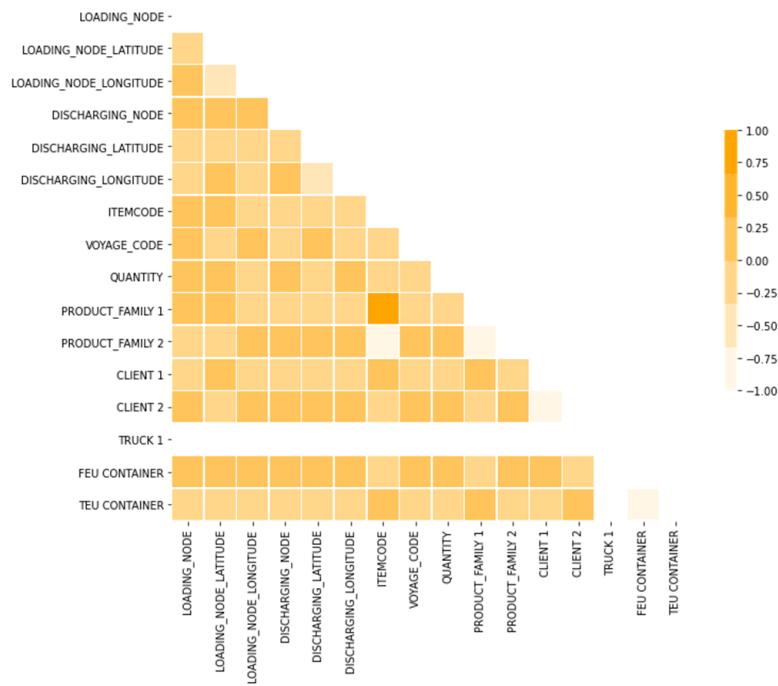


Figure 15.10: Correlation matrix of the movements dataset.

**Client Pareto** Time series analysis focuses on the workload. From another perspective, it is important to identify the source of the workload. Clients define the market demand and, almost always, their relevance follows the Pareto law: 20% of the customers generate 80% of the workload. Having information both on the quantities and the clients allows identifying the relative importance of each client. Figure 15.11 presents this information using a pie chart and a Pareto curve.<sup>4</sup>

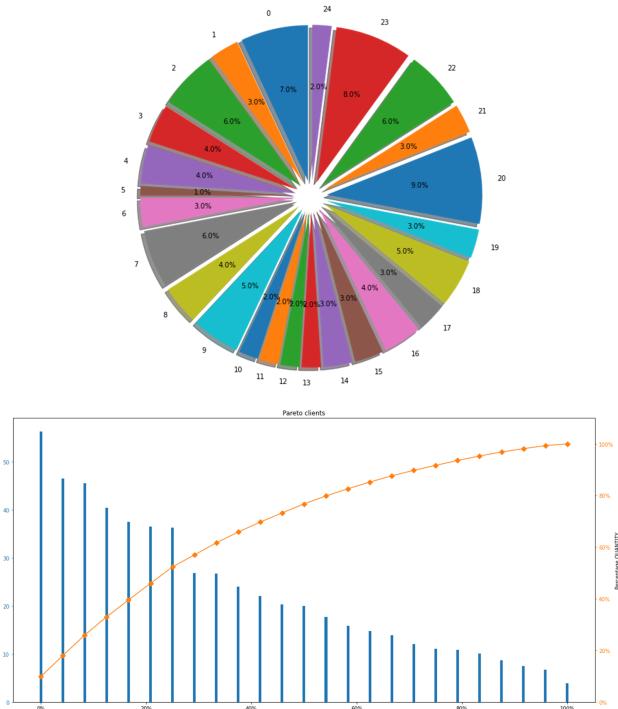


Figure 15.11: Pie chart and Pareto curve of a set of clients.

Similar information may be of interest when referred to a single terminal of the distribution network. Figure 15.12 illustrates the violin chart of the four most congested terminals of a distribution network. The chart identifies the demand (e.g. the quantity or the number of movements) of each client assigned to each terminal.<sup>5</sup>

**Product Pareto** If clients define the market demand, products define the offer. Again, it is common that 20% of the products realise 80% of sales

<sup>4</sup>The source code of Figure 15.11 is available [here](#).

<sup>5</sup>The source code of Figure 15.12 is available [here](#).

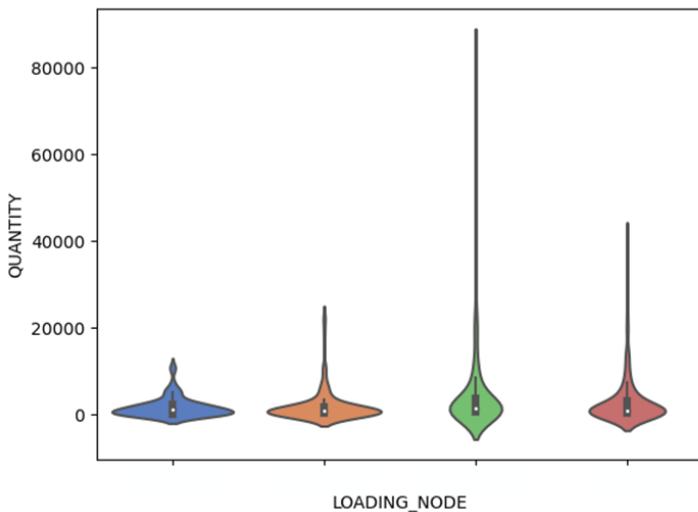


Figure 15.12: Violin chart with demand for each client for each terminal.

the volumes. The Pareto analysis helps to identify these products. Similar analyses to Figures 15.11, and 15.12 can be used to map the product mix.

**Information Pareto** Pareto analysis is important to assess the relevance of the information and the statistical coverage of any analysis. Let us define the “operators” as the different users uploading information in the dataset. When working with a supply chain network, it is easy to have many operators since there are tens of actors and multiple data sources. Each actor has a different relevance in terms of:

1. The number of records uploaded;
2. The amount of information generated by the records uploaded.

The last metric is not trivial since it considers the information added by a single operator compared to the information already known in the dataset. This metric is important when data are incomplete. Let assume we want to identify the route travelled by truck transporting a number of HUs. The truck receives order from multiple stakeholders. These stakeholders are operators since they upload data to our dataset.

Assuming a single stakeholder load the 80% of the truck, the amount of information generated by that single user is much more relevant than the others. The other operators have a low probability of adding information if a

single operator generates 80% of the knowledge. Figure 15.13 illustrates the behaviour of the information provided by different users of a vehicle. The first three users cover 80% of all the destinations of the vehicle, providing the principal amount of information on its route. Even without the information provided by the last three users, it is possible to use the data to estimate the route of the vehicle correctly.<sup>6</sup>

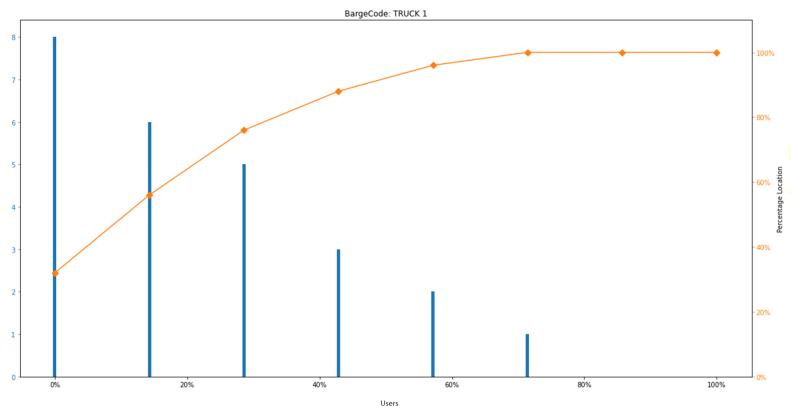


Figure 15.13: Information Pareto on client and vessel route.

### Operations profiling

**Movement Travel Time analysis** The analysis of the travel time reveals the time each HU  $i$  spend on a vehicle. By knowing the loading time window at node  $j$ , and the discharging time windows at the node  $k$ , it is possible to calculate the upper bound  $TT^{UB}$  and the lower bound  $TT^{LB}$  of the travel time as:

- $TT_{\pi}^{UB} = PTD_{ik}^{TO} - PTA_{ij}^{FROM}$
- $TT_{\pi}^{LB} = PTA_{ik}^{TO} - PTD_{ij}^{FROM}$

By aggregating these analyses, it is possible to identify the lead time  $LT_e$  of a route  $e$ . By knowing the actual time windows, the analysis can be repeated using:

- $TT_{\alpha}^{UB} = ATD_{ik}^{TO} - ATA_{ij}^{FROM}$
- $TT_{\alpha}^{LB} = ATA_{ik}^{TO} - ATD_{ij}^{FROM}$

Figure 15.14 compares the planned and actual travel time for the HUs handled in a distribution network.<sup>7</sup>

<sup>6</sup>The source code of Figure 15.13 is available [here](#).

<sup>7</sup>The source code of Figure 15.14 is available [here](#).

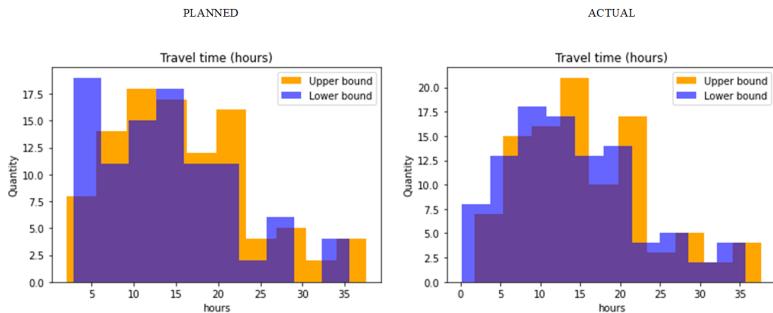


Figure 15.14: Planned and actual travel time.

By aggregating the actual travel times, the cycle time  $CT_e$  of a route  $e$  is revealed. By considering all the route  $e$  of a network  $G$ , the level of service of the network is calculated as the  $ProbCT_e \leq LT_e$  (see Figure 15.15).<sup>8</sup>

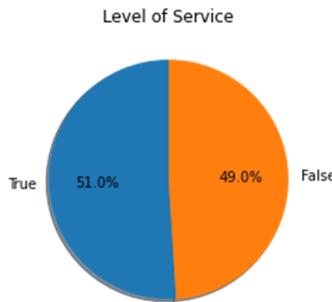


Figure 15.15: Pie chart representing the level of service of the distribution network.

**Product Lifecycle analysis** When the dataset contains a different code (specific item) for each different HU transported on the network, it is possible to have full traceability of the network and to reconstruct all the distribution stages. In particular, by knowing the loading and unloading timestamps, it is possible to define three status of the load:

- when a product was travelling;
- when a product was waiting (e.g. in a buffer or a storage system);
- when a product was loaded/unloaded.

<sup>8</sup>The source code of Figure 15.15 is available [here](#).

In addition, the commercial speed of the product is identified by considering the plot of the travelled distance (on the x-axis), and the timeline (on the y-axes). Figure 15.16 identifies the lifecycle of a HU with the timeline of the three states, and the commercial speed plot.<sup>9</sup>

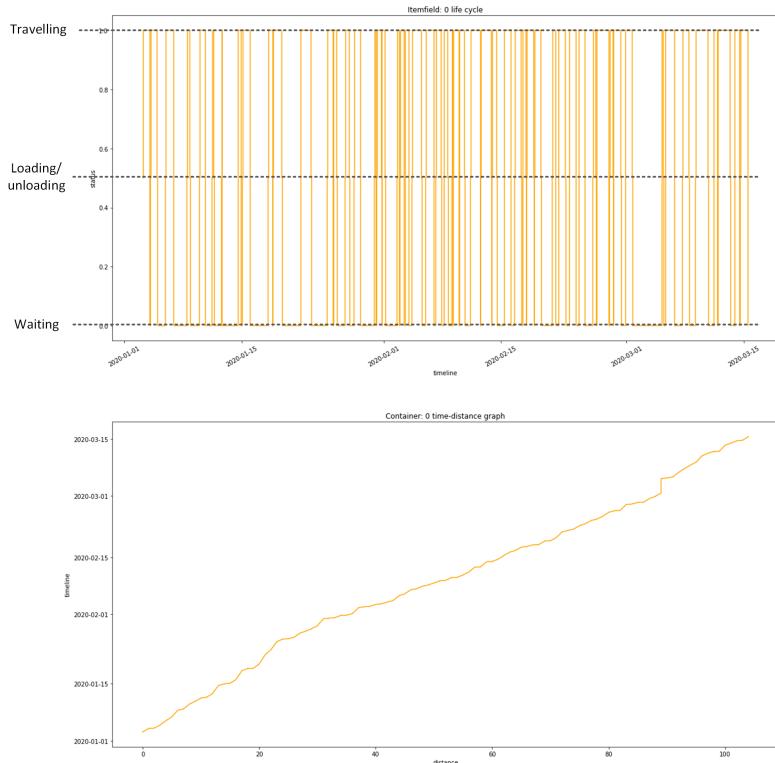


Figure 15.16: Plot of the lifecycle, and the commercial speed of a HU according to the three states

**Vehicle Inventory analysis** The analysis of the inventory position  $WIP_v$  is important to identify the utilisation  $U_v$  of a vehicle  $v$ , given its capacity  $C_v$ . It is necessary to know the planned/actual time windows at each terminal to infer the values of  $WIP_v$ ,  $U_v$ , and  $C_v$ . Usually, the value of  $C_v$  is fixed and depends on the type of vehicle or fleet. The other two values can be obtained by reconstructing the route of a vehicle  $v$ . Given a dataset with all the data, this is a heuristic procedure to get an estimate of the route.

- Filter the dataset by a vehicle  $v$ ;

---

<sup>9</sup>The source code of Figure 15.16 is available [here](#).

- Select the actual or provisional time windows;
- Sort the values by the visiting time;
- Calculate the cumulative of the movements to estimate  $WIP_j$ .

When there are no known values of the  $WIP_v(\tau)$  (e.g. from observation at time instant  $\tau$ ), the best estimate is obtained by shifting to positive values the cumulative function of the movements. Otherwise, the cumulative function can be added forward, and backward to the known value of  $WIP_j(\tau)$  (see Figure 15.17); the inventory information is also represented as the weight of a graph  $G(V, E)$ .<sup>10</sup>

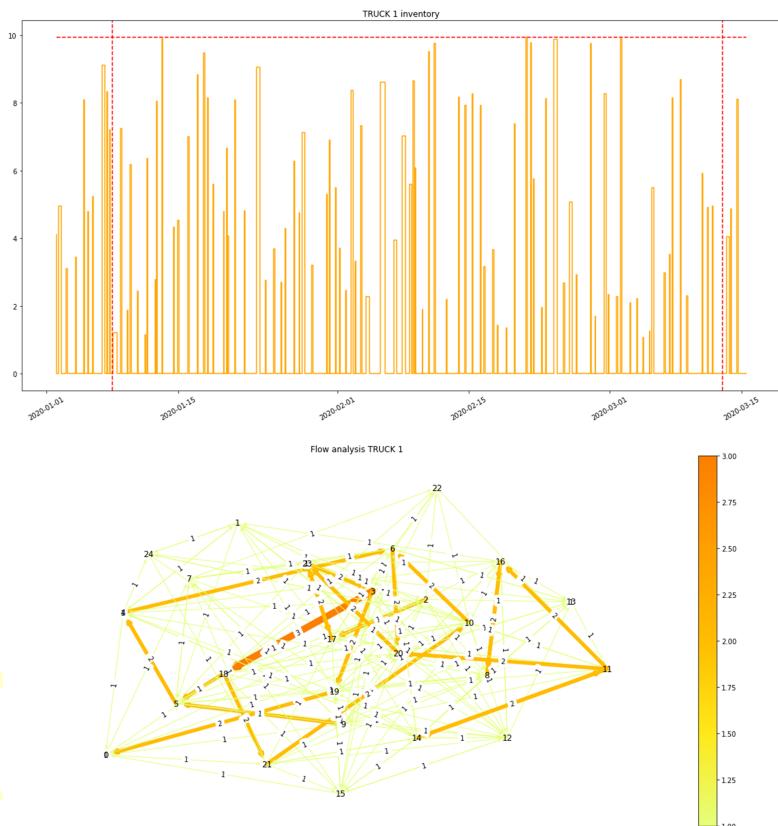


Figure 15.17: The function of the inventory position of a vehicle in the time and space domains.

<sup>10</sup>The source code of Figure 15.17 is available [here](#).

**Terminal Productivity analysis** The network moves at speed defined by its terminals. It may sound weird that the speed of the distribution is imposed by fixed infrastructure, but it is easier the capacity of a terminal is the bottleneck of a distribution network, more than the capacity of a vehicle. It is always simpler to add a truck, a vessel, a train or even an air cargo than to add loading and discharging capacity of these vehicles.

By collecting, for each movement, the information on the provisional and actual time windows, it is possible to say a lot on the planned and actual behaviour of the terminals. Figure 15.18 illustrates a scatterplot of the planned time windows of a terminal.<sup>11</sup> The x-axis identifies the span of the time windows, while the y-axis the amount of HUs loaded or discharged. The plot identifies two patterns, almost linearly distributed. This pattern uncovers that the terminal may use one or two cranes at the same time to load/discharge a vehicle.

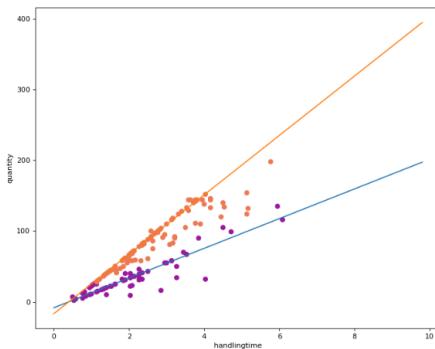


Figure 15.18: Productivity plot of a terminal.

This data reveals information on the capacity  $C_j$  of a terminal  $j$ . The inventory position  $WIP_j$  can be estimated with the same procedure illustrated in 15.2.2 by filtering on terminals, and not on vehicles. The throughput of a terminal usually depends on the time of the day. This information can be revealed by grouping the movements of a terminal on a specific day hour (see Figure 15.19).<sup>12</sup>

<sup>11</sup>The source code of Figure 15.18 is available [here](#).

<sup>12</sup>The source code of Figure 15.19 is available [here](#).

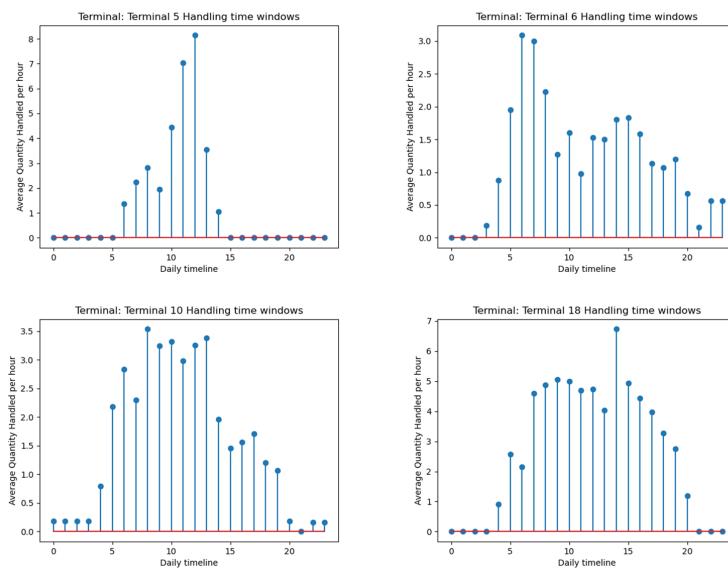


Figure 15.19: Productivity patterns of four terminals (planned, and actual).

## Network profiling

**Workload geography cost function** It is possible to represent a geographical cost function by matching together the pieces of information used in the previous paragraph and the geographical information (latitude and longitude) of the node of the network. In particular, the cost is represented by the quantity delivered at each node. Figure 15.20 illustrates the cost function using the map as a background to identify the geographical position and the size of the bubble to identify the intensity of the demand quantity. The colour of the bubble can represent the type of service (e.g., the type of the delivery node), or a gradient to represent the intensity.<sup>13</sup>

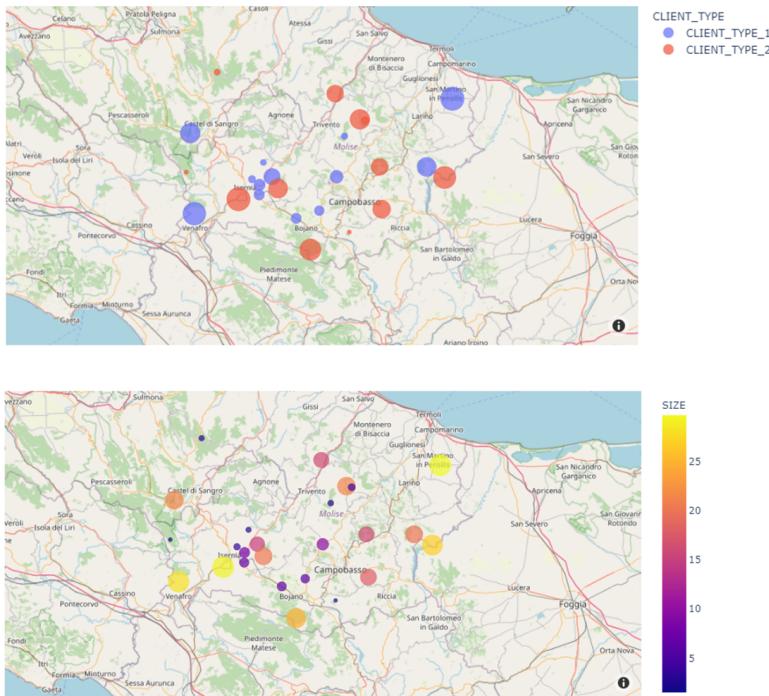


Figure 15.20: Workload of the network represented on a map.

**Network centre of mass** It is possible to define the centre of mass of the network, by considering the quantities, and the coordinates of the demand nodes of the network. Figure 15.21 compares the position of the centre of mass and the location of the plants serving the network.<sup>14</sup>

<sup>13</sup>The source code of Figure 15.20 is available [here](#).

<sup>14</sup>The source code of Figure 15.21 is available [here](#).

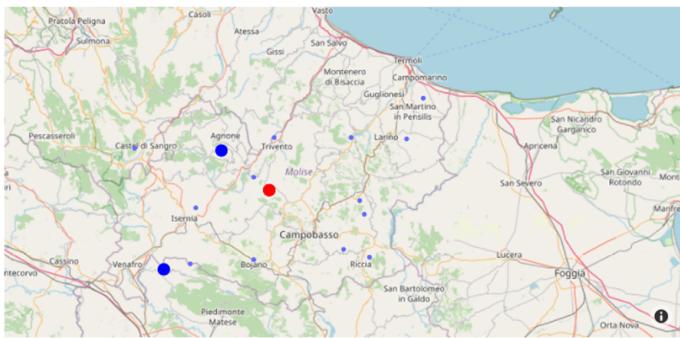


Figure 15.21: Current location of the plants of the network, and centre of mass.

**Demand geographical covering** It is possible to identify with different colours the node served by a specific plant of the network to identify how the production covers the demand on a geographical profile. Figure 15.22 illustrates the covering of the network.<sup>15</sup>

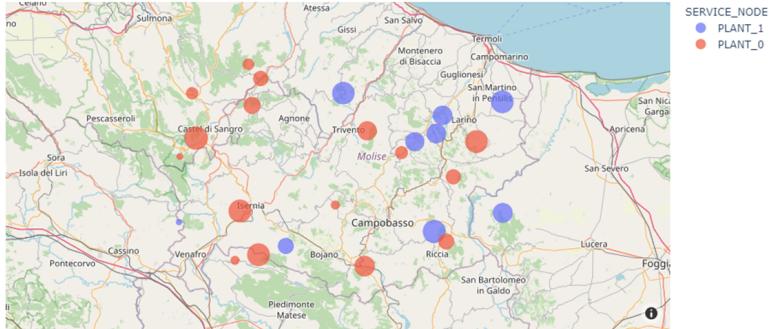


Figure 15.22: Covering of the network. Different colours identify nodes served by different facilities.

**Road graph analyses** Finally, it is possible to consider the road graph  $G(V, E)$  of the network to perform analysis on the real distances. The top of Figure 15.23 illustrates the edges of the graph. The subplot on the left uses rays to connect the nodes of the network exchanging the more significant

<sup>15</sup>The source code of Figure 15.22 is available [here](#).

amount of HUs. The subplot on the right use a colour gradient to identify the arcs travelled the most by the vehicles of the network.<sup>16</sup>

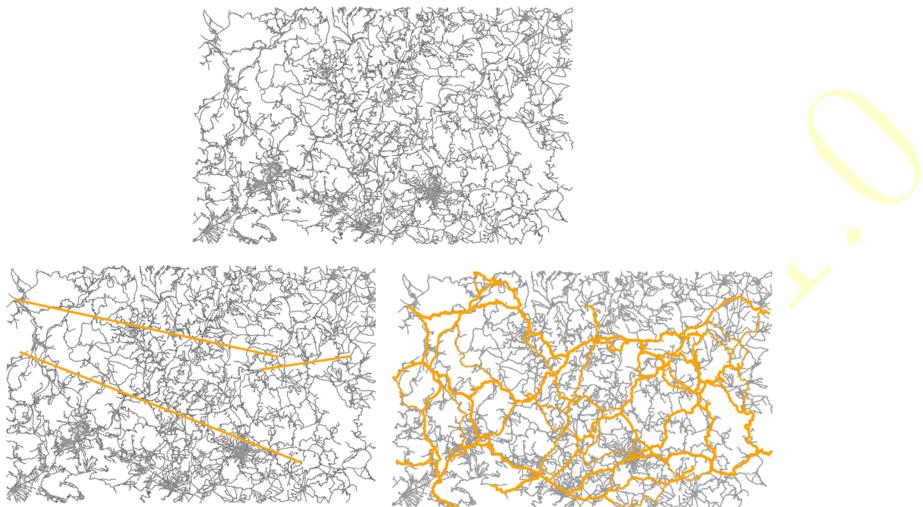


Figure 15.23: Road graph analyses.

## 15.3 Workload prediction (P9)

In a distribution network, it is crucial to forecast demand. The demand can be expressed in terms of:

- the number of orders (i.e. the workload).
- the inventory of a vehicle (i.e. the available space to allocate to handling units).

We introduce model-driven methods to forecast the number of orders, and data-driven methods to deal with the prediction of the available space of a vehicle.

### 15.3.1 Model-driven methods (PD2)

Model-driven methods to forecast the workload are provided by the statistical analysis of the time series. When dealing with order data of a distribution network, it is possible to obtain a time series by:

---

<sup>16</sup>The source code of Figure 15.23 is available [here](#).

1. considering the timestamp of the dataset;
2. cleaning the data by removing outliers;
3. grouping and summing the quantities of the dataset by the timestamps;
4. resampling the dataset to obtain an equispaced series  $y$  (e.g. weekly, daily or hourly).

At this stage, the series  $y$  is consistent and can be used to feed prediction models as the time series decomposition (see section 6.5.1), ARIMA models (see section 6.5.2), Fourier analysis to detect seasonality (see section 6.5.3), or other prediction models (e.g. fbprophet). Any of these models always provides:

- the predicted value in a future time lag;
- a confidence interval.

It is always necessary to consider the confidence interval. When it is too wide, the model does not provide robust predictions. In this case, it is possible to tune the model with different hyperparameters, provide additional data, or change the model.<sup>17</sup>

### 15.3.2 Data-driven methods (PD1)

As well as in storage and production system, forecasts on the number of movements (i.e. the number of HUs to handle) is essential for all the actors of a supply chain [28]. Another crucial variable is the inventory position of a vehicle  $v$ , important to assign transportation orders to vehicle. These forecasts are made by using time series methods; nevertheless, the  $WIP(t)$ , is more inclined to be affected by multiple factors, than the movements function. For this reason, machine learning algorithms are suitable to approach the predictions of the  $WIP(t)$ .

Once the route of a vehicle  $v$ , and its  $WIP_v$  have been estimated, as illustrated in 15.2.2, it is possible to train learning models for the prediction of the inventory position of the vehicle. In particular, the residual capacity  $r_j(t) = C_j - WIP_j(t)$  is of interest to investigate if a vehicle could handle more HUs than the ones already assigned to it. Learning models can be trained by using a training dataset  $X^e$  containing information on the route  $e$  (each row of the dataset is an arc travelled by a vehicle):

- the id of the vehicle;
- the capacity of the vehicle;

---

<sup>17</sup>The logproj package provides methods to deal with workload predictions [here](#).

- the node of departure;
- the time of departure;
- the node of arrival;
- the time of arrival.

While the target variable is represented by  $WIP_j(t)$ . The input dataset  $X^e$  contains many date and time information. Learning models work approximating functions, i.e. they need only numerical values as inputs. For this reason, pre-processing is necessary to convert timestamps and categorical variables. Timestamps are transformed using  $\sin$  and  $\cos$  functions to separate year, month, days, hours and minutes in different attribute, preserving a measure of proximity between them (e.g. 00.01 is hugely close to 23.59 even if the digits representing the timestamp are at the opposite of the hours and minute domains). This transformation produces nine numerical features for each timestamp (one representing the years, two representing the month numbers, two representing the day numbers, two representing the hours, two representing the minutes). Categorical variables are converted to dummy column having value one if the observation has the value of the column, zero otherwise. This conversion produces a number of additional columns to the dataset  $X^e$  equal to the number of categories of a categorical variable.

At this stage, many learning algorithms can be trained to approximate the value of  $WIP_j(t)$ . Any of the algorithms in chapters 9, and 11 can be used, e.g. linear regression, lasso, ridge regression, elastic net, regression tree, random forest, gradient boosting, AdaBoost, support vector regression and single perceptron neural network. It is recommended to implement a validation procedure, as the cross-validation (see 9.1) and to choose a proper loss function (e.g. the MSE).

If the value of the MSE is low enough to support robust predictions, the model can be used to support decisions and to allocate capacity in advance. There is an additional parameter to consider to evaluate the performance of a model. When the model is trained, it produces an error with a fixed amount of data. When the model is deployed, this amount of data may vary. For this reason, it is crucial to consider the learning curve of a model. Learning curves track the value of the loss function (e.g. the MSE) depending on the amount of input data. Each algorithm runs several times with a different size of the input dataset randomly bootstrapped from the complete dataset  $X^e$ . An example of learning curves is presented in Figure 15.24. The x-axes represent the number of records used to feed the learning algorithm (i.e. from 10% to 100% of the input dataset) while the y-axes indicate the average MSE produced by the algorithms with 5-folds CV.

The example shows that many algorithms (i.e., neural network, gradient boosting, random forest and elastic net) tend to significantly reduce the

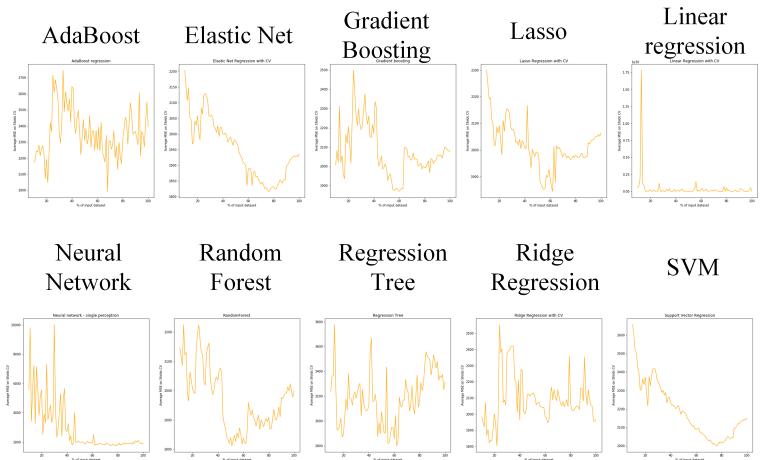


Figure 15.24: Example of the learning curves of regression algorithms to predict the residual capacity of a vehicle.

MSE while increasing the size of the dataset. For this reason, the model built by these algorithms may have significant potential in practice fed using bigger datasets than the one used in this example.<sup>18</sup>

## 15.4 Vehicle choice & synchromodality (P2)

The choice of an adequate vehicle to transport HUs from an origin to a destination can heavily affect the performance of the supply chain network in terms of the level of service and environmental impact. This problem is called *synchromodality* when the choice is performed online, i.e. without a predefines association between the HU and the vehicle (e.g. truck, train, barge). Recent literature put a strong effort into developing synchromodality models. For the sake of brevity, we introduce an example inspired to the operations of a logistic platform, i.e. a company with a business model based on platform economy aiming at simplifying the operations of a distribution network (see section 15.1.2).

### 15.4.1 Data-driven methods (PS2)

We simulate the operations of a logistic platform by measuring the performance of the network using the transportation cost. The platform has to assign a set of containers to transportation services using barges or truck,

<sup>18</sup>The logproj package provides methods to deal with inventory predictions [here](#).

where transportation of a container by truck costs 1.3 times more than a barge [29, 30]. For simplicity, we consider a single barge sailing the distribution network. The platform prefers to assign containers to the barge since it is cheaper than the truck. For this reason, their objective is the maximisation of the number of containers successfully transported by barge.

The platform uses a prediction model  $\Pi$  to forecast the inventory position of the barge, defining the available capacity  $\mu^{pred}$  of the barge in terms of available container slots. The accuracy of the available capacity is measured using the standard deviation  $\sigma^{pred}$ . The more accurate the prediction model  $\Pi$ , the lower  $\sigma^{pred}$ . The platform assigns a container to a truck when there is no space on a barge, based on  $\mu^{pred}$ ; or when the real available capacity  $\mu^{real} < \mu^{true}$ . The assignments performed by the platform can be wrong due to bad predictions of the model  $\Pi$ . In particular, a type I error is made when a container is allocated to barge when the barge is already full (i.e. false positive prediction); a type II error is made when a container is allocated to a truck when the barge has space (i.e. false negative prediction). Figure 15.25 identifies all the possible outcomes of the prediction model  $\Pi$ .

		True value	
		The barge has capacity	The barge does not have capacity
Prediction of the model $\Pi$	The barge has capacity	<b>True positive</b> <b>TP</b> Container assigned to the barge when $\mu_{pred} < \mu_{true}$	<b>False positive</b> <b>FP</b> Type I error Container assigned to the barge when $\mu_{pred} > \mu_{true}$
	The barge does not have capacity	<b>False negative</b> <b>FN</b> Type II error Container not assigned to the barge when $\mu_{pred} < \mu_{true}$	<b>True negative</b> <b>TN</b> Container not assigned to the barge when $\mu_{pred} > \mu_{true}$

● Good predictions      ● Bad predictions

Figure 15.25: Confusion matrix of a predictive model for synchromodality.

We implemented a simulation to investigate the impact of the accuracy  $\sigma^{pred}$  of the model  $\Pi$  on the performance of the platform, by measuring:

- the number of containers  $\alpha$  assigned to the barge when  $\mu^{pred} > \mu^{true}$  (i.e. the number of false positives);
- the number of containers  $\beta$  assigned to trucks due to wrong predictions, when  $\mu^{pred} < \mu^{true}$  (i.e. the number of false negatives);

- the total cost of the transportation service;
- the level of service, measured as the probability that the platform successfully assign a container to a barge (i.e. the percentage of true positives).

The simulation varies the accuracy of the predictions  $\sigma^{pred}$  in a range from 0 to  $\mu^{real}$ . Figure 15.26 illustrates the outcome of the simulation, having on the x-axes the variation coefficient  $\frac{\sigma^{pred}}{\mu^{true}}$ , and the KPIs identified above on the y-axes, for each plot. The graph reveals that when the model II has higher accuracy in the prediction of the inventory position of the barge (i.e. the state  $\Lambda$ ), the platform performs better, with a lower number of false positives and false negatives. This has a positive outcome on the level of service of the platform, and on the total transportation cost of the network.

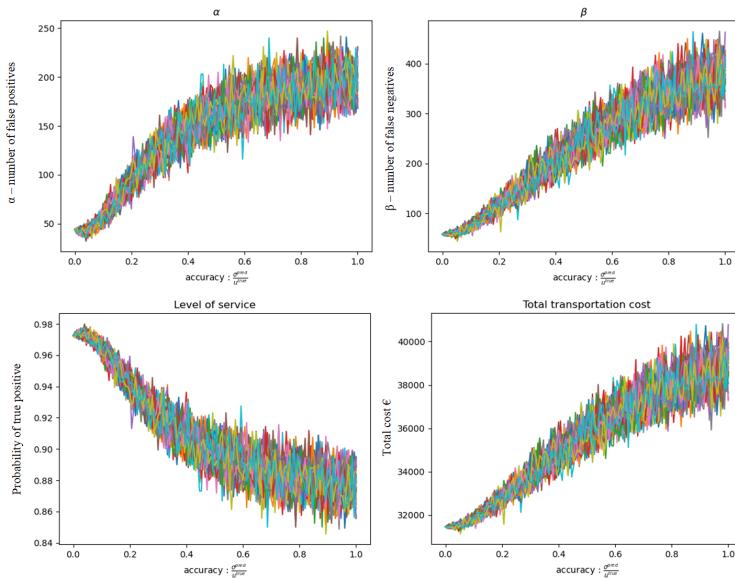


Figure 15.26: Results of the simulation of the operations of a logistic platform

This example shows that good prediction models targeting the inventory position of a barge can improve the level of service of the platform itself, and reduce the total transportation cost of a barge network. Differently from single actors of the supply chain, logistic platforms have the data and the possibility to implement these methodologies. The service they can deliver with this information is not only for their clients (e.g. the barge operators).

Other stakeholders like terminals, shipper, and port authorities, can benefit from the information produced by the platforms.

## 15.5 Vehicle routing (P10)

Vehicle routing is a prescriptive problem to assign HUs to vehicles and define vehicle routes such that the distribution cost is minimised. We consider, first the problem from a classic and model-driven operation research perspective.

### 15.5.1 Model-driven methods (PS1)

This paragraph considers three different operations research approaches. The first is the travelling salesman problem (TSP), aiming at the definition of the cheapest route connecting all the nodes of a graph (regardless of the resource capacity). Then, the vehicle routing problem (VRP) is introduced using a traditional descriptive model and a smarter solving strategy using column generation.

#### Separation algorithm for the TSP

We do not introduce the descriptive model of the TSP since it can be found in almost any operation research book. We start illustrating a smarter procedure to solve it by separation (see 13.3.1). Let consider the parameter  $d_a$  be the cost of travelling the arc  $a = (i, j)$ . Let introduce the variables:

$$w_a = \begin{cases} 1 & \text{if arc } (i, j) \text{ is included in the solution} \\ 0 & \text{otherwise} \end{cases} \quad (15.1)$$

$$w_i = \begin{cases} 1 & \text{if vertex } i \text{ is included in the solution} \\ 0 & \text{otherwise} \end{cases} \quad (15.2)$$

The TSP problem is defined as follows.

$$\min \sum_{a \in A} w_a d_a \quad (15.3)$$

$$\sum_{a \in \delta^+(i)} w_a = w_i, \forall i \in V \quad (15.4)$$

$$\sum_{a \in \delta^-(i)} w_a = w_i, \forall i \in V \quad (15.5)$$

$$\sum_{i \in V} w_i = N \quad (15.6)$$

$$w_a, w_i \in 0, 1 \quad (15.7)$$

The objective function (15.3) aims at the minimisation of the distribution cost. Constraints (15.4), and (15.5) impose to chose arcs both to enter and exit a vertex  $i$ , constraints (15.6) impose to visit all the vertices, constraint (15.7) impose integrality of the decision variables. A model set using equations (15.3)-(15.7) may produce a solution with multiple sub-tours. A set of sub-tour elimination constraints is needed to avoid this behaviour. The number of sub-tours in a graph can be exponential. For this reason, there is an exponential number of constraints to add to the problem (15.3)-(15.7). Models with an exponential number of constraints may take forever to reach optimality. For this reason, we relax this constraint and use a separation procedure adding one-by-one violated sub-tour elimination constraints to the model.

We introduce an optimisation problem called *separation problem* to find a sub-tour elimination constraint  $S^*$  violated by the initial problem. The problem has variables:

$$y_i = \begin{cases} 1 & \text{if vertex } i \in S^* \\ 0 & \text{otherwise} \end{cases} \quad (15.8)$$

$$z_a = \begin{cases} 1 & \text{if arc } (i, j) \in S^* \\ 0 & \text{otherwise} \end{cases} \quad (15.9)$$

The solution  $z_a^*$  of the following separation problem is used to identify a sub-tour elimination constraint which is violated by the solution of the initial problem  $w_a^*$ . The separation problem is defined as follows:

$$\min \sum_{i \in V} y_i - \sum_{a \in A} w_a^* z_a \quad (15.10)$$

$$\sum_{i \in V} y_i \leq 2 \quad (15.11)$$

$$\sum_{i \in V} y_i \geq N - 2 \quad (15.12)$$

$$y_i \leq z_a, \forall a = (i, j) \in A \quad (15.13)$$

$$y_j \leq z_a, \forall a = (i, j) \in A \quad (15.14)$$

$$z_a \leq y_i + y_j - 1, \forall a = (i, j) \in A \quad (15.15)$$

$$z_a, y_i \in 0, 1 \quad (15.16)$$

When the separation problem produces a solution value less than, or equal to '1', a constraint

$$\sum_{a \in A(S^*)} z_a^* > \sum_{i \in V(S^*)} y_i - 1 \quad (15.17)$$

is added to the initial problem. Then the initial problem is solved again (with the relaxation of the sub-tour elimination constraints), and the solution goes again to the separation problem. This algorithm stops when the separation problem does not produce additional constraints to add to the original problem. At this point, the initial problem is solved to optimality using a minimal number of sub-tour elimination constraints produced by the separation problem.

### Descriptive model for the VRP

The most comprehensive prescriptive model in the field of distribution networks is an evolution of the TSP taking care of the capacity of the vehicle visiting the vertices [31]. This is the vehicle routing problem (VRP). One of the most complex and comprehensive versions of this problem is the VRP with pickup and deliveries and time windows (VRPPDTW). The VRPPDTW is modelled considering:

- a fleet of vehicles  $v \in B$ ;
- a set of pickup and delivery orders  $o \in O$  to serve;
- a set of pickup nodes  $j \in P$  and a set of delivery nodes  $j \in D$  defining a directed graph  $G(V, A)$  where  $V = P \cup D$  and  $(i, j) \in A$ .

In this paragraph, we introduce a VRPPDTW problem to maximise the profit of a shipper of cargo vessels [32]. Despite the scope of the problem, its formulation is valid for any distribution network with pickup and deliveries and time windows. The parameters of the problem are illustrated in Table 15.3.

Parameter	Description
$i = 1, \dots, n \in V$	Set of pickup nodes
$i = n + 1, \dots, 2n \in V$	Set of delivery nodes
$(i, j) \in E$	Set of edges (i.e. routes) of a complete graph
$p = 1, \dots, m \in P$	Set of products
$(p, i, j) \in O: p \in P; (i, j) \in E$	Set of orders of product $p$ from node $i$ to node $j$
$q_{p,i,j}$	Quantity (tons) of order $p, i, j$
$t_{p,i,j}$	Timestamp indicating when order $p, i, j$ is ordered
$time_{i,j}$	Travel time on edge $i, j$
$cost_{i,j}$	Cost for travelling arc $i, j$
$Tstop_i$	Average time to wait for load/unload operations at node $i$
$Cstop_i$	Cost for stopping at node $i$
$price_{i,p}$	Market price of product $p$ on the market of node $i$
$a_i$	1 if node $i$ is the first leaving node (i.e., depot)
$Q$	Capacity of a single vehicle of the fleet

Table 15.3: Parameters of the VRP problem.

The mathematical model admits the following decision variables.

$$x_{i,j} = \begin{cases} 1 & \text{if arc } i,j \text{ is travelled} \\ 0 & \text{otherwise} \end{cases} \quad (15.18)$$

$$v_i = \begin{cases} 1 & \text{if node } i \text{ is visited} \\ 0 & \text{otherwise} \end{cases} \quad (15.19)$$

$$s_i = \text{timestamp upon leaving node } i \quad (15.20)$$

$$u_{p,i,j} = \begin{cases} 1 & \text{if order } p, i, j \text{ is served} \\ 0 & \text{otherwise} \end{cases} \quad (15.21)$$

$$\text{load}_{p,i} = \text{upper bound of the total pickup quantity upon leaving node } i \quad (15.22)$$

$$\text{unload}_{p,i} = \text{upper bound of the total delivered quantity upon leaving node } i \quad (15.23)$$

The feasible region is defined by the following sets of constraints.

$$\sum_j x_{i,j} = v_i, i \in V \quad (15.24)$$

$$\sum_i x_{i,j} = v_j, j \in V \quad (15.25)$$

$$s_i = 0, i \in V : a_i = 1 \quad (15.26)$$

$$s_j \geq s_i + \text{time}_{i,j} - M(1 - x_{i,j}), (i, j) \in E; i, j \in V \setminus \{j : a_j = 1\} \quad (15.27)$$

$$s_j \leq s_i + \text{time}_{i,j} + M(1 - x_{i,j}), (i, j) \in E; i, j \in V \setminus \{j : a_j = 1\} \quad (15.28)$$

$$u_{p,i,j} \leq v_i, (p, i, j \in O) \quad (15.29)$$

$$u_{p,i,j} \leq v_j, (p, i, j \in O) \quad (15.30)$$

$$load_{p,i} = \sum_j q_{p,i,j} \times u_{p,i,j}, p \in P, i \in V : a_i = 1 \quad (15.31)$$

$$load_{p,j} \geq -M(1 - x_{i,j}) + load_{p,i} + \sum_k q_{p,j,k} \times u_{p,j,k}, \\ p \in P \quad (i, j) \in E; i, j \in V \setminus \{j : a_j = 1\} \quad (15.32)$$

$$unload_{p,i} = 0, p \in P, i \in V : a_i = 1 \quad (15.33)$$

$$unload_{p,j} \geq -M(1 - x_{i,j}) + unload_{p,i} + \sum_k q_{p,k,j} \times u_{p,k,j}, \\ p \in P \quad (i, j) \in E; i, j \in V \setminus \{j : a_j = 1\} \quad (15.34)$$

$$0 \leq \sum_p load_{p,i} - unload_{p,i} \leq Q, i \in V \quad (15.35)$$

$$u_{p,i,j} \times t_{p,i,j} \leq s_j + time_{i,j}, (p, i, j) \in O \quad (15.36)$$

$$x_{i,j} \in 0, 1, (i, j) \in E \quad (15.37)$$

$$v_i \in 0, 1, i \in V \quad (15.38)$$

$$s_i \geq 0, i \in V \quad (15.39)$$

$$u_{p,i,j} \in 0, 1, (p, i, j) \in O \quad (15.40)$$

$$load_{p,i,j} \geq 0, (p, i, j) \in O \quad (15.41)$$

$$unload_{p,i,j} \geq 0, (p, i, j) \in O \quad (15.42)$$

Constraints (15.24) and (15.25) ensure the route is continuous among nodes. Constraints (15.26) set the leaving timestamp at the first node. Constraints (15.27) and (15.28) are used to track the leaving timestamp at each node. Constraints (15.29) and (15.30) impose a node must be visited to serve its orders. Constraints (15.31) set the pickup value at the first node while constraints (15.32) do the same for all the following. Constraints (15.33) set the delivery value at the first node while constraints (15.34) do the same for all the following. Constraints (15.35) ensure the capacity is never exceeded, and constraints (15.36) state that an order can be

served when it comes before the picking node is visited. Constraints (15.37), (15.38), (15.39), (15.40), (15.41), (15.42) set the values of the variables.

The maximisation of the profit of the shipper is chosen as objective function.

$$\sum_{p,i,j} q_{p,i,j} \times u_{p,i,j} \times price_{p,i,j} - \sum_i C_{stop_i} \times v_i - \sum_{i,j} c_{i,j} \times x_{i,j} \quad (15.43)$$

This problem has been solved using the branch and bound algorithm to test its functionality on a toy instance (i.e. an extremely small instance of the problem). The toy instance network is composed of the nine vertices illustrated in the left of Figure 15.27. The right part of Figure 15.27 illustrates the solution of the toy instance, similarly to the descriptive analytics illustrated in 15.2.2.

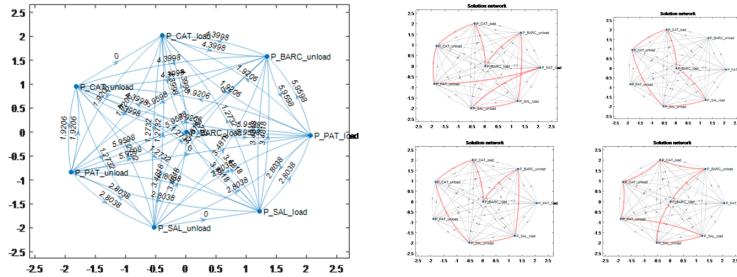


Figure 15.27: Solution of the toy instance of the VRP problem.

Since the problem involves time windows, for each route identified by the solution, it is possible to identify the timestamps of visit at the ports similarly to the analytics of 15.2.2 (see Figure 15.28).

It is crucial to utilise the same KPIs and analytics to evaluate both the actual scenario and the ones produced by prescriptive analytics.

### Column generation algorithms for the VRP

Operations research proposes many descriptive models to address the vehicle routing problem (VRP). Unfortunately, even if easy to understand, they are hard to solve since VRP is NP-complete. Still being NP-complete, there are smart algorithm known as *column generation algorithms* to expedite the research of the optimal solution. For this reason, we introduce this type of algorithms that works properly with relatively small networks (few hundreds of nodes). Let us consider the parameters of the problem presented in Table 15.4.

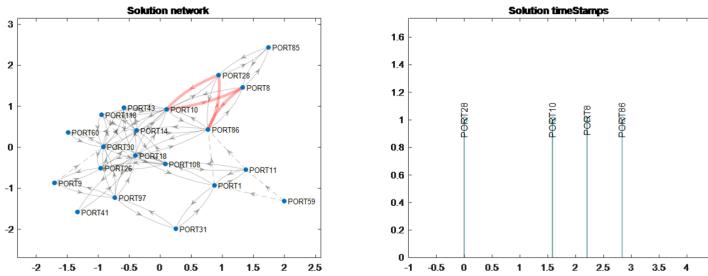


Figure 15.28: Descriptive analytics to evaluate the solution of the prescriptive problem.

Parameter	Description
$w_i$	Demand associated to order $i$
$d_a$	Distance on arc $a = (j, k)$
$W$	Capacity of a vehicle
$D$	Maximum allowable distance between two nodes of a route

Table 15.4: Parameters of the column generation algorithm for the VRP.

We define a set  $S$  containing all the feasible routes on the network (i.e. the routes respecting the capacity  $W$ , and where nodes are not farther than  $D$ ).

$$S = \left\{ s \subseteq \{i = 1, \dots, m\} : \sum_{i \in s} w_i \leq W, \max_{a:j,k \in s} d_a \leq D \right\} \quad (15.44)$$

The optimal solution to the VRP is the solution to the set covering problem (SCP) called *primal problem*, having the parameter  $c_s$  to define the cost of the route  $s$ , and a variable:

$$x_s = \begin{cases} 1 & \text{if configuration } s \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \quad (15.45)$$

The primal problem is defined as follows.

$$\min c_s x_S \quad (15.46)$$

$$\sum_{s \in S: i \in s} x_s \geq 1, i = 1, \dots, m \quad (15.47)$$

$$x_s \in \{0, 1\} \quad (15.48)$$

The SCP problem has an exponential number of variables that are unknown in advance (we do not know, at the beginning, all the feasible routes of set  $S$ ). For this reason, we initialise the primal problem with a trivial feasible solution of the SCP (i.e. an identity matrix with a high cost  $c_s$ ) and we use a dual problem to generate cheaper columns of the matrix. We relax the integrality of the primal problem and consider its *dual problem*, as follows.

$$\max \pi_i \quad (15.49)$$

$$\sum_{i \in S} \pi_i \leq c_s, s \in S \quad (15.50)$$

$$\pi_i \geq 0, i \in S \quad (15.51)$$

At this point, we need to violate a constraint of the set (15.50). When this constraint exists, a cheaper column to add to the  $S$  exists as well for the theorem of optimality by separation (see section 13.3.1). We use an optimisation model to find the violated constraint.

$$\mu_i = \begin{cases} 1 & \text{if } i \text{ belongs to column} \\ 0 & \text{otherwise} \end{cases} \quad (15.52)$$

$$\max \sum_{i=1}^m \mu_i (\pi_i - c_i) \quad (15.53)$$

$$\sum_{i=1}^m \pi_i w_i \leq W \quad (15.54)$$

$$\mu_i \geq z_a, a = (i, j) \quad (15.55)$$

$$\mu_j \leq z_a, a = (i, j) \quad (15.56)$$

$$\mu_i + \mu_j - 1 \leq z_a, a = (i, j) \quad (15.57)$$

$$z_a d_a \leq D, a = (i, j) \quad (15.58)$$

$$\pi_i, z_a \in \{0, 1\}, a = (i, j); i, j = 1, \dots, m \quad (15.59)$$

If this model produces a solution value  $\sum_{i=1}^m \mu_i(\pi_i - c_i) > 0$ , a column with profit is found, and it is added to the set  $S$  of the SCP. Otherwise, the set  $S$  contains all the routes to find the optimal solution to the problem. The solution of the primal SCP problem reveals the solution to the VRP.

Similarly, it is possible to use this approach to solve the VRPPDTW maximising the profit of the shipper, illustrated in the previous paragraph using the descriptive model. Let consider the set packing problem with the following variables:

$$C = \text{set of all maximal orderset served by a feasible route} \quad (15.60)$$

$$x_c = \begin{cases} 1 & \text{if orderset } c \text{ selected} \\ 0 & \text{otherwise} \end{cases} \quad (15.61)$$

The problem has a parameter  $\gamma_c$ , indicating the profit associated to the orderset (i.e.e the route)  $c$ . The primal problem is identified as:

$$\max \sum_{c \in C} x_c \gamma_c \quad (15.62)$$

$$\sum_{c \in C: o \in c} x_c \leq 1, o \in O \quad (15.63)$$

$$x_c \in \{0, 1\}, c \in C \quad (15.64)$$

Let define the dual problem at:

$$\max \sum_{o \in O} \pi_o \quad (15.65)$$

$$\sum_{o \in c} \pi_o^{\leq} - \gamma_c, c \in C \quad (15.66)$$

$$\pi_o^{\leq} \geq 0, i \in O \quad (15.67)$$

We are interested in finding a set  $c$  such that:

$$c : \sum_{o \in c} \pi_o^* + \gamma_c > 0 \quad (15.68)$$

Let consider a column generation problem setting the value of the decision variable:

$$y_o = \begin{cases} 1 & \text{if order } o \text{ in } c \\ 0 & \text{otherwise} \end{cases} \quad (15.69)$$

The problem is defined as follows:

$$\max \sum_{o \in O} (\pi_o^* - profit_o) y_o - \sum_i C_{stop} \times v_i - \sum_{i,j} c_{i,j} \times x_{i,j} \quad (15.70)$$

$$profit_o \leq q_{p,i,j} \times u_{p,i,j} \times price_{p,i,j}, o = (p, i, j) \quad (15.71)$$

$$u_{p,i,j} \leq y_o, o = (p, i, j) \quad (15.72)$$

Subject to all the constraints 15.24 to 15.42 defining the feasible region. Each time this problem is solved with a solution value greater than '0', a maximal orderset is found and added to the primal problem following the algorithm illustrated for the minimisation case.

### 15.5.2 Data-driven methods (PS4)

The models illustrated in the previous section are complex and extremely biased. Besides, they need long runtimes to produce an optimal solution. This running time and the hypotheses made to set the model (e.g. the linearity of all the behaviour described by the constraints) are hard to support in a real environment.

For this reason, we introduce a different approach, where the past observation of the network can help to find a solution to an online version of the VRP by using predictive algorithms. The models presented in 15.5.1 solve an offline version of the VRP, i.e. all the data are not only static but given before running the algorithm that solves the model. In practice, many parameters of the problem are unknown, or they change their value depending on many external factors. Finding an online solution of the VRP means finding a feasible assignment of a HU to a vehicle, given the current state of the system (i.e. the vehicles present in the network, their routes, and their available capacity). This can be seen as the Uber problem, who find a ride to share with other users, given a set of available cabs.

This is the perfect job for a logistic platform, collecting the data of different stakeholders of a distribution network. Let us consider, for example, a platform collecting the movements  $M_o^{j,v}$  from different conveyance operators, where  $o$  is a single transportation order,  $j$  the origin terminal, and  $v$  the vehicle. Let us define a state  $\tilde{\Lambda}(\tau)$  defined as the set of the inventory position of all the terminals  $j$ , and all the vehicles  $v$  of the network at time instant  $\tau$ . If the logistic platform can estimate  $\tilde{\Lambda}(\tau)$ , is it able to define if a vehicle has available capacity for an additional HU (i.e. a feasible solution of the online VRP).

The platform can estimate the inventory position of the barge, by using the movement function (using an algorithm similar to the one presented in

section 3.3.1). Another source of information comes from the productivity function of the terminals  $j$  (i.e.  $P_j^{IN}$ , and  $P_j^{OUT}$ ). These two data sources can be matched together by using a Kalman filter, to improve the estimate  $\tilde{\Lambda}$ . The number of movements  $n_v$  observed for each single vehicle  $v$  defines the completeness of  $M_v(t)$ , while the number of all the observations  $m$  defines the completeness of  $P_j^{IN}$ , and  $P_j^{OUT}$  for each terminal  $j$ . Two estimators of  $\tilde{\Lambda}$  can, then, be obtained by using two different approaches:

- $\tilde{\Lambda}^M$ , with an empirical approach, from the knowledge of the movements  $M_j(t)$ ;
- $\tilde{\Lambda}^K$ , with a probabilistic approach, from the definition of kinematic models based on the speeds of the terminals given by  $P_j^{IN}$ , and  $P_j^{OUT}$ .

An empirical approach directly applies the equations (3.7) and (3.8). A probabilistic approach defines a kinematic model  $K_j$  for each terminal  $j$ .  $K_j(\delta^t, P_j^{IN}, P_j^{OUT}, Prob_j^{OUT})$  returns the number of HUs loaded or offloaded when a barge stops at a terminal  $j$  with a service time windows  $\delta^t$ . The model considers the probability distribution functions of the productivity  $P_j^{IN}$  and  $P_j^{OUT}$ , and the probability that the terminal  $j$  loads or offload containers  $Prob_j^{OUT}$ .

The Kalman filter (see section 11.3.3) is introduced to mix the information from the empirical, and the probabilistic approach. A Kalman filter (KF) considers the input and outputs of a kinematic model, and it corrects the output value based on empirical measurements when they are available. The filter is suitable for a logistic platform since it can be continuously updated, allowing real-time implementations. The Kalman filter estimates the value of  $\tilde{\Lambda}(\tau)$  as a hidden state of a Markov model. Given the motion equation  $K_j$  for each terminal  $j$ , the filter updates  $\tilde{\Lambda}(\tau)$  by considering the empirical measurements  $M_o^{j,v}$ . The use of the filter always improves the accuracy of  $\tilde{\Lambda}(\tau)$  obtained by using only the probabilistic approach. Nevertheless, it may be useless when  $M_v(t)$  has an information content high enough to (e.g. when all the movements  $M_o^{j,v}$  of a vehicle  $v$  have been observed.

## Bibliography

- [1] Research and markets, “Logistics Market: Global Industry Trends, Share, Size, Growth, Opportunity and Forecast 2018-2023,” tech. rep., Research and markets, 2018.
- [2] Q. Li and A. Liu, “Big Data Driven Supply Chain Management,” in *52nd CIRP Conference on Manufacturing Systems*, vol. 83, pp. 814–818, 2019.

- [3] K. Lamba and S. P. Singh, "Big data in operations and supply chain management: current trends and future perspectives," *Production Planning and Control*, vol. 28, no. 11-12, pp. 877–890, 2017.
- [4] K. Govindan, T. C. Cheng, N. Mishra, and N. Shukla, "Big data analytics and application for logistics and supply chain management," *Transportation Research Part E: Logistics and Transportation Review*, vol. 114, no. March, pp. 343–349, 2018.
- [5] Kü, M. Ckelhaus, G. Chung, B. Gesing, G. Steinhauer, M. Heck, and K. Dierkx, *Artificial Intelligence in Logistics*. DHL Customer Solution & Innovation, 2018.
- [6] R. A. Zuidwijk and A. W. Veenstra, "The Value of Information in Container Transport," *Transportation Science*, vol. 49, no. 3, pp. 675–685, 2014.
- [7] S. Nambisan, M. Wright, and M. Feldman, "The digital transformation of innovation and entrepreneurship: Progress, challenges and key themes," *Research Policy*, vol. 48, no. 8, p. 103773, 2019.
- [8] J. D. Dana, "Remark on "Appropriateness and Impact of Platform-Based Product Development"," *Management Science*, vol. 49, no. 9, pp. 1264–1267, 2003.
- [9] M. Kenney and J. Zysman, "The rise of the platform economy," *Issues in Science and Technology*, vol. 32, no. 3, pp. 61–69, 2016.
- [10] S. Benjaafar, G. Kong, X. Li, and C. Courcoubetis, "Peer-to-peer product sharing: Implications for ownership, usage, and social welfare in the sharing economy," *Management Science*, vol. 65, no. 2, pp. 477–493, 2019.
- [11] H. Guda and U. Subramaniana, "Your uber is arriving: Managing on-demand workers through surge pricing, forecast communication, and worker incentives," *Management Science*, vol. 65, no. 5, pp. 1995–2014, 2019.
- [12] P. Huang, M. Ceccagnoli, C. Forman, and D. J. Wu, "Appropriability mechanisms and the platform partnership decision: Evidence from enterprise software," *Management Science*, vol. 59, no. 1, pp. 102–121, 2013.
- [13] S. Yu and Y. Cao, "Research of the 4th Party Logistics Network Platform Based on XML," in *International Conference on Transportation Engineering*, pp. 2494–2499, 2007.

- [14] X. Xiu and J. Zheng, "Study of integrated information platform of 4PL based on collaborative environment," *2010 2nd Conference on Environmental Science and Information Application Technology, ESIAT 2010*, vol. 1, pp. 690–693, 2010.
- [15] K. L. Choy, S. C. K. So, H. C. W. Lau, S. K. Kwok, and F. T. S. Chan, "Development of an integrated logistics information system for third party logistics facilitators," *International Journal of Business Performance Management*, vol. 8, no. 2-3, pp. 170–193, 2006.
- [16] J. J. Liu, S. C. So, K. L. Choy, H. Lau, and S. K. Kwok, "Performance improvement of third-party logistics providers - An integrated approach with a logistics information system," *International Journal of Technology Management*, vol. 42, no. 3, pp. 226–249, 2008.
- [17] M. A. Figliozi, H. S. Mahmassani, and P. Jaillet, "Impacts of auction settings on the performance of truckload transportation marketplaces," *Transportation Research Record*, no. 1906, pp. 89–96, 2005.
- [18] C. Lindsey, A. Frei, H. Mahmassani, Y. Park, D. Klabjan, M. Reed, G. Langheim, and T. Keating, "Predictive analytics to improve pricing and sourcing in third-party logistics operations," *Transportation Research Record*, vol. 2410, no. 2410, pp. 123–131, 2014.
- [19] R. S. Tibben-Lembke and D. S. Rogers, "Real options: Applications to logistics and transportation," *International Journal of Physical Distribution & Logistics Management*, vol. 36, no. 4, pp. 252–270, 2006.
- [20] R. Giusti, D. Manerba, G. Bruno, and R. Tadei, "Synchromodal logistics: An overview of critical success factors, enabling technologies, and open research issues," *Transportation Research Part E: Logistics and Transportation Review*, vol. 129, no. July, pp. 92–110, 2019.
- [21] R. Mason and C. Lalwani, "Transport integration tools for supply chain management," *International Journal of Logistics Research and Applications*, vol. 9, no. 1, pp. 57–74, 2006.
- [22] C. Pani, P. Fadda, G. Fancello, L. Frigau, and F. Mola, "A data mining approach to forecast late arrivals in a transhipment container terminal," *Transport*, vol. 29, no. 2, pp. 175–184, 2014.
- [23] F. Feng, Y. Pang, G. Lodewijks, and W. Li, "Collaborative framework of an intelligent agent system for efficient logistics transport planning," *Computers & Industrial Engineering*, vol. 112, pp. 551–567, 2017.
- [24] J. Mulder, W. van Jaarsveld, and R. Dekker, "Simultaneous Optimization of Speed and Buffer Times with an Application to Liner Shipping," *Transportation Science*, vol. 53, no. 2, pp. 365–382, 2019.

- [25] A. Tufano, R. Accorsi, A. Gallo, and R. Manzini, "SIMULATION IN FOOD CATERING INDUSTRY . A DASHBOARD OF PERFORMANCE," in *International Food Operations and Processing Simulation Workshop*, pp. 20–27, 2018.
- [26] G. Baruffaldi, R. Accorsi, L. Volpe, and R. Manzini, "A data architecture to aid life cycle assessment in closed-loop reusable plastic container networks," *Procedia Manufacturing*, vol. 33, pp. 398–405, 2019.
- [27] G. Baruffaldi, R. Accorsi, L. Volpe, R. Manzini, and F. Nilsson, "Sustainable operations in reusable food packaging networks," *Sustainable Food Supply Chains*, pp. 293–304, 2019.
- [28] H. Moon and H. Lee, "Mixed pooling of seasonality in time series pallet forecasting," 2019.
- [29] Q. Fu, L. Liu, and Z. Xu, "Port resources rationalization for better container barge services in Hong Kong," *Maritime Policy and Management*, vol. 37, no. 6, pp. 543–561, 2010.
- [30] R. Konings, "Network Design for Intermodal Barge Transport," 2003.
- [31] R. Accorsi, G. Baruffaldi, R. Manzini, and A. Tufano, "On the design of cooperative vendors' networks in retail food supply chains: a logistics-driven approach," *International Journal of Logistics Research and Applications*, vol. 21, no. 1, pp. 35–52, 2018.
- [32] R. Accorsi, E. Ferrari, R. Manzini, and A. Tufano, "An application of a vessel route planning model to the import / export of seasonal products," in *26thASOR National Conference for the Australian Society of Operations Research and Defence Operations Research Symposium (ASOR/DORS) 2018*, 2018.

Draft Version 1.0

# 16

## Distribution System Design

This chapter deals with the design of a distribution network (or the redesign of an existing supply chain). Controlling a supply chain focuses on the description of the processes and the day-by-day planning; on the other hand, the design of a network profoundly modify the distribution strategy for many actors of the chain. The design activities are classified depending on the decision patterns (see section 14.3) involved into:

1. location-allocation problem, i.e. clustering points of demand of the network into groups without exceeding the capacity of the resources assigned to each group;
2. network topology design, i.e. the definition of service routes within each cluster;
3. route frequency design, i. e. defining the frequency of service for each node of the network;
4. service time windows design, i.e. defining the time interval when each node of the network should be served;
5. shipping priority definition, i.e. identify dispatching rules for HUs.

These problems can be seen in the perspective of a hierarchical procedure to design a distribution network from scratch. First of all, for each terminal  $j$  of the network, it is necessary to investigate:

1. its position (i.e. the longitude, and the latitude);
2. its demand;

3. its serving infrastructure is identified (e.g. road, rail, water).

Then, the location-allocation problem LAP (1) identifies the number of resources, their capacity, and their location to satisfy the demand of the nodes. The design of the routeing topology (2) assign each terminal  $j$  to a route  $e$  travelled by a vehicle  $v$  connecting the terminal to one of the resources (e.g. production plant or storage system) identified by the LAP. The design of the route frequency (3) matches the demand of the terminal, and the capacity of the resources identifying the number of visits per unit of time of a vehicle  $v$ , on a route  $e$ . Service time windows design (4) identify how to set the loading/discharging time windows for each route  $e$ , at each terminal  $j$ . Shipping priority definition (5) identifies how to organise the loading/discharging operations at the terminal to maximise the service level of the network. Figure 16.1 illustrates the hierarchy of these decision problems, identifying whether the decision belongs to the network or to the single terminals.

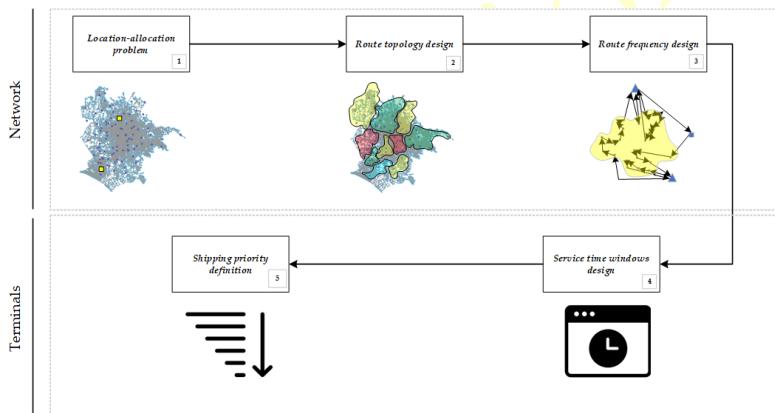


Figure 16.1: Hierarchy of decision problems for distribution network design.

## 16.1 Location – allocation problem (P6)

Location allocation problem (LAP) is a classic operations research problem defined as follows. Given:

- the  $n$  nodes of a distribution network;
- the requirements for each node;
- the shipping cost;

Determining:

1. a number  $m$  of resources to place
2. where to place each resource;
3. the capacity needed by each resource.

The capacity may be the inventory level of a storage system or the throughput of a production plant. The problem definition is general and links together the position on a graph with an amount of capacity.

### 16.1.1 Model-driven methods (PS3)

LAP problem requires a model to be solved since it is hard to collect observation on different realisations of a LAP solution. For this reason, we only approach the problem using a model-driven approach. One of the first approaches to the LAP problem sharply identify two main cost function to model and minimise to obtain the optimal solution of the problem [1]. Let us consider the cost functions:

$$C^{PLANT} = f(m) \quad (16.1)$$

$$C^{DIST} = g(m) \quad (16.2)$$

The total cost to minimise is given by  $E = C^{PLANT} + C^{DIST}$  where:

- $C^{PLANT}$  represents the cost connected to opening a number of resources  $m$  (e.g. the cost of the land, the building, the depreciation, the energy, the direct labour);
- $C^{DIST}$  represents the cost connected to serving the  $n$  nodes of the network using the  $m$  selected facilities (e.g. the cost of distribution).

From this perspective, the LAP problem represents a generalisation of the facility location problem introduced in section 22.2 where the optimal facility location is the coordinate where  $C^{DIST}$  has a minimum.

From a mathematical perspective, the optimal solution to the LAP problem is given by:

$$\frac{dE}{dm} = \frac{df(m)}{dm} + \frac{dg(m)}{dm} = 0 \quad (16.3)$$

Solving equation (16.3) in the variable  $m$  provides the optimal solution to the LAP problem. Unluckily, solving equation (16.3) is computationally expensive, and there is no guarantee on how functions  $f$ , and  $g$  have been defined. They are generally not linear, and hard to estimate. Thinking of moving on their surface (if a continuous surface exists) is science fiction.

Nevertheless, in practice, both  $C^{PLANT}$ , and  $C^{DIST}$  are easy and fast to estimate for a finite set of solutions of the LAP. A smart approach is to

evaluate the total cost  $E_\chi$  for a finite number of scenarios  $\chi$ . Often these scenarios already exist in the mind of the decision-makers. A comparison between these scenarios immediately leads to the most profitable one.

### 16.1.2 Application

We show an example of an application of the LAP problem by comparing a network in two alternative scenarios: as-is scenario describing the current situation of the network, and to-be scenario evaluating an alternative. The comparison method evaluates the differential cash flows in the two scenarios in a Montecarlo fashion.<sup>1</sup>

A cost model is defined to identify which cost items are relevant to compare the two scenarios, for example:

- the direct labour cost;
- the cost of logistics and distribution;
- the operating costs (e.g. the energy);
- the fixed costs to operate the plant;
- the depreciation cost;
- the investment cost (only for the to-be scenario).

The sum of these cost items (except for the depreciation cost) determined the cash flow in each scenario. The difference between the to-be and the as-is scenario determines the differential cash flow. Extending this analysis to a significant time horizon (e.g. ten years) identify the payback period of the to-be investment. When a probability distribution is given for all the cost items identified above, it is possible to determine the risk of the investment using a Montecarlo simulation. The Montecarlo simulation identifies different behaviour, randomly generating numbers from the distribution of the input cost items. Figure 16.2 illustrates the Montecarlo approach and the static approach applied to sample data. This approach can be used to evaluate multiple LAP alternatives compared to an existent scenario identifying which alternative produce an adequate return on the investment.

## 16.2 Network topology design (P2)

Defining the topology of a network consists of grouping terminals  $j$  together, such that a single facility can serve all of them without exceeding its ca-

---

<sup>1</sup>An example of this application is available [here](#).

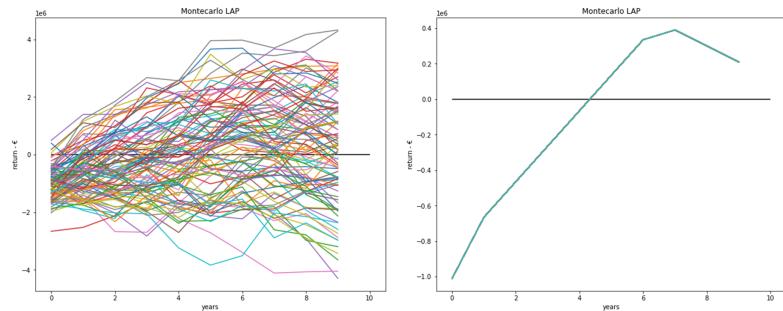


Figure 16.2: Montecarlo and static simulations to evaluate the cash flows of a LAP configuration compared to the existent scenario.

pacity. For example, assigning a distribution centre to a set of points of demand provides a solution to this problem.

This problem is similar to the LAP, but it has a different focus. The LAP determines the number, the location, and the capacity of the facilities to open. On the other side, the network topology design assigns terminals to facilities minimising the service cost and without exceeding the capacity of each facility.

### 16.2.1 Model-driven methods (PS1)

Operations research provides models for the capacitated facility-location problem to approach the network topology problem. These models can be thought of adaptations of a set covering problem where the terminals  $j$  are the points to be covered by the service of the facility. The basic models have the parameters illustrated in Table 16.1.

Parameter	Description
$b_k$	capacity of facility $k$
$f_k$	cost of operations for facility $k$
$d_j$	demand of terminal $j$
$c_{jk}$	cost to serve terminal $j$ from facility $k$

Table 16.1: Parameters of the capacitated facility-location problem.

The model has the following variables.

$$y_k = \begin{cases} 1 & \text{if facility } k \text{ is activated} \\ 0 & \text{otherwise} \end{cases} \quad (16.4)$$

$$x_{jk} = \begin{cases} 1 & \text{if terminal } j \text{ is served by facility } k \\ 0 & \text{otherwise} \end{cases} \quad (16.5)$$

The problem has the following objective function:

$$\min \sum_k f_k y_k + \sum_j \sum_k c_{jk} x_{jk} \quad (16.6)$$

Subjected to the linear constraints:

$$\sum_k x_{jk} = 1, \forall j \quad (16.7)$$

$$x_{jk} \leq y_k, \forall j, \forall k \quad (16.8)$$

$$\sum_j d_j x_{jk} \leq b_k y_k, \forall k \quad (16.9)$$

$$y_k, x_{jk} \in \{0, 1\}, \forall k, \forall j \quad (16.10)$$

Constraints (16.7) impose that all terminal  $j$  must be assigned to a facility; constraints (16.8) links the variables  $x$  and  $y$ ; constraints (16.9) imposes the respect of the capacity of each facility; constraints (16.10) check the integrality of the decision variables.

This model is easy to understand but has many problems in production. The problem is NP-complete, and the number of decision variables  $x_{jk}$  is exponential. For these reasons, a branch & bound algorithm may take forever to solve a real instance of the problem.

### 16.2.2 Data-driven methods (PS2)

Data-driven methodologies involve unsupervised learning (see chapter 8 that clusters observations into groups given their similarity). Clustering techniques effectively solve the network topology problem since they are computationally efficient. These techniques are uncapacitated. For this reason, we present five approaches to introduce the feasibility check on the capacity parameter. We assume having a number of facilities  $m$  with a fixed capacity  $C$ , equal for all the facilities. All the approaches consider similar input parameters, illustrated in Table 16.2.

Parameter	Description
$q_{ij}$	demand of terminal $i$
$C$	Capacity of the facility

Table 16.2: Parameters of the data-driven methods for the route topology design.

### Capacitated similarity tree (CST)

This algorithm builds clusters using a similarity tree produced by hierarchical clustering (see section 8.2.2). Hierarchical clustering aggregates the  $m$  observations relying on a proximity matrix  $m \times m$ , describing how close the observations are. To cluster terminals, we consider a similarity matrix  $S$  whose entries  $s_{ij}$  measure the closeness of two terminals.  $S$  is calculated as the inverse of the matrix of the distances (e.g. the road distance) between nodes. At each step of the algorithm  $s_{ij}$  are ordered and the two nodes with the maximum  $s_{ij}$  are clustered together if the capacity constraint is respected; otherwise the following  $s_{ij}$  is considered for clustering. The algorithm starts creating  $m$  clusters. This number is progressively reduced grouping nodes together when the capacity is respected. Algorithm 11 il-

lustrates the pseudocode for this algorithm.<sup>2</sup>

---

**Algorithm 11:** Capacitated similarity tree (CST)

---

```

 $i = 1, \dots, m \in V$  set of vertices
 $q_i$  demand of vertex  $i$ 
 $C$  maximum capacity of a cluster
 $s_{i,j}$  similarity between vertices  $i$ , and  $j$ 
for  $k \leftarrow 1 : (m - 1)$  do
    | set  $S = \emptyset$ 
    |  $found = false$ 
    | while  $not found$  do
        |   |  $v = \max_{(i,j) - S} (s_{i,j})$ 
        |   |  $(h, l) = \arg(v)$ 
        |   | if  $(q_h + q_l \leq C)$  then
        |   |   |  $found = true$ 
        |   | else
        |   |   |  $S = S \cup (h, l)$ 
        |   | end
    | end
    |  $q_h = q_h + q_l$ 
    | for  $r \leftarrow 1 : m$  do
        |   |  $s_{r,h} = \min(s_{r,h}, s_{r,l})$ 
        |   |  $s_{h,r} = \min(s_{h,r}, s_{l,r})$ 
        |   |  $s_{r,l} = -1$ 
        |   |  $s_{l,r} = -1$ 
    | end
end

```

---

### Construction heuristics on a similarity tree (CHST)

This algorithm works slightly differently from the previous. This algorithm still works on a similarity tree but generating one cluster at a time. When a cluster is full (no residual capacity is left), the algorithm opens a new cluster. A maximum allowable distance  $D$  is introduced as a parameter of the algorithm, to avoid that points too “far” enter the same cluster.

---

<sup>2</sup>The source code of Algorithm 11 is available [here](#).

Algorithm 12 illustrates the pseudocode for this algorithm.

---

**Algorithm 12:** Construction heuristics on a similarity tree (CHST)

---

```

 $i = 1, \dots, m \in V$  set of vertices
 $q_i$  demand of vertex  $i$ 
 $C$  maximum capacity of a cluster
 $d_{i,j}$  distance between vertices  $i$  and  $j$ 
 $D$  maximum allowable distance within a cluster
 $R = V$ 
 $incumbentCluster = false$ 
while ( $R$  not empty) do
    if ( $not\ incumbentCluster$ ) then
        |  $Z = \emptyset$ 
    end
     $S = \emptyset$ 
     $v = \min_{(i,j) - S} (d_{i,j})$ 
     $(h, l) = \arg(v)$ 
    if ( $q_h + q_l \leq C$  AND  $d_{hl} \leq D$ ) then
        |  $Z = Z \cup (h, l)$ 
        |  $R = R - (h, l)$ 
        | for  $r \leftarrow 1 : m$  do
            |   |  $s_{r,h} = \min(s_{r,h}, s_{r,l})$ 
            |   |  $s_{h,r} = \min(s_{h,r}, s_{l,r})$ 
            |   |  $s_{r,l} = -1$ 
            |   |  $s_{l,r} = -1$ 
        | end
    | else
        |   |  $S = S \cup (h, l)$ 
        |   | if ( $S == R$ ) then
        |   |   |  $incumbentCluster = false$ 
        |   | end
    | end
end

```

---

### Iterative k-means covering algorithm (KCOV)

This algorithm uses the well-known  $k$ -means algorithm (see section 8.2.1) as a generator of columns of a set covering problem (SCP) (see section 13.3.1). The problem uses the parameters  $c_j$  to identify the cost of a column, and the

following variable.

$$x_j = \begin{cases} 1 & \text{if column } j \text{ selected} \\ 0 & \text{otherwise} \end{cases} \quad (16.11)$$

The problem is defined as follows.

$$\min \sum_{j=1}^n x_j c_j \quad (16.12)$$

$$\sum_{j:i \in j} x_j \geq 1, i \in V \quad (16.13)$$

$$x_j \in \{0, 1\}, j = 1, \dots, n \quad (16.14)$$

A column  $j$  has value '1' at position  $i$  if terminal  $i$  is included in the cluster; otherwise, its value is '0'. The cost  $c_j$  of a column is a measure of saturation and it is calculated at  $C - \sum_{i \in j} q_i$ . Columns are generated by iteratively running the  $k$ -means algorithm with increasing values of  $k$  (starting from 1 to the number of considered terminals). Only clusters whose capacity falls within a given range are selected as columns of the SCP problem.  $k$ -means stops increasing when a feasible solution of the SCP exists. The SCP is then optimally solved using branch & bound algorithm. Algorithm 13 presents the pseudocode of this algorithm.

---

**Algorithm 13:** Iterative  $k$ -means covering algorithm (KCOV)

---

$k = s, \dots, K$  = neighborhood of the number of centroids

$r$  = number of replicates

$i = 1, \dots, m \in V$  set of vertices

$q_i$  demand of vertex  $i$

$D_i \in \mathbb{R}^n$  set of coordinates of point  $i$

$C$  maximum capacity of a cluster

$c$  minimum allowable capacity of a cluster

$S = \emptyset$

**for**  $k = 1 : m$  **do**

**for**  $j = 1 : r$  **do**

$s$  = solution of K-means( $k$ )

**for** cluster  $g \in s$  **do**

$cap = \sum_{i \in g} q_i$

**if**  $c \leq cap \leq C$  **then**

$S = S \cup s$

**end**

**end**

**end**

**end**

---

**Variable neighbourhood search K-means covering algorithm (VK-COV)**

Draft Version 1.0

This algorithm works as the previous with a pre-selection of the value of  $k$  (i.e. the number of clusters created by the k-means algorithm). All the values of  $k$  (i.e., from 1 to the number of terminals) are tested, and it chooses the values of  $k$  generating the highest fraction (e.g. the 95%) of feasible columns (respect with the capacity constraints). Algorithm 14

illustrates the pseudocode of this algorithm.

---

**Algorithm 14:** Variable neighbourhood search  $k$ -means covering algorithm (VKCOV)

---

$k = s, \dots, K$  = neighborhood of the number of centroids  
 $r$  = number of replicates  
 $i = 1, \dots, m \in V$  set of vertices  
 $q_i$  demand of vertex  $i$   
 $D_i \in \mathbb{R}^n$  set of coordinates of point  $i$   
 $C$  maximum capacity of a cluster  
 $c$  minimum allowable capacity of a cluster  
 $S = \emptyset$   
**for**  $k = 1 : m$  **do**  
  **for**  $j = 1 : r$  **do**  
     $s =$  solution of K-means( $k$ )  
    **for** cluster  $g \in s$  **do**  
       $cap = \sum_{i \in g} q_i$   
      **if**  $c \leq cap \leq C$  **then**  
         $S = S \cup s$   
      **end**  
    **end**  
  **end**  
**end**  
 $\delta =$  select  $k \in s, \dots, K$  generating 95 percentile of feasible solutions  
 $S = \emptyset$   
**for**  $k \in \delta$  **do**  
  **for**  $j = 1 : r$  **do**  
     $s =$  solution of K-means( $k$ )  
    **for** cluster  $g \in s$  **do**  
       $cap = \sum_{i \in g} q_i$   
      **if**  $c \leq cap \leq C$  **then**  
         $S = S \cup s$   
      **end**  
    **end**  
  **end**  
**end**

---

### Optimal column generation heuristics (CG)

This algorithm uses an optimal approach to generate the columns of the SCP presented in paragraph 16.2.2. The columns are generated while they provide a benefit (i.e. a reduced cost) of the objective function, identifying the closeness of the points within the clusters. A column is generated if:

- it has a positive profit;
- it respects the capacity constraints;
- it respects a distance constraint on the maximum distance of each arc.

The primal model of the column generation problem is defined as follows. Table 16.3 illustrates the parameters of the problem.

Parameter	Description
$w_i$	demand of item $i$
$d_a$	distance of arc $a = (i, j)$
$W$	service capacity of a route
$D$	maximum allowable distance between the nodes of a route

Table 16.3: Parameters of primal problem.

The set of the columns of the primal problem is

$$S = \left\{ s \subseteq \{i = 1, \dots, m\} : \sum_{i \in s} w_i \leq W, \max_{a: i, j \in s} d_a \leq D \right\} \quad (16.15)$$

where  $c_s$  is the cost of serving the set  $s$ . The primal problem has the following variable.

$$x_s = \begin{cases} 1 & \text{if configuration } s \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \quad (16.16)$$

The model is defined as follows.

$$\min c_s x_s \quad (16.17)$$

$$\sum_{s \in S: i \in s} x_s \geq 1, \forall i = 1, \dots, m \quad (16.18)$$

$$x_s \in \{0, 1\} \quad (16.19)$$

The dual problem is defined as follows.

$$\max \pi_i \quad (16.20)$$

$$\sum_{i \in S} \pi_i \leq c_s, \forall s \in S \quad (16.21)$$

$$\pi_i \geq 0 \quad (16.22)$$

The column generation problem find violated dual constraints by solving an optimisation column generation problem with variable

$$\mu_i = \begin{cases} 1 & \text{if } i \text{ belongs to column} \\ 0 & \text{otherwise} \end{cases} \quad (16.23)$$

The model is defined as follows.

$$\max \sum_{i=1}^m \mu_i (\pi_i - c_i) \quad (16.24)$$

$$\sum_{i=1}^m \pi_i w_i \leq W \quad (16.25)$$

$$\mu_i \geq z_a, \forall a = (i, j) \quad (16.26)$$

$$\mu_j \leq z_a, \forall a = (i, j) \quad (16.27)$$

$$\mu_i + \mu_j - 1 \leq z_a, \forall a = (i, j) \quad (16.28)$$

$$z_a d_a \leq D, \forall a = (i, j) \quad (16.29)$$

$$\pi_i, z_a \in \{0, 1\} \quad (16.30)$$

### 16.2.3 Application

We test the five algorithms mentioned above in the field of urban logistics while designing the waste collection service of a wide urban area. The problem involves clustering point of waste production (POWP) together, to assign them to vehicles.<sup>3</sup>

We define three indicators to evaluate the performance of the algorithm from a computational point of view (see Table 16.4). The average number of cluster is a measure of the compactness of the outcome clusters. It gives an idea of which algorithm produces, on average, a higher number of classes. It is important to remark that this indicator cannot be directly used as a logistics indicator since it may not be related to the travelled distance within a cluster. The optimal solution of the bin-packing problem (BPP) is calculated, to measure the efficiency of the algorithm in clustering points in the minor number of cluster possible. This solution defines the lower bound of the number of clusters to create. For each algorithm, we measure the

---

<sup>3</sup>An example of this application is available [here](#).

gap between its solution and the BPP solution. Besides, we measure the average solving time and the percentage of failing of the algorithms (i.e., the percentage of the instances when an algorithm is not able to find a feasible solution). Since CST and CHST are fully heuristic algorithms, they always provide a feasible solution in a relatively short time.

Algorithm	Average N of Clusters	Average Gap from BPP	Average Solving Time (sec)
CST	9,34	1,01	5,23
CHST	7,69	0,89	3,31
KCOV	11,35	1,36	1730,89
VNKCOV	10,76	1,36	1442,97
CG	21,99	3,15	363,44

Table 16.4: Performance of the clustering algorithms.

The logistics performance of each algorithm is evaluated, first, matching the number of clusters (generated by an algorithm) with the travelled distance (calculated *a posteriori* after solving the TSP problem for each cluster). Figure 16.3 presents these analyses showing a dot for each cluster generated by the algorithm in all the 498 instances. The figure shows the number of clusters generated by each algorithm on the  $x$ -axis of each subplot while the  $y$ -axis indicates the distance travelled (expressed in km) to serve the POWPs belonging to that cluster. Besides, the colour indicates the number of POWPs in each cluster according to the colour bar. CST and CHST have similar behaviours, but CHST tends to create fewer clusters where more populated clusters account for a higher travelled distance. KCOV and VNKCOV are extremely similar. CG creates clusters whose travelled distance increase with the number of clusters created. Only CG presents a correlation between the number of clusters and the travelled distance. Considering the number of clusters created, the hierarchical algorithms (i.e. CST and CHST) outperform the others since they serve all the nodes with a smaller number of clusters.

Figure 16.4 investigates the correlation between the number of nodes per service area and the travelled distance resulting from the clustering obtained with different algorithms.

As the figure shows, in principle, it always exists a correlation between the number of nodes to serve and the distance travelled to connect them. Nevertheless, different algorithms allow obtaining very diverse performance. The performance is measured using the mean value (red line) of the distance travelled to serve each cluster. The blue line identifies the linear regression of the points. CST has a lower average distance value than CHST. Nevertheless KCOV and VNKCOV produce denser clusters with a lower travelled distance. CG, in this case, does not lead to good performance. This can be partly explained thinking at the objective function of the CG algorithm that

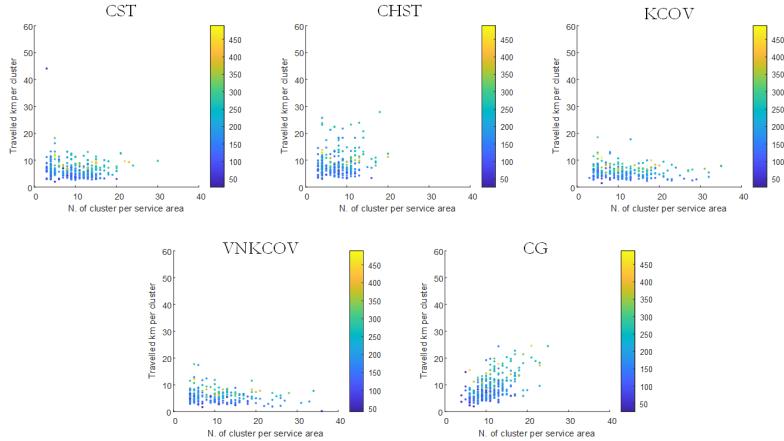


Figure 16.3: Scatterplot of the number of cluster per area (x-axis), and travelled km (y-axis).

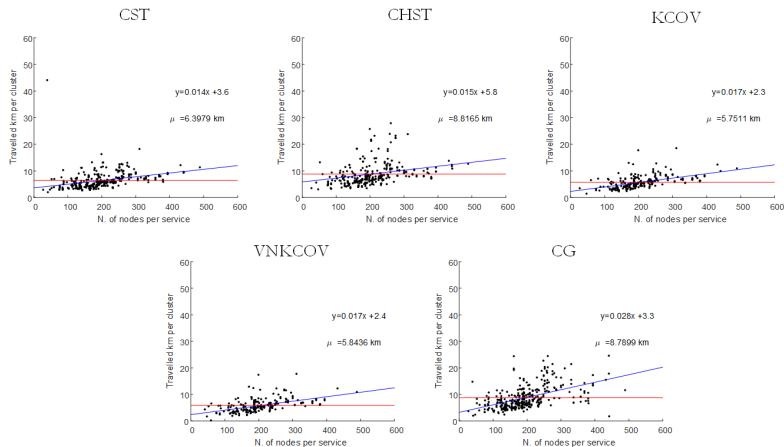


Figure 16.4: Scatterplot of the number of nodes (x-axis), and km (y-axis) one chart for each algorithm.

maximises the saturation of the trucks considering the travelling distance as a constraint (and not as an objective function).

Another important logistic indicator is the working time needed to perform a service. This amount of time must match the estimate of time needed to perform operations within each area of service. To get a measure of the feasibility in time of the solution proposed by the algorithms, historical data are considered to assign the time performance to the trav-

elled distance identified by the route design. An average collection speed of 10 km/h is considered and a fixed time of 30 minutes is considered for the operations at the collection points. Given these values, the time effect produced by the algorithms is statically computed. Figure 16.5 presents the histogram of the expected time necessary to serve each area of service with the solution proposed by the clustering algorithms. The feasibility in time of the routes is considered based on the existing working shift of 6 hours. CST and CHST provide a higher percentage of time-feasibility (66,6 and 76,8 respectively). About half of the solutions provided by KCOV, VNKCOV and CG are feasible in time while MDSCOV mainly does not fit the available time of the existing working shifts.

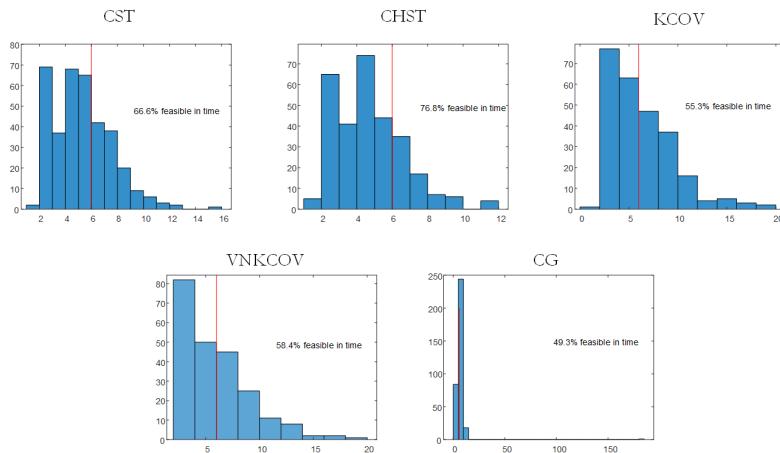


Figure 16.5: Histogram of the estimated service time per area.

## 16.3 Route frequency design (P5)

The design of the frequency of a route defines the capacity of the network, and also the saturation of the vehicle. This problem is approached using models to prescribe the power of each route of a network.

### 16.3.1 Model-driven methods (PS4)

Models to define the frequency of a route can be static or dynamic. When no information about the dynamic behaviour of the demand is available, a static model should be chosen. Static models always work by considering:

- $a_v^e$ , the capacity of the vehicle  $v$ , serving route  $e$ ;

- $b_j$ , the demand of a terminal  $j$ .

The minimum frequency of the route  $e$ , served by a vehicle  $v$  can be statically calculated as:

$$n_j = \left\lceil \frac{b_j}{\sum_{j \in e} a_j} \right\rceil \quad (16.31)$$

When the decision-maker has information on the dynamics of the demand, e.g. its seasonality, a discrete event simulation approach can be preferred to the static approach. Many models adapt real-time the capacity of a route. It is the case of bucket brigades [2], that adds resources to a route only when these resources are needed.

## 16.4 Service time windows design (P5)

Time windows design involves the definition of an optimal time horizon to visit all the terminals of a route  $e$ . Similarly to route frequency, this problem is approached by using models.

### 16.4.1 Model-driven methods (PS4)

Let consider the problem to define a time windows for a terminal  $j$  to be visited by a vehicle  $v$  [3]. We consider a random variable  $X$  describing the distribution of arrivals of a vehicle, where  $f$  and  $F$  are its probability distribution function (PDF) and cumulative distribution function (CDF), respectively. The problem is to identify a time window  $\tau \pm \Delta$  when the vehicle is allowed to visit the terminal  $j$ . A cost of earliness  $C^-$  is associated with an early arrival (e.g. corresponding to the cost of wait of the vehicle) as well as a cost of tardiness  $C^+$  is associated with a delay in the arrival (e.g., the waiting cost for the terminal). We assume that the cost of a delay is greater than the cost of early arrival ( $C^+ > C^-$ ), and we introduce  $C_r$ , the cost of the terminal resource per unit of time. Figure 16.6 introduces  $f(X)$ , with the associated costs.

Based on these parameters, we calculate:

$$\text{Prob}(X < \tau - \Delta) = F(\tau - \Delta) \quad (16.32)$$

$$\text{Prob}(X \geq \tau + \Delta) = 1 - F(\tau + \Delta) \quad (16.33)$$

Consequently, three cost items are defined as:

$$C_{\text{earliness}} = C^- \times F(\tau - \Delta) \quad (16.34)$$

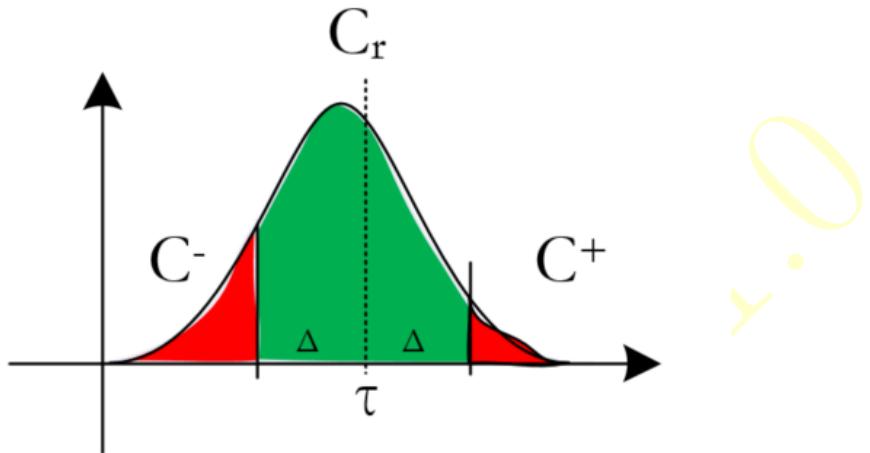


Figure 16.6: Probability distribution function of the arrival of a vehicle with the associated costs.

$$C_{lateness} = C^+ \times [1 - F(\tau + \Delta)] \quad (16.35)$$

$$C_{resource} = 2C_r \times \Delta \quad (16.36)$$

$$C_{tot} = C_{earliness} + C_{lateness} + C_{resource} \quad (16.37)$$

We need to find  $\tau$  and  $\Delta$  such that  $C_{tot}$  is minimised. We introduce the partial derivatives of the cost, and look for the values of  $\tau$ , and  $\Delta$  where the derivatives equal zero.

$$\mu_i = \begin{cases} \frac{\partial C_{tot}}{\partial \tau} = 0 \\ \frac{\partial C_{tot}}{\partial \Delta} = 0 \end{cases} \quad (16.38)$$

$$\frac{\partial C_{tot}}{\partial \tau} = C^- \times f(\tau - \Delta) - C^+ \times f(\tau + \Delta) \quad (16.39)$$

$$\frac{\partial C_{tot}}{\partial \Delta} = C^- \times f(\tau - \Delta) + C^+ \times f(\tau + \Delta) - 2C_r \quad (16.40)$$

Let assume  $X$  being normally distributed:  $\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}$ . Here it follows the development of the equation (16.39).

$$\begin{aligned}
 C^- \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-((\tau-\Delta)-\mu)^2}{2\sigma^2}} &= C^+ \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-((\tau+\Delta)-\mu)^2}{2\sigma^2}} \\
 \frac{C^-}{C^+} e^{\frac{-((\tau-\Delta)-\mu)^2}{2\sigma^2}} &= e^{\frac{-((\tau+\Delta)-\mu)^2}{2\sigma^2}} \\
 \log\left(\frac{C^-}{C^+} e^{\frac{-((\tau-\Delta)-\mu)^2}{2\sigma^2}}\right) &= \log\left(e^{\frac{-((\tau+\Delta)-\mu)^2}{2\sigma^2}}\right) \\
 \log\left(\frac{C^-}{C^+}\right) + \log\left(e^{\frac{-((\tau-\Delta)-\mu)^2}{2\sigma^2}}\right) &= \frac{-((\tau+\Delta)-\mu)^2}{2\sigma^2} \\
 \log\left(\frac{C^-}{C^+}\right) + \frac{-((\tau-\Delta)-\mu)^2}{2\sigma^2} &= \frac{-((\tau+\Delta)-\mu)^2}{2\sigma^2} \\
 2\sigma^2 \ln\left(\frac{C^-}{C^+}\right) - [(\tau-\Delta)^2 + \mu^2 - 2\mu(\tau-\Delta)] &= \\
 &\quad [(\tau+\Delta)^2 + \mu^2 - 2\mu(\tau+\Delta)] \tag{16.41} \\
 2\sigma^2 \ln\left(\frac{C^-}{C^+}\right) - [\tau^2 + \Delta^2 - 2\Delta\tau + \mu^2 - 2\mu(\tau-\Delta)] &= \\
 &\quad [\tau^2 + \Delta^2 + 2\Delta\tau + \mu^2 - 2\mu(\tau+\Delta)] \\
 2\sigma^2 \ln\left(\frac{C^-}{C^+}\right) + 2\Delta\tau + 2\mu(\tau-\Delta) &= -2\Delta\tau + 2\mu(\tau+\Delta) \\
 2\sigma^2 \ln\left(\frac{C^-}{C^+}\right) &= -4\Delta\tau - 2\mu\tau + 2\mu\Delta + 2\mu\tau + 2\mu\Delta \\
 2\sigma^2 \ln\left(\frac{C^-}{C^+}\right) &= -4\Delta\tau + 4\mu\Delta \\
 \frac{\sigma^2}{2} \ln\left(\frac{C^-}{C^+}\right) &= \Delta(\mu - \tau)
 \end{aligned}$$

For practical reason, the additional variable  $u = \frac{\sigma^2}{2} \ln\left(\frac{C^-}{C^+}\right)$  is considered, such that:

$$u = \Delta(\mu - \tau) \tag{16.42}$$

Equation (16.42) is now used to solve the equation (16.40). Please note that from the definition, the term  $u$  is a parameter (i.e., it is not function of the variables  $\tau$  and  $\Delta$ ). Here it follows the development of the equation

(16.40).

Draft Version 1.0

$$\begin{aligned}
 & C^- \times f(\tau - \Delta) + C^+ \times f(\tau + \Delta) - 2C_r \\
 = & C^- \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-((\tau-\Delta)-\mu)^2}{2\sigma^2}} + C^+ \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-((\tau+\Delta)-\mu)^2}{2\sigma^2}} - 2C_r \\
 & C^- e^{\frac{-((\tau-\Delta)-\mu)^2}{2\sigma^2}} + C^+ e^{\frac{-((\tau+\Delta)-\mu)^2}{2\sigma^2}} = 2C_r\sigma\sqrt{2\pi} = \star \quad (16.43) \\
 & C^- e^{\frac{-[(\tau-\Delta)^2 - \mu^2 - 2\mu(\tau-\Delta)]}{2\sigma^2}} + C^+ e^{\frac{-[(\tau+\Delta)^2 - \mu^2 - 2\mu(\tau+\Delta)]}{2\sigma^2}} = \star \\
 & C^- e^{\frac{-[\tau^2 + \Delta^2 - 2\tau\Delta + \mu^2 - 2\mu\tau\Delta + 2\mu\Delta]}{2\sigma^2}} + C^+ e^{\frac{-[\tau^2 + \Delta^2 + 2\tau\Delta + \mu^2 - 2\mu\tau\Delta - 2\mu\Delta]}{2\sigma^2}} = \star
 \end{aligned}$$

Let be  $h = \tau^2 + \Delta^2 + \mu^2 - 2\mu\tau$

$$\begin{aligned}
 C^- e^{\frac{-[h-2\tau\Delta+2\mu\Delta]}{2\sigma^2}} + C^+ e^{\frac{-[h+2\tau\Delta-2\mu\Delta]}{2\sigma^2}} &= \star \\
 C^- e^{\frac{-[h+2\Delta(\mu-\tau)]}{2\sigma^2}} + C^+ e^{\frac{-[h-2\Delta(\mu-\tau)]}{2\sigma^2}} &= \star \\
 C^- e^{\frac{-[h+2u]}{2\sigma^2}} + C^+ e^{\frac{-[h-2u]}{2\sigma^2}} &= \star \\
 C^- e^{\frac{-[h+2u]}{2\sigma^2}} + C^+ e^{\frac{-[h+2u-4u]}{2\sigma^2}} &= \star \\
 C^- e^{\frac{-[h+2u]}{2\sigma^2}} + C^+ e^{\frac{-[h+2u]}{2\sigma^2}} e^{\frac{4u}{2\sigma^2}} &= \star \\
 e^{\frac{-[h+2u]}{2\sigma^2}} \left\{ C^- + C^+ e^{\frac{4u}{2\sigma^2}} \right\} &= 2C_r \sigma \sqrt{2\pi} \\
 e^{\frac{-[h+2u]}{2\sigma^2}} &= \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \\
 \frac{-[h+2u]}{2\sigma^2} &= \log \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) \\
 h &= -2\sigma^2 \log \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u \tag{16.44} \\
 \tau^2 + \Delta^2 + \mu^2 - 2\mu\tau &= -2\sigma^2 \ln \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u \\
 \Delta^2 + \tau^2 + \mu^2 - 2\mu\tau &= -2\sigma^2 \ln \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u \\
 \Delta^2 + (\mu - \tau)^2 &= -2\sigma^2 \ln \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u \\
 \Delta^2 + \frac{u^2}{\Delta^2} &= -2\sigma^2 \ln \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u \\
 \Delta^4 - \Delta^2 \left( -2\sigma^2 \ln \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u \right) &= -u^2 \\
 \Delta^2 \left( \Delta^2 + 2\sigma^2 \log \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) + 2u \right) &= -u^2
 \end{aligned}$$

The equation admits up to four solutions. But the equation  $\Delta^2 = -u^2$  is discarded since the solution belongs to the imaginary space. The remaining solutions are given by the followings.

$$\Delta^2 = -u^2 - 2\sigma^2 \ln \left( \frac{2C_r \sigma \sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u \tag{16.45}$$

$$\Delta = \pm \sqrt{-u^2 - 2\sigma^2 \ln \left( \frac{2C_r\sigma\sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u} \quad (16.46)$$

Only the positive value of  $\Delta$  is taken into account. Finally, considering that:

$$u = \frac{\sigma^2}{2} \ln \left( \frac{C^-}{C^+} \right) \quad (16.47)$$

The optimal values for  $\tau$ , and  $\Delta$  are obtained as

$$\Delta = \sqrt{-u^2 - 2\sigma^2 \ln \left( \frac{2C_r\sigma\sqrt{2\pi}}{C^- + C^+ e^{\frac{4u}{2\sigma^2}}} \right) - 2u} \quad (16.48)$$

$$\tau = \frac{\Delta\mu - u}{\Delta} \quad (16.49)$$

We apply these results using an academic proof of concept. Let consider  $X$ , being normally distributed with parameters  $X \sim N(60, \sigma)$  and  $C^+ = 5$ ;  $C^- = 4$ ;  $C_r = 0.3$ . We may vary the value of  $\sigma$  to perform a sensitivity analysis on the results. Figure 16.7 illustrates the results of the simulation. The first box shows the shape of  $f$ , varying the value of  $\sigma$ . The other boxes illustrate the values of  $\tau$ ,  $\Delta$ , and the total cost. Realistic values exist until  $\sigma$  has real value. Imaginary values are plotted with value '-1'. The example shows that the cost increases while the uncertainty (i.e.  $\sigma$ ) increases. The value of  $\tau$  increases when the uncertainty increases since we assume a higher cost of the delay than the cost of earliness. The span of the time window  $\Delta$  increases up to a critical value of  $\sigma$  where the span of  $\Delta$  is large enough that the cost of the terminal resources dedicated to the vehicle equals the cost of lateness plus the cost of tardiness.

## 16.5 Shipping priority definition (P7)

The definition of shipping priority is the rationale to assign HUs to vehicles. Usually, shipping companies associate HUs to vehicles based on a FIFO rationale in a push fashion. As soon as an order is available, as soon it has to be assigned. Shipping priority rules answer the question: may we have some benefit by waiting before assigning a HU to a vehicle. The answer to this question is very firm-dependent and product-dependent.

When a product is perishable, it is wise to immediately deliver it to provide the consumer with a higher level of service. When a shipping firm has due date to comply with (e.g. deliver within 24, 48, 72 hours) is it appropriate to organise shipping respect with this due dates. In general,

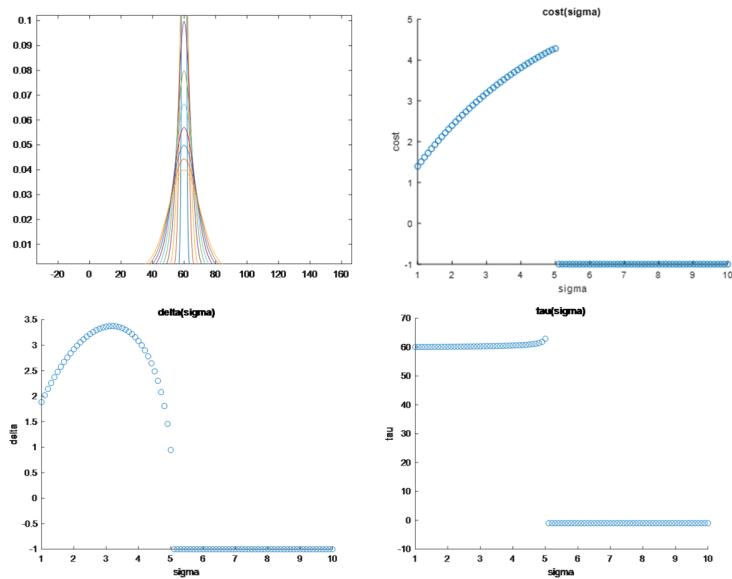


Figure 16.7: Results of the optimal time windows design.

the more a HU can wait, the higher the probability of obtaining a better saturation of the vehicle (e.g. loading last-minute orders), see section 15.5.2.

## Bibliography

- [1] L. Cooper, “Location-Allocation Problems,” *Informs*, vol. 11, no. 3, pp. 331–343, 1963.
- [2] J. J. Bartholdi, D. D. Eisenstein, and Y. F. Lim, “Bucket brigades on in-tree assembly networks,” *European Journal of Operational Research*, vol. 168, no. 3, pp. 870–879, 2006.
- [3] R. Zuidwijk, “Ports and global supply chains,” in *Ports and Networks: Strategies, Operations and Perspectives*, 2017.

Draft Version 1.0

# Part IV

# STORAGE SYSTEMS



# 17

## Diagnostic Models

This section analyses storage systems (also known as warehousing systems). The scientific field analysing storage system is warehouse science [1]. This topic is profoundly important in the management of a supply chain since warehouses are the first source of inefficiencies in a supply chain. The assignment of a proper inventory level in the warehouses of a supply chain network is the first step to smooth the flows and improve the operations of the entire chain.

Warehouses work as decoupling points between other supply chain nodes [2, 3]. Their role is unavoidable since they enhance the flexibility of the entire supply chain by separating the demand and the production rates. We often hear of *zero inventory* as the central paradigm of lean thinking. In truth, it is impossible to build a supply chain without inventory. Contrarily, it is absolutely possible, and always recommendable, to design supply chain able to keep the inventory levels under control (that is the real meaning of the *zero inventory* philosophy).

Nevertheless, the stock is a cost with, or without the control on the level of the inventory. Warehouses do not add any value to the product, but they always lead to costs. For this reason, it is necessary to pay close attention to the design and control of warehouses in order to avoid inefficiencies.

Warehouses retain all the ingredients for a data-driven approach. Operations within warehouses are pretty repetitive, and even the warehouse layout is easy-to-model; it is composed of many standard blocks working the same way. Besides, for traceability reasons, warehouses produce tons of data to keep track of put-away, picking operations, and inventory positions.

This chapter focuses on the definition of the keywords and key entities extending the ontology of Chapter 3 to storage nodes. After that, we introduce the diagnostic framework for production nodes using a relational data

structure and a dashboard of KPIs. A non-relational data structure is proposed as well, to enhance data-driven features by combining the information sources of multiple storage systems.

Chapter 18 focuses on data- and model-driven methods to control a storage node, while chapter 19 does the same focusing on the design methods.

## 17.1 Ontology

Section 3 introduces the general ontology of this book. This paragraph develops that ontology by applying its keys and metrics to a warehouse system.

### Entities

We identify the following entities:

**Part (i)**: it is a storage unit, better known as a stock keeping unit (SKU). An SKU is usually:

- a part, i.e. a single product;
- a carton, i.e. a (secondary) package, containing many parts, or
- a pallet, i.e. a (tertiary) package, containing many cartons.

**Processing node (j)**: it is a storage location. Each area able to host one or more SKUs is a processing node of the storage system.

**Edge (j, k)**: paths connecting the storage locations (the aisles of the warehouse) are the edges. Generally, aisles can be travelled:

- traversal (using a single direction);
- return (using both the directions).

Traversal policy implies that the graph of the storage system is directed. Otherwise, return policy implies an undirected graph.

**Vehicle (v)**: a vehicle is any unit able to travel on edges and load or unload parts (e.g. a forklift).

**Consumable (s)**: the energy to feed the vehicles, and the resources of a storage system.

**Route (e)**: the routes in a storage system are the set of edges travelled in sequence by a vehicle to complete a put-away or picking job. Routes are usually not predefined and can continuously change depending on the set of orders.

**Order (o)**: it is a set of products to put away or pick received by the customer

**Job (b):** it is the sequence of locations to visit and the quantity of SKUs to move to complete an order. A picking/put-away list is a job.

**System network:** the graph  $G(V, A)$  of nodes  $j \in V$  and edges  $(j, k) \in A$  composing the warehouse system.

## Metrics

We identify the following metrics to assess the performances of a processing node  $j$ .

**Throughput ( $TH_j$ ):** the put-away or picking rate of a set of locations. (parts per hour).

**Work in process ( $WIP_j$ ):** it is the number of parts stored in a location  $j$ .

**Work in process ( $WIP_{jk}$ ):** it is the number of parts being transported by a vehicle  $v$ .

**Capacity ( $C_j$ ):** it is the maximum storage capacity of a storage location  $j$ , usually measured in quantity or volume ( $dm^3$ ).

**Capacity ( $C_v$ ):** it is the maximum number of part transportable by a vehicle  $v$  at the same time.

**Utilisation ( $U_j$ ):** it is the average saturation of the space of a storage location.

**Utilisation ( $U_v$ ):** it is the average fraction of non-empty space on a vehicle.

**Lead time ( $LT_e$ ):** it is the time allocated to perform a route (i.e. to complete a picking or put-away list).

**Cycle time ( $CT_e$ ):** it is the average time from the receipt of an order to the completion of the associated picking list (job).

**Service level ( $SL_e$ ):**  $Prob\{cycle\ time \leq lead\ time\}$

## Information functions

Finally, we define the three information functions introduced 3.3.1: movements  $M$  are referred to put-away (IN) and picks (OUT) of SKUs to or from a storage location  $j$ ; the inventory  $I$  is referred to the number of SKUs stored in a storage location  $j$ . The productivity  $P$  refers to the inbound and outbound throughput of a set of locations. Usually, a set contains locations close to each other, depending on the warehouse technology (e.g. manual or automated) offering a different  $TH_j$  and with different productivity patterns. Table 17.1 summarises the definition of the three functions in a storage system.

Information function	Description
Movements $M_i^{j,v}(t)$	This function defines the put-away and picking of parts from/to a storage location $j$ using a vehicle $v$ .
Inventory $I_j(t)$	This function defines the inventory level on a location $j$
Inventory $I_v(t)$	This function defines the work in process on a vehicle $v$
Productivity $P_j^{IN}(t^*)$	This function defines the inbound productivity (e.g. part per hour) of a (set of) location(s).
Productivity $P_j^{OUT}(t^*)$	This function defines the outbound productivity (e.g. part per hour) of a (set of) location(s).

Table 17.1: Definition of the information functions of a storage system.

## 17.2 Data Structure

This paragraph introduces an ER model to describe the operations of a storage system. Warehouses operations are similar for different nodes and enterprises, and despite their level of automation, it is possible to define a rigid model able to identify the aspects of value from a logistics perspective. For this reason, the relational database infrastructures are still the most widely used in the industry [4].

### 17.2.1 A relational model for warehousing systems

The ER model has a table for each of the relevant aspects of a storage system:

1. the SKUs;
2. the storage locations;
3. the vehicles;
4. the inventory position;
5. the movements.

#### SKUs

The SKUs table contains all the information from the SKU master file. The features of the SKUs are relevant to identify the proper storage area for each SKU, depending on its dimensions, weight, volume, inflammability, shelflife, and many other industry-oriented features. For these reasons, the SKU table identifies:

- the id of the item;

- the description of the item;
- the category of the item (e.g. the commercial category);
- the id of the product family of the item;
- the temperature profile of the item (if any);
- the production route of the item (if any);
- the package route of the item (if any);
- the volume and weight of the item;
- the size (i.e. length, height, width) of the item;
- the supplier of the item.

The description, size volume and weight are the most important information to design the storage system and to identify the proper location of the item. Other information as the supplier, the category, the product family may be of interest to place similar items in different zones of the storage system. The production and the package flow are important information for the picker who has to place the item on the correct flow (e.g. in a packing area) before the shipping.

### Storage locations

Storage locations are the areas where each single SKU is stored and waits to be picked. Many information is tracked for each location to enhance the traceability and to speed up the put-away and picking process (in particular, the search of a specific location). For each location, this table keeps track of:

- the id of the storage system (i.e. the building);
- the id of the logical warehouse (i.e. a subset of locations that are logically but not necessarily physically separated by the others);
- the id of the location;
- the coordinate of the aisle which serves the location ( $x$  coordinate);
- the coordinates of the location ( $x, y, z$  coordinates);
- the number of the rack;
- the number of the bay;
- the number of the level;

- the id of the area where the location is placed;
- the category of the vehicle serving the location;
- the type of the package (part/carton/pallet) stored in the location.

This information is beneficial to analyse the flows generated by the warehouse on its plant layout. Nevertheless, the majority of the warehouse management systems (WMS) does not keep track of the physical coordinates of storage locations, creating a significant lack of data for warehouse control and improvement.

### Vehicles

The vehicles table identifies the macro category of vehicles handling SKUs within a warehouse. A non-exhaustive list comprises:

- manual operator (i.e. walking through aisles and shelves);
- forklift serving racks;
- forklift serving floor stack;
- man-on-board picking vehicle;
- vertical warehouse (e.g. Modula);
- miniload;
- automated storage & retrieval system (AR/RS).

### Inventory

This table defines the inventory level of each storage location at a specific date defining the quantity of the item codes stored in that location. Its attributes are:

- the id of the storage system (i.e. the building) inherited from the table locations;
- the id of the logical warehouse (i.e. a subset of locations that are logically but not necessarily physically separated by the others) inherited from the table locations;
- the id of the location inherited from the table locations;
- the id of the item inherited from the SKU table;
- the timestamp when the inventory is recorded;

- the number of parts of the item code above stored in the location.

The inventory information defines the state of a storage location at a precise timestamp.

### Movements

The movements table tracks each movement, i.e. put-away or picking activity modifying the state of a storage location (i.e. its storage level). The table records the following attributes:

- the id of the storage system (i.e. the building) inherited from the table locations;
- the id of the logical warehouse (i.e. a subset of locations that are logically but not necessarily physically separated by the others) inherited from the table locations;
- the id of the location inherited from the table locations;
- the id of the item inherited from the SKU table;
- the timestamp when the movement is recorded;
- the id of the order;
- the quantity picked or put away;
- the id of the picking/put-away mission;
- the sign of the movement (e.g. '+' for put-away and '-' for picking)
- the type of the order;
- the id of the picking operator/vehicle;
- the id of the package associated with the SKU.

Figure 17.1 presents the relational model linking all these tables by the definition of relationship and integrity constraints between entities.

#### 17.2.2 A non-relational model for warehousing systems

ER databases are the most used in the design of the WMS. Nevertheless, non-relational structures allow flexibility and easiness of use with benefits applied to the field of warehousing science. In particular, traditional WMS does not store information on the layout coordinates of the storage locations. In addition, many locations are virtual and only exists in the WMS to provide relational integrity without any physical relevance.

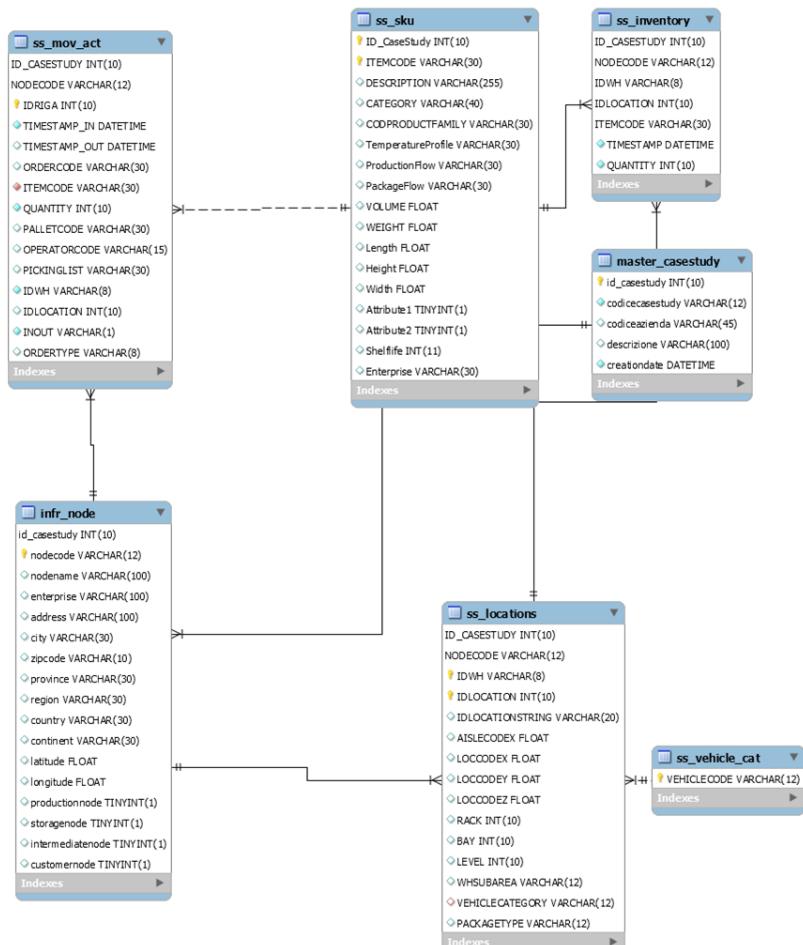


Figure 17.1: ER-model for storage systems.

An important feature to control a storage system is the possibility to measure the distance between storage locations. While dealing with an ER structure, it is possible to define a block-layout identifying:

- the input and output points;
- the presence of traversal aisles;
- a modular distance between storage locations of the same aisle;

- a modular distance between aisles.

Figure 17.2 identifies the layout, and its coordinates identified by these features. The coordinates are identified by using the letter  $a$  for the aisles, the letter  $r$  for the racks, and the letter  $t$  for the traversal aisles.



Figure 17.2: Warehouse layout model.

Unluckily, the application of modular distances does not always correctly represent the distances within a storage system, adding an avoidable bias. There can be multiple storage blocks within the same warehouses using different technologies, or different storage areas whose distances are poorly represented by this modular pattern. To overcome this limitation, we aim at defining a graph  $G(V, A)$  with a vertex  $j \in V$  for each storage location, and a set of arcs  $(i, j) \in A$  representing the available connections with their distance. A non-relational structure is the natural tool to both contain all the information of the relational model, and to save the graph  $G$  to enhance warehouse design and control. We introduce the collections and attributes of the non-relational data structure first. Then we illustrate a method to define the warehouse graph.

The non-relational data structure consists of five collections: *parts*, *storage locations*, *movements*, *inventory* and *graph*. For each collection, a minimal number of attributes defines the minimal viable model (MVM) with the essential information to map the operations of a storage system.

The collection *parts*, is similar to the table SKU of the relational model. It stores the information from the SKUs master file of the storage system. The id of each SKU is the only attribute necessary to have an MVM.

The collection *storage locations* records information on the storage location of the warehouse. The id of the storage location is the only attribute necessary to define the MVM. Additional attributes can be recorded as:

- the coordinates of the location;
- the coordinates of the aisle;
- the list of the type of vehicles serving the location;
- the available capacity of the location;
- the logical warehouse containing the location;
- the inventory function of the storage location.

The collection *movements* records all the activities performed in the warehouse. The necessary attributes of the MVM are:

- the timestamp;
- the id of the part;
- the id of the storage location;
- the id of the order;
- the quantity;
- the sign (i.e. “+” or “-“).

Other attributes that can be recorded are:

- the id of the picking list;
- the id of the operator;
- the weight of the movement;
- the volume of the movement;
- the destination of the movement.

The collection *inventory* stores observation of the inventory position of the warehouse. The MVM is composed by:

- the timestamp;

- the id of the part;
- the id of the location;
- the observed quantity.

Figure 17.3 uses the unified modelling language (UML) to represent the MVM of the non-relational data structure of a storage system.

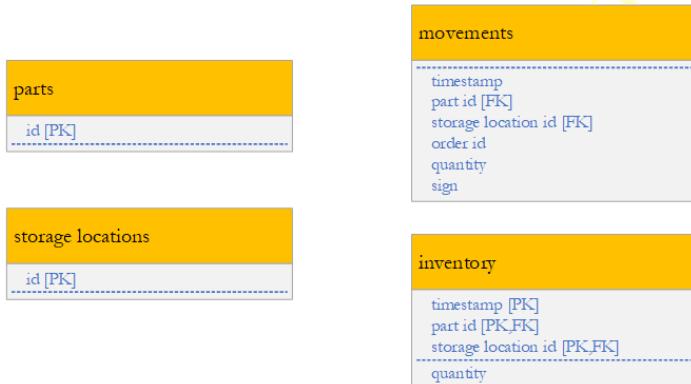


Figure 17.3: Diagram in UML notation of the minimum viable model for a non-relational structure of a storage system.

Finally, a collection graph stores a single graph object  $G(V, E)$ , representing the graph of the storage system, defined using a standard procedure<sup>1</sup>. Algorithm 15 illustrates this procedure applicable to any warehouse dataset. It aims at defining a graph with edges and vertices associated with all the storage locations recorded in the storage location collection. First, the procedure defines the set of vertices  $V$  by considering all the coordinates of the storage locations projected onto the aisles that serve them. Then a set of edges  $E$  is created to connect all these vertices virtualising the path of the aisles of the warehouse. Missing values (e.g. missing coordinates in the storage location collections) are replaced by using linear interpolation

<sup>1</sup>The source code to generate the graph of a storage system is available [here](#).

between the known values.

---

**Algorithm 15:** Definition of the warehouse graph.
 

---

```

Import storage locations  $s \in S$ 
Import I/O locations  $f \in F$ 
Set  $L = S \cup F$ 
Consider  $L_a$ , aisle coordinate for each storage location
Set  $V = L_a(x, y)$ 
Clean the coordinates  $L_a$  with linear interpolation of missing values
if  $F == \emptyset$  then
  | Define a I/O point in ( $\frac{\max\{L_a(y)\} - \min\{L_a(y)\}}{2}, \min\{L_a(x)\}$ )
end
Map fake locations with coordinates in  $F$ 
Set  $E = \emptyset$ 
 $E = E \cup \{\text{vertical edges connecting aligned coordinates}\}$ 
 $E = E \cup \{\text{horizontal on the front and back of the warehouse}\}$ 
 $E = E \cup \{\text{edges connecting the I/O points to the closest node}\}$ 
  
```

---

### 17.3 Decision Patterns

This section aims at defining the set of decision problems for the design and control of a storage system, using the decision patterns identified in 4.2. Problems are classified into:

1. Warehouse design problems, dealing with the design and placement of the physical entities of a storage system;
2. Warehouse control problems, dealing with the assessment and improvement of the performance of an existing storage system.

Different methodologies allow getting feasible solutions to these problems [5, 6, 7, 8]. Table 17.2 illustrates the entities and their definition according to the ontology in 3.1 involved in each decision pattern.

System	Class of the problem	Task	Decision	Entity	Ontology	Descriptive	Explanative	Data Science	Predictive	Precipitive	Decision Science	Methodologies
Warehouse node: P1 - Family problem	Design	Volume, weight, size estimation	Stock Keeping Unit	Part	•	○	○	○	○	○	D1, D2, D3	
Warehouse node: P5 - Power problem	Design	Definition of the inventory level	Stock Keeping Unit	Part	○	○	○	○	●	●	PS4	
Warehouse node: P2 - Technology Assignment problem	Design	Choice of the storage technology	Storage technology	Part, Vehicle	○	○	○	○	●	●	PS2, PS3	
Warehouse node: P5 - Power problem	Design	Storage allocation slotting	Stock Keeping Unit	Part	○	●	○	○	●	●	PS2, PS3	
Warehouse node: P2 - Assignment problem	Design	Storage assignment (Random/ABC/Dedicated)	Stock Keeping Unit	Part, processing node	○	●	○	○	●	●	PS2, PS3	
Warehouse node: P6 - Placement Problem	Design	Layout design (Zoning)	Stock Keeping Unit	Part, processing node	○	●	○	○	●	●	PS1, PS2	
Warehouse node: P7 - Dispatching rules	Design	Picking policy design (Single order/Batching)	Stock Keeping Unit	Part	○	○	○	○	●	●	PS4	
Warehouse node: P3 - Flow problem	Design	Route design (Return/Traversal)	Aisles	Edges	○	○	○	○	●	●	PS3	
Warehouse node: P5 - Power problem	Design	Inbound & Outbound area design	Buffers	Processing node	○	○	○	○	●	●	PS4	
<hr/>												
Warehouse node: P8 - Performance assessment	Control	Performance assessment	Storage system	System Network, Paths, Resources, Vehicles, Jobs, Routes	●	●	○	○	○	○	D1, D2, D3, D4	
Warehouse node: P9 - Workload prediction	Control	Workload forecast	Stock Keeping Unit	Part	○	○	●	○	○	○	PD1, PD2, PD3	
Warehouse node: P10 - Operations management	Control	Vehicle routing	Vehicle	Vehicle, Part	○	●	●	●	●	●	PS1, PS2, PD3	

Table 17.2: Decision problems classification in a storage system.

Twelve decision problems are identified in the design and control of a storage node:

1. Volume, weight, size estimation; it involves the definition or estimation of the attributes of an SKU.
2. Choice of the storage system; it involves the definition of an adequate set of vehicles to serve the storage needs.
3. Forward-reserve design; it involves the design of a reserve area (on the upper levels of a storage system) serving a fast pick area on the floor.
4. Storage allocation; it involves the design of the level of inventories on the fast-pick and reserve area.
5. Storage assignment; it involves the assignment of SKUs to storage locations.
6. Layout design; it involves the design of the areas (i.e. the zones) of the storage system.
7. Picking policy design; it involves the definition of the picking policies (e.g. single-order vs. batching and sorting).
8. Route design; it involves the definition of the direction allowable for each aisle (i.e. return or traversal)
9. Inbound and outbound area design; it involves the definition of the amount of space dedicated to inbound and outbound operations
10. Performance assessment (control); it involves the measurement of the performance of a storage system.
11. Workload forecast (control); it involves the prediction of the workload and workforce needed to perform the operation in the short-, mid- or long-term
12. Vehicle routing (control); it involves the definition of the picking/put-away lists and their assignment to vehicles.

Besides, Table 17.2 specifies which data science technique results adequate to propose a solution to the problem. While descriptive and prescriptive techniques are preferred for control problems, explorative and prescriptive techniques result adequate when dealing with design problems. Chapters 18 and 19 illustrate these techniques in details.

## Bibliography

- [1] J. J. Bartholdi and S. T. Hackman, "Warehouse & Distribution Science," 2017.
- [2] A. Aguezzoul, "Third-party logistics selection problem: A literature review on criteria and methods," *Omega (United Kingdom)*, vol. 49, pp. 69–78, 2014.
- [3] K. Selviaridis and M. Spring, "Third party logistics: a literature review and research agenda," *The International Journal of Logistics Management*, vol. 18, no. 1, pp. 125–150, 2007.
- [4] R. Accorsi, R. Manzini, and F. Maranesi, "A decision-support system for the design and management of warehousing systems," *Computers in Industry*, vol. 65, no. 1, pp. 175–186, 2014.
- [5] G. Cormier and E. A. Gunn, "A review of warehouse models," *European Journal of Operational Research*, vol. 58, no. 1, pp. 3–13, 1992.
- [6] A. E. Gray, U. S. Karmarkar, and A. Seidmann, "Design and operation of an order consolidation warehouse: Models and application," *European Journal of Operational Research*, vol. 58, pp. 14–36, 1992.
- [7] R. Manzini, Y. Bozer, and S. Heragu, "Decision models for the design, optimization and management of warehousing and material handling systems," *International Journal of Production Economics*, vol. 170, pp. 711–716, 2015.
- [8] J. P. V. D. van den Berg and W. H. M. Zijm, "Models for warehouse management: Classification and examples," *International Journal of Production Economics*, vol. 59, no. 1, pp. 519–528, 1999.

Draft Version 1.0

# 18

## Storage System Control

This chapter deals with the control of the operations of a storage system. Controlling the performance of a storage system is a crucial activity to maximise its productivity.

### 18.1 Performance assessment (P8)

This paragraph illustrates the approaches to evaluate the performance of a storage system. The measurement of the performance of a storage system involves a wide number of metrics since storage systems exist at any stage of the supply chain [1, 2, 3, 4].

#### 18.1.1 Model-driven methods (D4)

A storage system works as an intermediate buffer of a supply chain. It receives material flows from suppliers, and generate material flows directed to the customers. Many information flows connect suppliers and customers with the operations of a warehouse. Mapping this flow is a common technique to assess the exchange of materials and information from a qualitative point of view. The Business Process Model and Notation is used to identify entities, flows and responsibilities. When applying the BPMN to a storage system:

- activities are tasks necessary for the storage and handling of SKUs;
- events identify when an activity is terminated (e.g. a put-away or a picking list);

- gateways describe a different variant of the process (e.g. value-added activities as packing, quality control, labelling).
- pools identify the operators and the offices of the storage system, identifying their responsibilities.

BPMN defines a qualitative map of the production processes useful for manager and practitioners to identify the way their processes are realised. More difficult is the assessment of these processes from a quantitative point of view. For this reason, a dashboard of KPIs is introduced, coherently with the ontology of 17.1. The KPIs used in these chapters refers to the problems defined in 4.2. KPIs are organised according to four classes [5]:

1. Logistic KPIs, evaluate the logistic impact of a certain solution. They use metrics like time, distance and the performance parameters introduced in Paragraph 17.1.
2. Cost KPIs, evaluate the economic sustainability of a given solution. They are expressed in € or other currency.
3. Energy KPIs, evaluate the energy needed to feed a given solution. They use metrics as kW and kWh.
4. Environmental KPIs, evaluate the environmental impact of a given solution. They are expressing the equivalent  $CO_2$  produced per year.

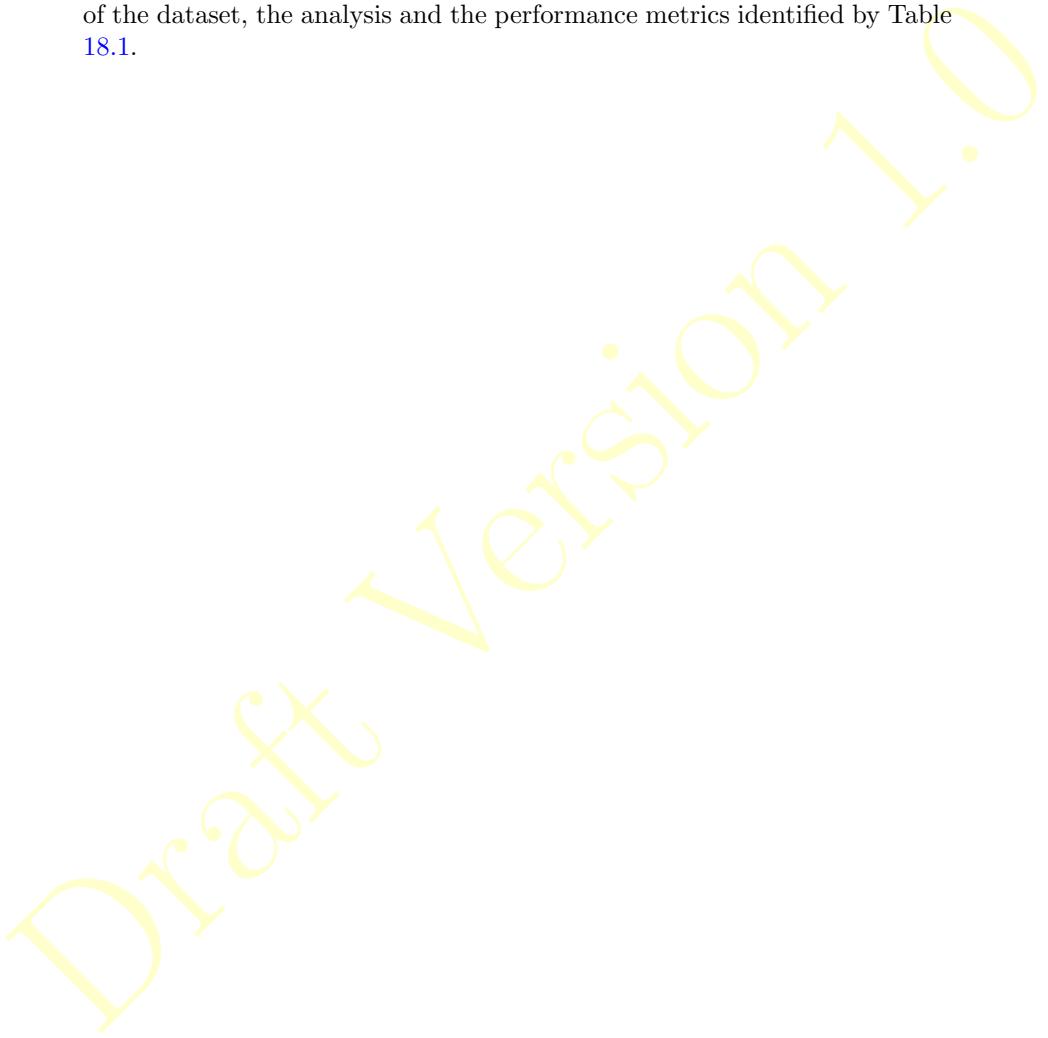
Table 18.1 identifies which KPI is relevant to each problem. In general, each problem can be assessed from multiple perspectives.

System	Class of the problem	Decision	Task	Logistic KPIs	Cost KPIs	Energy KPIs	Environmental KPIs
Warehouse node	P1 - Family problem	Design	Clustering of SKU's into process families	WIP <sub>s</sub> , LTe	○	○	○
Warehouse node	P2 - Technology Assignment problem	Design	Clothes of the storage system	LTe, THj	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P2 - Technology Assignment problem	Design	Forward Reserve Design	LTe, THj	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P5 - Power problem	Design	Storage allocation (EOS, EQJ, OPT)	Uj, C, LOS <sub>s</sub>	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P2 - Assignment problem	Design	Storage assignment (Random ABC/Dedicated)	LTe, THj	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P6 - Placement Problem	Design	Layout design (Zoning)	LOSe, Travelled distance per year (Km/y)	Initial investment (€)	○	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P7 - Dispatching rules	Design	Picking policy design (Single order/Batching)	WIP <sub>s</sub> , LTe, LOS <sub>s</sub>	Storage cost (€/year)	○	○
Warehouse node	P3 - Flow problem	Design	Route design (Return/Traversal)	(Km/year), Uj	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P5 - Power problem	Design	Inbound & Outbound area design	Uj, C, LOS <sub>s</sub>	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P8 - Performance assessment	Control	Performance assessment	CT <sub>w</sub> , U <sub>w</sub> , LOS <sub>w</sub>	Maintenance costs (€/year)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
Warehouse node	P9 - Workload prediction	Control	Workload forecast	CT <sub>w</sub> , U <sub>w</sub> , LOS <sub>w</sub>	Direct labour cost (€/year)	○	○
Warehouse node	P10 - Operations management	Control	Vehicle routing	CT <sub>w</sub> , U <sub>w</sub> , LOS <sub>w</sub>	○	○	○

Table 18.1: KPIs to evaluate the solutions to problems in a distribution network.

### 18.1.2 Data-driven methods (D2, D4)

While the operations of different storage systems have similar patterns, their data can have different organisation and features. The assessment of the performance of a storage system depends on the type and quality of the data recorded. Figure 18.1 illustrates the data pipelines connecting the attributes of the dataset, the analysis and the performance metrics identified by Table 18.1.



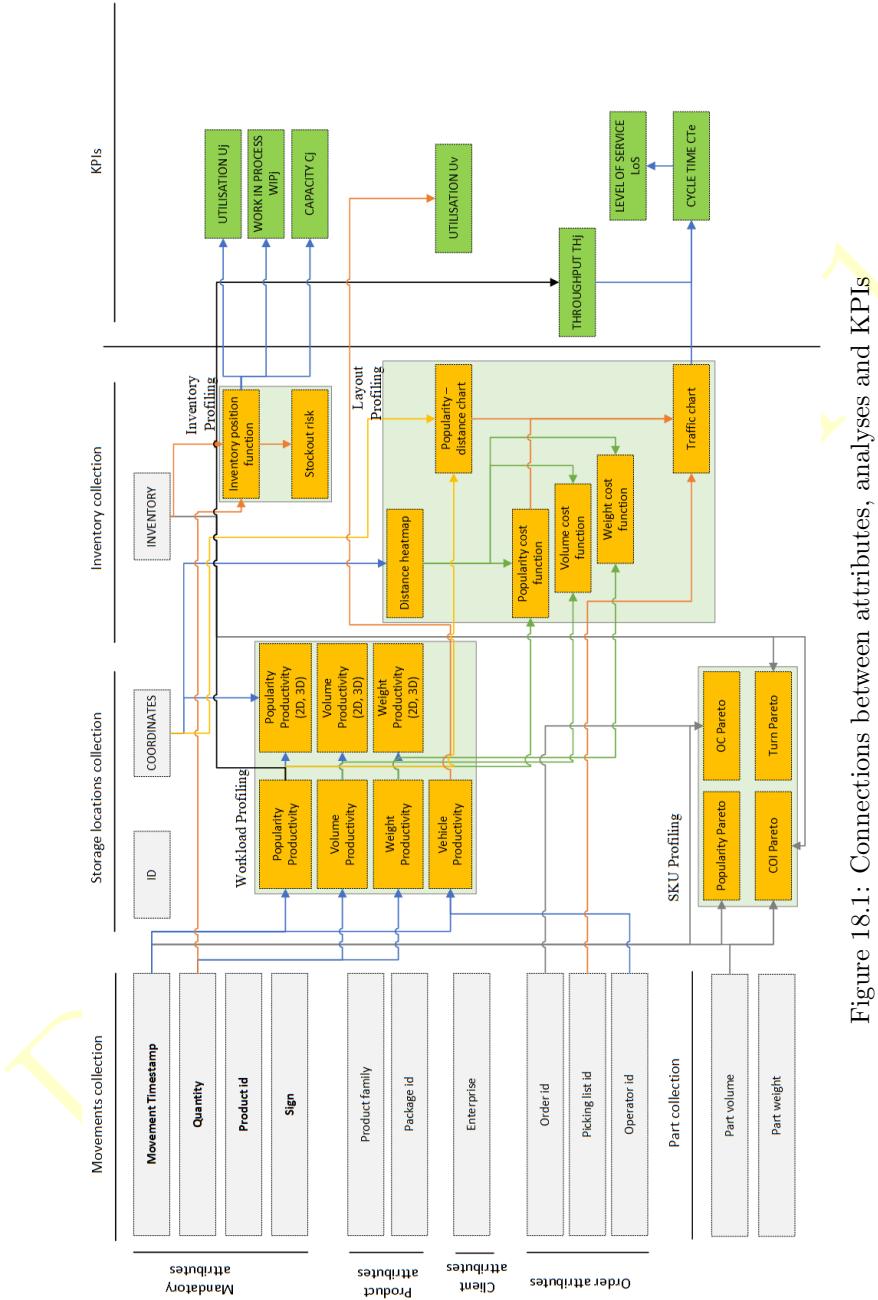


Figure 18.1: Connections between attributes, analyses and KPIs

The analysis (identified by the orange blocks in Figure 18.1) are grouped into four main areas, depending on the entity of the warehouse they affect: the SKUs, the inventory level, the workload, and the layout.

### SKU profiling

Warehouses contain thousands of different SKUs. For this reason, it is necessary to profile them and approach the design and control of the operations differently depending on the SKUs profile.

We introduce four different profiling indexes; some indexes are defined differently for inbound and outbound operations.

- the popularity index  $Pop_i^+, Pop_i^-$  counting the number of put-away/picks of SKU  $i$ ;
- the turn index  $Turn_i = \frac{Qty_i^-}{\overline{I_i(t)}}$ ; where  $Qty_i^-$  is the outbound quantity of SKU  $i$  measured in parts or  $dm^3$ , and  $\overline{I_i(t)}$  is the mean value of the inventory function of part  $i$  measured in parts or  $dm^3$ . The turn measures the rotation of the inventory of a part.
- the cube-per-order index  $COI_i^+ = \frac{Pop_i^+}{\overline{I_i(t)}}, COI_i^- = \frac{Pop_i^-}{\overline{I_i(t)}}$ ; where  $Pop_i^+$ , and  $Pop_i^-$  are the inbound or outbound popularity and  $\overline{I_i(t)}$  is the mean value of the inventory function of part  $i$  measured in  $dm^3$ . The COI mixes the two metrics above to have both a measure of the number of accesses and the space occupied by a part [6, 7].
- the order completion index  $OC_i = \sum_{e:i \in e} \frac{1}{card(e)}$ ; where  $e$  is an order or a picking list. The index measures the relative importance of a part  $i$  to complete a single order.

Mapping all these indexes provide a behaviour of each single SKU. This indexes can be grouped using the Pareto curve, and the frequency analysis to investigate the behaviour of an entire storage system (see Figure 18.2).<sup>1</sup>

---

<sup>1</sup>The source code of Figure 18.2 is available [here](#).

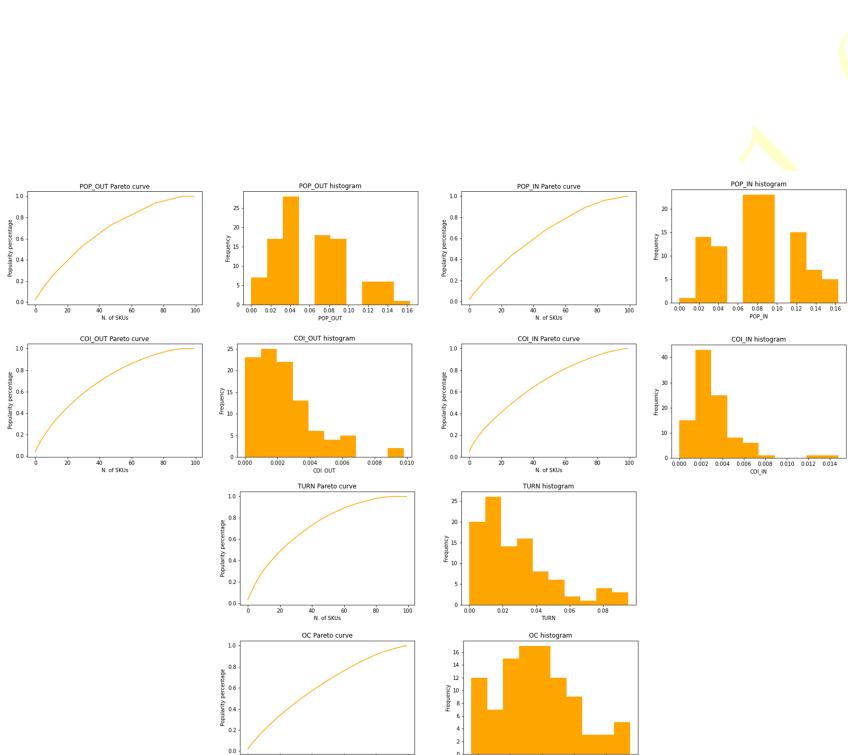


Figure 18.2: Pareto curves and frequency analysis of the indexes to classify the SKUs of a storage system.

### Inventory profiling

Inventory profiling is a fundamental activity to assess the efficiency of a storage system. The space efficiency of a storage system depends on the degree of saturation. The less the available space, the better the utilisation of the storage system. Nevertheless, the higher the saturation of the storage system, the higher the probability to have a lack of space for incoming goods. The design of the inventory level is analysed in paragraph 19.2 and relies on the results of the inventory profile defined here.

The definition of the inventory function  $I_i(t)$  for each SKU  $i$  is a preliminary activity to profile the inventory of a storage system. When the dataset contains an inventory snapshot  $I_i(\tau)$  at time instant  $\tau$ , the inventory is obtained by considering:

$$I_i(\tau + \epsilon) = I_i(\tau) + M_i^+(\tau + \epsilon) - M_i^-(\tau + \epsilon) \quad (18.1)$$

$$I_i(\tau - \epsilon) = I_i(\tau) - M_i^+(\tau + \epsilon) + M_i^-(\tau + \epsilon) \quad (18.2)$$

When there is no inventory snapshot it is possible to use the formulae (18.1), and (18.2) by setting  $I_i(\tau) = 0$ . The estimate of  $I_i(t)$  is obtained by shifting the function to positive values  $I_i(\tau) = I_i(\tau) - \min(I_i(\tau))$  when  $\min(I_i(\tau)) < 0$ .

By summing the inventory functions for all the SKUs, it is possible to measure the saturation of the storage system. It is recommendable to measure  $I(t)$  using  $dm^3$ ; otherwise, the estimate of the global inventory function will not be accurate using the number of parts. Figure 18.3 illustrates the inventory function  $I(t)$  of a storage system, together with the frequency analysis, and the cumulative distribution functions. The cumulative distribution function is used to design the inventory level of the warehouse (see section 19.2). <sup>2</sup>

---

<sup>2</sup>The source code of Figure 18.3 is available [here](#).

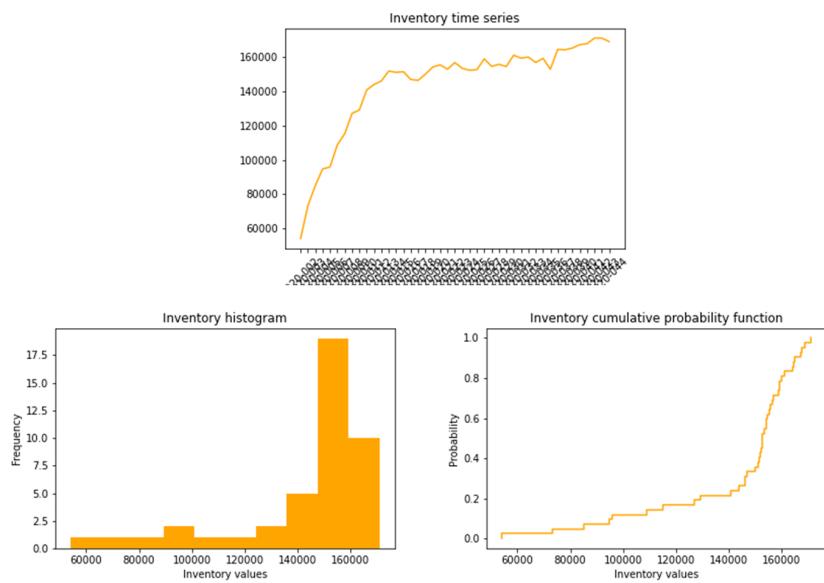


Figure 18.3: Inventory profile of a storage system.

## Workload profiling

Workload profiling aims at assessing patterns of workload per different types of areas or vehicles of the storage system. The workload of a storage system is measurable using:

- the popularity  $Pop_i^+(t)$ ,  $Pop_i^-(t)$ , indicating the number of accesses to part  $i$ ;
- the volume  $Vol_i^+(t)$ ,  $Vol_i^-(t)$ , indicating the put-away or picking volume of part  $i$ , calculated as quantity times volume of a single part;
- the weight  $W_i^+(t)$ ,  $W_i^-(t)$ , indicating the put-away or picking weight of part  $i$ , calculated as quantity times weight of a single part.

The quantity is rarely used as a workload estimator since the picking, or put-away quantities rarely have an effect on the operations. These functions refer to single parts and can be aggregated at any level, defining the workload of:

- the entire storage system;
- a specific area of the storage system;
- a vehicle or picker.

These analyses consider a single independent variable (i.e. the time  $t$ ). By considering the coordinates of the storage locations, it is possible to define multivariate functions to classify the operations of a storage system. Figure 18.4 illustrates the workload of a storage system using a single dimension (i.e. the time), two dimensions (i.e. the length and the depth of the storage system), and three dimensions (i.e. illustrating the workload within the warehouse space).<sup>3</sup>

---

<sup>3</sup>The source code of Figure 18.4 is available [here](#).

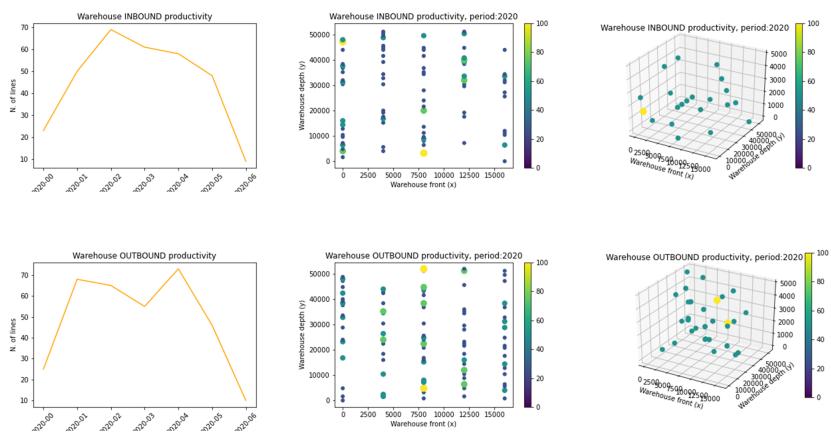


Figure 18.4: 1-dimensional, 2-dimensional, and 3-dimensional popularity workload analysis.

### Layout profiling

Profiling the behaviour of the resources on the layout of a storage system is the key to identify sources of inefficiencies. By considering the rectangular (also known as “city-block”) distance between the I/O area and each single storage location, it is possible to identify which locations are closer to reach. Placing the items with a high number of accesses close to the I/O point may increase the efficiency of a storage system.

By analysing the distance to reach the storage location where an SKU  $i$  is placed, and the popularity of the SKU  $i$ , it is easy to identify which SKUs are far from their optimal location, and generate a long travelling distance that can be avoided with a reassignment of SKUs to storage locations [8].

Finally, it is important to identify where the traffic is located within a storage system. When storage location coordinates are known, it is possible to define a graph  $G(V, E)$  as introduced in 17.2.2. The distance between a storage location and the input and output point is a piece of important information to identify the locations where placing SKUs with the highest popularity indexes (see the chart in the top left panel of Figure 18.5).<sup>4</sup>

By solving the shortest path between the sequence of locations visited by a picking list, it is possible to identify the traffic within the aisles of the storage system. This procedure reveals the travelled distance for each picking list and the part of the aisles where each vehicle travelled. A traffic chart is used to reveal this information (see the chart in the top right panel of Figure 18.5).

The optimisation of a storage system can be obtained by placing the SKUs in a different storage location, to reduce the travelled distance to pick them. The degree of suboptimisation of a storage system is measurable by using a chart plotting the popularity, and the distance from the input and output points for each storage location [8]. The storage system is optimised when the chart is similar to a negative exponential function (see the chart in the bottom right panel of Figure 18.5). Otherwise, the storage system is sub-optimised (see the chart in the bottom left panel of Figure 18.5).

<sup>4</sup>The source code of Figure 18.5 is available [here](#).

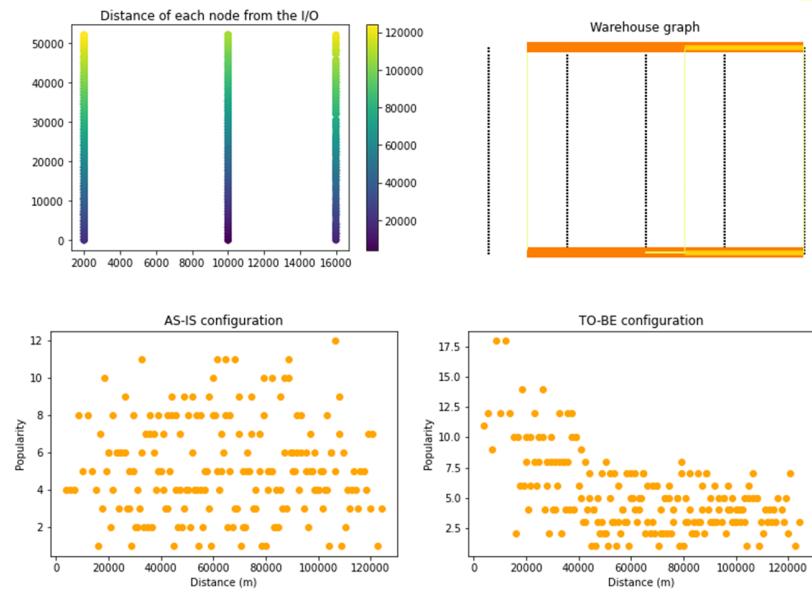


Figure 18.5: Analyses to profile the layout of a storage system.

## 18.2 Workload prediction (P9)

Storage system datasets track both the behaviour of its suppliers and customers by considering put-aways (labelled with the “+” sign), and pickings (labelled with the “-“ sign). For this reason, they have all the data to make forecasts of the workload in the future.

### 18.2.1 Model-driven methods (PD2)

Classic time series methods help to model and predict the workload of the storage system in the future. Usually, the number of movements (i.e. the lines) is much more representative than the quantities (more used in production forecasts) [9].

Storage systems have an additional peculiarity; their inventory function  $I(t)$  is much more inert than the inventory functions of distribution or production system. This is because warehouses exist to stock goods, and they are efficient in space when the value of  $I(t)$  is steadily high. For this reason, when analysing a storage system, it is important to remember the speed at which the inventory rotates (a sort of turn index of the entire warehouse). The following chapters provide methods to analyse and optimise the operations within a warehouse. Nevertheless, it is important to remember that when optimisation means reallocate thousands of SKUs, the cost of this reallocation may be higher than the benefit of the optimisation. For this reason, we introduce the concept of the frequency of  $I(t)$ , to evaluate the best season to optimise a storage system, and even the expected life of the efficiency from the optimisation.

### 18.2.2 Data-driven methods (D1, PD1)

This section introduces data-driven methods to forecast the value of the key variables of a storage system in the future. In particular, a metric of the cost of a storage system can be estimated by the time to put-away/pick a line of a picking list. Multiple tasks compose this time, depending on the type of product and process of the warehouse. A non-exhaustive list comprises:

- travel time;
- search time;
- pick time;
- pack time.

A rigorous time and motion campaign measures all these components on-field. Nevertheless, setting an accurate campaign is costly. Consider that the generalisation of the results of a measurement campaign needs at least

30 samples to apply the central limit theorem. Some tasks are performed rarely, and collecting 30 samples may take forever. Besides, a campaign lasts months, there is the risk that the inventory function  $I(t)$  already “rotated” (see section 18.1.2), and the samples are unrepresentative of the operations.

To overcome these limits, we use a different approach, still rigorous, even if with a lower level of details. Since the WMS always records a timestamp for each operation, it is possible to estimate the starting and the ending time of a picking list. Depending on the attributes of the dataset, it is possible to aggregate quantitative values and to train machine learning models to predict the total time to execute a picking list. For example, the input dataset  $X$ , can be composed of the following attributes for each picking list (each picking list is an observation):

- the number of lines of the picking list;
- the total quantities of the picking list;
- the total volume of the picking list;
- the total weight of the picking list;
- the area of the layout defined by the coordinates of the storage locations in the picking list;
- the enterprises of the SKUs involved in the picking list;
- the logical warehouses of the SKUs involved in the picking list;
- the warehouse technologies of the SKUs involved in the picking list.

This information can be used to define a learning table where each row assesses the value of these metrics of a picking list. Data exploration techniques allow investigating the behaviour of the inbound and outbound operations of a storage system. Figure 18.6 presents the pair plot with the histogram of each metric and the scatterplot of each pair of metrics. Different colours of the scatter plots identify inbound, outbound or other activities.<sup>5</sup>

It is possible to evaluate the correlation between these metrics by using a correlation matrix. Figure 18.7 illustrates the correlation matrices of the inbound and outbound lists.<sup>6</sup>

---

<sup>5</sup>The source code of Figure 18.6 is available [here](#).

<sup>6</sup>The source code of Figure 18.7 is available [here](#).



Figure 18.6: Pairplot of the key metrics to measure the picking lists.

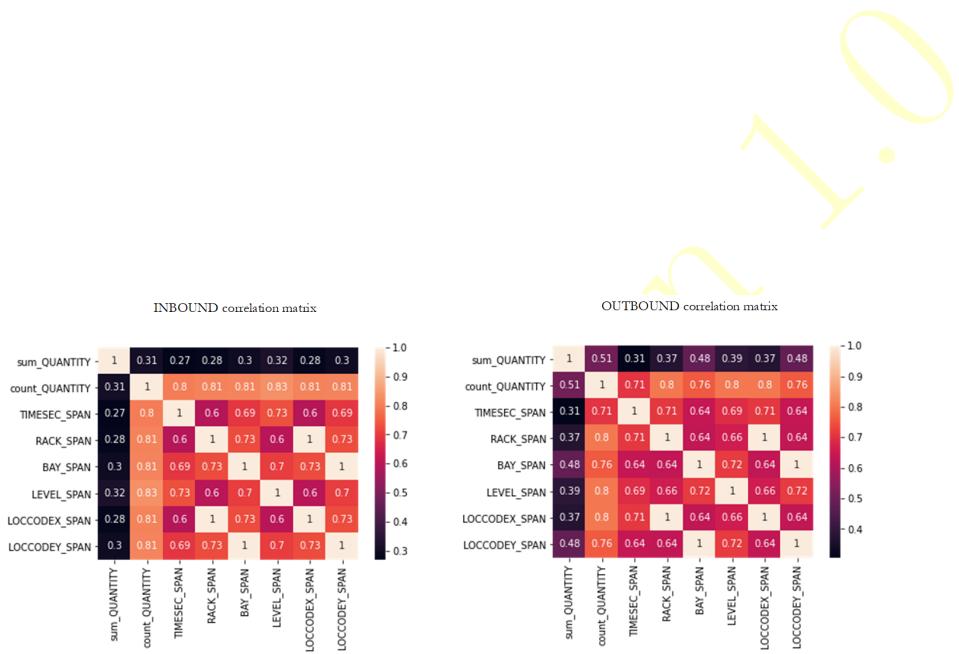


Figure 18.7: Inbound and outbound correlation matrices.

## 18.3 Vehicle routing (P10)

Warehouse day-by-day involves the generation of the picking lists. This problem is equal to routing vehicle of a distribution system or assigning jobs to production machines. It is necessary to group picking activities such that the total travelled distance is minimised. It is a vehicle routing problem (VRP), where the nodes to visit are the storage locations [10].

Despite our introduction already defines a graph  $G(V, A)$  with vertices and distances, sometime VRP within a storage system is more complex than it appears. First of all, the cost of picking is not only linked to the travelled distance. For a warehouse having thousands of SKUs, the cost of picking the wrong SKUs is much higher than travelling a few meters more. This is why random allocation and random picking policy are preferred when companies pay a high cost for a wrong picking (e.g. in the e-commerce).

Besides, VRP works offline, assuming that all the orders are known in advance. This is not always true for a WMS that can be frequently updated during the working shift. Finally, the set  $V$  of the graph  $G$  could have thousands of nodes (one for each storage location) resulting in a vast VRP problem, taking forever to solve.

Sometimes heuristics are used to approach this problem; e.g. by printing a picking list for each zone of the warehouse when zoning is used. Warehouses with traversal directions use a single-direction snake path travelling through all the aisles of the warehouses for each picking list.

Generally, simulation is used to investigate the best picking list generation policy [11, 12, 13, 14]. In truth, this work does not approach the problem since it is highly warehouse-dependent involving, in our opinion, too many parameters to be integrated into a single approach generalisable for any storage system.

## Bibliography

- [1] R. Accorsi, M. Bortolini, M. Gamberi, R. Manzini, and F. Pilati, “Multi-objective warehouse building design to optimize the cycle time, total cost, and carbon footprint,” *International Journal of Advanced Manufacturing Technology*, vol. 92, no. 1-4, pp. 839–854, 2017.
- [2] C. J. Malmborg and K. Al-Tassan, “An integrated performance model for orderpicking systems with randomized storage,” *Applied Mathematical Modelling*, vol. 24, pp. 95–111, 2000.
- [3] R. Manzini, M. Gamberi, A. Persona, and A. Regattieri, “Design of a class based storage picker to product order picking system,” *International Journal of Advanced Manufacturing Technology*, vol. 32, no. 7-8, pp. 811–821, 2007.

- [4] F. H. Staudt, G. Alpan, M. Di Mascolo, and C. M. Rodriguez, “Warehouse performance measurement: A literature review,” *International Journal of Production Research*, vol. 53, no. 18, pp. 5524–5544, 2015.
- [5] A. Tufano, R. Accorsi, A. Gallo, and R. Manzini, “SIMULATION IN FOOD CATERING INDUSTRY . A DASHBOARD OF PERFORMANCE,” in *International Food Operations and Processing Simulation Workshop*, pp. 20–27, 2018.
- [6] J. Heskett, “Cube-per-order index a key to warehouse stock location,” *Transport and Distribution Management*, vol. 3, pp. 27–31, 1963.
- [7] C. J. Malmborg and K. Bhaskaran, “A revised proof of optimality for the cube-per-order index rule for stored item location,” *Applied Mathematical Modelling*, vol. 14, no. 2, pp. 87–95, 1990.
- [8] R. Manzini, R. Accorsi, G. Baruffaldi, D. Santi, and A. Tufano, “Performance assessment in order picking systems : a visual double cross-analysis,” *The International Journal of Advanced Manufacturing Technology*, 2018.
- [9] T. van Gils, K. Ramaekers, A. Caris, and R. B. de Koster, “Designing efficient order picking systems by combining planning problems: State-of-the-art classification and review,” *European Journal of Operational Research*, vol. 0, pp. 1–15, 2017.
- [10] H. D. Ratliff and A. S. Rosenthal, “Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem,” *Operations Research*, vol. 31, no. 3, pp. 507–521, 1983.
- [11] E. P. Chew and L. C. Tang, “Travel time analysis for general item location assignment in a rectangular warehouse,” *European Journal of Operational Research*, vol. 112, no. 3, pp. 582–597, 1999.
- [12] R. W. Hall, “Distance approximations for routing manual pickers in a warehouse,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 25, no. 4, pp. 76–87, 1993.
- [13] B. I. Kim, R. J. Graves, S. S. Heragu, and A. S. Onge, “Intelligent agent modeling of an industrial warehousing problem,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 34, no. 7, pp. 601–612, 2002.
- [14] C. G. Petersen, “The impact of routing and storage policies on warehouse efficiency,” *International Journal of Operations & Production Management*, vol. 19, no. 10, pp. 1053–1064, 1999.

## Storage System Design

This section is about warehouse design, i.e. the design from scratch or the redesign of an existing storage system [1, 2]. This activity involves many decision problems belonging to different classes. In practice, these problems are approached in sequence, enlarging the focus of the decision from stock keeping units (SKUs), to storage areas to handling areas [3, 4, 5, 6, 7, 8]. Figure 19.1 identifies the hierarchical procedure used in this chapter, according to the definition of logistics problems introduced in 4.2. The SKUs are the parts stored within the storage system. The same SKUs have the same properties (e.g. id, weight, volume, package). Storage areas are a set of storage locations of the warehouse having a similar storage technology (e.g. served by forklifts, automated guided vehicles (AGV), automated storage and retrieval systems (AS/RS)). Handling areas are sets of storage areas, processing areas (e.g. packing, quality control, inbound and outbound areas), and edges (i.e. aisles connecting all these areas).

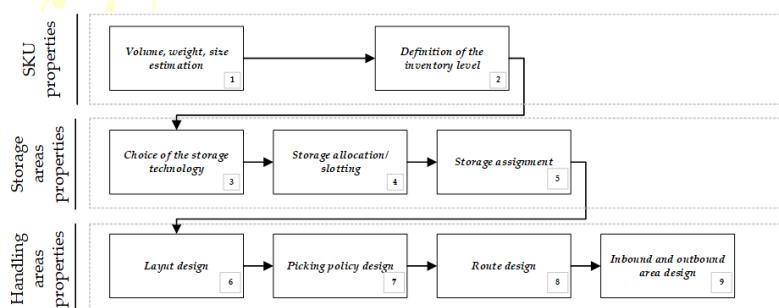


Figure 19.1: Hierarchical procedure for storage system design.

The hierarchical procedure identifies nine decision problems:

1. the estimation of weights and sizes for all the SKUs of the storage system;
2. the definition of the inventory level for each SKU;
3. the choice of the storage technology for each set of SKUs;
4. the allocation of storage space to SKUs;
5. the assignment (also known as “slotting”) of SKUs to storage locations;
6. the design of the layout of a handling area;
7. the definition of the picking policies (e.g. single order/batching/zoning);
8. the definition of the routing policy (i.e. returns; traversal);
9. the design of the inbound and outbound areas.

The following paragraphs introduce model- and data-driven methods to address these problems.

## 19.1 Weight and size estimation (P1)

The knowledge of the features of the SKUs is the first step to design a storage system adequately. It may appear evident that companies must know everything about their parts. Nevertheless, the SKU master file of a warehouse management system (WMS) is easily full of null values. The most important indicator to design a storage system is the volume of the SKUs; this value is often unknown due to the labour-intensive activity to measure the dimensions of thousands of SKUs. In practice, a warehouse designer needs to know the size (i.e. length, height and width), and the weight of an SKU.

### 19.1.1 Model-driven methods (D2)

When sizes and weights are unknown for all the SKUs of a storage system, on-field measurement is the only possibility to start generating this data. A scale is used to measure the weight; while the length, height, and width of an SKU are used to define its volume. When dealing with the measurement of the volume of the SKUs of an existing storage system, an approximate procedure exist. Let  $v_i$  be the volume of SKU  $i$  to measure. We estimate the volume  $\hat{v}_i$  by using  $\hat{v}_i = \frac{1}{\text{card}(j:n_{ij}>0)} \sum_{j:n_{ij}>0} \left[ \frac{V_j \cdot \eta_j}{n_{ij}} \right]$  where  $\eta_j \in [0, 1]$  is

the percentage saturation of each storage location  $j$ , and  $n_{ij}$  is the number of SKUs  $i$  stored in each location  $j$ .  $V_j$  is the volume of the storage location  $j$ . This way, it is possible to measure a single storage location  $j$ , to observe the storage locations to estimate  $\eta_j$ , and to consider the number of SKUs  $n_{ij}$  obtained, for example, from the inventory of the WMS. This procedure is approximated but faster than measuring thousands of SKUs and it provides information on how much the volumes of SKUs  $i$  are different.

### 19.1.2 Data-driven methods (D1)

When a feature (e.g. the volume) is given for a subset of SKUs of the SKU master file, this procedure can be used to extend the properties of the given features to the whole set of SKUs by using clustering [7].

The description of an SKU is always recorded together with its id in the WMS. This is mainly due for avoiding errors since pickers read on their terminal the description of the SKU they have to pick. We use text mining techniques to cluster SKUs based on their description. In particular, we build a bag of words (BOW) model using the descriptions of the SKUs.

A BOW is a frequency analysis on text strings. The BOW model counts the number of occurrences of each word, giving higher importance to the strings occurring the most. It is necessary to clean the input strings separating words (e.g. removing \_ and - characters) and removing special characters (i.e., +, /, —, ", ', . characters), in order to make the BOW model work properly. BOW defines a vocabulary of the storage system (SSV) which contains the one single words or a couple of words occurring the most. It is recommendable to set a threshold on the minimum number of occurrences for a word to enter the SSV (e.g., at least ten occurrences among all the descriptions) to prevent having huge and meaningless SSVs. Finally, each SKU is associated with a set  $B_i$  i.e., containing each word of the SSV contained by the description of the SKU  $i$ .

At this stage, unsupervised learning methods are used to find patterns among SKUs descriptions based on the values of the sets  $B_i$ . A matrix  $M$  is defined to measure the similarity between each couple of SKUs (e.g. SKUs  $i$ , and  $h$ ) based on the value of a similarity index (e.g. the Jaccard index) calculated on  $B_i, B_h$ . A hierarchical clustering algorithm (e.g. complete linkage CLINK, single linkage SLINK or average linkage UPGMA) is applied on  $M$  to define clusters of SKUs. Once clusters are defined, the given properties of an SKU are extended to all the SKUs of a cluster. If the features are known for many SKUs within the same cluster, their value is averaged before extending to the other SKUs.

## 19.2 Definition of the inventory level (P5)

The definition of the level of inventory of a storage system is a crucial decision to reduce the storage costs of the storage system [9, 10, 11, 12, 13, 14, 15, 16, 17, 18]. The definition of the inventory level is based on the statistical analysis of previous observation of the inventory values. For this reason, we only introduce data-driven methods.

### 19.2.1 Data-driven methods (D2, D3)

The inventory level of a warehouse is described by the function  $I(t) = \sum_i I_i(t)$ . This function is crucial since it describes the evolution of the state of the storage system. Unfortunately, WMS usually records only the current inventory of a storage system, losing all the information of the previous states. Luckily, by recording all the movements, this information can be inferred by using the Theorem 1. Let introduce this algorithm to define the inventory function of a part  $i$ <sup>1</sup>.

1. Consider all the inbound (+) and outbound (-) movements of a part  $i$ ;
2. Daily sample the movement, and group by day summing the quantities with their sign.
3. Sort the series by the time, and shift everything to positive values  $I_i(t) = I_i(t) - \min(I_i(t))$ , when  $\min(I_i(t)) < 0$ .
4. For any given inventory point  $I_i^{INV}(t = \tau)$  such that  $I_i^{INV}(\tau) > I_i^{INV}(\tau)$ , try to shift the function up  $I_i(t) = I_i(t) + (I_i^{INV}(\tau) - I_i^{INV}(\tau))$  to correct underestimations of the inventory function.

Once  $I_i(t)$  has been defined, it is possible to use statistics to describe it, and identify the optimal inventory level. For example, by considering the probability distribution function (PDF) of  $I_i(t)$ , and its cumulative distribution function (CDF), it is possible to observe the behaviour of the inventory in the past. The CDF  $F(x)$  identifies the probability of observing an inventory value lower or equal to  $x$ ,  $F(x) = \text{prob}\{I_i < x\}$ . By identifying the inverse of the probability as a risk (see Figure 19.2), the inventory value equal to a given risk is identified:  $I^*(\text{risk} = \rho) = F^{-1}(1 - \rho)$ .

It is necessary to remark that the aforementioned procedure uses one observation of the  $I_i(t)$ , i.e. the only observed for a part  $i$ . By extracting the features of the function  $I_i(t)$  it is possible to bootstrap  $I_i(t)$ , obtaining a probabilistic definition of the inventory behaviour of the part  $i$ .

---

<sup>1</sup>The source code to estimate the inventory of a storage system is available [here](#).

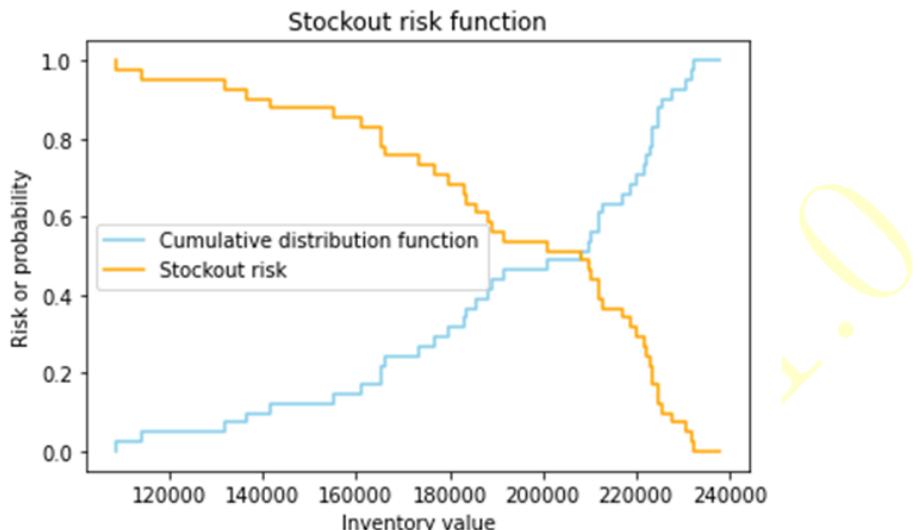


Figure 19.2: Cumulative distribution function of  $I_i(t)$ , and stockout risk function.

The movement function  $M_i(t)$  is defined based on  $I_i(t)$ , and the outbound movements  $M_i^-(t) = M_i(t) : M_i(t) < 0$  are considered. The mean value  $\mu_i^{M^-}$ , and the standard deviation  $\sigma_i^{M^-}$  of the quantity involved in the  $M_i^-(t)$  are defined. The mean interarrival time  $\mu_i^{M_{int}^-}$ , and its standard deviation  $\sigma_i^{M_{int}^-}$  are defined as well. At this stage, Montecarlo simulation is used to bootstrap a multitude of inventory functions  $I_i^{boot}(t)$ . Depending on the demand pattern of the part  $i$  (see section 21.2.1 for the classification), a different approach is used. When the part  $i$  is intermittent, or lumpy a Poisson distribution (see Section 6.3.4) is used to generate the time instant when there is outbound demand with  $\lambda$  equal to the number of movements of  $M_i^-(t)$ :

$$t \sim \text{Poisson}(\lambda = \text{card}\{M_i^-(t)\}) \quad (19.1)$$

When parts are erratic or stable, a Gaussian distribution generates the waiting time between the arrivals, using the parameters from the interarrival distribution:

$$t \sim \text{Normal}(\mu = \mu_i^{M_{int}^-}, \sigma = \sigma_i^{M_{int}^-}) \quad (19.2)$$

The quantities of the  $M_i^{boot}(t)$  are always generated by using a Gaus-

sian function:

$$M_i^{-\text{boot}}(t) \sim \text{Normal}(\mu = \mu_i^{M^-}, \sigma = \sigma_i^{M^-}) \quad (19.3)$$

Algorithm 16 identifies the pseudocode to bootstrap inventory functions.<sup>2</sup>

---

**Algorithm 16:** Bootstrap algorithm for inventory functions.
 

---

Consider the movement function  $M_i(t)$  of a part  $i$

Set  $\mu_M$  to the mean of  $M_i(t)$

Set  $\sigma_M$  to the standard deviation of  $M_i(t)$

Set  $\mu_M^{\text{int}}$  to the mean of the interarrival time of  $M_i(t)$

Set  $\sigma_M^{\text{int}}$  to the standard deviation of the interarrival time of  $M_i(t)$

Set  $B$  to the number of iterations

**for**  $b : 1 \rightarrow B$  **do**

**if** Demand pattern of  $i \in \{\text{'Intermittent'}, \text{'Lumpy'}\}$  **then**  
|  $t_{\text{outbound}} = \{\text{Poisson}(\lambda = ADI_i)\}$   
**end**

**if** Demand pattern of  $i \in \{\text{'Erratic'}, \text{'Stable'}\}$  **then**  
|  $t_{\text{outbound}} = \{t \text{ spaced by } \text{Normal}(\mu_M^{\text{int}}, \sigma_M^{\text{int}})\}$   
**end**

$Quantity_{\text{out}} = \text{Normal}(\mu_M, \sigma_M)\}$

Infer the inventory function  $I(t)$

Save  $\mu_{I(t)}^b$

Save  $\sigma_{I(t)}^b$

Save  $\min\{I(t)\}^b$

Save  $\max\{I(t)\}^b$

**end**

Extract the statistics of  $\mu_{I(t)}^B, \sigma_{I(t)}^B, \min\{I(t)\}^B, \max\{I(t)\}^B$ .

---

By considering the minimum, maximum and average value across all the bootstrapped inventory functions, it is possible to make decisions on the target inventory level.

## 19.3 Choice of the storage technology (P2)

Different storage technologies exist, offering a wide range of solution to stock SKUs. A non-exhaustive list of storage technology involves (see Figure 19.3):

---

<sup>2</sup>The source code to estimate the inventory of a storage system is available [here](#).

- block stacking. Unit loads are placed on the floor, and eventually stacked [19, 20, 21].
- Pallet Racks. Unit loads are placed on different levels by using fixed racks [22, 23, 24, 25, 26, 27].
- Drive-in Pallet Racks. Unit loads are placed side-by-side on racks. Forklifts can travel through the racks. Last-In-First-Out (LIFO) policy is used since unit loads in the back are blocked by the unit loads on the front [28].
- Push-Back Pallet Racks. Unit loads are placed by pushing back the unit loads already on the racks. LIFO policy is used.
- Movable Pallet Racks. Racks can be moved on trails to improve the space utilisation of the warehouse.
- Cantilever Racks. These racks are used to store oversize SKUs as metal or wooden bars.
- Manual Shelves. These racks host cartons or parts and are served by manual operators.
- Automated vertical warehouse.
- Carousel. It is an automated storage system where racks or shelves rotates along a track [29, 30, 31, 32].
- Miniload. These systems are fully automated and host cartons or single parts placed inside bins. Bins are automatically stored and retrieved by the cranes of the miniload.
- Automated Storage & Retrieval System (AS/RS). These systems work similarly to miniloads, but managing full-pallet [33, 34, 35, 36, 37, 38].
- Highly automated robotic warehouse. These systems use shuttles and robots able to move and stack bins containing carton or single parts. All the bins compose a cube with a high storage density, whose storage operations are fully managed by the robots [39].

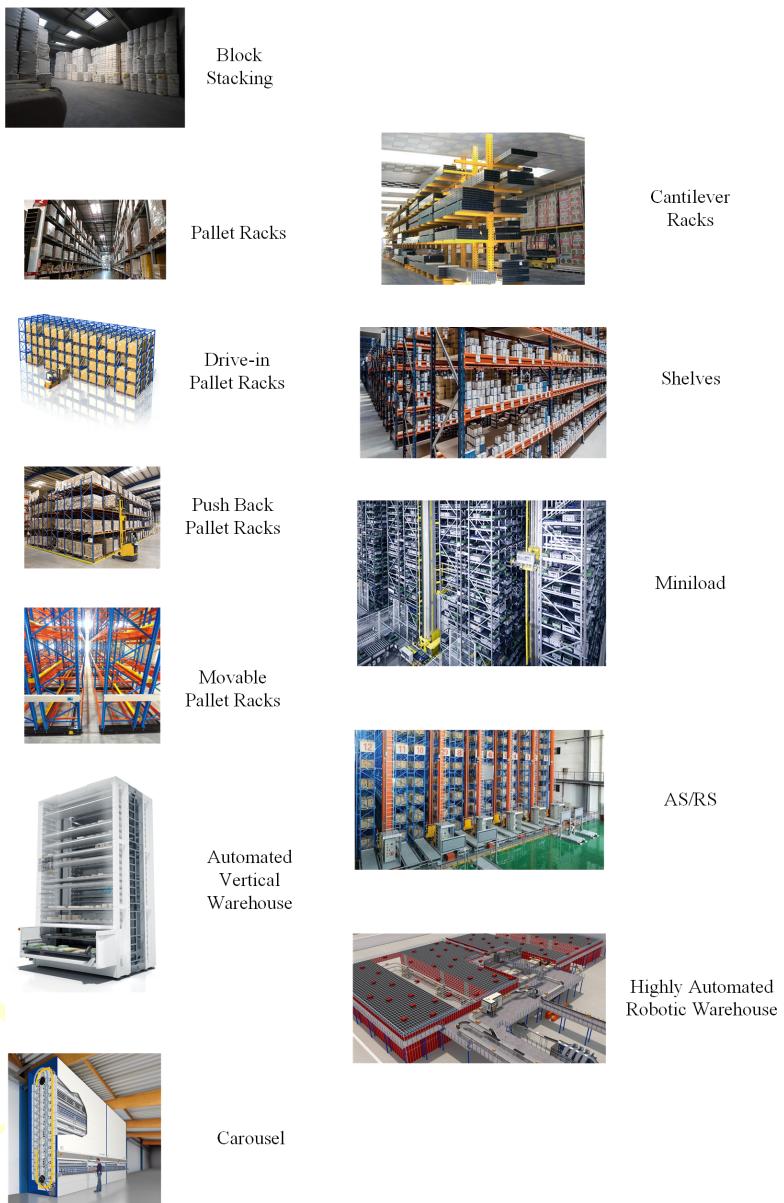


Figure 19.3: Classification of storage technologies.

The different storage technologies are integrated with different storage vehicles to put away and pick the SKUs. A non-exhaustive list of vehicles comprises (see Figure 19.4):

- forklift. The traditional handling vehicle used to move any type of SKUs in most of the industries.
- Side loader. Handling vehicle able to enter storage racks with narrow aisles.
- Pallet Jack. A manual handler to move one pallet at the time.
- Walkie Stacker. A forklift used for picking where the operator can stand and move together with the vehicle.
- Order picker. A vehicle able to bring a man on the higher levels of the storage system to permit manual picking [40].
- Crane. The automated crane of an AS/RS or a miniload.
- Shuttle. The automated robots of a highly automated storage system
- Operator. Manual operators can walk through racks or shelves performing manual picking operations.
- AGV forklift. Automated guided forklift able to autonomously put away and pick pallets.
- AGV shuttle. Automated guided shuttle able to autonomously move shelves.

Draft

Manual Vehicles	Motorised Vehicles	Automated vehicles
 Pallet Jack	 Side loader	 Crane
 Operator	 Forklift	 Shuttle
	 Order picker	
	 Walkie Stacker	 AGV forklift
		 AGV shuttle

Figure 19.4: Classification of handling vehicles.

Storage technologies and vehicles can be classified together depending on their degree of flexibility and automation. Figure 19.5 illustrates this classification.

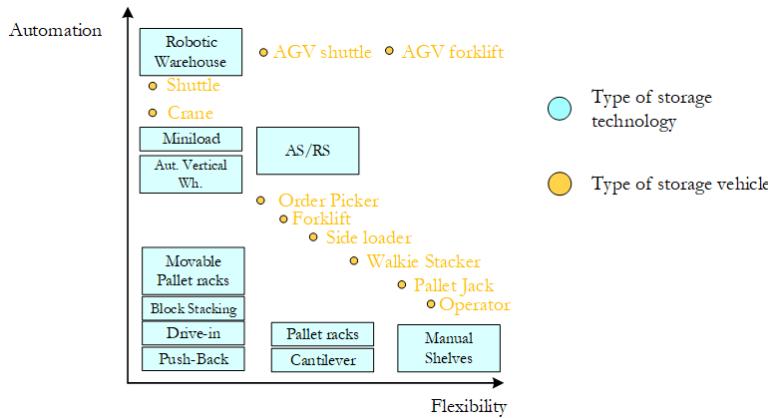


Figure 19.5: Classification of storage vehicles and storage technologies depending on the degree of automation and flexibility.

### 19.3.1 Model-driven methods (D4)

The choice of storage technology is the most challenging decision in the design of a storage system. There is no specific model to assist the decision-maker. In addition, there are a number of elements to take into account:

- the expected throughput;
- the available space;
- the space saturation of each technology;
- the risk due to fragile or inflammable SKUs;
- the dimension (i.e. size, and weight) of the SKUs and their stackability;
- the economic investment.

Generally, discrete event simulation is the preferred way to assess the behaviour of each storage technology, and identify the most suitable one. It is important to remark that a storage node may host multiple storage technologies to deal with different SKUs and to reach different levels of performance.

## 19.4 Storage allocation (P5)

Storage allocation procedures define the space of the storage system to allocate to each SKU [41, 42, 43, 44, 45, 46, 47, 48]. In particular, storage systems can be organised using:

- a *fast pick* area (FPA) (also known as “forward area”) placed at the ground level of the storage system, where pickers can immediately pick SKUs without wasting time to reach the higher levels of the storage system;
- a *reserve* area (RA) storing additional SKUs for all the SKUs in the fast pick area, and all the other SKUs (not in the fast pick area).

Storage allocation aims at defining when an FPA is needed, which SKUs should be placed in the FPA, and in which amount [47]. In storage system with multiple technologies, a storage block with a faster technology (e.g. a miniload or a cross-docking area) can be the FPA of a slower reserve storage area (e.g. a drive-in) [49, 50].

The methods for the definition of the inventory level already defined a recommendable amount  $I_i$  of an SKU  $i$  to keep inhouse. This is the starting parameter for the storage allocation of a greenfield storage system. An inventory snapshot is considered, when dealing with the re-allocation of an existing warehouse, instead.

We already introduced the difference between full pallet, carton, and part picking. The package type of the unit load guides the choice of the storage system. A storage system of full pallet does not need allocation since the benefit provided by an FPA is relevant when the number of picks  $p_i$  in the FPA is higher than the number of restocks  $r_i$  from the RA to the FPA. This condition can be verified when the picking process involves single parts or carton, but never when picking operations are full pallet. In addition, it is important to remember that the RA always hosts full pallet. These pallets are moved down to the FPA and opened to allow carton or parts picking. For each SKU  $i$  in the FPA, it is recommendable to have at least the space of two pallets. Otherwise, having a single pallet space in the FPA, it would not be possible to move a pallet from the RA before the pallet in the FPA is completely empty. It is straightforward that SKUs with a  $I_i$  lower than two pallets cannot enter the FPA.

### 19.4.1 Model-driven methods (PS4)

Allocation problem is solved using a model. Re-allocation of a storage system profoundly change the use of the space within the storage system, and it is hard to have observations of any possible configuration to use a data-driven model. We define, first the theoretical amount of space  $v_i$  to

allocate in fast-pick for each SKU  $i$ . Secondly, depending on the availability of space, we consider which SKU can enter the FPA.

Generally, not all the SKUs of a warehouse are suitable for the FPA. To define which SKUs are suitable to be stored in the FPA we can consider a ranking index, for example, the top 20% moving based on the  $Pop_i$  Pareto curve (see section 18.1.2). We can think this problem as an instance of the knapsack problem where the FPA has a fixed capacity, and it is filled based on the ranking resulting from the Pareto curve. It is important to remember that this procedure is suboptimal and hard to generalise. The filling heuristics of the knapsack by sorting for a ranking index does not imply the optimality of the solution. In addition, the actual volume to allocate in the FPA is still to be defined. Finally, there are many aspects to be considered. If the inventory levels are low for all the SKUs (e.g. one pallet), it does not make sense to have a fast pick area. Having a fast pick area with a subset of the items must consider the probability to complete an order within the FPA. If this probability is low, implying to retrieve SKUs from the higher levels in order to complete an order, the FPA does not lead to efficiency improvement. Ranking the SKUs on the  $OC_i$  could help; nevertheless, there is no optimal policy to define this allocation. Always remember of the rotation frequency of the inventory function  $I(t)$  (see section 18.1.2). The definition of the SKUs worth to be included in the FPA changes with the change of the  $I(t)$ .

The amount of space to allocate to SKU  $i$  in the FPA is calculated aiming at minimising the number of restocks  $r_i$  between the RA and the FPA. The cost of restocks is the augmented operational cost of setting an FPA; for this reason, it has to be minimised. Let  $f_i$  be the annual outbound volume of SKU  $i$ , and  $S$  the total space of the FPA (e.g. given by the sum of the volume of all the locations at the ground floor of the warehouse).

It is demonstrated [51] that the value of  $v_i$  minimising  $r_i$  is:

$$v_i^* = S \left( \frac{\sqrt{f_i}}{\sum_j \sqrt{f_j}} \right) \quad (19.4)$$

This optimal (OPT) strategy produces decimal values of  $v_i$ , that need to be rounded. In addition, this policy can be hard to be applied in practice, since each SKU can have a different space in the FPA. For this reason, two additional strategies are introduced. The equal space (EQS) strategy allocates the same amount of space for all the SKUs,  $v_i = \frac{S}{\text{card(SKU)}}$ . An equal time (EQT) strategy, defines the volume for each SKU  $i$ , such that all the SKUs have the same number of restocks  $r_i$  (i.e. all restocks have the same frequency),  $v_i = S \left( \frac{f_i}{\sum_j f_i} \right)$ . Table 19.1 identifies the allocated space  $v_i$ , the number of restocks  $r_i$ , and the total number of restocks  $R = \sum_i r_i$ .

The OPT strategy should be preferred when possible. Nevertheless, it

	OPT	EQS	EQT
$v_i$	$s\left(\frac{\sqrt{f_i}}{\sum_j \sqrt{f_j}}\right)$	$\frac{s}{card(SKU)}$	$s\left(\frac{f_i}{\sum_j f_i}\right)$
$r_i = f_i/v_i$	$\sqrt{f_i} \sum_j \sqrt{f_j}$	$card(SKU)f_i$	$\sum_j f_j$
$R = \sum_i r_i$	$\left(\sum_j \sqrt{f_j}\right)^2$	$card(SKU) \sum_i f_i$	$card(SKU) \sum_j f_j$

Table 19.1: Storage allocation policies.

could be infeasible, from an organisational perspective, to define slots of the FPA with different dimensions. In this case, the EQS strategy should be preferred. In case the restock activity results intense for some SKUs, the EQT strategy allows to restock all the SKUs together (on average) enhancing the planning of the restock activity.

## 19.5 Storage assignment/slutting (P2)

Storage assignment (also known in the literature as *slotting*) aims at defining a storage location for each SKU of the storage system. This activity is fundamental since it heavily affects the travelled distance to put away and pick SKUs during the operations. In practice, three assignment strategies exist [52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73]:

- random assignment strategy; SKUs are randomly assigned to the storage locations;
- class-based assignment strategy; SKUs are clustered into classes, and storage locations are clustered into the same classes. SKUs are randomly assigned within the set of storage locations of their class;
- dedicated assignment strategy; each SKU is assigned to specific storage locations.

While random assignment strategy does not need any additional model; both model- and data-driven models help in the definition of class-based and dedicated assignment.

### 19.5.1 Model-driven methods (PS3)

Ranking methods are particularly efficient to deploy a dedicated assignment strategy. In particular, they rank the SKUs by considering an SKU profiling

index (e.g. the  $Pop_i$ ), and they rank the storage location by summing the distance to travel from the input point to the storage location, to the output point. The two rankings provide the information to fill the storage system in a bin-packing fashion. Starting by placing the SKUs with a high value of the profiling index (e.g. the  $Pop_i$ ) to the locations with a smaller distance value. When a storage location has no space to host SKUs, the following storage location is used.<sup>3</sup>

A similar method can be used to define a class-based assignment. A common class-based strategy is called “ABC”, and it is based on the definition of three classes depending on an SKU profiling index (e.g. the  $Pop_i$ ). SKUs with high  $Pop_i$  belong to class A, with intermediate  $Pop_i$  to class B, with low  $Pop_i$  to class C. In practice the ranking of storage locations is used to define a Pareto curve on the value of their distance to the I/O, and to generate the three classes, for example:

- the first 20% of the closest storage locations to the I/O are labelled as class A locations;
- the following 30% of the closest storage locations to the I/O are labelled as class B locations;
- the remaining 50% are labelled as class C locations.

At this stage, storage locations are filled with SKUs, based on the ranking on an SKU profiling index, and SKUs inherits their class from the location where they are placed.

Both these data-driven methods, highly depend on the frequency of rotation of the inventory function  $I(t)$ , since after a period of time an SKU can be stored in a wrong position since its SKU profile index changed. For this reason, it is necessary to reconsider the storage assignment periodically. Decision support systems have been developed to support the warehouse managers in this choice [74].

### 19.5.2 Data-driven methods (PS2)

Data-driven methods provide tools to generate classes of SKUs by clustering them based on recurrent patterns of their features [75, 76, 77, 78, 79, 80, 81, 82].

Correlated storage assignment uses an incidence matrix  $M_{i,o}$  between SKUs and orders (i.e. ‘1’ if an SKU  $i$  has been picked during order  $o$ ; ‘0’ otherwise) to define a proximity matrix  $D_{i,j}$  where  $i$  and  $j$  are SKUs. A similarity index (e.g. the Jaccard index) is used to convert the incidence matrix into a proximity matrix. At this stage, hierarchical clustering is used

---

<sup>3</sup>The source code to assign SKUs to storage locations is available [here](#).

to define clusters (i.e. classes) of SKUs. The SKUs within the same cluster have been historically picked together. For this reason, they are placed close to each other.

Another data-driven method can be used to investigate the degree of health of the assignment strategy of a storage system. It always considers two metrics for each SKU: a profiling index (e.g. the  $Pop_i$ ), and the average distance  $D_i$  to reach the SKU (or the average of the distances when an SKU has multiple storage locations). When the assignment is optimal, the two metrics are linked with an exponential law [83]:  $Pop_i = \alpha e^{-\beta D_i}$ . During the operations of the warehouse, with the rotation of the frequency of the inventory function, the link between the two functions becomes similar to a Gaussian function (see Figure 18.5).

By considering the gap between these two functions, the cost saving of exchanging locations between locations can be measured.<sup>4</sup>

## 19.6 Layout design (P6)

Layout design deeply affects the total travelled distance of a storage system [84, 65, 85, 86, 87]. A storage system is generally composed of different storage blocks, with different storage technologies (identified in Section 19.3). For this reason, the storage allocation and assignment (Section 19.4 and section 19.5) can be repeated for the SKUs assigned to each storage block.

### 19.6.1 Model-driven methods (PS3)

At this stage, the layout design procedure produces a number of blocks with different technologies, performing different activities (e.g. picking, packing, quality control), which need to be placed within the same building. Plant layout design procedures (see section 22.6) applies to identify a feasible placement of the areas minimising the distances between the blocks.

It is important to remark that, so far, all the methods proposed aims at the minimisation of the travelled distance to pick each SKU within each warehouse block. When designing the entire storage system, it is important to identify the material and information flows between these blocks, and to place them such that they are minimised. Figure 19.6 identifies the material flows between different areas of the same storage system. Areas with an intense material flow should be placed closed to each other.

---

<sup>4</sup>The source code to estimate the saving obtained by exchanging the position of two storage locations is available [here](#).

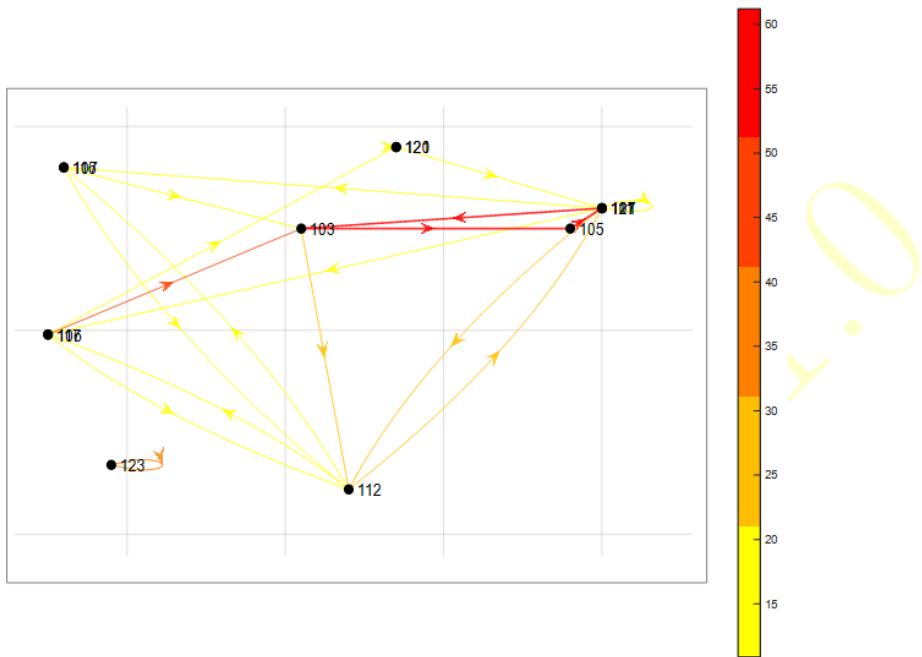


Figure 19.6: Spaghetti chart of a warehouse node.

### 19.6.2 Data-driven methods (D3)

The spaghetti chart introduced above is static, and it does not consider the dynamics of the operations of the storage system. To simulate the effect of time on the exchange of materials, a from-to matrix  $T$  with entries  $t_{j,k}$  is defined accordingly to the intensity of flow exchange between control points (CP), identifying different areas.  $t_{j,k}$  estimates the number of trips between CPs  $j$  and  $k$ . The value of  $t_{j,k}$  is static and does not consider how the system evolves. Nevertheless, it gives no information on the work in progress (WIP) between CPs (i.e. the number of SKUs waiting to be processed/shipped), that is an important parameter for the design of the buffering areas. A Markow Chain (MC) is introduced (see 11.3.2) to estimate this parameter. An MC applies statistical Markov properties on a directed graph to measure the probability of occurrences of an event (i.e. the node) given the probability of the transitions between the events (i.e., the arcs). In this case,  $T$  is the input matrix representing the probability of transitions between the events. A complete graph  $G(V, E)$  is defined as follows.

- a set of nodes, corresponding to the number of rows of the matrix (i.e. each CP is a node of  $G$ );
- a set of directed arcs representing the probability of transition between the nodes (i.e., the CPs). The weight of each arc represents the probability that the edge is travelled to transfer materials. This value is calculated as:  $\hat{t}_{j,k} = \frac{t_{j,k}}{\sum_j^n \sum_k^n t_{j,k}}$

The value of  $\hat{t}_{j,k}$  defines the probability of a transition from CP  $j$  to CP  $k$ . MC implements an initial state of the system (i.e., a number of SKUs located in each CP) and it simulates a given number of transitions. The transitions redistribute the initial state value among the others CPs according to the transition probability  $\hat{t}_{j,k}$ . The initial state of the MC is chosen accordingly to the specifics of the real storage system (e.g., the inbound node is fully loaded, and all the others are empty) and a number of transitions on the graph are performed to check how the WIP is redistributed after a number of transitions of the system. Figure 19.7 illustrates the outcome of a MC after a number of iterations, i.e. the probability to find WIP at each CP.

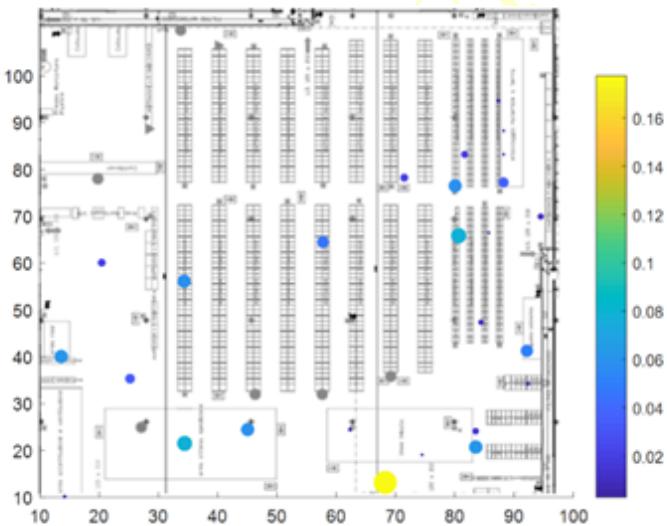


Figure 19.7: Markov chain outcome to estimate the work in process (WIP) at each control point (CP).

Even if this approach does not provide statistics on the number of SKUs in each buffering areas, it defines how the workload flows between control points. Some CPs have a higher probability of hosting WIP after a significant number of transitions. This fact suggests that these CPs deserve much

more area than the others. DES can be used to accurately design these CPs (see section 19.9).

## 19.7 Picking policy design (P7)

Depending on the type of technology involved in a storage system, it can be necessary to design a picking policy to connect the different storage blocks [88, 89, 90, 62, 91, 92, 93, 94, 95]. Fully automated storage technologies (e.g. AS/RS, miniload and shuttle systems) automatically manage both the assignment and the picking strategy actuated by the automation. Manual storage systems (shelves, racks, stacking areas) need policies for pickers to complete the orders. In addition, it is rare to find fully automated storage systems. For this reason, at some stage of the process, the parts picked by automated systems need to be consolidated with the others.

Picking policies are implemented only for the outbound activities, since they are generally responsible of the majority of the workload of a storage system:

- single-order: the picker receives a single picking list with a single order. He/she travel across all the storage system to pick the parts of the order;
- multi-order with batching: the picker receives a single picking list with multiple order. She/he is equipped with tools (e.g. a cart with slots, or a put-to-light cart) to pick parts belonging to different orders. Parts remain separated during the whole picking process, and the orders are complete at the end of the route or the picker [96, 97, 98];
- multi-order with zoning: Pickers are assigned to a specific zone of the storage system (i.e. a subset of storage locations). A picker receives a single picking list with multiple orders. He/she is equipped with tools (e.g. a cart with slots, or a put-to-light cart) to pick parts belonging to different orders. Parts remain separated during the whole picking process, and the orders are completed in the outbound area by aggregating the subset of parts picked from each zone;
- multi-order with zoning and sorting: Pickers are assigned to a specific zone of the storage system (i.e. a subset of storage locations). A picker receives a single picking list with multiple orders. Parts are collected together and separated after the picking process in a sorting zone close to the outbound area.

### 19.7.1 Model-driven methods (D4)

Discrete event simulation (DES) is the method used to compare the efficiency of different picking policies applied to the same storage system [99, 100]. DES allows to estimate the efficiency of each policy and to identify bottlenecks of the process.

## 19.8 Route design (P3)

Storage systems can be very congested areas with hundred of operators travelling to put away and pick parts. For this reason, as for the roads where we are used to riding our bicycles, precise policies exist [99, 101, 102, 103]:

- return policy: allows to travel the aisles and corridors (i.e. the edges of the set  $E$ ) of the storage system in both their directions;
- traversal policy: allow to travel the aisles and corridors (i.e. the edges of the set  $E$ ) of the storage system in one direction (one way).

## 19.9 Inbound and outbound area design (P5)

This phase aims at designing handling (i.e., buffering or processing) areas. In the handling areas, the operators prepare the SKUs for storage (e.g., inbound, inspection) or shipping (e.g., packing, order consolidation) [23, 104, 105]. Buffering areas are zones between activities where the SKUs wait for handling. The design of the buffering area is crucial since too small areas would generate congestions of the activities. Otherwise too large areas remove space from the storage/handling areas

### 19.9.1 Model-driven methods (D4)

To get statistics on the expected workload in this area, discrete event simulation (DES) is used. Typically, inbound and outbound areas deserve design validation via simulation due to their importance for all the receiving/shipping processes. Figures 19.8 and 19.9 illustrates the flowchart of a DES model and the data needed to support the design of these areas.

It is necessary to have data about the probability distribution of the execution time of the tasks, to feed the DES properly. Considering the inbound process, it is necessary to know the distribution of the arrivals of trucks and the distribution of the number of pallets transported by each truck. The arrivals distribution is usually available from the warehouse management system (WMS); otherwise, a random distribution over the daily shift can define their arrival time. After truck unloading, pallets wait in the inbound area (the one to be designed) for the following processes. It is necessary to

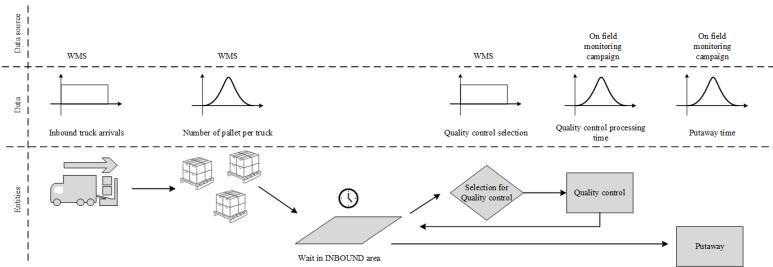


Figure 19.8: Discrete event simulation (DES) of the inbound processes of a storage system.

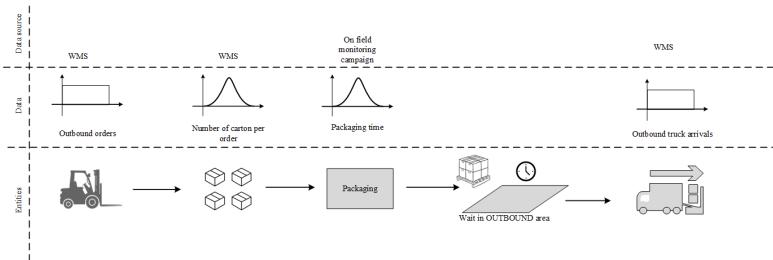


Figure 19.9: Discrete event simulation (DES) of the outbound processes of a storage system.

know the distribution of the execution time for each of the following processes and to know the number of resources dedicated to these activities. On-field monitoring campaign aims at collecting these data.

Outbound modelling is similar to the inbound modelling process. To design outbound areas, it is necessary to know the distribution of arrivals of the shipping trucks, and the distribution of picking and packing times. These values can be obtained, as well, via the on-field monitoring campaign or by analysing the records of the WMS.

The DES produces charts of the WIP at each control point. Figure 19.10 illustrates an example of the number of pallet waiting at the inbound and outbound area of a storage system.

At this stage, each storage and handling area is adequately designed and accounts for a precise area ( $m^2$ ) on the plant layout. Areas should be, then, placed on the available space such that CPs which exchange intense materials flows are close to each other.

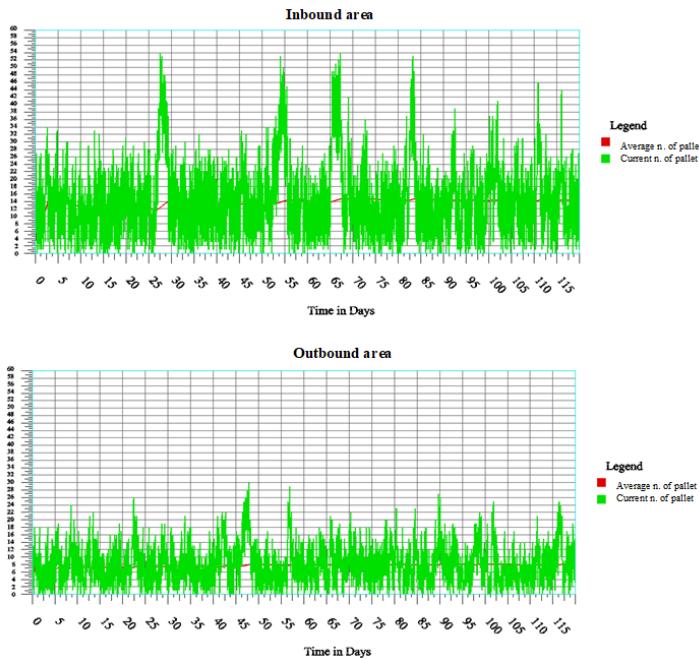


Figure 19.10: Work in process (WIP) time series of the inbound and outbound areas of a storage system.

## Bibliography

- [1] J. Ashayeri and L. F. Gelders, "Warehouse design optimization," *European Journal of Operational Research*, vol. 21, no. 3, pp. 285–294, 1985.
- [2] T. van Gils, K. Ramaekers, A. Caris, and M. Cools, "The use of time series forecasting in zone order picking systems to predict order pickers' workload," *International Journal of Production Research*, vol. 55, no. 21, pp. 6380–6393, 2017.
- [3] R. Accorsi, R. Manzini, and M. Bortolini, "A hierarchical procedure for storage allocation and assignment within an order-picking system. A case study," *International Journal of Logistics Research and Applications*, vol. 15, no. 6, pp. 351–364, 2012.
- [4] P. Baker and M. Canessa, "Warehouse design: A structured approach," *European Journal of Operational Research*, vol. 193, no. 2, pp. 425–436, 2009.

- [5] M. Dotoli, N. Epicoco, M. Falagario, N. Costantino, and B. Turchiano, “An integrated approach for warehouse analysis and optimization: A case study,” *Computers in Industry*, vol. 70, no. 1, pp. 56–69, 2015.
- [6] Y. H. Park and D. B. Webster, “Modelling of three-dimensional warehouse systems,” *International Journal of Production Research*, vol. 27, no. 6, pp. 985–1003, 1989.
- [7] A. Tufano, R. Accorsi, R. Manzini, and L. Volpe, “Data-driven models to deal with data scarcity in warehousing system design,” in *XXIV Summer School “Francesco Turco”*, 2019.
- [8] C. S. Yoon and G. P. Sharp, “A structured procedure for analysis and design of order pick systems,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 28, no. 5, pp. 379–389, 1996.
- [9] A. G. Cormier, E. A. Gunn, G. Cormier, and E. A. Gunn, “Modelling and Analysis for Capacity Expansion Planning in Warehousing,” vol. 50, no. 1, pp. 52–59, 1999.
- [10] G. Cormier and E. A. Gunn, “A review of warehouse models,” *European Journal of Operational Research*, vol. 58, no. 1, pp. 3–13, 1992.
- [11] M. Goh, O. Jihong, and T. Chung-Piaw, “Warehouse sizing to minimize inventory and storage costs,” *Naval Research Logistics*, vol. 48, no. 4, pp. 299–312, 2001.
- [12] M. S. Hung and J. C. Fisk, “Economic sizing of warehouses. A linear programming approach,” *Computers and Operations Research*, vol. 11, no. 1, pp. 13–18, 1984.
- [13] J. Levy, “The optimal size of a storage facility,” *Naval Research Logistics*, pp. 319–326, 1974.
- [14] T. J. Lowe, R. L. Francis, and E. W. Reinhardt, “A greedy network flow algorithm for a warehouse leasing problem,” *AIEE Transactions*, vol. 11, no. 3, pp. 170–182, 1979.
- [15] a.K. Rao and M. Rao, “Solution procedures for sizing of warehouses,” *European Journal of Operational Research*, vol. 108, no. 1, pp. 16–25, 1998.
- [16] M. J. Rosenblatt and Y. Roll, “Warehouse design with storage policy considerations,” *International Journal of Production Research*, vol. 22, no. 5, pp. 809–821, 1984.
- [17] M. J. Rosenblatt and Y. Roll, “Warehouse capacity in a stochastic environment,” *International Journal of Production Research*, vol. 26, no. 12, pp. 1847–1851, 1988.

- [18] J. White and R. L. Francis, "Normative models for some warehouse sizing problems," *AIEE Transactions*, vol. 3, no. 3, pp. 185–190, 1971.
- [19] R. Accorsi, G. Baruffaldi, and R. Manzini, "Design and manage deep lane storage system layout. An iterative decision-support model," *International Journal of Advanced Manufacturing Technology*, vol. 92, no. 1-4, pp. 57–67, 2017.
- [20] M. Goetschalckx and H. D. Ratliff, "Optimal lane depths for single and multiple products in block stacking storage systems," *IIE Transactions (Institute of Industrial Engineers)*, vol. 23, no. 3, pp. 245–258, 1991.
- [21] T. Nishi and M. Konishi, "An optimisation model and its effective beam search heuristics for floor-storage warehousing systems," *International Journal of Production Research*, vol. 48, no. 7, pp. 1947–1966, 2010.
- [22] F. Azadivar, "Optimum allocation of resources between the random access and rack storage spaces in an automated warehousing system," *International Journal of Production Research*, vol. 27, no. 1, pp. 119–131, 1989.
- [23] Y. Bassan, Y. Roll, and M. J. Rosenblatt, "Internal layout design of a warehouse," *AIEE Transactions*, vol. 12, no. 4, pp. 317–322, 1980.
- [24] M. Bortolini, M. Faccio, M. Gamberi, and R. Manzini, "Diagonal cross-aisles in unit load warehouses to increase handling performance," *International Journal of Production Economics*, vol. 170, pp. 838–849, 2015.
- [25] L. F. Cardona, D. F. Soto, L. Rivera, and H. J. Martínez, "Detailed design of fishbone warehouse layouts with vertical travel," *International Journal of Production Economics*, vol. 170, pp. 825–837, 2013.
- [26] S. D. Roberts and R. Reed, "Optimal warehouse bay configurations," *AIEE Transactions*, vol. 4, no. 3, pp. 178–185, 1972.
- [27] L. M. Thomas and R. D. Meller, "Developing design guidelines for a case-picking warehouse," *International Journal of Production Economics*, vol. 170, pp. 741–762, 2013.
- [28] R. Manzini, R. Accorsi, G. Baruffaldi, and T. Cennerazzo, "Travel time models for deep-lane unit-load autonomous vehicle storage and retrieval system ( AVS / RS )," vol. 7543, no. March, 2016.

- [29] W. Hua, C. Zhou, W. E. I. Hua, and C. Zhou, "Clusters and filling-curve-based storage assignment in a circuit board assembly kitting area," 2008.
- [30] H. Hwang and J. W. Ha, "An optimal boundary for two class-based storage assignment policy in carousel system," *Computers and Industrial Engineering*, vol. 27, no. 1-4, pp. 87–90, 1994.
- [31] M. K. Lee and H. Hwang, "An approach in the design of a unit-load automated carousel storage system," *Engineering Optimization*, vol. 13, no. 3, pp. 197–210, 1988.
- [32] R. G. Vickson and X. Lu, "Optimal product and server locations in one-dimensional storage racks," *European Journal of Operational Research*, vol. 105, no. 1, pp. 18–28, 1998.
- [33] R. De Koster and Y. Yugang, "Optimal storage rack design for a 3-dimensional compact AS/RS," vol. 7543, no. November, 2007.
- [34] Y. Karasawa, H. Nakayama, and S. Dohi, "Trade-off analysis for optimal design of automated warehouses," *International Journal of Systems Science*, vol. 11, no. 5, pp. 567–576, 1980.
- [35] R. Manzini, M. Gamberi, and A. Regattieri, "Design and control of an AS/RS," *International Journal of Advanced Manufacturing Technology*, vol. 28, no. 7-8, pp. 766–774, 2006.
- [36] A. Regattieri, G. Santarelli, R. Manzini, and A. Pareschi, "The impact of dwell point policy in an Automated Storage/Retrieval System," *International Journal of Production Research*, vol. 51, no. 14, pp. 4336–4348, 2013.
- [37] M. J. Rosenblatt, Y. Roll, and V. Zyser, "A combined optimization and simulation approach for designing automated storage/retrieval systems," *IIE Transactions (Institute of Industrial Engineers)*, vol. 25, no. 1, pp. 40–50, 1993.
- [38] E. Tappia, G. Marchet, M. Melacini, and S. Perotti, "Incorporating the environmental dimension in the assessment of automated warehouses," *Production Planning and Control*, vol. 26, no. 10, pp. 824–838, 2015.
- [39] T. Lerher, "Travel time model for double-deep shuttle-based storage and retrieval systems," *International Journal of Production Research*, vol. 54, no. 9, pp. 2519–2540, 2016.

- [40] P. Taylor and M.-k. Lee, “A storage assignment policy in a man-on-board automated storage / retrieval system,” no. August 2014, pp. 37–41, 2007.
- [41] J. J. Bartholdi and S. T. Hackman, “Allocating space in a forward pick area of a distribution center for small parts,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 40, no. 11, pp. 1046–1053, 2008.
- [42] C. Battista, A. Fumi, L. Laura, and M. M. Schiraldi, “Multiproduct slot allocation heuristic to minimize storage space,” *International Journal of Retail & Distribution Management*, vol. 42, no. 3, pp. 172–186, 2014.
- [43] E. Bottani, M. Cecconi, G. Vignali, and R. Montanari, “Optimisation of storage allocation in order picking operations through a genetic algorithm,” *International Journal of Logistics Research and Applications*, vol. 15, no. 2, pp. 127–146, 2012.
- [44] F. Guerriero, O. Pisacane, and F. Rende, “Comparing heuristics for the product allocation problem in multi-level warehouses under compatibility constraints,” *Applied Mathematical Modelling*, vol. 39, no. 23-24, pp. 7375–7389, 2015.
- [45] E. Hassini, “Storage space allocation to maximize inter-replenishment times,” vol. 35, pp. 2162–2174, 2008.
- [46] J. P. Van Den Berg, G. P. Sharp, A. J. Gademann, and Y. Pochet, “Forward-reserve allocation in a warehouse with unit-load replenishments,” *European Journal of Operational Research*, vol. 111, no. 1, pp. 98–113, 1998.
- [47] R. Walter, N. Boysen, and A. Scholl, “The discrete forward-reserve problem - Allocating space, selecting products, and area sizing in forward order picking,” *European Journal of Operational Research*, 2013.
- [48] Y. Yu, R. B. M. De Koster, and X. Guo, “Class-Based Storage with a Finite Number of Items: Using More Classes is not Always Better,” *Production and Operations Management*, vol. 24, no. 8, pp. 1235–1247, 2015.
- [49] S. T. Hackman, M. J. Rosenblatt, and J. M. Olin, “Allocating items to an automated storage and retrieval system,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 22, no. 1, pp. 7–14, 1990.
- [50] Z. Li, M. Y. H. Low, and R. Y. G. Lim, “Optimal decision-making on product allocation for crossdocking and warehousing operations,”

- International Journal of Services Operations and Informatics*, vol. 4, no. 4, p. 352, 2009.
- [51] J. J. Bartholdi and S. T. Hackman, “Warehouse & Distribution Science,” 2017.
  - [52] R. Accorsi, G. Baruffaldi, and R. Manzini, “Picking efficiency and stock safety. A bi-objective storage assignment policy for temperature-sensitive products,” *Computers & Industrial Engineering*, 2017.
  - [53] M. Ang, Y. Fong Lim, and M. Sim, “Robust Storage Assignment in Unit-Load Warehouses,” *Management Science*, vol. 58, no. 11, pp. 2114–2130, 2012.
  - [54] D. Battini, M. Calzavara, A. Persona, and F. Sgarbossa, “Order picking system design: The storage assignment and travel distance estimation (SA&TDE) joint method,” *International Journal of Production Research*, vol. 53, no. 4, pp. 1077–1093, 2015.
  - [55] M. Bortolini, L. Botti, A. Cascini, M. Gamberi, C. Mora, and F. Pilati, “Unit-load storage assignment strategy for warehouses in seismic areas,” *Computers & Industrial Engineering*, vol. 87, pp. 481–490, 2015.
  - [56] L. F. Cardona, L. Rivera, and H. Jairo Martínez, “Analytical Optimization for the Warehouse Sizing Problem Under Class-Based Storage Policy,” *Ingeniería y Ciencia*, vol. 12, no. 24, pp. 221–248, 2016.
  - [57] L. Chen, A. Langevin, and D. Riopel, “A tabu search algorithm for the relocation problem in a warehousing system,” *International Journal of Production Economics*, vol. 129, no. 1, pp. 147–156, 2011.
  - [58] M. E. Fontana and C. A. V. Cavalcante, “Use of Promethee method to determine the best alternative for warehouse storage location assignment,” *International Journal of Advanced Manufacturing Technology*, vol. 70, no. 9-12, pp. 1615–1624, 2014.
  - [59] M. Goetschalckx and H. D. Ratliff, “Shared Storage Policies Based on the Duration Stay of Unit Loads,” *Management Science*, vol. 36, no. 9, pp. 1120–1132, 1990.
  - [60] F. Guerriero, R. Musmanno, O. Pisacane, and F. Rende, “A mathematical model for the Multi-Levels Product Allocation Problem in a warehouse with compatibility constraints,” *Applied Mathematical Modelling*, vol. 37, no. 6, pp. 4385–4398, 2013.

- [61] X. Guo, Y. Yu, and R. B. De Koster, “Impact of required storage space on storage policy performance in a unit-load warehouse,” *International Journal of Production Research*, 2016.
- [62] J. M. Jarvis and E. D. McDowell, “Optimal product layout in an order picking warehouse,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 23, no. 1, pp. 93–102, 1991.
- [63] C. Kallina and J. Lynn, “Application of the Cube-Per-Order Index Rule for Stock Location in a Distribution Warehouse,” *Interfaces*, vol. 7, no. 1, pp. 37–46, 1976.
- [64] T. N. Larson, H. March, and A. Kusiak, “A heuristic approach to warehouse layout with class-based storage,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 29, no. 4, pp. 337–348, 1997.
- [65] T. Le-Duc and R. De Koster, “Layout optimization for class-based storage strategy warehouses,” *Supply Chain Management - European Perspectives*, pp. 191–214, 1999.
- [66] A. J. Mallette and R. L. Francis, “A generalized assignment approach to optimal facility layout,” *AIEE Transactions*, vol. 4, no. 2, pp. 144–147, 1972.
- [67] C. Malmborg and B. Krishnakumar, “Optimal storage assignment policies for multiaddress warehousing systems,” *IEEE transaction on system, man and cybernetics*, vol. 19, no. 1, 1989.
- [68] R. Manzini, R. Accorsi, M. Gamberi, and S. Penazzi, “Modeling class-based storage assignment over life cycle picking patterns,” *International Journal of Production Economics*, vol. 170, pp. 790–800, 2015.
- [69] V. R. Muppani (Muppant) and G. K. Adil, “Efficient formation of storage classes for warehouse storage location assignment: A simulated annealing approach,” *Omega*, vol. 36, no. 4, pp. 609–618, 2008.
- [70] J. C. H. Pan, P. H. Shih, M. H. Wu, and J. H. Lin, “A storage assignment heuristic method based on genetic algorithm for a pick-and-pass warehousing system,” *Computers and Industrial Engineering*, vol. 81, pp. 1–13, 2015.
- [71] S. Quintanilla, Á. Pérez, F. Ballestín, and P. Lino, “Heuristic algorithms for a storage location assignment problem in a chaotic warehouse,” *Engineering Optimization*, vol. 47, no. 10, pp. 1405–1422, 2015.

- [72] N. Sooksaksun, V. Kachitvichyanukul, and D. C. Gong, "A class-based storage warehouse design using a particle swarm optimisation algorithm," *International Journal of Operational Research*, vol. 13, no. 2, p. 219, 2012.
- [73] J. Xie, Y. Mei, A. T. Ernst, X. Li, and A. Song, "Scaling Up Solutions to Storage Location Assignment Problems by Genetic Programming," in *Simulated Evolution and learning*, pp. 691–702, 2014.
- [74] R. Accorsi, R. Manzini, and F. Maranesi, "A decision-support system for the design and management of warehousing systems," *Computers in Industry*, vol. 65, no. 1, pp. 175–186, 2014.
- [75] F. Bindi, R. Manzini, A. Pareschi, and A. Regattieri, "Similarity-based storage allocation rules in an order picking system: an application to the food service industry.,," *International Journal of Logistics Research and Applications*, vol. 12, no. 4, pp. 233–247, 2009.
- [76] H. Brynzèr and M. I. Johansson, "Storage location assignment : Using the product structure to reduce order picking times," *International Journal of Production Economics*, vol. 46-47, pp. 595–603, 1996.
- [77] C.-m. Liu, "Clustering techniques for stock location and order-picking in a distribution center," *Computers & Operations Research*, vol. 26, pp. 989–1002, 1999.
- [78] D. Ming-huang Chiang, C.-p. Lin, and M.-c. Chen, "The adaptive approach for storage assignment by mining data of warehouse management system for distribution centres," *Enterprise Information systems*, pp. 37–41, 2013.
- [79] M. Moshref-Javadi and M. R. Lehto, "Material handling improvement in warehouses by parts clustering," *International Journal of Production Research*, vol. 54, no. 14, pp. 4256–4271, 2016.
- [80] K. W. Pang and H. L. Chan, "Data mining-based algorithm for storage location assignment in a randomised warehouse," *International Journal of Production Research*, vol. 55, no. 14, pp. 4035–4052, 2017.
- [81] J. S. Smith and L. Yingde, "Dynamic slotting optimization based on SKUs correlations in a zone-based wave-picking system," 2012.
- [82] J. Xiao and L. Zheng, "Correlated storage assignment to minimize zone visits for BOM picking," *International Journal of Advanced Manufacturing Technology*, vol. 61, no. 5-8, pp. 797–807, 2012.

- [83] R. Manzini, R. Accorsi, G. Baruffaldi, D. Santi, and A. Tufano, “Performance assessment in order picking systems : a visual double cross-analysis,” *The International Journal of Advanced Manufacturing Technology*, 2018.
- [84] W. A. Jaimes, “Optimization of a warehouse layout used for storage of materials used in ship construction and repair,” vol. 5, pp. 59–70, 2012.
- [85] J. Pliskin and D. Dori, “Ranking Alternative Warehouse Area Assignments: A Multiattribute Approach.,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 14, no. 1, pp. 19–26, 1982.
- [86] M. Tutam and J. A. White, “A Conventional Warehouse Design with Multiple Docks,” in *Industrial and Systems Engineering Research Conference*, pp. 333–342, 2015.
- [87] G. Q. Zhang, J. Xue, and K. K. Lai, “A class of genetic algorithms for multiple-level warehouse layout problems,” *International Journal of Production Research*, vol. 40, no. 3, pp. 731–744, 2002.
- [88] J. J. Bartholdi and D. D. Eisenstein, “BUCKET BRIGADES A Self-Balancing Order-Picking System for a Warehouse,” p. 22, 1996.
- [89] K. Choe and G. Sharp, “Small parts order picking: design and operation,” 1991.
- [90] S. Ene and N. Ozturk, “Storage location assignment and order picking optimization in the automotive industry,” *International Journal of Advanced Manufacturing Technology*, vol. 60, no. 5-8, pp. 787–797, 2012.
- [91] C. H. Lin and I. Y. Lu, “Procedure of determining the order picking strategies in distribution center,” *International Journal of Production Economics*, vol. 60, pp. 301–307, 1999.
- [92] G. Marchet, M. Melacini, and S. Perotti, “Investigating order picking system adoption: a case-study-based approach,” *International Journal of Logistics Research and Applications*, vol. 18, no. 1, pp. 82–98, 2015.
- [93] C. G. Petersen and G. Aase, “A comparison of picking, storage, and routing policies in manual order picking,” *International Journal of Production Economics*, vol. 92, no. 1, pp. 11–19, 2004.
- [94] K. J. Roodbergen, I. F. A. Vis, and G. Don Taylor Jr, “Simultaneous determination of warehouse layout and control policies,” *International Journal of Production Research*, vol. 53, no. 11, pp. 3306–3326, 2015.

- [95] T. van Gils, K. Ramaekers, K. Braekers, B. Depaire, and A. Caris, “Increasing order picking efficiency by integrating storage, batching, zone picking, and routing policy decisions,” *International Journal of Production Economics*, 2017.
- [96] M. C. Chen, C. L. Huang, K. Y. Chen, and H. P. Wu, “Aggregation of orders in distribution centers using data mining,” *Expert Systems with Applications*, vol. 28, no. 3, pp. 453–460, 2005.
- [97] C. A. Valle, J. E. Beasley, and A. S. da Cunha, “Optimally solving the joint order batching and picker routing problem,” *European Journal of Operational Research*, vol. 262, no. 3, pp. 817–834, 2017.
- [98] I. Žulj, S. Kramer, and M. Schneider, “A hybrid of adaptive large neighborhood search and tabu search for the order-batching problem,” *European Journal of Operational Research*, vol. 264, no. 2, pp. 653–664, 2018.
- [99] F. Caron, G. Marchet, A. Perego, F. Caron, P. Milano, G. Marchet, P. Milano, and A. Perego, “Layout design in manual picking systems : a simulation approach,” 2007.
- [100] W. Lu, D. McFarlane, V. Giannikas, and Q. Zhang, “An algorithm for dynamic order-picking in warehouse operations,” *European Journal of Operational Research*, 2016.
- [101] C. G. Petersen, “The impact of routing and storage policies on warehouse efficiency,” *International Journal of Operations & Production Management*, vol. 19, no. 10, pp. 1053–1064, 1999.
- [102] K. J. Roodbergen and R. De Koster, “Routing methods for warehouses with multiple cross aisles,” *International Journal of Production Research*, vol. 39, no. 9, pp. 1865–1883, 2001.
- [103] K. Roodbergen and I. Vis, “A model for warehouse layout,” *IIE Transactions (Institute of Industrial Engineers)*, vol. 38, no. 10, pp. 799–811, 2006.
- [104] S. Heragu, L. Du, R. Mantel, and P. Schuur, “Mathematical model for warehouse design and product allocation,” *International Journal of Production Research*, vol. 43, pp. 327–338, jan 2005.
- [105] R. Pandit and U. S. Palekar, “Response Time Considerations for Optimal Warehouse Layout Design,” vol. 115, no. August 1993, 1993.

Draft Version 1.0

Draft Version 1.0

## Part V

# PRODUCTION SYSTEMS



# 20

## Diagnostic Models

The last part of this book is about production nodes. Production nodes are where the *magic* happens: they add value to raw materials by transforming them to semifinished or finished products.

Here we encounter the highest degree of complexity with low standardisation of the methods, technologies and processes. Nevertheless, it is not a good reason to abandon a data-driven approach. On the contrary, it is the field where decision-makers benefit the most from this approach linking the model-driven and the data-driven approaches.

This chapter focuses on the definition of the keywords and key entities extending the ontology of chapter 3 to production nodes. After that, it introduces the diagnostic framework for production nodes with a relational data structure and some KPIs. It will be necessary to pay special attention to the definition of the data structure since it needs the flexibility to host data from very diverse production systems. The same rationale applies to the KPIs, developed from the definition in chapter 3, and applicable to any industry. A smaller subset is, then, introduced to be industry-oriented.

Chapter 21 focuses on the control of a production node by using analytics to get knowledge on the existing processes and to improve their efficiency. Chapter 22 studies the design of a production facility, and chapter 23 the design of production processes from a data- and model-driven approach.

### 20.1 Ontology

Here we develop the general ontology introduced in chapter 3, by applying it to a production node.

## Entities

We identify the following entities.

**Part ( $i$ ):** it is a piece of raw material, component, subassembly or assembly. Where:

- Raw material is a part purchased out of the production node (i.e. from the suppliers).
- Component is an individual piece assembled into more complex products.
- Subassembly is an assembled unit further assembled into more complex products.
- Assembly/final assembly/finished product is the fully assembled product, i.e. the outcome of the production node.

**Processing node ( $j$ ):** a processing node is a resource within the production nodes (e.g. machines and workbenches). In this section, they are also called resources or machines.

**Edge ( $j, k$ ):** the paths connecting resources (e.g. conveyors, corridors)

**Vehicle ( $v$ ):** is a unit travelling on edges to handle materials between resources; some examples are forklifts, AGVs and conveyors.

**Consumable ( $s$ ):** it is an additional material (often a fluid) or a service necessary to perform activities within the production node. Some examples are electricity, compressed air or steam.

**Route ( $e$ ):** it is the production cycle of a part  $i$ , i.e. the sequence (ordered set) of resources (machines or workbenches) to visit to transform it to a finished product.

**Order ( $o$ ):** it is a production order received from a customer.

**Job ( $b$ ):** it is a planned activity (a sequence of tasks) to perform on a part  $i$  using resources  $j$ .

**System network** the graph  $G(V, A)$  of nodes  $j \in V$  and edges  $(j, k) \in A$ , composing a job-shop, flow-shop or a production line.

## Metrics

We identify the following metrics to assess the performances of a processing node  $j$ .

**Throughput ( $TH_j$ ):** the throughput is the productivity of a resource, i.e. the average number of parts in output per unit of time (e.g., parts produced per hour)..

**Work in process ( $WIP_j$ ):** it is the number of parts (i.e., the level of inventory) waiting for processing nearby a resource  $j$ .

**Work in process ( $WIP_{jk}$ ):** it is the number of parts being transported by a vehicle  $v$ .

**Capacity** ( $C_j$ ): it is the upper bound of the throughput of a resource.

**Capacity** ( $C_v$ ): it is the maximum number of part transportable by a vehicle  $v$  at the same time.

**Utilisation** ( $U_j$ ): it is the average fraction of time that a resource is not idle for lack of parts.

**Utilisation** ( $U_v$ ): it is the average fraction of non-empty space on a vehicle.

**Lead time** ( $LT_e$ ): it is the time allocated for a given route (i.e. to complete the production cycle of a part).

**Cycle time** ( $CT_e$ ): it is the average time from the release of a job to the end of its route.

**Service level** ( $SL_e$ ):  $\text{Prob}\{\text{cycle time} \leq \text{lead time}\}$

### Information functions

Finally, we define the three information functions: movements  $M$  are referred to load/unload of parts on resources; inventories  $I$  are referred to the work in process on a resource  $j$ , or a vehicle  $v$ . The productivity  $P$  refers to the inbound and outbound absorption rates of a resource. Since production resources do not have a decoupling purpose (as, for example, storage system does) it is possible to refer to  $P_j = P^{IN} = P^{OUT}$ . Table 20.1 summarises the definition of the three functions in a storage system.

Information function	Description
Movements $M_i^{j,v}(t)$	This function defines the loadings and unloadings of a part $i$ on a machine or workbench.
Inventory $I_j(t)$	This function defines the work in process on a resource $j$ .
Inventory $I_v(t)$	This function defines the work in process on a vehicle $v$ .
Productivity $P_j^{IN}(t^*)$	This function defines the inbound productivity (e.g. part per hour) of a machine or workbench.
Productivity $P_j^{OUT}(t^*)$	This function defines the outbound productivity (e.g. part per hour) of a machine or workbench.

Table 20.1: Definition of the information functions of a production node.

## 20.2 Data Structure

Production environments are extremely biased. This paragraph aims at providing data structures that are robust and adaptable to many different production environments. Non-relational structures have this characteristic. For the sake of completeness, this paragraph analyses a traditional relational data structure first. The criticalities of this structure are, then, identified moving to the definition of a non-relational model.

### 20.2.1 A relational model for production environments

Modelling a system with an ER structure means identifying the physical entities with their attributes and the relationship among them. The relationships between entities lead to extraordinarily rigid and instance-oriented data structures which are hardly generalisable in production systems. For this reason, the ER model could not be the best choice to build a model usable for analytics and data-driven modelling, at least not in the case of a production node. We support this fact by empirical evidence starting from the design of an ER model to structure data from the food industry and showing that the model is too rigid to host information from other types of industries. We introduce, then a non-relational model for a production environment with all the ingredients to enable data-driven generalizable models.

Modelling is performed by using a top-down approach, starting with the definition of macro-areas that involve similar entities. Afterwards, it is necessary to identify each table (i.e. entities and relations) and the attributes for each table. Following this direction, we apply a top-down approach to a production environment of a food catering industry, identifying three macro-areas to model:

1. supply chain network;
2. production plant;
3. product quality and control.

Within each of these macro-areas, we identify a set of entities to model using tables.

#### Supply chain network

This macro area considers all the entities linked to the operations performed out of the production plant, which may affect the operations within the production plant. It is the case of customers, suppliers, carriers and other stakeholders connected with the production plant.

**Customers-based tables** These tables deal with distribution flows between production nodes final customer. A table *Nodes* collects the information of the plants involved as producers or suppliers. The attributes of this table are the address, latitude and longitude, working days. A table *Client* stores similar information for the customers: address, latitude and longitude and the client's profile (depending on the product/service customisation). A table *Tour* includes the information on the shipping tours. Each tour is performed at a different frequency with a due date for its departure. This

time profoundly affects the operations in production since everything must be completed and ready before that time. Besides, each tour has delivery time windows to respect for each client. A relationship between *Tour* and *Client* maps this information.

**Demand-based tables** The table *Orderlist* collects all the information on the customers' order that a production plant needs to process. This table maps the order code, the quantity for each item, the customer, and the service level (i.e. the due time) to deliver the finished product.

### Production plant

Within this macro-area, we model all the entities involved in the realisation of the final product within the production plant.

**Resource-based tables** These tables map the production resources with their attributes and relationships. A table *Machine category* identifies the different technological group of the machines in the production plant (e.g. lathe, mill, oven). A table *Machine park* identifies all the different variants of the resources in the production area. This table has attributes to record all the specifics of a resource (name, manufacturer, speed, performance, capacity, energy absorption, etc.).

The relationship between *Machine category* and *Machine park* defines the type of capacity of the machine too. Especially while dealing with the design of the number of resources, it is necessary to identify a proper measure of capacity. Unfortunately, many different ways exist to define the capacity of a machine. Here we identified the five different types of capacity specifying the units of measure. A machine belongs to one of these types:

1. type 1: is a capacity measured with a continuous size metric (e.g. kg or litres);
2. type 2: is applied to measure the capacity of a machine which has a discrete number of available slots to process parts. Each slot is available/occupied and can host a single part  $i$ ;
3. type 3 is applied to bottleneck machines that have a throughput metric of capacity. This capacity is measured in a metric belonging to type 1 or type 2 over a time unit (e.g. Kg/h or parts/h);
4. type 4 is used for buffer and stock areas whose capacity is expressed in cube meters of inventory;
5. type 5 is used for machines which process entities containing a set of parts (e.g., a tray). Their capacity is expressed in terms of container entities (e.g. number of trays).

A table *Tasks* identifies all the possible activities executable by the resources on the plant layout. Attributes of this table map if a task needs energy to be executed and/or the supervision of a human.

**Plant Layout-based tables** These tables model the placement of the resources on the layout of a production plant. A table *Departments* identifies all the departments of the production area. A table *Control points* identify a set of known coordinates ( $x; y; z$ ) on the plant layout assigning each of them to a department. A table *Machine cp assignment* links a resource from the table *Machine park* to a control point modelling the physical position of a resource on the plant-layout.

A table *Machine cycle assignment* defines the relationship between the resources and the production tasks specifying which resource can perform a task.

**Auxiliary systems-based table** These tables identify all the entities (e.g. packages, consumables, energy) which are necessary to perform production tasks. A table *Packages* maps all the packages (e.g. cartons, trays) with their dimensions (i.e. height, length, width) used in the production plant to handle semifinished and finished products. A table *Vehicles* identify all the resource used for the handling of products and packages in the production area. A table *Energy cost* identify the cost for each source of energy of the production plant. A table *Efficiency parameters* identify the behaviour of each machine supplied with a specific source of energy. In particular, for each machine and source of energy, a curve of efficiency is defined as a function of the working time.

### Product, quality and control

These tables revolve around the product mapping all the information on the product itself (i.e. the bill of materials), the production cycle to realise it, the packaging cycles to pack and customise it and the safety constraints (e.g. temperature and quality decay) associated with the finished product.

**Product-based tables** A table *Products* map all the products identifying their characteristics (e.g., length, width, height, weight, volume), product family and other attributes as:

- the slicing profile (mono/multi-portions);
- the temperature profile (cook-warm/cook-chill);
- the organic profile (organic/non-organic);
- the post-processing (eventually mixed after cooking);

- the shelflife (in hours or days);
- the nutritional values (energy, carbs, fats, protein).

A table *Bill of materials* defines, for each product, the graph where raw materials converge into semi-finished and finished products. The table defines the nodes and edges of the graph specifying which quantity of raw material is necessary to assemble the following semifinished. A table *Cycles* identify the production cycle to transform raw materials into the finished products. Each cycle row defines:

- the type of the task;
- the quantity of the raw materials (which must be coherent with a machine type capacity);
- the working time;
- the time type;
- the department where the task must be performed;
- the temperature-humidity profile.

The working time indicates the time required to perform a specific task, but the real duration of the task depends on the time type. Time type '0' indicates handling activities: the processing time depends on the distance travelled between control points. Time type '1' is a fixed time, not dependent by the processing quantity (e.g. a machine setup). Time type '2' represents cooking time; this time is not conditioned by the quantity but can be affected by temperature and humidity. Time type '3' models a manual operation which depends on the processed quantity. This dependence is hardly linear: less than linear (e.g. logarithmic) curves have been used to estimate the duration of the task.

A table *Packaging cycles* defines all the options to package a finished product by identifying the type of packaging and thermal treatment to perform after cooking.

**Quality- and control-based tables** These tables are needed to monitor the quality of the products from the safety point of view that is mainly represented by the temperature in the food industry. A table *Product category* identifies the different thermal category (e.g. cook-chill, cook-warm) of the products and a safe holding temperature for each category. A table *Cycles temperature* collects time-series information from sensors monitoring the temperature and humidity of the products at different stages of the production process.

The paragraphs above give a clear picture of the entities and relationship of a food catering plant. The same structure is reported in Figure 20.2 illustrating the resulting ER-model.

Draft Version 1.0

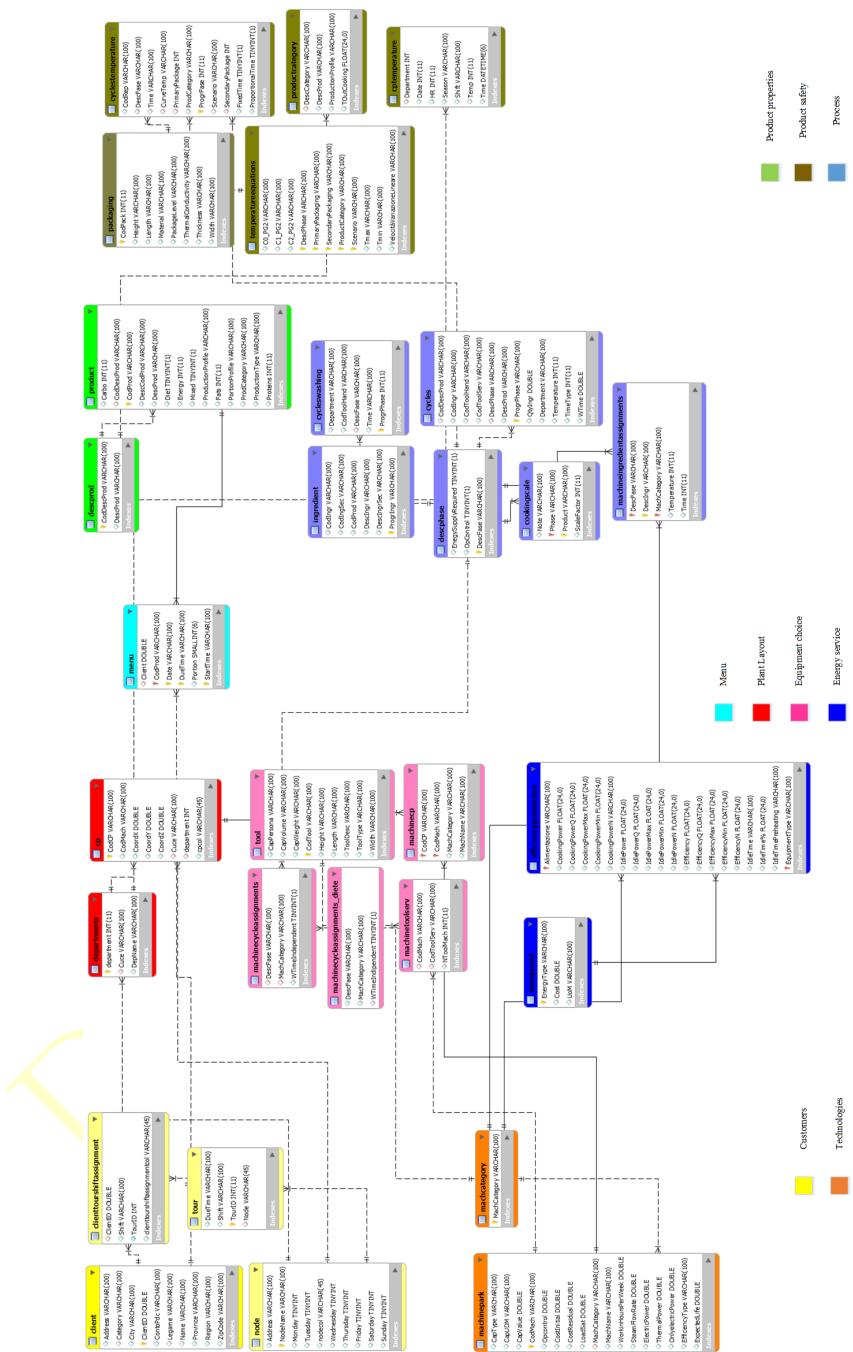


Table 20.2: ER-model for a production environment in a food catering industry.

Figure 20.2 remarks the high level of complexity covering a production plant and its modelling. Regardless of the complexity, the bias due to the market niche of the plant leads to the impossibility of a general-purpose model to model entities and processes of any production site.

This rigidity may be useful in a production environment to keep track of everything within a robust structure. Nevertheless, it is not a good starting point to analyse data, compare it with different production plants and generalise the results, which is the primary aim of research and this book. For this reason, the following paragraph proposes a different approach using a non-relational structure which removes some bias and allows to generalise some entities for any production system.

### 20.2.2 A Non-relational model for production environments

This section presents a non-relational data structure able to embed the same information content of the ER model introduced in 20.2.1 but able to host any other additional attribute. In addition, the model has a minimum number of required attributes that enables define a minimum viable model (MVM) enhancing analysis applicable to any production node. This is the natural implementation of the MIP model introduced in section 3.3.1. The model consists of four collections whose data are recorded by the manufacturing execution system (MES).

A collection *Part* identifies all the parts  $i$  processed within the production plant. The id of the item and the description are required to define the MVM. Other attributes can be stored as well, like:

- the size,
- the weight,
- the supplier,
- the product family.

A collection *Movements* defines the movement function of the MIP model storing information on when a part  $i$  enter or leave the production node (or one of its subsystems, as a resource  $j$ ). Attributes timestamp, part id, and quantity are needed for the MVM. Each document can record other attributes as:

- the type of package,
- the id of the operator,
- the id of the order,

- the id of the machine performing a task.

A collection *Temmeasurement* store information on on-field time and motion analysis which corrects and complete the measurements of the MES. The id the observation and its duration in seconds define the MVM.

A collection *Plant* contains a single document with the information of the production node. The node code is the only attribute necessary to define an MVM. Other attributes can be recorded, like:

- the latitude and longitude of the plant,
- the address of the plant,
- the list of customers served by the plant
- the overall productivity of the plant,
- the energy absorption of the plant.

Figure 20.1 uses the unified modelling language (UML) to represent the MVM of the non-relational structure of a production system.

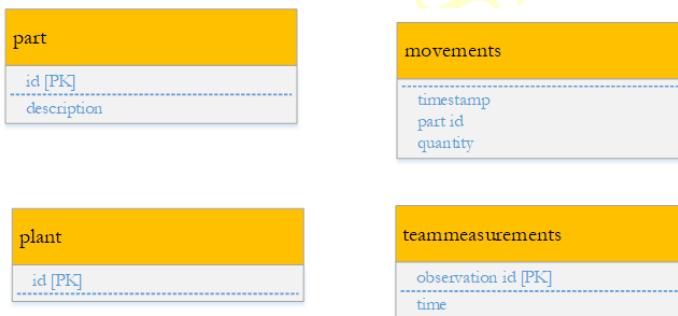


Figure 20.1: Diagram in UML notation of the minimum viable model for a non-relational structure of a production system.

## 20.3 Decision Patterns

This section introduces the decision patterns (see section 4.2) analysed in chapters 21, 22 and 23 focusing on the role of analytics to solve production node control, design and process design. Each chapter addresses a family of problems:

1. production control problems, deal with the assessment and improvement of the performance of an existing production plant.

2. plant design problems, deal with the design and placement of the physical assets of a production plant;
3. process design problems, deal with the definition of the production processes.

Different methodologies allow getting feasible solutions to these problems. Table 20.3 illustrates the entities and their definition according to the ontology in Paragraph 3.1.

Draft Version 1.0

System	Class of the problem	Decision	Task	Entity	Ontology	Descriptive	Explanative	Predictive	Prescriptive	Decision Science	Methodologies
Production node: P1 • Faulty problem	Plant Design	Clustering items into product families	Stock Keeping Unit	Part	•	○	○	○	○	D1, D2, D3	
Production node: P6 • Plant layout Problem	Plant Design	Facility location	Production system	System Network	○	○	○	○	○	PS4	
Production nodes: P4 • Mechanical plant and equipment design	Plant Design	Auxiliary systems design	Energy Thermal plant and services	Consumables	○	○	○	○	●	PS4	
Production node: P2 • Technology Assignment problem	Plant Design	Technology and asset choice	Machines, workstations and resources	Processing node	○	●	○	○	●	PS2, PS3	
Production node: P3 • Power problem	Plant Design	Definition of the number of assets	Workstations	○	○	○	○	●	●	PS4	
Production node: P5 • Placement Problem	Plant Design	Layout design	Production system, resources	Processing node	○	●	○	○	●	PS1	
Production node: P7 • Dispatching rules	Process Design	Inventory policy design	Products	Part	○	○	○	○	●	PS4	
Production node: P8 • Flow problem	Process Design	Handling design	Conveyors/forklift	Vehicle	○	○	○	○	●	PS3, PS4	
Production node: P9 • Mechanical plant and equipment design	Process Design	Workstation design	Workstations	Processing node	○	○	○	○	●	PS3, PS4	
P8 • Performance assessment	Production Control	Performance assessment	Storage system	System Network, Parts, Resources, Vehicles, Jobs, Routes	●	●	○	○	○	D1, D2, D3, D4	
Production node	Production Control	Workload forecast	Stock Keeping Unit	Part	○	●	○	○	○	PD1, PD2, PD3	
Production node: P9 • Workload prediction	Production Control	Job scheduling	Machine resource	Processing node, Part	○	●	●	●	●	PS1, PS2, PS3	
Production node: P10 • Operations management	Control										

Table 20.3: Decision problems classification in a production node.

Production control involves the following decisions:

1. the assessment of the performance of production processes;
2. the prediction of the market demand and the workload in the future;
3. the prescription of job scheduling to assign jobs to resources.

Plant design involves the following decisions:

1. the definition of homogeneous product families to process and analyse together;
2. the definition of the location of the physical plant;
3. the design of the auxiliary system to feed with energy and fluid asset and resources;
4. the definition of the technology of asset and resources to provide adequate throughput;
5. the definition of a proper amount of resources identifying a proper production capacity;
6. the design of the plant layout.

Process design involves the following decisions:

1. The definition of an inventory policy and inventory level for parts;
2. The design of the handling systems to manage the flow of materials within the production plant;
3. The design of workstation and workbench to smooth production flows, keep order and provide an ergonomic working place to the operators.

# 21

## Production Node Control

This chapter focuses on the problems involving the control of a production node. The control activity, from a logistic point of view, is necessary to ensure the production processes are smooth and efficient.

### 21.1 Performance assessment (P8)

*“If you can’t measure it, you can’t manage it”.* This famous quote of the philosopher and economist Peter Drucker clearly defines the importance of performance assessment. The aim is the modelling of processes and the measurement of their efficiency from different points of view. Here qualitative and quantitative methods are introduced to measure and understand the processes.

#### 21.1.1 Model-driven methods (D4)

Business process modelling is a qualitative mapping activity aiming at the definition of the relationships between physical and information entities involved in any process. The modelling of production processes allows defining their interactions with operators, resources, parts and information systems. The Business Process Model and Notation (see section 5.1) is a graphical model for this purpose using predefined symbols to model tasks, activities and interactions. Mapping a production environment:

- activities are any production or auxiliary task necessary for the realisation of the product (manufacturing, traceability, supplier and customers interactions);

- events identify the step of the processes concluding an activity, and the part increases its value (e.g. the end of the production or the inbound process);
- gateways describe different production path (e.g. variants of the products);
- pools identify the department or the office in charge of specific tasks.

BPMN defines a qualitative map of the production processes useful for manager and practitioners to identify the way their processes are realised. The hard assessment takes place with the definition of quantitative KPIs. The KPIs used in these chapters refers to the metrics defined in 3.1. KPIs are organised according to four classes, relevant to the logistics and operation design [1]:

1. Logistic KPIs, evaluate the logistic impact of a certain solution. They use metrics like time, distance and the performance parameters introduced in section 3.1.
2. Cost KPIs, evaluate the economic sustainability of a given solution. They are expressed in € or other currency.
3. Energy KPIs, evaluate the energy needed to feed a given solution. They use metrics as kW and kWh.
4. Environmental KPIs, evaluate the environmental impact of a given solution. They are expressing the equivalent  $CO_2$  produced per year.

Table 21.1 identifies which KPI is relevant to each problem. In general, each problem can be assessed from multiple perspectives.

### 21.1.2 Data-driven methods (D1, D2, D3, D4)

The definition of the KPIs is a modelling activity where the definition of a BPMN may help to identify which are the KPIs essential to monitor. With the advent of Big Data, the amount of collected record can be ample, and the pure definition of the KPI neglects a significant amount of information.

Data-driven methods use descriptive analytics to manipulate these data and extract information. Clustering methods (see section 8.2) allows defining clusters. In particular, having a time series of a KPIs it is possible to identify if a value diverges to out-of-control of anomalous values. These values are often associated with bottlenecks and inefficiencies of the production process. Inferential statistics (see section 6.2) is used to describe the behaviour of a KPI as a random variable. Almost always the value of a KPI changes, and it is relevant to identify its statistics to evaluate the

Class of the problem	Logistic KPIs	Cost KPIs	Energy KPIs	Environmental KPIs
P1 - Family problem	WIP <sub>j</sub> , LTe	◦	◦	◦
P2 - Technology Assignment problem	LTe, THj Travelling distance (Km/year), U <sub>j</sub>	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
P3 - Flow problem	◦	◦	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
P4 - Mechanical plant and equipment design	◦	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
P5 - Power problem	U <sub>j</sub> , C <sub>v</sub> , LOS <sub>e</sub>	Initial investment (€)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
P6 - Placement Problem	LOSe, Travelling distance per year (Km/y)	◦	◦	◦
P7 - Dispatching rules	WIP <sub>j</sub> , LTe, LOS <sub>e</sub>	Storage cost (€/year) Production costs (€/year) Maintenance costs (€/year)	◦	◦
P8 - Performance assessment	C <sub>T<sub>b</sub></sub> , U <sub>p</sub> , LOS <sub>s</sub>	Storage costs (€/year) Direct labour cost (€/year)	Required energy (kWh/year)	Environmental impact (CO <sub>2</sub> /year)
P9 - Workload prediction	C <sub>T<sub>b</sub></sub> , U <sub>p</sub> , LOS <sub>s</sub>	◦	◦	◦
P10 - Operations management	C <sub>T<sub>b</sub></sub> , U <sub>p</sub> , LOS <sub>s</sub>	◦	◦	◦

Table 21.1: KPIs to evaluate the solution of the problems in a production node.

robustness and steadiness of a production process. Bayesian statistic (see section 11.3) can be used to define the statistics of a KPI when only partially reliable measurements are available. Finally, simulation approaches (e.g. Monte Carlo see section 6.2.5) uses different input measurement to identify the behaviour of the output from a statistical point of view.

## 21.2 Workload prediction (P9)

Workload prediction aims at forecasting the value of the workload in the future. Companies rely on a demand forecast that is the starting point to derive all the other predictions.

### 21.2.1 Model-driven methods (D1)

Demand forecast depends on the type of demand pattern of a part. Identifying the right demand pattern can be seen as a clustering problem (i.e. solved using data-driven methods) [2]. Figure 21.1 identifies the four demand patterns.

Parts demand is classified by considering the variability of the quantity and the lead time between orders. These two metrics are identified by:

- $ADI = \sum_i^n \frac{\gamma_i}{n}$  ;
- $CV^2 = \left( \frac{\sigma_q}{\bar{q}} \right)^2$ .

Where  $\gamma_i$  is the interarrival time between two batches of goods;  $\sigma_q$  is the standard deviation of the quantity for each batch,  $\bar{q}$  is the average quantity

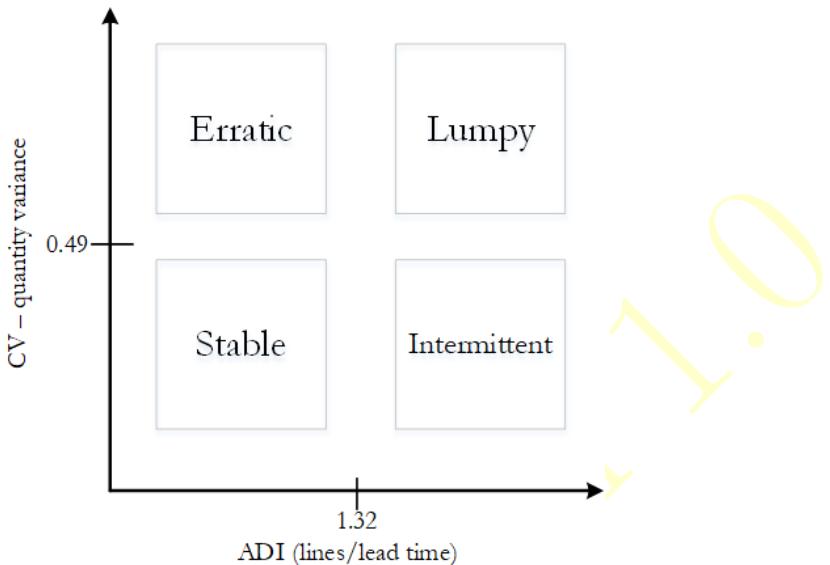


Figure 21.1: Demand patterns classification.

for each batch. Thresholds on these metrics are identified to classify the demand [3]:

- stable; these parts have a high demand frequency ( $ADI < 1.32$ ), and low variability in quantities ( $CV^2 < 0.49$ ). Their behaviour is stable and easy to forecast.
- Intermittent; these parts have a low demand frequency ( $ADI \geq 1.32$ ), and low variability in quantities ( $CV^2 < 0.49$ ). Their demand is extremely sporadic, and few data points are available making harder to predict their behaviour in the future.
- Erratic; these parts have a high demand frequency ( $ADI < 1.32$ ), and high variability in quantities ( $CV^2 \geq 0.49$ ). Their demand is frequent but extremely variant in quantities. Forecasts are possible but their accuracy remains a crucial issue.
- Lumpy; these parts have a low demand frequency ( $ADI \geq 1.32$ ), and high variability in quantities ( $CV^2 \geq 0.49$ ). Their behaviour is difficult to predict since there are few data points with great variability.

Depending on the nature of the pattern demand, different analytics should be used to address the demand prediction.

### 21.2.2 Data-driven methods (PD1, PD2)

Once the demand pattern has been identified, an appropriate predictive method should be chosen to make forecasts.

When dealing with stable and erratic parts, many data points should be available since these parts have a short average interarrival time (i.e. at least 20 data points per year). In this case, time series analysis is an appropriate methodology to analyse and forecast the demand series. In particular time series decomposition (see 6.5.1) is simple and appropriate. ARIMA models (see 6.5.2) work as well with some additional efforts in the data manipulation step since these models require stationary time series.

When dealing with intermittent and lumpy parts, the variance of the interarrival times is high, and it is hard to identify a robust sampling interval to estimate the demand (e.g. parts per week, per months, per year). In these cases, the probability distribution of the number of pieces per time unit cannot be approximated to the uniform, and a different distribution must be used to approximate the rareness of the events. For this reason, the Poisson distribution (see section 6.3.4) is used since it can well describe the behaviour of rare events.

The Poisson method identifies the probability of absorption of  $x$  parts within a time interval  $\tau$  having an average demand  $d$  calculated in a time interval of the same span of  $\tau$ . The Poisson distribution assumes  $\lambda = d\tau$ .

$$P_{d,\tau}(x) = \frac{\lambda^x e^{-\lambda}}{x!} \quad (21.1)$$

Alternatively, when a complex dataset  $X$  containing other attributes in addition to the time series, to train a learning model (see 9) can lead to more accurate results.

## 21.3 Job scheduling (P10)

When a production plant is ready to produce, a final word is needed to assign tasks to resources. Job scheduling aims at allocating the space and time of a resource to perform a task [4]. The nature of this problem requires a prescriptive solution which, usually, aims at minimising some objective function.

### 21.3.1 Model-driven methods (PS1)

Let  $c_i$  be the time instant where a part  $i$  is completed and  $d_i$  its completion due date, the production engineer aims at minimising the lateness  $L_i = c_i - d_i$ .

Many optimisation algorithms perform this assignment by setting the value of the decision problem  $P$ :

$$x_s = \begin{cases} 1 & \text{if job } j \text{ assigned to resource } i \\ 0 & \text{otherwise} \end{cases} \quad (21.2)$$

$$\min z \quad (21.3)$$

$$\sum_{i=1}^n p_{ij} x_{ij} \leq z, \quad i = 1, \dots, m \quad (21.4)$$

$$\sum_{i=1}^m x_{ij} = 1, \quad j = 1, \dots, n \quad (21.5)$$

$$x_{ij} \in \{0, 1\} \quad i = 1, \dots, m ; j = 1, \dots, n \quad (21.6)$$

The problem  $P$  usually embeds many other constraints to describe industry-oriented specifics which prevents from the feasibility of a solution (e.g. setup time, machines dedicated to specific tasks, scheduling priorities). Job scheduling is almost entirely prescriptive, and many contributions illustrate models, optimal and suboptimal algorithms to get feasible solutions [5].

## 21.4 Applications

This section illustrates the application of the methodologies above to design the control systems in two different production nodes. The first production node works in the catering sector, while the second operates in the 3PL of the automotive industry.

### 21.4.1 Control of a food catering plant

#### Performance assessment

The collection of historical data and time series from the company's ERP is a crucial activity to control system a catering plant. Unfortunately, the ERP of the company may not include the definition of the production cycles of the items. For this reason, a monitoring campaign helps to collect information to model the processes and to populate a relational database structured as in section 20.2.1. Figure 21.2 illustrates the ER structure obtained after monitoring the production processes. The database maps 213 customers, 29 delivery routes, 500 products, and more than 800 production orders on a time horizon of a week.

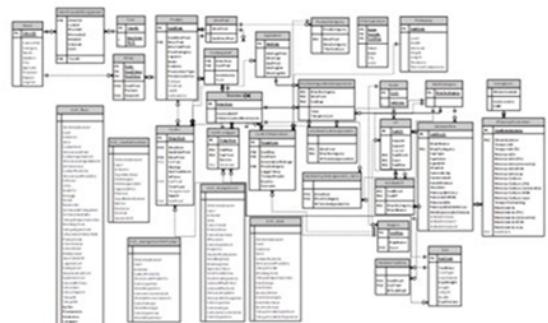


Figure 21.2: ER model of a production node.

The production processes have been, also, modelled from a qualitative point of view by using a business process model. Logistics and operations processes are highly related to the temperature profile of the products (cook-warm, cook-chill, cook-chill-re-warm). Figure 21.3 shows that depending on the temperature profile, products must follow different physical paths.

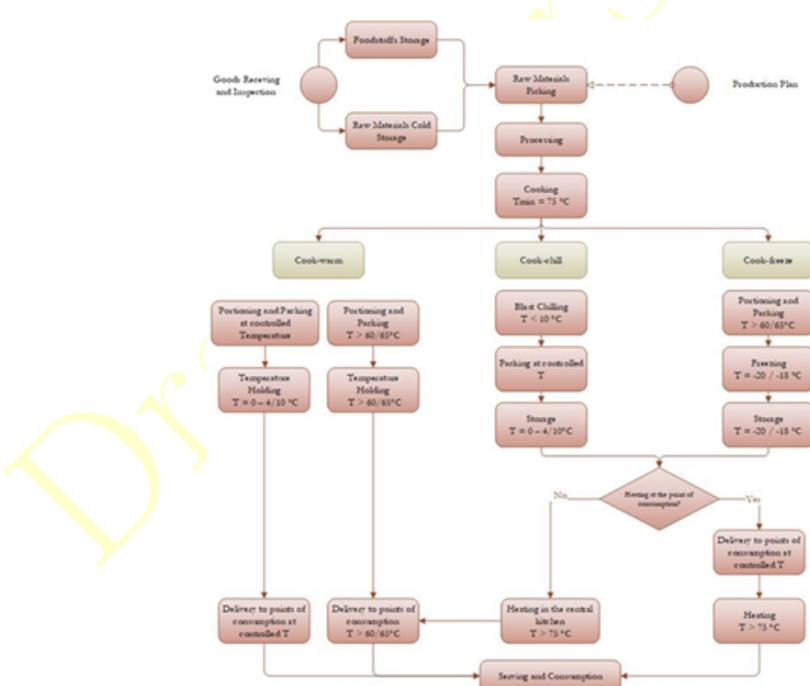


Figure 21.3: Demand patterns classification.

After modelling the processes, the production flows and tasks appeared more evident to the management and the analysts. It was, then, possible to perform a quantitative analysis by defining KPIs on the single product or process [1]. Figure 21.4 illustrates a visual KPIs representing the average processing time (on the x-axis) and the average cost (on the y-axes) for each product. They both are random variable represented as error bars of the dot-plot. Besides, the dashboard shows both the Pareto curve of average times and costs, highlighting that a small number of parts generates the most considerable workload in terms of time and costs.

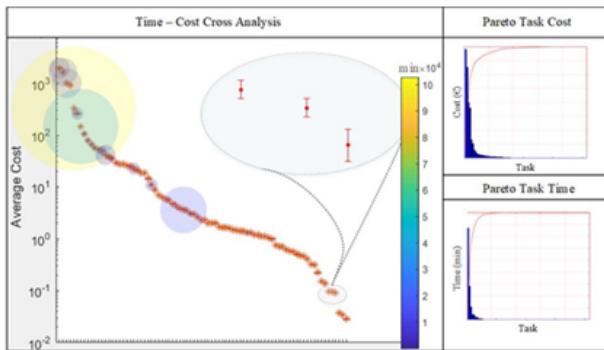


Figure 21.4: Dashboard of KPIs for time and costs of a catering plant.

Another visual analytic tool (see Figure 21.5) allows identifying the energy and environmental performance for each resource of the food plant. The histogram identifies the consumption of electric energy and gas, together with the estimate of the environmental impact measured in equivalent  $CO_2$  for each production task identified by the monitoring campaign. This way, it is possible to determine which tasks and which resources need to be carefully managed to limit energy consumption.

### Workload prediction

Once a set of KPIs to assess the performance of the system has been identified, it is necessary to classify and make forecasts on the demand that guide any other control or design step. Historical data are collected and organised to define the value of  $ADI$  and  $CV^2$  for each part. The dot-plot in Figure 21.6 identifies each product with a dot where the x- and y-axes represent the  $ADI$ , and  $CV^2$  calculated within the reference time horizon of the dataset (one week). The size of the dot represents the number of lines (i.e. the number of orders) for each part; the bigger the dot, the higher the number of orders. The heatmap on the right of Figure 21.6 identifies the demand patterns of the entire production system. In this case, the demand is mainly

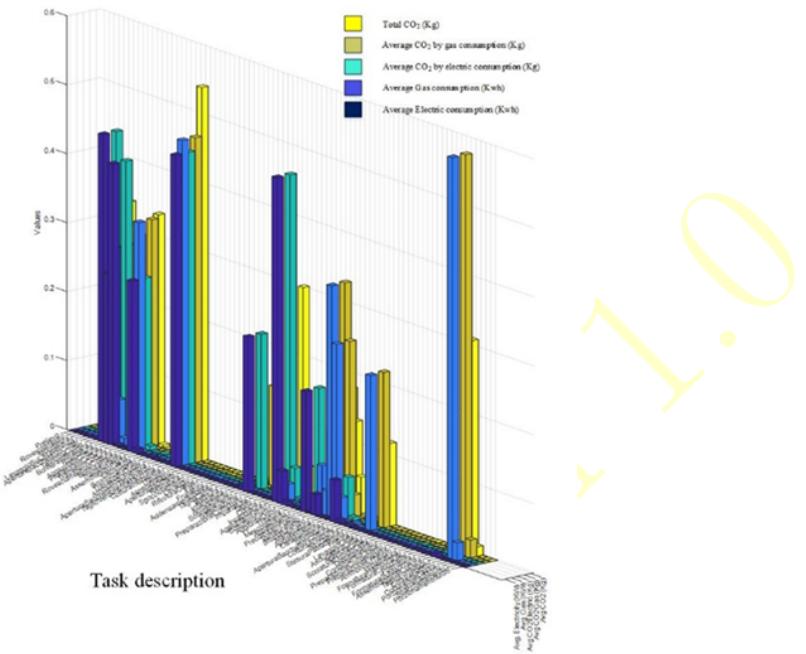


Figure 21.5: Energy and environmental impact for each production task.

intermittent, with low variability but significant time spans between orders of the same part. The production system has, then, to be flexible, allowing the production of the same average quantity of a vast product mix.<sup>1</sup>

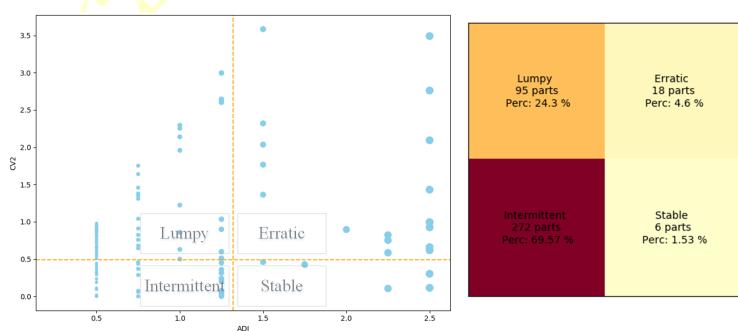


Figure 21.6: Classification and patterns of the demand for a food catering production plant.

<sup>1</sup>The source code of Figure 21.6 is available [here](#).

## Job scheduling

Job scheduling in food catering involves the pre-processing, cooking and packing departments of the plant. This activity deeply affects the quality of the finished products since the more the time they wait at the end of the production, the more the quality and taste decay due to the holding temperature [6]. Figure 21.7 graphically identifies the variable to control in the scheduling of a recipe: the resource  $r$ , the processing quantity and the cycle time  $c_j$  affect on the final time and temperature of the production lot.

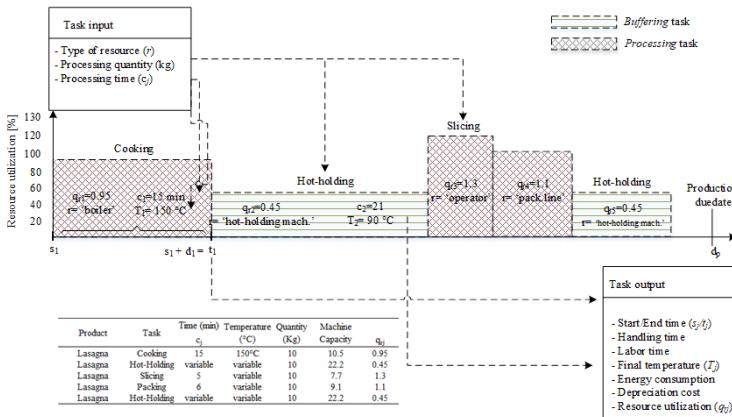


Figure 21.7: Scheduling variables in a food catering plant.

In a food context, the quality is an additional constraint of the job scheduling problem; for this reason, finding the optimal solution of the model presented in 21.3 becomes even more difficult since the problem is overconstrained. In addition, the minimisation of the makespan is only one of the objective function since the maximisation of the food is considered an objective as well. A multiobjective problem is proposed To meet both these purposes. It is solved by a metaheuristic algorithm called simulated annealing [7]. The solutions to the problem produce the Pareto frontiers built on five daily instances (one per each production day of the input dataset). The frontiers in Figure 21.8 show how preserving the quality of the food product is a much more sensitive objective than the minimisation of the makespan.

Finding the best food safety solution only slightly affect the scheduling from a makespan perspective while adding significant value to the quality of the finished product. For this reason, with particular reference to the end-of-line of cook-warm product, it is crucial to design areas with hot-holder able to maintain the quality and the food safety of the products. The hot

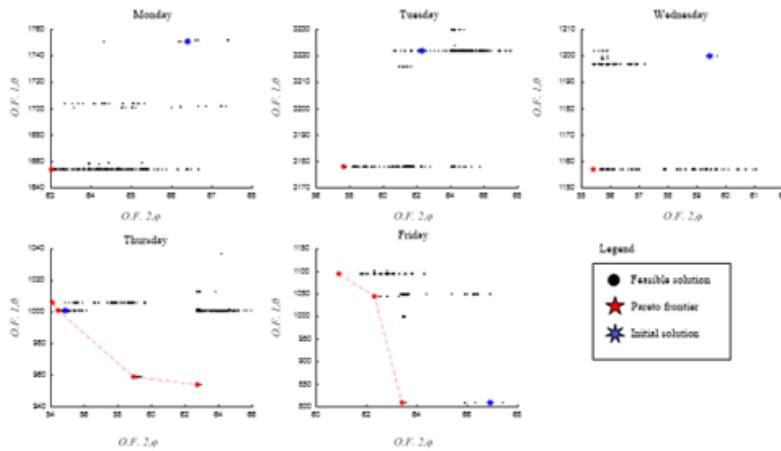


Figure 21.8: Pareto frontier of the bi-objective problem.

holding areas can be over-sized since their investment cost is limited but it reduces the risk of food safety and food loss due to quality and temperature decay.

### 21.4.2 Control of a 3PL packaging plant for automotive spare parts

#### Performance assessment

The management of this production node relies on a third-party company that performs the operations. Business processes are then fractioned, unclear, inefficient and even unknown to the managers. For this reason, the business process mapping and notation (BPMN) is used to investigate the physical and information flows between entities and actors involved in production and planning. Figure 21.9 illustrates an example of the BPMN together with the georeferencing of the processes on the plant layout. This technique enables visual analysis of the routes of different processes and identifies which resources and which areas are responsible for processing a specific task.

A dashboard of KPIs was defined to identify the workload and assess its impact on the working time and the distance travelled within the production site. The dashboard of KPIs in Figure 21.10 shows these metrics.

The graph in the left upper corner identifies the workload per week (number of processed lines); the bar chart in the right upper corner identifies which type of tasks are responsible for the workload in terms of time. The dot-plot in the left bottom corner identifies the trend between the number of processed lines and the distance travelled between the workstations to

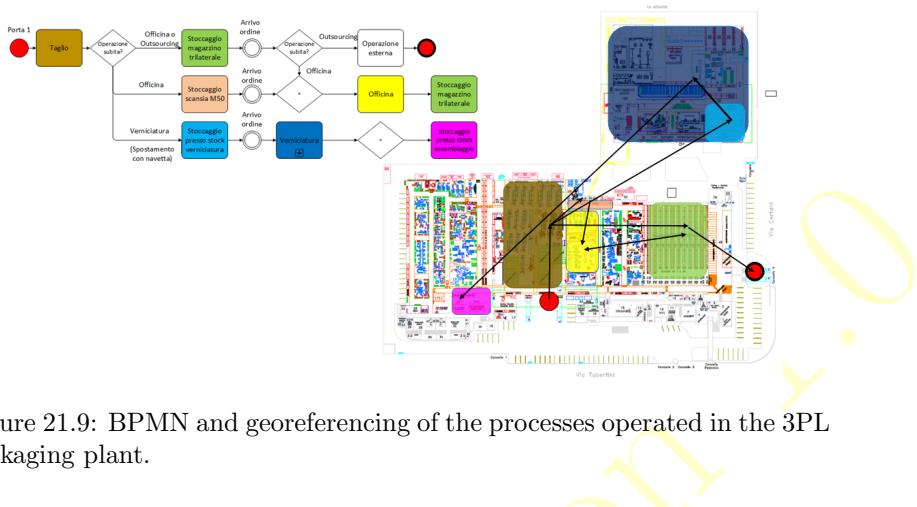


Figure 21.9: BPMN and georeferencing of the processes operated in the 3PL packaging plant.

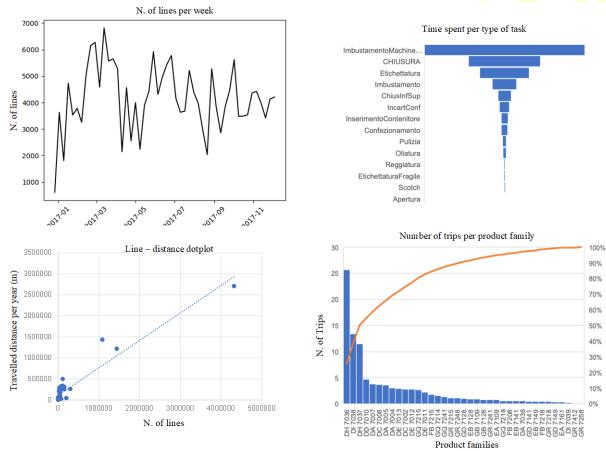


Figure 21.10: Control dashboard of the processes operated in the 3PL packaging plant.

handle those parts. The last Pareto chart in the right bottom corner shows that a minor amount of product families is responsible for the majority of the handling activities, in terms of the distance travelled between workstations.

### Workload prediction

The model in 21.2.1 is applied to identify demand patterns and choose the most appropriate methods to control and design the production node. Figure 21.11 classifies the demand of the production system, highlighting a

significant number of intermittent and lumpy parts.<sup>2</sup>

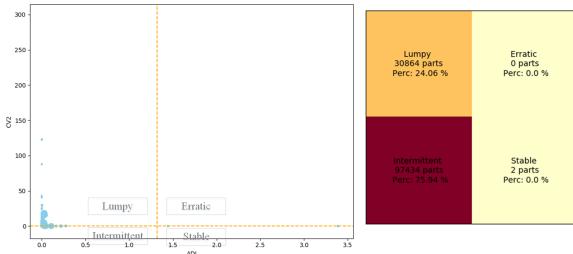


Figure 21.11: Classification and patterns of the demand for a 3PL packaging plant.

Given this pattern of demand, the production node must be flexible to meet productivity variance and time requirements. In addition, demand forecast must be carefully addressed since lumpy parts have significant variance.

The demand trend is assessed by using descriptive analytics techniques. Figure 21.12 identifies the global trends of quantities and the number of lines processed by the entire production plant. The trends are aggregated weekly and daily.<sup>3</sup>

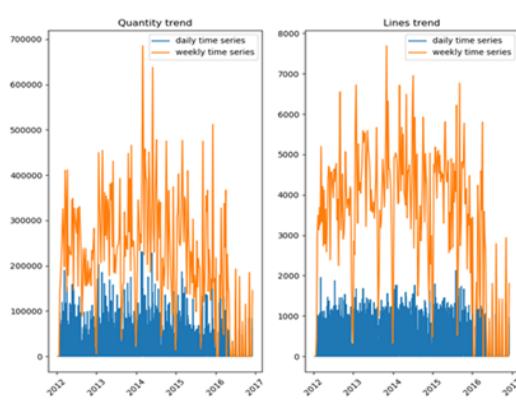


Figure 21.12: Daily and weekly trend of the demand in a 3PL packaging plant.

Decomposition is applied, and the trend, seasonal and residual compo-

<sup>2</sup>The source code of Figure 21.11 is available [here](#).

<sup>3</sup>The source code of Figure 21.12 is available [here](#).

nents are identified. Due to the length of the time series (more than six years), the series is weekly aggregated before the decomposition.

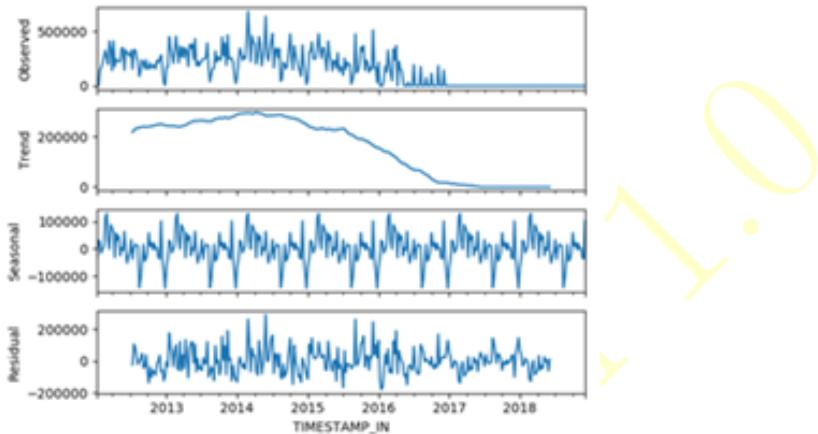


Figure 21.13: Decomposition of the market demand in a 3PL packaging plant.

Figure 21.13 illustrates the decomposition showing a descending trend component and a seasonality occurring twice a year. The residual component is significant; this is due to the large variability of the demand already identified with the demand patterns in Figure 21.11.<sup>4</sup> The yearly trend is predicted using the *fbProphet* prediction model. Figure 21.14 illustrates the predictions and confidence intervals. ARIMA models cannot be applied to this time series since no transformation (square root, power, log, boxcox) leads to a stationary series.<sup>5</sup>



Figure 21.14: Prediction of the market demand in a 3PL packaging plant using the *fbprophet* algorithm.

<sup>4</sup>The source code of Figure 21.13 is available [here](#).

<sup>5</sup>The source code of Figure 21.14 is available [here](#).

## Bibliography

- [1] A. Tufano, R. Accorsi, A. Gallo, and R. Manzini, "SIMULATION IN FOOD CATERING INDUSTRY . A DASHBOARD OF PERFORMANCE," in *International Food Operations and Processing Simulation Workshop*, pp. 20–27, 2018.
- [2] M. M. Mikalsen, "Managing the Demand for Spare Parts," tech. rep., Implement Consulting Group.
- [3] A. A. Syntetos, J. E. Boylan, and J. D. Croston, "On the categorization of demand patterns," *Journal of the Operational Research Society*, vol. 56, no. 5, pp. 495–503, 2005.
- [4] J. S. H. Lim, D. C. Y. Foo, D. K. S. Ng, R. Aziz, and R. R. Tan, "Graphical tools for production planning in small medium industries (SMIs) based on pinch analysis," *Journal of Manufacturing Systems*, vol. 33, no. 4, pp. 639–646, 2014.
- [5] M. L. Pinedo, *Planning and Scheduling in Manufacturing and Services*. 2009.
- [6] A. Tufano, R. Accorsi, and R. Manzini, "A Simulated Annealing algorithm for the allocation of production resources in the food catering industry," *British Food Journal*, 2020.
- [7] L. B. Reinhardt, T. Clausen, and D. Pisinger, "Synchronized dial-a-ride transportation of disabled passengers at airports," *European Journal of Operational Research*, vol. 225, no. 1, pp. 106–117, 2013.

Draft Version 1.0

# 22

## Plant Design

This chapter and the following one address the design from scratch (i.e. greenfield) of a production node. Greenfield design is often a hard task since it involves risk, uncertainty and the definition of many assumptions which are hard to set since the production node only exists in our minds.

In general, this activity is complicated. Luckily, there are some general rules and framework that can help in this activity. When historical data from previous experience (or similar production nodes) are available, the data-driven approach helps to work with few but robust information. When historical data are not available, the definition of a kinematic/engineering model may help to get trustful estimates.

This chapter focuses on the problems related to the design of the plant:

1. clustering items into product families;
2. facility location;
3. technology and asset choice;
4. auxiliary systems design;
5. definition of the number of assets;
6. layout design.

Chapter 23 focuses on the problems related to the design of the processes:

1. Inventory policy design;
2. Workstation design;
3. Handling design.

Figure 22.1 presents a cascade procedure, inspired to [1] showing these nine tasks for the design of a greenfield production node. This procedure is known in the literature as “facility design” and guides all the activity to transform a greenfield to a production plant.

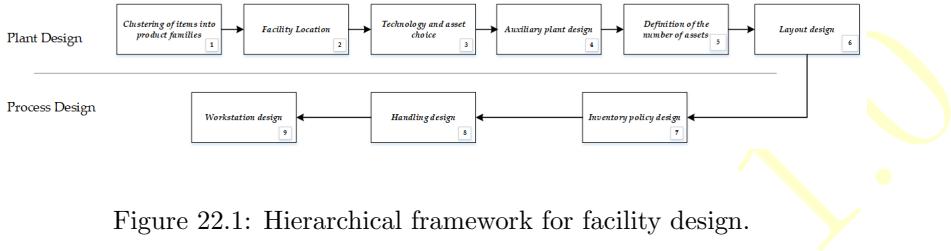


Figure 22.1: Hierarchical framework for facility design.

## 22.1 Clustering parts into product families (P1)

Engineering is about complexity. Companies rely on engineers and engineering science since they are able to use tools and method to:

1. pickup a complex problem;
2. reduce its complexity;
3. understand the problem;
4. find a method to solve the problem;
5. solve the problem.

Production nodes are responsible for creating goods, i.e. parts  $i$ . Globalisation and mass customisation trends lead production nodes to incredibly high levels of complexity. This complexity is measurable in terms of:

- The size of the product portfolio (i.e. the number of different parts  $i$ );
- The production volume  $Q_{G,i}$  (i.e. the number of items produced by the plant  $G$ , for each type of part  $i$ ).

These metrics are important to identify an adequate production technology, with a proper throughput  $TH_j$  for each resource  $j$ ; but this task can be hard when the number of parts  $i$  is high (e.g. more than 1000). For this reason, it is good practice to cluster items into homogeneous families with similar features in order to reduce the complexity of the problem and focus on similar entities. According to Section 4.2, this is a version of the family problem, applied to a production system.

### 22.1.1 Model-driven methods (D2)

The design of a production system is all about creating an environment for a smooth and efficient production of goods. Empirically, the evidence shows that grouping parts  $i$  with similar production cycle (i.e. route  $e$ ) is an excellent way to improve efficiency. Model-driven approaches rely on the definition of an incidence matrix  $X_{ij}$ , defined as:

$$X_{ij} = \begin{cases} 1 & \text{if part } i \text{ needs resource } j \text{ in its production process} \\ 0 & \text{otherwise} \end{cases} \quad (22.1)$$

The diagonalisation of the matrix  $X_{ij}$  into a matrix  $X_{ij}^\delta$  is used to identify groups of homogeneous parts (see Figure 22.2).

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10
Part 1	1			1	1					1
Part 2					1					
Part 3			1	1					1	
Part 4	1	1	1	1				1		
Part 5	1	1	1					1		
Part 6	1				1	1				
Part 7		1							1	
Part 8	1	1				1	1	1		
Part 9		1		1	1		1		1	
Part 10	1					1	1	1		

	R2	R1	R8	R3	R4	R10	R9	R6	R7	R5
Part 8	1	1	1				1	1		
Part 10			1	1				1	1	
Part 4	1	1	1	1	1	1				
Part 5	1			1	1			1		
Part 9	1			1			1			
Part 1		1		1	1	1	1			
Part 3	1			1	1					
Part 6		1						1	1	
Part 7	1							1		
Part 2									1	

Figure 22.2: Diagonalisation of a part-resource incidence matrix.

Any diagonalisation algorithm is suitable to identify clusters of parts and resources since the model assumes that operations can be improved when the clusters of parts using the same subset of resources are handled together. For the sake of completeness, we mention the direct clustering algorithm [2], and the rank order clustering algorithm [3] that produce the diagonalization of  $X_{ij}$ .

Once product families  $\pi \in \Pi$  are defined, the rest of the design of the production node should consider the family  $\pi$  instead of all the parts  $i \in \pi$ , so that the complexity due to the variety of the size of the product portfolio is reduced. In case, the number of families is still difficult to handle (e.g. higher than 100 families). A Pareto analysis may help. For example, if a small subset of families (e.g. the 20%) produces the major amount of the volumes  $Q_{G,i}$ , it is good to consider only the first 20% of the families to reduce the complexity and the bias in the other design stages (see Figure 22.3).

Clustering does not guarantee cluster are feasible in practice. For example, a cluster to be assigned to a group of machines may exceed the available working time of the resources. To solve this feasibility problem, an original

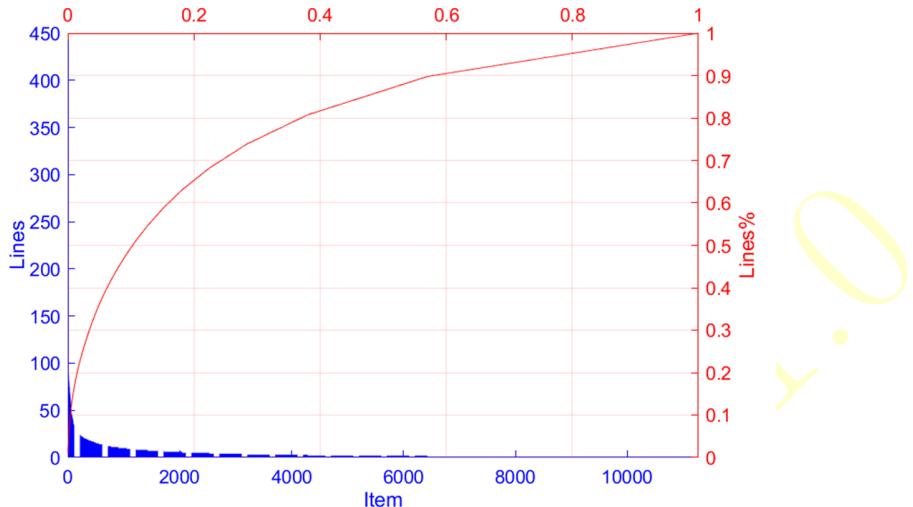


Figure 22.3: Pareto curve of a production plant where the first 40% of the items produces the 80% of the production volume (i.e. the number of production lines)

capacitated clustering algorithm is proposed [4]. The algorithm is inspired to hierarchical clustering with a capacity constraint. Let  $d_i$  be the *demand* (e.g. the total working time) of a point (i.e. a part)  $i$  and  $C$  the maximum capacity of a cluster. The algorithm works similarly to algorithm 11<sup>1</sup>. The outcome of this algorithm is a set of clusters whose cardinality is unknown in advance. Each cluster  $k$  has a total demand  $d = \sum_{i \in k} d_i$ , with  $d \leq C$ .

### 22.1.2 Data-driven methods (D1)

The matrix  $X_{ij}$  can be seen as the learning table introduced in chapter 8. It consists of  $n$  observations (one for each part  $i = 1, \dots, n$ ) and  $p$  features (one for each resource  $j = 1, \dots, p$ ). Data driven-methods enhance more powerful applications, by defining the similarity of the resources  $j$ , given the set of parts they can process. The incidence matrix  $X_{ij}$  can, then, be transformed into a proximity matrix  $D_{ik}$  (where  $i$ , and  $k$  are both parts) by using a similarity index (see section 8.2.2). Similarity indexes were born in the biological, and are defined using:

- $a$ , the number of resources processing by both  $i$ , and  $k$ ;
- $b$ , the number of resources processing only by  $i$ , and not by  $k$ ;

<sup>1</sup>The source code of Algorithm 11 is available [here](#)

- $c$ , the number of resources processing only by  $k$ , and not by  $i$ ;
- $d$ , the number of resources processing neither by  $i$ , nor by  $k$ .

Using  $a$ ,  $b$ ,  $c$  and  $d$ , the matrix  $D_{ik}$  is defined for all the couples of resources  $(i, k)$  using  $d_{ik} = 1$ , where  $s_{ik}$  is a similarity index. The most famous is the Jaccard index:

$$s_{ik} = \frac{a}{a + b + c} \quad (22.2)$$

For the sake of completeness, there are other many other definitions of similarity coefficients  $s_{ij}$ , as: the simple or relative matching coefficients [5], Yule coefficients, Rogers and Tanimoto coefficients [6], Baroni-Urbani coefficients [7], Sorenson coefficient [8]. Given  $D_{ik}$ , hierarchical clustering (see section 8.2.2) is used to agglomerate parts into clusters.

The same procedure can be applied to create clusters of similar resources  $j$  to locate close to each other on the plant layout. So far, the data-driven approach does not add too much to the classical model-driven approach.

There are unlucky but ordinary circumstances where the route  $e$  of a product is unknown (e.g. not available in the information system during the planning). In this case, the data-driven approach is useful by combining different type of data to determine product families. There are many features that can enter the matrix  $X_{ij}$ , for example:

- the set of processing resources  $j$ ;
- the size of the part  $i$ ;
- the volume and weight of  $i$ ;
- the description of the item  $i$ ;
- the package or the vehicle needed to transport  $i$ ;
- the supplier of  $i$ ;
- the customer of  $i$ ;
- the bill of materials of  $i$ .

Given a matrix  $X_{ij}$  with all the available information, it is necessary to investigate (e.g. using historical data) if a correlation exists between one of the features and the route  $e$ . If a correlation exists, feature selection should be performed, and unsupervised learning algorithms can be applied. Many unsupervised learning algorithms can be used as:

- the k-means (see section 8.2.1), suitable when all the data are numerical and with the same unit of measure (e.g. length, height and width);

- the Gaussian mixture model (see section 8.2.3), suitable when all the data are numerical and with the same unit of measure (e.g. length, height and width) and values are normally distributed within the same cluster;
- hierarchical clustering (see section 8.2.2), when data are categorical or provided by a binary incidence matrix  $X_{ij}$  or a proximity matrix  $D_{ik}$ .

Measuring the goodness of a cluster is a hard task both using a data-driven and a model-driven approach. The underlying assumption is that when clusters are homogeneous, the  $WIP_j$  of the resources is minimised since flows are smooth. In addition, if resources processing the same product family are placed close to each other, the  $LT_e$  is reduced and provides a higher  $LoS_e$ .

## 22.2 Facility location (P6)

Facility location problem regards the definition of an adequate location (i.e. latitude and longitude) to place a production node. The production plant should be placed within an area where the costs are minimised. This type of decision is highly prescriptive and guided by engineering models since it is difficult to collect data on the previous realisation of this choice.

### 22.2.1 Model-driven methods (PS4)

The main underlying assumption of these models is that a plant should be placed such that the costs linked with its location are minimised. Some other crucial aspects connected with the facility location are:

- the cost of the direct labour of a location;
- the cost of the energy;
- the cost of the land and the building;
- the connection to distribution networks (e.g. rail/water)
- the availability of raw materials (e.g. sand)
- the transportation costs.

This problem can be solved using optimisation when all this information is available for any alternative. When alternative locations are close to each other, transportation cost may be the only significant variable to take into account (and to minimise) in the definition of the facility location.

In this case, the problem can be solved by using a kinematic model based on the minimisation of the travelled distance. Let  $S$ , with  $s \in S, s = 1, \dots, k$  be the set of customers and suppliers of the production plant to place. Each point  $s$  is characterised by its longitude and latitude  $(lon_s, lat_s)$ , and an estimate of the number of trips  $w_s$  travelled between the production node and  $s$ . To find the point minimising the distance, it is necessary to transform the longitude and the latitude into cartesian coordinates. There are many methods allowing to do that. Here the Mercator projection [9] is used, where:

$$x_s = R \times lon_s^{RAD} \quad (22.3)$$

$$y_s = R \times \ln \left[ \left( \frac{1 - e \times \sin(lat_s^{RAD})}{1 + e \times \sin(lat_s^{RAD})} \right)^{\frac{e}{2}} \times \tan \left( \frac{\pi}{4} + \frac{lat_s^{RAD}}{2} \right) \right] \quad (22.4)$$

Where  $lon_s^{RAD}$  and  $lat_s^{RAD}$  are the coordinates using radians,  $R = 6378.14$  km is the equatorial radius,  $e = 0.0167$  is the eccentricity of the Earth. Given  $x_s$  and  $y_s$  for all  $s \in S$ , the problem is to find the coordinates  $(x, y)$  to place the production plant, minimising:

$$z = \min \{ d[(x, y), (x_s, y_s)] \times w_s \} \quad (22.5)$$

Where  $d$  is an arbitrary distance function, for example:

- Rectangular (or city block) distance:  $|x - x_s| + |y - y_s|$ ;
- Euclidean distance:  $\sqrt{(x - x_s)^2 + (y - y_s)^2}$ ; item Squared Euclidean (or gravity) distance:  $(x - x_s)^2 + (y - y_s)^2$ .

The mathematical minimum can be found by minimising  $z$  [10]. Nevertheless, it is almost impossible that this point coincides with a feasible physical location for the plant. For this reason, a gradient approach results much more useful to support the decision-maker. Let define a set  $\Phi = (x_f, y_f)$  containing the coordinates (using equations (22.3) and (22.4)) of a number of available locations  $f$ . By considering a broader geofence, containing all the points in  $\Phi$  and defining the value of  $z$  for any point within the geofence, the gradient of  $z$  can be identified. This way, the decision-maker could identify which direction is better to find an area for the new production plant. Many alternative procedures to minimise  $z$  depending on  $d$  can be found in [11].

Nowadays, due to the variability of the market demand and the shortness of the duration of the contract, the reliability of a static approach to

solving the facility location problem may be reduced. As well as with time series, which considers the evolution of an event over time, by repeating the calculation of the optimal location with different time horizon it is possible to define optimal location using a probabilistic approach. Let  $z^*$  be the set of optimal locations  $(x_t^*, y_t^*)$  calculated on a time horizon  $T$ , where  $t \in T$ . Then, the minimum of the function:

$$z^\pi = \min \left\{ \sum_{t \in T} d[(x, y), (x_t^*, y_t^*)] \right\} \quad (22.6)$$

Identifies the optimal location over a time horizon  $T^2$ .

## 22.3 Auxiliary systems design (P4)

Auxiliary systems are any technological asset that does not directly add value to the finished product, but it is necessary to perform tasks. Examples of the auxiliary systems are:

- the lighting systems of the production site;
- the air conditioning system (i.e. heating and cooling);
- the systems for the production of energy (e.g. electricity/steam),
- the systems for the production of other technological fluids (e.g. compressed air);
- the systems for the reduction of the noise.

All of these systems involve precise prescriptive design models based on engineering models. These models cannot be validated before the realisation of the system that is usually costly. For this reason, the whole design phase relies on engineering procedure based on indices and coefficient belonging to different knowledge domains that are not deeply analysed in this work [12].

## 22.4 Technology and asset choice (P2)

This activity involves the determination of the proper technologies and level of automation to perform production tasks. This is a technology assignment problem since, given the features of the products and the demand, it is necessary to identify the throughput  $TH_j$  for each resource and the lead times  $LT_e$  associated with the production cycles. Depending on the level of automation and flexibility, it is possible to identify different production, layout and automation paradigms (see Figure 22.4) [13].

---

<sup>2</sup>The package logproj provides method to deal with facility location problems [here](#)

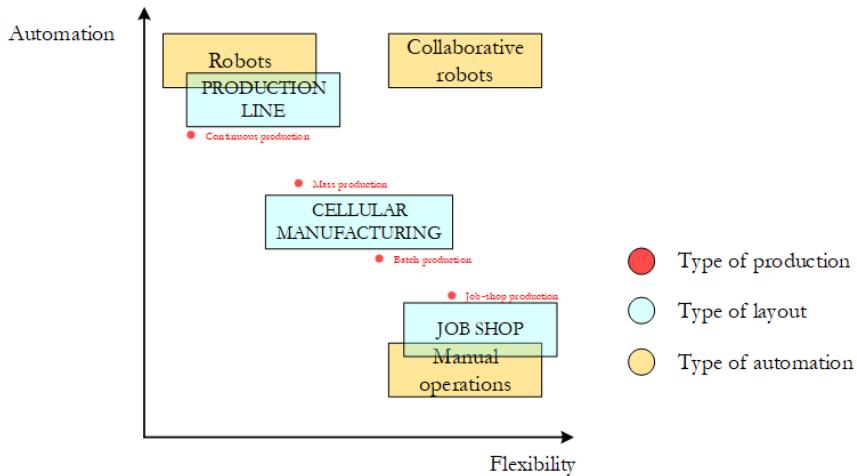


Figure 22.4: Flexibility-automation matrix for technology choice.

Three main layout configurations are identified:

1. the production line: all the resources necessary to transform raw materials into a finished product are placed in line.
2. cellular manufacturing: all the resources necessary to transform raw materials into a limited set of finished products are placed together. There is the possibility some tasks need resources outside the cell.
3. resources are organised per type (milling machines, lathing machines, pressing machines).

There are four production paradigms connected to these layout configurations:

- continuous production: the production flow is continuous, with a fixed cycle time;
- mass production: the production flow is fast but not continuous since it requires small customisations (e.g. form-postponement, different labelling) in the end-of-line;
- batch production: the production flow has major interruptions due to the setup of machines (e.g. to change the die of a press);
- job-shop production: the production flow is slow and fragmented among the different departments.

The type of automation is different as well, since:

- fully automated production line works with robots allowing high accuracy and modest cycle time, but with no flexibility to readapt the production;
- collaborative robots may work together with human operators in many applications (e.g. assembly lines) enhancing both the power of the automation and the flexibility of the manual operations;
- manual operations have the highest degree of flexibility, but with limited cycle time and a significant probability of errors.

The decision-maker should choose the possibilities that maximise the profit of the company within a pre-defined time horizon. In particular, it is important to have reliable forecasts on the workload and information on the investment cost for different technologies.

#### 22.4.1 Model-driven methods (PS3)

Model-driven methods consider two metrics (already seen in section 22.1) to identify an adequate technological configuration:

- the size of the product portfolio (i.e. the number of different parts  $i$ );
- the production volume  $Q_{G,i}$  (i.e. the number of items produced by the plant  $G$ , for each type of part  $i$ ).

By performing a Pareto analysis on the production volume for each part (see Figure 22.5):

- few parts with high production volumes should be assigned to production lines;
- the majority of parts with extremely low volumes should be processed in the departments of a flow shop;
- cellular manufacturing, flexible manufacturing systems (FMS) and reconfigurable manufacturing systems (RMS) should be considered for parts with an intermediate behaviour.

For each product, it is possible to identify the decoupling production volume  $TH_i$  (part/hours) corresponding to an economic convenience between a production line and a job-shop production [14]. Let the saturation of a resource  $j$  be:

$$U_j(TH_i) = \frac{TH_i \times t_j}{n_j(TH_i) \times 3600} \quad (22.7)$$

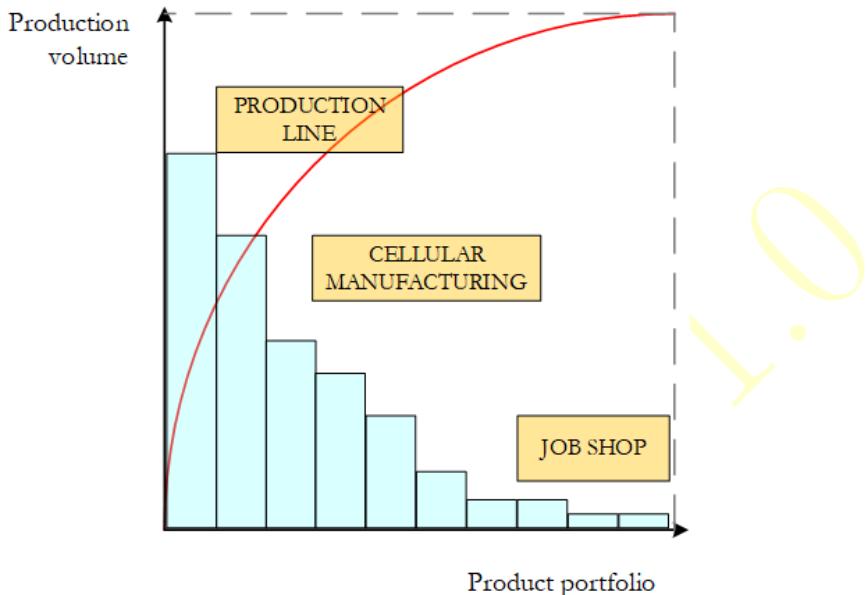


Figure 22.5: Classification of parts for technology and asset choice.

Where  $TH_i$  is the production throughput (parts/hours),  $t_j$  is the processing time per part (sec/part),  $n_j(TH_i)$  the number of resources of type  $j$ , necessary to reach a throughput  $TH_i$ , 3600 is the number of available seconds per hour of resource  $j$ .

Considering a target saturation  $K$  for a production line (e.g. the 90%), the decoupling production volume for a part  $i$  is  $U(TH_i) : U(TH_i) > K$ , where:

$$U(TH_i) = \frac{\sum_j U_j \times n_j(TH_i)}{\sum_j n_j(TH_i)} \quad (22.8)$$

Resources  $j$  are usually some tens. When they have a very different investment cost  $C_J$  between each other, it is recommended to take into account this cost by setting:

$$U(TH_i) = \frac{\sum_j U_j \times C_j \times n_j(TH_i)}{\sum_j C_j \times n_j(TH_i)} \quad (22.9)$$

Once the function  $U(TH_i)$  is defined, the decision between production line or job-shop should be made by setting  $TH_i$  equal to the demand takt-time. If  $U(TH_i) > K$ , then there are good reasons to take into account the design of a production line. Otherwise, other layout models (cellular manufacturing or job-shop) should be considered.

### 22.4.2 Data-driven methods (PS2)

The choice of the production technology and the assets can be seen as an assignment problem where parts  $i$  need to be assigned to an adequate technology. Observations of parts belonging to different production models help to identify the proper cluster. Classification models (see chapter 10) can be used for the definition of the decision boundaries between the clusters. Figure 22.6 illustrates a scatterplot identifying three different production technology. Blue dots are processed within the departments of a job-shop system. Red dots are products realised by machines organized ad production cells and flexible manufacturing systems (FMS). Green dots are processed on manual workbenches able to perform any production task for any type of product.

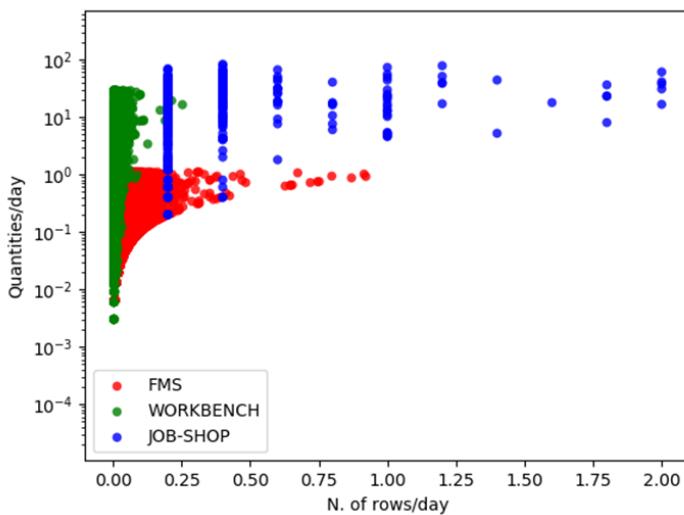


Figure 22.6: Classification of parts using the number of orders and the number of produces quantities per day.

The definition of such clusters within the same production node allows to immediately identify which layout configuration results suitable the most in case of re-design. In addition, the design of a classification model (see section 10), can help to design the production flow of new products depending on the expectation of their market demand (i.e. the number of rows and the order quantity).

## 22.5 Definition of the number of assets (P5)

The definition of the number of assets identifies the power (i.e. the capacity) and the saturation of the resources. The utilisation  $U_j$  must be as higher as possible for economic reasons, while the capacity  $C_j$  should be enough to satisfy the market demand. Both these metrics impact the level of service since capacity act as a buffer (higher capacity allow for processing a production volume within a shorter time). As many power problems in other engineering disciplines, this problem is prescriptive and lead by an engineering model.

### 22.5.1 Model-driven methods (PS4)

Models to define the number of assets can be static or dynamic. When no information about the time is available, a static model should be chosen. Static models always work by considering:

- $a_j$ , the amount of time a resource  $j$  is available over a time span (e.g. hours/day);
- $b_j$ , the expected amount of working time required by the parts  $i$  that need to be processed on  $j$  (e.g. hours/day).

The minimum number of assets of type  $j$  can be statically calculated as:

$$n_j = \left\lceil \frac{b_j}{a_j} \right\rceil \quad (22.10)$$

When the decision-maker has time-based information as:

- the probability distribution  $f_i(t)$  of the arrival of parts for each resource  $j$ ;
- the probability distribution  $g_{ij}(t)$  of the working time of each resource  $j$ .

A simulation approach can be used to identify the number of resources virtually. Simulation is a complex task performed on commercial simulation software. Each resource is associated with a queue where parts  $i$  are placed when the resource is busy. By randomly generating parts  $i$  and processing time (from  $f$ , and  $g$ ) operations are virtualised, and the number of items waiting in queue is estimated.  $U_j$ ,  $C_j$  and  $LoS_e$  can be estimated as well by iterating instances of the simulation. The probability distributions of the KPIs and the queue size suggest the decision-maker if the number of assets in the simulated configuration is appropriate or should be modified. By running different scenarios with a different number of assets, a satisfying configuration can be obtained.

## 22.6 Layout design (P6)

The layout design is a placement problem involving the definition of the location on the plant layout for each resource  $j$ . It is a combinatorial problem, usually solvable by a model-driven approach using optimisation.

### 22.6.1 Model-driven methods (PS1)

The combinatorial problem can be modelled using the quadratic assignment problem as follows. Table 22.7 identifies the parameters of the problem.

Parameter	Description
$j \in C : j = 1, \dots, h, \dots n$	Set of resources
$l_k \in L : k = 1, \dots, l, \dots n$	Set of available locations
$c_{jkh}$	Cost of assigning $j$ to $k$ , when $h$ is assigned to $l$

Figure 22.7: Parameters of the quadratic assignment problem.

The model is as follows.

$$X_{jk} = \begin{cases} 1 & \text{if } j \text{ assigned to } k \text{ process} \\ 0 & \text{otherwise} \end{cases} \quad (22.11)$$

$$\begin{aligned} \min \sum_{j \in C} \sum_{k \in L} \sum_{h \in C} \sum_{l \in L} c_{jkh} x_{jk} x_{hl} \\ \sum_{j \in C} x_{jk} = 1, k = 1, \dots, n \\ \sum_{k \in L} x_{jk} = 1, j = 1, \dots, n \\ x_{jk} \text{ integer} \end{aligned} \quad (22.12)$$

By setting  $c_{jkh}$  equal to the distance between control points  $k$ , and  $l$ , times the number of trips exchanged between resources  $j$ , and  $h$ , the problem consists in finding the layout configuration which minimises the total travelled distance. Unfortunately, the problem is quadratic and a solver may take forever to find the configuration of decision variables  $x_{jk}$  corresponding to the minimum solution value. For this reason, a number of suboptimal algorithms are introduced to find adequate configuration (without the warranty of optimality). These algorithms are organized into:

- construction algorithms, to find a feasible solution starting from the input data;
- local search algorithms, to improve an existing feasible solution.

### Construction algorithms

Construction algorithms split the problem of the layout configuration into two different subproblems:

1. control points ranking;
2. control points placement.

Relevant methods are the total-closeness-rating, the ALDEP [15], the CORELAP [16] and the relationship diagramming method [17]. All of these have a ranking and a placement procedure.

### Local search algorithms

The solution produced by a construction algorithm may be far from optimality. For this reason, local search algorithms perform *moves* (i.e. little modification of the incumbent solution) to improve the solution value.

Local search algorithms are the 2-opt, 3-opt exchange [18] and CRAFT [19]. The 2-opt and 3-opt are general-purpose local search methods that make exchanges between groups of 2 or 3 elements.

Regardless of the methodology used to solve the plant layout problem, a graph  $G(V, A)$  of the system can be defined and visually investigated to identify criticalities. The flows exchanged between control points can be aggregated by arcs or nodes and represented by projecting  $G$  on the plant layout (see Figure 22.8)<sup>3</sup>.

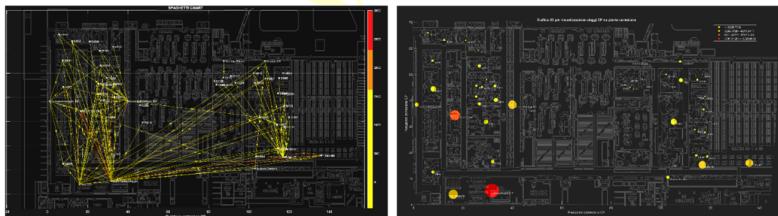


Figure 22.8: Visual representation of the material flows aggregated by arcs (on the left) and by nodes (on the right).

## 22.7 Applications

This section illustrates two applications of plant and process design within two different industrial sectors (food and automotive). The aim is to show the generalisation of the methods proposed in the previous chapters, and

---

<sup>3</sup>The source code to deal with the representation of a from-to matrix is available [here](#).

the effect of the data-driven methods in applied design case studies. In particular, the first case study is mainly model-driven, while the second is more data-driven.

### 22.7.1 Plant Design of a food catering plant

This section applies the methodology to design a food catering plant from scratch. The food catering industry is responsible for supplying warm, safe, and tasty food to schools, hospitals, and nurseries, as well as to company canteens. The production plant of a food catering industry is known as a *centralised kitchen* (CEKI).

Typically, a middle-size CEKI can prepare, pack, and deliver more than 10.000 meals per day by working in a short-spanned time batch (e.g., from 5.00 AM to 11.30 AM for the lunch service); it provides food service within a short-range, usually at distances under 30 km. The type of service is also characterized by the temperature at which the meals are distributed. Three alternative temperature profiles are possible:

- cook-warm: the product is cooked and maintained warm (above 65 °C) until it is consumed;
- cook-chill: the product is cooked, blotted, and delivered to the customer who re-warm it before service [20];
- cook-chill-re-warm: this is when cook-and-chill products are rewarmed at the CEKI and delivered according to the cook-warm profile.

Among these profiles, cook-warm is the most critical for safety issues because the product's temperature must be maintained above the danger zone (4–65 °C) at which bacteria propagate exponentially [21]. The cook-warm meals should be conserved at conditions that are outside the danger temperature range to comply with safety rules and standards from the time they are produced until their consumption and should be necessarily subjected to continuous and expensive hazard analyses and critical control points (HACCP), as well as monitoring and control of the processing tasks.

A CEKI can implement all these three production temperature profiles leading to a more complex organisation of the operations. The assessment of the current production environment is a fundamental step to properly design a new plant. This procedure has already been illustrated in 21.4.1.

#### Clustering products into families

For analysing the extant production mix, the previous product families are considered. Product families are based on the commercial class to which a product belongs. Different families are defined for the components of a

menu (e.g. main dish, side dish, fruit, dessert, vegetables, mineral water). Figure 22.9 illustrates the relative importance of each product family by using a Pareto analysis.

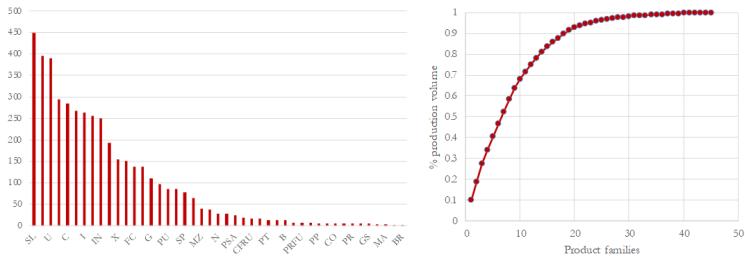


Figure 22.9: Histogram and Pareto chart of the product volumes within each product family.

Figure 22.9 show that few product families generate the majority of the product mix. This is a key metric to consider to improve the design in the following steps.

### Facility location

A similar profiling activity has been performed on the points of demand to group them according to the temperature profile(s) they demand. This activity leads to the decision of designing a CEKI able to realize products belonging to all the three temperature profiles. Then, by considering the number of trips to every single point of demand, it was possible to define the optimal location. Euclidean distances have been used since empirical tests showed that they were the most accurate estimate of the real distances in this area. Figure 22.10 illustrates the output of the facility location investigation identifying an optimal point for each time period (e.g. a week). This analysis evidences that the network suffers changes of the flows intensity exchanged between the plants and the nodes of the network.<sup>4</sup>

### Auxiliary systems design, technology and asset choice, definition of the number of assets and layout design

Due to the peculiarity of the catering industry, a wide number of professionals with various competencies and backgrounds were involved in the design phase. To support the design of the new plant and to investigate how their decisions affect each other, a comprehensive decision support system (DSS)

<sup>4</sup>The source code of Figure 22.10 is available [here](#).

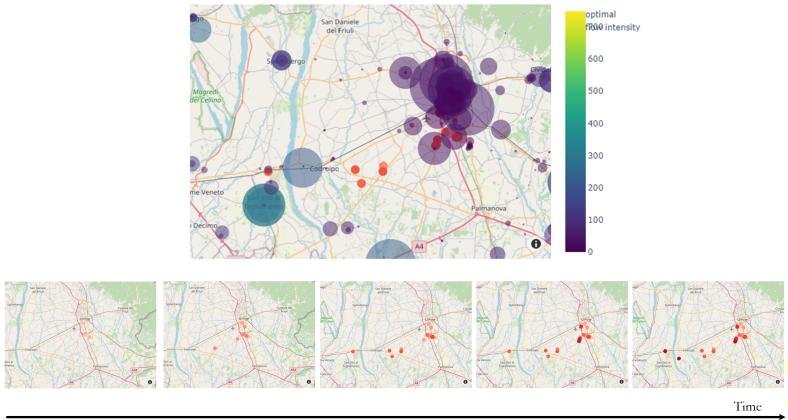


Figure 22.10: Transition of the optimal location of the network considering different time periods.

was developed [22]. The DSS simulates the production operations by using a Montecarlo approach. Decision-makers can change the value of the decision levers regarding:

- the auxiliary energy system to use (electricity, gas or steam);
- the type of asset to use;
- the number of assets for each type;
- the position of the control point where resources are placed.

The relational database illustrated in section 20.2.1 is the input dataset to feed the DSS. The DSS produces descriptive and visual analytics on the results of the simulation as, for example, the spaghetti chart (see Figure 22.11), representing the intensity of the material flow exchanged between control points.

Figure 22.12 proposes insights about the effect of decision-making on the workload of the auxiliary systems for the production of energy and the workload of a single workstation (i.e. a machine). The two analytics look similar as they represent the expected utilization of the loading capacity of a generic working station and power load required by the plant (i.e., the electrical or thermal energy expressed in equivalent kWh of energy or gas) or gas loads over the timeline, respectively. The blue lines represent the utilization coefficient of a resource. This coefficient is calculated as the ratio between the volume of a product loaded on the resource to its loading capacity. A value of utilization less than unity indicates when a working

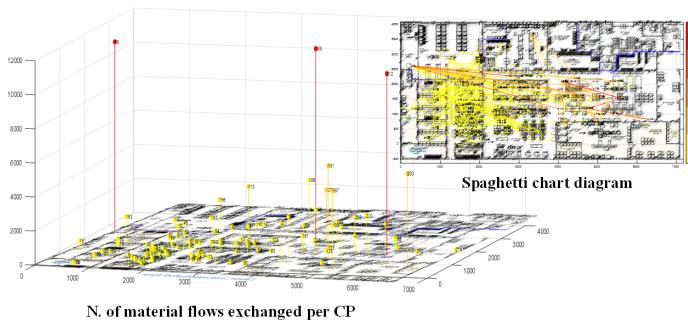


Figure 22.11: Spaghetti chart and material flow intensity chart.

station is underutilized, while values above unity indicate the theoretical number of resources needed to avoid queues.

The energy workload chart identifies the power load for different energy types, estimating the size and capacity of the power supply system. When different energy supplies are allowed for a given resource (i.e., trivalent oven), two alternative graphs are reported, and energy costs are quantified accordingly to assess the most convenient energy source.

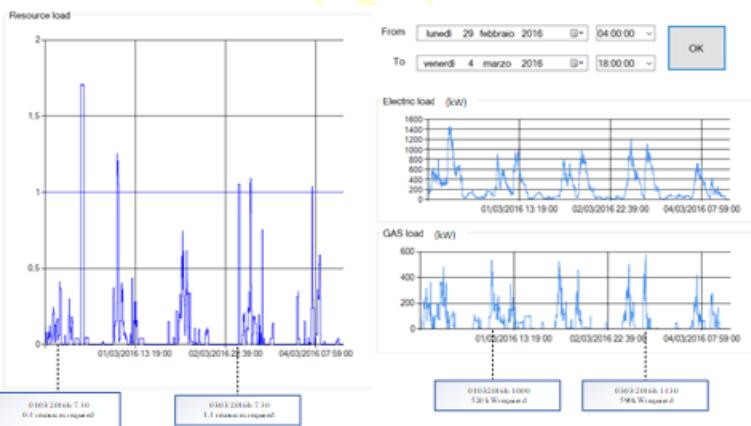


Figure 22.12: Representation of the workload and the energy profile of a set of different resources of the CEKI.

To investigate the behaviour of the system depending on the number of assets, the DSS implements three different rationales to identify the number of assets [23]. The simulation can be set up using:

- a single machine for each type (machine-based rationale), to identify the workload curve of the machine;
- the minimum number of machine to avoid queues (recipe-based), to identify the behaviour of the system without queues;
- an arbitrary number of machine defined by the decision-maker (layout-based), to identify the performance of a given design alternative.

By using the DSS, managers can immediately investigate the logistic effect of their decisions. Additional information on the expected production time and costs are provided to identify which product may represent a bottleneck for the production system (see Figure 22.13).



Figure 22.13: Dashboard identifying the cost and the schedule of each part processed by the CEKI.

The DSS allows them to develop many different production scenarios and to test their behaviour by using a simulation approach. This approach presents a high bias since it cannot be generalised to any production nodes. However, it gives the flexibility to the decision-makers of investigating how the decisions belonging to their specific field of responsibility (e.g. architectural, power plants, layout design, asset choice) modify the operational performance of the entire production system.

### 22.7.2 Plant Design of a 3PL packaging plant for automotive spare parts

This section illustrates the methodology to design a 3PL packaging plant for automotive spare parts from scratch.

The new plant is designed to replace an existing plant processing end-of-line tasks (e.g. picking, packing, placing, oiling, labelling) on more than 58.000 spare parts automotive supply chain. This type of processing plants works as an intermediate stage of the automotive supply chain where incoming products are collected, packaged and labelled according to the clients' needs. The clients are production plants where cars or tractors are assembled and prepared for shipping to the final user. Since these clients mainly work Just-In-Time (JIT), the 3PL packaging plant has to absorb an unpredictable demand in a very short time.

The plant delivers its service using three different due times (24, 48 and 72 hours) with an LoS of the 90% (i.e. 90% of the probability to deliver a service within the given LoS). The operations of the 3PL packaging plant consist of oiling, packing and labelling spare parts.

To properly design the new plant, it is necessary to assess the current state of the system and predict future demand as illustrated in [21.4.2](#).

### Clustering products into families

The first step for the design of the new production node is to define product families [\[4\]](#). As Figure 1 shows, the workload is highly variable. In addition, the quantity processed is variable too and slightly correlated with the number of lines processed. For these reasons, the definition of product families can help the design and control of the production systems. Since the production cycle is not available in the information system of the company, the idea is to cluster parts based on their feature, assuming the production cycle is similar. A correlation heatmap is built, considering the service type, the size, volume, weight and type of package of the items, to check this assumption. Figure [22.14](#) shows a heatmap built on about two millions of orders for seven years.

Figure [22.14](#) reveals an obvious significant correlation between the dimensions of the package and the items. Besides, there are significant correlations between the dimensions of packages and items and some service type. This result suggests that there is the possibility to cluster item based on the service type and assign them to specific workbenches in order to reduce the complexity and the inventory of packages needed on each workbench.

Operators perform the tasks of a specific service type on manual workbenches with no automation. All the operators on the 12 workbenches can process any of the 58.000 products. This fact leads to a very low specialisation of the operators, and unpredictable material flows since any of the workbenches can request all the 1500 different types of packages. Besides, some clients require a customised tertiary package structured as a shelf of the dimension of a pallet. These shelves are placed directly to a workstation of the client's assembly line. For this reason, the 3PL package plant has to deal with high work-in-process (WIP) levels on the workbenches due to

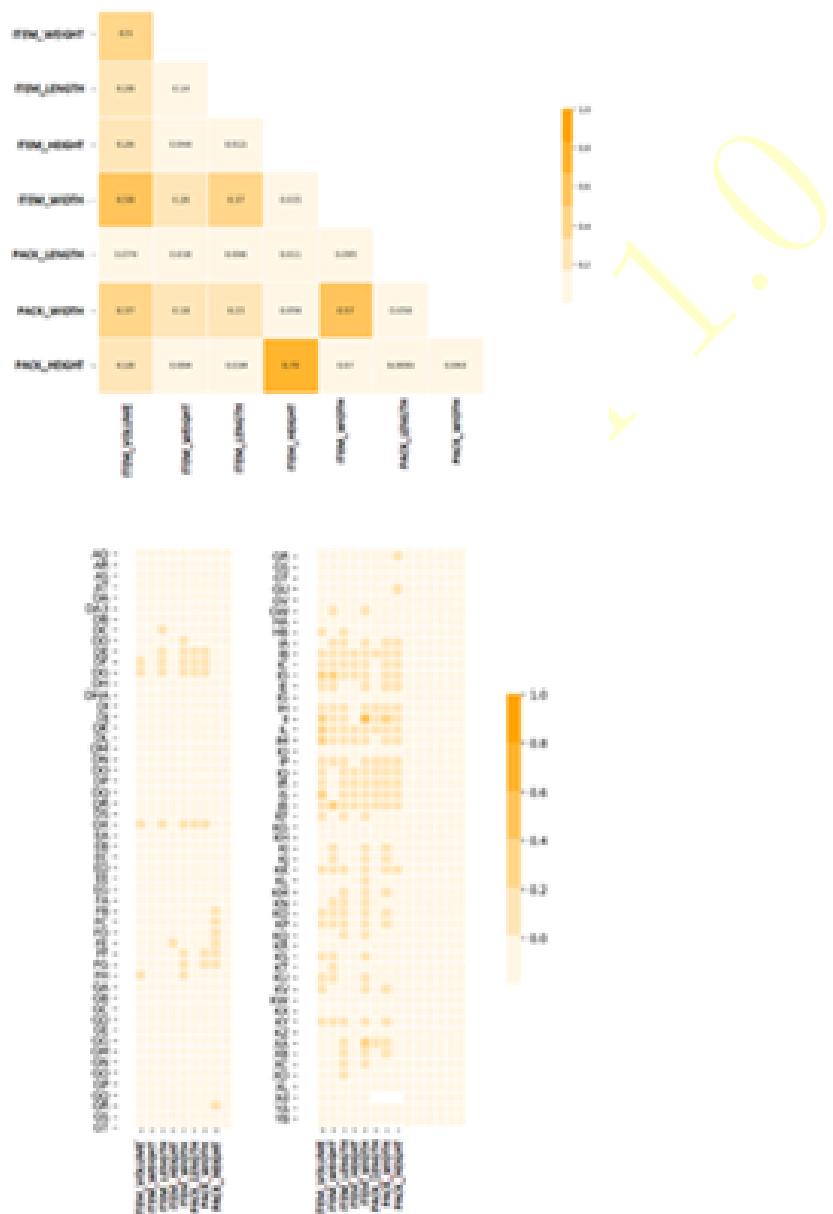


Figure 22.14: Correlation matrix of the features of the input dataset.

products, packages (with the size of a carton box) and customised shelves (with the size of a pallet). To deal with this randomised material flows and WIP of the 3PL plants, we applied the proposed methodology aiming at the definition of a number of families. We start with an increasing number of cluster (from two to ten) using different clustering techniques (see section 8.2). This value is, then, compared to an estimation of the real number of workbenches with the application of the capacitated clustering algorithm. Figure 22.15 presents the graphical results of the algorithms in the different clustering scenarios and with a different number of clusters. Each dot is one of the 58.000 products, while the axis of each subplot represents the first two principal components of the input dataset. Different colours indicate different product families.

The output of unsupervised clustering is compared with the capacitated clustering algorithm (see algorithm 11). The capacitated clustering algorithm considers a maximum allowable capacity that is fixed and equal for all the cluster and an amount of demand required by each product. To feed the algorithm with this data, we set a time and motion monitoring campaign in order to identify an average processing time required by each product. This data collection applied on a subset of the products (i.e. the items belonging to the 95° percentile of the total number of processed lines) due to the very high number of items. The amount of time required by the products with the highest workload defines the maximum capacity for each cluster. The capacitated clustering produces 20 clusters. This number is used to compare the performance with the best performing unsupervised models: the Gaussian Mixture Model and the Complete Linkage Clustering based on the Descriptions, setting the number of clusters  $k = 20$ .

Figure 22.16 illustrates the outcome of this comparison using a visual analytics technique called t-SNE. This technique visually identifies clusters based on the matrix  $X$ ,  $n \times p$  of the observation that is projected onto a 2-dimensional space preserving the proximity of each observation according to the t-distribution. The colours are associated accordingly with the cluster assignment given by the algorithms. Figure 22.16 shows that it is difficult to identify a topology of the cluster (as it happens in Figure 22.15) since the number of clusters is high and the input data are scattered. On the other side, analysing Table 22.1 it is possible to evaluate the performance of the algorithm from a logistic point of view, identifying the variability of the processes organised according to this clustering.

Table 22.1 illustrates the KPIs and compares their variance (using absolute and relative value compared to the capacitated case) calculated on a time horizon of 7 years. It is easy to check that the capacitated clustering provides the highest balanced scenario with the lowest variance. The variance in workload (i.e. seconds) between the 20 clusters has an average of 180 hours per year per workbenches. This is a low gap, considering that the variability in the number of products and packages is dramatically reduced

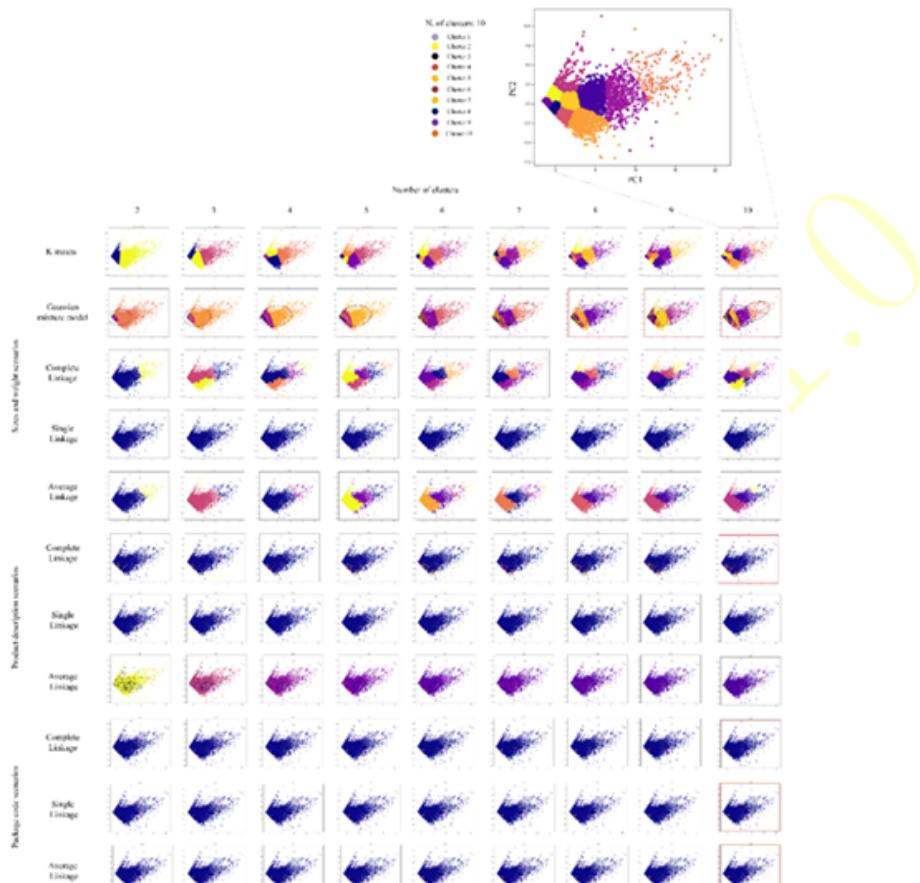


Figure 22.15: Clustering of parts using different clustering algorithms, and different number of clusters.

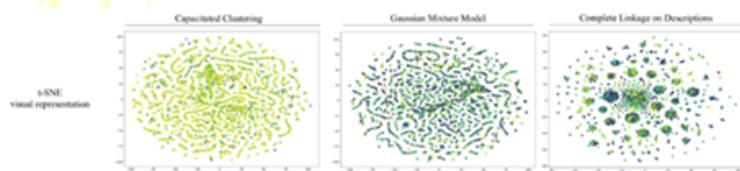


Figure 22.16: Comparison of the topology of the cluster using the t-SNE algorithm. Colors identifies parts belonging to the same cluster.

compared to the other scenarios.

Algorithms	N. of orders (std)	N. of products (std)	Processed quantity (std)	N. of packages (std)	Total working time (std)
Capacitated	5665.3	299.1	700459.9	299.1	4330213.5
GMM	23120.0	1184.3	3067547.0	1184.3	25706129.7
Descr. Compl. Link	21284.7	1184.3	5285096.9	1154.3	40495702.6
<b>Percentage comparison</b>					
Capacitated	100%	100%	100%	100%	100%
GMM	43%	39%	43%	39%	59%
Descr. Compl. Link	37%	39%	75%	38%	93%

Table 22.1: Comparison of the logistic impact between capacitated and uncapacitated algorithms.

Gaussian Mixture Model provides a poorer result that has to be manually checked and assessed before a physical implementation since a couple of clusters results extremely small in workload compared to the average of the others. Nevertheless, it is important to remember that GMM provides the uncapacitated result in short running time (i.e. about 5 minutes) compared to a long runtime of the capacitated algorithm which needs around 20 hours of runtime on a computer equipped with 8Gb memory and a 2.7GHz processor.

### Facility location

Facility location has been studied by using a probabilistic approach, identifying for each year of the input dataset the optimal location in three different scenarios. The scenarios are defined by the distance metrics, i.e. euclidean, rectangular and squared euclidean distance. Figure 22.17 identifies the results on the map where each row is a different distance scenario, and each column identifies how the optimal location – according to the chosen distance – moves on the map. The bubbles identify the intensity of the yearly aggregated flows exchanged with suppliers and customers.<sup>5</sup>

<sup>5</sup>The source code of Figure 22.17 is available [here](#).

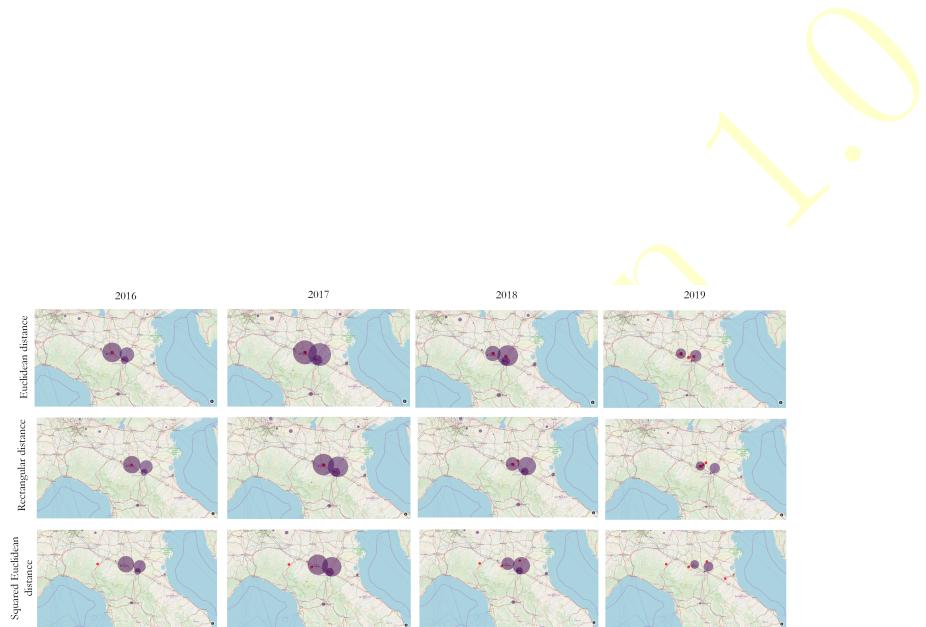


Figure 22.17: Comparison of the optimal location produced using different distance metric (i.e. euclidean, rectangular, and gravity) and different time periods.

## Bibliography

- [1] S. S. Heragu, *Facilities Design, Third Edition*. Crc Press, 2008.
- [2] H. M. Chan and D. A. Milner, “Direct clustering algorithm for group formation in cellular manufacture,” *Journal of Manufacturing Systems*, vol. 1, no. 1, pp. 65–75, 1982.
- [3] J. R. King, “Machine-component grouping in production flow analysis: An approach using a rank order clustering algorithm,” *International Journal of Production Research*, vol. 18, no. 2, pp. 213–232, 1980.
- [4] A. Tufano, R. Accorsi, and R. Manzini, “Machine learning methods to improve the operations of 3PL logistics,” *Procedia Manufacturing*, vol. 42, no. 2019, pp. 62–69, 2020.
- [5] J. F. Heltshe, “Jackknife Estimate of the Matching Coefficient of Similarity,” *Biometrics*, vol. 44, no. 2, pp. 447–460, 1988.
- [6] D. A. Jackson, K. M. Somers, and H. H. Harvey, “Similarity coefficients: measures of co-occurrence and association or simply measures of occurrence?”, *American Naturalist*, 1989.
- [7] M. W. Buser, “Similarity of binary data,” *Systematic Zoology*, 1976.
- [8] Y. Yin and K. Yasuda, “Similarity coefficient methods applied to the cell formation problem: A comparative investigation,” *Computers and Industrial Engineering*, vol. 48, no. 3, pp. 471–489, 2005.
- [9] J. P. Snyder, “Space Oblique Mercator Projection,” *Photogrammetric Engineering and Remote Sensing*, vol. 44, no. 5, pp. 585–596, 1978.
- [10] G. O. Wesolowsky, “Rectangular Distance Location under the Minimax Optimality Criterion,” *Transportation Science*, vol. 6, no. 2, pp. 103–113, 1972.
- [11] S. Salhi and E. Drezner, “Facility Location: A Survey of Applications and Methods.,” *The Journal of the Operational Research Society*, 1996.
- [12] G. Colombo, *Manuale del'Ingegnere*. 2012.
- [13] M. P. Groover, *Automation, Production Systems and Computer-Integrated Manufacturing*. 2015.
- [14] A. Pareschi, *Impianti Industriali. Criteri di scelta, Progettazione, Realizzazione*. 2007.
- [15] M. J. Rosenblatt, “The facilities layout problem: A multi-goal approach,” *International Journal of Production Research*, 1979.

- [16] K. Adendorff, "Layout planning by computer simulation," *AIEE Transactions*, 1972.
- [17] F. L. Plotnick, "RDM - Relationship diagraming method," in *28th Annual National Conference of the American Society for Engineering Management 2007 - Innovation Management: Innovation in a Flattened World, ASEM 2007*, 2007.
- [18] J.-Y. Potvin, G. Lapalme, and J.-M. Rousseau, "A Generalized K-Opt Exchange Procedure For The MTSP," *INFOR: Information Systems and Operational Research*, 1989.
- [19] M. Scriabin and R. C. Vergin, "CLUSTER-ANALYTIC APPROACH TO FACILITY LAYOUT.,," *Management Science*, 1985.
- [20] P. G. Williams, "Vitamin retention in cook/chill and cook/hot-hold hospital foodservices," *Journal of the American Dietetic Association*, vol. 96, no. 5, pp. 490–498, 1996.
- [21] S. M. Rahman, *Handbook of Food Preservation*, vol. 35. 2002.
- [22] A. Tufano, R. Accorsi, F. Garbellini, and R. Manzini, "Plant design and control in food service industry. A multi-disciplinary decision-support system," *Computers in Industry*, vol. 103, pp. 72–85, 2018.
- [23] A. Tufano, R. Accorsi, G. Baruffaldi, and R. Manzini, "Design-support methodologies for job-shop production system in the food industry," in *Sustainable Food Supply Chains*, pp. 115–128, 2019.

# 23

## Process Design

Process design is a strategic activity aiming at choosing how a production system should behave in production. In particular, all these activities have a strong impact on the inventory level (i.e. the WIP) that, in general, must be kept under control to avoid useless costs, as storage costs, damages costs, cost of searching for materials, cost of the unalignment between physical and information inventory. This chapter focuses on:

1. the design of the inventory policy of a production system;
2. the design of the material handling system;
3. the design of the workstations.

### 23.1 Inventory policy design (P7)

Storing goods has a cost. Any supply chain manual, despite its focus, prescribe the *zero inventory* philosophy [1, 2]. Unfortunately, storage levels will never reach zero due to the complexity of the supply chains, their global extensiveness and the fluctuations of the market demand. Hopefully, the right tools can help to set the inventory at an adequate level and to keep it under control.

#### 23.1.1 Model-driven methods (PS4)

The problem of the inventory design involves the definition of the storage levels for raw materials, semifinished and finished products within the production plant. This problem is addressable by using prescriptive methods which defined the right storage level and the proper rules to replenish the

storage (e.g. a re-order point), aiming at reducing storage costs. Storage cost embeds many cost items that are often difficult to estimate. Let assume  $h_i = f(q, t)$  be a comprehensive storage cost function, defined as a function of the storage quantity  $q$ , and the storage time  $t$ . The problem is to find a policy to minimise the cost  $h$ , such that:

- the inventory level is under statistical control;
- the shelflife of items complies with regulations and customers' expectations;
- stockouts are minimised (i.e. the market demand is always satisfied).

The type of market demand suggests different prescriptive approaches to the design of the inventory levels. First of all, a key metric to consider is the lead time  $LT_i$  necessary to obtain part  $i$  from the previous stage of the supply chain (e.g. a supplier or the previous workstation). When  $LT_i$  is larger than the amount of time the following node of the supply chain (e.g. a client) is willing to wait, inventory management should be based on predictions. Otherwise, inventory management is deterministic since stock levels can be maintained low by acquiring parts only when needed. Figure 23.1 illustrate a model identifying different inventory policies depending on the position of the decoupling point. The decoupling point separates the part of the supply chain where production is organised based on predictions from the part where it is based on the orders. When customers' willingness to wait is null (e.g. supermarket) or significantly large (e.g. luxury goods as a yacht), the model in Figure 23.1 can be stretched on the right (everything is based on the forecasts) or on the left (everything is based on market demand).

The demand pattern and the lead time are two typical drivers to choose between stock models and market demand models. There are other drivers as:

- the value of the part;
- the shared use of the part between many processed (e.g. oil or screws);
- the responsiveness and the lead time of the supply chain;
- the complexity of the supply chain.

In general, stable and erratic parts both have a high frequency of demand. In this case, inventory management can be lead by market demand. The inventory level is defined by making accurate predictions on the market demand (see section 21.2). These predictions can be used to feed two inventory management paradigms:

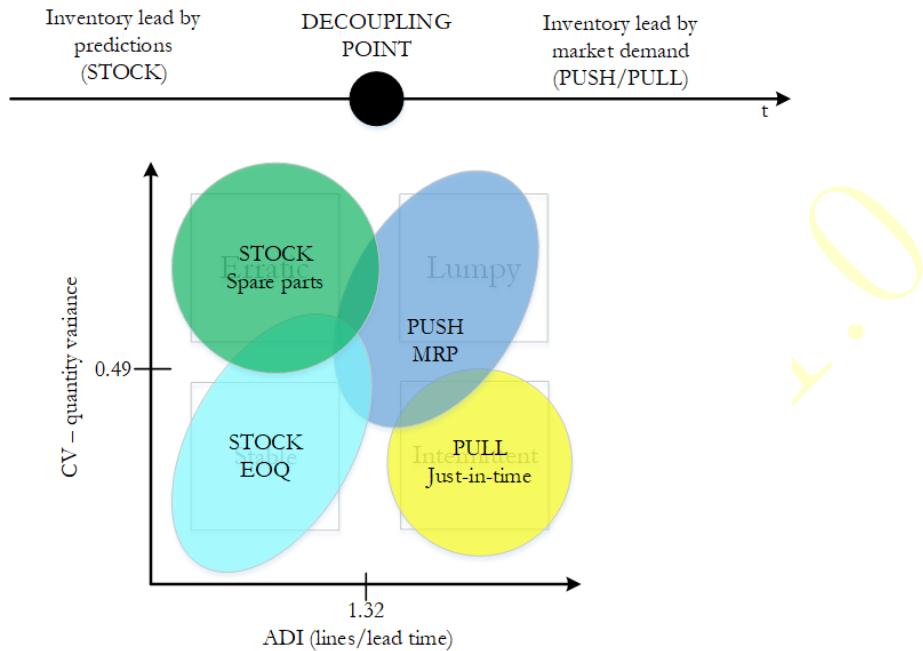


Figure 23.1: Inventory management policies matched with demand patterns.

- the push paradigm;
- the pull paradigm (or Just-in-time JIT).

On the other side, when predictions are unreliable, the inventory level should be oversized to meet unexpected demand and avoid stockouts. This is the case of:

- lot sizing (economic order quantity and safety stock);
- spare parts management (e.g. for maintenance reasons).

### The push paradigm

The push paradigm works by pushing finished products into the market. It can be summarised into the statement: *if you can produce a part i, then produce i*. In other words, when production capacity and raw materials are available, production processes should go on.

Under these assumptions, inventory levels should always be enough to feed the production. A standard method to feed production is the Material Requirements Planning (well known as MRP). The MRP identify the order

quantities for each part  $i$  and for each time period  $t$  (time is usually sampled into weeks). This method does not keep the inventory level under control. It only enables the production to satisfy the demand. The method works by considering:

- an estimate of the average demand  $d_{i,t}$  of part  $i$ , for each time period  $t$ ;
- the quantities of raw materials  $\rho$ , semifinished  $\phi$  and subassemblies  $\sigma$  necessary to produce a part  $i$ ;
- the production lead time  $LT_i$ ;
- the supply lead times  $LT_\rho$ ,  $LT_\phi$ ,  $LT_\sigma$ ;
- the current inventory levels  $WIP_{i,t}$ ,  $WIP_{\rho,t}$ ,  $WIP_{\phi,t}$ ,  $WIP_{\sigma,t}$ .

The order quantity for a part  $i$  at time  $t$  is defined as the difference between the demand  $d_{i,t}$  and the inventory level  $WIP_{i,t}$ . This quantity is, then, corrected to consider the supply and production lead times. Theoretically, the inventory level is out-of-control, but it is kept at a minimum to satisfy the market demand always. In practice, MRP systems are fed with inaccurate predictions and wrong lead times, leading to substantial inventory levels.

### The pull paradigm

The pull paradigm has the sole objective of keeping the inventory level under control. The inventory level of a part  $i$  is set to:

$$WIP_i = d_i \times (1 + SS) \quad (23.1)$$

Where  $SS$  is the percentage of safety stock and  $d_i$  is the average market demand. When setting a pull system, it is essential part  $i$  has a stable demand pattern; otherwise,  $SS$  will be high.  $SS$  depends on the ratio between the supply and the production rate of the part. If storage is checked once a day and an additional day is necessary to feed the production,  $SS$  should be greater than 2 (given that  $d_i$  is expressed parts/days).

Usually, a kanban system is used to implement the pull paradigm. A kanban is a label associated with a production order of a given quantity that can be placed within a container with a fixed capacity  $q$ . The problem is to set the number of kanban for each workstation. This can be calculated as  $K = \left\lceil \frac{WIP_i}{q} \right\rceil$ .

### Lot sizing & safety stock

Lot sizing aims at the definition of the optimal quantity  $Q_i$  for an order to run:

1. when a re-order point  $WIP_i^r$  is reached;
2. at uniform time intervals (e.g. each week).

Given that this inventory policy makes extensive use of the stock, two costs are considered:

- the storage cost;
- the cost to send an order.

The cost model (EOQ-buy) is defined as follows.

$$C_i^{BUY}(Q) = C'_i \left( \frac{Y_i}{Q} \right) + \frac{hQ}{2} \quad (23.2)$$

Where  $C_i^{BUY}$  is the cost of setting the size of a lot to  $Q$ .  $C'_i$  is the cost to run an order of part  $i$ ,  $Y_i$  is the total expected demand (e.g. parts/year) while  $h$  is the storage cost per part. The model assumes a steady demand rate; for this reason, the storage cost is estimated at  $\frac{h}{2}$ . By setting  $\frac{\partial C_i^{BUY}(Q)}{\partial D} = 0$ , then  $Q^* = \sqrt{\frac{2C'_i Y}{h}}$ .

A similar model can be used to set the size of production lot by taking into account:

- the storage cost;
- the cost of the setup of the machine to process part  $i$ .

The cost model (EOQ-make) is defined as follows.

$$C_i^{MAKE}(Q) = C''_i \left( \frac{Y_i}{Q} \right) + \frac{hQ'}{2} \quad (23.3)$$

Where  $C_i^{MAKE}$  is the cost of setting the size of a lot to  $Q$ .  $C''_i$  is the cost of the setup of a workstation to process parts  $i$ ,  $Y_i$  is the total expected demand (e.g. parts/year),  $h$  is the storage cost per part,  $\frac{Q'}{2}$  is an estimate of the average inventory level of parts  $i$ , and  $Q' = Q \left( \frac{X-Y}{X} \right)$ , where  $X$  is the production quantity. By setting  $\frac{\partial C_i^{BUY}(Q)}{\partial D} = 0$ , then  $Q^* = \sqrt{\frac{2C''_i Y}{h}} \times \sqrt{\frac{X}{X-Y}}$ .

In general, when the order quantity  $Q$  is set, orders can be sent when a re-order point is reached,  $WIP_i^r = Y \times LT_i$ , where  $LT_i$  is the supply lead time for part  $i$ . Otherwise, in many practical applications, it results

simpler to send an order at regular time intervals  $RI$  (e.g. every week or every month):  $WIP_i^r = Y \times (RI + LT_i)$ .

In practice, the storage cost of  $WIP_i^r$  is lower than the cost of a stockout. For this reason, the storage level  $WIP_i^r$  is increased by a safety stock  $SS_i$ . The level of service (i.e. the probability of having a part  $i$  available to serve an order) defines the value of  $SS_i$ :  $\text{Prob}\{WIP_i^r + SS_i \geq d_i\} \geq SL_i$ .

### Spare parts inventory management

When dealing with spare parts (lumpy and intermittent parts) the cost minimisation criterion is appropriate as well in the definition of the inventory level for each part. A cost model can be defined by considering the sum of the storage cost and the stockout cost for a part  $i$ . Let  $B_i$  the number of parts required within the time interval considered and  $n$  the maximum number of parts that can be stored.

$$\begin{aligned} C_i^\Delta(Q) = & h \sum_{q=0}^Q q \times \text{Prob}\{B_i = Q - q\} + \\ & + C_m d \left( \sum_{q=Q}^n \text{Prob}\{B_i = Q - q\} \right) \end{aligned} \quad (23.4)$$

Where  $h$  is the storage cost in  $\frac{\text{e}}{\text{part}}$ ,  $C_m$  is the cost of the stockout of a part,  $d$  is the average absorption rate  $\frac{\text{parts}}{\tau}$ , and  $\tau$  is the reference time interval. The  $\text{Prob}\{B_i = Q - q\}$  indicates the probability of consumption of a number of parts  $B_i$  within a time interval  $\tau$ . An appropriate probability distribution can estimate this value. For example, the Poisson distribution introduced in section 6.3.4. By setting  $\frac{\partial C_i^\Delta(Q)}{\partial Q} = 0$  the amount of parts  $Q$  corresponding to the minimum cost can be found.

#### 23.1.2 Data-driven methods

The definition of the optimal inventory level  $Q_i$  of a part  $i$  is deeply linked with variables falling out of the control of the production node, as the supply lead time, the demand variability, the service level, the reliability of supply and customers and many other apparently unpredictable variables as global trends, diseases, war and people subjective perceptions.

It clearly appears that there is no model able to identify and investigate all these parameters producing a reliable  $Q_i$ . Big data and the platform economy can help to approach this problem from a different perspective.

When supply chain players are willing to share their data using the same logistic platform (see section 15.1.2), the amount of available information is larger, faster and leads to more accurate predictions of the market demand. Under these conditions, the players have robust information at their disposal to set the values of  $Q_i$ .

## 23.2 Handling design (P3)

The handling system allows exchanging material flows between the control points of a production node. The choice of the handling system must be coherent with the chosen inventory policy since it affects the level of WIP of the storage node. Handling systems are characterised using two parameters:

- the capacity; i.e. the amount of good that can be transported simultaneously;
- the throughput; i.e. the number of goods transported in the unit of time (e.g. parts per hour).

The choice of the handling technology is similar to the choice of the processing technology, and the same classification of 22.4 is used. Figure 23.2 classified handling systems depending on their level of flexibility and automation.

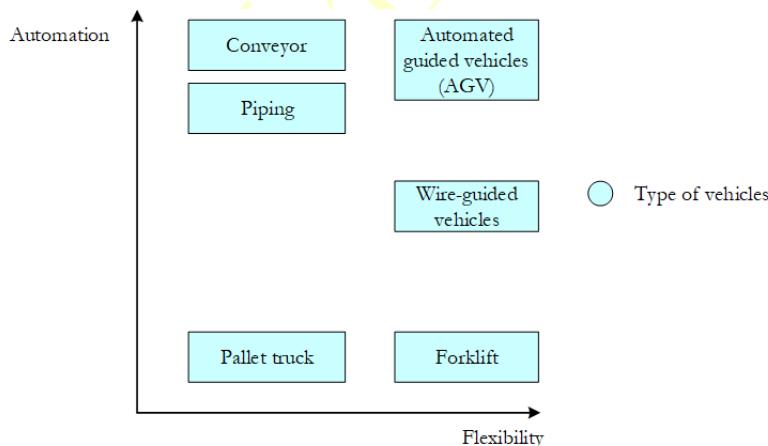


Figure 23.2: Flexibility-automation matrix for handling system choice.

Figure 23.2 identifies the most common handling solutions and their position in the classification scheme. The solutions with a high degree of automation help to keep a smooth flow of the production, but they have a

fixed throughput that deeply depends on the fleet size or the space available on the plant layout. A conveyor between two working station is a good strategy to fix the amount of WIP to the length of the conveyor; nevertheless, this is possible only if the layout has enough space to hold the conveyor. On the other side, forklift and pallet truck are flexible and adaptable since they only need operators to work. However, their flow is less controllable and more likely to errors and avoidable material flows.

### Model-driven methods (PS3)

The models to define the handling solution are essentially prescriptive. In many plants, the constraints are so many that the choice of the handling system barely has more than a single alternative.

The most common strategy is to define a matrix  $F$  with entries assessing the flow of materials  $f_{jk}$  exchanged between two workstations  $j$ , and  $k$ . Given the graph  $G$  with the arcs connecting the workstations,  $G$  defines distances  $d_{jk}$  with travel time  $t_{jk}$ . The capacity  $C_v$  of a vehicle can be used to define the number of trips  $tr_{jk} = \left\lceil \frac{f_{jk}}{C_v} \right\rceil \left[ \frac{\text{trips}}{\text{unit of time}} \right]$ . The availability of the vehicle a measured in units of times can be used to calculate the minimum fleet size:

$$N = \frac{tr_{jk} \times t_{jk}}{a} \quad (23.5)$$

$N$  is the minimum number since it does not consider all the trips when the vehicle is empty, and it does not consider the dynamics of the system. In addition, it cannot be used for rigid systems as conveyors and pipelines where the dimension of the fleet is represented by the length of the conveyor (the problem is similar to the definition of the optimal inventory level  $Q_i$  between workstations  $j$ , and  $k$ ).

For these reasons, discrete events simulation is preferred to assess handling systems from a logistic point of view. All the aforementioned variable are transformed into random variables by considering their probability distribution and the load of the system is simulated instance per instance assessing:

- the average saturation of each vehicle, and
- the queues where parts wait to be loaded.

This information allows deciding if a solution is suitable from a logistic point of view (i.e. if the throughout of the fleet is appropriate). The final decision must consider multiple alternatives evaluating not only the throughput but also the cost of the handling system. Investment costs are the fixed costs for the equipment. Variable costs usually depend on the level of automation:

- both the cost of direct labour and consumables (e.g. the energy) are significant for manual handling systems;
- the cost of energy is significant for automated solutions.

The logically feasible solution with the best economic compromise should be chosen as the handling system.

## 23.3 Workstation design (P4)

The design of the workplace is often considered a minor activity from a supply chain perspective since it is highly peculiar, and guidelines are defined in specific norms which differs depending on the type of work performed. Nevertheless, this stage results relevant from an efficiency perspective since the performance of the operator is enhanced by a well-designed working place.

This activity is mainly prescriptive and sometimes related to ergonomics regulations compliance.

### 23.3.1 Model-driven methods (PS4)

Here we introduce a method to assess the performance of different workstation setting and support the choice between the alternatives [3]. Two alternatives are usually a business-as-usual (AS-IS) scenario and a TO-BE scenario where the configuration of the workstation changes. The model is valid as well while considering different TO-BE alternatives to be compared. The model identifies three groups of performance indicators:

- process parameters, aiming at the evaluation of the efficiency of the workstation;
- ergonomic parameters, aiming at the evaluation of the ergonomic workload on the operators of the workstation;
- economic parameters, aiming at the evaluation of the fixed and variable costs of the workstation.

Figure 23.3 identifies these performance parameters. In addition, since many of these choices involves the use of automation of collaborative automation on a workbench, the functional units of the automated scenario are identified. It is important to remember that to perform this type of improvement, together with the economic and technical feasibility, it is necessary to find a high managerial commitment since operators are usually not willing to change the way they are learnt how to work.

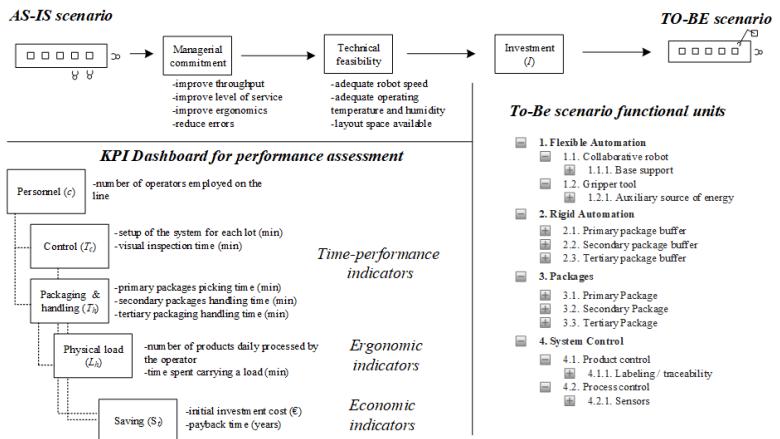


Figure 23.3: A prescriptive model for workbench alternatives comparison.

The main drivers to lead the choice are identified by the performance parameter. If a workstation is able to perform faster, the savings on the hours of direct labour could be enough to support the initial investment. On the other side, when the physical load is significantly reduced, such that:

- operators are more precise enhancing the quality of the finished product, or
- operators have a lower risk of ergonomic disease.

The initial investment may result supported as well. The support and the point of view of the operator are fundamental in this kind of choices to assess whether a workbench configuration results adequate or not.

## 23.4 Applications

### 23.4.1 Process Design of a food catering plant

This section analyses the study and redesign of some production and packaging processes in a food catering facility.

#### Inventory policy design

The design of the inventory policy of a food catering facility results deeply linked with the degree of safety of the finished products. While the raw materials storage system follows a rigid first-expiring first-out (FEFO) policy, the problem appears more complex for semifinished and finished products.

In particular, a task of the production process becomes critical if attached to a critical control point, according to the HACCP protocol. Since the temperature of a food product must be outside the so-called danger zone (see section 22.7.1), cook-warm products must be kept over 65 °C after cooking. All the tasks performed after the cooking area of the plant must be separated by hot-holder working as ovens set to a safe temperature (e.g. 85 °C). Figure 23.4 illustrates the result of a monitoring campaign performed on the temperature of the products leaving the cooking area. Each column identifies a product family while rows identify the tasks performed after cooking.

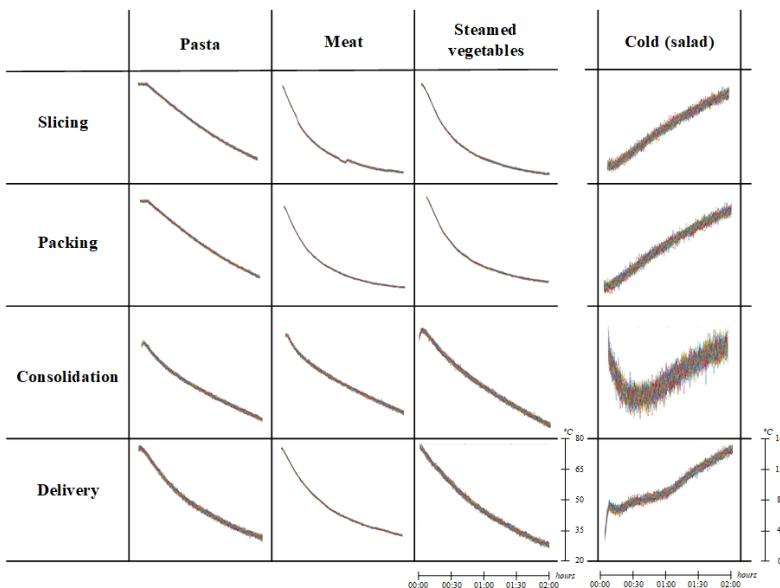


Figure 23.4: Temperature decay profiles of products after the cooking department.

The number of the hot-holders define their throughput  $TH_j$  and the inventory level  $Q_i$  between each working station of the end-of-line of a CEKI. By considering the average processing time for each task, it was possible to estimate the temperature decay and to simulate the effect on different production lot. The output of the simulation identifies the adequate number of hot-holder between workstations and between the consolidation area and the shipping bays (for finished products). A minor investment in hot-holding machine and additional space for the plant layout reduced the errors and increased the quality of the finished product [4].

## Handling design

The design of the handling of a CEKI is static and relies on the high flexibility of manual handling using roll containers. Roll dimensions compatible with blast-chiller and ovens and entire lots can be cooked or refrigerated without additional handling. In addition, the low throughput of the plant does not identify the need for automation of the handling systems.

## Workstation design

The activity on a packing workstation of a CEKI is highly repetitive with many identical moves that may lead to errors (e.g. a product placed in the wrong pack) when the operator is tired. For this reason, the technical and economic feasibility of the implementation of automated collaborative technology in manual handling/production processes is studied [3]. The technical and economic assessment evaluate the placement of a cobot replacing the human operator in charge of loading the plastic boxes containing the finished products.

The cobot has a kinematic that can be easily programmed and simulated; for this reason, it is fast to extrapolate the expectation of the operative time and motions (these values have the superscript  $\alpha$ ). These values have to be compared with the human performance that is measured on-field in the business-as-usual scenario (marked by the superscript  $\beta$ ). The saving per year is calculated as:

$$S_t = \left( (t^{idle} + t_{pick}^\alpha) \times l^\alpha + (t_{pick}^\beta + t_{depot}^\beta) \times \left[ \frac{l^\alpha}{v^{\alpha,\beta}} \right] - t^{trav} \right) \times l \times d \times c - c^{main} \quad (23.6)$$

Where:  $t^{idle}$  is the idle time of the operator waiting for packed meals at the end-of-line (sec);  $t_{pick}^\alpha$  is the time to label a primary package (sec);  $l^\alpha$  is the number of primary packages per production lot;  $t_{pick}^\beta$  is the time to pick an empty secondary package and put it into a filling buffer (sec);  $t_{depot}^\beta$  time to put a full secondary package into a tertiary package (sec);  $v^{\alpha,\beta}$  is the number of primary packages per secondary package;  $t^{trav}$  is the travelling time to supervise production tasks (sec);  $l$  is the number of lots per working shift;  $d$  is the number of working shifts per year;  $c$  is the cost of an operator for a working shift (€);  $c^{main}$  is the maintenance cost of the cobot per year.

Since the value of  $S_t$  increases rapidly year by year, the investment in automation resulted convenient for the company.

## Bibliography

- [1] J. P. Womack, D. T. Jones, and D. Roos, “The machine that changed the world,” *Business Horizons*, 1992.
- [2] D. T. Womack, J. P., & Jones, “Lean Thinking by Womack and Jones,” *Review Literature And Arts Of The Americas*, 2000.
- [3] R. Accorsi, A. Tufano, A. Gallo, F. Galizia, G. Cocchi, M. Ronzoni, A. Abbate, and R. Manzini, “An application of collaborative robots in a food production facility,” in *29th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2019)*,, 2019.
- [4] A. Tufano, R. Accorsi, and R. Manzini, “A Simulated Annealing algorithm for the allocation of production resources in the food catering industry,” *British Food Journal*, 2020.