SAPIENZA
UNIVERSITÀ DI ROMA

Estimation of Distribution Parameters as a Tool for Model-Based System Engineering and Model Identification

Ph.D. Programme in Computer Science – XXXII Cycle

Candidate

Angela Pappagallo
ID number 689176

Thesis Advisor

Prof. Enrico Tronci

Co-Advisor

Prof. Annalisa Massini
Prof. Gaia Maselli

15 September 2020

Estimation of Distribution Parameters as a Tool for Model-Based System Engineering and Model Identification
Ph.D. thesis. Sapienza University of Rome

This thesis has been typeset by LaTeX.

Version: 31 dicembre 2020

Author's email: pappagallo@di.uniroma1.it

To Maya

Abstract

The estimation of the parameters of a probability distribution (*e.g.*, moments) plays an important role both in the model-based system engineering (*e.g.*, analysis and verification through Statistical Model Checking (SMC)) and in the identification of parameters of predictive models (*e.g.*, systems biology, social networks).

The contribution of this PhD thesis is both on the algorithm side and on the modeling side. On the algorithm side, we overview a set of Monte Carlo-based Statistical Model Checking tools and algorithms for the verification of Cyber-Physical Systems, and we provide selection criteria for the verification problem at hand. Furthermore, we present an efficient Monte Carlo based algorithm to estimate the expected value of a multivariate random variable, when marginal density functions are not known. We prove the correctness of our algorithm, we give an Upper Bound and a Lower Bound to its complexity and we present experimental results confirming our evaluations.

On the modeling side, we present a mechanistic and identifiable model to predict, at node level and at set of nodes level, the expected value of the retweeting rate of a message inside a social network, at a certain time. Our model parameters are random variables, whose distribution parameters are estimated from an available dataset. We experimentally show that our model reliably predicts both the qualitative and the quantitative time behavior of retweeting rates. This is confirmed by the high correlation between the predicted and the observed data. These results enable simulation based analysis of users or of set of users behaviors inside a network.

Acknowledgments

I would like to offer my sincere gratitude and thanks to my advisor Professor Enrico Tronci. I would like to thank you for being such a good mentor and for accepting the challenge to support and follow my PhD after many years since my Master Degree.

I would like to thank also Professor Annalisa Massini for all the precious advices she gave me during these years. Thanks also to Professor Toni Mancini for his technical support and to Professor Marzia Anterior for helping me to find the motivations of my research work on Twitter.

I would like to thank also my external reviewers, Professor Axel Legay and Professor Alberto Lluch Lafuente, for their suggestions and comments to a preliminary version of my thesis.

A special thanks to all the McLab guys, Stefano, Vadim, Francesco and Marco, for being always so nice and helpful, especially in the most critical moments. Thanks also to Federico Mari for his continuous encouragement.

Of course, my thanks go to ISTAT, that financed my PhD. But I would like also to express my appreciation to the colleagues, especially Linda, who suggested me to attend the PhD school and to all the colleagues who gave me time to complete my studies.

Finally, my biggest thanks to all my friends and to my family, Daniele and my daughter Maya. Thanks for always helping and supporting me. Nothing would have been possible without them.

Contents

Chapter 1

Introduction

The estimation of the parameters of a probability distribution (*e.g.*, moments) plays an important role both in the model-based system engineering (*e.g.*, analysis and verification through Statistical Model Checking (SMC) [3, 76, 118, 28, 67, 74, 82, 83]) and in the parameters identification of predictive models, for example, systems biology [123, 122, 60, 41, 104, 105] and social networks [79, 44, 49, 138].

In such a setting, this thesis contribution is twofold: on the algorithm side and the modeling side.

On the algorithm side, we overview a set of the Monte Carlo-based SMC tools and algorithms for the verification of Cyber-Physical Systems and we provide selection criteria to select the most suitable tool for the problem at hand. Furthermore, we present an efficient Monte Carlo-based algorithm to compute an approximation of the expected value of a multivariate random variable.

On the modeling side, we present a mechanistic model for the retweeting rate, at the node level and the set of users level, of a message inside a social network (Twitter).

In the following, we give a brief outline of our contributions.

1.1 Statistical Model Checking

We present a review of a set of Monte Carlo-based SMC tools for the verification of Cyber-Physical Systems that, because of their ever-increasing deployment, exacerbates the need of the need for efficient formal verification methods. SMC is a simulation-based technique that uses statistical inference over observations to establish the correctness of the system under verification, within a statistical confidence bound. On the basis of the available SMC techniques and tools, we provide criteria, relying on the kind of system and the property to be verified, to select the most suitable SMC tool for the verification problem at hand.

1.2 Multivariate Monte Carlo

The Monte Carlo-based algorithm, $\mathcal{AA}$, of Dagum et al [30] is mainly used in Statistical Model Checkers that estimate the expected value of a random variable modelling some relevant system properties. Examples are: APMC [74] and APD Analyser [82, 83].

There are many situations in which the property of interest depends on a multivariate random variable. This motivates the investigation of efficient algorithms to estimate the expected value of a multivariate random variable.

We present an efficient (as for the number of samples generated) algorithm, $\mathcal{MAA}$, to compute an (ϵ, δ)-approximation of the expected value of a multivariate random variable when marginal probability density functions are unknown. Our algorithm is built on top of the optimal $\mathcal{AA}$ algorithm in [30] for univariate random variables. We prove the *correctness* of $\mathcal{MAA}$. Then, we give a Lower Bound to the complexity for any (ϵ, δ)-approximation algorithm to estimate the expected value of a multivariate random variable, when marginal probability density functions are unknown. Furthermore, we give an Upper Bound to the *complexity* of our algorithm, that is, to the expected value of the number of samples generated. We experimentally show that there exists an input s.t. the Upper Bound is attained and that, on a suitable input, our $\mathcal{MAA}$ algorithm attains the above mentioned Lower Bound.

1.3 Mechanistic Model for Twitter

We present a mechanistic model for the retweeting rate of a message inside a social network (Twitter), along with a model validation methodology. More specifically, we define a discrete-time dynamical system modeling the message resharing mechanism of Twitter. We model each network user behavior as a stochastic variable and we compute its parameters (mean and variance) from an input dataset. The estimate of these parameters is performed from an available dataset. The so defined user model, together with the above-mentioned discrete-time model, allows predicting the time evolution of the expected value of the retweeting rate of a message, for a user or a set of users of the network. Our model is mechanistic and, as it is built on top of identifiable parameters, is identifiable.

We use a dataset to train our model. We experimentally show that, when a new message is spread into the network, our model reliably predicts the qualitative as well as quantitative time behavior of retweeting rates.

1.4 Summary

This thesis is organized as follows. In Chapter 2 we give a review of the SMC, by focusing on Monte Carlo-based algorithms for Cyber-Physical System (CPS). In Chapter 3 we present our Monte Carlo-based algorithm to estimate the expected value of a multivariate random variable. In Chapter 4 we present our mechanistic model to predict, at the node level, the expected value of the retweeting rate of a message inside Twitter. In Chapter 5 we give some conclusions.

Chapter 2

Statistical Model Checking

In this section, we give a short review of a set of Monte Carlo-based Statistical Model Checking tools and algorithms for the verification of properties over Cyber-Physical Systems. The content of this section is related to the paper [94].

2.1 Introduction

The ever-increasing deployment of autonomous Cyber-Physical Systems (CPSs) [6] (*e.g.*, autonomous cars, Unmanned Autonomous Vehicles) exacerbates the need for efficient formal verification methods [27]. In this setting, the main obstacle to overcome is the huge number of scenarios to be evaluated (*scenario explosion*). Statistical Model Checking (SMC) [75] holds the promise to overcome this obstacle by using statistical methods to sample the set of scenarios.

SMC algorithms use a simulation-based approach and statistical inference over observations to establish the correctness of the System Under Verification (SUV), within statistical confidence bound. Statistical inference can be applied for two different purposes: i) *Hypothesis Testing (HT)*, allowing to infer if the observations provide statistical evidence of the satisfaction of a property defined over the system; ii) *Estimation*, allowing to compute approximated values for some system parameters, whose probability distribution has to be known *a priori.*

Many SMC tools have been developed, suggesting how to carry out simulations and how many simulations to perform. The simulation runs affect the performance of each tool, that significantly varies according to several features. As a consequence, it is hard to know a priori the best tool suited for the verification problem at hand. In this work, we will overview a set of the Monte Carlo based SMC tools, currently available for research purposes, to provide selection criteria based on Key Performance Indicator (KPI) about: the verification activity (*e.g.*, minimize verification time or cost); the environment; the kind of system model; the property specification language; the statistical inference approach and the algorithm that implements it. For example, if the verification KPI is to ease parallelization of the simulation activity, our analysis allows us to identify the tools that compute beforehand the number of simulation runs needed to attain given statistical confidence. On the other hand, if the main verification KPI is the number of simulation runs, our analysis allows us to identify the tools that minimize the number of scenarios to be sampled.

Of course, many SMC reviews are available. For example, Agha *et al.* in [3] give a very extensive survey of most of the tools available to the academic community, giving details about the kind of input models, the logic used to define properties, emphasizing current limitations and trade-offs between precision and scalability. In [101], Reijsbergen *et al.* present a comprehensive overview of different algorithms performing HT (that will be discussed in Section 2.4.1). For each algorithm the work in [101] focuses on its characteristics and performance. In [14], some of the most popular and well-maintained SMC tools are reviewed. The tools are compared based on their modelling and property specification languages, capabilities, and performances, to help users to select the most suitable tool to use.

Summing up, the output of our analysis is the following. First, providing tool selection criteria based on the 5-tuple consisting of: SMC tool, environment model, SUV model (*e.g.*, Discrete Time Markov Chain (DTMC), Continuous Time Markov Chain (CTMC), etc), property specification, statistical inference approach. On the other hand, making a comparison with the above-mentioned reviews: [3] focuses on tool selection criteria based on the pair consisting of SMC tool, SUV application domain (*e.g.*, CPS, biological system, etc); [101] focuses on identifying which SMC tool implements a given verification algorithm; [14] focuses on tool selection criteria based on the triple consisting of SMC tool, SUV model, property definition language.

Second, identifying open research challenges in the field of SMC tools, namely suggesting the deployment of SMC tools for the unbounded verification of hybrid systems with discrete or continuous time.

This section is structured as follows. In Section 2.2, we give a brief introduction to CPSs. In Section 2.3, we give some background information on the models used to represent the system to be verified and on the logics used to define properties. In Section 2.4, we discuss the main differences between the two approaches to statistical inference in SMC, *i.e.*, HT and Estimation. We also give some hints about *Bayesian Analysis* (see, *e.g.*, [139]), although we will not go into details. In Section 2.5, we provide a taxonomy of the tools implementing Monte Carlo based SMC techniques. In Section 2.6 we analyze our taxonomy and discuss open research topics.

2.2 Cyber-Physical Systems

Our verification setting consists of three main components: the environment, the System Under Verification (SUV) and the specifications (see Figure 2.1).

For verification purposes we need to model both the set of operational scenarios (SUV environment) as well as our SUV. This can be done by considering deterministic systems with stochastic inputs (the typical Control Engineering approach) or by considering stochastic systems (the typical Computer Science approach). Considering that from a uniformly distributed random variable in the real interval [0, 1] we can get a random variable with any specific (possibly dependent on the SUV state) distribution (*e.g.*, as in Chapter 2. of [36]) we can easily transform one modelling approach into the other. So, for example, a *Discrete-Time Markov Chain* $\mathcal{M}$ can be modeled with a *Discrete-Time Dynamical System* $\mathcal{S}$ with a uniformly random stochastic input from which we get $\mathcal{M}$ state-dependent transition probability using

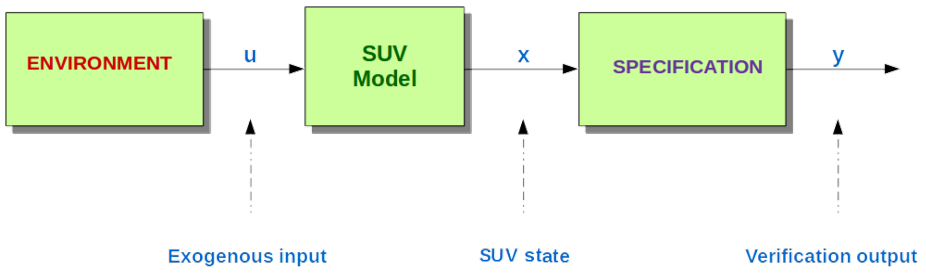


Figure 2.1. Simulation-based verification setting

techniques like the those in [36]. Finally, from a practical point of view, we note that simulators will generate random numbers using a pseudo-random number generator. Thus, both modelling approaches lead to the same implementation. Accordingly, for our purposes, we will consider them equivalent.

Many statistical model checkers work in a *black box* fashion by interfacing with widely used commercial as well as open source simulators for the design and the verification of CPSs in many application domains (*e.g.*, automotive, avionics, IoT). Examples of such simulators are: Simulink [114], Dymola [38], SimulationX [113], Wolfram SystemModeler [127], MWorks [137], Open Modelica [92]. Since the SUV model is not visible to such *black box* statistical model checkers (they will just run the simulator and check its outputs) we will not make any distinction between synchronous or asynchronous systems for them. Of course, the the situation is quite different for *white box* model checkers. They typically address this problem by defining probabilistic or non-deterministic schedulers as done, for example, in PRISM [73] or SMV [86], *e.g.*, following the approach in Chapter 2 of [13]. Using such approach also *white box* model checkers can model both synchronous as well as asynchronous systems.

Since most statistical model checkers work in a *black box* fashion with simulators here we will follow the Control Engineering approach and model our SUV as a deterministic system with stochastic inputs.

2.2.1 Modelling the Environment

The *environment* (Figure 2.1) for the SUV defines the set of *operational scenarios* our SUV is supposed to withstand. In other words, an environment defines the set of *admissible inputs* for our SUV. Such inputs are usually named *exogenous* or *uncontrollable* inputs since they are not under the control of the SUV. Typical examples of uncontrollable inputs are faults, change in system parameters, inputs from users or from other systems.

In this section, we formalize the notion of *space of input functions* for a system. Such a space defines the system *environment*, *i.e.*, the set of *admissible operational scenarios* for our SUV. To this end, first of all, we define the set of time instants, *time set* (Definition 2.1). In the following, as usual, the notation B^A denotes the set of functions from A to B.

Definition 2.1 (Time set)**.** *A* time set *T is a subgroup of $(\mathbb{R}, +)$.*

For example, when considering discrete-time systems T will be the set $\mathbb{Z}$ of all integer numbers whereas for continuous-time systems T will be the set $\mathbb{R}$ of all real numbers.

Definition 2.2 (Space of input functions)**.** *Given a set U (of* input values*) and a time set T, a* space of input functions *$\mathcal{U}$ on (U, T) is a subset of $U^T = \{u \mid u : T \to U\}$*

This allows us to easily model application domain-dependent probability distributions on the SUV input functions as well as stochastic sampling strategies on the SUV input functions.

The above approach is quite convenient since in a simulation-based verification setting the simulation model is typically deterministic and stochastic behaviors stems from uncontrollable events.

Of course, the more liberal the environment model, the larger the set of operational scenarios under which our SUV will be verified. On the other hand, the more liberal the environment model, the heavier the computational load entailed by the verification activity. Accordingly, a trade off is typically sought by verification experts between the *degree of assurance* (*i.e.*, to which extent all meaningful operational scenarios have been considered) and the computational cost (and thus time cost) of the verification activity.

2.2.2 Modelling the SUV

We focus mainly on the case where the System Under Verification (SUV) is a Cyber-Physical System (CPS), *i.e.*, a system consisting of both hardware and software components. CPSs can be found basically in any domain. Examples are: biomedical devices, aerospace (*e.g.*, airplanes, Unmanned Autonomous Vehicle (UAV), satellites), automotive, smart grid, etc. Many CPSs are indeed safety or mission-critical systems that, through software components, are endowed with some degree of *autonomous* behavior. From this stems the need for thorough verification of correctness of the software controlling the CPS. This motivates our focus on formal verification of CPSs.

From a formal point of view, a CPS is basically a dynamical system whose state variables can undergo continuous as well as discrete changes. Typically, continuous dynamics model the physical components of a CPS whereas the discrete dynamics model the software components.

To formalize our notion of system the following definition will be useful.

Definition 2.3 (Restriction)**.** *Let $\mathcal{I}$ be a time interval (i.e., an interval $\mathcal{I} \subseteq T$). Given a function $u \in U^{\mathcal{I}}$ (see Definition 2.2) and two positive real numbers $t_1 \le t_2$, we denote with $u\mid_{[t_1,t_2)}$ the restriction of u to the interval $[t_1, t_2)$, i.e. the function $u\mid_{[t_1,t_2)}: [t_1, t_2) \to \mathcal{U}$, such that $u\mid_{[t_1,t_2)}(t) = u(t)$ for all $t \in [t_1, t_2)$. We denote $\mathcal{U}^{[t_1,t_2)}$ the* restriction *of $\mathcal{U}^{\mathcal{I}}$ to the domain $[t_1, t_2)$. That is, $\mathcal{U}^{[t_1,t_2)} = \{u\mid_{[t_1,t_2)} \mid u \in \mathcal{U}^{\mathcal{I}}\}$.*

The above considerations lead us to model (Definition 2.4) our SUV as a dynamical system, along the lines of [117].

Definition 2.4 (Dynamical System)**.** *A* Dynamical System*, $\mathcal{H}$ is a tuple (X, U, Y, T, $\mathcal{U}$, φ, ψ), where:*

- *X, the* space of state values *of $\mathcal{H}$, is a non-empty set whose elements are said* states *of $\mathcal{H}$;*
- *U, the* space of input values *of $\mathcal{H}$, is a non-empty set whose elements are said* input values *for $\mathcal{H}$;*
- *Y, the* space of output values *of $\mathcal{H}$, is a non-empty set whose elements are said* output values *for $\mathcal{H}$;*
- *T is a time set;*
- *$\mathcal{U}$, the* space of input functions *of $\mathcal{H}$, is a non-empty subset of U^T;*
- *$\varphi : T \times T \times X \times \mathcal{U} \to X$, is the* transition map *of $\mathcal{H}$;*
- *$\psi : T \times X \times U \to Y$ is the* observation function *of $\mathcal{H}$.*

Function φ satisfies the following properties:

- Causality*. For all $t_0 \in T$, $t \geq t_0$, $x_0 \in X$, $u, u' \in \mathcal{U}$:*

$$u \mid_{[t_0,t)} = u' \mid_{[t_0,t)} \Rightarrow \varphi(t, t_0, x_0, u \mid_{[t_0,t)}) = \varphi(t, t_0, x_0, u' \mid_{[t_0,t)})$$

- Consistency*. For all $t \in T$, $x_0 \in X$, $u, \in \mathcal{U}$:*

$$\varphi(t, t, x_0, u) = x_0$$

- Semigroup*. For all $t_0 \in T$, $t > t_1 > t_0$, $x_0 \in X$, $u \in \mathcal{U}$:*

$$\varphi(t, t_0, x_0, u \mid_{[t_0,t)}) = \varphi(t, t_1, \varphi(t_1, t_0, x_0, u \mid_{[t_0,t_1)}), u \mid_{[t_1,t)})$$

In the following, unless otherwise stated, $\mathcal{H}$ denotes the tuple (X, U, Y, T, $\mathcal{U}$, φ, ψ).

From a formal point of view, a CPS is just a dynamical system where T is the set of nonnegative real numbers and state variables may undergo continuous (to model the physical part of the system) as well as discrete (to model the software part of the system) changes. This typically means that φ is not continuous (but typically may be assumed continuous *almost everywhere*). Definition 2.4 is general enough to encompass discrete as well as continuous-time systems and finite as well as infinite-state systems.

We refer the reader to Section 1 of [117] for examples of how familiar systems fit Definition 2.4. In particular on how from a *state-based* description of the system dynamics (*e.g.*, through Ordinary Differential Equation (ODE) for continuous-time systems or through recurrence equations for discrete-time systems) we can compute the transition function φ. In this respect we note that typically, the transition function φ is computed by simulation using numerical techniques (*e.g.*, see [25]), since it is seldom available in a closed form (except, *e.g.*, for linear, time-invariant systems).

2.2.3 Modelling the Specifications

A specification (Figure 2.1) defines the requirements our SUV should satisfy for all operational scenarios. In our simulation-based setting, a specification is also defined through a system as outlined below.

Notation 2.1 (Product of input spaces). *Let T be a time set and $\mathcal{U} \subseteq U^T$, $\mathcal{V} \subseteq V^T$ be input spaces. By abuse of language we denote with $\mathcal{U} \times \mathcal{V}$ the set $\{\theta \in (U \times V)^T \mid \exists u \in \mathcal{U}, v \in \mathcal{V}$ s.t. $\theta(t) = (u(t), v(t))\}$.*

A specification takes as input the SUV input (*operational scenario*) and state and returns a real value assessing the extent to which requirements are satisfied. This is formalized in Definition 2.5.

Definition 2.5 (CPS Specification). *Let $\mathcal{H}$ = (X, U, Y, T, $\mathcal{U}$, φ, ψ) be a system. A specification for $\mathcal{H}$ is a system $\mathcal{Q}$ = (Z, $U \times X$, $\mathbb{R}$, T, $\mathcal{U} \times X^T$, μ, θ), where:*

- *Z is the space of state values of the system Q;*
- *$U \times X$ is the space of input values of the system Q;*
- *$\mathbb{R}$ is the space of output values of the system Q;*
- *T is the* time set *of the system Q (and $\mathcal{H}$);*
- *$\mathcal{U} \times X^T$ is the space of input functions of Q;*
- *μ is the transition map of Q;*
- *θ is the observation function of Q.*

Example 2.1 (Example of CPS Specification). *Let $\mathcal{H}$ be the system defined by the ODE*

$$\dot{x} = -3x + u$$

with u constant. Then, $\mathcal{H}$ = (X, U, Y, T, $\mathcal{U}$, φ, ψ) with:

- *$X = U = Y = T = \mathbb{R}$;*
- *$\mathcal{U}$ is the set of constant functions from T to U;*
- *$\varphi(t_0, t, x_0, u) = e^{-3(t-t_0)}(x_0 - \frac{u}{3}) + \frac{u}{3}$;*
- *$\eta(t, \bar{x}, u) = \bar{x}$.*

Of course, in general, the function φ is not available in a closed form and can only be computed through simulation (e.g., using simulators like Simulink, Dymola or Open Modelica).

Suppose that we want to check that the output from $\mathcal{H}$ is always close enough *to $\frac{u}{3}$. This could be assessed by computing the* Root Mean Squared Error *(RMSE) of x with respect to $\frac{u}{3}$. This leads to the specification $\mathcal{Q}$ = (Z, $U \times X$, $\mathbb{R}$, T, $\mathcal{U} \times X^T$, μ, θ), where: $Z = U = X = T = \mathbb{R}$ and the space of input functions of Q consists of pairs of functions (u, x), where u is a constant function and x maps reals to reals.*

Finally: $\mu(t_0, t, z_0, u, x) = z_0 + \int_{t_0}^{t} (x(\tau) - \frac{u}{3})^2 d\tau$ (RMSE) and $\theta(t, \bar{z}, u, x) = \bar{z}$.

To formalize the relationship between a system and its specification we need to define the notion of *monitored system* (Definition 2.6) [98], [119], [16] (in the following, as usual in modern programming languages (*e.g.*, Java, Python), the lambda notation $\lambda t.f(t)$ denotes the function f that, applied to t returns $f(t)$).

Definition 2.6 (Monitored system)**.** *Let $\mathcal{H}$ be a system and $\mathcal{Q}$ be a specification for it. The ($\mathcal{H}$, $\mathcal{Q}$)* monitored system *is the system $\mathcal{M}$ = ($X \times Z$, U, $\mathbb{R}$, T, $\mathcal{U}$, Φ, Ψ) where:*

$$\Phi(t, t_0, (x_0, z_0), u) = (\varphi(t, t_0, x_0, u), \mu(t, t_0, z_0, (u, \lambda t.\varphi(t, t_0, x_0, u))))$$

and

$$\Psi(t, (x, z), v) = \theta(t, z, (v, x))$$

Example 2.2 (Example of monitored system)**.** *Using the systems $\mathcal{H}$ and $\mathcal{Q}$ from Example 2.1 we have that the monitored system ($\mathcal{H}$, $\mathcal{Q}$) is the system $\mathcal{M}$ = ($X \times Z$, U, $\mathbb{R}$, T, $\mathcal{U}$, Φ, Ψ) with:*

- *X, Z, U, T and $\mathcal{U}$ as in Example 2.1;*
- *$\Phi(t, t_0, (x_0, z_0), u) = (e^{-3(t-t_0)}(x_0 - \frac{u}{3}) + \frac{u}{3}$, $z_0 + \int_{t_0}^{t}((e^{-3(\tau-t_0)}(x_0 - \frac{u}{3}) + \frac{u}{3}) - \frac{u}{3})^2 d\tau)$;*
- *$\Psi(t, (\bar{x}, \bar{z}), u) = \bar{z}$*

Thus the output $\Psi(t, \Phi(t, t_0, (x_0, z_0), u), u)$ of the monitored system at time t is $z_0 + \int_{t_0}^{t}((e^{-3(\tau-t_0)}(x_0 - \frac{u}{3}) + \frac{u}{3}) - \frac{u}{3})^2 d\tau$.

The real value returned by Ψ defines our KPI evaluating to which extent the system meets its requirements under the *operational scenario* defined by input function u. If a KPI is Boolean (*i.e.*, it takes only values in $\{0, 1\}$ then we have a *crispy* classification (*i.e.*, the system will *pass* or *fail* to meet the given specification) else we have a *metric* classification measuring the extent requirements are violated. On such a basis our verification problem can be formulated as follows.

Definition 2.7 (Verification problem)**.** *Let $\mathcal{H}$ be a system, $\mathcal{Q}$ be a specification for $\mathcal{H}$ and $t_0 \in T$ a time instant. We say that $\mathcal{H}$* satisfies *its specification $\mathcal{Q}$ from t_0 if for all $(x_0, z_0) \in (X \times Z)$, for all $u \in \mathcal{U}$, for all $t > t_0$, we have that: $\Psi(t, \Phi(t, t_0, (x_0, z_0), u), u(t)) > 0$.*

The verification problem *($\mathcal{H}$, $\mathcal{Q}$, t_0) consists in checking if $\mathcal{H}$ satisfies its specification $\mathcal{Q}$ from t_0 (*PASS*) or, otherwise (*FAIL*) in providing a counterexample, i.e., a state $(x_0, z_0) \in (X \times Z)$, an admissible input (operational scenario) $u \in \mathcal{U}$ and a time instant $t > t_0$ such that $\Psi(t, \Phi(t, t_0, (x_0, z_0), u), u(t)) \leq 0$.*

To limit complexity, an upper limit h (*horizon*) is usually given to the value of t in Definition 2.7. In this case, we have a *time-bounded verification* problem.

2.2.4 Statistical Model Checking

From Definition 2.7 we see that a verification problem can be cast as the problem of finding a state, an admissible input (operational scenario) and a time instant such that the system KPI is negative (*i.e.*, requirements are violated).

Of course, the above problem is computationally prohibitive in general. Thus many techniques have been developed to compute an approximate solution to it with formal assurance about the verification outcomes. Basically, we have two main options: *first*, focus on simple models (*e.g.*, finite state systems) to make the problem tractable; *second*, use statistical techniques to sample the environment (set of operational scenarios) in order to answer the verification problem (Definition 2.7) with some given statistical confidence.

We note that probabilistic model checking (*e.g.*, PRISM [73]) is an exact approach since it computes exactly the probability of reaching certain states and it suffers of the state explosion problem, much as the deterministic model checking. To overcome this, statistical techniques are used. Furthermore, non exhaustive approaches, for example *falsification* [1], do not provide a formal guarantee about the verification outcome. Thus, both probabilistic model checking (see [69]) and non exhaustive verification are out of our scope, because: probabilistic model checking returns an exact result; non exhaustive verification does not provide a statistical confidence level.

SMC follows the *second* approach and indeed offers a set of methods and software tools to solve the above problem in many different settings.

In the following, we will survey the available tools to carry out formal verification via SMC and categorize them on the basis of the environment model (*i.e.*, space of input functions in Definition 2.2), SUV model (*i.e.*, system in Definition 2.4) and specification model (*i.e.*, SUV specification in Definition 2.5) they can support. This will help users in deciding which tool to use for a given verification task trading off between completeness of the environment model (all operational scenarios represented), faithfulness of the SUV model (all possible behaviors represented) and expressiveness of the specification language (all requirements are adequately formalized).

Finally, from a computational perspective, we will categorize tools with respect to the SMC algorithm used. This will enable users to evaluate which kind of parallelism they can expect to easily implement on the basis of the sampling mechanism used.

2.3 Background

By defining a probability measure on our set of operational scenarios $\mathcal{U}$ (space of input functions - see Definition 2.2), we can regard the deterministic system (SUV) in Definition 2.4 as a stochastic system. In such a context, the verification problem in Definition 2.7 will aim at estimating the probability that the system specification is satisfied. Such a probability can be computed using numerical as well as statistical techniques.

Numerical Model Checking (NMC) techniques (*e.g.*, [133], [12]) need an explicit finite state model (*white box*) for the SUV. They compute accurate results, but do not scale very well to large systems because they suffer from the *state space explosion problem*.

SMC methods avoid an explicit representation of the SUV state space. They use a simulation-based approach and statistical inference over observations to establish if a property, specified through a stochastic temporal logic (which is then transformed into a specification for the SUV as in Definition 2.5), is true within statistical confidence bounds. Compared to numerical techniques, SMC overcomes the state explosion problem and scales better to big systems, but only guarantees correctness in a statistical sense. Nevertheless, sometimes it is the only feasible approach to the problem.

SMC takes as input a model defining the dynamics of a system $\mathcal{S}$ and a property φ, defined as a logical formula. The goal of SMC is to decide whether $\mathcal{S}$ satisfies φ with a probability greater than or equal to a certain threshold α. In formal terms, we denote the SMC problem as $\mathcal{S} \models P_{\geq\alpha}(\varphi)$.

In the next sections, we describe the models that can be used to represent the system and the kind of temporal logics used to express the properties. Since all approaches are simulation-based, we have that all SUV models and languages to define properties discussed in the following can be transformed, respectively, into the general definitions in Definition 2.4 and Definition 2.5.

2.3.1 System Models

Quite often rather than considering deterministic systems with stochastic inputs as in Definition 2.4, SMC considers systems without inputs with nondeterministic or stochastic transitions. This can be accomodated in our setting as discussed at the beginning of Section 2.2. Accordingly, in our context, we will consider these two modelling approaches as equivalent.

SMC input systems can be modelled through different kind of structures, such as *DTMC* or *CTMC* [134, 12] and by *Generalized Semi Markov Process (GSMP)* [108].

A DTMC is defined as a tuple $M = (S, s_0, P, L)$, where: S is a set of states (*state space*), $s_0 \in S$ is the initial state, $P : S \times S \to [0, 1]$ is the transition function, and L is a function labeling states with atomic propositions. A CTMC is a *Markov Chain* where time runs continuously. GSMP [126] is a stochastic process where the transition from one state to another depends on a set of several possible events associated with the current state.

Furthermore, SMC can be applied to *DESP*, consisting of a set of states, that can be infinite, and whose dynamic depends on a set of discrete events. DESP can be modelled as *Generalized Stochastic Petri Net (GSPN)* [15], that is a bipartite graph consisting of two classes of nodes: *places*, representing the state of the modelled system; *transitions*, encoding the model dynamics.

SMC algorithms for *Probabilistic Timed Automata (PTA)* are also available [90]. Using PTA we can represent systems with probabilistic and real-time features.

More recently, SMC has been applied also to timed and hybrid systems with a stochastic behavior, like *Stochastic Hybrid Automata (SHA)* and *Stochastic Timed Automata (STA)* (see [31]). Some SMC algorithms accept as input models also *Markov Decision Processs (MDPs)* [77], which are particular sequential decision models [100] characterized by a set of states, a set of actions, probabilistic transitions, and rewards, or costs, associated to the actions, such that each decision depends only on the current state and not on the past. MDPs are often used to model concurrent process optimization problems.

SMC tools like VeSTa or MultiVeSTa, that will be discussed in the following Section 2.5.1 and Section 2.5.2, take as input models whose behavior is described through *Probabilistic rewrite theories* [2], which is a general high-level formalism to specify probabilistic systems.

The SMC tool SAM (in Section 2.5.16) works on systems modelled through StoKLAIM [34], which is a Markovian extension of KLAIM (*Kernel Language for Agents Interaction and Mobility*) [33], a language used to model mobile and distributed systems.

2.3.2 System Properties

The properties to be evaluated represent a set of behaviors of the system under verification and can be qualitative or quantitative. Verifying a *qualitative* property means deciding between two mutually exclusive hypotheses, namely if the probability to satisfy the property is above or below a certain threshold. *Quantitative* properties concern computing the estimation of a stochastic measure, *i.e.*, the expected value of the probability that a given property is satisfied.

Properties are expressed through a temporal logic and in literature many different kinds of logics exist, as outlined below.

Linear Temporal Logic (LTL) [103] is the top family of temporal logics that reason along linear traces through time, that is each instant is followed by only one future step. *Bounded Linear Temporal Logic (BLTL)* extends *LTL* by adding upper time bounds (bounded time of steps when the time domain is discrete) to temporal operators. *Probabilistic Bounded Linear Temporal Logic (PBLTL)* adds probabilistic operators to BLTL. *PBLTLc* corresponds to PBLTL with numeric constraints. Metric Temporal Logic (MTL) [87] and Metric Interval Temporal Logic (MITL) [7] both extend LTL by introducing an explicit representation of time.

Another class of stochastic logics considers time evolving in a branching fashion, rather than on a line. This class includes the *Computational Tree Logic (CTL)* [26], used to express qualitative properties over (non-stochastic) transition systems. *Probabilistic Computational Tree Logic (PCTL)* [50] extends CTL by including operators and time bounds to express the quantitative properties of a DTMC, say M. If s is a state of M and φ is a PCTL formula, then $s \models P_{\geq\alpha}(\varphi)$ means that "the probability that φ is satisfied in the next state of outgoing paths from state s is greater than or equal to α".

Several extensions of PCTL exist, like *Continuous Stochastic Logic (CSL)* [109], used to express quantitative properties of *CTMCs*; *Quantitative Temporal Expressions (QuaTEx)* [2], a query language used to investigate quantitative aspects of probabilistic rewrite systems; *PCTL** [11], used to specify quantitative properties of *DTMCs* or *MDPs*. *PCTL** formulas can be interpreted over the states of a fully or concurrent probabilistic system. CSL^{TA} [37] is a superset of CSL for CTMC that allows specifying path formulas through a 1-clock Deterministic Timed Automaton (DTA).

Hybrid Automata Stochastic Language (HASL) is a temporal logic formalism used to define properties over DESP (Section 2.3.1). A HASL formula consists of an automaton and an expression. The *automaton* is a *Linear Hybrid Automaton (LHA)*, *i.e.*, a hybrid automaton with data variables whose dynamic depends on the modelled system states. The *expression* refers to the moments of path random variables associated with path executions of the system.

Finally, *Mobile Stochastic Logic (MoSL)* is a temporal stochastic logic introduced in [34]. MoSL is based on CSL and allows us to refer to the spatial structure of a mobile network. MoSL can be used to specify both qualitative and quantitative properties.

2.4 Statistical Inference Approaches

When using SMC algorithms, the system is simulated many times by executing Monte Carlo experiments. The property φ to be verified is checked at each simulation (sample). Let us associate a Bernoulli random variable Z to one simulation. We assume that Z is 1 if φ is satisfied, 0 otherwise, and that $Pr[Z = 1] = p$ and $Pr[Z = 0] = 1 - p$.

HT, described in Section 2.4.1, *Estimation*, Section 2.4.2, and *Bayesian analysis*, Section 2.4.3, can be used as engines for the SMC algorithms, as outlined below.

2.4.1 Hypothesis Testing

The hypothesis testing approach consists of the evaluation of two mutually exclusive hypotheses, namely the *null hypothesis* $H_0 : p \geq \theta$, and the *alternative hypothesis* $H_1 : p < \theta$, where θ is a given threshold value. If we assert that H_0 is false while it is true, we make a *Type I* error (or *false positive*); if we decide to reject H_1, while it is true, we make a *Type II* error (or *false negative*). Let us denote with α the probability of Type I errors and with β the probability of Type II errors.

HT can be used to analyze both *qualitative* and *quantitative* properties. It can be performed on a set of samples that can be generated in advance or on demand. In the former case, the sample size is fixed. In the latter case, called *sequential testing*, the samples are collected incrementally and their number is not predetermined. In a sequential testing approach, the decision of whether or not to continue sampling is made based on the samples generated up to that point. Mixtures of the two types of sampling generation are possible. This is discussed in [101], where several approaches to HT are described, differing from each other in the guarantees they give about the correctness of the result. The algorithms implementing these approaches are the following: the *Chernoff C.I.* method, using a fixed sample size test, resting on a confidence interval based on the Chernoff–Hoeffding bound [56]; the *Gauss C.I.*, that is a similar procedure based on the Gaussian approximation to determine the confidence interval; the *Chow-Robbins* test, a sequential test method that continues sampling until the width of the confidence interval has reached a given value; the *Sequential Probability Ratio Test (SPRT)* technique of Wald [125] and its fixed sample size variant, the *Gauss-SSP* test, where SSP stands for *Single Sampling Plan* (see [130] for details).

Compared to simple sequential testing, SPRT is optimal in the sense that it minimizes the average sample size before a decision is made. The last method is the *Chernoff-SSP* test, which is a variant of the Gauss-SSP test using the Chernoff-Hoeffding bound instead of the Gaussian approximation.

When properties to be verified are rare, that is the probability that they hold is low, the number N of simulations required can explode. Fortunately, two techniques such as *Importance Sampling (ISA)* and *Importance Splitting (ISS)* can be applied to reduce N. ISA works by sampling through a weighted distribution, such that a rare property is more likely to be observed. Then, the results are compensated by the weights, to estimate the probability under the original distribution. ISS works by decomposing the sampling from the rare probability distribution into the sampling from the product of less rare conditioned probabilities. See [63], [62] and [76] for an in-depth discussion about the two mentioned rare properties techniques.

2.4.2 Estimation

Estimation is the kind of inference approach typically used to verify quantitative properties. Existing SMC algorithms rely on classical Monte Carlo based techniques to estimate a property with a corresponding *confidence interval*. The system model is repeatedly simulated and, if φ is the property to be evaluated and p is the probability to satisfy φ, an (ϵ, δ) approximation $\hat{p}$ of p is computed, with the assurance that the probability of error is $Pr(|\hat{p} - p| \geq \epsilon) \leq \delta$, where ϵ and δ are, respectively, the precision and the confidence.

The confidence interval can be computed using different approaches, each one affecting the number of simulations N to perform. For instance, according to the Chernoff-Hoeffding bound [56] N is fixed a priori by setting $N = (\ln 2 - \ln \delta)/(2\epsilon^2)$. The *Confidence Interval (CI)* method, taken as input two parameters α and δ, samples until the size of the $(1-\alpha) \times 100\%$ confidence interval, computed using the *Student's t-test*, is bounded by δ.

In [46, 47] Grosu and Smolka show how to compute an (ϵ, δ) approximation $\hat{p}$ of p satisfying the property:

$$Pr[p(1-\epsilon) \leq \hat{p} \leq p(1+\epsilon)] \geq (1-\delta) \tag{2.1}$$

This can be done with an optimal number of simulation traces N, through the Optimal Approximation Algorithm $\mathcal{OAA}$ introduced by Dagum *et al.* in [30]. $\mathcal{OAA}$ uses the minimum number N of traces to compute an (ϵ, δ) approximation, within a constant factor, of a random variable distributed in $[0, 1]$. $\mathcal{OAA}$ algorithm improves the classic Monte Carlo algorithm, based on the hypothesis testing approach, since N is computed at runtime. This result is obtained by applying the sequential testing approach of Wald [125] to decide when to stop simulating. As well as for HT, discussed in Section 2.4.1, when quantitative properties to be verified are rare, ISA or ISS can be applied (see [63], [62] and [76]), together with the algorithm selected to solve the estimation problem, to reduce the number of simulations to perform.

2.4.3 Bayesian analysis

The Bayesian approach to SMC is based on the usage of *Bayes's theorem* and *sequential sampling*, and can be associated with both HT and Estimation. Also in this case, as for Hypothesis Testing (Section 2.4.1) and Estimation (Section 2.4.2) algorithms, the system model is repeatedly simulated. The difference is that Bayes's theorem requires prior knowledge of the model to be verified, which is the *prior* density of a given random variable. Sequential sampling, as discussed in Section 2.4.1, means that the number of simulations to perform is decided at run-time. Properties to be verified are expressed as BLTL formulas and input systems are modelled as Discrete Time Hybrid Systems (DTHS) (see Section 2.2).

The algorithm to perform SMC by Bayesian HT, introduced in [64], at each trace of the system model, generated by simulation, checks if a certain property is satisfied or not. Then, it uses the *Bayes factor* $\mathcal{B}$ to decide if the *null hypothesis* H_0 or the *alternative hypothesis* H_1 has to be accepted.

Zuliani *et al.* in [139] propose a Bayesian statistical Estimation algorithm. The main goal of this approach is estimating the (unknown) probability p that a random execution trace of the system model satisfies a fixed property ϕ. The estimate corresponds to a *confidence interval*, *i.e.*, an interval that contains p with a high probability. Since p is unknown, it must be assumed that p is given by a random variable whose density is known *a priori*. The Bayesian Estimation algorithm iteratively generates traces of the system model and for each trace executes the following steps: checks whether ϕ is satisfied; computes the posterior mean, that is the Bayes estimator, of p; computes an interval estimate; produces a posteriori probability of p.

Finally, Bortolussi *et al.* in [22] present a novel approach, to the statistical model checking, based on the Bayes analysis and called *Smoothed Model Checking*. Their goal is computing a statistical estimation (and a confidence interval) of the satisfaction of a MITL property against an *Uncertain* CTMC, that is a CTMC indexed by a parameter vector.

2.4.4 Summary of the Algorithms Used for HT and Estimation

In Table 2.1, we summarize the main algorithms, explained in the previous sections, that can be used to carry out Hypothesis testing or Estimation. For each algorithm, we specify if the number N of simulation runs to execute is predetermined or not.

2.5 Statistical Model Checking Tools

In this section, we describe some of the existing Monte Carlo based SMC tools, focusing on those that are directly available to the academic community. There exist tools using simulation-based techniques, but whose output is not a statistical confidence interval, that are not included in our review, like: SyLVaaS [81], a Web-based software-as-a-service tool for System Level Formal Verification of CPSs; S-TaLiRo [9], that is a tool performing temporal logic falsification of properties over non-linear hybrid systems; Ariadne [24], a tool for the formal verification of CPSs, using reachability analysis for nonlinear hybrid automata; SpaceEx, a tool

Algorithm	HT	E	#Samples fixed a priori
Gauss-SSP	•		*yes*
C.I.		•	*yes*
Chernoff C.I.	•	•	*yes*
Chernoff SSP.	•	•	*yes*
Chow-Robbins	•	•	*no*
SPRT	•		*no*
Bayesian HT	•		*no*
Bayesian E		•	*no*
$\mathcal{OAA}$		•	*no*

Table 2.1. Summary of the algorithms used to perform *Hypothesis Testing* and *Estimation*. In the last column *yes* means that the algorithm pre-computes the number of samples, *no* means that the number of samples is computed at runtime.

platform for reachability and safety verification of continuous and hybrid systems [40]; Symbolic PathFinder [78], a SMC tool implementing approximate algorithms for MDP.

For each surveyed tool, we give a brief description and some details about: the kind of models supported; the languages used to specify the properties to be verified; the statistical inference approach used (HT and/or Estimation and/or Bayesian analysis). The information about the environment model, that stem from the kind of model, are summarized in Table 2.2.

Tools performing statistical inference by algorithms not discussed in Section 2.4 (*e.g.*, SAM, GreatSPN) will not be included in the taxonomy given in Table 2.2.

2.5.1 (P)VeStA

VeStA is a Java-based tool for statistical analysis of probabilistic systems. It implements the statistical methods from [134] and [109], based on Monte-Carlo simulation and simple statistical hypothesis testing [57].

Model and Property. This tool can verify properties expressed as CSL/PCTL formula (Section 2.3.2), against probabilistic systems specified as CTMCs or DTMCs (Section 2.3.1). VeStA is able also to statistically evaluate, in a Monte Carlo based way, the expected value of properties expressed in *QuaTEx* (Section 2.3.2), over the observations performed on *probabilistic rewrite theories* models. In this last case, models are described through PMAUDE, which is an executable algebraic specification language [2].

Statistical Inference approach. VeStA performs SMC by using classic statistical hypothesis testing (Section 2.4.1), rather than sequential hypothesis testing, according to the algorithm described in [110]. In particular, VeStA implements the *Gauss-SSP* hypothesis testing (Section 2.4.1), which is a fixed sample size version of the SPRT algorithm of Wald (Section 2.4.1). As a consequence, it is easily parallelizable. PVeStA [5] is the tool extending and parallelizing the SMC algorithms implemented in VeStA.

2.5.2 MultiVeStA

MultiVeStA [107] extends VeStA (and PVeStA) with an important feature that is the integration with several existing discrete event simulators, in addition to the already supported ones.

Model and property. Properties to be verified are expressed in *Multi Quantitative Temporal Expressions (MultiQuaTEx)* query language, which extends *QuaTEx* (Section 2.3.2) and allows to query more variables at a time through multiple observations on the same simulation. This represents an improvement of the performance obtained when evaluating several expressions. The supported SUV models are Discrete Event Systems (DESs).

Statistical Inference approach. MultiVeStA performs estimation of the expected value of *MultiQuaTEx* properties by HT Chow-Robbins method (Section 2.4.1).

2.5.3 Simulation-based SMC for Hybrid Systems

There are many tools supporting SMC for hybrid systems, for example: ProbReach [112], Probably Approximately Correct (PAC) for hybrid systems verification [128, 129], Plasma [99]. Here we focus on tool that can interface in a black-box fashion with any available simulator for hybrid systems. One of those is Plasma [61, 23] which is a SMC platform that may be invoked from the command line or embedded in other software as a library. This tool uses external simulators (*e.g.*, Simulink, SystemC) to define the SUV as well as the property to be verified.

Model and Properties. Plasma Lab accepts properties described as BLTL extended with customized temporal operators, against stochastic models such as CTMCs and MDPs (Section 2.3.1).

Statistical Inference approach. Plasma Lab can verify both qualitative and quantitative properties. In fact, the tool implements, among others, the following algorithms: the Monte Carlo probability estimation based on the Chernoff-Hoeffding bound [56], to decide *a priori* the number of simulations to execute; the SPRT algorithm for hypothesis testing; *ISA* when properties are "rare" (Section 2.4.1).

2.5.4 APMC

The *Approximate Probabilistic Model Checker* (APMC) is a model checker implementing an efficient Monte Carlo-based method to approximate the probability that a monotone property is satisfied, with high confidence, by a fully probabilistic system. This algorithm is described in [54]. In [74] the optimal approximation algorithm $\mathcal{OAA}$ (Section 2.4.2) is used to obtain a randomized approximation scheme with multiplicative error parameter.

Model and property. APMC is used to verify quantitative properties over fully probabilistic transitions systems or DTMCs (Section 2.3.1). In 2006, APMC has been extended to manage also CTMCs (see [97]). Properties to be checked are expressed as LTL (Section 2.3.2) formulas.

Inference approach. This tool performs estimation through a Monte-Carlo sampling technique, based on the Chernoff-Hoeffding bound (Section 2.4.1), which is naturally parallelizable.

2.5.5 PRISM

PRISM [73] is a Probabilistic Symbolic Model Checker that from the version 4.0 offers support for SMC of several types of probabilistic models. Furthermore, in [53] a parallelized version of PRISM has been extended to handle MDPs.

Model and property. DTMCs, CTMCs, MDPs models (Section 2.3.1) are described through the PRISM modelling language while properties to be verified are defined by several probabilistic temporal logics, incorporating PCTL, PCTL*, CSL and LTL (Section 2.3.2).

Statistical Inference approach. The tool uses the SPRT (Section 2.4.1) to verify qualitative properties and the following algorithms to verify the quantitative properties (Section 2.4.2): *CI* method, *Asymptotic Confidence Interval (ACI)* method, and *Approximate Probabilistic Model Checking (APMC)* method. All these algorithms precompute the number of samples to be generated. See [95] for an updated detailed description.

2.5.6 Ymer

Ymer, illustrated in [131], is a command-line tool written in C and C++, that implements the SMC techniques based on discrete event simulation and acceptance sampling. Furthermore, in the Ymer tool, two sampling-based algorithms for the statistical verification of time-unbounded properties of DTMCs are implemented (see in [132]).

Model and property. This tool can verify CSL properties against CTMCs and PCTL properties against DTMCs (Sections 2.3.1 and 2.3.2).

Statistical Inference approach. Ymer implements both sampling with a fixed number of observations and sequential acceptance sampling, performed through the SPRT method (Section 2.4.1). Ymer includes support for distributed acceptance sampling, *i.e.*, the use of multiple machines to generate observations, which can result in significant speedup as each observation can be generated independently. The work in [132] implements in Ymer also estimation through two different approaches, the first based on Chernoff C.I and the second based on the Chow-Robbins sequential method.

2.5.7 UPPAAL-SMC

UPPAAL-SMC [32] is a stochastic and statistical model checking extension of UPPAAL [18], that is a toolbox for the verification of real-time systems, jointly developed by Uppsala University and Aalborg University.

Model and Properties. UPPAAL-SMC implements techniques to verify both quantitative and qualitative properties of timed and hybrid systems with a stochastic behavior, whose dynamic can be specified by *SHA*, effectively defining ODEs, and by STA [31]. Properties are expressed through a weighted extension of the temporal logic MITL (Section 2.3.2).

Statistical Inference approach. This tool carries out quantitative properties verification through a Monte Carlo based estimation algorithm using the Chernoff-Hoeffding bound (Section 2.4.1), where the number of samples to be generated is predetermined. Qualitative properties are verified through the SPRT algorithm (Section 2.4.1).

2.5.8 COSMOS

COSMOS [15] is a statistical verification tool implemented in C++.

Model and property. This tool analyzes DESP, a class of stochastic models including CTMCs, represented in the form of a GSPN (Section 2.3.1). Properties to be verified are expressed as HASL formulae (Section 2.3.2).

Statistical Inference approach. COSMOS relies on Confidence Interval based methods to estimate the probability that the property under verification holds, by implementing two possible approaches: the *static sample size estimation*, based on the Chernoff-Hoeffding bound (Section 2.4.1), where the sample size is fixed *a priori*; the *dynamic sample size estimation*, where the sample size depends on a stopping condition, such as that provided by Chow and Robbins (Section 2.4.1). COSMOS provides also the SPRT method.

2.5.9 GreatSPN

GreatSPN [8] is a tool that supports the design and the qualitative and quantitative analysis of GSPN. GreatSPN contains, in particular, two modules: a CTL model checker and a CSL^{TA} model checker.

Model and Properties. GreatSPN can verify: CTL properties against models represented as GSPN or its colored extension, defined as *Stochastic Symmetric Nets* (SSN); CSL^{TA} properties (Section 2.3.2) against CTMCs.

Statistical Inference approach. The CTL model checker of GreatSPN verifies CTL properties by numeric symbolic algorithms (see Section 2.3). The CSL^{TA} module is a probabilistic model checker for estimation of properties that can interact also with external tools like PRISM (Section 2.5.5) and MRMC (Section 2.5.10).

2.5.10 MRMC

MRMC (Markov Reward Model Checker) [68] is a command-line tool, written in *C* language, that implements SMC techniques. This tool is a *Probabilistic Model Checker* rather than a SMC. However, as MRMC performs statistical inference through Monte Carlo-based techniques, it is included in our review, for sake of completeness.

Model and property. The tool can verify CSL and PCTL properties (Section 2.3.2) against CTMCs and DTMCs (Section 2.3.1).

Statistical Inference approach. MRMC performs probability estimation by the *Confidence interval* method based on the Chow-Robbins test (Section 2.4.1), with a *dynamic sample size.* The only problem is that since MRMC always loads Markov chain representation completely in memory, it can lose the benefits deriving from simulating instead of using numerical techniques.

2.5.11 SBIP

SBIP [91] is a statistical model checker for the BIP (Behavior, Interaction, Priority) general framework for the design of component-based systems developed at Verimag (see [124]).

Model and property. It supports DTMC, CTMC and GSMP (Section 2.3.1) as input models. Properties to be verified can be expressed as PBLTL and MTL formula (Section 2.3.2).

Statistical Inference approach. The tool implements several statistical testing algorithms for stochastic systems verification of both qualitative and quantitative properties. Qualitative properties are checked through one of the following algorithms: *Single Sampling Plan (SSP)* [75], where the number of samples is predetermined, and SPRT, where the number of samples is generated at runtime (Section 2.4.1). Quantitative properties are verified through a *Probability Estimation* procedure, based on the Chernoff-Hoeffding bound (Section 2.4.1).

2.5.12 MARCIE

MARCIE [52] is a tool written in C++ and made up of several analysis engine, in particular a Stochastic Simulation Engine based on statistical estimation.

Model and property. This tool can verify both quantitative and qualitative properties over systems modelled as GSPN, including CTMC (Section 2.3.1). Properties can be defined by CSL, Continuous Stochastic Reward Logic (CSRL) or PLTLc. CSRL includes CSL and adds reward intervals to the temporal operators (Section 2.3.2).

Statistical Inference approach. The component of MARCIE dedicated to estimation implements an algorithm performing several simulation runs depending on the variance of the system stochastic behavior and determined through a *Confidence interval* method based on the Chernoff-Hoeffding bound (Section 2.4.1).

2.5.13 Modest toolset discrete event simulator: `modes`

Modest [21] is a high-level compositional modelling language for STA. It is completely supported by the Modest Toolset, available at [88], which includes tools to analyze different subsets of the STA, e.g PTA, MDP, DTMC, CTMC (Section 2.3.1). In particular, the `modes` tool, a discrete event simulator, based on the Modest language, is available. From version 1.4 it offers SMC functionalities.

Model and property. `modes` supports the analysis of SHA, STA, PTA, and MDP. Properties to be verified are expressed in Modest language.

Statistical Inference approach. `modes` verifies quantitative properties through a *confidence interval* based algorithm, that allows deciding at runtime how many simulations to do; qualitative properties through the SPRT method (Section 2.4.1).

2.5.14 APD Analyser

Aggregated Power Demand (APD) Analyser ([82, 83]) is a SMC based *domain-specific* tool to compute the APD probability distribution in a smart grid scenario in presence of highly dynamic individualised electricity price policies [84, 51].

Model and property. The input model consists of a probabilistic model of end-user deviations with respect to their expected behaviors. The tool computes a single *domain-specific* property: the APD probability distribution in Electric Distribution Networks. Further post-processing of this distribution allows the Distribution System Operators (DSO) to compute the safety properties of interest.

Statistical Inference approach. As an exact computation of the required APD probability distribution is computationally prohibitive, because of its exponential dependence on the number of users, APD-Analyser computes Monte Carlo based (ϵ, δ)-approximation, through an efficient *High Performance Computing (HPC)*-based implementation of the $\mathcal{OAA}$ algorithm, discussed in Section 2.4.2.

2.5.15 ViP generator

The works in [121, 85, 115] present a tool called *ViP* (Virtual Patient generator). This tool uses a SMC based approach to compute complete populations of virtual patients for *in silico* clinical trials and the model-based synthesis of optimal personalised pharmaceutical treatments [80, 116].

Model and property. The input model is a system of ODEs and the boolean property to be satisfied is the completeness of the virtual patient set generated.

Statistical Inference approach. HT (Section 2.4.1) is used to check, with high statistical confidence, completeness of the virtual patient set S generated so far. After defining a probability threshold ϵ and a confidence threshold δ, the SMC algorithm in [121] randomly extracts, from a parameter space Λ, a sample $\{\lambda\}$ that, if admissible, is added to S. On the basis of [46, 47], the algorithm ends when S remains unchanged after $N = \ln \delta / \ln(1 - \epsilon)$ attempts.

2.5.16 SAM

SAM (Stochastic Analyzer for Mobility) [34] is a tool supporting the analysis of mobile and distributed systems. SAM uses a statistical model checking algorithm to verify whether a mobile system satisfies a certain property.

Model and Properties. Systems are specified in StoKLAIM (Section 2.3.1). Properties to be verified are expressed through MoSL (Section 2.3.2).

Statistical Inference approach. SAM performs the estimation of quantitative properties.

2.5.17 Bayesian Tool

The Bayesian analysis consists of the introduction of Bayesian statistics to the SMC approach, as discussed in Section 2.4.3.

Model and property. Models to be analyzed are DTHS (see Section 2.2) defined as Stateflow/Simulink models, while properties are expressed as BLTL formula.

Statistical Inference approach. The Bayesian analysis includes: i) an algorithm to perform SMC using *Bayesian hypothesis testing* to verify BLTL properties against Stateflow/Simulink models with hybrid features; ii) a *Bayesian estimation* algorithm, to compute an interval estimate of the probability that a BLTL formula is satisfied in a stochastic hybrid system model.

2.5.18 Tool Comparison

In Table 2.2, we show a taxonomy of the surveyed tools, based on their main properties, namely: the environment model, characterized by the time (discrete or continue), the set of events values (that can be finite/infinite); the SUV model, consisting of the kind of models or language used to represent the system and the set of states of the model (that can be finite/infinite); the property specification, that is the stochastic logic used to specify the properties to be verified and the search horizon (that can be bounded/unbounded); the verification technique, consisting of the statistical inference procedure used (HT and/or Estimation and/or Bayesian analysis) together with the algorithm implemented to perform the inference. The algorithm is useful to know if the number of simulation runs is fixed beforehand or not (as shown in Table 2.1).

In addition to the tools extensively reviewed in [101] and [3], we have included the Bayesian analysis and two other SMC domain-specific software tools, the *APD Analyser* and the *ViP* (Virtual Patient) *generator*.

2.6 Discussion

The taxonomy showed in Table 2.2 highlights the following trade-offs.

The *Environment model* is: *complete* if it contains all possible operational scenarios; *sound* if it contains only the operational scenarios that can occur in a real situation. Ideally, we would like a complete and sound environment model. However, in practice, this is not easy to define. Depending on priorities, correctness, or efficiency, we may select a complete (possibly unsound) model or a sound (possibly incomplete) model.

A *SUV model* has to capture the dynamics of the system. If the properties to be verified are of the *safety* kind, tools taking as input an *over-approximated* model (*i.e.*, containing more behaviors than the real system) have to be preferred. If *liveness* properties have to be verified, tools accepting *under-approximated* models can be used.

The property language has to be selected with respect to its capability to define the property to be verified. The search horizon should be *large enough*, but not *too large*, in order to avoid making simulations overly time consuming. In this case the trade-off is between the expressiveness and the computational complexity of the verification activity.

The verification technique selection depends on the goal (namely, estimation or HT) and on the need to know beforehand, or not, the number of samples to be generated. Accordingly, all the tools implement the SPRT to perform HT on qualitative statements, except: (P)VeStA that uses the Gauss-SSP test, through which the sample size is predetermined; the Bayesian analysis, that bounds the sample size using a test based on the Bayes factor. SBIP, besides SPRT, implements also SSP.

The estimation of quantitative properties is performed through different techniques. (P)VeSta uses the Confidence Interval method to evaluate QuaTEx formulae; MultiVeStA, COSMOS (from version 1.0) and MRMC (from version 1.5) implement the Chow-Robbins test; Plasma Lab, UPPAAL-SMC, PRISM (from

	ENVIRONMENT MODEL		SUV MODEL		SPECIFICATION		VERIFICATION TECHNIQUE	
TOOL	Time	Event values	Model	Set of states	Property language	Search horizon	Inference	#samples computing
(P)VeStA [110]	C	fin	CTMC, DTMC/ PMaude	fin/inf	CSL, PCTL, QuaTEx	ubnd	HT: Gauss-SSP; E: C.I.	HT and E: a priori
MultiVeStA [107]	D/C	fin	DES	inf	MultiQuaTEx	ubnd	E: Chow-Robbins	E: at runtime
Plasma [99]	C	fin/inf	CTMC, MDP	inf	BLTL	bnd	HT: SPRT; E: Chernoff C.I.	HT: at runtime; E: a priori
APMC [48, 74]	D	inf	DTMC, CTMC	fin	LTL	ubnd on monotone LTL	E: Chernoff-SSP/$\mathcal{OAA}$	E: a priori/at runtime
PRISM [73, 53]	D/C	fin	DTMC, CTMC, MDP	fin	BLTL	ubnd	HT: SPRT; E: Chernoff C.I.; NS	HT: at runtime; E: a priori
Ymer [131, 132]	D/C	inf	DTMC, CTMC	fin	PCTL, CSL	ubnd	HT: SPRT/Gauss-SSP; E: Chow-Robbins/ Chernoff C.I.	HT: at runtime/a priori; E: at runtime/a priori; NS
UPPAAL-SMC [32]	C	inf	SHA	inf	MITL	bnd	HT: SPRT; E: Chernoff C.I.	HT: at runtime; E: a priori
COSMOS [15]	C	fin	GSPN	fin	HASL	ubnd	HT: SPRT; E: Chernoff C.I. / Chow-Robbins	HT: at runtime; E: a priori/at runtime
MRMC [68]	D/C	fin	DTMC, CTMC	fin	PCTL, CSL	ubnd	E: Chow-Robbins; NS	E: at runtime
SBIP [91]	D/C	inf	DTMC, CTMC, GSMP	inf	PBLTL	bnd	HT: SPRT; E: Chernoff C.I.	HT: at runtime; E: a priori
MARCIE [52]	C	fin	GSPN	fin	CSL, CSRL, PCTL	ubnd	E: Chernoff C.I.; NS	E: at runtime
`modes` [21]	C	fin	SHA, STA, PTA, MDP	fin	MODEST	ubnd	HT: SPRT; E: Chernoff C.I.	HT: at runtime; E: a priori
APD Analyser [83]	D	fin	Custom model	inf	Safety properties	bnd	E: $\mathcal{OAA}$	E: at runtime
ViP generator [121, 85]	D	fin	ODEs	inf	Boolean properties	bnd	HT: SPRT	HT: at runtime
Bayesian tools [139, 22]	D	fin	DTHS, Uncertain CTMC	inf	BLTL, MTL,MITL	bnd	HT: Bayesian HT; E: Bayesian E	HT and E: at runtime

Table 2.2. Monte Carlo simulation-based SMC tools comparison table. In the **Time** column *D* stands for discrete, *C* for continue. In the column **Model**, the model is specified as its representation structure or as the language describing the model. In the columns **Event values** and **Set of states**, fin means *finite* and inf means *infinite*. In the **Search horizon** column, *bnd* stands for *bounded*, *ubnd* for *unbounded*. In the **Inference** column, *HT* stands for *Hypothesis Testing*; *E* stands for *Estimation* and *NS* stands for *Numeric-symbolic* methods (see Section 2.3). In the same column, for each inference approach, the name of the algorithm used is specified. For further details about the algorithms see Table 2.1. Depending on the algorithm, the number of samples can be computed a priori or at runtime.

version 4.0), SBIP, and MARCIE use the Chernoff-C.I. method; Ymer uses the Gauss-SSP test; APMC (from v3.0) implements the so-called Chernoff-SSP test; the APD Analyser implements the $\mathcal{OAA}$ algorithm (see Table 2.1). The Bayesian Estimation procedure is different from all the others, because it is based on the Bayes factor test, but like with the Chow Robbins test, the sample size is not fixed *a priori*.

If we have to choose a Monte Carlo based SMC tool minimizing the number of simulations to do, in case of qualitative statements, all the tools implementing the SPRT to solve HT are suitable. On the other hand, an a priori fixed sample size technique is easily parallelizable, then sometimes this kind of approach could be preferable. In the case of quantitative statements, the tools implementing the Chow-Robbins test or $\mathcal{OAA}$ are the fastest. The Bayesian analysis also allows us to decide at runtime how many simulations to do, but it needs to know in advance the probability distribution of the variables to be estimated.

As a last result, Table 2.2 highlights an open research challenge, that is the lack of tools to perform an unbounded verification of hybrid systems whose environment has continuous- or discrete-time.

2.7 Conclusions

We have reviewed most of SMC tools currently available for research purposes, focusing on the complexity, in terms of sample size, of the Monte Carlo-based techniques used to evaluate quantitative and qualitative properties through *HT*, *Estimation*, and *Bayesian analysis*.

For each tool, we have highlighted: the environment model; the SUV model; the stochastic logic used to define the properties to be verified, together with the search horizon type; the statistical inference procedure and the corresponding algorithm used, by explicitly indicating if the number of samples is generated at runtime or not. After an in-depth comparison, we produced the taxonomy in Table 2.2.

First, we observe that most of the tools assume that the environment is *sound*, thus events are taken from a finite set. In this scenario, the SUV model set of states is typically finite and the search horizon for the properties to be checked is *unbounded*. Otherwise, if the SUV model set of states is infinite, the search horizon is always *bounded*.

Second, 90% of the tools performing *HT* use algorithms deciding at runtime the number of samples to be generated; about 70% of the tools implementing *Estimation* employ algorithms computing beforehand the number of samples. Thus, according to the problem at hand, we suggest to use SMC tools computing *a priori* the number of samples if there is a need to parallelize, and, instead, using tools generating one sample at each iteration if it is more important to reduce the number of simulations.

Finally, our taxonomy points out the challenging task to deploy tools to perform unbounded verification of discrete- or continuous-time hybrid systems.

Chapter 3

An Efficient Algorithm for Multivariate Monte Carlo Estimation

Monte Carlo simulation is a computational technique that has been studied for decades and is widely used to analyze a huge range of problems in many different fields. Monte Carlo approach consists of building a model of the system and carrying out simulations on it, by sampling from a random variable that follows a certain probability distribution.

A typical usage of Monte Carlo-based algorithms is computing the estimation of an unknown quantity. The optimal $\mathcal{AA}$ algorithm of Dagum et al. [30] computes an (ϵ, δ)-approximation of the expected value of a univariate random variable Z distributed in $[0, 1]$. The $\mathcal{AA}$ algorithm is mainly used in Statistical Model Checkers that estimate the expected value of a random variable modelling some relevant system properties. However, there are many situations in which the property of interest depends on a multivariate random variable. This motivates the investigation of efficient algorithms to estimate the expected value of a *multivariate* random variable.

The main challenge of the Monte Carlo approach is minimizing the number of samples to be generated. A naïve use of $\mathcal{AA}$ to compute the expected value of a *multivariate* random variable Z, of size n, would require a number of samples given by the sum of the number of samples generated for each Z component. We propose the $\mathcal{MAA}$ algorithm, a multivariate version of $\mathcal{AA}$, to compute an (ϵ, δ)-approximation of the expected value of a *multivariate* random variable Z. $\mathcal{MAA}$ generates one sample of the input variable Z that is re-used to compute the approximated expected value of each Z component. Our approach assures the same guarantees of the $\mathcal{AA}$ algorithm, but is strictly faster than $\mathcal{AA}$, both in terms of number of samples to be generated and of CPU time elapsed.

The content of this section is related to the paper [93] that has been submitted for publication.

3.1 Introduction

Algorithms to estimate the expected value of a bounded random variable are among the enabling technologies for Statistical Model Checking (SMC), see, *e.g.*, [3, 107, 101, 23], APMC [74] and APD Analyser [82, 83].

Furthermore, the problem of estimating the expected value of a bounded random variable arises in many different fields, see, *e.g.*, [106, 39, 120, 85]. Such a problem is typically solved using Monte Carlo based approaches, such as those in [30, 56, 46, 47, 101, 139]. Basically, given a density function for a bounded univariate random variable X and positive real numbers ϵ and δ, most available algorithms compute an estimation $\tilde{\mu}$ ((ϵ, δ)-approximation) of the expected value μ of X such that, with probability at least $1 - \delta$ it holds that $|\tilde{\mu} - \mu| < \epsilon$.

Since our goal here is to minimise the number of samples to be generated, in the following we will focus on the $\mathcal{AA}$ algorithm from [30], since it is optimal (up to a constant factor).

Although $\mathcal{AA}$ has been presented for the univariate case, of course it can be used also to compute the expected value $\mu_Z = (\mu_{Z_1}, \dots, \mu_{Z_n})$ of a bounded *multivariate* random variable $Z = (Z_1, \dots, Z_n)$. If the marginal density functions p_{Z_i}, $i = 1, \dots n$, of Z are known we can sample from them. In this case the complexity of *multivariate* $\mathcal{AA}$, in terms of expected number of samples $\mathbb{E}[N_Z]$ to be generated, is the sum, for $i = 1, \dots n$, of the expected number of samples $\mathbb{E}[N_{Z_i}]$ generated to estimate the expected value of Z_i by using $\mathcal{AA}$ on p_{Z_i}. That is, $\mathbb{E}[N_Z] = \sum_{i=1}^{n} \mathbb{E}[N_{Z_i}]$. If only the joint density function p_Z is available we can of course still use $\mathcal{AA}$ to compute the expected value of each component Z_i of Z. In fact, sampling from p_{Z_i} can be performed by sampling from p_Z and then disregarding all components but the i-th one. However, in this case, the computational cost becomes $\mathbb{E}[N_Z] = n \sum_{i=1}^{n} \mathbb{E}[N_{Z_i}]$. This is n times the complexity of the case where marginal density functions are known.

The above described state of affairs motivates design of efficient algorithms to estimate the expected value of a bounded *multivariate* random variable when marginal density functions are not known.

3.1.1 Contributions

Building on the $\mathcal{AA}$ algorithm in [30], we present an efficient (as for the number of samples generated) algorithm, $\mathcal{MAA}$ (Section 3.4), to compute an (ϵ, δ)-approximation of the expected value $\mu_Z = (\mu_{Z_1}, \dots, \mu_{Z_n})$ of a *multivariate* random variable $Z = (Z_1, \dots, Z_n)$ when marginal probability densities are not known. Our contributions can be summarized as follows.

First, we show that for $i \in \{1, \dots, n\}$, $\mathcal{MAA}$ computes an (ϵ, δ)-approximation of the expected value of Z_i (*correctness*, Theorem 3.2).

Second, we give a Lower Bound (Theorem 3.3) to the complexity for any (ϵ, δ)-approximation algorithm to estimate the expected value of a multivariate random variable, when marginal probability density functions are not known. We show also that when marginal probability density functions are known the above described multivariate version of $\mathcal{AA}$ is *optimal* (Remark 3.3).

Third, we give an Upper Bound to the *complexity* (Theorem 3.2) of our algorithm, that is to the expected value of the number of samples generated by $\mathcal{MAA}$, namely $\mathbb{E}[N_Z^{MAA}]$.

Fourth, we give experimental results confirming that there exists an input s.t. the Upper Bound, we gave to the complexity of our algorithm $\mathcal{MAA}$, is attained.

Furthermore, we *experimentally* show that on a suitable input our $\mathcal{MAA}$ algorithm attains the above-mentioned Lower Bound.

3.1.2 Related Work

Our proposed algorithm aims at minimizing the number of samples to be generated. Accordingly, it is based on $\mathcal{AA}$ in [30], which is (up to a constant) optimal in this respect. Huber et al in [59] present an algorithm that is at least 2.5 times as fast as $\mathcal{AA}$. Both these two algorithms can be used to estimate the expected value of a multivariate random variable. However, when marginal probability density functions are unknown, their computational complexity (as for the number of samples) is given by the number of samples generated for each random variable component, multiplied by the number of components. Our algorithm, on the other hand, is strictly faster than $\mathcal{AA}$.

Estimation of the expected value of multivariate variables has been mostly studied in the context of formal verification tools supporting SMC [3]. Here are a few examples. MultiVeStA [107] computes an estimation of the expected value of a multivariate random variable by an algorithm based on the sequential Chow Robbins method [101]. Plasma-lab [23] implements a Monte Carlo probabilistic estimation algorithm working on multivariate distributions and based on the Chernoff-Hoeffding bound [56]. In a smart-grid setting, the Aggregated Power Demand (APD) analyzer Monte Carlo-based algorithm, in [83], computes the (ϵ, δ)-approximation of the APD probability distribution in Electric Distribution Networks (EDN). MultiVeStA, Plasma-Lab, and the APD analyzer generate samples at runtime. Basically, the aforementioned tools provide algorithms to compute the expected value of a multivariate random variable from its joint probability density in a SMC context. However, they do not provide a formal assessment of the computational complexity of the proposed estimation method. In this respect, we advance the state of the art by providing a general-purpose algorithm to compute the expected value of a multivariate random variable when marginal probability density functions are not known along with an assessment of its computational complexity.

Section 3.2 gives some background information. Section 3.3 recalls the univariate (ϵ, δ)-approximation algorithm, $\mathcal{AA}$, of [30]. Section 3.4 presents our multivariate (ϵ, δ)-approximation algorithm, whereas Section 3.5 shows its correctness and complexity and Section 3.6 shows the Lower Bound to the complexity. Finally, Section 3.7 gives experimental results.

3.2 Background

Let $Z = (Z_1, \dots, Z_n)$ be a n-dimensional random variable. Unless otherwise specified, we assume that each component Z_i ($i \in \{1, \dots, n\}$), is in the interval $[0, 1]$, has mean μ_{Z_i} and variance $\sigma^2_{Z_i}$. In the following we write $\mu_Z = (\mu_{Z_1}, \dots, \mu_{Z_n})$ and $\sigma^2_Z =$

$(\sigma^2_{Z_1}, \ldots, \sigma^2_{Z_n})$ to denote, respectively, the mean and the variance of Z. We denote with p_Z and with F_Z (Eq. 3.1), respectively, the *probability density function* and the *cumulative distribution function* of Z. We denote with Z_{-i} the vector consisting of all components of Z but the i-th one. That is, $Z_{-i} = (Z_1, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_n)$. Furthermore, we denote with $p_{Z_i}(z_i)$, for $i \in \{i, \ldots, n\}$, the density of the i-th component of Z ([20]), defined as in Eq. 3.2. The expected value μ_{Z_i} of the Z_i component of Z can be computed in two ways (see, *e.g.*, [10]): through a n-dimensional integral (Eq. 3.3), using the joint density function $p_Z(z) = p_Z(z_1, \ldots, z_n)$, or through a one-dimensional integral (Eq. 3.4)), using the density function $p_{Z_i}(z_i)$, in Eq. 3.2.

$$F_Z(z) = F_Z(z_1, \ldots, z_n) = \int_Z p_Z(x)\, dx =$$

$$\int_{Z_1} \cdots \int_{Z_n} p_Z(x_1, \ldots, x_n)\, dx_1 \ldots dx_n \tag{3.1}$$

$$p_{Z_i}(z_i) = \int_{Z_{-i}} p_Z(x)\, dx_{-i} \tag{3.2}$$

$$\mu_{Z_i} = \int_Z x_i p_Z(x)\, dx \tag{3.3}$$

$$\mu_{Z_i} = \int_{Z_i} x_i p_{Z_i}(x_i)\, dx_i \tag{3.4}$$

Notation 3.1. *Monte Carlo based algorithms for estimation of the expected value of a univariate random variable, typically, use the number of samples to be generated as a complexity measure, that is: each sample has computational cost* 1*. Along the same line, in the following, when sampling from a multivariate random variable Z of dimension n, the computational cost of sampling will be n.*

3.3 Univariate Monte Carlo Approximation Algorithm AA

For the sake of completeness (and self-containment), in this section we summarize the main steps of the $\mathcal{AA}$ algorithm of [30], on the basis of which our algorithm is built.

Given a univariate random variable Z distributed in $[0, 1]$, the Monte Carlo-based approximation algorithm $\mathcal{AA}$ computes an (ϵ, δ)-approximation $\tilde{\mu}_Z$ of the expected value, μ_Z, of Z. $\mathcal{AA}$ works with no a priori information about the probability distribution of Z. To decide the number of samples needed to estimate $\tilde{\mu}_Z$, $\mathcal{AA}$ computes an initial approximation of μ_Z, obtained by running a *Stopping Rule Algorithm* [30], and an initial estimate of the variance σ^2_Z.

In particular, given the parameters ϵ and δ, the following values are defined:

$$\lambda = (e - 2) \approx .72 \tag{3.5}$$

$$\Upsilon = 4\lambda ln(2/\delta)/\epsilon^2 \tag{3.6}$$

$$\rho_Z = max\{\sigma^2_Z, \epsilon\mu_Z\} \tag{3.7}$$

The above mentioned values are used in $\mathcal{AA}$ to return an estimate of μ_Z after a number N_Z of experiments have been performed. The $\mathcal{AA}$ *Theorem* and the *Lower*

Bound Theorem in [30] provide the optimal number of experiments N_Z, with respect to a constant parameter c. We summarize this result in Theorem 3.1.

Theorem 3.1. *Let Z be any random variable distributed in $[0,1]$, let $\mu_Z = \mathbb{E}[Z] > 0$ be the mean of Z, σ_Z^2 the variance of Z, and let ρ_Z be equal to $max\{\sigma_Z^2, \epsilon\mu_Z\}$. Let $\tilde{\mu}_Z$ be the approximation of μ_Z computed by $\mathcal{AA}$ and let N_Z be the number of experiments run by $\mathcal{AA}$ with respect to Z on input parameters ϵ and δ. Then:*

(1) $\mathbb{P}[\mu_Z(1-\epsilon) \leq \tilde{\mu}_Z \leq \mu_Z(1+\epsilon)] \geq 1-\delta$

(2) $\mathbb{E}[N_Z] = \Theta(\Upsilon\rho_Z/\mu_Z^2)$

In Theorem 3.1, the Θ notation (see [29]) is used to state that $\mathbb{E}[N_Z]$ is at least $c_1\Upsilon\rho_Z/\mu_Z^2$ and at most $c_2\Upsilon\rho_Z/\mu_Z^2$, for some constants c_1 and c_2. In the following, we summarize in Algorithm 1 the main steps of $\mathcal{AA}$.

```
Input: $\epsilon, \delta, p_Z()$
Output: $\tilde{\mu}_Z$
$\Upsilon_2 \leftarrow 2(1+\sqrt{\epsilon})(1+2\sqrt{\epsilon})(1+\ln(3/2)/\ln(2/\delta))\Upsilon$
// $\Upsilon_2 \approx 2\Upsilon$ for small $\epsilon$ and $\delta$
$\hat{\mu}_Z \leftarrow$ SR($min(1/2, \sqrt{\epsilon}), \delta/3, p_Z()$)
// see function SR in Algorithm 2
$\hat{\rho}_Z \leftarrow$ EV($\hat{\mu}_Z, \epsilon, \Upsilon_2, p_Z()$)
// see function EV in Algorithm 3
$\tilde{\mu}_Z \leftarrow$ EM($\hat{\mu}_Z, \hat{\rho}_Z, \Upsilon_2, p_Z()$)
// see function EM in Algorithm 4
```

Algorithm 1: Approximation Algorithm $\mathcal{AA}$

In order to ease the description of the functions used by the Algorithm 1, in the following we denote with *sample* the function that generates samples of the univariate random variable Z. That is, if p_Z is the probability density function of Z, then *sample*(p_Z) returns a sample of Z according to the density function p_Z.

Algorithms 2 to 4 show the pseudo-code of the functions used in Algorithm 1.

```
function SR()
  Input: $\epsilon, \delta, p_Z()$
  Output: $\hat{\mu}_Z$
  $\Upsilon_1 \leftarrow 1 + (1+\epsilon)\Upsilon$;
  $N \leftarrow 0$; $S \leftarrow 0$;
  while $S < \Upsilon_1$ do
    $Z_N \leftarrow sample(p_Z())$;
    $N \leftarrow N + 1$;
    $S \leftarrow S + Z_N$;
    $\hat{\mu}_Z \leftarrow \Upsilon_1$/N
```

Algorithm 2: The SR (StoppingRule) function computes an estimate $\hat{\mu}_Z$ of μ_Z using the value Z_N, obtained through *sample*$(p_Z())$ and the input parameters (ϵ, δ).

```
function EV()
  Input: $\hat{\mu}_Z, \epsilon, \Upsilon_2, p_Z()$
  Output: $\hat{\rho}_Z$
  $N \leftarrow \Upsilon_2\epsilon/\hat{\mu}_Z$; $S \leftarrow 0$;
  for i $\leftarrow$ 1 to N do
    $X_{2i-1} \leftarrow sample(p_Z())$;
    $X_{2i} \leftarrow sample(p_Z())$;
    $S \leftarrow S + (X_{2i-1} - X_{2i})^2/2$
  $\hat{\rho}_Z \leftarrow max(S/N, \epsilon\hat{\mu}_Z)$
```

Algorithm 3: The EV (EstimateVariance) function computes an estimate $\hat{\rho}_Z$ that is within a constant factor ρ, with probability at least $1 - \delta$.

```
function EM()
  Input: $\hat{\mu}_Z, \hat{\rho}_Z, \Upsilon_2, p_Z()$
  Output: $\tilde{\mu}_Z$
  N $\leftarrow \Upsilon_2\hat{\rho}_Z/\hat{\mu}_Z^2$; S $\leftarrow$ 0;
  for i $\leftarrow$ 1 to N do
    $X_i \leftarrow sample(p_Z())$;
    S $\leftarrow S + X_i$
  $\tilde{\mu}_Z \leftarrow S/N$
```

Algorithm 4: The EM (EstimateMean) function computes the needed number of samples N to be generated with the aim to compute an (ϵ, δ)-estimate $\tilde{\mu}_Z$ of μ_Z.

3.4 A Multivariate Monte Carlo Approximation Algorithm MAA

Let $Z = (Z_1, \dots, Z_n)$ be a multivariate random variable distributed according to a density function p_Z and ϵ and δ be positive real numbers. Through a Monte Carlo based approach we would like to compute an estimation $(\tilde{\mu}_{Z_1}, \dots \tilde{\mu}_{Z_n})$ ((ϵ, δ)-approximation) of the expected value $\mu_Z = (\mu_{Z_1}, \dots, \mu_{Z_n})$ of $Z = (Z_1, \dots, Z_n)$, such that with probability at least $1 - \delta$ it holds that, for all $i = 1, \dots n$, $|\tilde{\mu}_{Z_i} - \mu_{Z_i}| < \epsilon$.

First, we note that *any* algorithm, say $\mathcal{BB}$, computing an (ϵ, δ)-approximation $\tilde{\mu}_Z$ of the expected value μ_Z of a univariate random variable Z can be used to estimate the expected value of a multivariate random variable. In this respect, two different scenarios are possible, as discussed in the following remarks.

Remark 3.1. *The marginal density functions $p_{Z_1}, \dots, p_{Z_n}$ are unknown. In this case $\mathcal{BB}$ can compute μ_{Z_i}, for $i \in \{1, \dots, n\}$, by generating samples of Z, through p_Z, and then projecting on the i-th component, accordingly to Eq. 3.3.*

Let N_{Z_i} be the number of samplesgenerated by $\mathcal{BB}$ to estimate μ_{Z_i} from the univariate variable Z_i. Then, the cost of estimating μ_{Z_i} from the multivariate variable Z with $\mathcal{BB}$ is $n\mathbb{E}[N_{Z_i}]$, since each sample has a cost of n. Then, the expected value for the number N_Z of samples to be generated to estimate μ_Z through $\mathcal{BB}$ is:

$$\mathbb{E}[N_Z] = n \sum_{i=1}^{n} \mathbb{E}[N_{Z_i}] \tag{3.8}$$

Remark 3.2. *The marginal density functions $p_{Z_1}, \ldots, p_{Z_n}$ are known. In this case $\mathcal{BB}$ can compute μ_{Z_i}, for $i \in \{1, \ldots, n\}$, by generating N_{Z_i} samplesof Z_i, through p_{Z_i}, accordingly to Eq. 3.4. Then, as in this case each sample has cost 1, the expected value for the number N_Z of samples to be generated to estimate μ_Z through $\mathcal{BB}$ is:*

$$\mathbb{E}[N_Z] = \sum_{i=1}^{n} \mathbb{E}[N_{Z_i}] \tag{3.9}$$

The approaches outlined in Remarks 3.1 and 3.2 solve the estimation problem, however they do not provide any guarantee about computational optimality (as for samples to be generated). Of course, if we start from an algorithm which is not optimal for the univariate case, then there is no hope for optimality in the multivariate case. Accordingly, in the following we focus on algorithms that are optimal at least for the univariate case.

A widely used one in our setting is the $\mathcal{AA}$ algorithm (see Section 3.3).Given a univariate random variable Z, $\mathcal{AA}$ computes an (ϵ, δ) approximation $\tilde{\mu}_Z$ of the expected value μ_Z of Z, with no a priori information about the probability distribution of Z. By abuse of language, in a multivariate setting, in the following we will refer with $\mathcal{AA}$ to the algorithm outlined in Remark 3.2 when $\mathcal{BB}$ is $\mathcal{AA}$.

Unfortunately, starting from an optimal algorithm for the univariate case, as $\mathcal{AA}$, does not guarantee optimality for the multivariate case. In fact, as we will see in Theorem 3.2, even replacing $\mathcal{AA}$ for $\mathcal{BB}$, the algorithm outlined in Remark 3.1 is not optimal. On the other hand, as we will see in Section 3.6, the algorithm in Remark 3.2 with $\mathcal{AA}$ is optimal. Accordingly, by abuse of language, in a multivariate setting, we will refer to it also as $\mathcal{AA}$.

On the basis of the above considerations, in the rest of the sections we present our $\mathcal{AA}$ based algorithm for the estimation of the expected value of a multivariate random variable.

Our algorithm, $\mathcal{MAA}$, shown in Algorithm 5, is not optimal but is more efficient than $\mathcal{AA}$ when $n \geq 2$. When $n = 1$, $\mathcal{MAA}$ corresponds to $\mathcal{AA}$.

```
function $\mathcal{MAA}$()
  Input: $\epsilon, \delta, p_Z$
  // $0 < \epsilon \leq 1$ and $0 < \delta \leq 1$
  Output: $\tilde{\mu}_Z = (\tilde{\mu}_{Z_1}, \ldots, \tilde{\mu}_{Z_n})$
  $\lambda \leftarrow e - 2$; $\Upsilon \leftarrow \frac{4\lambda}{\epsilon^2} \ln(\frac{2}{\delta})$;
  $\Upsilon_2 \leftarrow 2(1 + \sqrt{\epsilon})(1 + 2\sqrt{\epsilon})(1 + \ln(\frac{3}{2}) / \ln(\frac{2}{\delta}))\Upsilon$;
  $\hat{\mu}_Z \leftarrow \mathrm{MSR}(min(\frac{1}{2}, \sqrt{\epsilon}), \frac{\delta}{3}, p_Z, \Upsilon)$; // Algorithm 6
  $\hat{\rho}_Z \leftarrow \mathrm{MEV}(\hat{\mu}_Z, \epsilon, \Upsilon_2, p_Z)$; // Algorithm 7
  $\tilde{\mu}_Z \leftarrow \mathrm{MEM}(\hat{\mu}_Z, \hat{\rho}_Z, \Upsilon_2, p_Z)$; // Algorithm 8
```

Algorithm 5: Multivariate Approximation Algorithm $\mathcal{MAA}$.

In the functions used by the Algorithm 5, we denote with *sample* the sample generator function (as already done for $\mathcal{AA}$ in Section 3.3) that, given the probability density function p_Z of a multivariate random variable Z, returns a sample of Z according to p_Z. The function *sample* can be implemented by one of the available C libraries, such as the GNU Scientific Library *gsl* (see [42]).

The functions used in $\mathcal{MAA}$ (Algorithm 5) are: the *MSR* (*MultiStoppingRule*) function, shown in Algorithm 6 and based on the *SR* (*StoppingRule*) function in Al-

```
function MSR()
  Input: $\epsilon, \delta, p_Z, \Upsilon$
  // $0 < \epsilon < 1$ and $\delta > 0$
  Output: $\tilde{\mu}_Z = (\tilde{\mu}_{Z_1}, \ldots, \tilde{\mu}_{Z_n})$
  $\Upsilon_1 \leftarrow 1 + (1+\epsilon)\Upsilon$
  for $i = 1$ to $n$ do
    $N_i \leftarrow 0$; $S_i \leftarrow 0$;
  update $\leftarrow 0$;
  while update$==1$ do
    update $\leftarrow 0$;
    $Z \leftarrow sample(p_Z)$;
    for $i = 1$ to $n$ do
      if $S_i < \Upsilon_1$ then
        $N_i \leftarrow N_i + 1$; $S_i \leftarrow S_i + Z_i$;
        update $\leftarrow 1$;
  for $i = 1$ to $n$ do
    $\hat{\mu}_{Z_i} \leftarrow \frac{\Upsilon_1}{N_i}$
```

Algorithm 6: *MultiStoppingRule* Function, implementing the Stopping Rule Algorithm for multivariate random variables.

```
function MEV()
  Input: $\hat{\mu}_Z, \epsilon, \Upsilon_2, p_Z$
  Output: $\hat{\rho}_Z = (\hat{\rho}_{Z_1}, \ldots, \hat{\rho}_{Z_n})$
  for i $\leftarrow$ 1 to $n$ do
    $N_i \leftarrow \Upsilon_2 \frac{\epsilon}{\hat{\mu}_{Z_i}}$;
    $S_i \leftarrow 0$;
  $N_{max} \leftarrow$ Max(N); // see (3.10)
  for k $\leftarrow$ 1 to $N_{max}$ do
    $X^{2k-1} \leftarrow sample(p_Z)$;
    $X^{2k} \leftarrow sample(p_Z)$;
    for i $\leftarrow$ 1 to $n$ do
      if $k < N_i$ then
        $S_i \leftarrow S_i + \frac{({X^{2k-1}}_i - {X^{2k}}_i)^2}{2}$;
  for i $\leftarrow$ 1 to $n$ do
    $\hat{\rho}_{Z_i} \leftarrow max(\frac{S_i}{N_i}, \epsilon\hat{\mu}_{Z_i})$;
```

Algorithm 7: *MultiEstimateVariance* function, computing the approximated value $\hat{\rho}_{Z_i}$ of each ρ_{Z_i}, for $i \in \{1, \ldots, n\}$.

gorithm 2; the *MEV* (*MultiEstimateVariance*) function, shown in Algorithm 7 and based on the *EV* (*EstimateVariance*) function in Algorithm 3; the *MEM* (*MultiEstimateMean*) function, shown in Algorithm 8 and based on the *EM* (*EstimateMean*) function in Algorithm 4. The *MEV* and *MEM* functions are used to compute, respectively, the estimated values of variance and mean of the multivariate random variable Z.

In Algorithm 6, the *MSR* algorithm computes an estimate $\hat{\mu}_Z$ of μ_Z by using the sample Z, obtained by $sample(p_Z)$, and the input parameters ϵ, δ. As N and S are vectors of the same size n of Z, their values are computed by introducing a guard variable, *update*, to check if the condition $S_i < \Upsilon_1$ is verified, for $i \in \{1, \ldots, n\}$.

Furthermore, we observe that in Algorithm 7 and Algorithm 8 instead of iterat-

```
function MEM()
  Input: $\hat{\mu}_Z, \hat{\rho}_Z, \Upsilon_2, p_Z$
  Output: $\tilde{\mu}_Z = (\tilde{\mu}_{Z_1}, \dots, \tilde{\mu}_{Z_n})$
  for i ← 1 to n do
    $N_i \leftarrow \Upsilon_2 \frac{\hat{\rho}_{Z_i}}{\hat{\mu}^2_{Z_i}}$;
    $S_i \leftarrow 0$;
  $N_{max} \leftarrow$ Max(N); // see (3.10)
  for k ← 1 to $N_{max}$ do
    $X^k \leftarrow sample(p_Z)$;
    for i ← 1 to n do
      if $k < N_i$ then
        $S_i \leftarrow S_i + {X^k}_i$;
  for i ← 1 to n do
    $\tilde{\mu}_{Z_i} \leftarrow \frac{S_i}{N_i}$;
```

Algorithm 8: $MultiEstimateMean$ function, computing the approximated value $\tilde{\mu}_{Z_i}$ of each μ_{Z_i}, for $i \in \{1, \dots, n\}$.

ing on the number of samplesN_i needed for each Z_i component, the iteration is done on the maximum number of samples N_{max}, obtained by calling the Max function over n-dimensional vectors V, defined as follows:

$$Max(V) = max(V_1, \dots, V_n). \tag{3.10}$$

Inside the loop on N_{max}, the generated samples of Z, namely X^{2k-1}, X^{2k} in Algorithm 7 and X^k in Algorithm 8, are used to compute, respectively, each $\hat{\rho}_Z$ and $\tilde{\mu}_Z$ component. In this way the overall complexity of the functions is reduced.

3.5 Correctness and Complexity of MAA

In this section, we discuss (Theorem 3.2) correctness and computational complexity of $\mathcal{MAA}$ (Algorithm 5). In the following let $i \in \{1, \dots n\}$, $Z = (Z_1, \dots, Z_n)$ be a multivariate random variable ($Z_i \in [0,1]$), ϵ, δ be real numbers in $(0,1)$, $\mu_Z = (\mu_{Z_1}, \dots, \mu_{Z_n})$ be the expected value of Z (with $\mu_{Z_i} > 0$), ${\sigma^2}_Z = ({\sigma^2}_{Z_1}, \dots, {\sigma^2}_{Z_n})$ be the variance of Z, and, finally, let $\rho_Z = (\rho_{Z_1}, \dots, \rho_{Z_n})$ with $\rho_{Z_i} = max({\sigma^2}_{Z_i}, \epsilon\mu_{Z_i})$.

Let ${N_Z}^{MAA}$ be the number of samples generated by $\mathcal{MAA}$ with respect to Z, on input parameters ϵ and δ. That is, ${N_Z}^{MAA}$ is the number of times function $sample()$ is executed in Algorithm 5 (*i.e.*, Algorithms 6 to 8) multiplied by n since each call to function $sample()$ generates n samples.

Let ${N_Z}^{AA}$ be the number of samples generated by $\mathcal{AA}$ from p_Z, as in Remark 3.1 and let N_{Z_i} be the number of samplesgenerated by $\mathcal{MAA}$ with respect to Z_i. That is, N_{Z_i} is the number of times variable S_i is updated in Algorithm 5 (*i.e.*, Algorithms 6 to 8).

Proposition 3.1 shows that $\mathcal{MAA}$ is strictly faster than $\mathcal{AA}$ when $n \geq 2$ and marginal distributions are unknown.

Proposition 3.1. *If* $n \geq 2$ *then* $\mathbb{E}[{N_Z}^{MAA}] < \mathbb{E}[{N_Z}^{AA}]$.

Proof. Let us consider Algorithm 6 (part of Algorithm 5). Taking into account that $Z_i \in [0,1]$ we have that the `while` loop of Algorithm 6 will be executed at least Υ_1 times since at each iteration S_i can be incremented at most by 1. Thus, we have that $N_{Z_i} \geq \Upsilon_1 > \Upsilon$. Thus we can write: $N_{Z_i} = \Upsilon_1 + \tilde{N}_{Z_i}$ with $\tilde{N}_{Z_i} \geq 0$. Taking into account that ${N_Z}^{AA} = n\sum_{i=1}^{n} N_{Z_i}$ and $n \geq 2$ we have: $\mathbb{E}[{N_Z}^{MAA}] = \mathbb{E}[n \max\{N_{Z_i} | i = 1, \dots, n\}] = \mathbb{E}[n \max\{\Upsilon_1 + \tilde{N}_{Z_i} | i = 1, \dots, n\}] = \mathbb{E}[n[\Upsilon_1 + \max\{\tilde{N}_{Z_i} | i = 1, \dots, n\}] \leq \mathbb{E}[n[\Upsilon_1 + \sum_{i=1}^{n} \tilde{N}_{Z_i}]] < \mathbb{E}[n[n\Upsilon_1 + \sum_{i=1}^{n} \tilde{N}_{Z_i}]] = \mathbb{E}[n\sum_{i=1}^{n}[\Upsilon_1 + \tilde{N}_{Z_i}]] = \mathbb{E}[n\sum_{i=1}^{n} N_{Z_i}] = \mathbb{E}[{N_Z}^{AA}]$.

□

From Proposition 3.1 follows that, when marginal density functions are unknown, $\mathcal{MAA}$ is always faster than $\mathcal{AA}$, that is $\frac{\mathbb{E}[{N_Z}^{AA}]}{\mathbb{E}[{N_Z}^{MAA}]} > 1$. Our experimental results in Sections 3.7.4 and 3.7.5 show that $\mathcal{MAA}$ can be as much as n times faster than $\mathcal{AA}$, namely $\frac{\mathbb{E}[{N_Z}^{AA}]}{\mathbb{E}[{N_Z}^{MAA}]} \approx n$.

Remark 3.3. *Given a multivariate random variable $Z = (Z_1, \dots, Z_n)$, let us assume that for each Z_i $(i \in \{1, \dots, n\})$, the univariate density function p_{Z_i} is known. In this case, there is no advantage, in terms of number of samples, in using $\mathcal{MAA}$ instead of $\mathcal{AA}$. In fact, in this case, (as in Remark 3.2): we have that ${N_Z}^{AA} = \sum_{i=1}^{n} N_{Z_i}$. Thus: $\mathbb{E}[{N_Z}^{AA}] = \mathbb{E}[\sum_{i=1}^{n} N_{Z_i}] \leq \mathbb{E}[n \max\{N_{Z_i} | i = 1, \dots, n\}] = n\mathbb{E}[\max\{N_{Z_i} | i = 1, \dots, n\}] = \mathbb{E}[{N_Z}^{MAA}]$.*

The above shows that, when marginal density functions for Z are known, (the expected values of the) computational complexity of $\mathcal{AA}$ is not greater than that of $\mathcal{MAA}$.

In Sections 3.7.2 and 3.7.3 we show that there exist density functions for Z for which $\mathbb{E}[{N_Z}^{AA}]$ is strictly less than $\mathbb{E}[{N_Z}^{MAA}]$.

Theorem 3.2 below provides the analogous for $\mathcal{MAA}$ of the $\mathcal{AA}$-Theorem in [30] (Υ is defined as in Algorithm 5).

Theorem 3.2. *Let $\tilde{\mu}_Z = (\tilde{\mu}_{Z_1}, \dots, \tilde{\mu}_{Z_n})$ be the approximation of μ_Z computed by $\mathcal{MAA}$. Then:*

(1) $\mathbb{P}[\mu_{Z_i}(1-\epsilon) \leq \tilde{\mu}_{Z_i} \leq \mu_{Z_i}(1+\epsilon)] \geq 1-\delta$, *for* $i \in \{1, \dots, n\}$

(2) There exist a universal constants c such that:

$$\mathbb{P}[{N_Z}^{MAA} \leq n\, c\, \Upsilon \max\{\tfrac{\rho_{Z_i}}{{\mu^2}_{Z_i}} | i = 1, \dots, n\}] \geq (1-\delta)^n,$$

(3) There exist a universal constant c such that: $\mathbb{E}[{N_Z}^{MAA}] \leq n\, c\, \Upsilon \sum_{i=1}^{n} \frac{\rho_{Z_i}}{{\mu^2}_{Z_i}}$.

Proof. **Item (1)** (Correctness). Given a multivariate random variable Z, whose probability density function p_Z is known, in general the marginal density function p_{Z_i} of each Z_i component, for $i \in \{1, \dots, n\}$, cannot be computed in an analytical way. However, each p_{Z_i} can be computed through the multivariate density function p_Z by projecting on the i-th component, as in Section 3.2.

As stated in Remark 3.1, $\mathcal{AA}$ can be used to compute the approximated expected value $\tilde{\mu}_{Z_i}$, for $i \in \{1, \dots, n\}$, by generating samplesof Z_i through p_Z. Then, both $\mathcal{AA}$ and $\mathcal{MAA}$ compute the approximated expected value $\tilde{\mu}_{Z_i}$, for $i \in \{1, \dots, n\}$, through p_Z. We know from the *item 1* of the $\mathcal{AA}$-Theorem in [30] that Item (1) holds component-wise for $\mathcal{AA}$. This proves the correctness of $\mathcal{MAA}$.

Item (2)(Probabilistic complexity). From the *item 2* of the $\mathcal{AA}$-Theorem in [30], we have:
$\mathbb{P}[{N_Z}^{MAA} \leq n\ c\ \Upsilon \max\{\frac{\rho_{Z_i}}{{\mu^2}_{Z_i}} | i = 1, \dots, n\}] = \mathbb{P}[n \max\{N_{Z_i} | i = 1, \dots, n\} \leq n\ c\ \Upsilon \max\{\frac{\rho_{Z_i}}{{\mu^2}_{Z_i}} | i = 1, \dots, n\}] = \mathbb{P}[\max\{N_{Z_i} | i = 1, \dots, n\} \leq c\ \Upsilon \max\{\frac{\rho_{Z_i}}{{\mu^2}_{Z_i}} | i = 1, \dots, n\}] \geq \mathbb{P}[\forall i \in \{1, \dots n\}[N_{Z_i} \leq c\Upsilon \frac{\rho_{Z_i}}{{\mu^2}_{Z_i}}]] = \prod_{i=1}^{n} \mathbb{P}[N_{Z_i} \leq c\Upsilon \frac{\rho_{Z_i}}{{\mu^2}_{Z_i}}] \geq (1 - \delta)^n$.

Item (3)(Expected value of complexity). As for $\mathcal{AA}$, the complexity of $\mathcal{MAA}$ is defined in terms of the number N_Z of samples of Z to be generated. We showed in equation (3.8) that the complexity of $\mathcal{AA}$ to compute $\tilde{\mu}_Z$ is given by $n \sum_{i=1}^{n} \mathbb{E}[N_{Z_i}]$.

In order to compute $\tilde{\mu}_Z = (\tilde{\mu}_{Z_1}, \dots, \tilde{\mu}_{Z_n})$, $\mathcal{MAA}$ stops when no Z_i requires further sampling. Thus, for the expected value $\mathbb{E}[{N_Z}^{MAA}]$ we have: $\mathbb{E}[{N_Z}^{MAA}] = n\mathbb{E}[\max\{N_{Z_i} | i = 1, \dots, n\}] \leq n\mathbb{E}[\sum_{i=1}^{n} N_{Z_i}] = n \sum_{i=1}^{n} \mathbb{E}[N_{Z_i}] \leq n \sum_{i=1}^{n} (c\Upsilon \frac{\rho_{Z_i}}{{\mu^2}_{Z_i}}) = n\ c\ \Upsilon\ \sum_{i=1}^{n} \frac{\rho_{Z_i}}{{\mu^2}_{Z_i}}$.

$\square$

3.6 Lower Bound to MAA complexity

In this section, we show that when marginal probability density functions are known the straightforward multivariate $\mathcal{AA}$ (Remark 3.1) is optimal, whereas when marginal probability density functions are not known, our $\mathcal{MAA}$ is more efficient than $\mathcal{AA}$ (see Theorem 3.3). Thus, given a multivariate random variable $Z = (Z_1, \dots, Z_n)$, with $Z_i \in [0, 1]$, for $i \in \{1, \dots, n\}$, we define a *lower bound* to the number of samples for any (ϵ, δ)-approximation algorithm to estimate μ_Z.

To formalize this lower bound, we first define the class of algorithms we consider. Let Z be a multivariate random variable whose density function is p_Z, mean is $\mu_Z = (\mu_{Z_1}, \dots, \mu_{Z_n})$ (with $\mu_{Z_i} > 0$, for $i \in \{1, \dots, n\}$), and variance is ${\sigma^2}_Z = ({\sigma^2}_{Z_1}, \dots, {\sigma^2}_{Z_n})$. Let $\rho_Z = (\rho_{Z_1}, \dots, \rho_{Z_n})$, where $\rho_{Z_i} = max({\sigma^2}_{Z_i}, \epsilon\mu_{Z_i})$, for $i \in \{1, \dots, n\}$. Let $\mathcal{BB}$ be any algorithm that, on input (ϵ, δ), works as follows. $\mathcal{BB}$ takes as input a sequence $Z_1, Z_2, \dots$ of multivariate random variables i.i.d. according to p_Z. At any step k, $\mathcal{BB}$ receives one Z_k and performs an experiment. The execution time of $\mathcal{BB}$ is the number of experiments $\mathcal{BB}$ runs. $\mathcal{BB}$ stops running experiments based on any given criteria. When $\mathcal{BB}$ stops, it returns an (ϵ, δ)-approximation $\tilde{\mu}_Z = (\tilde{\mu}_{Z_1}, \dots, \tilde{\mu}_{Z_n})$ of μ_Z. Note that $\mathcal{AA}$ as well as $\mathcal{MAA}$ fits in the above described class of algorithms.

Theorem 3.3 provides a lower bound to the number of experiments $\mathcal{BB}$ must perform (Υ below is as in Algorithm 5).

Theorem 3.3. *Let $\mathcal{BB}$ be any algorithm that works as described above on input (ϵ, δ) and on a multivariate random variable Z. Let μ_Z and ${\sigma^2}_Z$ be, respectively, the mean and the variance of Z. Let $\tilde{\mu}_Z$ be the approximation computed by $\mathcal{BB}$ and let ${N_Z}^{BB}$ be the number of experiments performed by $\mathcal{BB}$. Let us suppose that for all Z with $\mu_Z > 0$, the following properties hold:*

(1) $\mathbb{E}[{N_Z}^{BB}] < \infty$

(2) $\mathbb{P}[\mu_Z(1 - \epsilon) \leq \tilde{\mu}_Z \leq \mu_Z(1 + \epsilon)] > 1 - \delta$.

Then, there exist a positive universal constant c s.t. for $i \in \{1, \ldots, n\}$, and, for all Z:

(1) if the marginal density functions $p_{Z_1}, \ldots, p_{Z_n}$ are known, then $\mathbb{E}[N_Z{}^{BB}] \geq c\Upsilon \sum_{i=1}^{n} \frac{\rho_{Z_i}}{\mu_{Z_i}^2}$;

(2) if the marginal density functions $p_{Z_1}, \ldots, p_{Z_n}$ are unknown, then $\mathbb{E}[N_Z{}^{BB}] \geq nc\Upsilon \max\{\frac{\rho_{Z_i}}{\mu_{Z_i}^2} | i = 1, \ldots, n\}$.

Proof. From the *Lower Bound Theorem* of $\mathcal{AA}$ in [30] we know that there exist a positive real number c such that for all $i \in \{1, \ldots n\}$ and for any distribution p_{Z_i} we have $\mathbb{E}[N_i{}^{BB}] \geq c\Upsilon \frac{\rho_{Z_i}}{\mu_{Z_i}^2}$.

As for **Item (1)**, that matches Remark 3.2, we have that:
$\mathbb{E}[N_Z{}^{BB}] = \sum_{i=1}^{n} \mathbb{E}[N_i{}^{BB}] \geq \sum_{i=1}^{n} c\Upsilon \frac{\rho_{Z_i}}{\mu_{Z_i}^2} = c\Upsilon \sum_{i=1}^{n} \frac{\rho_{Z_i}}{\mu_{Z_i}^2}$.

As for **Item (2)**, that matches Remark 3.1, we have that:
$\mathbb{E}[N_Z{}^{BB}] = n\mathbb{E}[\max\{N_i{}^{BB} | i = 1, \ldots, n\}] \geq n \max\{\mathbb{E}[N_i{}^{BB}] | i = 1, \ldots, n\} = n \max\{c\Upsilon \frac{\rho_{Z_i}}{\mu_{Z_i}^2} | i = 1, \ldots, n\} = nc\Upsilon \max\{\frac{\rho_{Z_i}}{\mu_{Z_i}^2} | i = 1, \ldots, n\}$.

□

Remark 3.4. *From Item (3) of Theorem 3.2 and Item (2) of Theorem 3.3 we have:* $nc\Upsilon \max\{\frac{\rho_{Z_i}}{\mu_{Z_i}^2} | i = 1, \ldots, n\} \leq \mathbb{E}[N_Z{}^{MAA}] \leq nc\Upsilon \sum_{i=1}^{n} \frac{\rho_{Z_i}}{\mu^2{}_{Z_i}}$.

Thus, although from Proposition 3.1 we have that $\mathcal{MAA}$ is strictly faster than $\mathcal{AA}$, we cannot conclude that $\mathcal{MAA}$ is optimal, unless $n = 1$ which is the case studied in [30]. In Section 3.7.6 we show that both the Lower and the Upper Bounds to $\mathbb{E}[N_Z{}^{MAA}]$ can be attained, at some input instances.

3.7 Experimental Results

In this section we present experimental results to evaluate the complexity of $\mathcal{MAA}$ Algorithm 5 in terms of number of samples to be generated and of CPU time. We present two case studies.

The first one (see Section 3.7.2 and Section 3.7.3) matches the Remark 3.3, and aims at showing that when the marginal density functions *are known* one should use $\mathcal{AA}$ through the algorithm outlined in Remark 3.2 (just $\mathcal{AA}$ in the following) since it is optimal (Section 3.6).

The second one (see Section 3.7.4 and Section 3.7.5) matches the result of Proposition 3.1 and aims at showing that when the marginal density functions *are unknown* one should use our $\mathcal{MAA}$ algorithm since it is faster than $\mathcal{AA}$.

Furthermore, we perform a third experiment (see Section 3.7.6) that matches the result in Remark 3.4, that is both the Lower Bound and the Upper Bound to the expected value $\mathbb{E}[N_Z^{MAA}]$ can be attained.

3.7.1 Computational Environment

We implemented (multivariate) $\mathcal{AA}$ and our $\mathcal{MAA}$ in C. All experiments have been carried out on a Cluster consisting of three nodes: one node is an Intel(R)

Celeron(R) with a 2.27 GHz CPU, 2 cores and 4 GB RAM whereas the other two nodes are both Intel(R) Xeon(R) with two 2.83 GHz CPU, 16 cores and 8 GB RAM.

3.7.2 Known Marginal Density Functions: Case Study

Let $Z = (Z_1, \ldots, Z_n)$ be a multivariate random variable s.t. each Z_i component, for $i \in \{1, \ldots, n\}$, is distributed according to a known distribution. In our experiments we choose the *Beta* distribution, that is a continuous probability distribution defined on the interval $[0, 1]$. Then, $Z_i \sim \mathit{Beta}(\alpha_i, \beta_i)$, where $\alpha_i > 0$ and $\beta_i > 0$, and $p_{Z_i}()$ is the density function of the *Beta Distribution* (*e.g.*, see [66]). For $i \in \{1, \ldots, n\}$, the density function p_{Z_i} is used to generate samplesof Z_i. More details about the *Beta* distribution are given in Section 3.7.2.1.

3.7.2.1 Beta Distribution Parameters

Let Z be a random variable following the Beta distribution [66], that is $Z \sim \mathit{Beta}(\alpha, \beta)$, where $\alpha > 0$ and $\beta > 0$. From the properties of the *Beta* distribution, we know that mean μ_Z and variance $\sigma^2{}_Z$ are computed as in equations (3.11) and (3.12).

$$\mu_Z = \frac{\alpha}{\alpha + \beta} \tag{3.11}$$

$$\sigma^2{}_Z = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \tag{3.12}$$

We want to select $Z_1, \ldots, Z_n$ univariate random variables s.t. their means $\mu_1, \ldots, \mu_n$ take values in the interval $(0, 1)$. Accordingly, let us choose $a = 0.1$ and $b = 0.9$. Then, for $i \in \{1, \ldots, n\}$, it holds:

$$\mu_i = a + \frac{b - a}{n - 1}(i - 1) \tag{3.13}$$

From equation (3.13) it follows that: if $i = 1$ then $\mu_i = 0.1$, if $i = n$ then $\mu_i = 0.9$. The parameters α_i and β_i can be defined as functions of μ_i, as follows:

$$\alpha_i = \begin{cases} 2 & \text{if } \mu_i \leq 1/2 \\ 2(\frac{\mu_i}{1-\mu_i}) & \text{if } \mu_i > 1/2 \end{cases} \tag{3.14}$$

$$\beta_i = \alpha_i(\frac{1 - \mu_i}{\mu_i}) \tag{3.15}$$

In the Figure 3.1, we show the values of α, β, μ and σ^2 for the variables whose index i corresponds, in percentage, to the fraction k/n, for $k \in \{1, \ldots, n\}$. We multiplied μ by 10 and σ^2 by 300 in order to represent all the values with the same scale. We observe that: μ grows as i grows; σ^2 values are symmetric respect to $i = n/2$; α decreases when i grows, β increases when i grows; α and β have the same value when $i = n/2$.

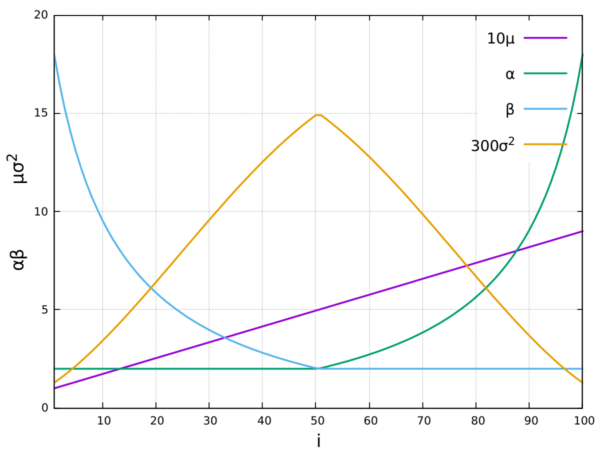


Figure 3.1

3.7.3 Known Marginal Density Functions: Results

We ran our experiments by choosing $\epsilon = \delta = 10^{-2}$ both for $\mathcal{MAA}$ and $\mathcal{AA}$. We took as input a multivariate random variable $Z = (Z_1 \ldots, Z_n)$ s.t. each $Z_i \sim Beta(\alpha_i, \beta_i)$, for $i \in \{1, \ldots, n\}$. We choose probability densities for Z components by setting their expected values, $\mu_{Z_1}, \ldots, \mu_{Z_n}$ as follows: $\mu_{Z_1} = 0.5$ and $\mu_{Z_i} = 0.1$, for $i = 2, \ldots n$.

We choose different values for size n in the set $\{10, 20, \ldots, 100\}$. We iterated $\mathcal{MAA}$ for 10 times. Then, we computed the expected value μ and the standard deviation σ of, respectively, the resulting number of samples ${N_Z}^{MAA}$ and the CPU time (in seconds) elapsed, ${CPU_Z}^{MAA}$. Our results for $\mathcal{MAA}$ are in Table 3.1.

	$\mathbf{N_Z}^{\mathbf{MAA}}$		$\mathbf{CPU_Z}^{\mathbf{MAA}}$	
n	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$
10	195536260	89152	2.16	0.005
20	782107120	327613	4.39	0.28
30	1760659740	673415	6.58	0.47
40	$3.13021e+09$	$1.44318e+06$	8.63	0.24
50	$4.89078e+09$	$1.34344e+06$	11	0.81
60	$7.04403e+09$	$2.69477e+06$	12.82	0.09
70	$9.58954e+09$	$4.699e+06$	15.12	0.35
80	$1.2523e+10$	$4.02549e+06$	17.39	0.86
90	$1.58517e+10$	$5.89855e+06$	19.34	0.41
100	$1.95697e+10$	$8.77443e+06$	21.84	1.17

Table 3.1. Mean μ and standard deviation σ of ${N_Z}^{MAA}$ and ${CPU_Z}^{MAA}$, resulting from experiments performed on $\mathcal{MAA}$.

Let N_{Z_1} and CPU_{Z_1} be, respectively, the number of samplesgenerated by $\mathcal{AA}$ for Z_1 and the CPU time (in seconds) elapsed. As the other univariate random variables $Z_2, \ldots, Z_n$ have the same mean value μ, the expected value for the number of samplesgenerated (and, as a consequence, the CPU time) are the same for all of them. Let N_{Z_2} and CPU_{Z_2} be, respectively, the number of samplesgenerated by $\mathcal{AA}$ for Z_i and the CPU time elapsed, for $i \in \{2, \ldots, n\}$. The expected number of samplesgenerated by $\mathcal{AA}$ and the expected CPU time elapsed for Z are defined by the following equations:

$$\mathbb{E}[N_Z{}^{AA}] = \mathbb{E}[N_{Z_1}] + (n-1)\mathbb{E}[N_{Z_2}] \tag{3.16}$$

$$\mathbb{E}[CPU_Z{}^{AA}] = \mathbb{E}[CPU_{Z_1}] + (n-1)\mathbb{E}[CPU_{Z_2}] \tag{3.17}$$

From the experimental results it holds that $\mathbb{E}[N_{Z_1}] = 438823$, $\mathbb{E}[CPU_{Z_1}] = 0.0308741$, $\mathbb{E}[N_{Z_2}] = 1951125.6$, $\mathbb{E}[CPU_{Z_2}] = 0.1313634$. The functions defined in the Eq. 3.16 and 3.17 have the same trend, as shown in Figure 3.2. This confirms that the number of samples is a reasonable complexity measure, as from *Notation* 3.1.

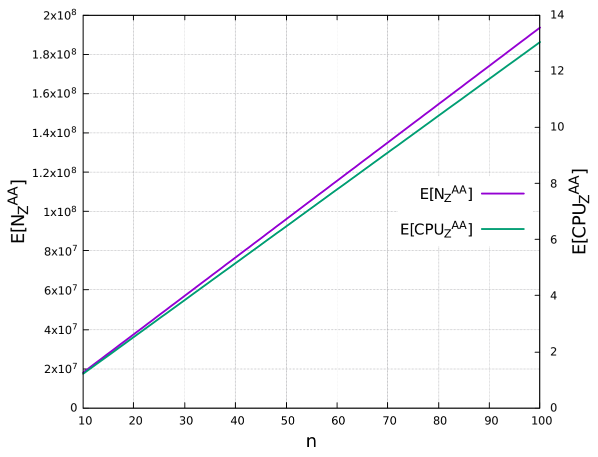


Figure 3.2. Functions of the lines representing the expected values $\mathbb{E}[N_Z{}^{AA}]$ and $\mathbb{E}[CPU_Z{}^{AA}]$, in seconds.

In Figure 3.3 we show the functions representing: the expected value $\mathbb{E}[N_Z{}^{MAA}]$ of the number of samplescomputed by $\mathcal{AA}$; the expected value $\mathbb{E}[N_Z{}^{AA}]$ of the number of samplescomputed by $\mathcal{MAA}$; the expected values of the CPU time elapsed for both the algorithms. We observe that $\mathbb{E}[N_Z{}^{MAA}] > \mathbb{E}[N_Z{}^{AA}]$ and $\mathbb{E}[CPU_Z{}^{MAA}] > \mathbb{E}[CPU_Z{}^{AA}]$. This confirms optimality of (multivariate) $\mathcal{AA}$ in this case (Section 3.6) and that there is no advantage in using $\mathcal{MAA}$ rather than $\mathcal{AA}$ to compute $\tilde{\mu}_Z$ when $Z_1, \ldots, Z_n$ probability density functions are known.

3.7.4 Unknown Marginal Density Functions: Case Study

Let $Z = (Z_1, \ldots, Z_n)$ be a multivariate random variable distributed according to a known distribution. We performed our experiments by choosing the *Dirichlet Distribution*, with proper parameters. Then, $Z \sim Dir(\alpha)$ and $p_Z()$ is the density function of the *Dirichlet Distribution*, used to generate samples of Z. More details about the choice of the parameters of the *Dirichlet* distribution are given in the next Section 3.7.4.1.

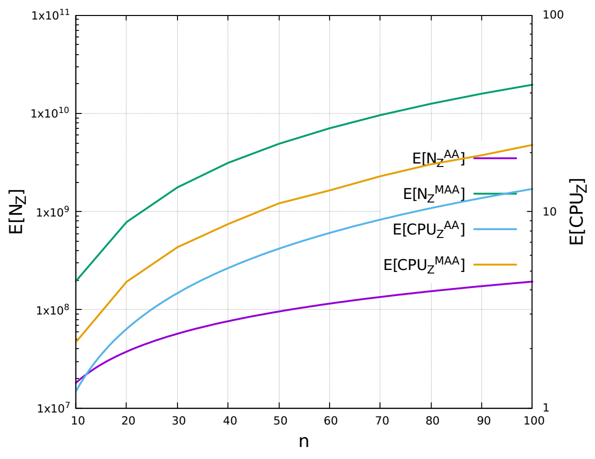


Figure 3.3. Functions representing the expected number of samplesand the expected CPU time for both (multivariate) $\mathcal{AA}$ and $\mathcal{MAA}$.

3.7.4.1 Dirichlet Distribution Parameters

The *Dirichlet Distribution* is a family of continuous multivariate probability distributions (see [72]). Let Z be a multivariate random variable distributed according to the *Dirichlet Distribution*, that is $Z \sim Dir(\alpha)$.

The vector $\alpha = (\alpha_1, \dots, \alpha_n)$ is the concentration parameter of the Dirichlet distribution s.t.: $\alpha_i > 0$, for $i \in \{1, \dots, n\}$. Furthermore, it holds $\sum_{i=1}^{n} z_i = 1$ and $z_i \in (0, 1)$. Let $\alpha_0 = \sum_{k=1}^{n} \alpha_k$. From the properties of Dirichlet distribution, we know that the mean μ_{Z_i} and the variance ${\sigma^2}_{Z_i}$ of each Z_i are computed as in equations (3.18) and (3.19).

$$\mu_{Z_i} = \frac{\alpha_i}{\alpha_0} \qquad (3.18) \qquad\qquad {\sigma^2}_{Z_i} = \frac{\alpha_i(\alpha_0 - \alpha_i)}{{\alpha_0}^2(\alpha_0 + 1)} \qquad (3.19)$$

Z_i values obtained from the Dirichlet distribution depend on the choice of α. We want that $\exists Z_i$ s.t. ${\sigma^2}_{Z_i}$ be large. Therefore, we define, in equation (3.20), a custom function $f : \mathbb{N} \to \mathbb{R}^+$ to generate the α vector components α_i: $f(i) = \alpha_i$. We want that α_0 depends on the size of Z, thus we set $\alpha_0 = n$. From equation (3.19) we observe that ${\sigma^2}_{Z_i}$ attains its maximum value when $\alpha_i = \alpha_0 - \alpha_i$, that is $\alpha_i = \alpha_0/2 = n/2$. Then, we set $\alpha_1 = n/2$. After that, we divide the other $\alpha_0 - 1 = n - 1$ elements in two sets, among which we distribute the remaining $n/2$ quantity. Accordingly: a first set of $a - 1$ items, each one having as value a fraction θ of $n/2$, that is a medium value; a second set of $n - a$ items, each one having as value a $(1 - \theta)$ fraction of $n/2$, that is a small value.

$$f(i) = \begin{cases} n/2 & \text{if } i = 1 \\ \theta \frac{n}{2} \frac{1}{a-1}, & \text{if } i \in \{2, \dots, a\} \\ (1-\theta) \frac{n}{2} \frac{1}{n-a}, & \text{if } i \in \{a+1, \dots, n\} \end{cases} \qquad (3.20)$$

In equation (3.21) we show that $\sum_{i=1}^{n} f(i) = n$, then $\sum_{i=1}^{n} \alpha_i = n$. In fact:

$$\sum_{i=1}^{n} f(i) = f(1) + \sum_{k=2}^{a} f(k) + \sum_{k=a+1}^{n} f(k) \tag{3.21a}$$

$$= \frac{n}{2} + \sum_{k=2}^{a} \theta \frac{n}{2(a-1)} + \sum_{k=a+1}^{n} (1-\theta) \frac{n}{2(n-a)} \tag{3.21b}$$

$$= \frac{n}{2} + (a-1)\theta \frac{n}{2} \frac{1}{a-1} + (n-a)(1-\theta) \frac{n}{2} \frac{1}{n-a} \tag{3.21c}$$

$$= n \tag{3.21d}$$

After obtaining α, for each Z_i we can compute the expected value μ_{Z_i} through the formula in (3.18), and the variance ${\sigma^2}_{Z_i}$, through (3.19), as shown in equations (3.22) and (3.23).

$$\mu_{Z_i} = \begin{cases} 1/2 & \text{if } i = 1 \\ \frac{\theta}{2(a-1)} & \text{if } i \in \{2, \dots, a\} \\ \frac{1-\theta}{2(n-a)} & \text{if } i \in \{a+1, \dots, n\} \end{cases} \tag{3.22}$$

$${\sigma^2}_{Z_i} = \begin{cases} \frac{1}{4(n+1)} & \text{if } i = 1 \\ \frac{2\theta(a-1)-\theta^2}{4(a-1)^2(n+1)} & \text{if } i \in \{2, \dots, a\} \\ \frac{2(1-\theta)(n-a)-(1-\theta)^2}{4(n-a)^2(n+1)} & \text{if } i \in \{a+1, \dots, n\} \end{cases} \tag{3.23}$$

We want that Z_1 variance ${\sigma^2}_{Z_1}$ is bigger than the variance of the other components. Then, we arbitrarily set $a = min(n/2, 100)$ and $\theta = 1/2$, as we want $\theta \in (0, 1)$ and, thus, we obtain the following values of μ_{Z_i} and ${\sigma^2}_{Z_i}$, for $i \in \{1, \dots, n\}$:

$$\mu_{Z_i} = \begin{cases} 1/2 & \text{if } i = 1 \\ \frac{1}{2(n-2)} & \text{if } i \in \{2, \dots, n/2\} \\ \frac{1}{2n} & \text{if } i \in \{n/2+1, \dots, n\} \end{cases} \tag{3.24}$$

$${\sigma^2}_{Z_i} = \begin{cases} \frac{1}{4(n+1)} & \text{if } i = 1 \\ \frac{2n-5}{4(n-2)^2(n+1)} & \text{if } i \in \{2, \dots, n/2\} \\ \frac{2n-1}{4n^2(n+1)} & \text{if } i \in \{n/2+1, \dots, n\} \end{cases} \tag{3.25}$$

As evidenced by equation (3.25), ${\sigma^2}_{Z_i}$ and ${\sigma^2}_{Z_j}$ are similar, for $i \in \{2, \dots, n/2\}$ and $j \in \{n/2+1, \dots, n\}$. In fact:

$$\lim_{n \to \inf} \frac{{\sigma^2}_{Z_i}}{{\sigma^2}_{Z_j}} = 1 \tag{3.26}$$

On the contrary, Z_1 variance is leading respect to the variance of the other components. In fact:

$$\lim_{n \to \inf} \frac{{\sigma^2}_{Z_i}}{{\sigma^2}_{Z_1}} = 0 \text{ , for } i \in \{2, \dots, n/2\} \tag{3.27a}$$

$$\lim_{n \to \inf} \frac{{\sigma^2}_{Z_i}}{{\sigma^2}_{Z_1}} = 0 \text{ , for } i \in \{n/2+1, \dots, n\} \tag{3.27b}$$

This result is shown also from the ratio of variance values in pairs, represented in Figure 3.4.

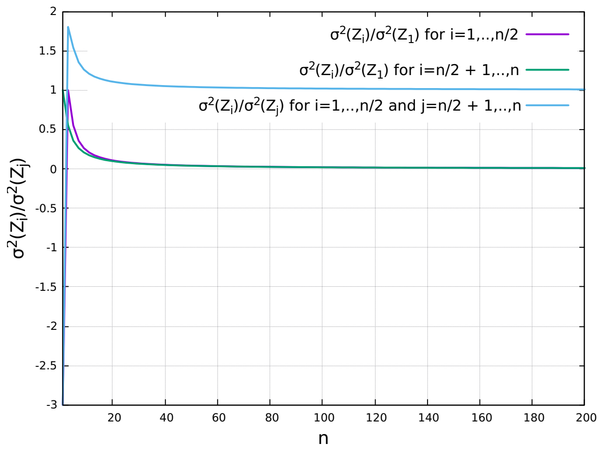


Figure 3.4. Functions representing the following ratios: ${\sigma^2}_{Z_i}/{\sigma^2}_{Z_1}$, for $i \in \{2, \dots, n/2\}$; ${\sigma^2}_{Z_i}/{\sigma^2}_{Z_1}$, for $i \in \{n/2+1, \dots, n\}$; ${\sigma^2}_{Z_i}/{\sigma^2}_{Z_j}$, for $i \in \{2, \dots, n/2\}$ and $j \in \{n/2 + 1, \dots, n\}$

3.7.5 Unknown Marginal Density Functions: Results

We ran our experiments by choosing different values of (ϵ, δ) parameters for both $\mathcal{MAA}$ and (multivariate) $\mathcal{AA}$ and by taking as input a multivariate random variable Z of size n. For each $n \in \{2, 5, 10, 15, 20, 25, 30\}$, we iterated $\mathcal{MAA}$ and $\mathcal{AA}$ for 10 times. Then, we computed the expected value μ and the standard deviation σ of the number of samples and of the CPU time (in seconds) elapsed. Our results for $\mathcal{MAA}$ are in Table 3.2 and for $\mathcal{AA}$ are in Table 3.3.

We fixed the maximum execution time to 1 day, after which we stopped the experiment and saved the number of samples and the CPU time. We observe that the CPU time increases as the size of Z increases and as ϵ and δ decrease. By comparing the results in Table 3.3 with the results in Table 3.2, it comes out that $\mathbb{E}[{N_Z}^{AA}]/\mathbb{E}[{N_Z}^{MAA}] \sim n$ and $\mathbb{E}[{CPU_Z}^{AA}]/\mathbb{E}[{CPU_Z}^{MAA}] \sim n$.

In Table 3.4 we show the results about: the ratios, respectively, of the expected values and the standard deviations of the number of samples generated from $\mathcal{AA}$ and the number of samples generated from $\mathcal{MAA}$; the ratios, respectively, of the expected values and the standard deviations of the CPU time (in seconds) elapsed for $\mathcal{AA}$ and for $\mathcal{MAA}$. As from Theorem 3.2, it holds that $\mathbb{E}[{N_Z}^{AA}]/\mathbb{E}[{N_Z}^{MAA}] \sim n$ and, as a consequence, $\mathbb{E}[{CPU_Z}^{AA}]/\mathbb{E}[{CPU_Z}^{MAA}] \sim n$.

To further explore the performance of MAA, we executed other experiments on multivariate random variables of (large) size n in $\{100, \dots, 1000\}$. We choose $\epsilon = \delta = 10^{-2}$ and we executed both $\mathcal{MAA}$ and $\mathcal{AA}$ for only one time. When an algorithm required more than 1 day of execution, we choose not to run it. The experimental results in Table 3.5 show that: in many more cases the CPU time elapsed for $\mathcal{AA}$ is over 1 day; in the only case where $\mathcal{AA}$ stops before one day, it holds that (as from Theorem 3.2) ${N_Z}^{AA} \sim n \times {N_Z}^{MAA}$ and ${CPU_Z}^{AA} \sim n \times {CPU_Z}^{MAA}$.

n	ϵ, δ	$\mathbf{N_Z}^{\mathbf{MAA}}$		$\mathbf{CPU_Z}^{\mathbf{MAA}}$	
		μ	σ	μ	σ
2	$10^{-2}, 10^{-2}$	$1.74499e+06$	6004.4	0.15	0.0006
	$10^{-2}, 10^{-3}$	$2.47727e+06$	7291.16	0.19	0.003
	$10^{-2}, 10^{-4}$	$3.22877e+06$	6110.68	0.26	0.013
5	$10^{-2}, 10^{-2}$	$1.46106e+07$	22712.5	2.2	0.004
	$10^{-2}, 10^{-3}$	$2.08052e+07$	26331.2	3.05	0.023
	$10^{-2}, 10^{-4}$	$2.69946e+07$	46553.4	3.99	0.064
10	$10^{-2}, 10^{-2}$	$4.29281e+07$	39780.7	6.13	0.0104
	$10^{-2}, 10^{-3}$	$6.11994e+07$	77270	8.09	0.064
	$10^{-2}, 10^{-4}$	$7.95835e+07$	70101.2	10.54	0.13
15	$10^{-2}, 10^{-2}$	$9.75433e+07$	73276.8	18.29	0.05
	$10^{-2}, 10^{-3}$	$1.39328e+08$	62269.3	25.83	0.303
	$10^{-2}, 10^{-4}$	$1.81203e+08$	63831	33.37	0.12
20	$10^{-2}, 10^{-2}$	$1.57604e+08$	70556.8	22.92	0.06
	$10^{-2}, 10^{-3}$	$2.25298e+08$	74570.8	30.64	0.28
	$10^{-2}, 10^{-4}$	$2.93051e+08$	71283.6	39.95	0.54
25	$10^{-2}, 10^{-2}$	$2.51116e+08$	114838	48.10	0.93
	$10^{-2}, 10^{-3}$	$3.59141e+08$	115617	67.69	1.403
	$10^{-2}, 10^{-4}$	$4.67235e+08$	129522	87.79	0.36
30	$10^{-2}, 10^{-2}$	$3.42959e+08$	63059.1	50.25	0.31
	$10^{-2}, 10^{-3}$	$4.90665e+08$	104208	66.96	0.28
	$10^{-2}, 10^{-4}$	$6.38364e+08$	64180.2	88.75	4.49

Table 3.2. Mean μ and standard deviation σ of number of samples and CPU time (in seconds) elapsed resulting from experiments performed on $\mathcal{MAA}$.

3.7.6 Lower and Upper Bounds: Results

In this section we show the experimental results proving that both the Lower Bound in Item (2) of Theorem 3.3 and the Upper Bound in Item (3) of Theorem 3.2 to $\mathbb{E}[{N_Z}^{MAA}]$ can be attained, at some input instances.

First, we choose a multivariate random variable $Z = (Z_1, \ldots, Z_n)$ distributed according to the *Dirichlet Distribution*, by choosing the same parameters as in Section 3.7.4.1. We show that, under these conditions, the Upper Bound to $\mathbb{E}[{N_Z}^{MAA}]$ is attained (see eq. 3.28), while the Lower Bound is not (see eq. 3.29).

$$\lim_{n\to\infty} \frac{\mathbb{E}[{N_Z}^{MAA}]}{n\sum_{i=1}^{n} \frac{\rho_{Z_i}}{\mu^2_{Z_i}}} \textbf{ is finite} \tag{3.28}$$

$$\lim_{n\to\infty} \frac{\mathbb{E}[{N_Z}^{MAA}]}{n \max\{\frac{\rho_{Z_i}}{\mu^2_{Z_i}} | i = 1, \ldots, n\}} \textbf{ is } \infty \tag{3.29}$$

See Figure 3.5 showing these results.

Second, we choose $Z \sim Dir(\alpha)$ with the concentration parameter $\alpha = (\alpha_1, \ldots, \alpha_n)$ s.t. $\alpha_i = 1$ and, for $i \in \{1, \ldots, n\}$. Thus, $\alpha_0 = \sum_{k=1}^{n} = n$. By running $\mathcal{MAA}$ to compute the expected value of this multivariate random variable, the Lower Bound to $\mathbb{E}[{N_Z}^{MAA}]$ is attained (see eq. 3.30), but the Upper Bound is not (see eq. 3.31).

n	ϵ, δ	$\mathrm{N_Z}^{\mathrm{AA}}$		$\mathrm{CPU_Z}^{\mathrm{AA}}$	
		μ	σ	μ	σ
2	$10^{-2}, 10^{-2}$	$2.45664e+06$	2395.04	0.19	0.17
	$10^{-2}, 10^{-3}$	$3.50445e+06$	7429.58	0.29	0.006
	$10^{-2}, 10^{-4}$	$4.52634e+06$	7639.66	0.337	0.002
5	$10^{-2}, 10^{-2}$	$5.04413e+07$	51296.8	7.33	0.07
	$10^{-2}, 10^{-3}$	$7.19403e+07$	96836.5	10.64	0.09
	$10^{-2}, 10^{-4}$	$9.33048e+07$	66818.6	13.5	0.27
10	$10^{-2}, 10^{-2}$	$3.54633e+08$	182911	46.25	1.53
	$10^{-2}, 10^{-3}$	$5.06222e+08$	217414	70.61	0.07
	$10^{-2}, 10^{-4}$	$6.57925e+08$	192921	85.45	1.73
15	$10^{-2}, 10^{-2}$	$1.22159e+09$	269794	225.33	6.01
	$10^{-2}, 10^{-3}$	$1.74629e+09$	353977	321.4	0.35
	$10^{-2}, 10^{-4}$	$2.27087e+09$	695214	414.9	5.75
20	$10^{-2}, 10^{-2}$	$2.85494e+09$	583383	385.46	12.76
	$10^{-2}, 10^{-3}$	$4.08209e+09$	847826	581.84	2.96
	$10^{-2}, 10^{-4}$	$5.31017e+09$	532720	712.81	16.99
25	$10^{-2}, 10^{-2}$	$5.60223e+09$	580161	1041.31	5.1
	$10^{-2}, 10^{-3}$	$8.0147e+09$	1102402	1510.04	24.95
	$10^{-2}, 10^{-4}$	$1.04266e+10$	917583	1945.88	20.09
30	$10^{-2}, 10^{-2}$	$9.62337e+09$	1286080	1307.92	36.11
	$10^{-2}, 10^{-3}$	$1.37711e+10$	841766	1972.38	3.98
	$10^{-2}, 10^{-4}$	$1.79187e+10$	1858160	2424.42	30.18

Table 3.3. Mean μ and standard deviation σ of number of samples and CPU time (in seconds) elapsed resulting from experiments performed on $\mathcal{AA}$.

$$\lim_{n\to\infty} \frac{\mathbb{E}[{N_Z}^{MAA}]}{n \max\{\frac{\rho_{Z_i}}{\mu^2_{Z_i}} | i = 1, \dots, n\}} \text{ is finite} \tag{3.30}$$

$$\lim_{n\to\infty} \frac{\mathbb{E}[{N_Z}^{MAA}]}{n \sum_{i=1}^{n} \frac{\rho_{Z_i}}{\mu^2_{Z_i}}} = 0 \tag{3.31}$$

These results are shown in Figure 3.6.

3.8 Conclusions

We addressed the problem of computing in an efficient way an (ϵ, δ)-approximation $\tilde{\mu}_Z$ of the expected value of a multivariate random variable $Z = (Z_1, \dots, Z_n)$. When the marginal probability densities for Z are known a straightforward multivariate version of the Monte Carlo-based algorithm $\mathcal{AA}$ from [30] solves our problem. However, when marginal probability densities for Z are not known such a multivariate $\mathcal{AA}$ does not yield an optimal (as for number of samples) algorithm.

We presented an efficient algorithm $\mathcal{MAA}$ to compute an (ϵ, δ)-approximation $\tilde{\mu}_Z$ of the expected value of a multivariate random variable $Z = (Z_1, \dots, Z_n)$ when marginal probability densities for Z are not known. For $\mathcal{MAA}$ we presented correctness and an Upper Bound to the complexity (given in terms of the number of generated samples). Furthermore, we gave a Lower Bound to the expected number

n	$\boldsymbol{\epsilon, \delta}$	$\frac{\mathbb{E}[N_Z{}^{AA}]}{\mathbb{E}[N_Z{}^{MAA}]}$	$\frac{\sigma(N_Z{}^{AA})}{\sigma(N_Z{}^{MAA})}$	$\frac{\mathbb{E}[CPU_Z{}^{AA}]}{\mathbb{E}[CPU_Z{}^{MAA}]}$	$\frac{\sigma(CPU_Z{}^{AA})}{\sigma(CPU_Z{}^{MAA})}$
2	$10^{-2}, 10^{-2}$	1.41	0.004	1.26	0.111
	$10^{-2}, 10^{-3}$	1.41	0.005	1.52	0.044
	$10^{-2}, 10^{-4}$	1.40	0.004	1.32	0.064
5	$10^{-2}, 10^{-2}$	3.45	0.007	3.32	0.032
	$10^{-2}, 10^{-3}$	3.46	0.005	3.49	0.047
	$10^{-2}, 10^{-4}$	3.46	0.007	3.38	0.1
10	$10^{-2}, 10^{-2}$	8.26	0.007	7.54	0.25
	$10^{-2}, 10^{-3}$	8.27	0.012	8.73	0.071
	$10^{-2}, 10^{-4}$	8.27	0.007	8.11	0.11
15	$10^{-2}, 10^{-2}$	12.52	0.009	12.32	0.337
	$10^{-2}, 10^{-3}$	12.53	0.005	12.44	0.15
	$10^{-2}, 10^{-4}$	12.53	0.004	12.43	0.15
20	$10^{-2}, 10^{-2}$	18.11	0.008	16.81	0.56
	$10^{-2}, 10^{-3}$	18.12	0.008	18.99	0.19
	$10^{-2}, 10^{-4}$	18.12	0.005	17.84	0.52
25	$10^{-2}, 10^{-2}$	22.31	0.009	21.65	0.456
	$10^{-2}, 10^{-3}$	22.31	0.007	22.31	0.6
	$10^{-2}, 10^{-4}$	22.31	0.006	22.16	0.27
30	$10^{-2}, 10^{-2}$	20.06	0.005	26.03	0.781
	$10^{-2}, 10^{-3}$	28.07	0.006	29.45	0.136
	$10^{-2}, 10^{-4}$	28.07	0.003	27.32	1.36

Table 3.4. This table shows that $\mathbb{E}[N_Z{}^{AA}]/\mathbb{E}[N_Z{}^{MAA}] \sim n$ and, as a consequence, $\mathbb{E}[CPU_Z{}^{AA}]/\mathbb{E}[CPU_Z{}^{MAA}] \sim n$.

n	ϵ, δ	MAA		AA	
		N_Z	CPU_Z	N_Z	CPU_Z
100	$10^{-2}, 10^{-2}$	$3.61945e+09$	477.97 seconds	$3.546e+09$	50750.66 seconds
200	$10^{-2}, 10^{-2}$	$1.44761e+10$	0.81 hours	n.a.	$>$ 24 hours
300	$10^{-2}, 10^{-2}$	$4.3428e+10$	3.07 hours	n.a.	$>$ 24 hours
400	$10^{-2}, 10^{-2}$	$8.68432e+10$	3.88 hours	n.a.	$>$ 24 hours
500	$10^{-2}, 10^{-2}$	$1.44742e+11$	6.9 hours	n.a.	$>$ 24 hours
600	$10^{-2}, 10^{-2}$	$2.17099e+11$	10.64 hours	n.a.	$>$ 24 hours
700	$10^{-2}, 10^{-2}$	$3.03952e+11$	15.1 hours	n.a.	$>$ 24 hours
800	$10^{-2}, 10^{-2}$	$4.05259e+11$	20.77 hours	n.a.	$>$ 24 hours
900	$10^{-2}, 10^{-2}$	$5.21015e+11$	26.4 hours	n.a.	$>$ 26.4 hours
1000	$10^{-2}, 10^{-2}$	$6.51272e+11$	33.27 hours	n.a.	$>$ 33.27 hours

Table 3.5. Number of samples generated and CPU time elapsed resulting from experiments performed on $\mathcal{MAA}$ and $\mathcal{AA}$ on a multivariate random variable of big size.

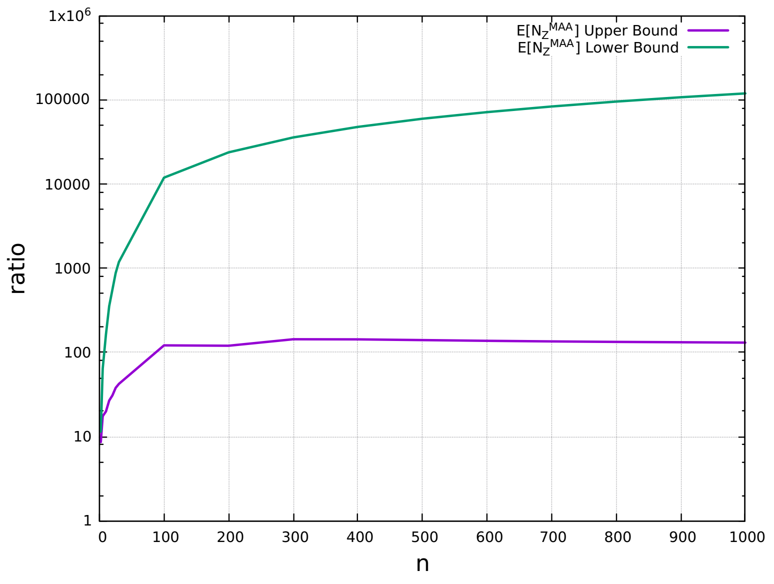


Figure 3.5. Sampling from a multivariate random variable $Z \sim Dir(\alpha)$ s.t. α vector is chosen as in Section 3.7.4.1, the Upper Bound to $\mathbb{E}[N_Z{}^{MAA}]$ is attained but the Lower Bound is not. In fact, the function in purple represents the ratio in eq. 3.28 that tends to a finite value, while the function in green represents the ratio in eq. 3.29, that tends to infinity.

of samples for any (ϵ, δ)-approximation algorithm estimating the expected value of a multivariate random variable. Then, we presented experimental results confirming our evaluations.

We implemented both multivariate $\mathcal{AA}$ and $\mathcal{MAA}$ and evaluated their performance. Our experimental results show that, as to be expected, when marginal density functions are known, there is no advantage, in terms of number of samples, in using $\mathcal{MAA}$ rather than $\mathcal{AA}$ to compute $\tilde{\mu}_Z$. On the other hand, when marginal density functions are not known, $\mathcal{MAA}$ is strictly faster than multivariate $\mathcal{AA}$,

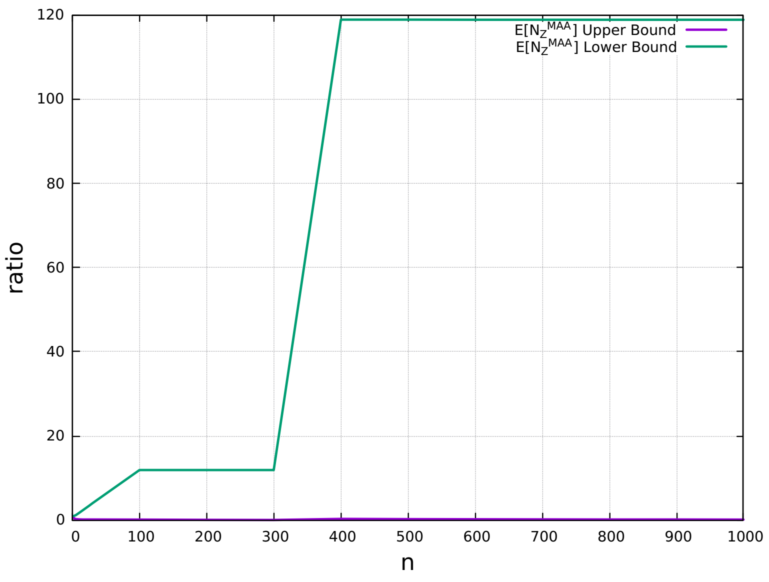


Figure 3.6. Sampling from a multivariate random variable $Z \sim Dir(\alpha)$ s.t. for each α vector component it holds $\alpha_i = 1$ and, for $i \in \{1, \dots, n\}$, the Lower Bound to $\mathbb{E}[N_Z{}^{MAA}]$ is attained but the Upper Bound is not. In fact, the function in green represents the ratio in eq. 3.30 that tends to a finite value, while the function in purple represents the ratio in eq. 3.31, that tends to zero.

when $n \geq 2$, and, at some input instances, can as much as n times faster than $\mathcal{AA}$.

Finally, there exist multivariate density functions s.t. the presented Upper Bound and Lower Bound to $\mathbb{E}[N_Z^{MAA}]$ can be attained.

Chapter 4

A Mechanistic Model for Retweet Dynamics

In this section we present our contribution to the estimation of a probability distribution parameters in the field of predictive models.

The content of this section is related to the paper *A Mechanistic Model for Retweet Dynamics* (authors: A. Pappagallo, A.Massini and E.Tronci) that will be submitted.

4.1 Introduction

Nowadays web platforms offering microblogging and social services are used by billions of people, with very different age, gender, education, or nationality. These systems, better known as *social networks*, provide users with several kinds of activities. For example, a user can just read up-to-date information, or share news, video, images. He can interact with other users all around the world, by giving them feedback or sharing his posts to show interest in his activity. Users' goals when using social networks may be of different natures, *e.g.*, professional, recreational.

In this work, we focus on Twitter that in the last years has gathered the interest of an increasing number of firms, politicians, organizations, and even public institutions. This exponential growth of interest is due to the availability of a large amount of user-contributed data enabling almost real-time social data analytics. In a Twitter network, users can be *followers* or *followees* and they can post messages, called *tweets*, that can be sent or re-shared (*retweeted*).

In such a context, studying the dynamics of the information spread inside the Twitter network and the interactions among users is very important for many reasons, such as: to understand users trends, to do marketing campaigns, to exercise political influence, to detect potentially malicious behaviors. It turns out that is crucial predicting how responsive a user is to a message spread into the network. More in details, it is very useful to know how many times a user (or set of users) will retweet, on average, a message (*e.g.*, [65]).

4.1.1 Contributions

We propose a mechanistic model for the dynamics of the retweeting rate of a message inside a Twitter network, along with a model validation methodology. More specifically, our contributions are the following.

First, we develop a discrete-time dynamical system modeling the retweeting mechanism of Twitter. Second, we model the behavior of the users participating in a Twitter network as a stochastic variable and we compute its parameters (mean and variance) from an available dataset. This, together with the above mentioned discrete-time system, allows us to define a model to predict the time evolution of the expected value of the retweeting rate for each user in the network. Our model is mechanistic and identifiable, as it is built on top of identifiable parameters.

Third, we validate our model with respect to real data taken from a Twitter dataset. We show that our model predicts, for each user and for each message: the average retweeting rate for all the time slots in a time frame; the retweets number and the retweeting users at a certain time and until a certain time. Furthermore, our model allows computing some error measures, *e.g.*, RMSE and MAPE, both for a single node (user) and for an arbitrary set of nodes.

Our experimental results about new messages spread into the network show that our model predicts the qualitative time behavior and also, reasonably well, the quantitative time behavior. Furthermore, with a probability of 80% the $MAPE$ of the average retweeting rate is between $[0, 50\%]$, while the maximum $MAPE$ value of the retweeting users' number is between $[0.39, 4.6]$. Finally, the expected value of the $MAPE$ of the average retweeting rate computed for a set containing all the network nodes is in the range $[9.7\%, 39\%]$.

4.1.2 Related Work

Prediction of the re-sharing rate in a social network has been widely studied in the last years.

In [135] a probabilistic collaborative filtering model to predict individual retweets in Twitter is developed. This model performs a pairwise prediction of the retweeting of a specific message, within one hour. In [96] a model to predict if a message will be retweeted or not is presented. Their model works through a machine learning approach based on a passive-aggressive algorithm. However, the aim here is to perform a binary classification task, rather than to estimate a value. The work in [43] proposes an extended reinforced Poisson process model for the retweeting dynamics of a message in the early stages of its spreading. The aim is to predict the future popularity of a message. In [136] and [71] a statistical model based on the theory of the self-exciting point processes is presented. The focus is, again, on predicting the popularity of a tweet, in terms of its final retweet count in a considered time frame. Ma et al. in [79] define a mechanistic model of spreading, accounting the network structure of Twitter and behavioral patterns, obtained by an empirical analysis of a dataset. They show that there exist a close relationship between the spreading sub-network structure and the final number of retweets. Their model allows predicting the final popularity using the out-degree distribution of the retweet tree during the early stage of spreading. The work in [44] addresses the problem of identifying how *memes* (hashtag or URL within a tweet) emerge and concur in social networks. They provide a model for on-line sharing behavior that is analytically tractable and accounts the memory time of users and the connectivity structure of the social network.

Summing up, the aforementioned works have used different types of models to predict the popularity of a message. However, to the best of our knowledge, no models are available for the time evolution, at the node level, of the retweeting rate of a message inside the Twitter network.

Furthermore, there exist literature focusing on the identification of influential users in a social network. The works in [58] and [45] mainly use the knowledge of the underlying network topology. In [58] and [4] the influence of users in Twitter is quantified by different influence measures and various centrality measures. In [19] they present an unsupervised algorithm to identify a set of influential users in a social network by exploiting the temporal sequence of retweets in Twitter cascades.

Despite these works have dealt with the analysis of the most active users activity in a social network, our model allows predicting the average retweeting rate in the network, at a certain time, both at the node level and at the set of users level.

4.2 Background

In this section we characterize Twitter as a *follow graph* [89], that is a directed graph $G = (V, E)$ where: V is the set of users; E is the set of edges (i, j), each one representing the *follow* relationship between the user i (*follower*) and the user j (*followee*). Let $Nodes = V$ be the set, of size n, containing all users identifiers. In the Twitter scenario, shown in Figure 4.1, users may tweet or re-tweet instances (*tweets*) of messages $m \in \mathcal{M}$. Let $\mathcal{M}$ be the set of messages spread into the network. In our setting, we consider different *tweets* of the same message as equal to each other.

In the rest of the sections we will consider a snapshot of Twitter, included in a time frame $\mathcal{T} \subset \mathbb{N}^+$ of about 12 days, that will be discussed in Section 4.8.1. We consider the time frame $\mathcal{T}$ as made up of *bins*, each one made up of time slots k of size τ (*e.g.*, τ may be one hour). Let $[k]$ be the bin containing the time slot k. Let Δ be the size of each bin, i.e. $|[k]| = \Delta$. Let $bin(k) = \lfloor \frac{k}{\Delta} \rfloor$ be the identifier of $[k]$.

4.3 Notation

In Table 4.1 we summarize the notation used along the sections.

4.4 Overview

In the next sections, we are going to present our strategy to estimate the expected value of the retweeting rate of a message m, spread into the Twitter network, for each user and for each time slot. Our strategy consists of the following steps:

- We define a user model (see Section 4.5.1) to compute the retweeting rate of the message m between a user and each one of its followees, for each time slot

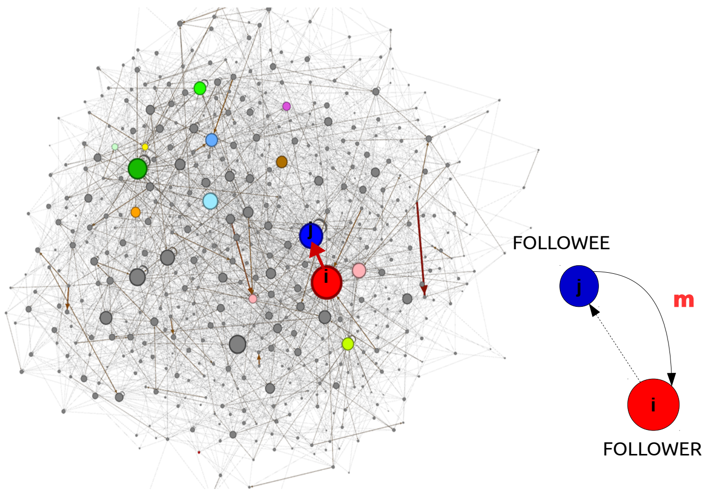


Figure 4.1. Twitter network made up of (follower,followee) pairs, such as (i, j)

Symbol	Description
m	message taken from a set $\mathcal{M}$
k	time slot
τ	size of the time slot k
$[k]$	*bin* containing the time slot k
Δ	number of time slots in a bin, *i.e.*, $\|[k]\|$
d	delay, with values in $0, \ldots, D$. $\|d\| = \tau$, *i.e.*, the same size of k
i	user i corresponding to a node of the network. $i \in Nodes$
(i, j)	pair of (follower,followee) in the network
$X_{i,m}(k)$	re-tweeting rate of m, for user i at the time slot k
$\bar{X}_{i,m}(k)$	average retweeting rate of m, for user i at the time slot k
$A^m_{i,j,d}(k)$	retweeting rate of m, for user i from its followee j, with delay d, at the time slot k
$\bar{A}_{i,j,d}([k])$	expected value of each $A^m_{i,j,d}(k)$, depending only on the bin $[k]$
$\Delta_{i,m}(k)$	error on $X_{i,m}(k)$
$Var(A_{i,j,d}([k]))$	variance of each $A^m_{i,j,d}(k)$, depending only on the bin $[k]$

Table 4.1. List of symbols

k. This model will be represented as a stochastic variable;

- We compute the user model parameters (see Section 4.6), *i.e.*, mean and variance, from an available dataset providing a huge number of observations and, thus, high statistical confidence about the parameters;

- We define a retweeting rate stochastic model (see Section 4.5.2) for each user i, for each message m and for each time slot k. This model is built on the previously described user model;

- Building on the previous model, we define an average retweeting rate model (see Section 4.5.3). This is a discrete-time system that will be simulated for a certain number of time slots, to predict the average retweeting rate of a new message spread into the network.

4.5 Twitter Model

In this section, we present our model of the dynamics of the *Twitter network*. More specifically, we model the time evolution of the retweeting rate of a message for each user of the network. Our model is built on a stochastic input, namely a random variable representing the interaction between a pair of users.

First of all, in Section 4.5.1, we introduce a model for the stochastic input, which is a user (*follower*) model, for each pair of *(follower, followee)*. In the next sections, we present the mechanistic model of the retweeting rate of each Twitter network node, based on the user model. Then, as we are interested in estimating the expected value of the retweeting rate of each node, we define an average retweeting rate model. Finally, we define an error model that, for each node and each message, computes the difference between the predicted average retweeting rate and the actual rate. We will show that the expected value of the error is null because we have only one observation of the global retweeting activity of the Twitter subnetwork available from the dataset.

4.5.1 User Model

In this section, we define a model for the behavior of each user $i \in Nodes$ with respect to a specific message $m \in \mathcal{M}$ and delay $d = 0, \ldots, D$.

Accordingly, let $F_i = \{\alpha(i,1), \ldots, \alpha(i,\beta(i))\}$ be the set of user i followees. Let t_0^m be the first time slot when m is posted on Twitter. Let $A^m_{i,j,d}(t_0^m + k)$, for $j \in F_i$, be a random variable modeling the rate of instances of m that user i retweets, with delay d, at the time slot $t_0^m + k$ and received from his followee j at the time slot $k - d$.

Remark 4.1. *The time slot t_0^m, for each message $m \in \mathcal{M}$, has to be aligned to the first day of the time frame $\mathcal{T}$, as below:*

$$t_0^{m*} = t_0^m - \left\lfloor \frac{t_0^m}{24} \right\rfloor 24 \tag{4.1}$$

In order to ease the reading, we denote $A^m_{i,j,d}(t^m_0 + k)$ with $A^m_{i,j,d}(k)$.

More informally, $A^m_{i,j,d}(k)$ represents the interaction between the pair (i, j) of (follower, followee) for a time slot k and a delay d. Let $p^m_{i,j,d}(k, a)$ be the probability density function of the $A^m_{i,j,d}(k)$ random variable, generating values $a \in [0, 1]$. We make the hypothesis that the density function of each $A^m_{i,j,d}(k)$ does not depend on the message m and that for two time slots k and k' in the same bin $[k]$, $A^m_{i,j,d}(k)$ and $A^m_{i,j,d}(k')$ have the same density function. Thus we denote as $p_{i,j,d}([k], a)$ the density function of all the $A^m_{i,j,d}(k)$ random variables.

This implies that the expected value of $A^m_{i,j,d}(k)$ variable depends only on the bin $[k]$ and has to be computed as in eq. 4.2.

$$\mathbb{E}[A^m_{i,j,d}(k)] = \frac{1}{|M} \frac{1}{|[k]|} \sum_{m \in M} \sum_{k \in [k]} A^m_{i,j,d}(k) = \bar{A}_{i,j,d}([k]) \tag{4.2}$$

Furthermore, as we can reasonably assume that a pair (i, j) interacts independently from any other pair (p, q) of *(follower, followee)*, we make the hypothesis that the $A^m_{i,j,d}(k)$ variables are independent for different (i, j) pairs. This hypothesis is acceptable if we observe each $A^m_{i,j,d}(k)$ variable for a long enough time.

Let $A^m_d(k) = \{A^m_{i,j,d}(k) | i \in Nodes, j \in F_i, d = 0, \ldots, D\}$ be the vector of all $A^m_{i,j,d}(k)$ at the time slot k, that is the k^{th} extraction of values from all the $p_{i,j,d}([k], a)$ density functions. $A^m_d(k)$ represents the model of all the users at the time slot k, for a delay d. We will denote $A^m_d(k)$ with $A^m(k)$, as it holds for all $d = 0, \ldots, D$.

Regarding the behavior of a pair (i, j) of (follower,followee), we can assume that what i retweets from j at a time slot k does not depend on what i retweeted in the past time slots. Then, the users are modeled as a memoryless stochastic process. Formally, we require that:

$$\mathbb{P}[A^m(k+1) | A^m(k), A^m(k-1), \ldots, A^m(0)] = \mathbb{P}[A^m(k+1)] \tag{4.3}$$

From the above eq. 4.3 it follows that, for each time slot k, $A^m(k+1)$ does not depend on $A^m(0), \ldots, A^m(k)$.

The user model so far defined will be used as a stochastic input for the mechanistic model of the network, that will be presented in the next section.

4.5.2 Retweeting Rate Dynamic Model

So far we have introduced the model $A^m_{i,j,d}(k)$ for the retweeting rate of the follower i, at the time slot k, from the followee j with delay d. Now, for each network node $i \in Nodes$, we are going to define a retweeting rate model built on $A^m_{i,j,d}(k)$, for each followee $j \in F_i$.

Definition 4.1. *For each node $i \in Nodes$ and each message $m \in \mathcal{M}$, we define $X_{i,m}(k) \in \mathbb{R}$ as the re-tweeting rate of a message m from user i at the time slot k.*

Let $Z_{m,i}(k)$ be the number of instances of m retweeted from user i until the time slot k. Then, it holds:

$$X_{i,m}(k) = \frac{Z_{m,i}(k+1) - Z_{m,i}(k)}{\tau} \tag{4.4}$$

.

Remark 4.2. *$X_{i,m}(k)$ stands for $X_{i,m}(t_0^m + k)$, where t_0^m is the first time when the message m is spread into the network (as already said for $A^m_{i,j,d}(k)$ in the previous Section 4.5.1).*

Remark 4.3. *$X_{i,m}(k)$ represents the state of node i at the time slot k.*

In order to know how node i state is updated from the time slot k to the next time slot $k+1$, we need to define a model of $X_{i,m}(k)$, as in eq. 4.5. This model is stochastic because it depends on the random variable $A^m_{i,j,d}(k)$, representing the input and defined in Section 4.5.1.

$$X_{i,m}(k+1) = \sum_{j \in F_i, d=0}^{D} A^m_{i,j,d}(k) X_{j,m}(k-d) \tag{4.5}$$

The eq. 4.5 means that the re-tweeting rate of user i at the time slot $k+1$, depends on what i re-tweeted from j in the previous time slot k, represented by $A^m_{i,j,d}(k)$, and on what each followee $j \in F_i$, retweeted in the past ($i.e.$, $X_{j,m}(k-d)$) time slots.

Remark 4.4. *$X_{i,m}(0), \ldots, X_{i,m}(D)$ are random variables whose values are taken from a set of initial rates.*

We want to prove that $X_{i,m}(k)$ is statistically independent from $A^m_{i,j,d}(k)$, for the time slot k, for $j \in F_i$ and for $d = 0, \ldots, D$.

Theorem 4.1. *$X_{i,m}(t)$ is statistically independent from $A^m_{i,j,d}(k)$, for each time slot k, for each delay $d = 0, \ldots, D$, for each followee $j \in F_i$ and for each time slot t, s.t. $t \leq k$. Formally:*

$$\mathbb{P}(X_{i,m}(t) = x | A^m_{i,j,d}(k) = a) = \mathbb{P}(X_{i,m}(t) = x) \tag{4.6}$$

The proof of Theorem 4.1 is given in Appendix A.

4.5.3 Average Retweeting Rate Dynamic Model

In the previous section, we introduced the model $X_{i,m}(k)$ for each state of the network. We could use this model to perform many Monte Carlo simulations and finally computing the mean value of the results, to obtain the average re-tweeting rate of message $m \in \mathcal{M}$ for user i at the time slot k. But as we want to avoid the expensive Monte Carlo-based approach, we introduce the average retweeting rate model $\bar{X}_{i,m}(k)$, that is the discrete-time model described in the following Theorem 4.2.

From Theorem 4.1 it holds that $A^m_{i,j,d}(k)$ and $X_{i,m}(k)$ are statistically independent random variables. Then, in the model $X_{i,m}(k)$ in eq. 4.5, it holds that $A^m_{i,j,d}(k)$ and $X_{j,m}(k)$ are s.i. random variables, for each $j \in F_i$.

Theorem 4.2. *Let $\bar{A}_{i,j,d}([k])$ be the expected value of the random variable $A^m_{i,j,d}(k)$. Then:*

$$\bar{X}_{i,m}(k+1) = \sum_{j \in F_i, d=0}^{D} \bar{A}_{i,j,d}([k]) \bar{X}_{j,m}(k-d) \tag{4.7}$$

The proof of Theorem 4.2 is in Appendix B.

The average model $\bar{X}_{i,m}(k)$ allows computing the expected value of the retweeting rate of user i at the time slot k with only one simulation.

Furthermore, we observe that there is no need to know the probability distribution of each random variable $A^m_{i,j,d}(k)$, because only its expected value $\bar{A}_{i,j,d}([k])$ is used in the computation.

4.5.4 Error Dynamic Model

In this section we define the error model (eq. 4.8) to compute the difference between the average retweeting rate $\bar{X}_{i,m}(k)$ and the rate $X_{i,m}(k)$ resulting from the models, respectively in 4.7 and in 4.5.

$$\Delta_{i,m}(k) = X_{i,m}(k) - \bar{X}_{i,m}(k) \tag{4.8}$$

Let us consider the time slot $k+1$. Let $\varepsilon^m_{i,j,d}(k) = A^m_{i,j,d}(k) - \bar{A}_{i,j,d}([k])$ be the distance between the random variable $A^m_{i,j,d}(k)$ value and its expected value. It holds:

$$\begin{aligned}
\Delta_{i,m}(k+1) &= X_{i,m}(k+1) - \bar{X}_{i,m}(k+1) \qquad (4.9)\\
&= \sum_{j\in F_i,d=0}^{D} A^m_{i,j,d}(k)X_{j,m}(k-d) - \sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])\bar{X}_{j,m}(k-d)\\
&= \sum_{j\in F_i,d=0}^{D} (\bar{A}_{i,j,d}([k]) + \varepsilon^m_{i,j,d}(k))X_{j,m}(k-d)\\
&- \sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])\bar{X}_{j,m}(k-d)\\
&= \sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])(X_{j,m}(k-d) - \bar{X}_{j,m}(k-d))\\
&+ \sum_{j\in F_i,d=0}^{D} \varepsilon^m_{i,j,d}(k)X_{j,m}(k-d)\\
&= \sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])\Delta_{j,m}(k-d) + \sum_{j\in F_i,d=0}^{D} \varepsilon^m_{i,j,d}(k)X_{j,m}(k-d)
\end{aligned}$$

4.5.4.1 Average Error Dynamic Model

We prove that the expected value of the error defined in eq. 4.8 is null. First of all, we show, in eq. 4.10, that the expected value of $\varepsilon^m_{i,j,d}(k)$, defined in the previous section, is null.

$$\begin{aligned}
\mathbb{E}[\varepsilon^m_{i,j,d}(k)] &= \mathbb{E}[A^m_{i,j,d}(k) - \bar{A}_{i,j,d}([k])] \qquad (4.10)\\
\mathbb{E}[\varepsilon^m_{i,j,d}(k)] &= \bar{A}_{i,j,d}([k]) - \bar{A}_{i,j,d}([k])\\
\mathbb{E}[\varepsilon^m_{i,j,d}(k)] &= 0
\end{aligned}$$

It follows that:

$$\begin{aligned}\mathbb{E}[\Delta_{i,m}(k+1)] &= \mathbb{E}[\sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])\Delta_{j,m}(k-d) + \sum_{j\in F_i,d=0}^{D} \varepsilon^{m}_{i,j,d}(k)X_{j,m}(k-d)] \quad (4.11)\\ &= \sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])\mathbb{E}[\Delta_{j,m}(k-d)] + \sum_{j\in F_i,d=0}^{D} \mathbb{E}[\varepsilon^{m}_{i,j,d}(k)X_{j,m}(k-d)]\\ &= \sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])\mathbb{E}[\Delta_{j,m}(k-d)] + \sum_{j\in F_i,d=0}^{D} \mathbb{E}[\varepsilon^{m}_{i,j,d}(k)]\mathbb{E}[X_{j,m}(k-d)]\\ &= \sum_{j\in F_i,d=0}^{D} \bar{A}_{i,j,d}([k])\mathbb{E}[\Delta_{j,m}(k-d)] \qquad (4.12)\end{aligned}$$

Proposition 4.1. *$\mathbb{E}[\Delta_{i,m}(k)] = 0$, for each $i \in Nodes$ and for each time slot k.*

The proof of Proposition 4.1 is given in Appendix C.

4.6 Computing User Model Parameters from Data

In Section 4.5.2 we introduced the model $X_{i,m}(k)$ for the retweeting rate of each user i of the Twitter network. But we don't have discussed how the stochastic input $A^{m}_{i,j,d}(k)$, representing the user model, is built. In Section 4.5.1 we said that for all messages $m \in M$ and for all the time slots k in the same bin, the $A^{m}_{i,j,d}(k)$ random variables are i.i.d. More formally, these $A^{m}_{i,j,d}(k)$ are sampled from the same probability density function $p_{i,j,d}([k], a)$.

Note that $p_{i,j,d}([k], a)$ is unknown *a priori*. However, we are able to compute $A^{m}_{i,j,d}(k)$ parameters, that is the mean $\bar{A}_{i,j,d}([k])$ and the variance $Var(A_{i,j,d}([k]))$, from an available dataset (see Section 4.8.1).

Remark 4.5. *As well as the expected value $\bar{A}_{i,j,d}([k])$, also the variance of the $A^{m}_{i,j,d}(k)$ random variable is computed for all messages $m \in M$ and depends only on the bin $[k]$. Thus, we denote this variance as $Var(A_{i,j,d}([k]))$.*

First of all, we need to compute $A^{m}_{i,j,d}(k)$, for each pair (i, j) of (follower, followee), for $m \in \mathcal{M}$, for $d = 0, \dots, D$ and for the time slot k. Each time slot may represent one hour or a subset of hours in the time frame $\mathcal{T}$. Thus, let us define $X_{i,m}(k)$ through a new variable $W^{m}_{i,j}(k)$, defined in eq. 4.14, representing the rate of messages retweeted by i at the time slot k and received from j, as below.

$$X_{i,m}(k) = \sum_{j\in F_i} W^{m}_{i,j}(k) \qquad (4.13)$$

$$W^{m}_{i,j}(k) = \sum_{d=0}^{D} A^{m}_{i,j,d}(k)X_{j,m}(k-d) \qquad (4.14)$$

Let us replace each addend in the summation of eq. 4.14 with the random variable $W^{m}_{i,j,d}(k)$, defined in eq. 4.15. This variable is the rate of instances of message $m \in \mathcal{M}$ received from j and retweeted by i at the time slot k with delay d. $W^m_{i,j,d}(k)$ is available from the dataset.

$$W^m_{i,j,d}(k) = A^m_{i,j,d}(k) X_{j,m}(k-d) \tag{4.15}$$

Note that $A^m_{i,j,d}(k)$ in eq. 4.15 represents a fraction of the rate of instances of $m \in \mathcal{M}$ retweeted by user j at the time slot $k-d$. Then, $A^m_{i,j,d}(k)$ can be computed from the dataset as in eq. 4.16.

$$A^m_{i,j,d}(k) = \frac{W^m_{i,j,d}(k)}{X_{j,m}(k-d)} \tag{4.16}$$

Summing up, by replacing $W^m_{i,j,d}(k)$ in eq. 4.14 and than, backwards, $W^m_{i,j}$ in eq. 4.13, we obtain:

$$W^m_{i,j}(k) = \sum_{d=0}^{D} W^m_{i,j,d}(k) \tag{4.17}$$

$$X_{i,m}(k) = \sum_{j \in F_i} \sum_{d=0}^{D} W^m_{i,j,d}(k) \tag{4.18}$$

$$= \sum_{j \in F_i} \sum_{d=0}^{D} A^m_{i,j,d}(k) X_{j,m}(k-d) \tag{4.19}$$

Now that we know how to compute $A^m_{i,j,d}(k)$, that is our *sample*, we are interested in computing the *sample mean*, $\bar{A}_{i,j,d}([k])$, and the *sample variance* $Var(A_{i,j,d}([k]))$.

In order to compute $\bar{A}_{i,j,d}([k])$, we need to sum all $A^m_{i,j,d}(k)$ values for all the time slots in the bin $[k]$ and for all messages, and then to divide by the number of messages and by the size of the bin, *i.e.*, Δ, as below:

$$\bar{A}_{i,j,d}([k]) = \frac{1}{|M|\Delta} \sum_{m \in M} \sum_{h \in [k]} \frac{W^m_{i,j,d}(h)}{X_{j,m}(h-d)} \tag{4.20}$$

As well, $Var(A_{i,j,d}([k]))$ can be computed as in the following eq. 4.21:

$$Var(A_{i,j,d}([k])) = \mathbb{E}[(A^m_{i,j,d}(k))^2] - [\bar{A}_{i,j,d}([k])]^2 \tag{4.21}$$

In eq. 4.21 we already know $[\bar{A}_{i,j,d}([k])]^2$, from eq. 4.20. While $\mathbb{E}[(A^m_{i,j,d}(k))^2]$ can be computed as in eq. 4.22

$$\mathbb{E}[(A^m_{i,j,d}(k))^2] = \frac{1}{|M|\Delta} \sum_{m \in M} \sum_{h \in [k]} \frac{[W^m_{i,j,d}(h)]^2}{[X_{j,m}(h-d)]^2} \tag{4.22}$$

However, we want to be confident that the sample mean and the sample variance so computed reflect the correspondent *population* mean μ and variance σ^2 of the $A^m_{i,j,d}(k)$ random variables. Accordingly, we are interested in computing two intervals of values containing, respectively, the sample mean $\bar{A}_{i,j,d}([k])$ and variance

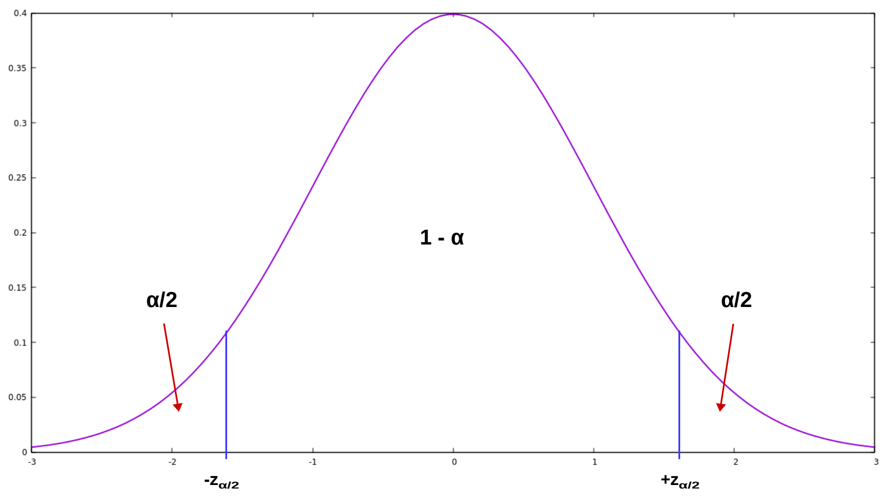


Figure 4.2. $100(1-\alpha)\%$ confidence interval, such that the area in each tails is $\alpha/2$, thus the sum of tails area is α.

$Var(A_{i,j,d}([k]))$ with a high *confidence level* $(1-\alpha)$, that is a percentage value between 0 and 100%. Each one of these intervals is called $100(1-\alpha)\%$ confidence interval and we are going to compute them with the approach in [57].

In Section 4.5.1 we made the hypothesis that, for each $k \in [k]$, the $A^m_{i,j,d}(k)$ random variables, representing the items of our sample, are i.i.d. As each bin $[k]$ contains a huge number n of time slots k $(n > 30)$, we can assert that our sample is of big size. Thus, thanks to the "Central Limit Theorem" we can say that the *sample mean* $\bar{A}_{i,j,d}([k])$ is a random variable following a Normal distribution, *i.e.*, $\bar{A}_{i,j,d}([k]) \sim \mathcal{N}(\mu, \sigma^2/n)$. The confidence level has to be identified in terms of the area $1-\alpha$ below the standard normal distributed random variable $(\bar{A}_{i,j,d}([k]) - \mu)/\sigma/\sqrt{n}$ and between two critical values $-z_{\alpha/2}$ and $-z_{\alpha/2}$, which are unknown (see Figure 4.2).

We want the probability of the following event to be at least $(1-\alpha)$:

$$\mathbb{P}\left[-z_{\alpha/2} < \frac{\bar{A}_{i,j,d}([k]) - \mu}{\sigma/\sqrt{n}} < z_{\alpha/2}\right] \geq (1-\alpha) \tag{4.23}$$

Computing the above probability corresponds to integrate the $\bar{A}_{i,j,d}([k])$ probability density function $p_{\bar{A}_{i,j,d}([k])}(\mu, \sigma^2/n, x)$ between $-z_{\alpha/2}$ and $z_{\alpha/2}$, as in eq. 4.24.

$$\int_{-z_{\alpha/2}}^{z_{\alpha/2}} p_{\bar{A}_{i,j,d}([k])}(\mu, \sigma^2/n, x)dx \geq (1-\alpha) \tag{4.24}$$

Note that the $p_{\bar{A}_{i,j,d}([k])}(\mu, \sigma^2/n, x)$ corresponds to the probability density function of the Normal distribution $\mathcal{N}(\mu, \sigma^2/n)$, that is:

$$p_{\bar{A}_{i,j,d}([k])}(\mu, \sigma^2/n, x) = \frac{1}{(\sigma/\sqrt{n})\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma/\sqrt{n}})^2/n} \tag{4.25}$$

$$= \frac{\sqrt{n}}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2/n}} \tag{4.26}$$

Let us split the integral in eq. 4.24 in two integrals and let us replace $p_{\bar{A}_{i,j,d}([k])}(\mu, \sigma^2/n, x)$ with the Normal density function of eq. 4.25, as in eq. 4.27. Note that thanks to the Normal distribution symmetric property, the two integrals are equals.

$$\int_{-z_{\alpha/2}}^{0} \frac{\sqrt{n}}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2/n}} dx + \int_{0}^{z_{\alpha/2}} \frac{\sqrt{n}}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2/n}} = \tag{4.27}$$

$$2\int_{0}^{z_{\alpha/2}} \frac{\sqrt{n}}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2/n}} dx = \tag{4.28}$$

$$2\frac{\sqrt{n}}{\sigma\sqrt{2\pi}} \int_{0}^{z_{\alpha/2}} e^{-\frac{(x-\mu)^2}{2\sigma^2/n}} dx \geq (1-\alpha) \tag{4.29}$$

Note that the integral in eq. 4.29 contains the known *Gaussian error function*, as shown below:

$$2\frac{\sqrt{n}}{\sigma\sqrt{2\pi}} \int_{0}^{z_{\alpha/2}} e^{-\frac{(x-\mu)^2}{2\sigma^2/n}} dx = \tag{4.30}$$

$$2\sqrt{n}\frac{1}{\sigma\sqrt{2\pi}} \int_{0}^{z_{\alpha/2}} e^{-\frac{(x-\mu)^2}{2\sigma^2/n}} = \tag{4.31}$$

$$2\sqrt{n}\frac{1}{2} erf\left(\frac{z_{\alpha/2}-\mu}{\sqrt{2}\sigma/\sqrt{n}}\right) = \tag{4.32}$$

$$\sqrt{n} erf\left(\frac{z_{\alpha/2}-\mu}{\sqrt{2}\sigma/\sqrt{n}}\right) \tag{4.33}$$

Thus, after setting α, by solving the eq. 4.34, we find the critical values $z_{\alpha/2}$ and the symmetric $-z_{\alpha/2}$. Note that the eq. 4.34 has to be solved numerically, as it cannot be solved analytically.

$$\sqrt{n} erf\left(\frac{z_{\alpha/2}-\mu}{\sqrt{2}\sigma/\sqrt{n}}\right) \geq (1-\alpha) \tag{4.34}$$

The critical values let us compute the probability in eq. 4.23, which can be rewritten as in eq. 4.35.

$$\mathbb{P}\left[\bar{A}_{i,j,d}([k]) - \frac{z_{\alpha/2}\sigma}{\sqrt{n}} < \mu < \bar{A}_{i,j,d}([k]) + \frac{z_{\alpha/2}\sigma}{\sqrt{n}}\right] \geq (1-\alpha) \tag{4.35}$$

As $\bar{A}_{i,j,d}([k])$ population standard deviation σ is unknown, we replace σ with its maximum likelihood estimator, that is the standard deviation S resulting from the sample, $S = \sqrt{Var(A_{i,j,d}([k]))}$ (see [57]), as below.

$$\mathbb{P}\left[\bar{A}_{i,j,d}([k]) - z_{\alpha/2}\frac{S}{\sqrt{n}} \leq \mu \leq \bar{A}_{i,j,d}([k]) + z_{\alpha/2}\frac{S}{\sqrt{n}}\right] \geq (1-\alpha) \tag{4.36}$$

Furthermore, again according to the approach in [57], given our sample variance, *i.e.*, $S^2 = Var(A_{i,j,d}([k]))$, the random variable $\frac{nS^2}{\sigma^2}$ follows a χ^2 distribution with $n-1$ degrees of freedom, *i.e.*, $\frac{nS^2}{\sigma^2} \sim \chi^2(n-1)$.

Thus a $100(1-\alpha)\%$ confidence interval for the $A^m_{i,j,d}(k)$ population variance σ^2 has to be computed by solving the eq. 4.37, to find two critical values a and b:

$$\mathbb{P}\left[a < \frac{nS^2}{\sigma^2} < b\right] = \tag{4.37}$$

$$\mathbb{P}\left[\frac{nS^2}{b} < \sigma^2 < \frac{nS^2}{a}\right] \geq (1-\alpha) \tag{4.38}$$

Note that a and b are the appropriate right-hand and left-hand of a $\chi^2(n-1)$ distribution. Solving the eq. 4.37 corresponds to compute the integral of the $\chi^2(n-1)$ density function $p_{\chi^2(n-1)}(x)$ between a and b, as below.

$$\int_a^b p_{\chi^2(n-1)}(x)dx = \int_a^b \frac{1}{2^{(n-1)/2}\Gamma((n-1)/2)} x^{(n-1)/2-1} e^{-x/2} dx \tag{4.39}$$

The data-driven approach so far described is useful to compute good statistics (mean and variance) for the *population* of the $A^m_{i,j,d}(k)$ random variables from a set of observations (*i.e.*, the *sample*), that is represented by our dataset.

However, we want to use the user model $A^m_{i,j,d}(k)$ to predict the retweeting rate $X_{i,w}(k)$ of each node i with respect to a new message $w \notin \mathcal{M}$. Accordingly, we want to take advantage of the statistics computed on the network with respect to a set of already seen messages. This will be discussed in the following section.

4.6.1 User Model Parameters for Prediction

When a new message w is spread into the network, for each user i we want to use the model defined in Theorem 4.2 to compute the average retweeting rate of w, for each time slot k.

For this purpose, we will exploit the knowledge acquired from the already observed messages. Formally, we will use the user model parameters $\bar{A}_{i,j,d}([k])$ and $Var(A_{i,j,d}([k]))$, computed from already seen messages $m \in M$, in the model defined in eq. 4.7, to obtain $\bar{X}_{i,w}(k)$.

4.7 Predicting Retweeting Rates for a New Message

In this section, we put all together the models presented so far and we summarize our strategy to predict the average retweeting rate of a message w, spread inside Twitter, for a user.

The retweeting rate model, $X_{i,w}(k)$, defined in Section 4.5.2 is built on a model, $A^{w}_{i,j,d}(k)$ (see Section 4.5.1), representing the stochastic input. The random variable $A^{w}_{i,j,d}(k)$ represents the user i retweeting activity with respect to the message w and its followees. In the previous Section 4.6, we showed how to compute $A^{w}_{i,j,d}(k)$ mean and variance from the dataset, respectively through eq. 4.20 and eq. 4.21.

In the following Algorithm 9, we show our algorithm to predict, for each user i and for each time slot k, the average $\bar{X}_{i,w}(k)$ retweeting rate of message w.

```
function $avgRateX(i, w, k, D)$
  if $k <= D$ then
    return $\bar{X}_{i,w}(k)$
  if $k > D$ then
    $\bar{X}_{i,w}(k) \leftarrow \sum_{j \in F_i, d=0}^{D} \bar{A}_{i,j,d}([k])\bar{X}_{j,w}(k-1-d)$
    return $\bar{X}_{i,w}(k)$
```

Algorithm 9: Average $\bar{X}_{i,w}(k)$ prediction. This function implements the eq. 4.7 of Section 4.5.3. If $k \in \{0, \ldots, D\}$, the average retweeting rate has to be computed from the observed data. If $k > D$, the average $\bar{A}_{i,j,d}([k])$ of the random variable $A^{w}_{i,j,d}(k)$ has to be computed from the dataset according to eq. 4.20

4.8 Experiments

In this section, we describe the Twitter dataset and we present the experiments we performed to validate our strategy (discussed in Section 4.7) to predict $\bar{X}_{i,m}(k)$, for each message $m \in \mathcal{M}$ and each user i.

4.8.1 Data Set Description

The data analyzed in this work are taken from a public dataset available for research purposes [55]. However, this dataset can be rebuilt also by the Twitter APIs, that allow to get all users of a subnet of Twitter and, for each user, the list of its followers and followees.

Our dataset contains tweets with embedded URLs posted on Twitter over 12 days of October 2010. More in detail: 2859760 tweets for 66059 different messages. The URLs in the messages are usually shortened by some service, such as *newzfor.me* or *tinyurl*, and are used as markers for information spreading into the network. Thus, messages containing the same URL are considered equal to each other. Most of the messages are retweeted a few times, that is, in a range of $[0, 50]$, while a small set of messages are retweeted more than 50 times (see Figure 4.3). There are 736930 users, most of whom retweeting less than 2000 times, while few of them retweet more (see Figure 4.4). Finally, there are 36743400 pairs (i, j) of *(follower, followee)*. In summary, from the above analysis, it comes out that in our dataset there is a small number of very active users and a small number of very popular messages.

We performed our experiments by selecting the messages retweeted from at least one pair of (follower, followee) retweeting at least other 34 messages. The 35 selected

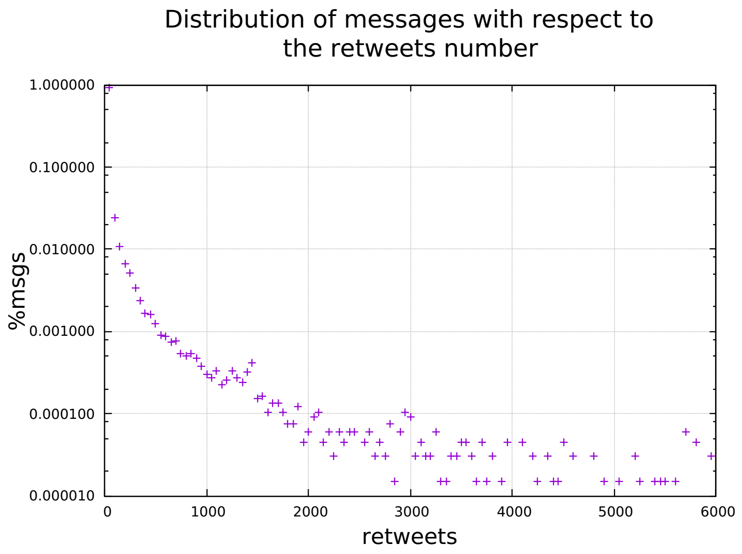


Figure 4.3. Graphic showing how most messages in the dataset are retweeted a few times, while just a few messages are retweeted more than 50 times.

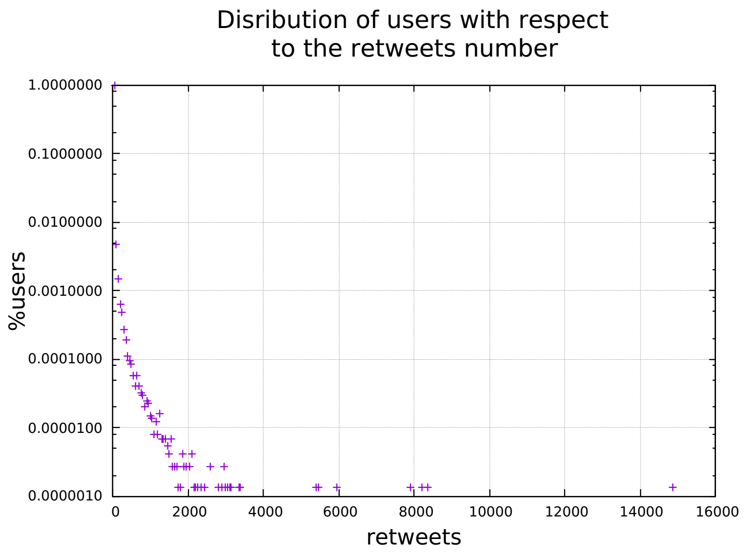


Figure 4.4. Graphic showing how most users retweet less than 2000 times, while a few number of users retweet more than 2000 times.

messages are summarized in the following Table 4.2. The total amount of users in the subnetwork identified by the selected messages contains 989 nodes (users). We assigned a numeric identifier to each distinct message. In Table 4.3 are summarized the relevant properties of each selected message m, that is: its average number of retweets for user; the number of times when m is retweeted; the total number of users and the total number of pairs retweeting m. In particular, the average number of retweets for user suggests that every user retweets each message m on average 1 time, which is a realistic Twitter activity scenario.

URL	Msg id
http://newzfor.me/?7h8p	305
http://newzfor.me/?78zo	440
http://newzfor.me/?735m	506
http://newzfor.me/?7mbo	512
http://newzfor.me/?75ln	523
http://newzfor.me/?7d7o	536
http://newzfor.me/?7amo	577
http://newzfor.me/?7zio	623
http://newzfor.me/?7ibo	679
http://newzfor.me/?7x4p	786
http://newzfor.me/?7w2o	858
http://newzfor.me/?7azo	864
http://newzfor.me/?7dqn	891
http://newzfor.me/?7dap	932
http://newzfor.me/?7c4m	939
http://newzfor.me/?7xtn	977
http://newzfor.me/?7a6p	981
http://newzfor.me/?7skn	990
http://newzfor.me/?7tjo	1030
http://newzfor.me/?71bm	1052
http://newzfor.me/?7a9o	1086
http://newzfor.me/?752p	1241
http://newzfor.me/?7cqn	1407
http://newzfor.me/?70zo	1439
http://newzfor.me/?7nbo	1516
http://newzfor.me/?7pqp	1525
http://newzfor.me/?74ap	1670
http://newzfor.me/?799o	1680
http://newzfor.me/?7trp	1700
http://newzfor.me/?7hbo	1704
http://newzfor.me/?727p	1752
http://newzfor.me/?7gcp	1839
http://newzfor.me/?78rp	2040
http://newzfor.me/?7san	2662
http://newzfor.me/?7s3p	3030

Table 4.2. Selected messages table: the url identifying the message and the assigned numeric identifier.

We split the 35 messages in Table 4.2 into two subsets, that is: a training set $\mathcal{M}_{training}$ of 30 messages and a test set $\mathcal{M}_{test}$ of 5 messages. We don't need to hold back also a validation set because our average retweeting rate model, namely $\bar{X}_{i,w}(k)$, has no hyperparameters to be tuned (see [102]). This means that if we would validate our model without success, we should define a different model.

msg id	avg retweet	#retweets	#users	#pairs
305	1.689	1079	639	439
440	1.481	828	559	313
506	1.209	689	570	274
512	1.283	716	558	267
523	1.298	696	536	262
536	1.256	662	527	249
577	1.298	667	514	228
623	1.723	691	401	207
679	1.327	584	440	186
786	1.048	499	479	154
858	1.672	510	305	140
864	1.682	523	311	139
891	1.318	447	339	134
932	1.015	458	451	126
939	1.097	439	400	126
977	1.247	444	356	120
981	1.118	418	376	119
990	1.109	406	366	118
1030	1.000	411	411	114
1052	1.443	417	289	112
1086	1.522	504	331	108
1241	1.233	376	305	90
1407	1.383	343	248	77
1439	1.101	294	267	75
1516	1.003	283	282	70
1525	1.006	313	311	69
1670	1.004	267	266	62
1680	1.096	331	302	62
1700	1.093	294	269	61
1704	1.217	314	258	61
1752	1.000	294	294	58
1839	1.004	248	247	54
2040	1.000	268	268	46
2662	1.143	168	147	30
3030	1.000	157	157	24

Table 4.3. Selected messages properties. For each message m: the average retweets number for user; m retweets total number; the total number of users reetweeting m; the total number of pairs exchanging m..

4.8.2 Definition of the Experiments

We choose time slots k of size $\tau = 1$ hour. Then, we perform the following experiments:

(1) First, we perform a *sanity check*, with the aim to show that our model is expressive with respect to already seen messages. For each training message $m \in \mathcal{M}_{training}$, we observe its evolution in the first D (*i.e.*, the maximum delay time) time slots of the time frame $\mathcal{T}$, *i.e.*, for $t = t_0^{m*}, \ldots, t_0^{m*} + D$. Let

us recall that, for every message m, t_0^{m*} is the time slot t_0^m realigned to the first day of $\mathcal{T}$ (see Remark 4.1). Then, for each user i and for each time slot k, we compute the average expected value $\bar{A}^m_{i,j,d}([k])$ over all the time slots $k \in [k]$. Finally, we use $\bar{A}^m_{i,j,d}([k])$ to compute $\bar{X}_{i,m}(k)$, by Algorithm 9. As an expected result, the distance between the predicted and the observed average retweeting rate has to be small.

(2) Second, we perform our *model validation*. We pick a message $m \in \mathcal{M}_{training}$ and we observe the evolution of m in the first D time slots k of $\mathcal{T}$. Then, for each user i and for each time slot k, we compute the average expected value $\bar{A}_{i,j,d}([k])$ over all the training messages and all the time slots $k \in [k]$. Finally, we use $\bar{A}_{i,j,d}([k])$ to compute $\bar{X}_{i,m}(k)$, by Algorithm 9.

(3) Third, we execute the *prediction testing*. For each test message $w \in \mathcal{M}_{test}$, we observe w evolution in the first D time slots k of $\mathcal{T}$. Then we use $\bar{A}_{i,j,d}([k])$ to predict the average value $\bar{X}_{i,w}(k)$ by Algorithm 9, as in the previous experiment.

At the end of both the above experiments, we compute the error between the $\bar{X}_{i,m}(k)$ obtained by our strategy and the average rate obtained from the dataset. In Section 4.9.1 we will show the metrics we use to measure the error.

However, before presenting the experimental results about the average retweeting rate for each message, we are going to show, in Section 4.8.3, how we compute and validate the user model $A^m_{i,j,d}(k)$ parameters, *i.e.*, $\bar{A}_{i,j,d}([k])$ and $Var(A_{i,j,d})$.

4.8.3 Average User Model: Computation and Validation

In this section we show how we computed the user model parameters $\bar{A}_{i,j,d}([k])$ and $Var(A_{i,j,d}([k]))$, through, respectively, eq. 4.20 and eq. 4.21, in order to perform our experiments. First of all, we had to choose the size of the bin $[k]$. The trade-off was between a large size bin, that is a bin containing a huge number of time slots, and a small size bin (the minimum is 1 hour). We observed that large bins did not allow us to capture the users retweeting activity oscillations during the day. On the other hand, a small bin makes lower the statistical confidence about the $\bar{A}_{i,j,d}([k])$ parameters, because it reduces the number of samples in the confidence computation. In fact, although there is a high overlapping between the pairs (i, j) retweeting the messages $m \in M_{training}$, not many pairs retweet in time slots k contained in the same bin $[k]$ or with the same delay d. Thus, we choose bins of size $\Delta = 6$, because it turned out that our strategy performed reasonably well.

Let us recall that our dataset time frame $\mathcal{T}$ is made up of 12 days. As we chose time slots k of 1 hour and as each day contains 24 time slots, the total number of bins in $\mathcal{T}$ is $\mathcal{T} * 24/\Delta$.

Second, we chose a maximum delay value D of 12 hours, to be on the safe size. If after 12 hours a message m is not retweeted, according to our prediction strategy, the life message ends. If m is retweeted again after 12 hours, it is considered as a new message spread into the network, thus we need to perform a new simulation of our model, defined in eq. 4.7, to compute $\bar{X}_{i,m}(k)$.

In Figure 4.5 and Figure 4.6 we show the cumulative distribution of $\bar{A}_{i,j,d}([k])$ and $Var(A_{i,j,d}([k]))$ values. Note that with probability 1 the average values are less than 0.16 while the variance values are less than 0.14. We show also the confidence intervals width distribution, computed both for the mean $\bar{A}_{i,j,d}([k])$ (Figure 4.7) and the variance $Var(A_{i,j,d}([k]))$ (see Figure 4.8).

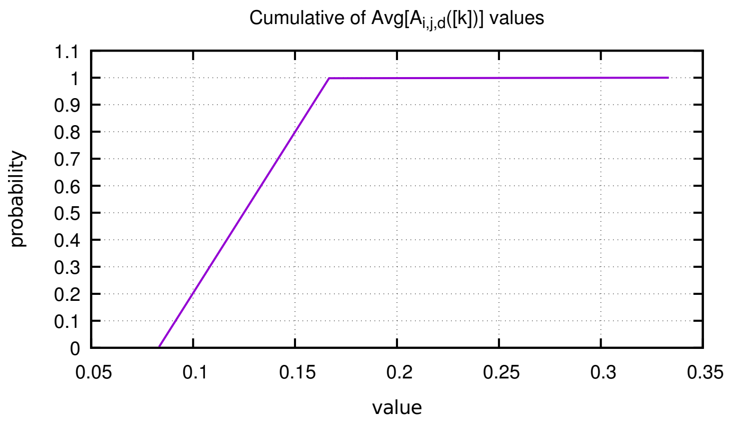


Figure 4.5. Graphic showing the cumulative distribution of the $\bar{A}_{i,j,d}([k])$ values. With probability 1 the average values are less than 0.16.

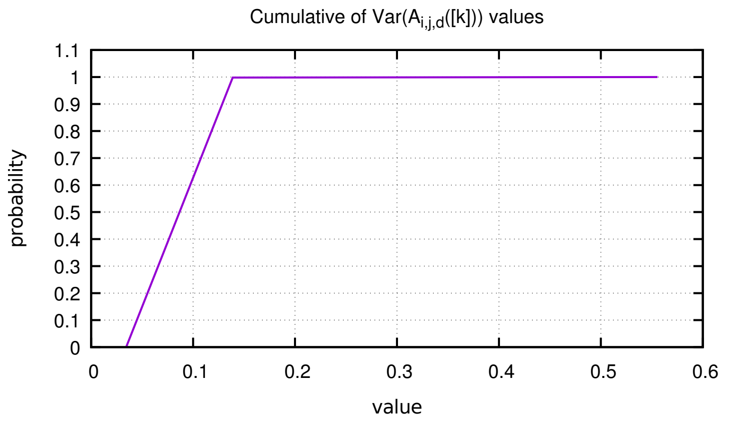


Figure 4.6. Graphic showing the cumulative distribution of the $Var(A_{i,j,d}([k]))$ values. With probability 1 the variance values are less than 0.14.

4.9 Experimental Results

In this section, we show our experimental results and the evaluation metrics used.

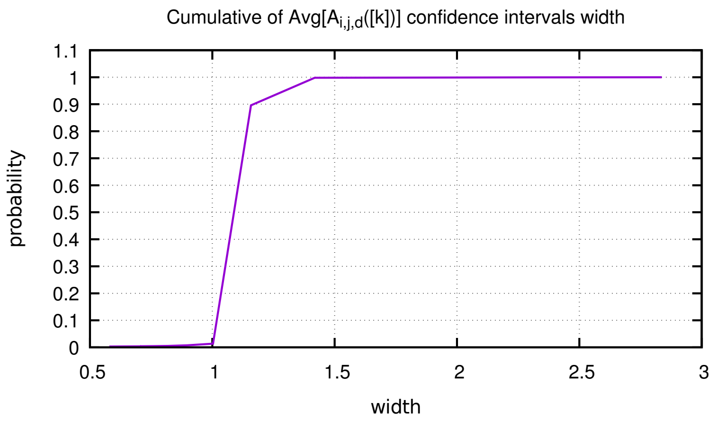


Figure 4.7. Graphic showing the cumulative distribution of the width of the 95% confidence intervals computed for the mean $\bar{A}_{i,j,d}([k])$, for each pair (i, j) of users, for each delay d, for each bin $[k]$.

4.9.1 Evaluation Metrics

We use the following metrics to evaluate the accuracy of our prediction results. See [111] for more details.

Let $\bar{X^*}_{i,m}(k)$ be the predicted average retweeting value for the node i and the message m. Let $B = \mathcal{T} * 24/\Delta$ be the total number of bins in the time frame $\mathcal{T}$ (see Section 4.8.3).

We compute the **Root Mean Squared Error** (eq. 4.40) to evaluate the accuracy of the average retweeting rate predicted by our model, for each node i and for each message $m \in \mathcal{M}$.

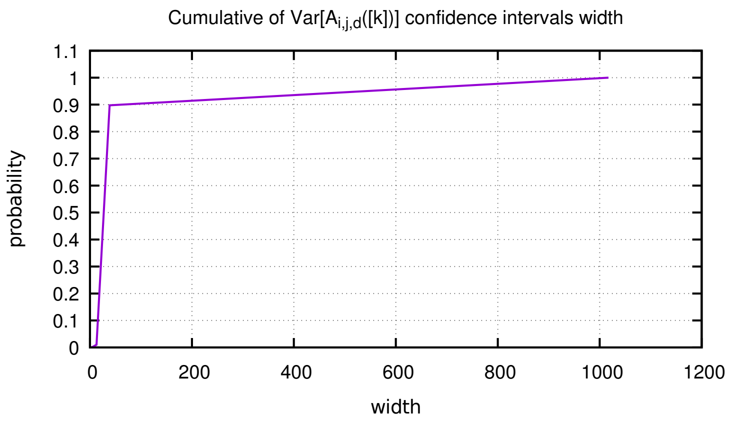


Figure 4.8. Graphic showing the cumulative distribution of the width of the 95% confidence intervals computed for the variance $Var(A_{i,j,d}([k]))$, for each pair (i, j) of users, for each delay d, for each bin $[k]$. As to be expected, the confidence intervals for the variance are quite wide.

$$RMSE_{i,m} = \sqrt{\frac{1}{B\Delta}\sum_{k=0}^{B\Delta}(\bar{X}_{i,m}(k) - \bar{X}^{*}{}_{i,m}(k))^2} \tag{4.40}$$

We compute also the **Mean Absolute Error**, as in eq. 4.41, that measures the average absolute error between the observed and the predicted average retweeting rate.

$$MAE_{i,m} = \frac{1}{B\Delta}\sum_{k=0}^{B\Delta}\left|\bar{X}_{i,m}(k) - \bar{X}^{*}{}_{i,m}(k)\right| \tag{4.41}$$

Furthermore, we compute the **Normalized Mean Absolute Error** error with respect to the observed average rate value, as in eq. 4.42.

$$NMAE_{i,m} = \frac{MAE}{\frac{1}{B\Delta}\sum_{k=0}^{B\Delta}\bar{X}_{i,m}(k)} \tag{4.42}$$

As it is a very important measure, we compute also the **Mean Absolute Percentage Error**, that measures the average absolute distance between the predicted and the real average retweeting rate. The MAPE is a percentage error and is defined in eq. 4.43. When $\bar{X}_{i,m}(k) = 0$, we replace it with a *fudge factor* of 10^{-3}.

$$MAPE_{i,m} = \frac{1}{B\Delta}\sum_{k=0}^{B\Delta}\left|\frac{\bar{X}_{i,m}(k) - \bar{X}^{*}{}_{i,m}(k)}{\bar{X}_{i,m}(k)}\right| \tag{4.43}$$

Finally, we compute the APE measuring the distance between the forecast and the real retweeting users' number of a message m at each time slot k, as in the following eq. 4.44. If $\#users_{i,m}(k) = 0$, we replace it with a *fudge factor* of 1.

$$APE_{\#users,m}(k) = \left|\frac{\#users_m(k) - \#users^{*}_m(k)}{\#users_m(k)}\right| \tag{4.44}$$

4.9.2 Evaluation Metrics for Set of Users

Many retweeting activity properties are related, in particular, to a set of users, *i.e.*, nodes of the Twitter network. In fact, in different fields [35, 70] it is interesting to detect the most active user in a subset of the network. Of course, it is always possible to analyze all the nodes, as we have done so far. However, sometimes it can be useful to focus, in a first moment, on a subset of n nodes and then, eventually, on other subsets.

Then, we use the metrics defined in the previous Section 4.9.1 to compute the expected values of RMSE, MAE, NMAE and MAPE for set of users. Let $N = |Nodes|$ be the total number of users in the network. First, we fix the size n of the set we analyze, for $n = \frac{1}{10}N, \frac{1}{100}N, N$. Second, we select n random users. Finally, we compute the expected value of the above mentioned errors, through the equations below.

$$\mathbb{E}[RMSE] = \frac{1}{n}\sum_{i=1}^{n} RMSE_{i,m} \tag{4.45}$$

$$\mathbb{E}[MAE] = \frac{1}{n}\sum_{i=1}^{n} MAE_{i,m} \tag{4.46}$$

$$\mathbb{E}[NMAE] = \frac{1}{n}\sum_{i=1}^{n} NMAE_{i,m} \tag{4.47}$$

$$\mathbb{E}[MAPE] = \frac{1}{n}\sum_{i=1}^{n} MAPE_{i,m} \tag{4.48}$$

4.9.3 Predictions Results

We run our prediction Algorithm 9 to compute $\bar{X}_{i,m}(k)$, for each user i and for each message $m \in \mathcal{M}_{training} \cup \mathcal{M}_{test}$. Then, we compute the prediction errors, according to the metrics defined in the previous Section 4.9.1 and Section 4.9.2. Note that we could run our algorithm indefinitely. However, we stop running at the time slot 288, *i.e.*, the last time slot of the dataset time frame $\mathcal{T}$, because we need to compare the predicted rate with the rate observed from the dataset. Note that the CPU time and the RAM usage of our algorithm are both negligible.

In the next subsections, we present the results about: the *model validation* Item (2) and the *sanity check* experiments Item (1), involving only the training messages; the *prediction testing* experiment Item (3), involving the test messages. For each experiment, for each message m, we show the plots related to: the forecast and the actual retweets number, at the time slot k and until the time slot k; the forecast and the actual number of retweeting users at the time slot k and until the time slot k.

As to be expected, our model predicts the qualitative behavior at the time slot k (missing the height of the peaks). On the other hand, as evidenced by the plots, until the time slot k our model predicts reasonably well also the quantitative behavior. This is confirmed by the Pearson correlation coefficients [17], that, especially for the retweet number and the retweeting users' number until k, show a high correlation between the predicted and the observed data. In fact, experimental results for the prediction testing show that: the correlation coefficient for the number of retweets until the time slot k is in the range $[0.89, 0.99]$; the correlation coefficient for the number of users until the time slot k is in the range $[0.68, 0.97]$ (see Table 4.6).

Furthermore, for each message, we show the plots of the cumulative distribution of the $RMSE_{i,m}$, the $MAE_{i,m}$, the $NMAE_{i,m}$ and the $MAPE_{i,m}$, and the $APE_{\#users,m}$ distribution (see Section 4.9.1) of the retweeting users' number with respect to the time slots. In particular, prediction testing experiment results show that: with probability 0.8 the $MAPE_{i,m}$ value is in the range $[0, 50\%]$ (Figure 4.104 and Figure 4.76); with probability 1 the $RMSE_{i,m}$ is less than a value in the range $[0.03, 0.065]$ (Figure 4.117 and Figure 4.61). The maximum APE value of the retweeting users' number until the timeslot k is between $[0.39, 4.6]$ (Figure 4.108 and Figure 4.80). For sake of completeness, for each message m we include also the plots of the $APE_{\#users,m}$ distribution at the time slot k. However, they are not relevant to our evaluation of the results, as we are more interested in predicting *until* a certain time slot k rather than *at* k.

Finally, for the model validation and the prediction testing experiments, we compute the expected values of RMSE, MAE, NMAE and MAPE for a set of users (see Section 4.9.2). We observe that, in general, as the set size increases the prediction errors decrease since they cancel out.

Experimental results for prediction testing show that the $\mathbb{E}[MAPE]$ of the retweeting rate computed for a set containing all the users of the network is in the range $[9.7\%, 39\%]$ (Figure 4.96 and Figure 4.82).

4.9.3.1 Sanity Check Results

In this section, we show some of the results for the sanity check experiment. For the complete set of results, see Appendix D.1.

We show in Table 4.4 the Pearson correlation coefficients, computed for each training message, after the sanity check experiment.

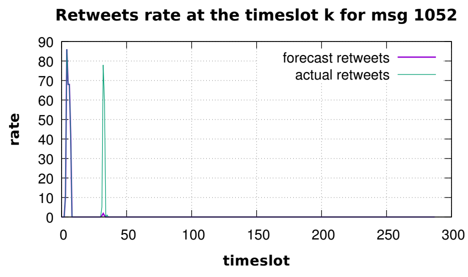


Figure 4.9. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1052.

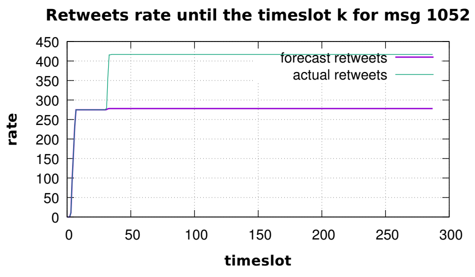


Figure 4.10. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1052.

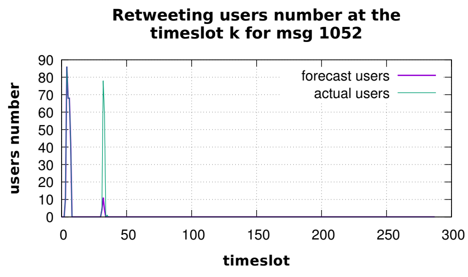


Figure 4.11. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1052.

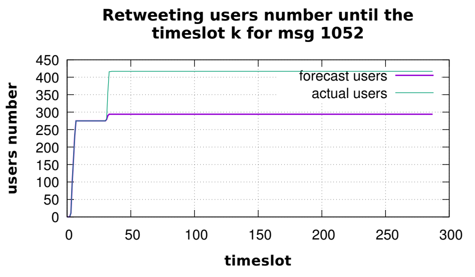


Figure 4.12. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1052.

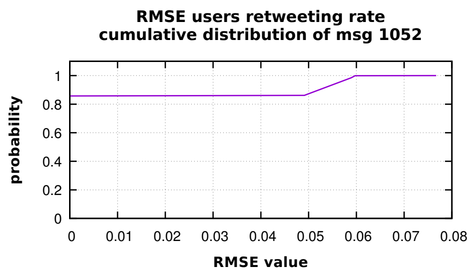


Figure 4.13. RMSE cumulative distribution, for message 1052. With probability 1 the $RMSE < 0.06$.

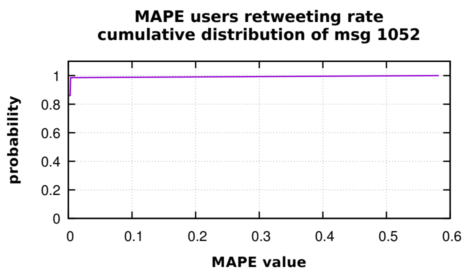


Figure 4.14. MAPE cumulative distribution, for message 1052. With probability 0.8 the $MAPE$ is 0

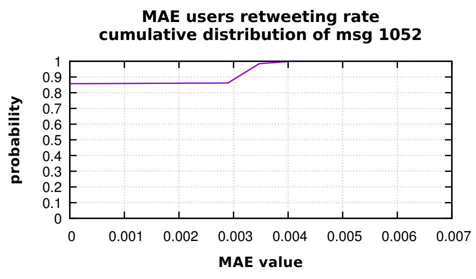


Figure 4.15. MAE cumulative distribution, for message 1052. With probability 1 the $MAE < 0.004$

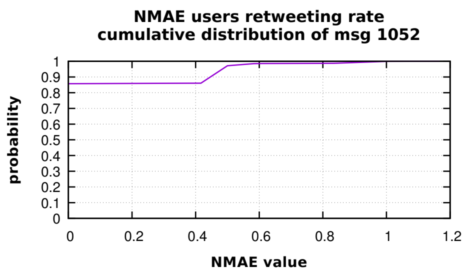


Figure 4.16. NMAE cumulative distribution, for message 1052. With probability 0.95 the $NMAE$ is 41%

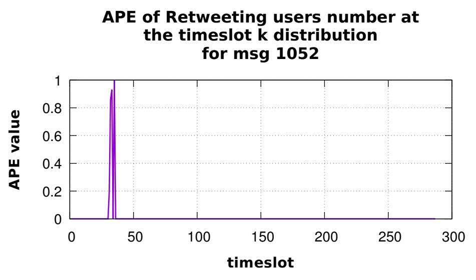


Figure 4.17. Distribution of APE on the retweeting users number at the time slot k, for message 1052.

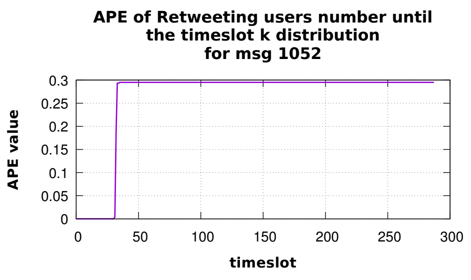


Figure 4.18. Distribution of APE on the retweeting users number until time slot k, for message 1052.

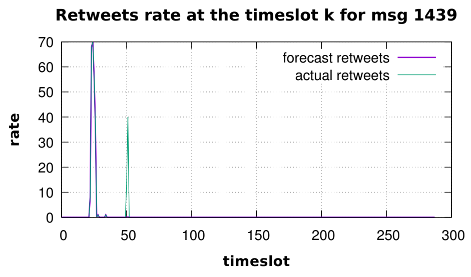


Figure 4.19. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1439.

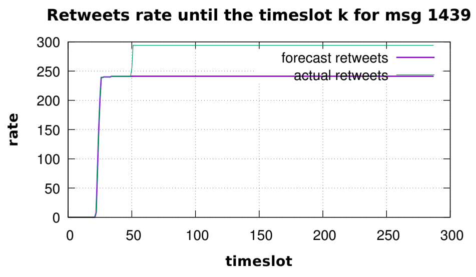


Figure 4.20. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1439.

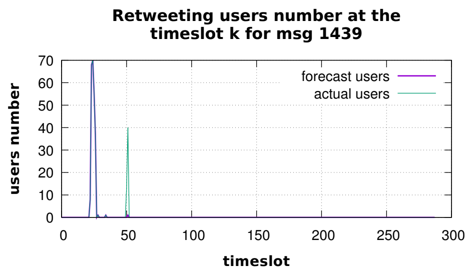


Figure 4.21. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1439.

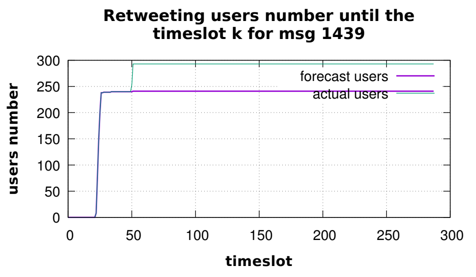


Figure 4.22. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1439.

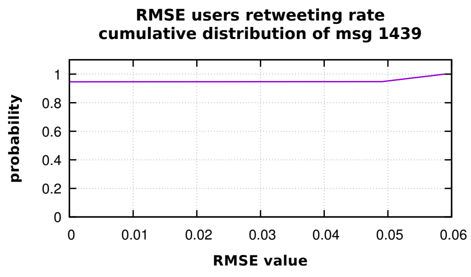


Figure 4.23. RMSE cumulative distribution, for message 1439. With probability 1 the $RMSE < 0.06$

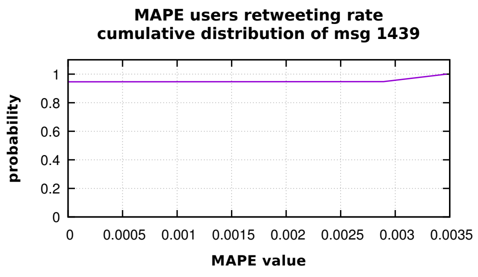


Figure 4.24. MAPE cumulative distribution, for message 1439. With probability 0.95 the $MAPE < 0.3\%$

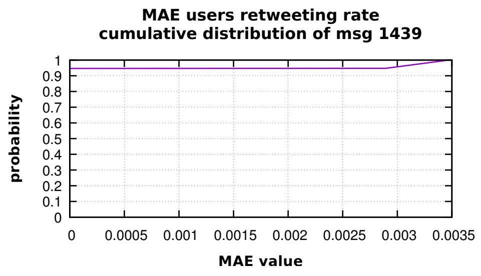


Figure 4.25. MAE cumulative distribution, for message 1439. With probability 1 the $MAE < 0.0035$

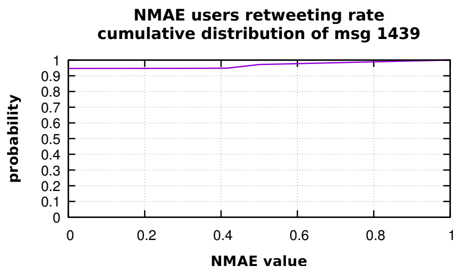


Figure 4.26. NMAE cumulative distribution, for message 1439. With probability 0.95 the $NMAE$ is 41%

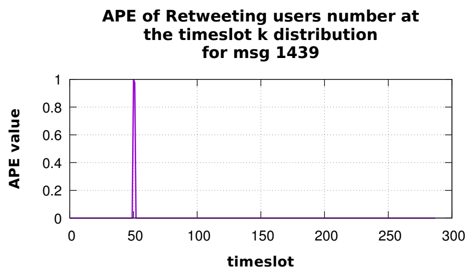


Figure 4.27. Distribution of APE on the retweeting users number at the time slot k, for message 1439.

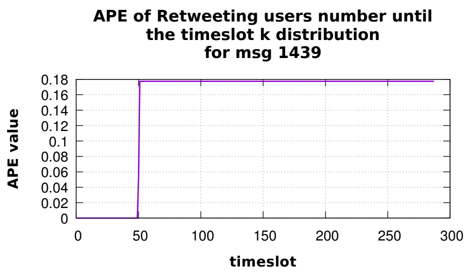


Figure 4.28. Distribution of APE on the retweeting users number until time slot k, for message 1439.

msg id	**#retweets at** k	**#retweets to** k	**#users at** k	**#users to** k
536	0.18414	0.56984	0.19925	0.69336
577	0.34681	0.76603	0.36853	0.85956
623	0.72795	0.80821	0.72674	0.85410
679	0.84705	0.92914	0.84609	0.94482
786	0.99947	0.99854	0.99946	0.99862
864	0.73894	0.90200	0.75880	0.94701
891	0.75001	0.93678	0.75489	0.94435
932	0.99990	0.99982	0.99990	0.99982
939	0.15492	0.50717	0.23536	0.81294
977	0.33325	0.81213	0.36654	0.85359
981	0.97969	0.98127	0.98058	0.98203
990	0.14189	0.72816	0.15870	0.77923
1030	1	1	1	1
1052	0.81855	0.80461	0.85613	0.87022
1086	0.68795	0.73432	0.70268	0.76205
1407	0.55916	0.81234	0.55817	0.83220
1439	0.94224	0.98271	0.94406	0.98325
1516	1	1	1	1
1525	0.99998	0.99999	0.99999	0.99999
1670	1	1	1	1
1752	1	1	1	1
2040	1	1	1	1
2662	0.37732	0.83300	0.37732	0.83300
3030	1	1	1	1
1680	0.12842	0.55011	0.14077	0.64092
305	0.82454	0.78161	0.84948	0.84724
440	0.57010	0.86370	0.65979	0.92617
506	0.31529	0.59354	0.38835	0.90141
512	0.89632	0.94838	0.89625	0.95979
523	0.39580	0.80953	0.40607	0.85799

Table 4.4. This table shows, for each training message, the Pearson correlation coefficients computed after the *sanity check* experiment (see Item (1)), related to: the number of retweets at the time slot k and until k; the number of retweeting users at k and until k.

4.9.3.2 Model Validation Results

In this section, we show some of the results for the model validation experiment. For the complete set of results, see Appendix D.2.

We show in Table 4.5 the Pearson correlation coefficients, computed for each training message, after the model validation experiment.

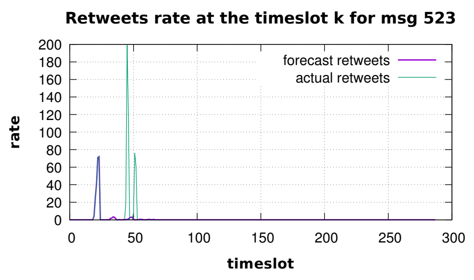


Figure 4.29. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 523.

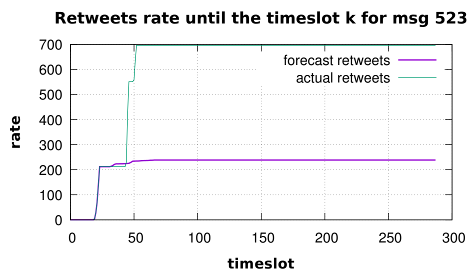


Figure 4.30. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 523.

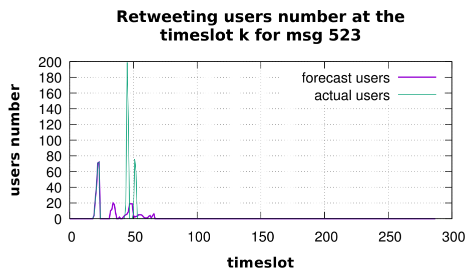


Figure 4.31. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 523.

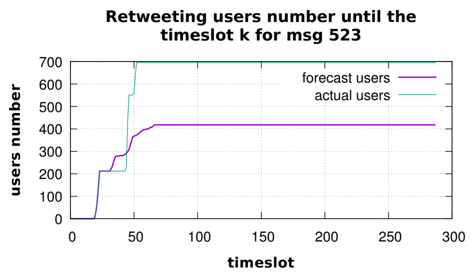


Figure 4.32. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 523.

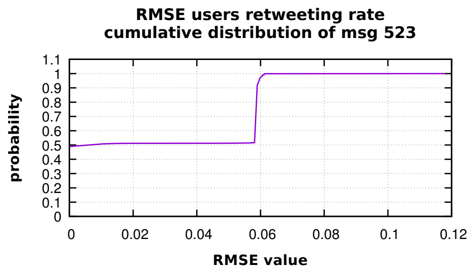


Figure 4.33. RMSE cumulative distribution, for message 523. With probability 1 the $RMSE < 0.06$

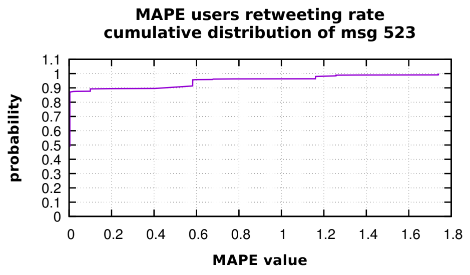


Figure 4.34. MAPE cumulative distribution, for message 523. With probability 0.89 the $MAPE$ is 10%

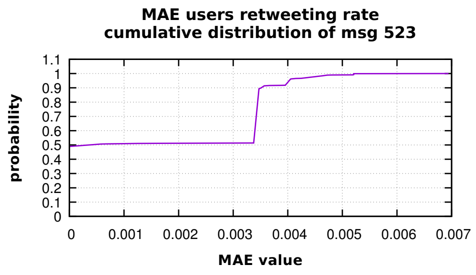


Figure 4.35. MAE cumulative distribution, for message 523. With probability 1 the $MAE < 0.007$

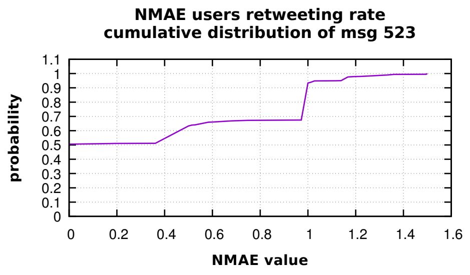


Figure 4.36. NMAE cumulative distribution, for message 523. With probability 0.65 the $NMAE$ is 50%

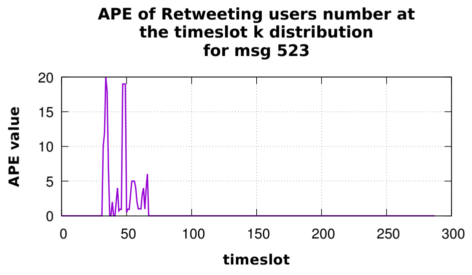


Figure 4.37. Distribution of APE on the retweeting users number at the time slot k, for message 523.

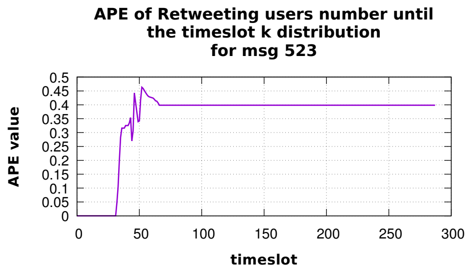


Figure 4.38. Distribution of APE on the retweeting users number until time slot k, for message 523.

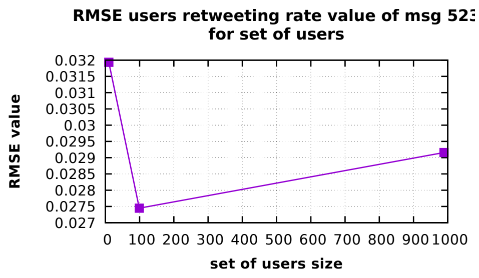


Figure 4.39. RMSE values distribution with respect to the set of users size, for message 523. When the set includes all network nodes, $RMSE = 0.029$

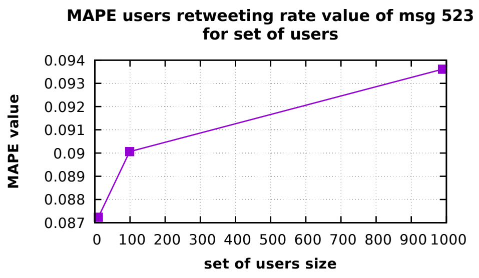


Figure 4.40. MAPE values distribution with respect to the set of users size, for message 523. When the set includes all network nodes, $MAPE = 0.0935$

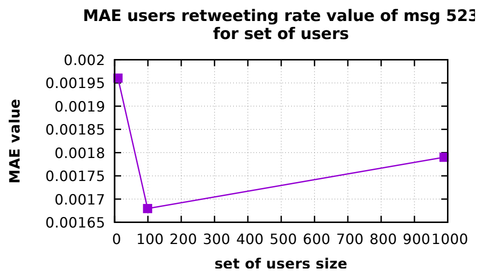


Figure 4.41. MAE values distribution with respect to the set of users size, for message 523. When the set includes all network nodes, $MAE = 0.0018$

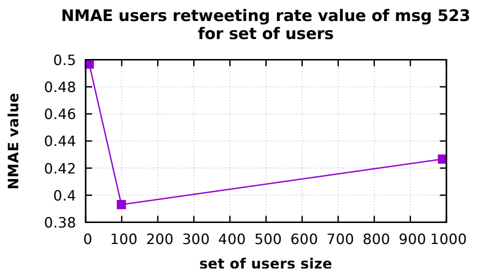


Figure 4.42. NMAE values distribution with respect to the set of users size, for message 523. When the set includes all network nodes, $NMAE = 0.0425$

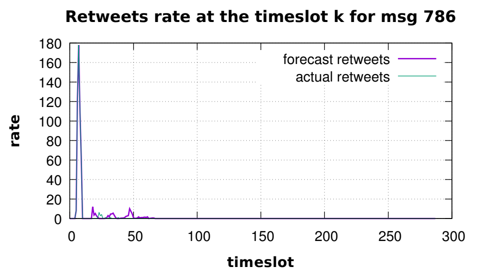


Figure 4.43. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 786.

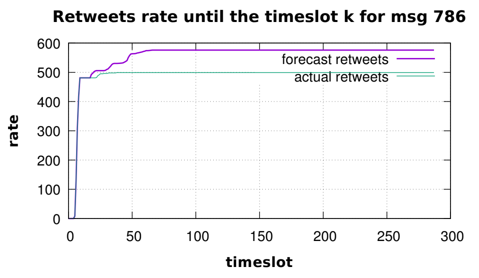


Figure 4.44. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 786.

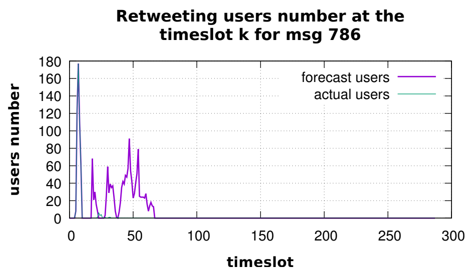


Figure 4.45. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 786.

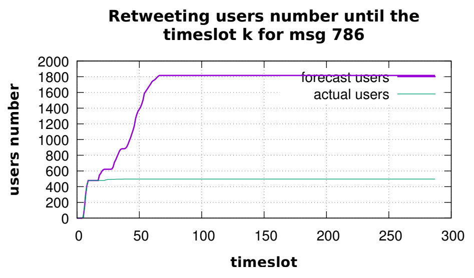


Figure 4.46. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 786.

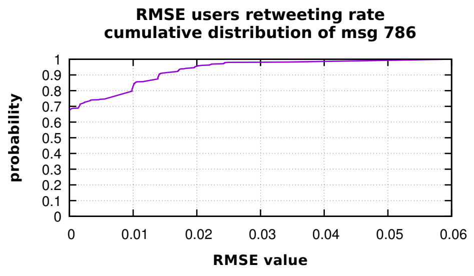


Figure 4.47. RMSE cumulative distribution, for message 786. With probability 1 the $RMSE < 0.06$

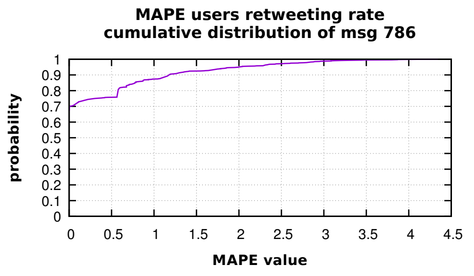


Figure 4.48. MAPE cumulative distribution, for message 786. With probability 0.75 the $MAPE$ is 10%

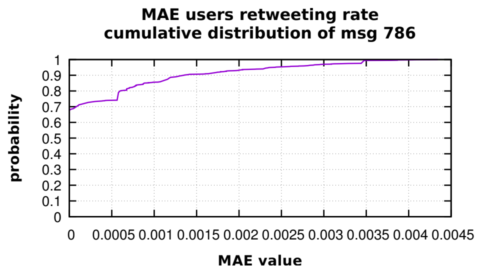


Figure 4.49. MAE cumulative distribution, for message 786. With probability 1 the $MAE < 0.004$

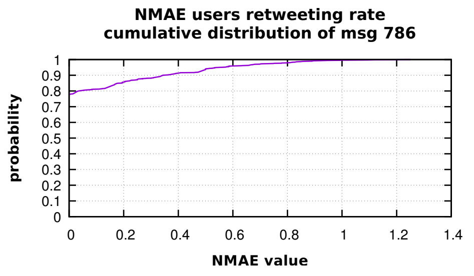


Figure 4.50. NMAE cumulative distribution, for message 786. With probability 0.85 the $NMAE$ is 20%

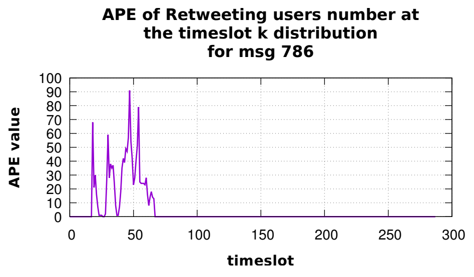


Figure 4.51. Distribution of APE on the retweeting users number at the time slot k, for message 786.

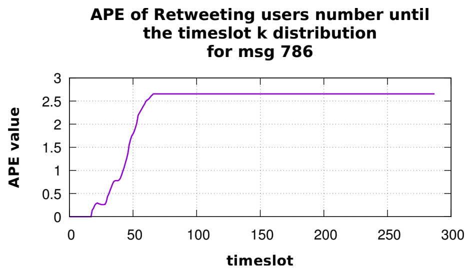


Figure 4.52. Distribution of APE on the retweeting users number until time slot k, for message 786.

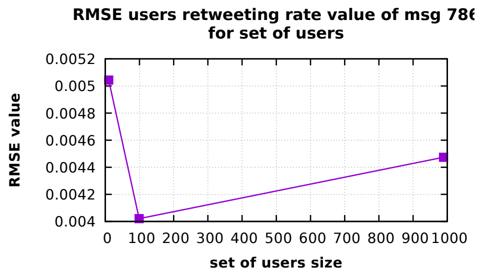


Figure 4.53. RMSE values distribution with respect to the set of users size, for message 786. When the set includes all network nodes, $RMSE = 0.0045$

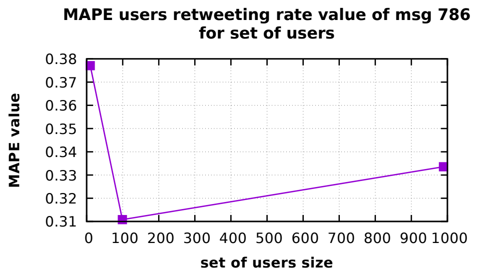


Figure 4.54. MAPE values distribution with respect to the set of users size, for message 786. When the set includes all network nodes, $MAPE = 0.335$

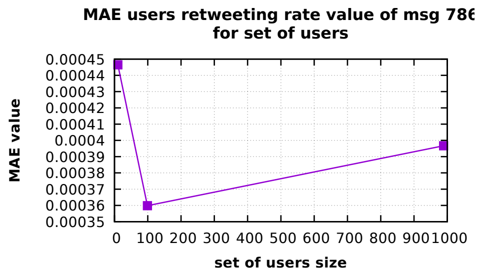


Figure 4.55. MAE values distribution with respect to the set of users size, for message 786. When the set includes all network nodes, $MAE = 0.0004$

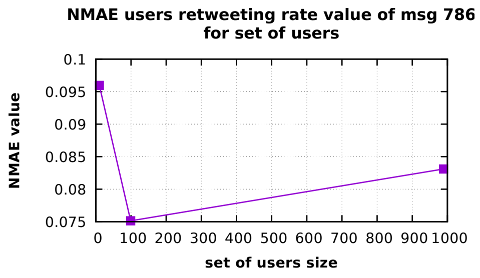


Figure 4.56. NMAE values distribution with respect to the set of users size, for message 786. When the set includes all network nodes, $NMAE = 0.083$

msg id	**#retweets at** k	**#retweets to** k	**#users at** k	**#users to** k
536	0.20739	0.76352	0.33418	0.91719
577	0.37026	0.86189	0.49702	0.97231
623	0.71933	0.91834	0.40682	0.97023
679	0.85487	0.97973	0.71195	0.87658
786	0.99569	0.97342	0.72265	0.59382
864	0.751818	0.94047	0.73929	0.99454
891	0.754620	0.96921	0.70214	0.98157
932	0.99638	0.98025	0.77690	0.70384
939	0.20961	0.83774	0.30190	0.83019
977	0.35541	0.86019	0.51650	0.97811
981	0.98174	0.99282	0.78665	0.72818
990	0.15581	0.77297	0.31716	0.94282
1030	0.99238	0.97988	0.59015	0.72621
1052	0.81557	0.95570	0.38996	0.83303
1086	0.70533	0.84514	0.67399	0.94493
1407	0.57368	0.87474	0.57333	0.98578
1439	0.950293	0.99538	0.86204	0.96250
1516	0.992043	0.96063	0.62679	0.65394
1525	0.997766	0.99498	0.88214	0.88461
1670	0.997158	0.97329	0.84293	0.67598
1752	0.996829	0.97121	0.79426	0.63956
2040	0.997614	0.99481	0.88193	0.88985
2662	0.389770	0.91133	0.44232	0.97251
3030	0.993197	0.84509	0.54793	0.36585
1680	0.129303	0.60073	0.14808	0.80945
305	0.8397347	0.88964	0.80162	0.94209
440	0.5953833	0.89110	0.79853	0.98245
506	0.3423043	0.88163	0.29527	0.83450
512	0.9071478	0.99023	0.76172	0.85221
523	0.3972747	0.84949	0.43340	0.96843

Table 4.5. This table shows, for each training message, the Pearson correlation coefficients computed after the *model validation* experiment (see Item (2)), related to: the number of retweets at the time slot k and until k; the number of retweeting users at k and until k.

4.9.3.3 Prediction Testing Results

In this section, we show all the results for the prediction testing experiment.

We show in Table 4.6 the Pearson correlation coefficients, computed for each test message.

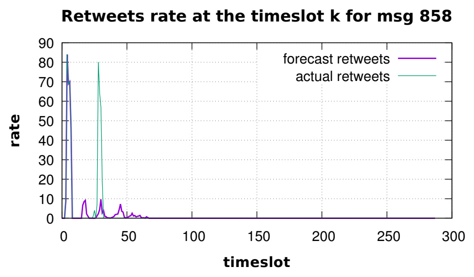


Figure 4.57. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 858.

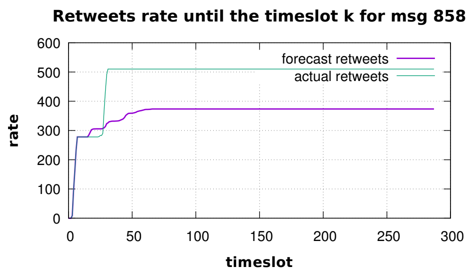


Figure 4.58. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 858.

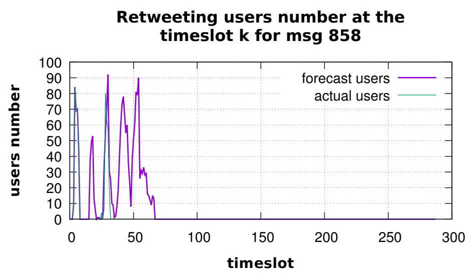


Figure 4.59. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 858.

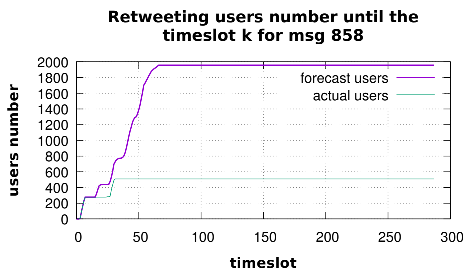


Figure 4.60. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 858.

msg id	#retweets at k	#retweets to k	#users at k	#users to k
858	0.78393	0.92790	0.54605	0.84044
1241	0.89827	0.95840	0.39714	0.68305
1700	0.93007	0.99067	0.90126	0.96826
1704	0.53501	0.89490	0.52805	0.93353
1839	0.98823	0.97037	0.50847	0.70667

Table 4.6. This table shows, for each training message, the Pearson correlation coefficients computed after the *prediction testing* experiment (see Item (3)), related to: the number of retweets at the time slot k and until k; the number of retweeting users at k and until k.

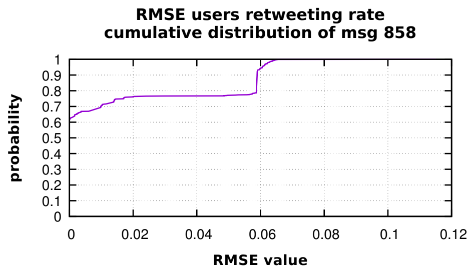


Figure 4.61. RMSE cumulative distribution, for message 858. With probability 1 the $RMSE < 0.065$

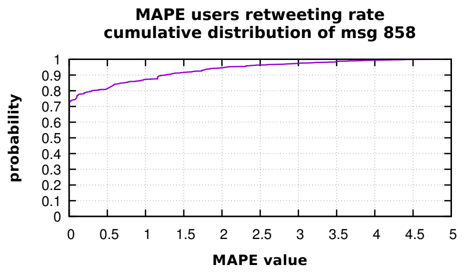


Figure 4.62. MAPE cumulative distribution, for message 858. With probability 0.8 the $MAPE$ is of 25%

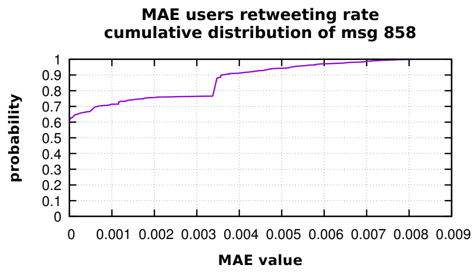


Figure 4.63. MAE cumulative distribution, for message 858. With probability 1 the $MAE < 0.008$

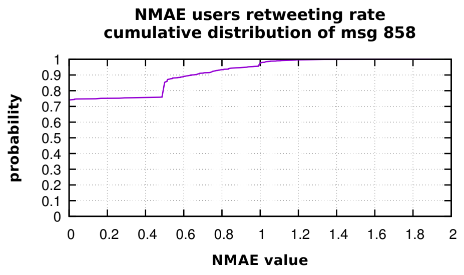


Figure 4.64. NMAE cumulative distribution, for message 858. With probability 0.8 the $NMAE$ is of 50%

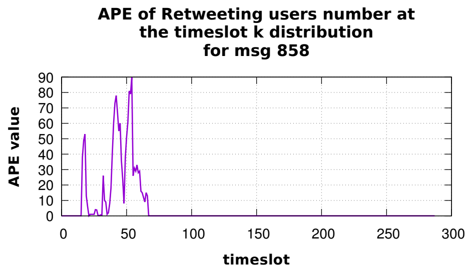


Figure 4.65. Distribution of APE for the retweeting users number at the time slot k, for message 858

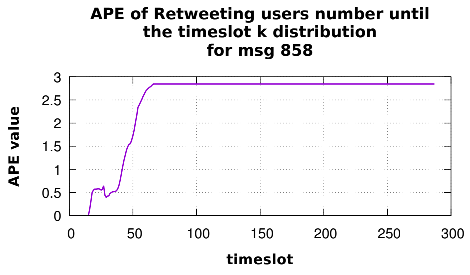


Figure 4.66. Distribution of APE for the retweeting users number until time slot k, for message 858

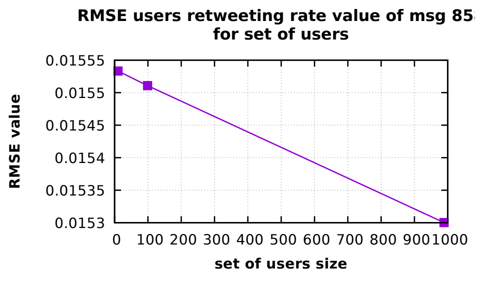


Figure 4.67. RMSE values distribution with respect to the set of users size, for message 858. When the set includes all network nodes, $RMSE = 0.0153$

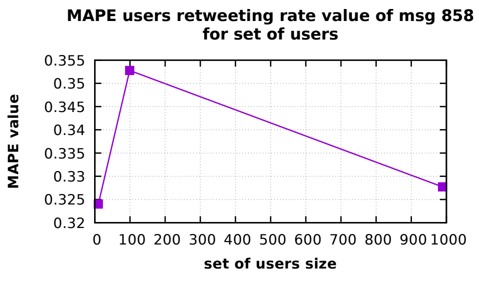


Figure 4.68. MAPE values distribution with respect to the set of users size, for message 858. When the set includes all network nodes, $MAPE = 0.327$

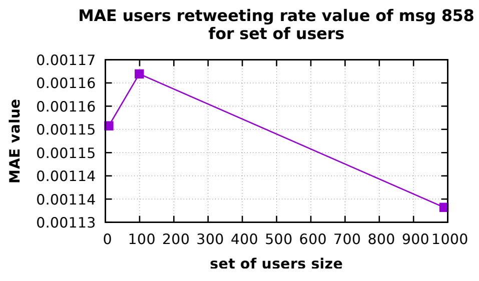


Figure 4.69. MAE values distribution with respect to the set of users size, for message 858. When the set includes all network nodes, $MAE = 0.01138$

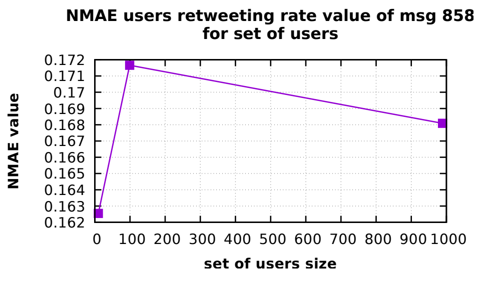


Figure 4.70. NMAE values distribution with respect to the set of users size, for message 858. When the set includes all network nodes, $NMAE = 0.168$

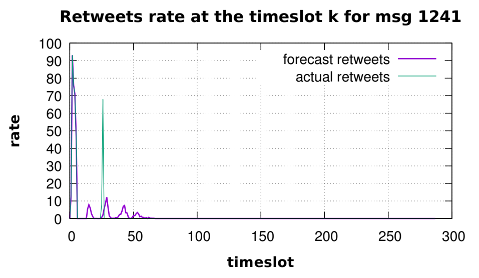


Figure 4.71. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1241.

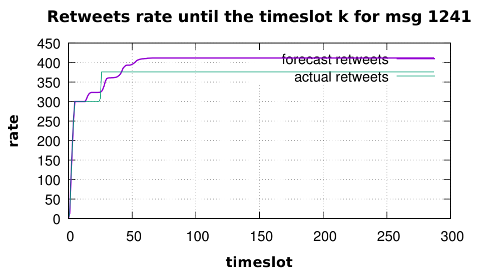


Figure 4.72. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1241.

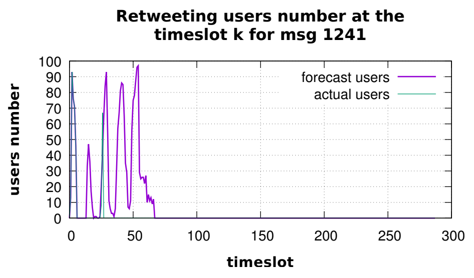


Figure 4.73. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1241.

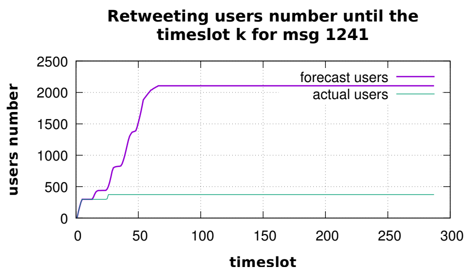


Figure 4.74. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1241.

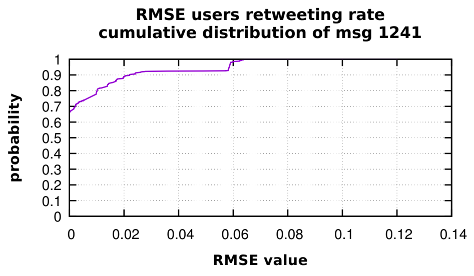


Figure 4.75. RMSE cumulative distribution, for message 1241. With probability 1 the $RMSE < 0.062$

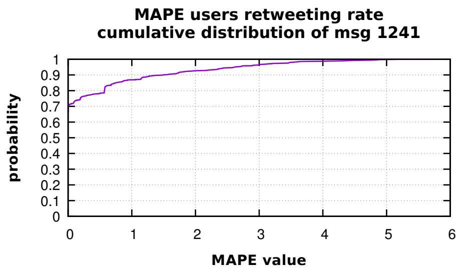


Figure 4.76. MAPE cumulative distribution, for message 1241. With probability 0.8 the $MAPE$ is of 50%

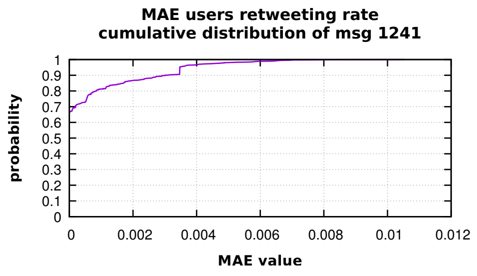


Figure 4.77. MAE cumulative distribution, for message 1241. With probability 1 the $MAE < 0.008$

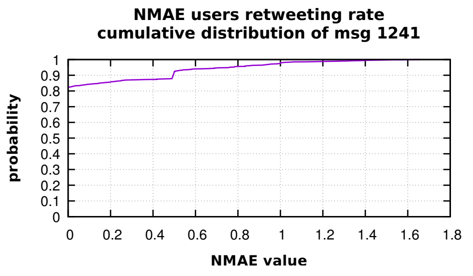


Figure 4.78. NMAE cumulative distribution, for message 1241. With probability 0.8 the $NMAE$ is of 50%

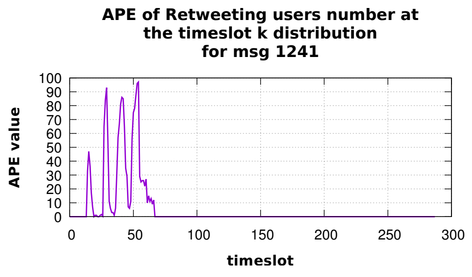


Figure 4.79. Distribution of APE for the retweeting users number at the time slot k, for message 1241.

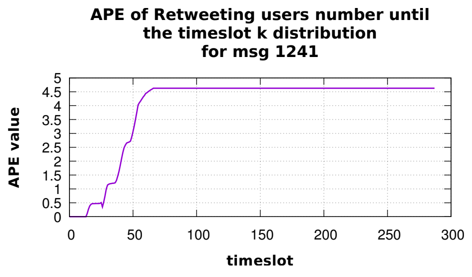


Figure 4.80. Distribution of APE for the retweeting users number until time slot k, for message 1241.

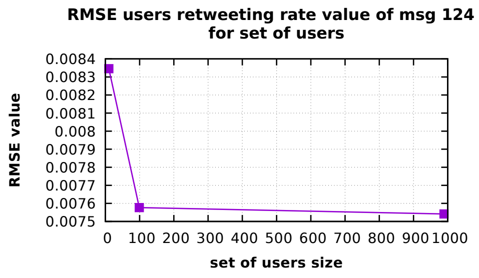


Figure 4.81. RMSE values distribution with respect to the set of users size, for message 1241. When the set includes all network nodes, $RMSE = 0.00755$

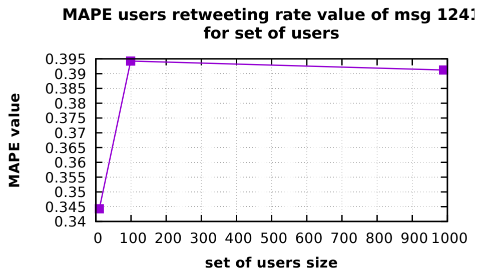


Figure 4.82. MAPE values distribution with respect to the set of users size, for message 1241. When the set includes all network nodes, $MAPE = 0.39$

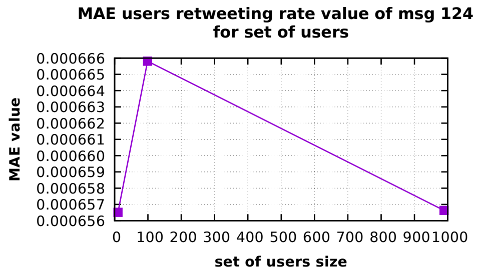


Figure 4.83. MAE values distribution with respect to the set of users size, for message 1241. When the set includes all network nodes, $MAE = 0.000657$

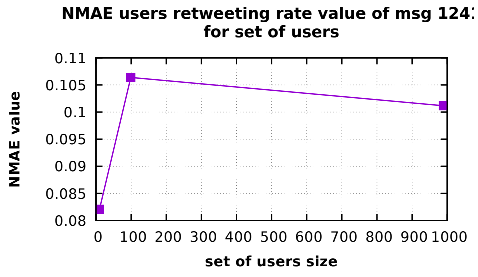


Figure 4.84. NMAE values distribution with respect to the set of users size, for message 1241. When the set includes all network nodes, $NMAE = 0.1$

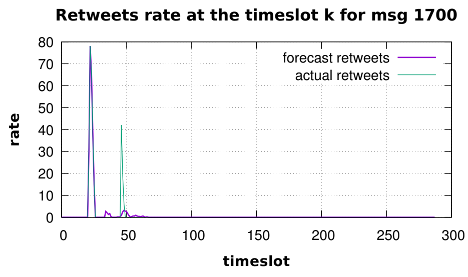


Figure 4.85. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1700.

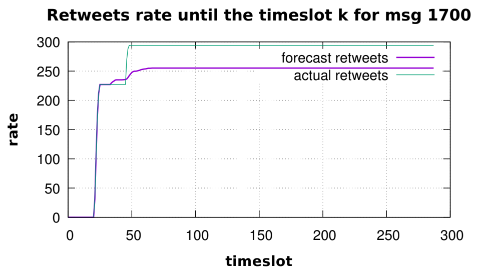


Figure 4.86. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1700.

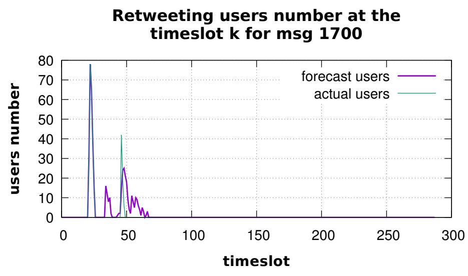


Figure 4.87. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1700.

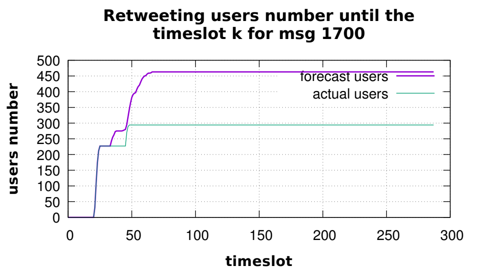


Figure 4.88. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1700.

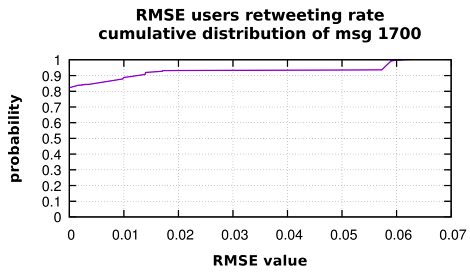


Figure 4.89. RMSE cumulative distribution, for message 1700. With probability 1 the $RMSE < 0.06$

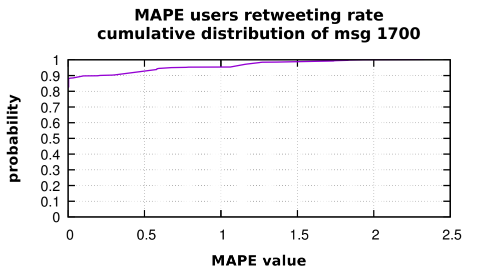


Figure 4.90. MAPE cumulative distribution, for message 1700. With probability 0.8 the $MAPE$ is 0

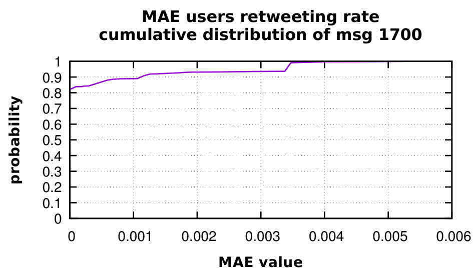


Figure 4.91. MAE cumulative distribution, for message 1700. With probability 1 the $MAE < 0.005$

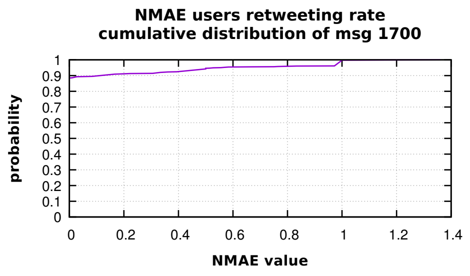


Figure 4.92. NMAE cumulative distribution, for message 1700. With probability 0.8 the $NMAE$ is of 0. With probability 0.91 is of 20%

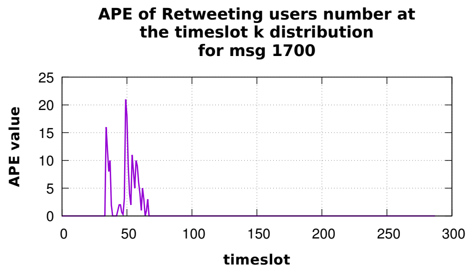


Figure 4.93. Distribution of APE on the retweeting users number at the time slot k, for message 1700.

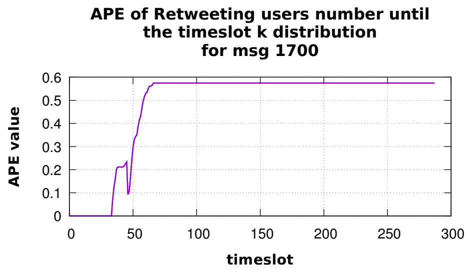


Figure 4.94. Distribution of APE on the retweeting users number until time slot k, for message 1700.

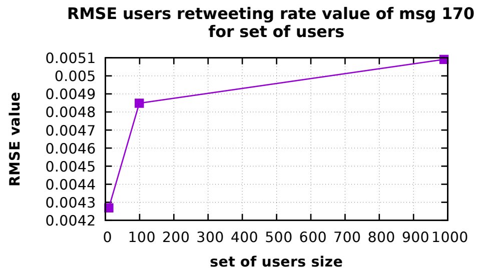


Figure 4.95. RMSE values distribution with respect to the set of users size, for message 1700. When the set includes all network nodes, $RMSE = 0.0051$

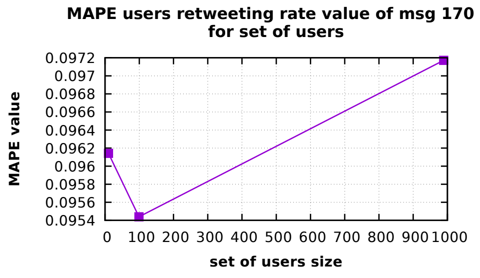


Figure 4.96. MAPE values distribution with respect to the set of users size, for message 1700. When the set includes all network nodes, $MAPE = 0.0972$

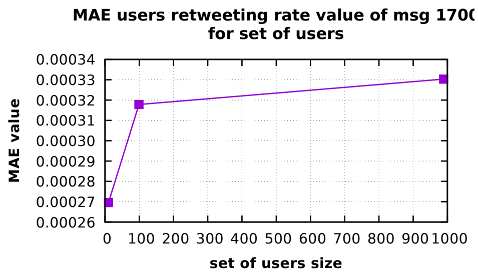


Figure 4.97. MAE values distribution with respect to the set of users size, for message 1700. When the set includes all network nodes, $MAE = 0.00033$

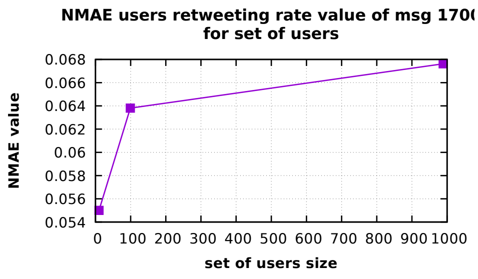


Figure 4.98. NMAE values distribution with respect to the set of users size, for message 1700. When the set includes all network nodes, $NMAE = 0.068$

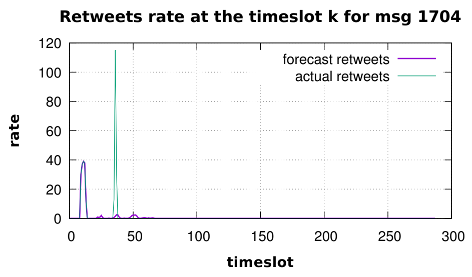


Figure 4.99. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1704.

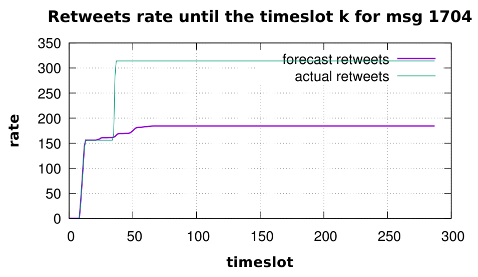


Figure 4.100. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1704.

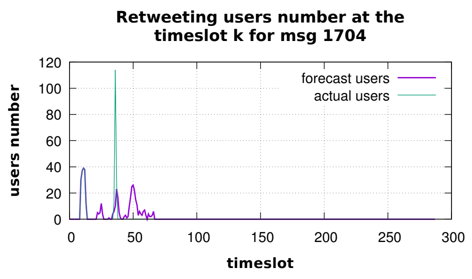


Figure 4.101. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1704.

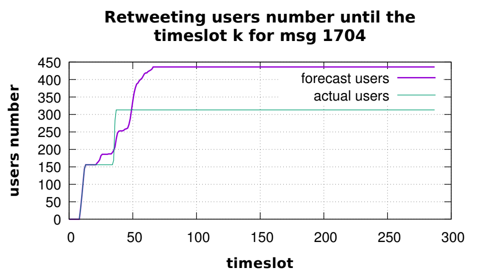


Figure 4.102. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1704.

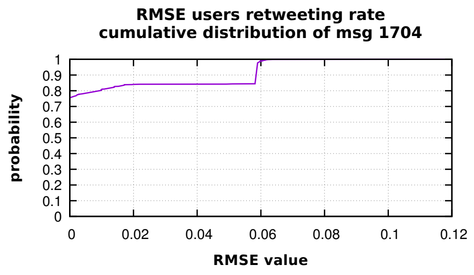


Figure 4.103. RMSE cumulative distribution, for message 1704. With probability 1 the $RMSE < 0.06$

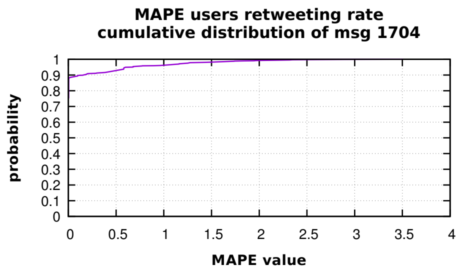


Figure 4.104. MAPE cumulative distribution, for message 1704. With probability 0.8 the $MAPE$ is 0

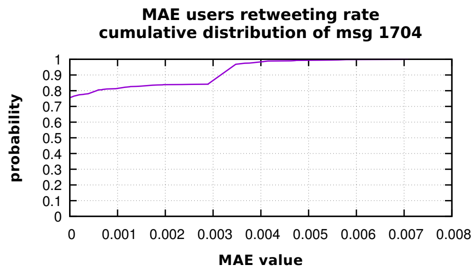


Figure 4.105. MAE cumulative distribution, for message 1704. With probability 1 the $MAE < 0.006$

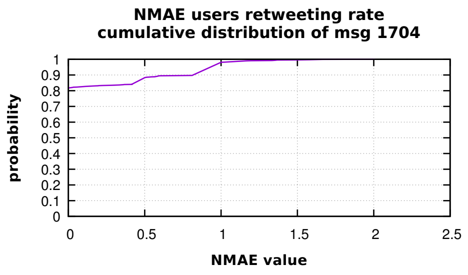


Figure 4.106. NMAE cumulative distribution, for message 1704. With probability 0.8 the $NMAE$ is 0.4

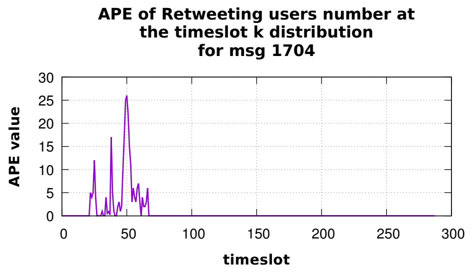


Figure 4.107. Distribution of APE on the retweeting users number at the time slot k, for message 1704.

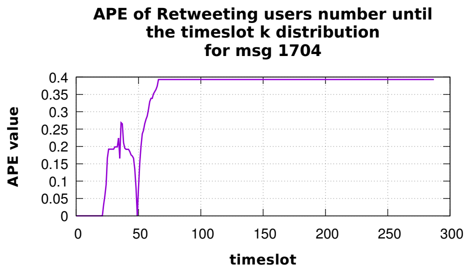


Figure 4.108. Distribution of APE on the retweeting users number until time slot k, for message 1704.

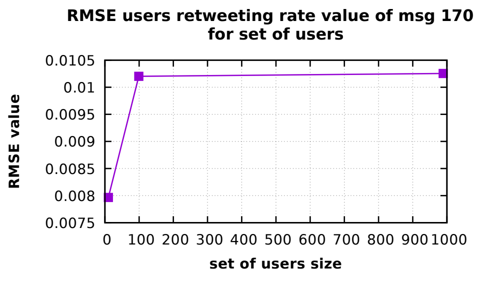


Figure 4.109. RMSE values distribution with respect to the set of users size, for message 1704. When the set involves all network nodes, $RMSE = 0.0103$

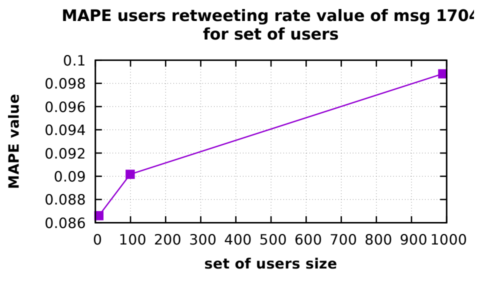


Figure 4.110. MAPE values distribution with respect to the set of users size, for message 1704. When the set involves all network nodes, $MAPE = 0.099$

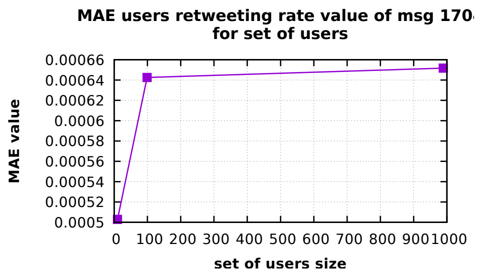


Figure 4.111. MAE values distribution with respect to the set of users size, for message 1704. When the set involves all network nodes, $MAE = 0.00065$

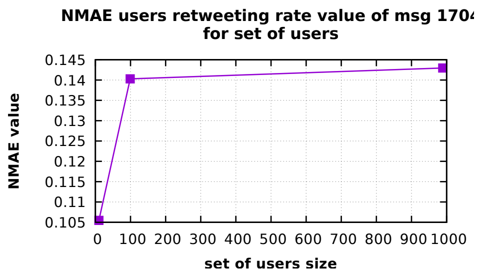


Figure 4.112. NMAE values distribution with respect to the set of users size, for message 1704. When the set involves all network nodes, $NMAE = 0.0143$

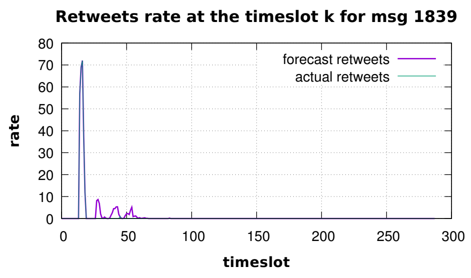


Figure 4.113. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1839.

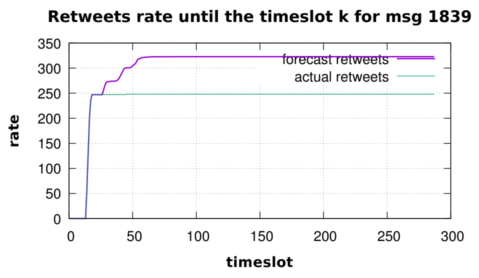


Figure 4.114. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1839.

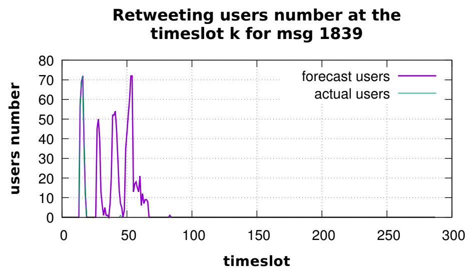


Figure 4.115. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1839.

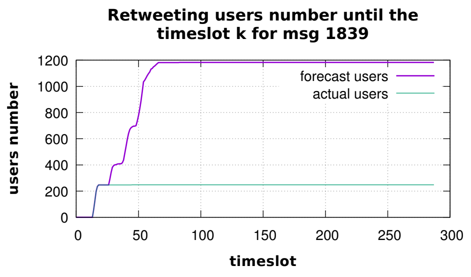


Figure 4.116. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1839.

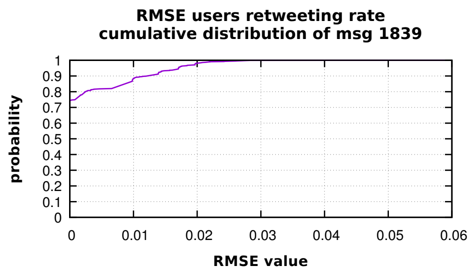


Figure 4.117. RMSE cumulative distribution, for message 1839. With probability 1 the $RMSE < 0.03$

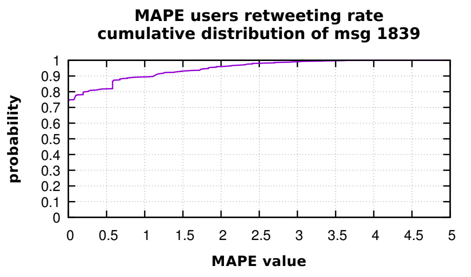


Figure 4.118. MAPE cumulative distribution, for message 1839. With probability 0.8 the $MAPE$ is of 20%

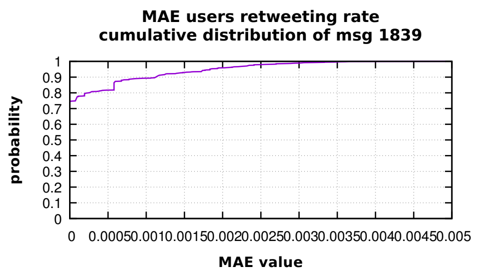


Figure 4.119. MAE cumulative distribution, for message 1839. With probability 1 the $MAE < 0.0035$

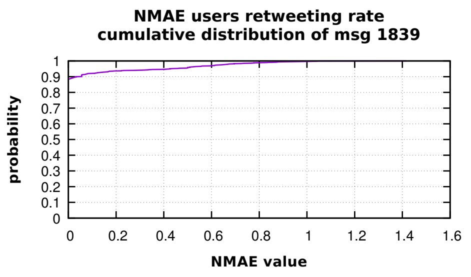


Figure 4.120. NMAE cumulative distribution, for message 1839. With probability 0.8 the $NMAE$ is 0, with probability 0.9 is of 5%

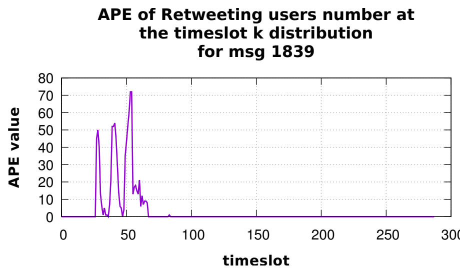


Figure 4.121. Distribution of APE on the retweeting users number at the time slot k, for message 1839.

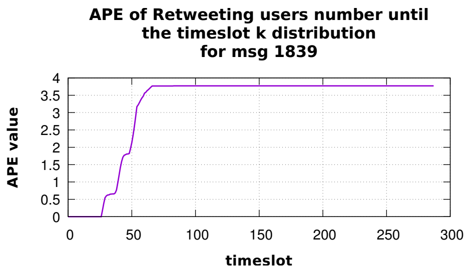


Figure 4.122. Distribution of APE on the retweeting users number until time slot k, for message 1839.

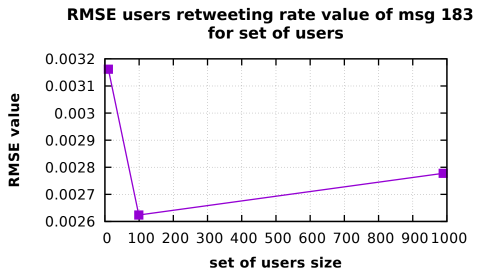


Figure 4.123. RMSE values distribution with respect to the set of users size, for message 1839. When the set involves all network nodes, $RMSE = 0.0028$

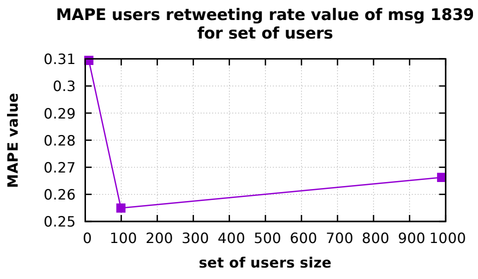


Figure 4.124. MAPE values distribution with respect to the set of users size, for message 1839. When the set involves all network nodes, $MAPE = 0.267$

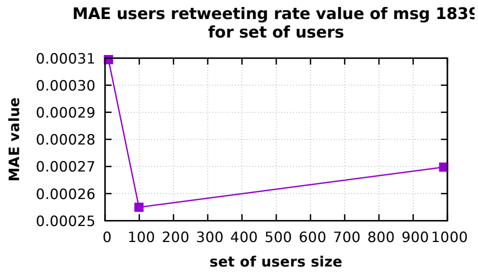


Figure 4.125. MAE values distribution with respect to the set of users size, for message 1839. When the set involves all network nodes, $MAE = 0.00027$

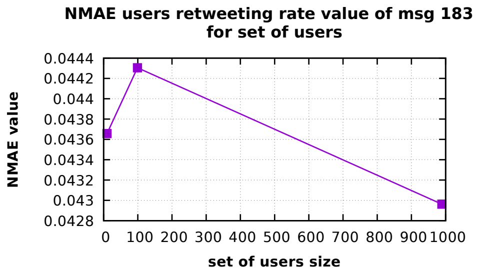


Figure 4.126. NMAE values distribution with respect to the set of users size, for message 1839. When the set involves all network nodes, $NMAE = 0.043$

4.9.4 Computational Environment

We implemented our simulator to compute $\bar{X}_{i,m}(k)$ in C programming language. All experiments have been carried out on a PC Intel® $Core^{TM}$ $i5 - 6200U$ CPU $2.30GHz \times 4$. The user parameters $\bar{A}^{m}_{i,j,d}([k])$ computation has been implemented in R language and has been executed on a Cluster made up of one login node and nine nodes used for computation. The features of these 9 nodes are shown in Table 4.7.

Mach. id	**Type and num. of CPUs**	**CPUs freq.**	**total #cores**	**overall RAM**	**mach. cat.**
0	1 × Intel(R) Celeron(R)	2.27 GHz	2	4 GB	Z
1	2 × Intel(R) Xeon(R)	2.83 GHz	8	8 GB	A
2	2 × Intel(R) Xeon(R)	2.83 GHz	8	8 GB	A
3	2 × Intel(R) Xeon(R)	2.66 GHz	8	32 GB	B
4	2 × Intel(R) Xeon(R)	2.66 GHz	8	32 GB	B
5	2 × Intel(R) Xeon(R)	2.27 GHz	16	24 GB	C
6	2 × Intel(R) Xeon(R)	2.27 GHz	16	24 GB	C
7	2 × Intel(R) Xeon(R)	2.27 GHz	16	24 GB	C
8	2 × Intel(R) Xeon(R)	2.27 GHz	16	24 GB	C

Table 4.7. Configuration of our cluster machines.

4.10 Conclusions

We presented a mechanistic model to predict the expected value of the retweeting rate of a message in a Twitter network, for each user and for each time slot of a considered period. Our model is identifiable, as it is built on top of stochastic variables, representing the users' behavior in the network and whose parameters (mean and variance) have been computed from a dataset. We provided also a validation methodology for our model.

Our experimental results show that when a new message is posted and re-shared into the network, our model predicts the qualitative behavior at a time slot k and also, fairly well, the quantitative behavior until the time slot k. This is confirmed by the high correlation between the predicted and the observed data. Furthermore, our experimental evaluations show that the RMSE and MAPE at node level and at a set of nodes of nodes level are small.

These results have high potential in the analysis of the behaviors of users or of set of users in a social network.

Chapter 5

Conclusions

In this thesis we addressed the problem of estimating the parameters of a probability distribution in two different settings. Namely: the analysis and the verification of model-based systems through Statistical Model Checking (SMC) techniques; the identification of parameters of predictive models.

In Chapter 2 we presented an overview of a set of the Monte Carlo-based SMC tools for Cyber-Physical Systems, currently available for research purposes. Furthermore, we provided selection criteria that based on the kind of system and the property to be verified, allow to select the most suitable tool for the problem at hand.

In Chapter 3 we presented an efficient Monte Carlo-based algorithm to estimate the expected value of a multivariate random variable, when marginal density functions are unknown. We proved that our algorithm is correct and we gave an Upper Bound and a Lower Bound to the complexity (in terms of the generated number of samples). Finally, we presented experimental results confirming our evaluations.

In Chapter 4 we presented a mechanistic model for the expected value of the retweeting rate of a message in a Twitter network, for each user or set of users. Our model is a discrete-time system whose parameters are stochastic variables, representing the users activity towards their followees. We estimated these stochastic variables parameters (mean and variance) from an available dataset. We experimentally showed that for a new message spread into the network, our model reliably predicts the qualitative time behavior and the quantitative time behavior. This is confirmed by the high correlation between the predicted and the observed data. Our experimental evaluations show that both the RMSE and MAPE at the node level and at the set of nodes level are small. These results enable the analysis of the behavior of a single user or set of users inside a network.

Appendix A

Proof of Theorem 4.1

In this section we give the proof of Theorem 4.1.

Theorem. *$X_{i,m}(t)$ is statistically independent from $A^m_{i,j,d}(k)$, for each time slot k, for each delay $d = 0, \dots, D$, for each followee $j \in F_i$ and for each time slot t, s.t. $t \leq k$. Formally:*

$$\mathbb{P}(X_{i,m}(t) = x | A^m_{i,j,d}(k) = a) = \mathbb{P}(X_{i,m}(t) = x) \tag{A.1}$$

Proof. Unless otherwise specified, let us denote $\mathbb{P}(X_{i,m}(k) = x)$ with $\mathbb{P}(X_{i,m}(k))$ and $\mathbb{P}(A^m_{i,j,d}(k) = a)$ with $\mathbb{P}(A^m_{i,j,d}(k))$.

By induction, when $t = 0, .., D$, $\mathbb{P}(X_{i,m}(t)|A^m_{i,j,d}(k)) = \mathbb{P}(X_{i,m}(t))$. In fact, in this case $X_{i,m}(t)$ is a known value that can be computed from the observations and that does not depend from any other variable.

Let us take $t > D$. By inductive hypothesis, it holds that:

$$\mathbb{P}(X_{i,m}(t)|A^m_{i,j,d}(k)) = \mathbb{P}(X_{i,m}(t)) \tag{A.2}$$

We want to prove that $\mathbb{P}(X_{i,m}(t+1)|A^m_{i,j,d}(k)) = \mathbb{P}(X_{i,m}(t+1))$. By eq. 4.5, it holds:

$$\mathbb{P}(X_{i,m}(t+1)|A^m_{i,j,d}(k)) = \mathbb{P}(\sum_{j \in F_i, d=0}^{D} A^m_{i,j,d}(t) X_{j,m}(t-d) | A^m_{i,j,d}(k)) \tag{A.3}$$

Note that, in eq. A.3, $X_{j,m}(t-d)$ and $A^m_{i,j,d}(k)$ are statistically independent by inductive hypothesis and that $A^m_{i,j,d}(t)$ and $A^m_{i,j,d}(k)$ are i.i.d., by the hypothesis made in Section 4.5.1.As a consequence, a linear combination of $X_{j,m}(t-d)$ and $A^m_{i,j,d}(t)$ is independent from $A^m_{i,j,d}(k)$ as well. Thus, it follows:

$$\begin{aligned}\mathbb{P}(\sum_{j \in F_i, d=0}^{D} A^m_{i,j,d}(t) X_{j,m}(t-d) | A^m_{i,j,d}(k)) &= \mathbb{P}(\sum_{j \in F_i, d=0}^{D} A^m_{i,j,d}(t) X_{j,m}(t-d)) && \text{(A.4)} \\ &= \mathbb{P}(X_{i,m}(t+1)) && \text{(A.5)}\end{aligned}$$

□

Appendix B

Proof of Theorem 4.2

In this section we give the proof of Theorem 4.2.

Theorem. *Let $\bar{A}_{i,j,d}([k])$ be the expected value of the random variable $A^m_{i,j,d}(k)$. Then:*

$$\bar{X}_{i,m}(k+1) = \sum_{j \in F_i, d=0}^{D} \bar{A}_{i,j,d}([k]) \bar{X}_{j,m}(k-d) \tag{B.1}$$

Proof. Let us consider $X_{i,m}(k+1)$ according to the eq. 4.5. It holds:

$$\mathbb{E}[X_{i,m}(k+1)] = \mathbb{E}[\sum_{j \in F_i, d=0}^{D} A^m_{i,j,d}(k) X_{j,m}(k-d)] \tag{B.2}$$

$$= \sum_{j \in F_i, d=0}^{D} \mathbb{E}[A^m_{i,j,d}(k) X_{j,m}(k-d)] \tag{B.3}$$

As for Theorem 4.1 it holds that $A^m_{i,j,d}(k)$ and $X_{i,m}(k)$ are s.i., and then also $A^m_{i,j,d}(k)$ and $X_{j,m}(k-d)$, it holds:

$$\mathbb{E}[X_{i,m}(k+1)] = \sum_{j \in F_i, d=0}^{D} \mathbb{E}[A^m_{i,j,d}(k)] \mathbb{E}[X_{j,m}(k-d)] \tag{B.4}$$

$$= \sum_{j \in F_i, d=0}^{D} \bar{A}_{i,j,d}([k]) \bar{X}_{j,m}(k-d) \tag{B.5}$$

This proves the result in eq. B.1. □

Appendix C

Proof of Proposition 4.1

In this section we give the proof of Proposition 4.1.

Proposition. $\mathbb{E}[\Delta_{i,m}(k)] = 0$*, for each* $i \in Nodes$ *and for each time slot* k*.*

Proof. Let us consider the time slot $k = 0$. It holds $\Delta_{i,m}(0) = 0$ because $X_{i,m}(0) = \bar{X}_{i,m}(0)$, in eq. 4.8. Thus, also $\mathbb{E}[\Delta_{i,m}(0)] = 0$.

Let us assume that $\mathbb{E}[\Delta_{i,m}(k)] = 0$, then by induction it holds:

$$\mathbb{E}[\Delta_{i,m}(k+1)] = \sum_{j \in F_i, d=0}^{D} \bar{A}_{i,j,d}([k])\mathbb{E}[\Delta_{j,m}(k-d)] = 0 \tag{C.1}$$

□

Appendix D

Complete set of Experimental Results

D.1 Sanity check

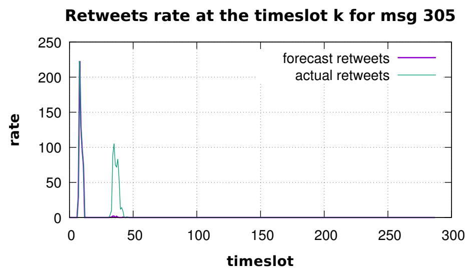


Figure D.1. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 305.

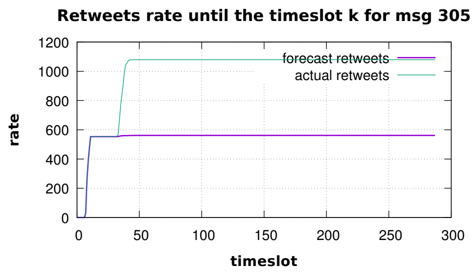


Figure D.2. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 305.

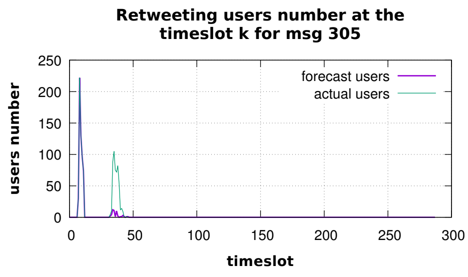


Figure D.3. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 305.

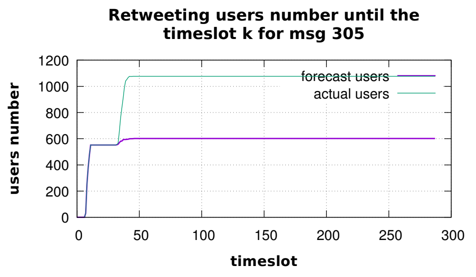


Figure D.4. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 305.

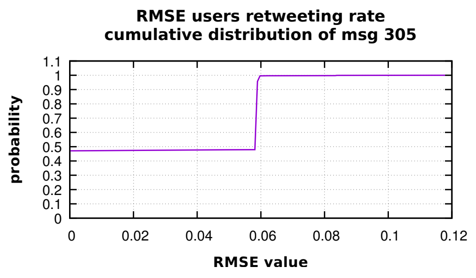


Figure D.5. Graphic showing the RMSE cumulative distribution, for message 305.

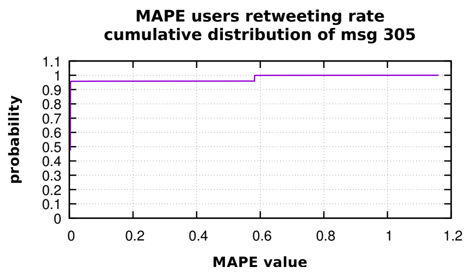


Figure D.6. Graphic showing the MAPE cumulative distribution, for message 305.

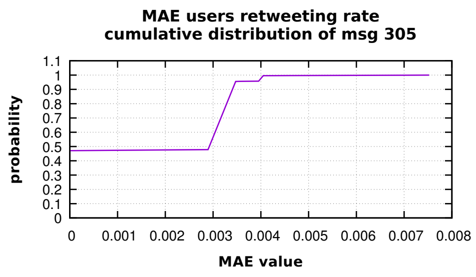


Figure D.7. Graphic showing the MAE cumulative distribution, for message 305.

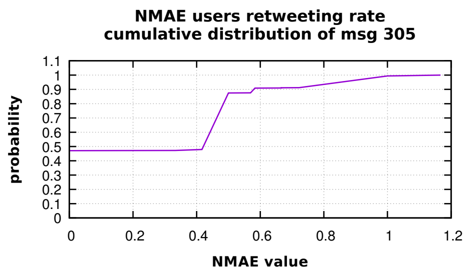


Figure D.8. Graphic showing the NMAE cumulative distribution, for message 305.

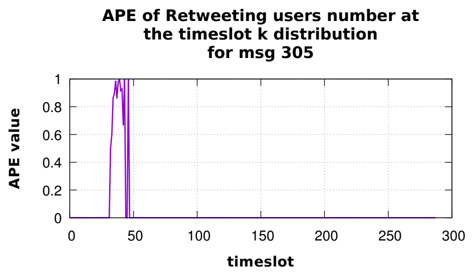


Figure D.9. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 305.

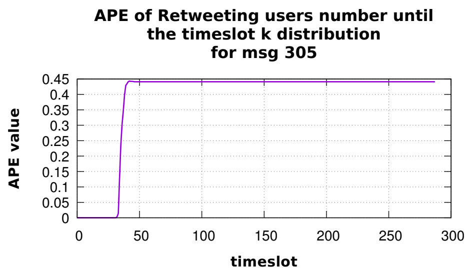


Figure D.10. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 305.

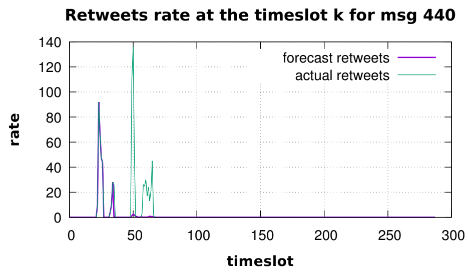


Figure D.11. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 440.

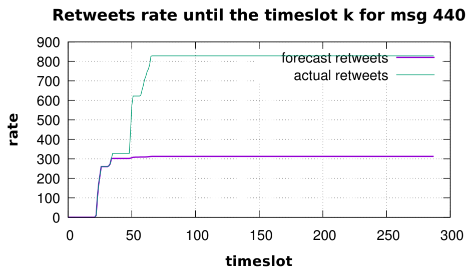


Figure D.12. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 440.

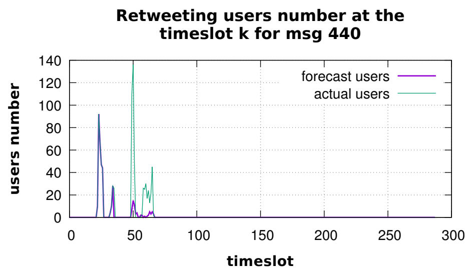


Figure D.13. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 440.

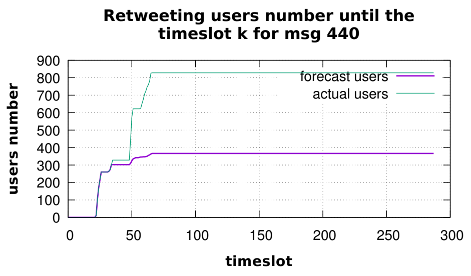


Figure D.14. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 440.

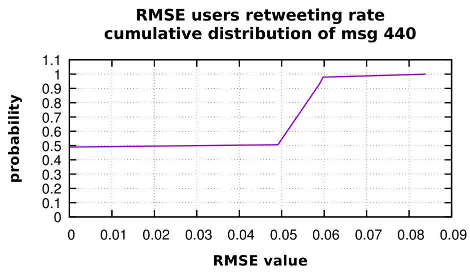


Figure D.15. Graphic showing the RMSE cumulative distribution, for message 440.

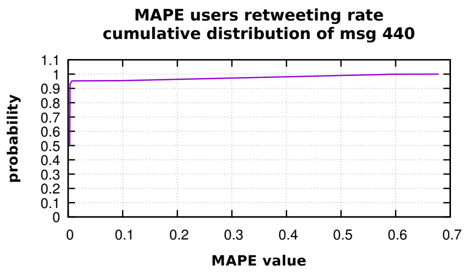


Figure D.16. Graphic showing the MAPE cumulative distribution, for message 440.

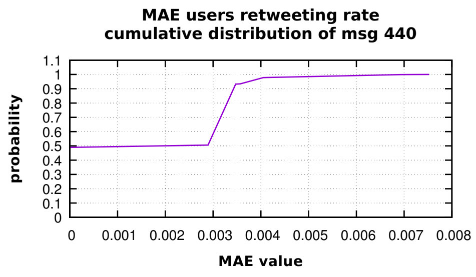


Figure D.17. Graphic showing the MAE cumulative distribution, for message 440.

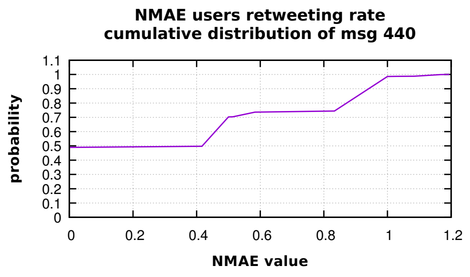


Figure D.18. Graphic showing the NMAE cumulative distribution, for message 440.

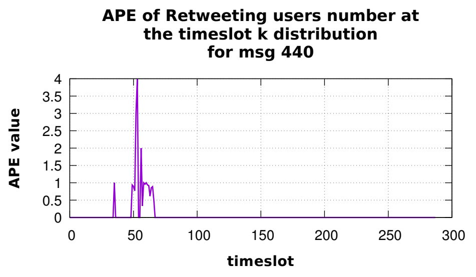


Figure D.19. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 440.

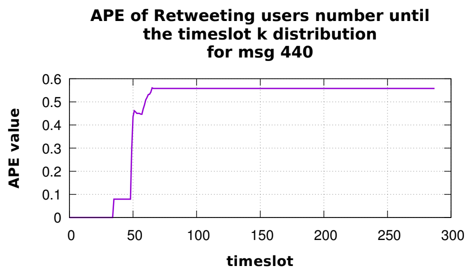


Figure D.20. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 440.

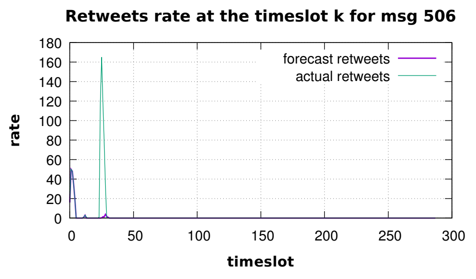


Figure D.21. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 506.

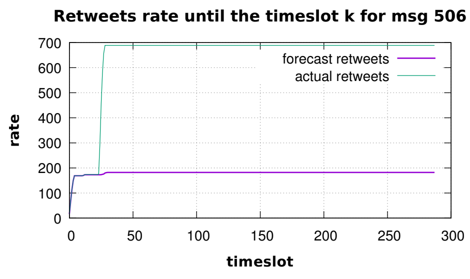


Figure D.22. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 506.

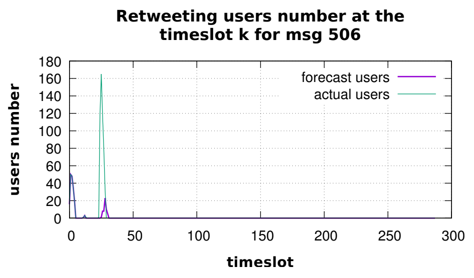


Figure D.23. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 506.

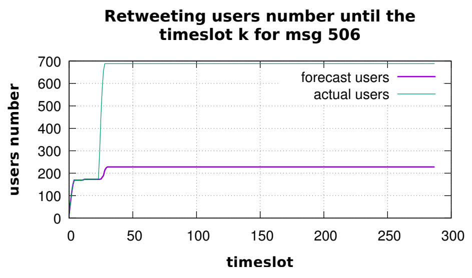


Figure D.24. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 506.

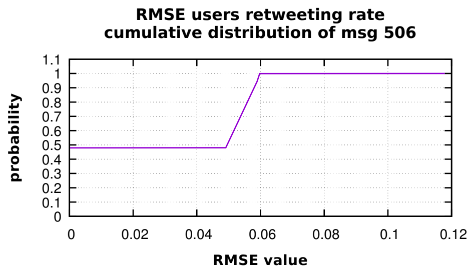


Figure D.25. Graphic showing the RMSE cumulative distribution, for message 506.

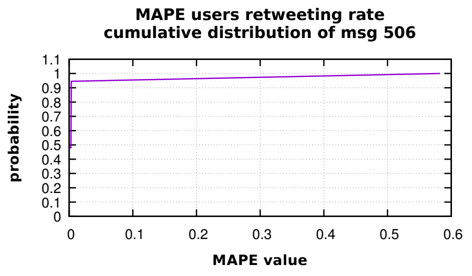


Figure D.26. Graphic showing the MAPE cumulative distribution, for message 506.

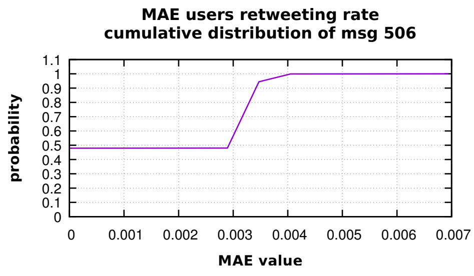


Figure D.27. Graphic showing the MAE cumulative distribution, for message 506.

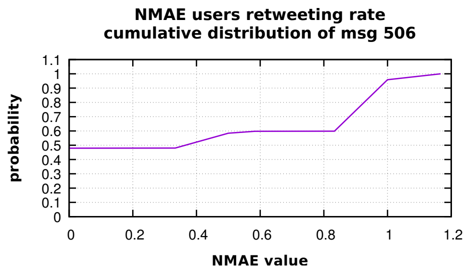


Figure D.28. Graphic showing the NMAE cumulative distribution, for message 506.

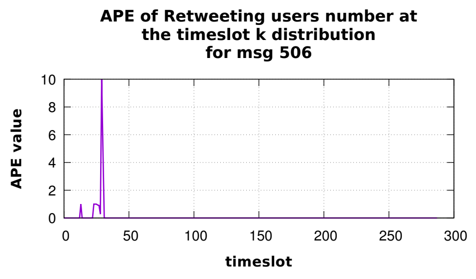


Figure D.29. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 506.

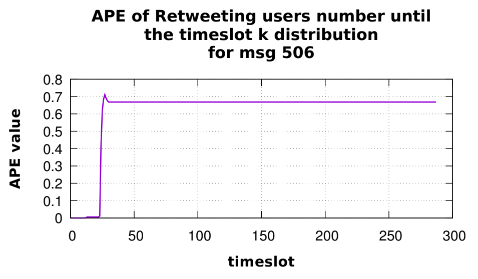


Figure D.30. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 506.

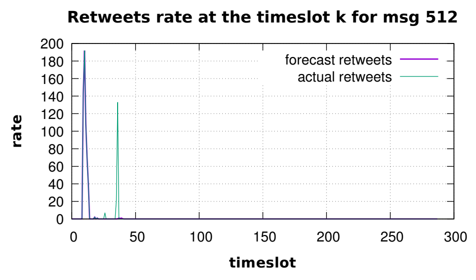


Figure D.31. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 512.

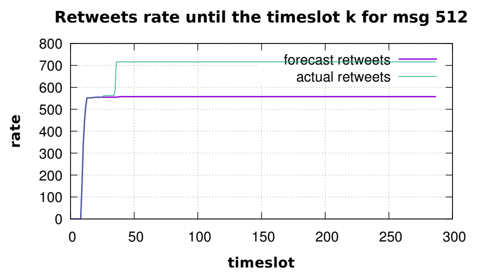


Figure D.32. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 512.

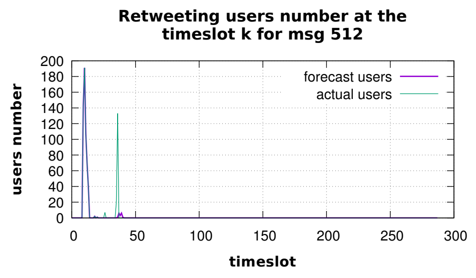


Figure D.33. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 512.

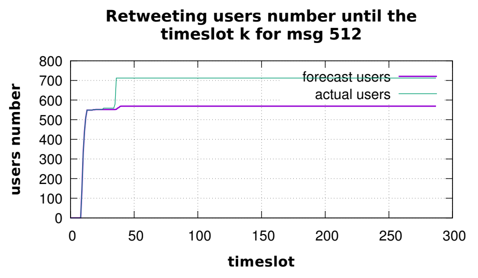


Figure D.34. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 512.

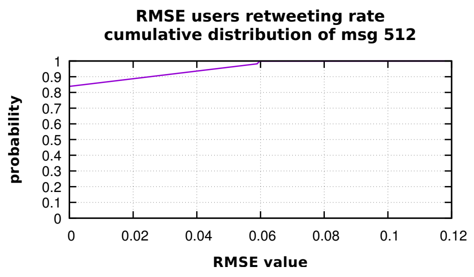


Figure D.35. Graphic showing the RMSE cumulative distribution, for message 512.

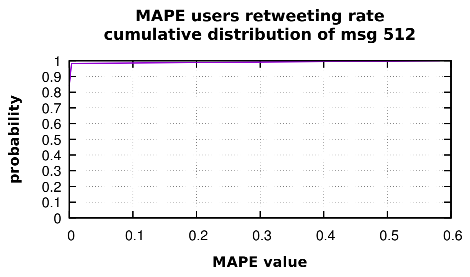


Figure D.36. Graphic showing the MAPE cumulative distribution, for message 512.

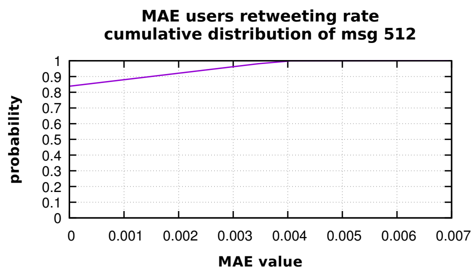


Figure D.37. Graphic showing the MAE cumulative distribution, for message 512.

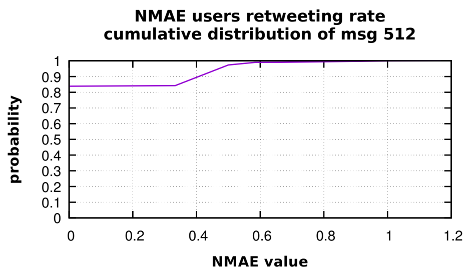


Figure D.38. Graphic showing the NMAE cumulative distribution, for message 512.

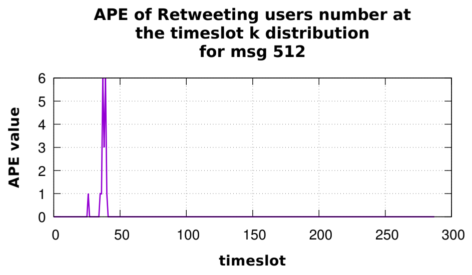


Figure D.39. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 512.

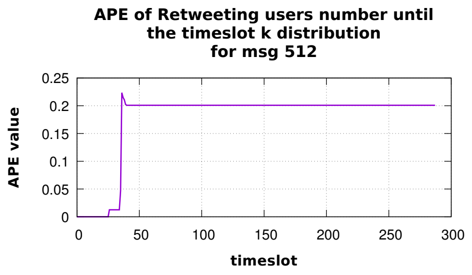


Figure D.40. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 512.

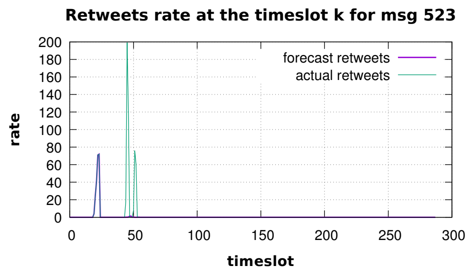


Figure D.41. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 523.

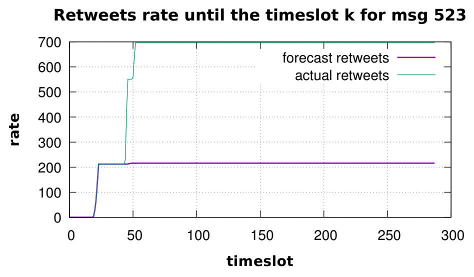


Figure D.42. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 523.

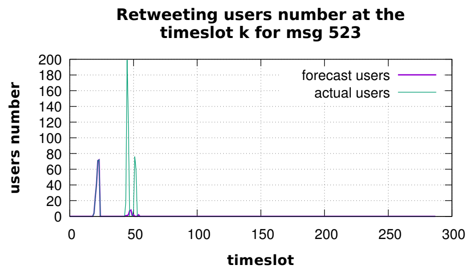


Figure D.43. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 523.

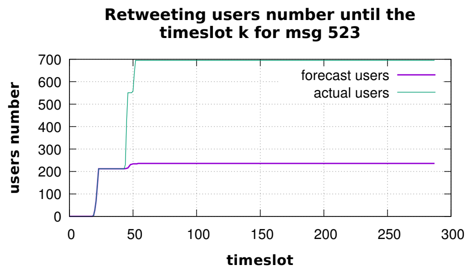


Figure D.44. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 523.

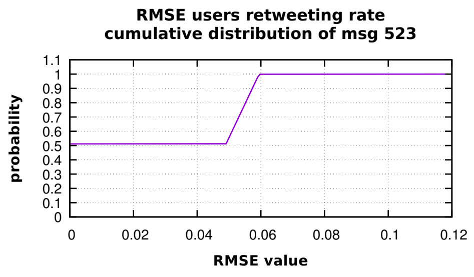


Figure D.45. Graphic showing the RMSE cumulative distribution, for message 523.

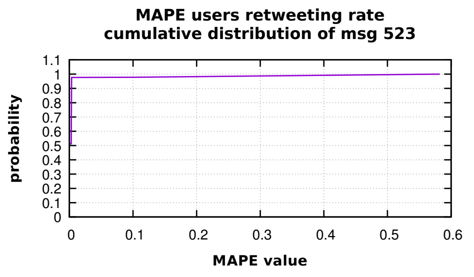


Figure D.46. Graphic showing the MAPE cumulative distribution, for message 523.

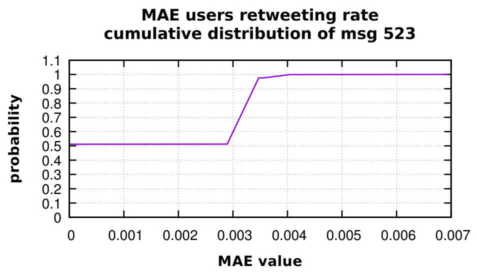


Figure D.47. Graphic showing the MAE cumulative distribution, for message 523.

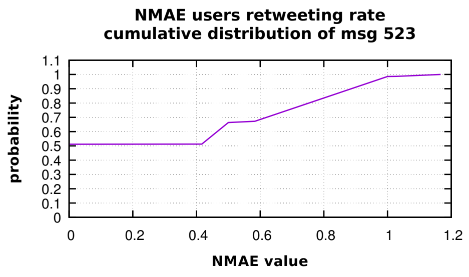


Figure D.48. Graphic showing the NMAE cumulative distribution, for message 523.

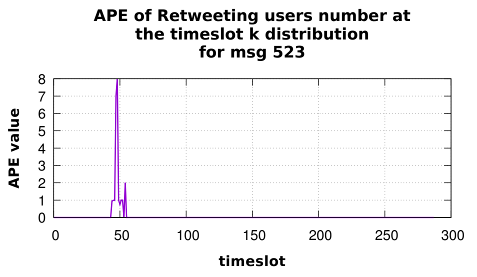


Figure D.49. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 523.

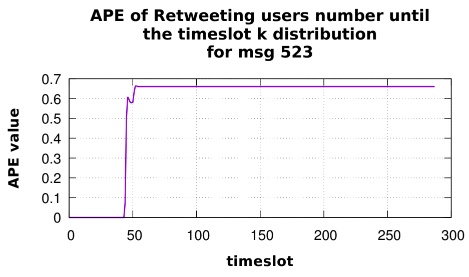


Figure D.50. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 523.

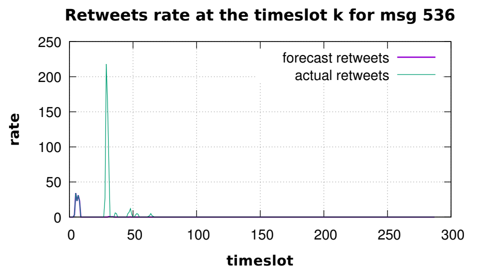


Figure D.51. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 536.

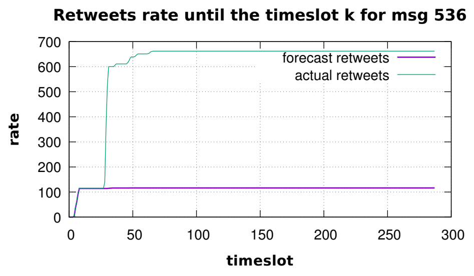


Figure D.52. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 536.

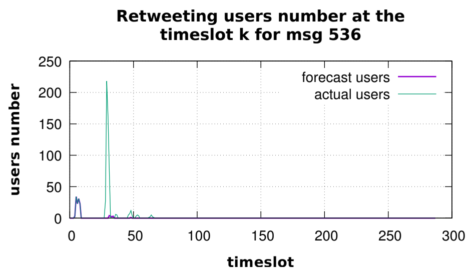


Figure D.53. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 536.

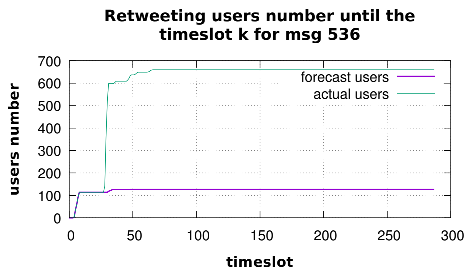


Figure D.54. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 536.

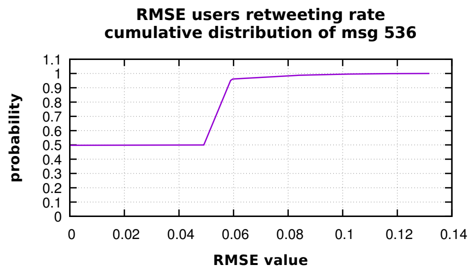


Figure D.55. Graphic showing the RMSE cumulative distribution, for message 536.

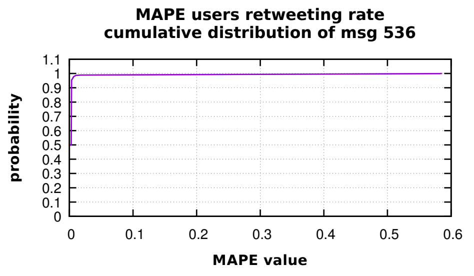


Figure D.56. Graphic showing the MAPE cumulative distribution, for message 536.

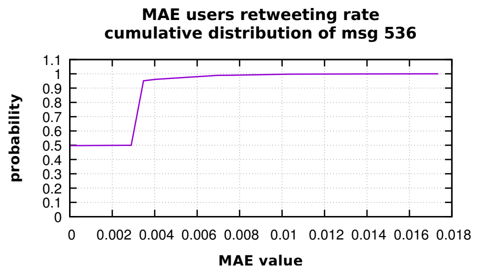


Figure D.57. Graphic showing the MAE cumulative distribution, for message 536.

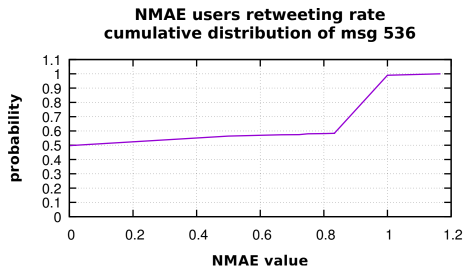


Figure D.58. Graphic showing the NMAE cumulative distribution, for message 536.

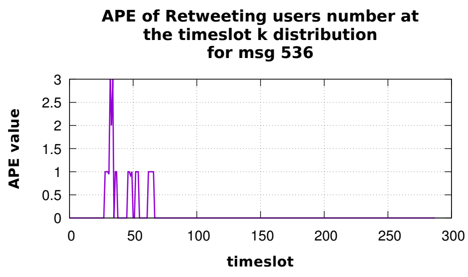


Figure D.59. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 536.

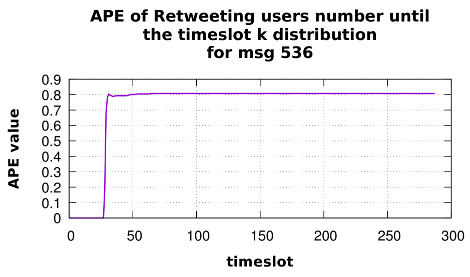


Figure D.60. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 536.

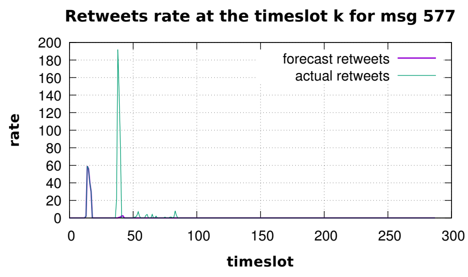


Figure D.61. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 577.

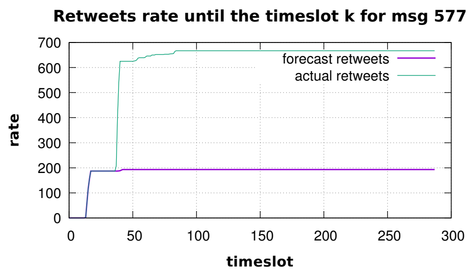


Figure D.62. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 577.

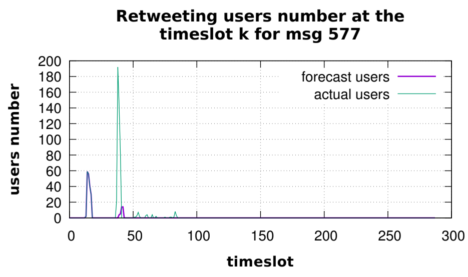


Figure D.63. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 577.

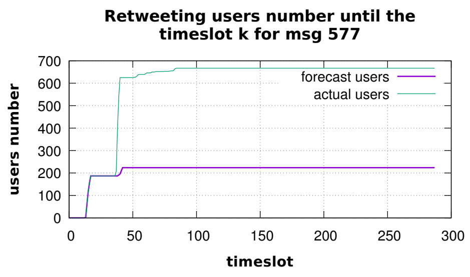


Figure D.64. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 577.

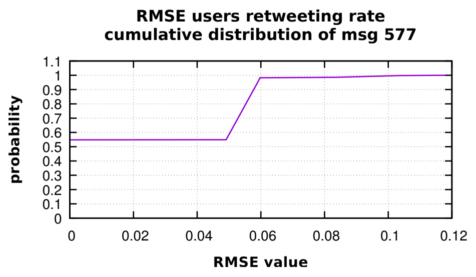


Figure D.65. Graphic showing the RMSE cumulative distribution, for message 577.

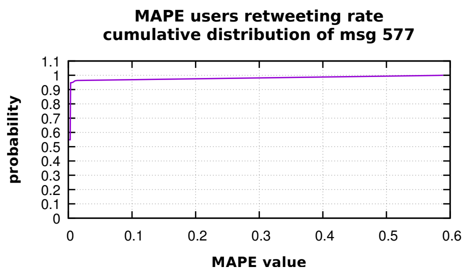


Figure D.66. Graphic showing the MAPE cumulative distribution, for message 577.

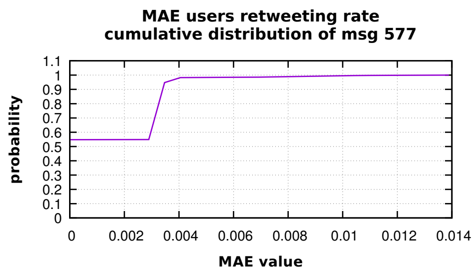


Figure D.67. Graphic showing the MAE cumulative distribution, for message 577.

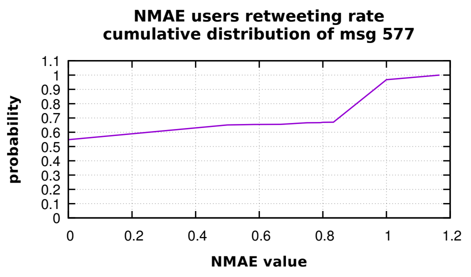


Figure D.68. Graphic showing the NMAE cumulative distribution, for message 577.

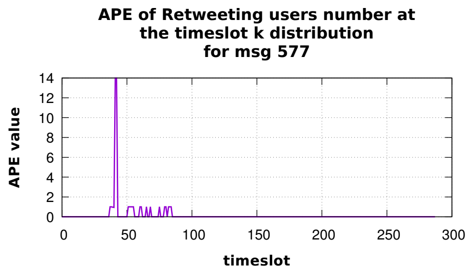


Figure D.69. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 577.

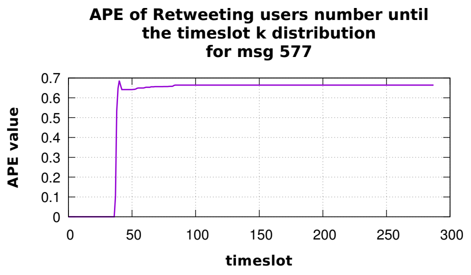


Figure D.70. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 577.

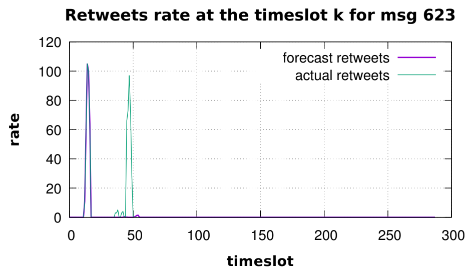


Figure D.71. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 623.

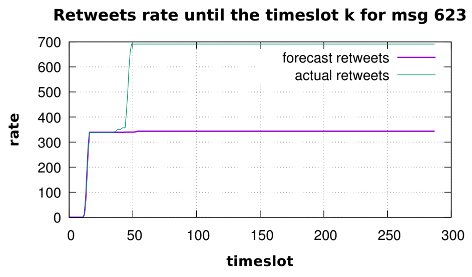


Figure D.72. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 623.

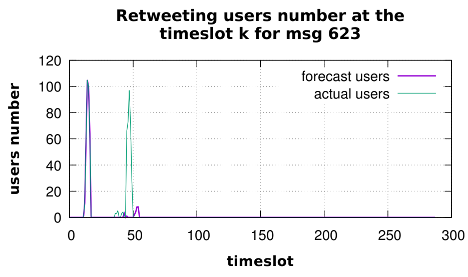


Figure D.73. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 623.

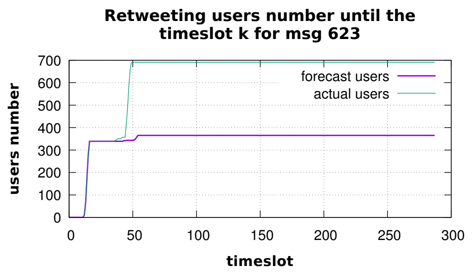


Figure D.74. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 623.

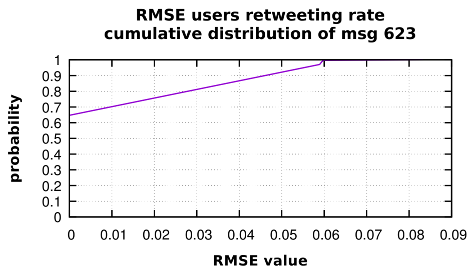


Figure D.75. Graphic showing the RMSE cumulative distribution, for message 623.

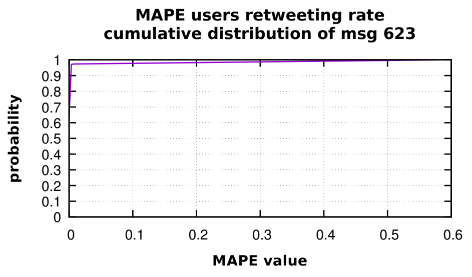


Figure D.76. Graphic showing the MAPE cumulative distribution, for message 623.

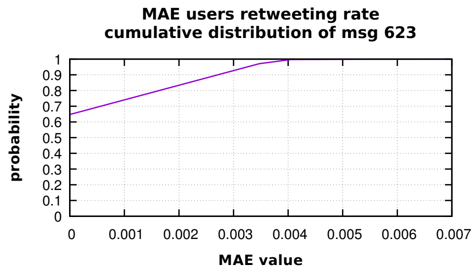


Figure D.77. Graphic showing the MAE cumulative distribution, for message 623.

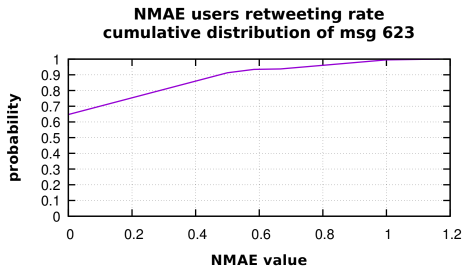


Figure D.78. Graphic showing the NMAE cumulative distribution, for message 623.

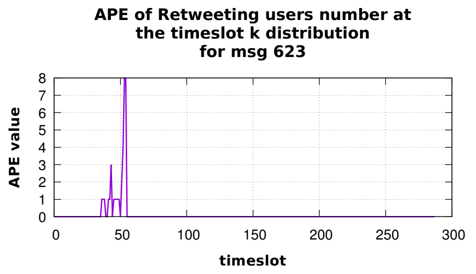


Figure D.79. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 623.

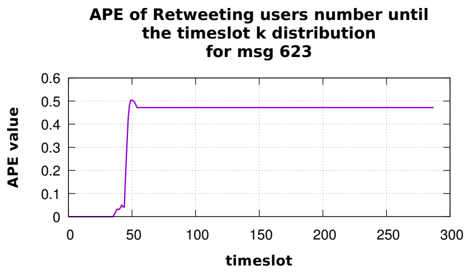


Figure D.80. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 623.

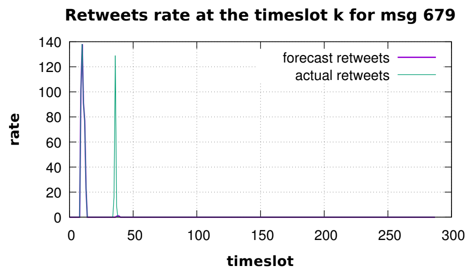


Figure D.81. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 679.

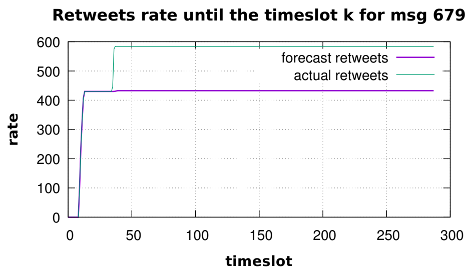


Figure D.82. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 679.

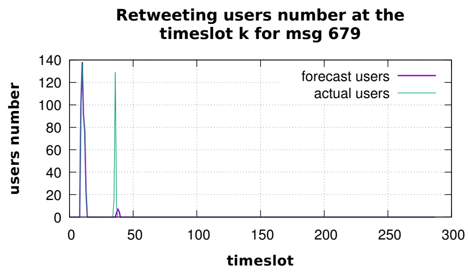


Figure D.83. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 679.

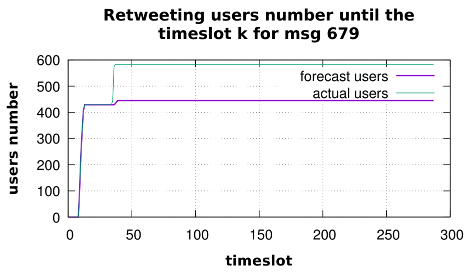


Figure D.84. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 679.

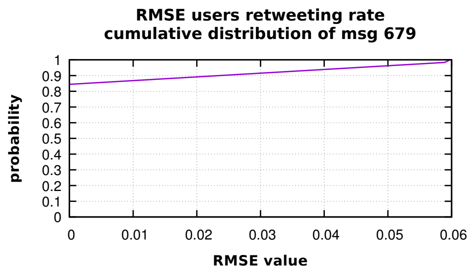


Figure D.85. Graphic showing the RMSE cumulative distribution, for message 679.

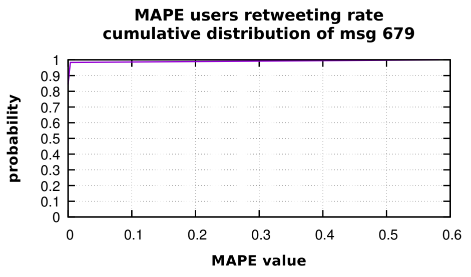


Figure D.86. Graphic showing the MAPE cumulative distribution, for message 679.

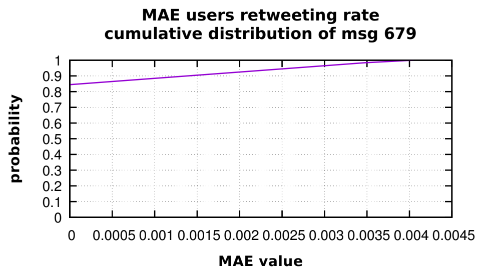


Figure D.87. Graphic showing the MAE cumulative distribution, for message 679.

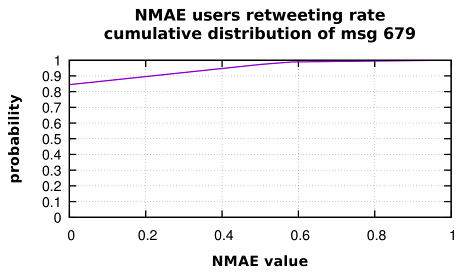


Figure D.88. Graphic showing the NMAE cumulative distribution, for message 679.

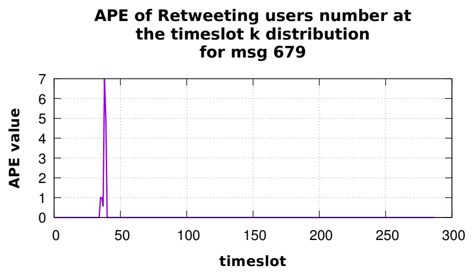


Figure D.89. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 679.

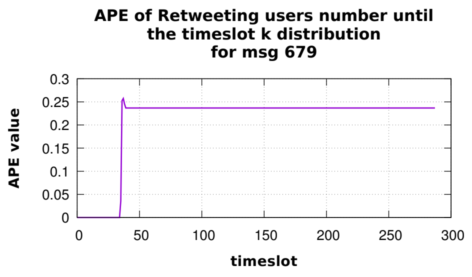


Figure D.90. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 679.

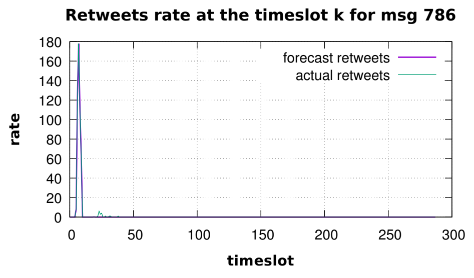


Figure D.91. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 786.

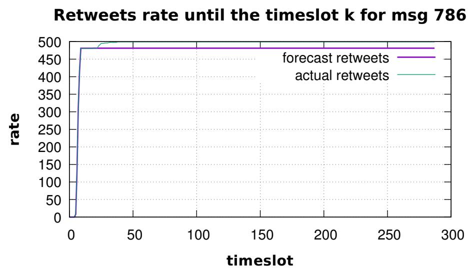


Figure D.92. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 786.

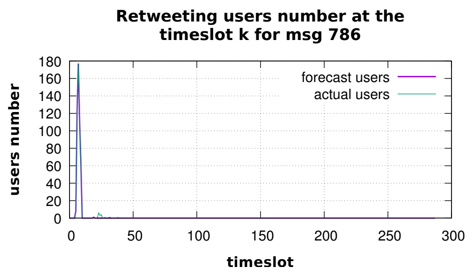


Figure D.93. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 786.

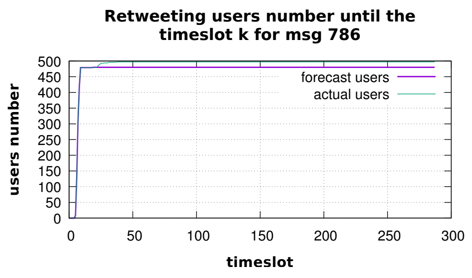


Figure D.94. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 786.

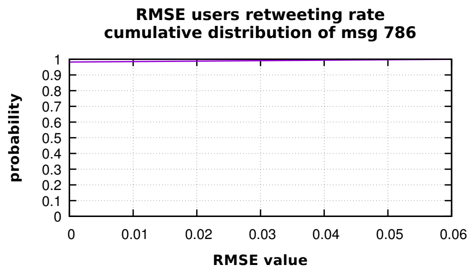


Figure D.95. Graphic showing the RMSE cumulative distribution, for message 786.

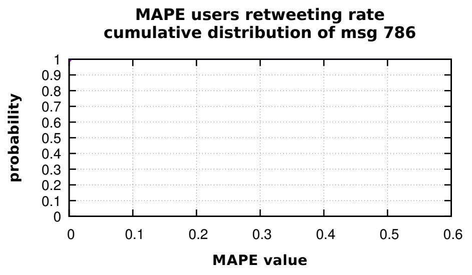


Figure D.96. Graphic showing the MAPE cumulative distribution, for message 786.

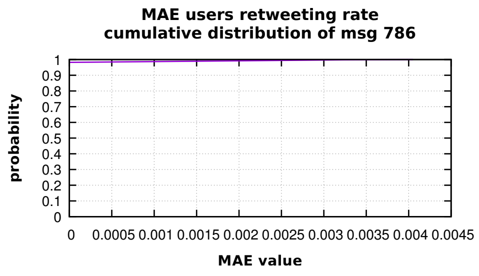


Figure D.97. Graphic showing the MAE cumulative distribution, for message 786.

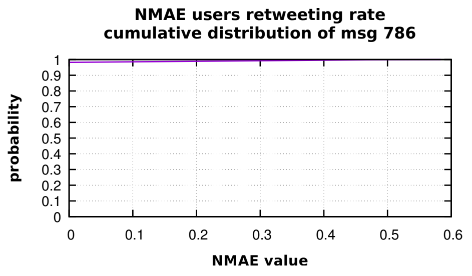


Figure D.98. Graphic showing the NMAE cumulative distribution, for message 786.

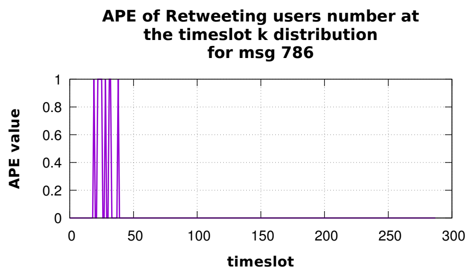


Figure D.99. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 786.

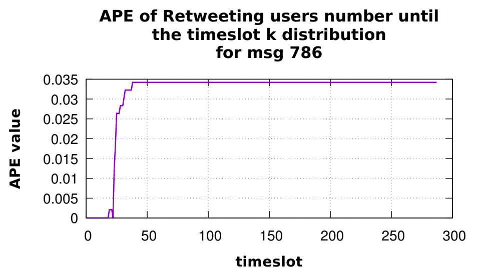


Figure D.100. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 786.

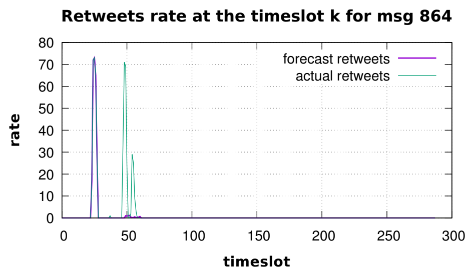


Figure D.101. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 864.

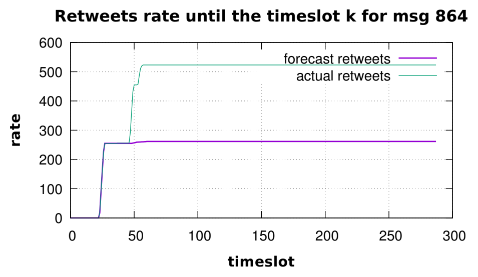


Figure D.102. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 864.

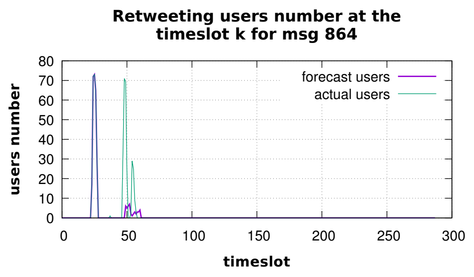


Figure D.103. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 864.

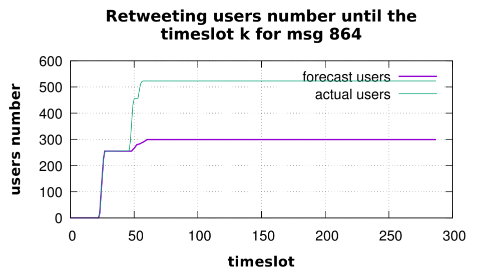


Figure D.104. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 864.

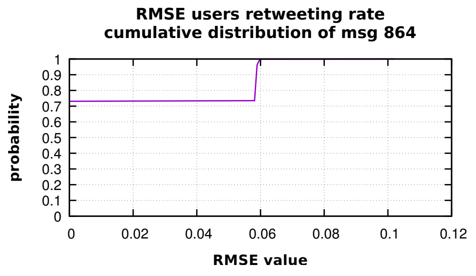


Figure D.105. Graphic showing the RMSE cumulative distribution, for message 864.

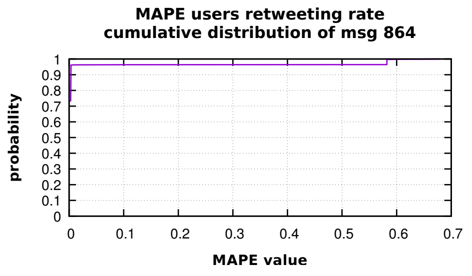


Figure D.106. Graphic showing the MAPE cumulative distribution, for message 864.

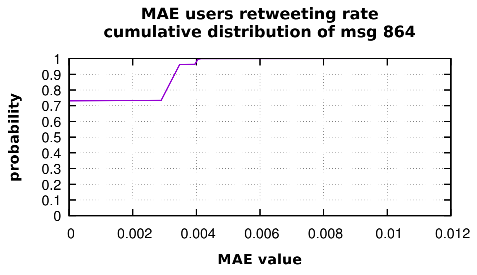


Figure D.107. Graphic showing the MAE cumulative distribution, for message 864.

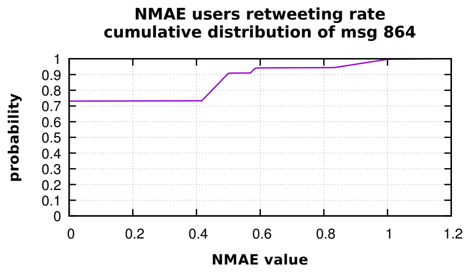


Figure D.108. Graphic showing the NMAE cumulative distribution, for message 864.

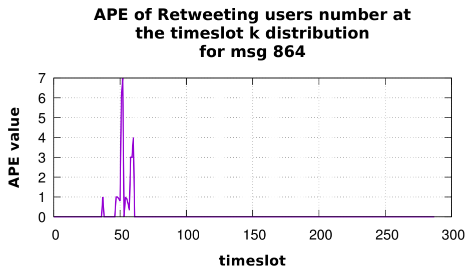


Figure D.109. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 864.

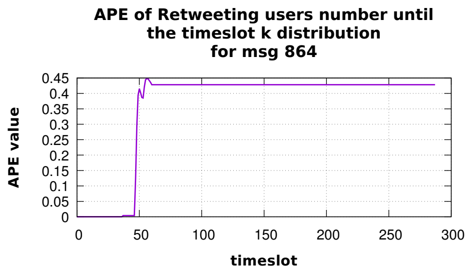


Figure D.110. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 864.

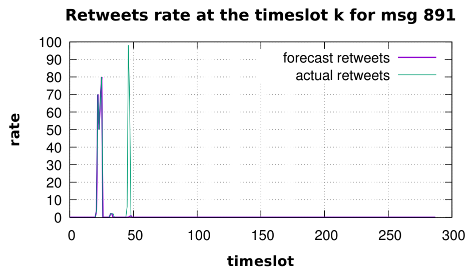


Figure D.111. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 891.

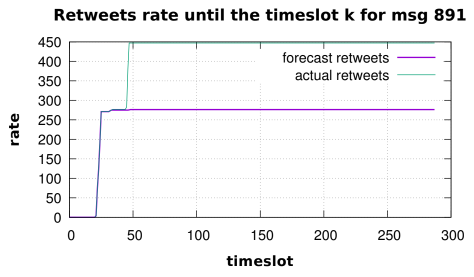


Figure D.112. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 891.

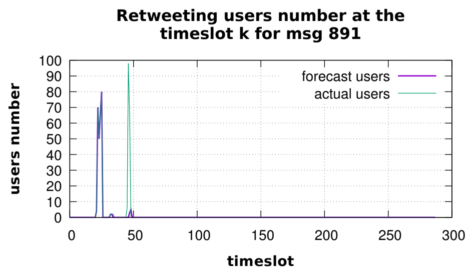


Figure D.113. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 891.

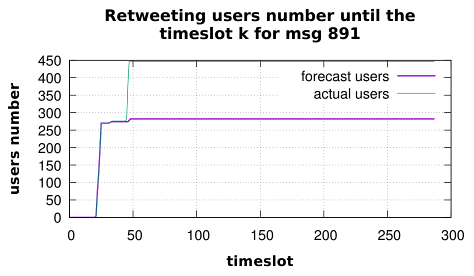


Figure D.114. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 891.

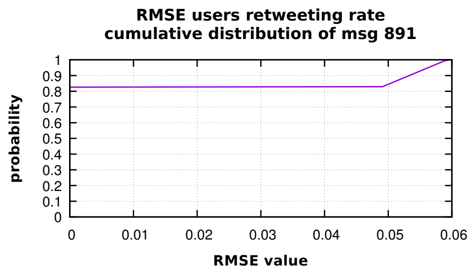


Figure D.115. Graphic showing the RMSE cumulative distribution, for message 891.

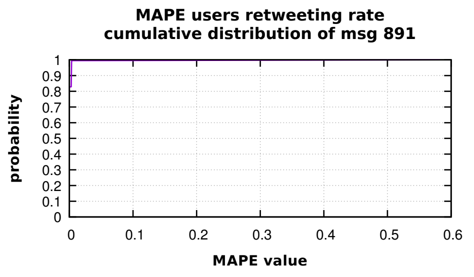


Figure D.116. Graphic showing the MAPE cumulative distribution, for message 891.

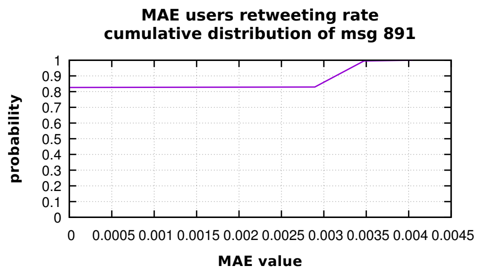


Figure D.117. Graphic showing the MAE cumulative distribution, for message 891.

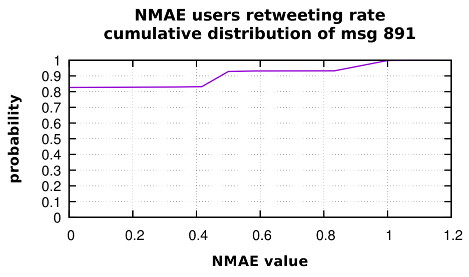


Figure D.118. Graphic showing the NMAE cumulative distribution, for message 891.

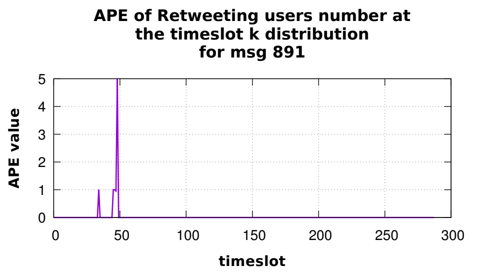


Figure D.119. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 891.

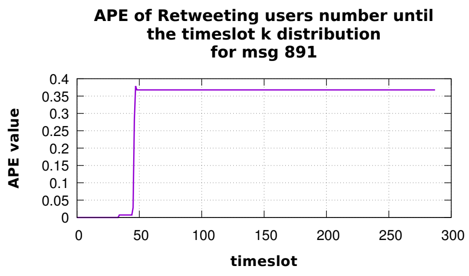


Figure D.120. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 891.

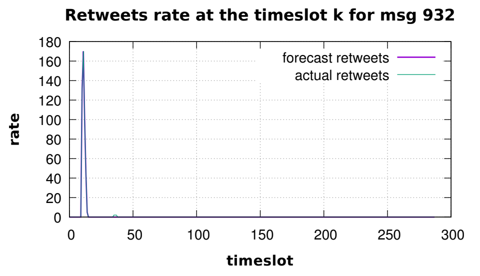


Figure D.121. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 932.

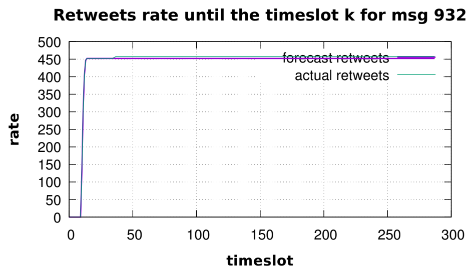


Figure D.122. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 932.

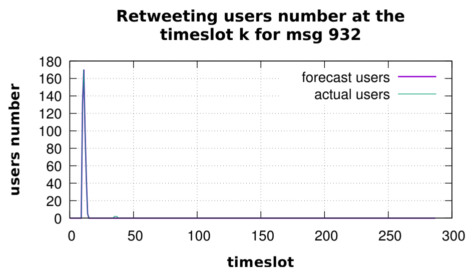


Figure D.123. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 932.

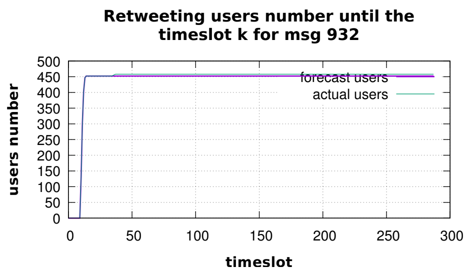


Figure D.124. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 932.

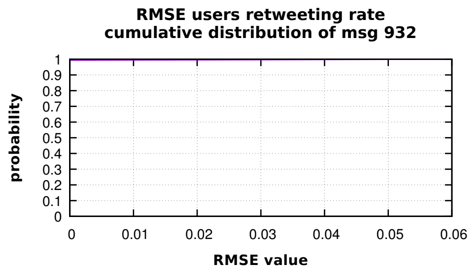


Figure D.125. Graphic showing the RMSE cumulative distribution, for message 932.

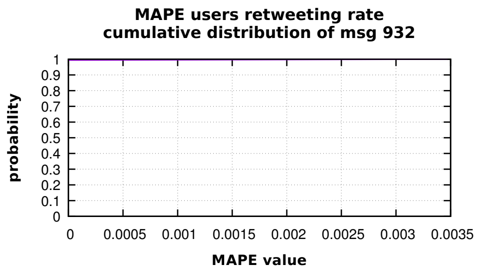


Figure D.126. Graphic showing the MAPE cumulative distribution, for message 932.

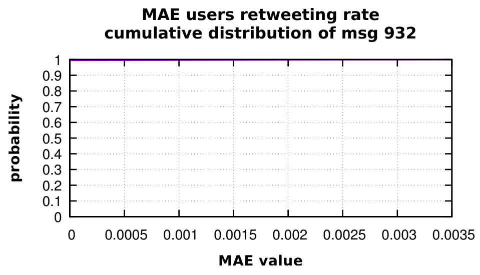


Figure D.127. Graphic showing the MAE cumulative distribution, for message 932.

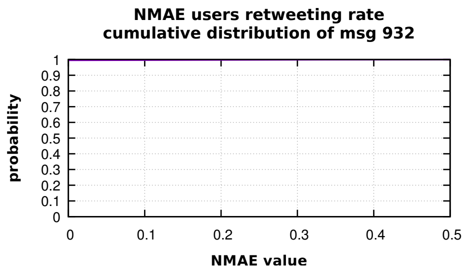


Figure D.128. Graphic showing the NMAE cumulative distribution, for message 932.

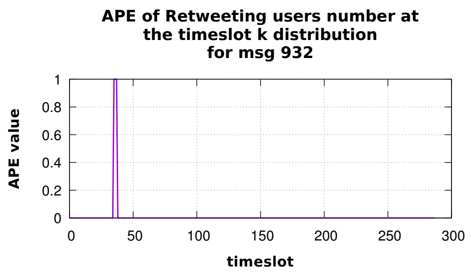


Figure D.129. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 932.

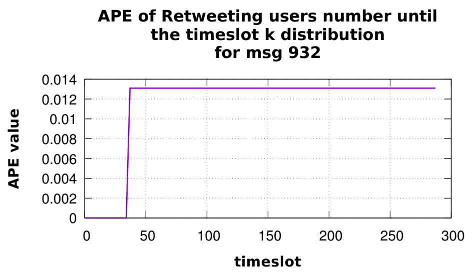


Figure D.130. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 932.

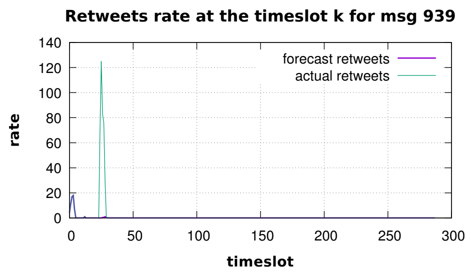


Figure D.131. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 939.

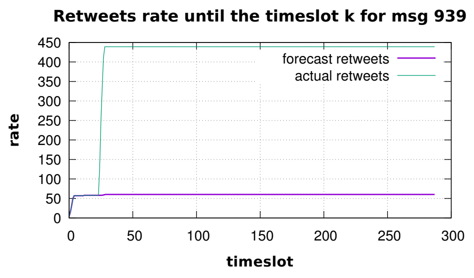


Figure D.132. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 939.

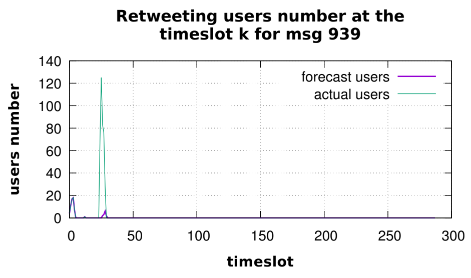


Figure D.133. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 939.

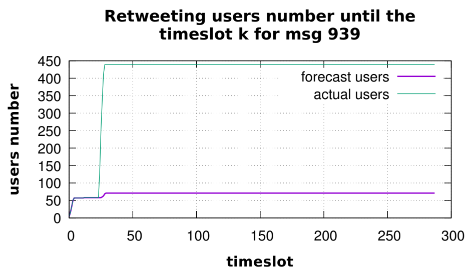


Figure D.134. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 939.

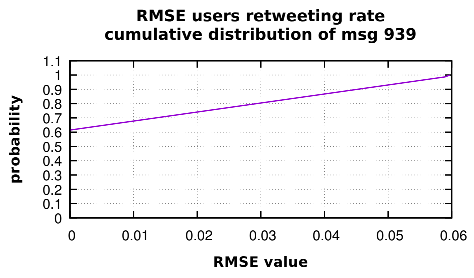


Figure D.135. Graphic showing the RMSE cumulative distribution, for message 939.

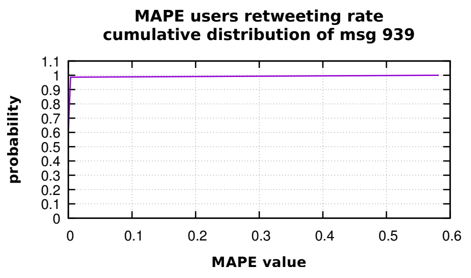


Figure D.136. Graphic showing the MAPE cumulative distribution, for message 939.

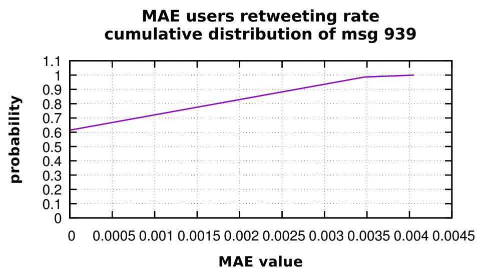


Figure D.137. Graphic showing the MAE cumulative distribution, for message 939.

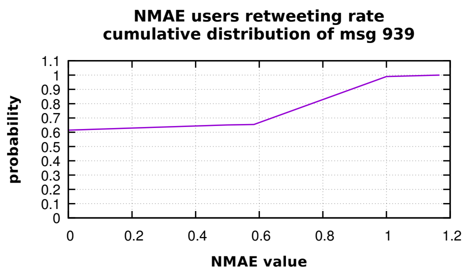


Figure D.138. Graphic showing the NMAE cumulative distribution, for message 939.

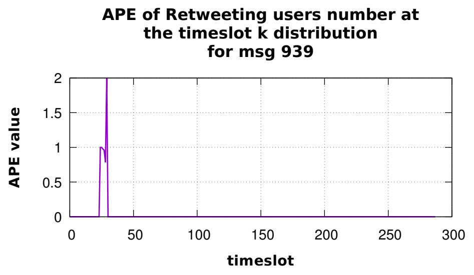


Figure D.139. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 939.

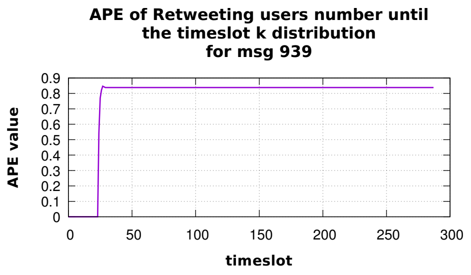


Figure D.140. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 939.

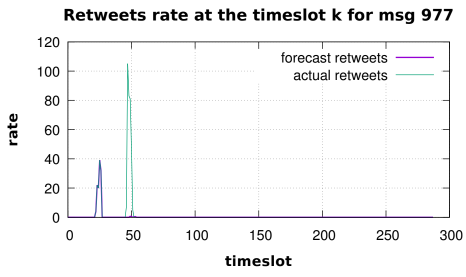


Figure D.141. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 977.

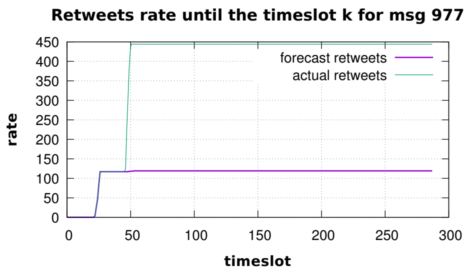


Figure D.142. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 977.

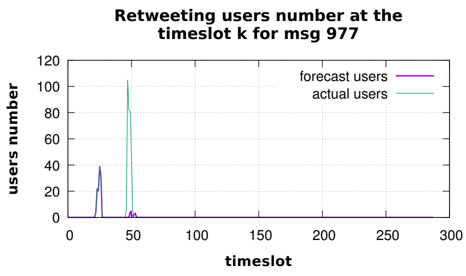


Figure D.143. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 977.

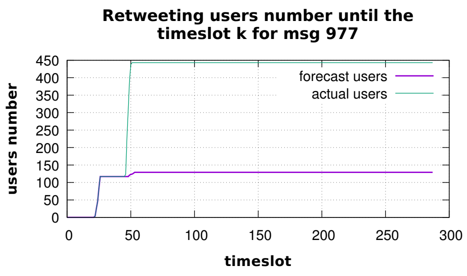


Figure D.144. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 977.

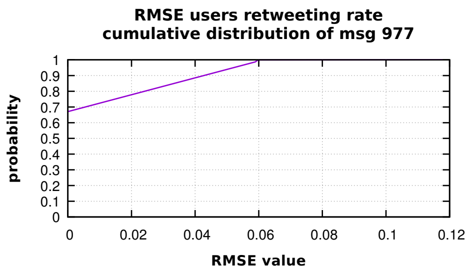


Figure D.145. Graphic showing the RMSE cumulative distribution, for message 977.

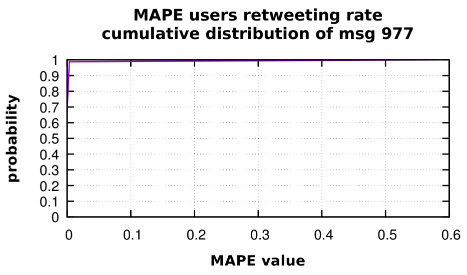


Figure D.146. Graphic showing the MAPE cumulative distribution, for message 977.

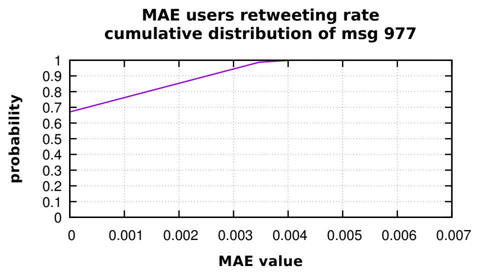


Figure D.147. Graphic showing the MAE cumulative distribution, for message 977.

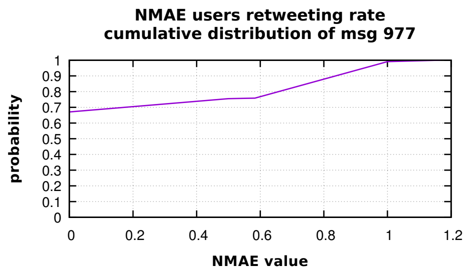


Figure D.148. Graphic showing the NMAE cumulative distribution, for message 977.

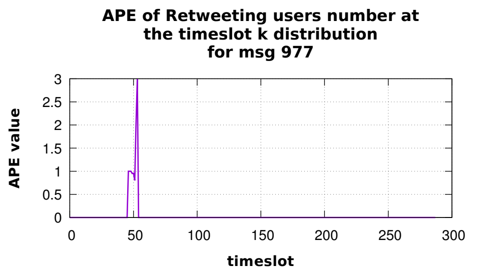


Figure D.149. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 977.

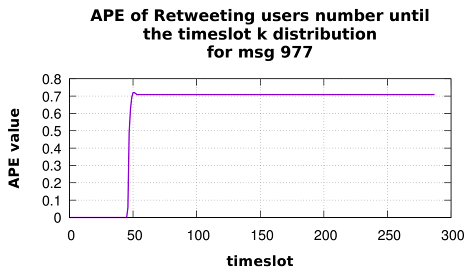


Figure D.150. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 977.

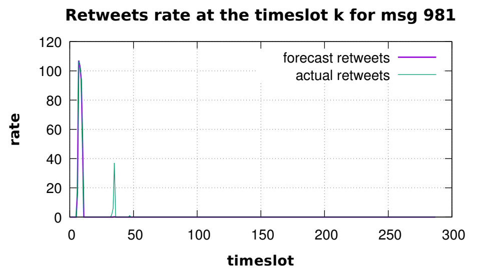


Figure D.151. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 981.

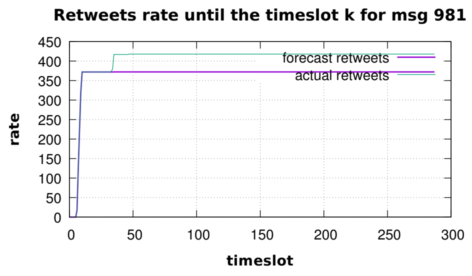


Figure D.152. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 981.

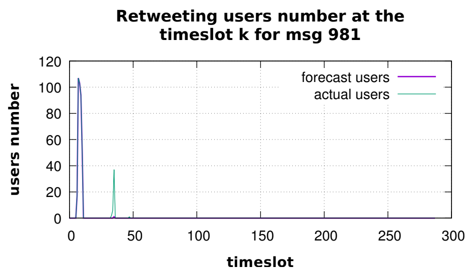


Figure D.153. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 981.

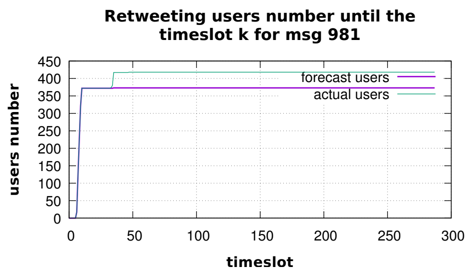


Figure D.154. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 981.

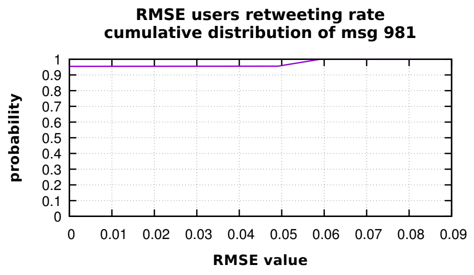


Figure D.155. Graphic showing the RMSE cumulative distribution, for message 981.

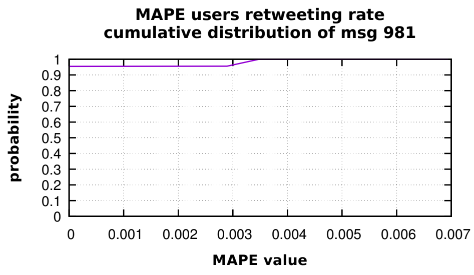


Figure D.156. Graphic showing the MAPE cumulative distribution, for message 981.

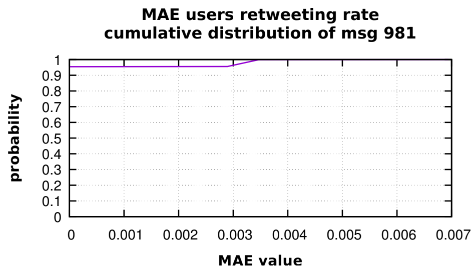


Figure D.157. Graphic showing the MAE cumulative distribution, for message 981.

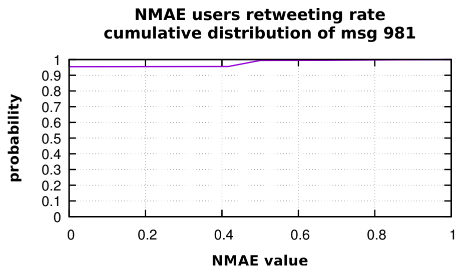


Figure D.158. Graphic showing the NMAE cumulative distribution, for message 981.

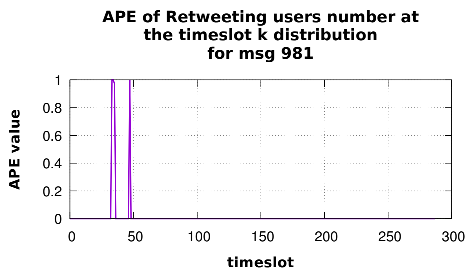


Figure D.159. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 981.

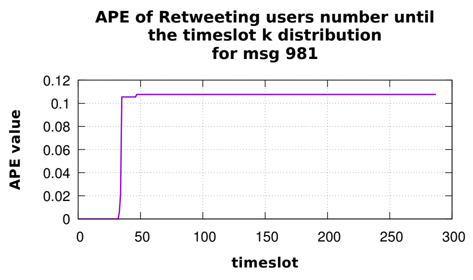


Figure D.160. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 981.

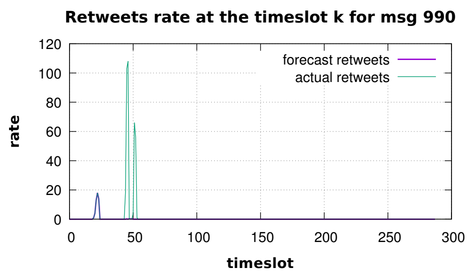


Figure D.161. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 990.

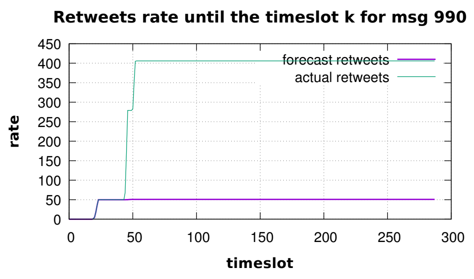


Figure D.162. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 990.

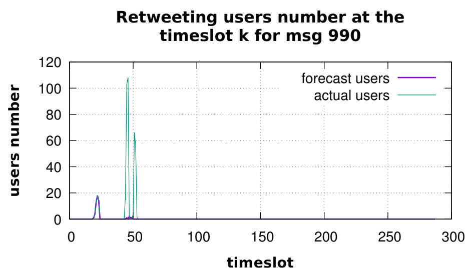


Figure D.163. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 990.

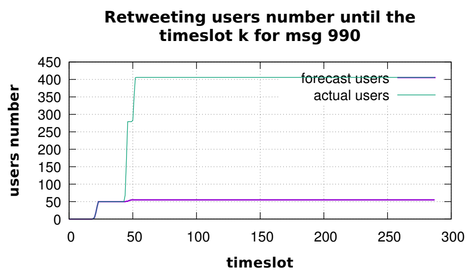


Figure D.164. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 990.

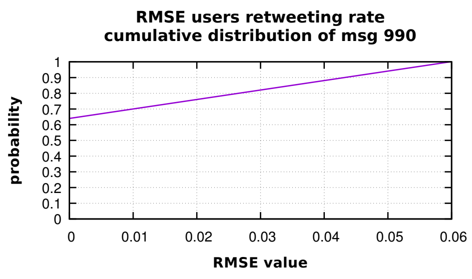


Figure D.165. Graphic showing the RMSE cumulative distribution, for message 990.

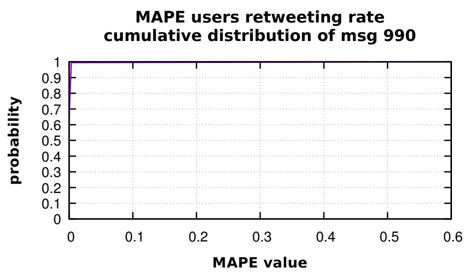


Figure D.166. Graphic showing the MAPE cumulative distribution, for message 990.

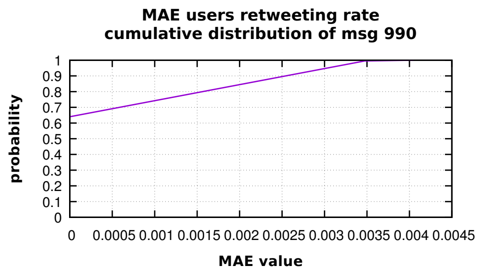


Figure D.167. Graphic showing the MAE cumulative distribution, for message 990.

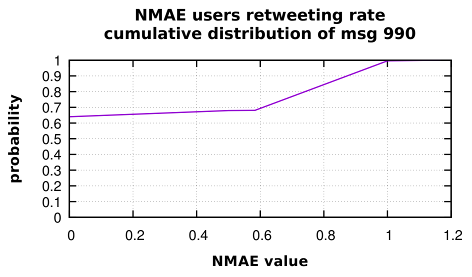


Figure D.168. Graphic showing the NMAE cumulative distribution, for message 990.

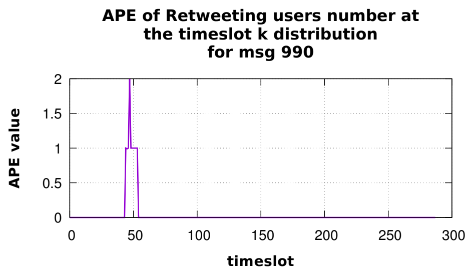


Figure D.169. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 990.

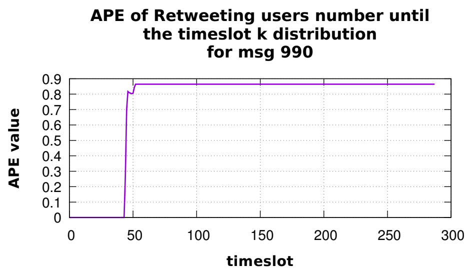


Figure D.170. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 990.

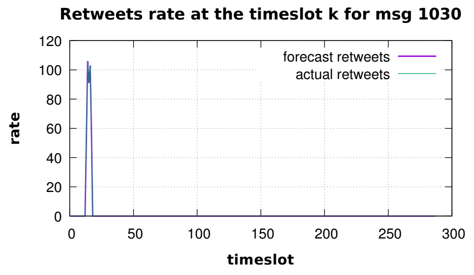


Figure D.171. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1030.

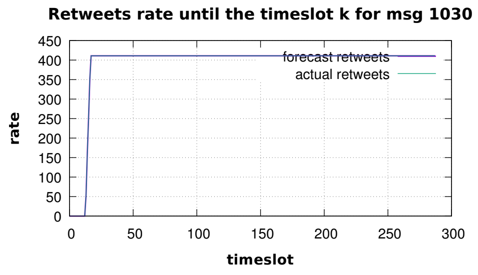


Figure D.172. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1030.

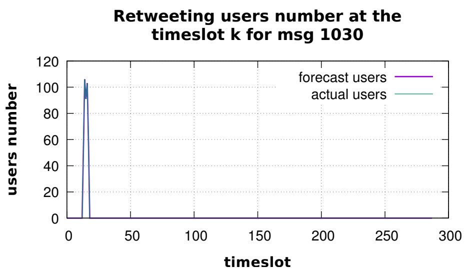


Figure D.173. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1030.

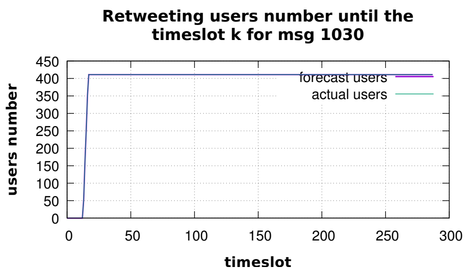


Figure D.174. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1030.

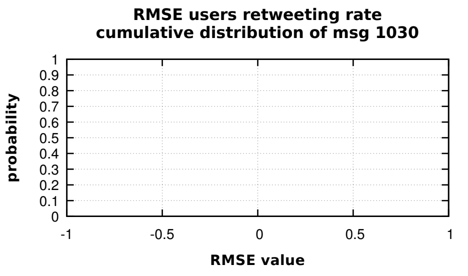


Figure D.175. Graphic showing the RMSE cumulative distribution, for message 1030.

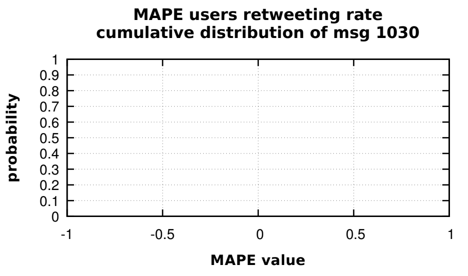


Figure D.176. Graphic showing the MAPE cumulative distribution, for message 1030.

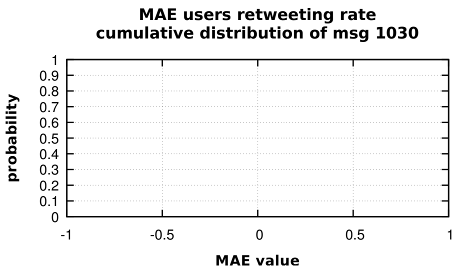


Figure D.177. Graphic showing the MAE cumulative distribution, for message 1030.

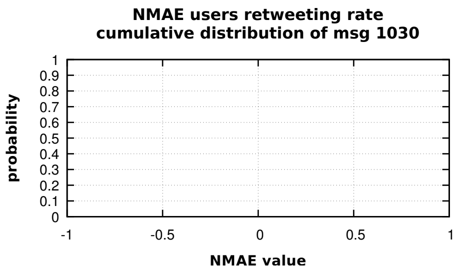


Figure D.178. Graphic showing the NMAE cumulative distribution, for message 1030.

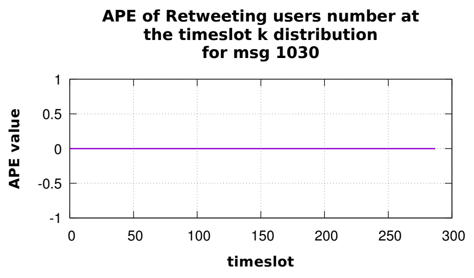


Figure D.179. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1030.

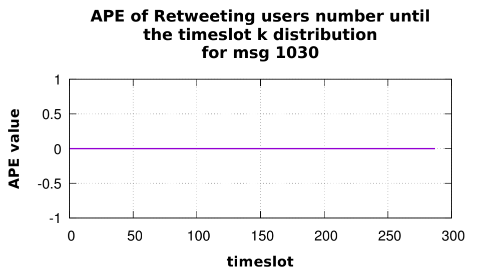


Figure D.180. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1030.

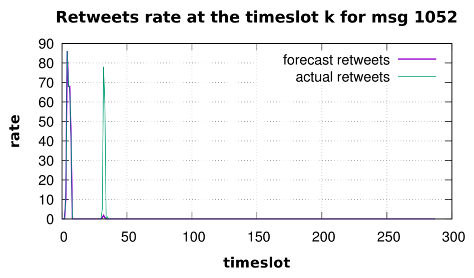


Figure D.181. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1052.

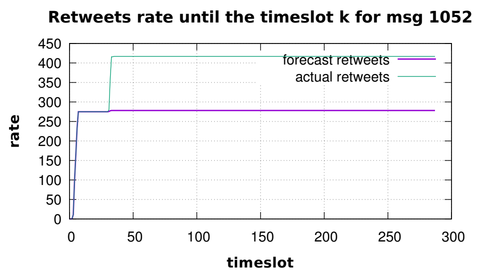


Figure D.182. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1052.

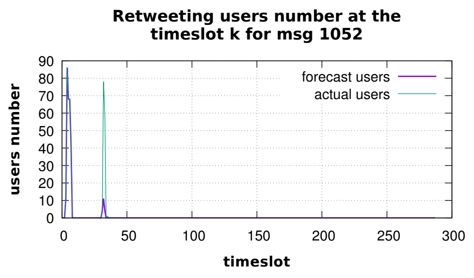


Figure D.183. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1052.

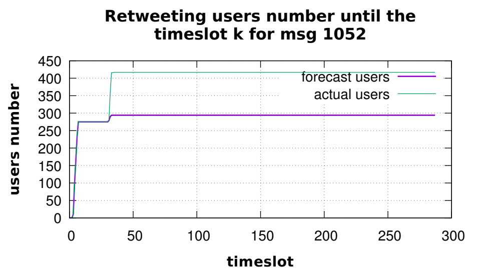


Figure D.184. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1052.

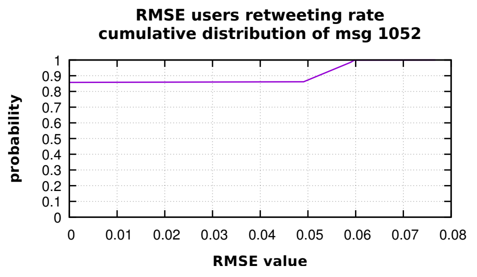


Figure D.185. Graphic showing the RMSE cumulative distribution, for message 1052.

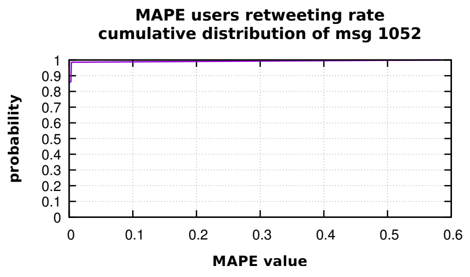


Figure D.186. Graphic showing the MAPE cumulative distribution, for message 1052.

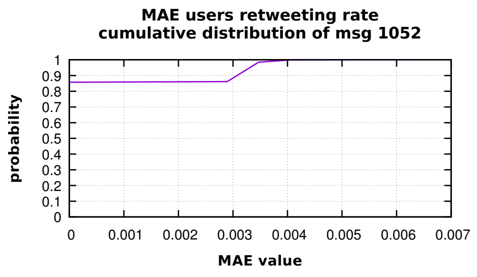


Figure D.187. Graphic showing the MAE cumulative distribution, for message 1052.

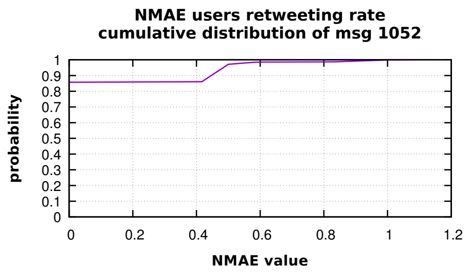


Figure D.188. Graphic showing the NMAE cumulative distribution, for message 1052.

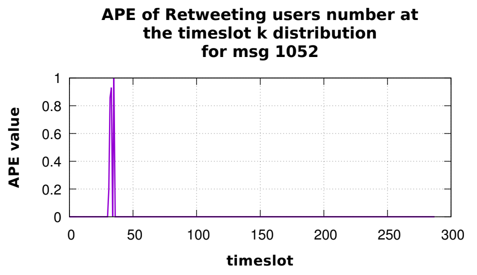


Figure D.189. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1052.

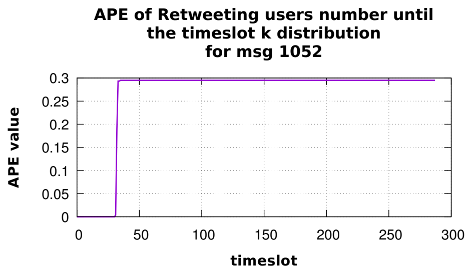


Figure D.190. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1052.

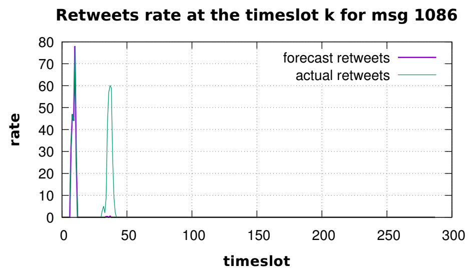


Figure D.191. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1086.

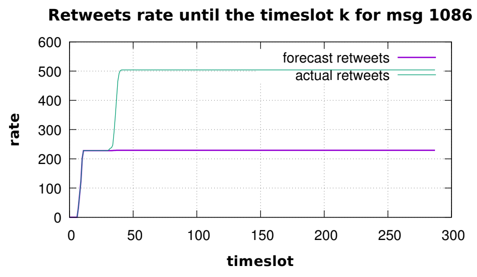


Figure D.192. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1086.

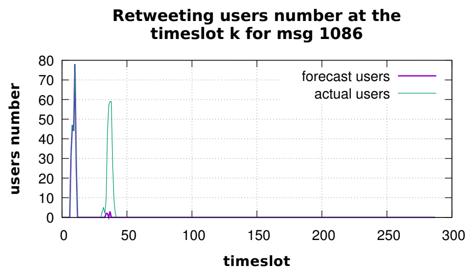


Figure D.193. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1086.

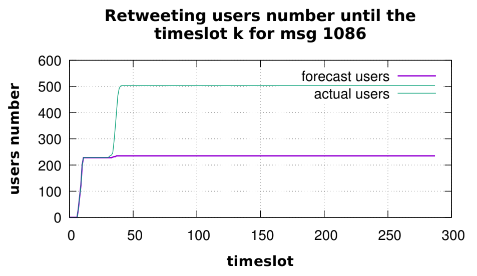


Figure D.194. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1086.

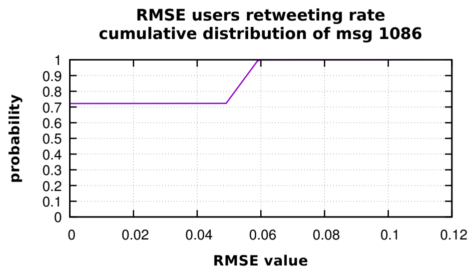


Figure D.195. Graphic showing the RMSE cumulative distribution, for message 1086.

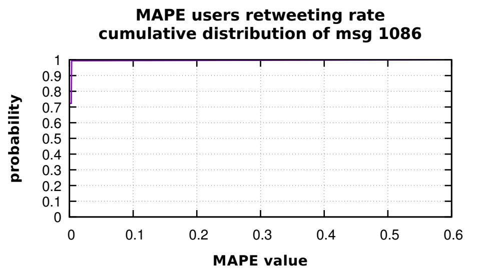


Figure D.196. Graphic showing the MAPE cumulative distribution, for message 1086.

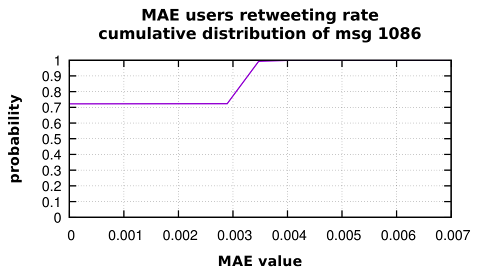


Figure D.197. Graphic showing the MAE cumulative distribution, for message 1086.

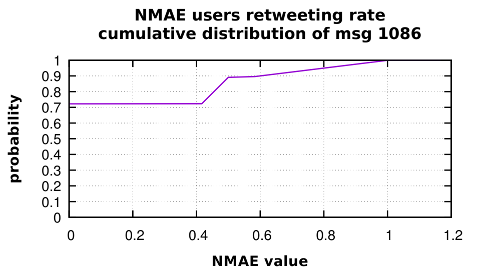


Figure D.198. Graphic showing the NMAE cumulative distribution, for message 1086.

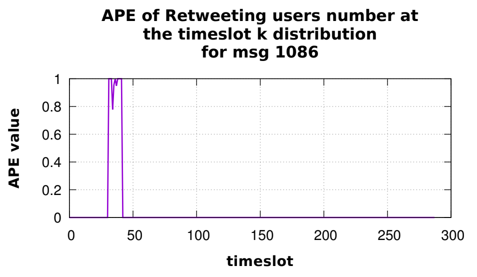


Figure D.199. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1086.

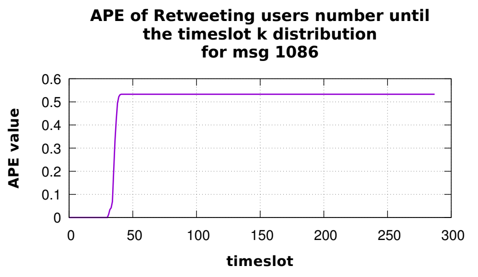


Figure D.200. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1086.

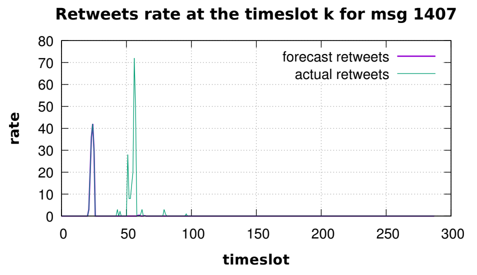


Figure D.201. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1407.

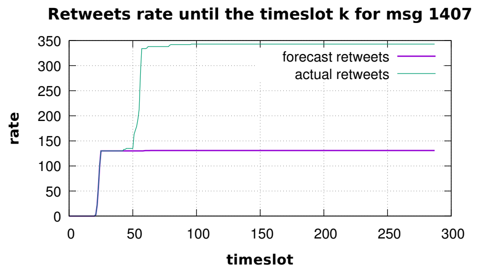


Figure D.202. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1407.

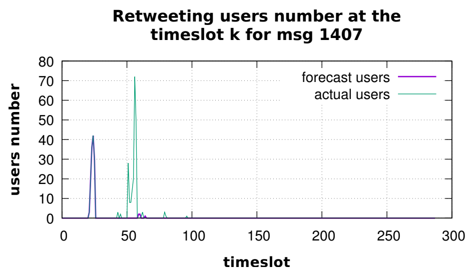


Figure D.203. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1407.

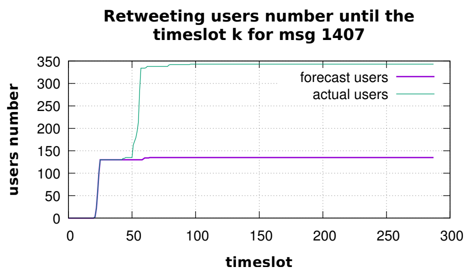


Figure D.204. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1407.

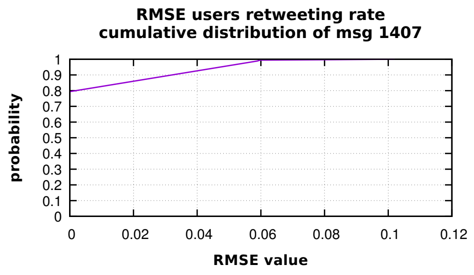


Figure D.205. Graphic showing the RMSE cumulative distribution, for message 1407.

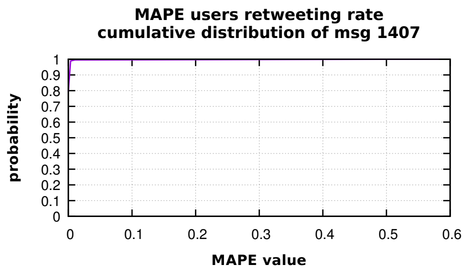


Figure D.206. Graphic showing the MAPE cumulative distribution, for message 1407.

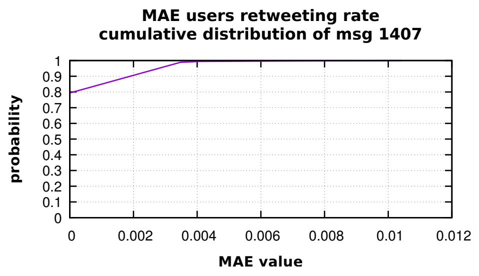


Figure D.207. Graphic showing the MAE cumulative distribution, for message 1407.

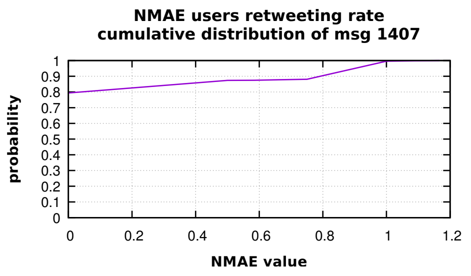


Figure D.208. Graphic showing the NMAE cumulative distribution, for message 1407.

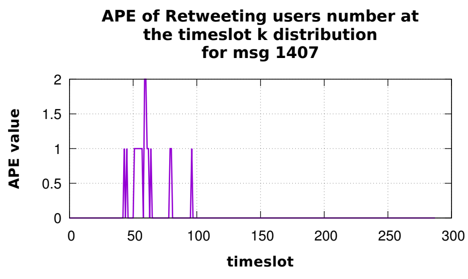


Figure D.209. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1407.

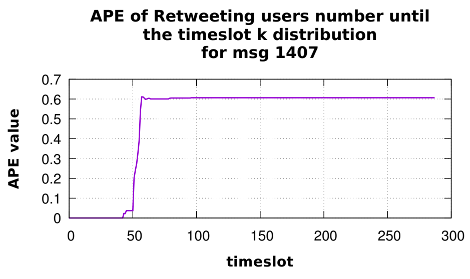


Figure D.210. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1407.

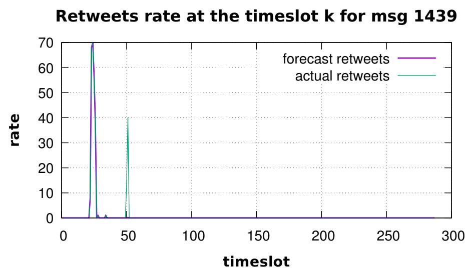


Figure D.211. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1439.

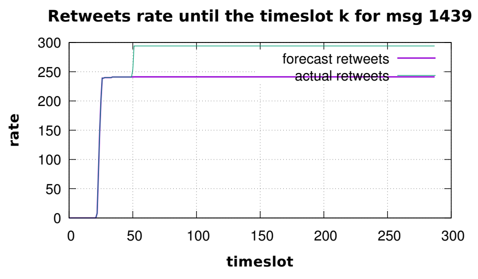


Figure D.212. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1439.

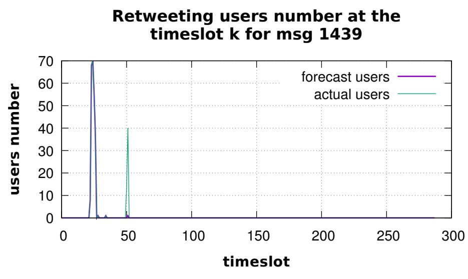


Figure D.213. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1439.

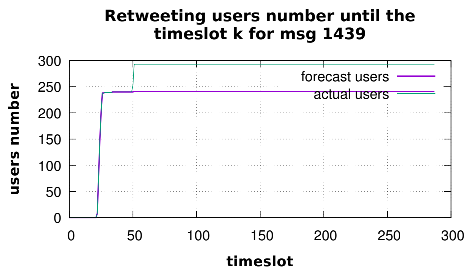


Figure D.214. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1439.

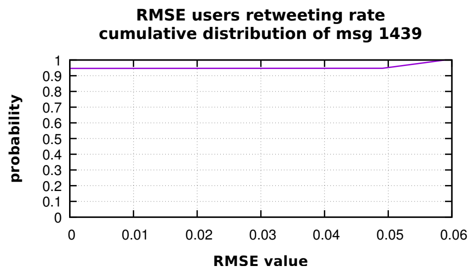


Figure D.215. Graphic showing the RMSE cumulative distribution, for message 1439.

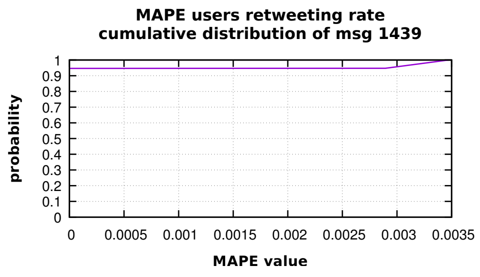


Figure D.216. Graphic showing the MAPE cumulative distribution, for message 1439.

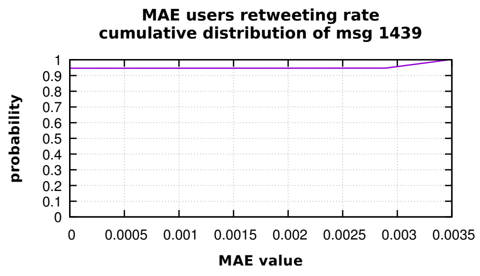


Figure D.217. Graphic showing the MAE cumulative distribution, for message 1439.

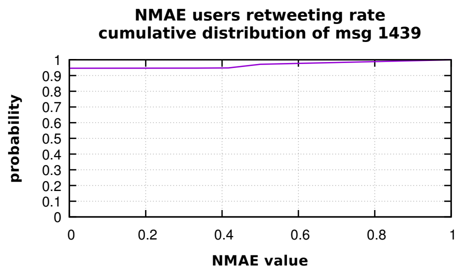


Figure D.218. Graphic showing the NMAE cumulative distribution, for message 1439.

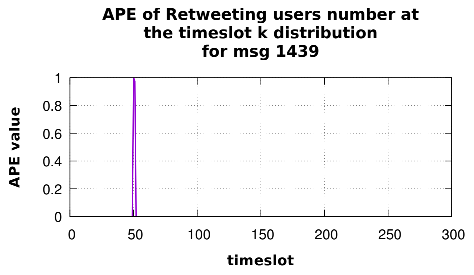


Figure D.219. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1439.

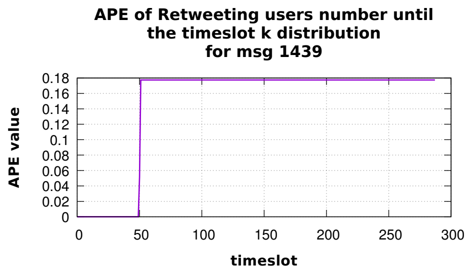


Figure D.220. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1439.

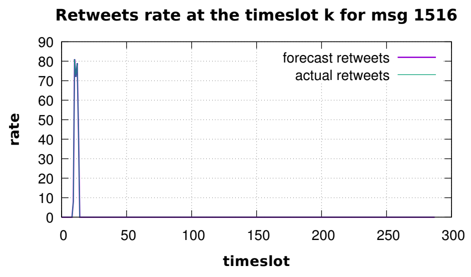


Figure D.221. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1516.

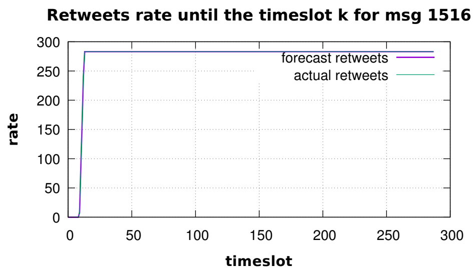


Figure D.222. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1516.

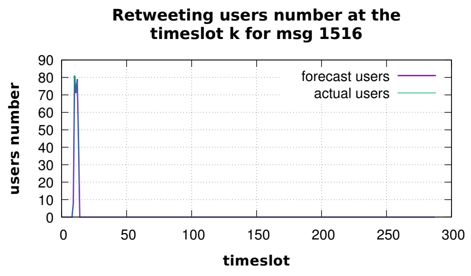


Figure D.223. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1516.

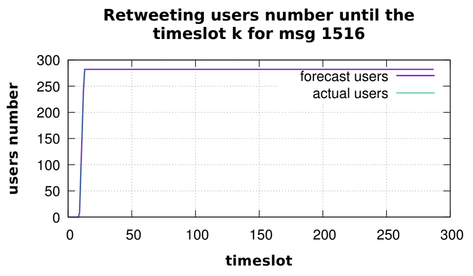


Figure D.224. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1516.

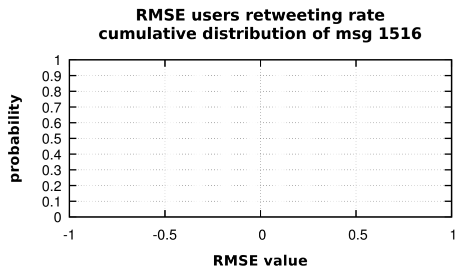


Figure D.225. Graphic showing the RMSE cumulative distribution, for message 1516.

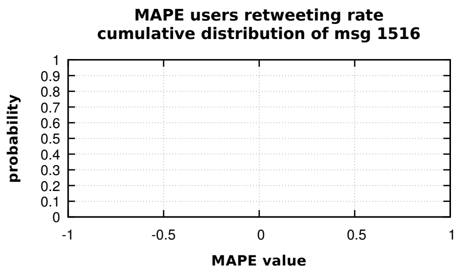


Figure D.226. Graphic showing the MAPE cumulative distribution, for message 1516.

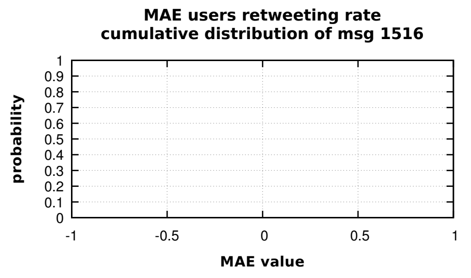


Figure D.227. Graphic showing the MAE cumulative distribution, for message 1516.

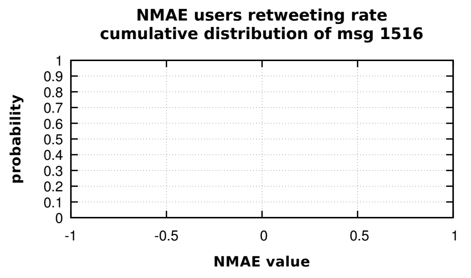


Figure D.228. Graphic showing the NMAE cumulative distribution, for message 1516.

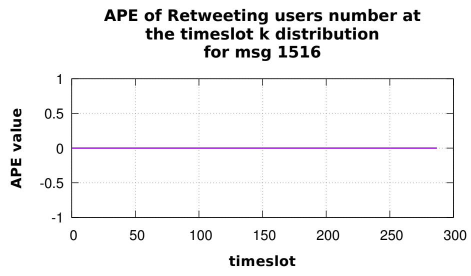


Figure D.229. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1516.

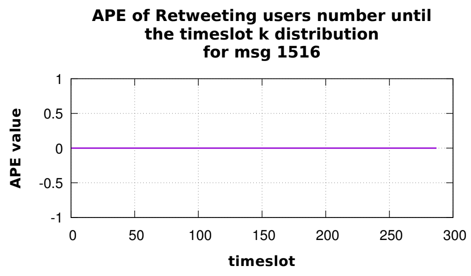


Figure D.230. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1516.

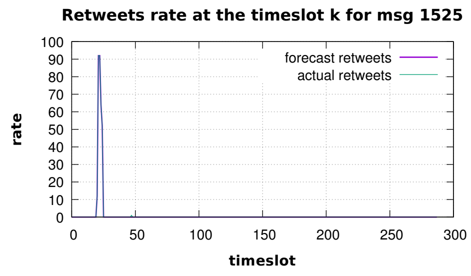


Figure D.231. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1525.

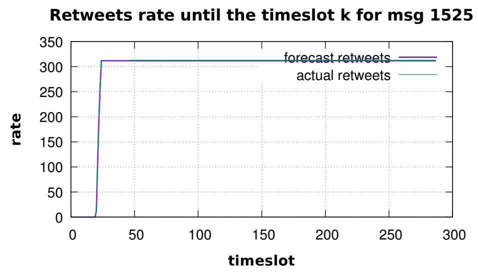


Figure D.232. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1525.

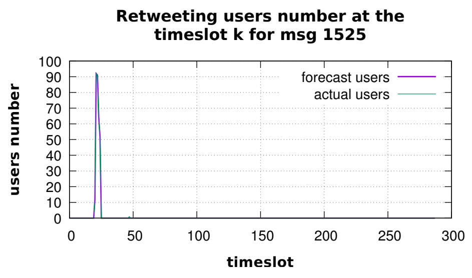


Figure D.233. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1525.

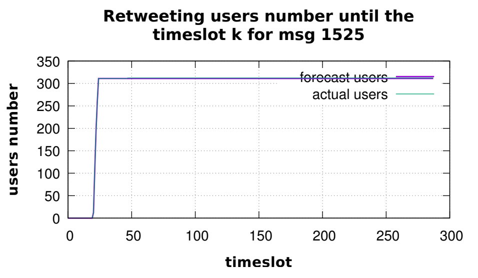


Figure D.234. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1525.

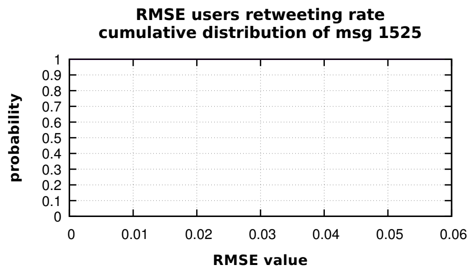


Figure D.235. Graphic showing the RMSE cumulative distribution, for message 1525.

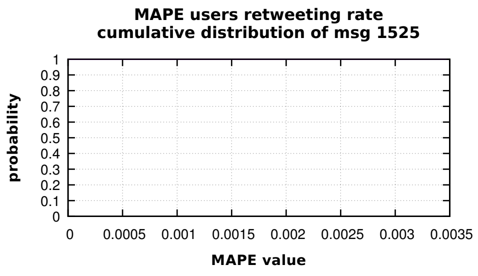


Figure D.236. Graphic showing the MAPE cumulative distribution, for message 1525.

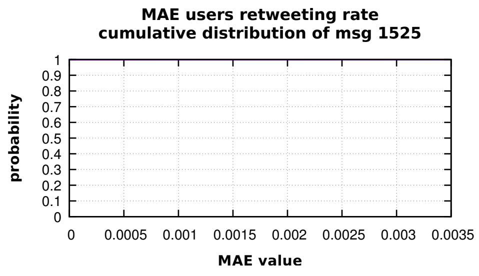


Figure D.237. Graphic showing the MAE cumulative distribution, for message 1525.

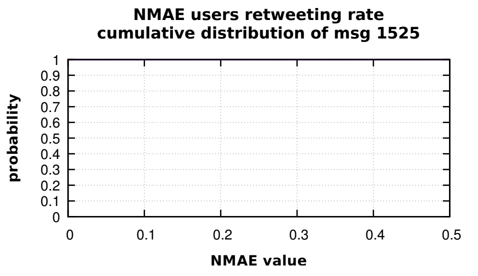


Figure D.238. Graphic showing the NMAE cumulative distribution, for message 1525.

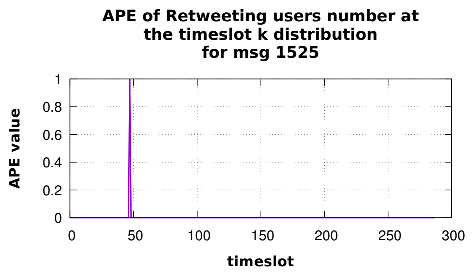


Figure D.239. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1525.

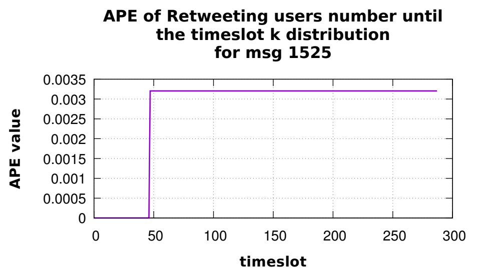


Figure D.240. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1525.

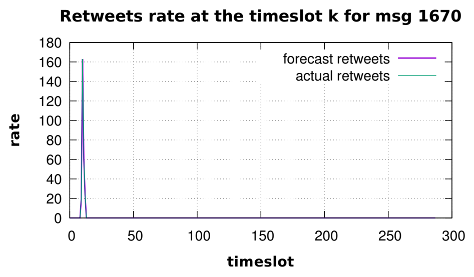


Figure D.241. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1670.

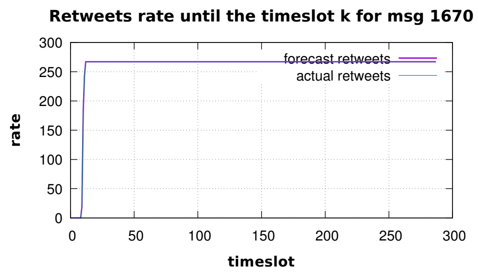


Figure D.242. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1670.

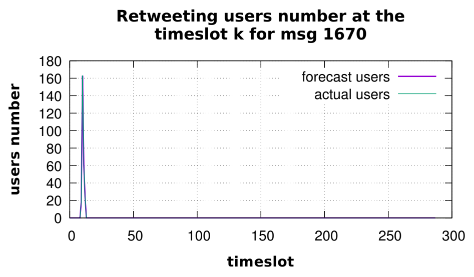


Figure D.243. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1670.

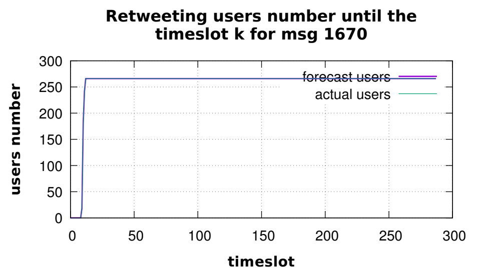


Figure D.244. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1670.

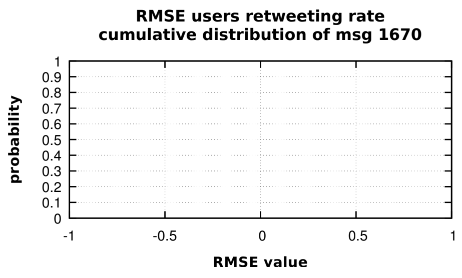


Figure D.245. Graphic showing the RMSE cumulative distribution, for message 1670.

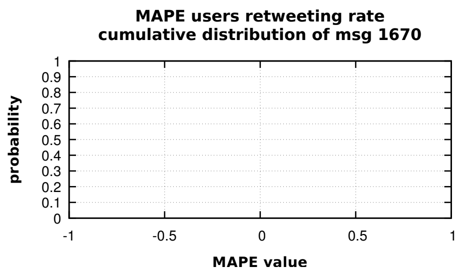


Figure D.246. Graphic showing the MAPE cumulative distribution, for message 1670.

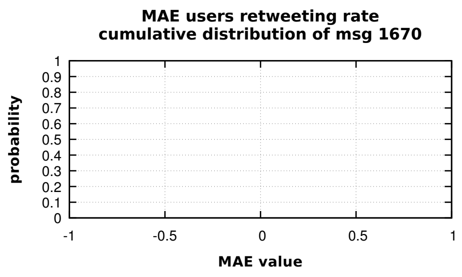


Figure D.247. Graphic showing the MAE cumulative distribution, for message 1670.

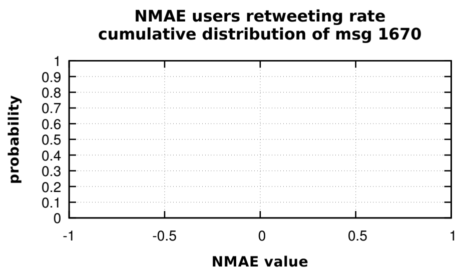


Figure D.248. Graphic showing the NMAE cumulative distribution, for message 1670.

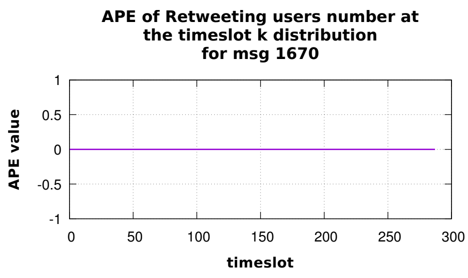


Figure D.249. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1670.

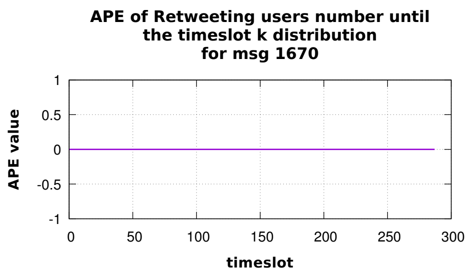


Figure D.250. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1670.

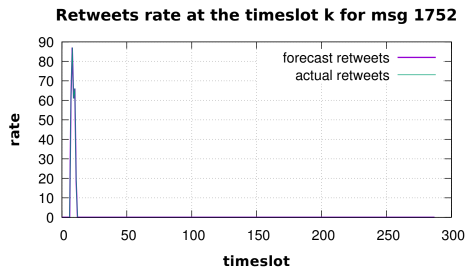


Figure D.251. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1752.

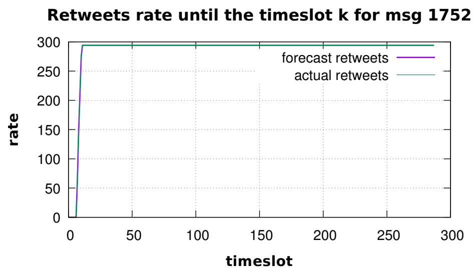


Figure D.252. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1752.

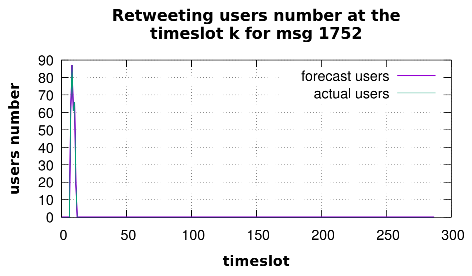


Figure D.253. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1752.

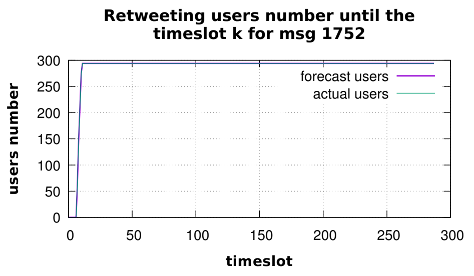


Figure D.254. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1752.

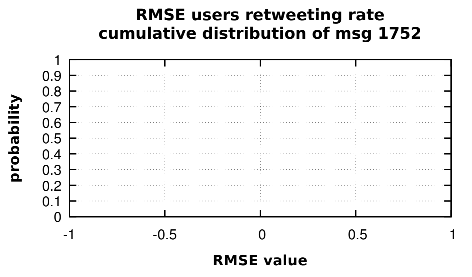


Figure D.255. Graphic showing the RMSE cumulative distribution, for message 1752.

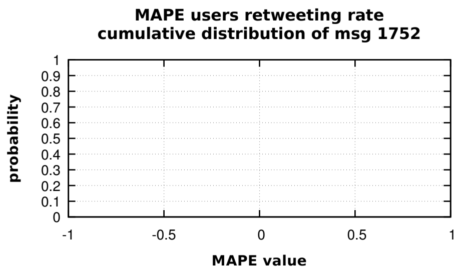


Figure D.256. Graphic showing the MAPE cumulative distribution, for message 1752.

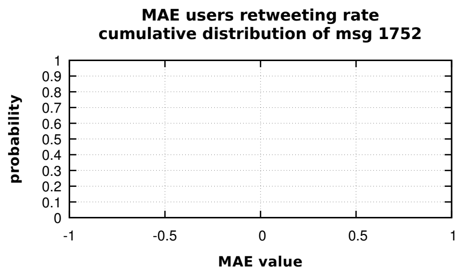


Figure D.257. Graphic showing the MAE cumulative distribution, for message 1752.

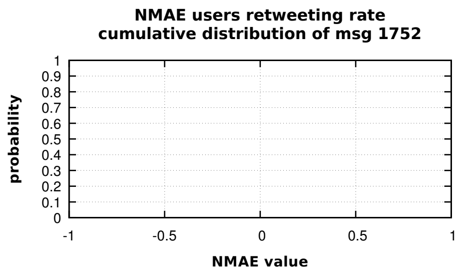


Figure D.258. Graphic showing the NMAE cumulative distribution, for message 1752.

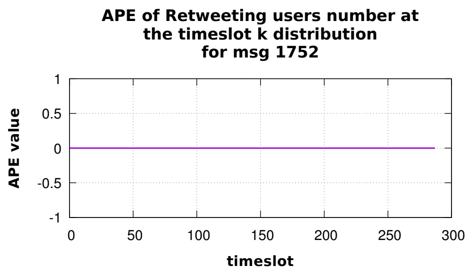


Figure D.259. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1752.

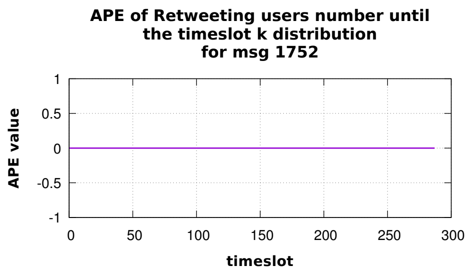


Figure D.260. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1752.

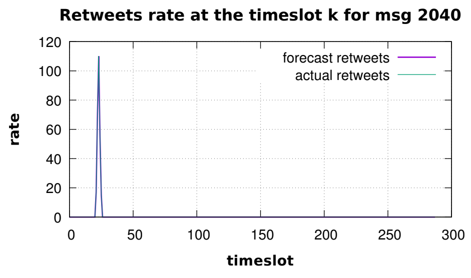


Figure D.261. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 2040.

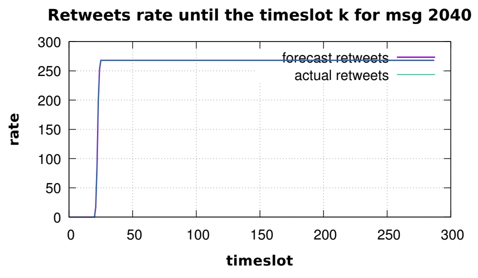
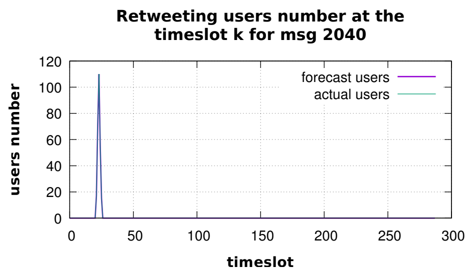


Figure D.262. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 2040.



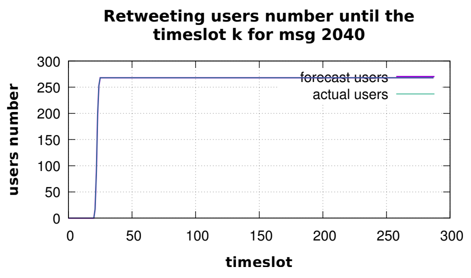
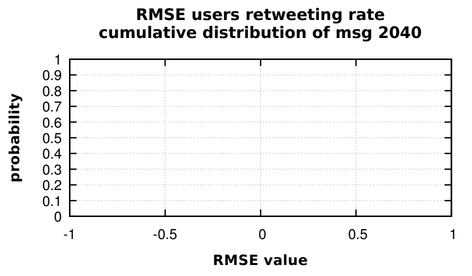


Figure D.263. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 2040.

Figure D.264. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 2040.



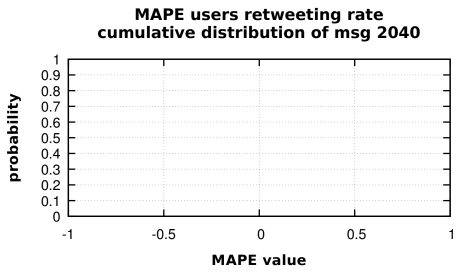
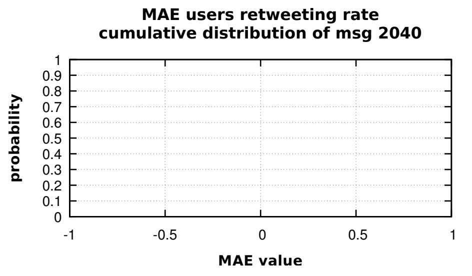


Figure D.265. Graphic showing the RMSE cumulative distribution, for message 2040.

Figure D.266. Graphic showing the MAPE cumulative distribution, for message 2040.



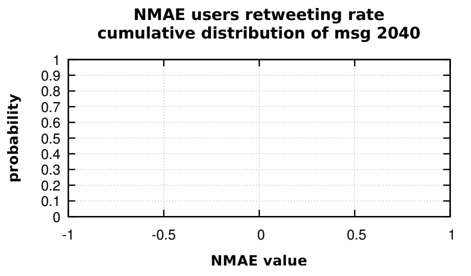


Figure D.267. Graphic showing the MAE cumulative distribution, for message 2040.

Figure D.268. Graphic showing the NMAE cumulative distribution, for message 2040.

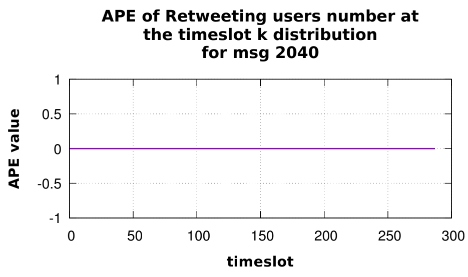


Figure D.269. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 2040.

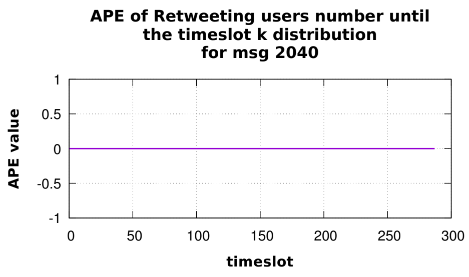


Figure D.270. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 2040.

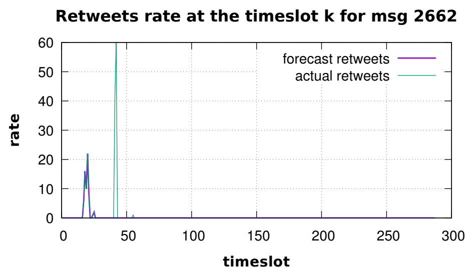


Figure D.271. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 2662.

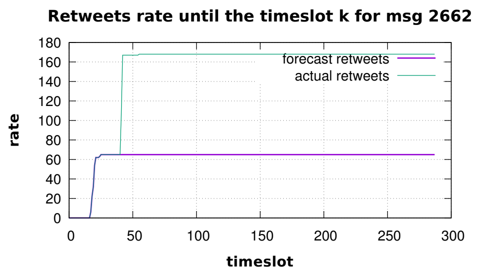


Figure D.272. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 2662.

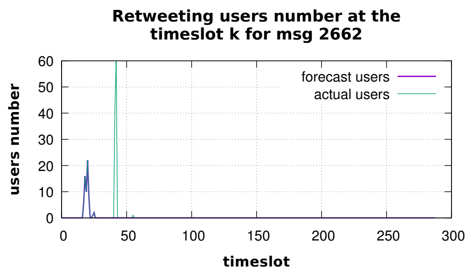


Figure D.273. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 2662.

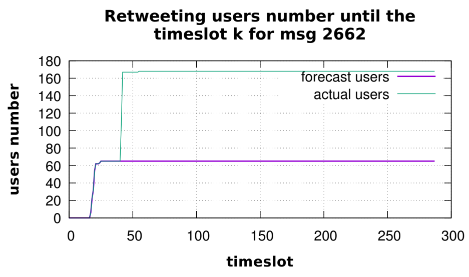


Figure D.274. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 2662.

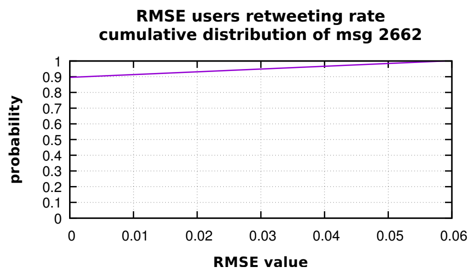


Figure D.275. Graphic showing the RMSE cumulative distribution, for message 2662.

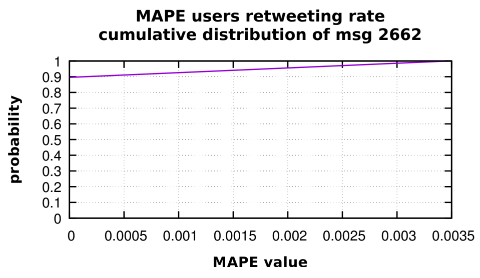


Figure D.276. Graphic showing the MAPE cumulative distribution, for message 2662.

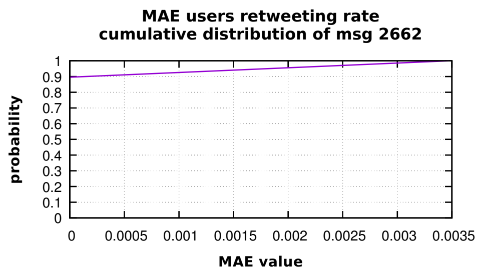


Figure D.277. Graphic showing the MAE cumulative distribution, for message 2662.

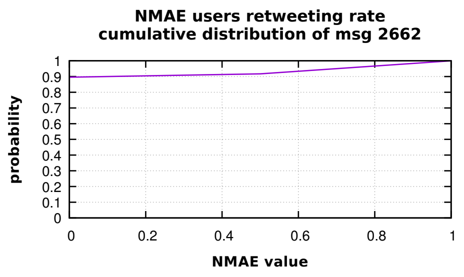


Figure D.278. Graphic showing the NMAE cumulative distribution, for message 2662.

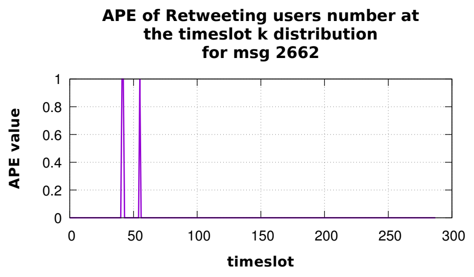


Figure D.279. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 2662.

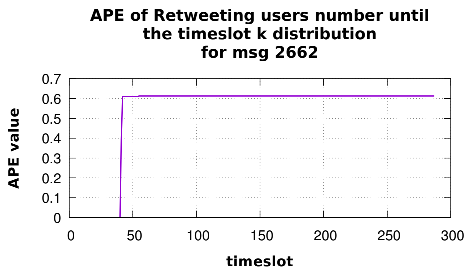


Figure D.280. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 2662.

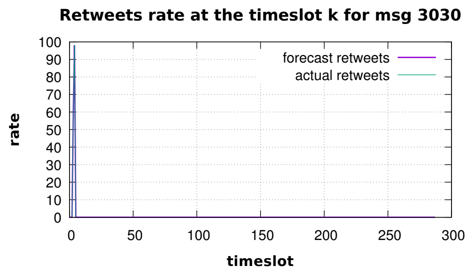


Figure D.281. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 3030.

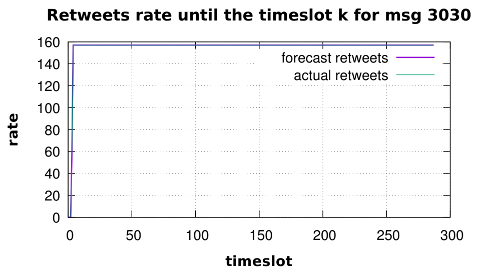


Figure D.282. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 3030.

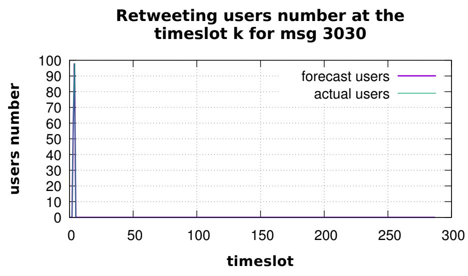


Figure D.283. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 3030.

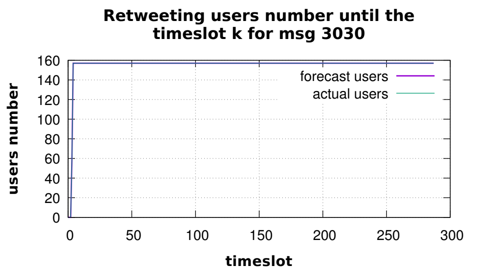


Figure D.284. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 3030.

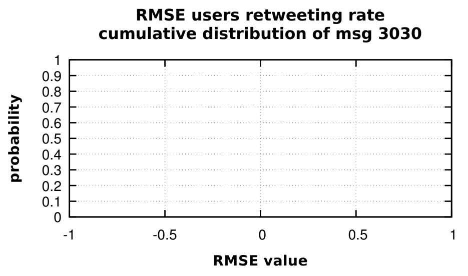


Figure D.285. Graphic showing the RMSE cumulative distribution, for message 3030.

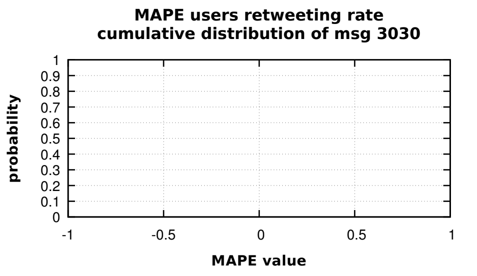


Figure D.286. Graphic showing the MAPE cumulative distribution, for message 3030.

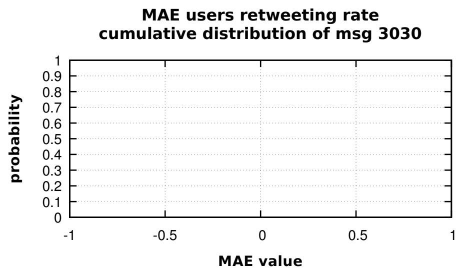


Figure D.287. Graphic showing the MAE cumulative distribution, for message 3030.

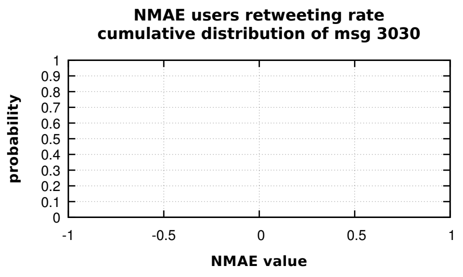


Figure D.288. Graphic showing the NMAE cumulative distribution, for message 3030.

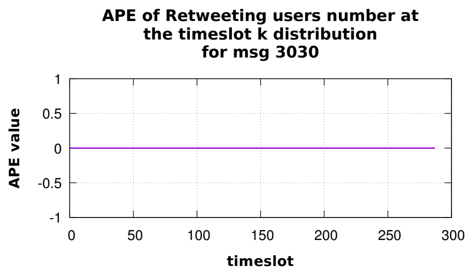


Figure D.289. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 3030.

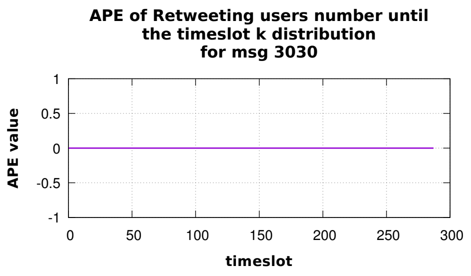


Figure D.290. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 3030.

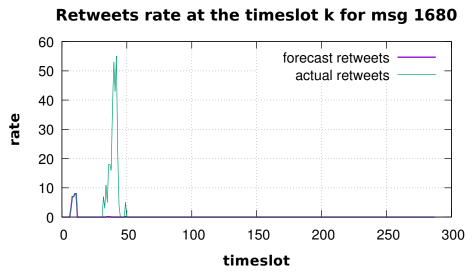


Figure D.291. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1680.

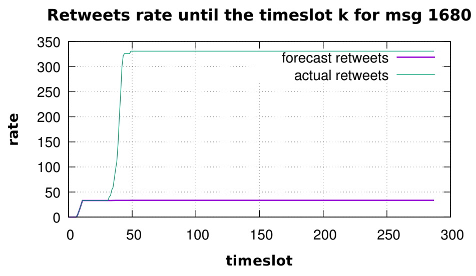


Figure D.292. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1680.

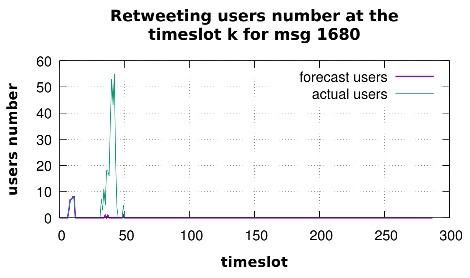


Figure D.293. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1680.

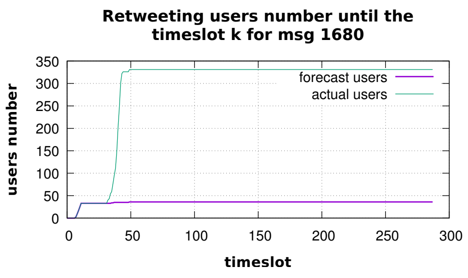


Figure D.294. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1680.

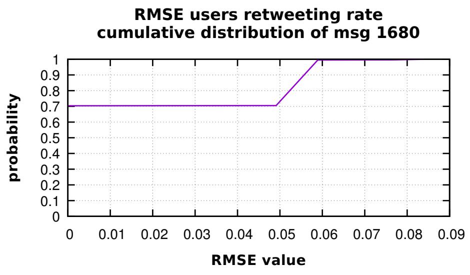


Figure D.295. Graphic showing the RMSE cumulative distribution, for message 1680.

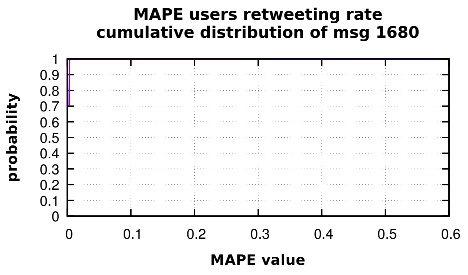


Figure D.296. Graphic showing the MAPE cumulative distribution, for message 1680.

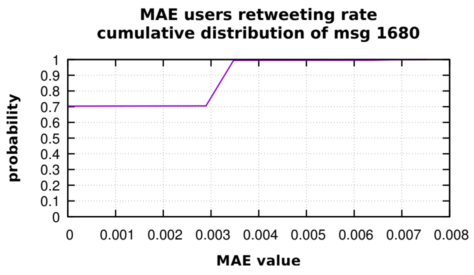


Figure D.297. Graphic showing the MAE cumulative distribution, for message 1680.

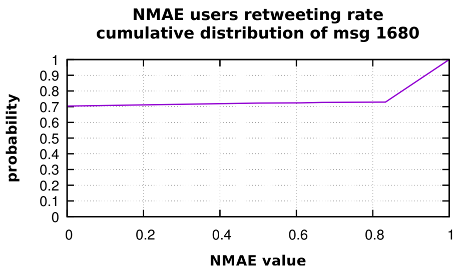


Figure D.298. Graphic showing the NMAE cumulative distribution, for message 1680.

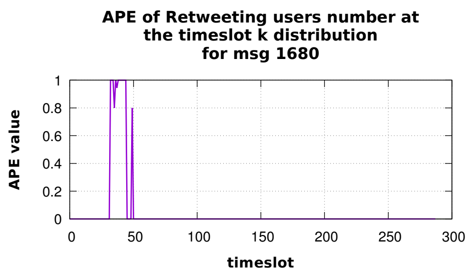


Figure D.299. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1680.

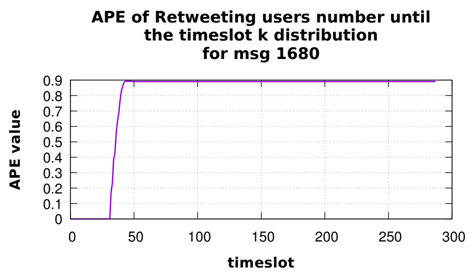


Figure D.300. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1680.

D.2 Model Validation

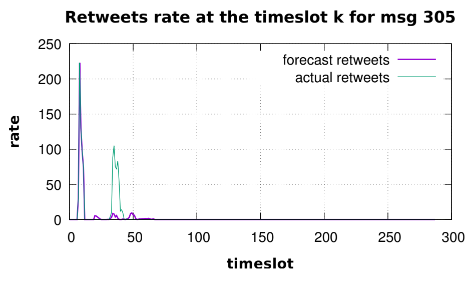


Figure D.301. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 305.

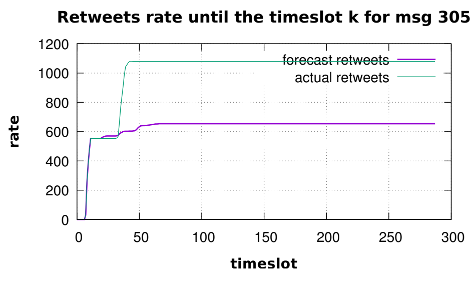


Figure D.302. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 305.

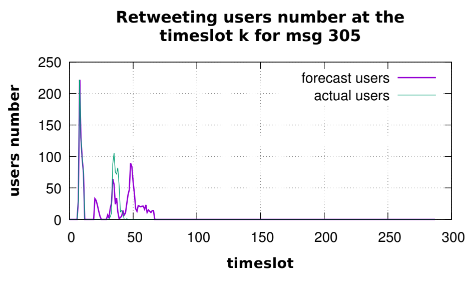


Figure D.303. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 305.

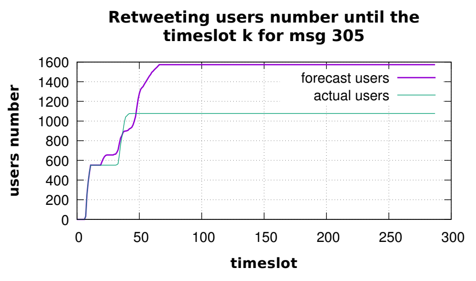


Figure D.304. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 305.

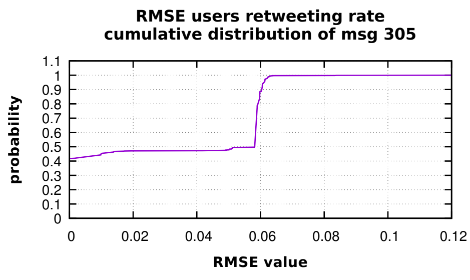


Figure D.305. Graphic showing the RMSE cumulative distribution, for message 305.

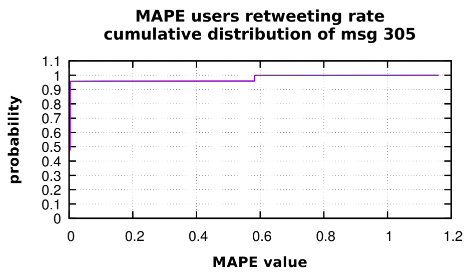


Figure D.306. Graphic showing the MAPE cumulative distribution, for message 305.

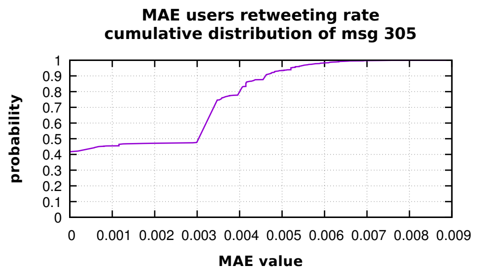


Figure D.307. Graphic showing the MAE cumulative distribution, for message 305.

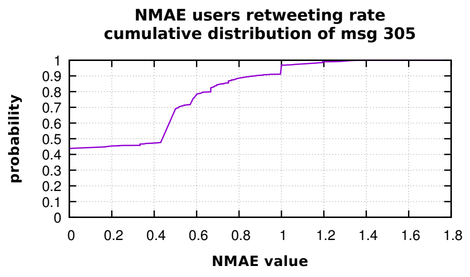


Figure D.308. Graphic showing the NMAE cumulative distribution, for message 305.

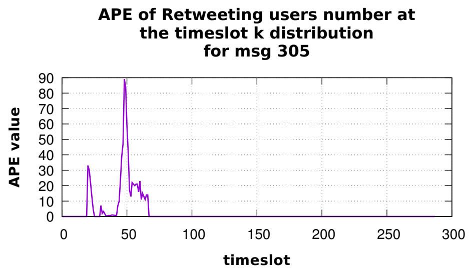


Figure D.309. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 305.

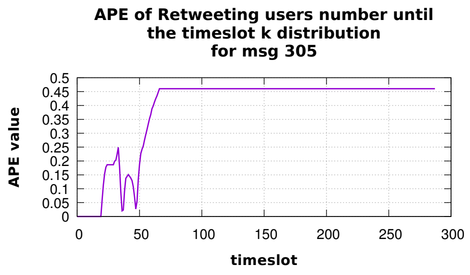


Figure D.310. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 305.

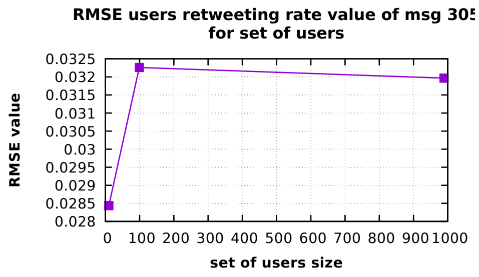


Figure D.311. Graphic showing the RMSE values distribution with respect to the set of users size, for message 305.

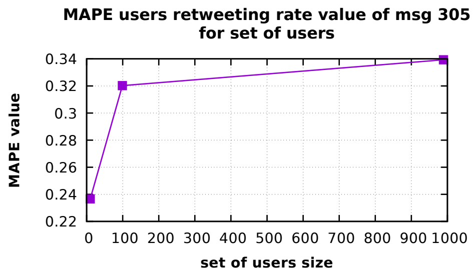


Figure D.312. Graphic showing the MAPE values distribution with respect to the set of users size, for message 305.

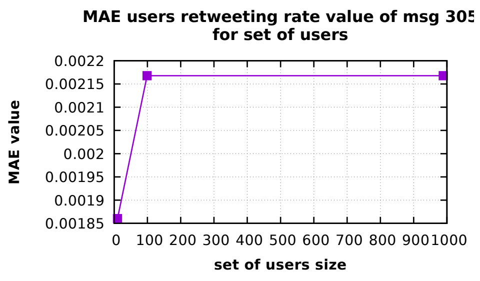


Figure D.313. Graphic showing the MAE values distribution with respect to the set of users size, for message 305.

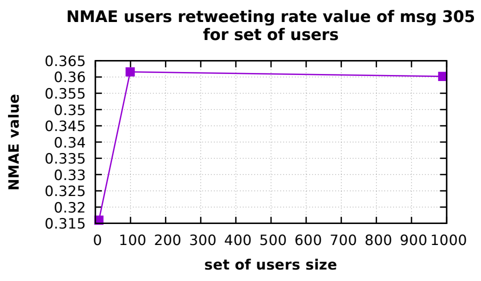


Figure D.314. Graphic showing the NMAE values distribution with respect to the set of users size, for message 305.

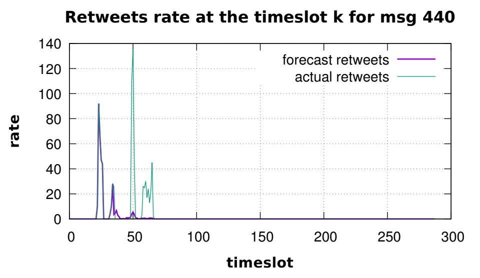


Figure D.315. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 440.

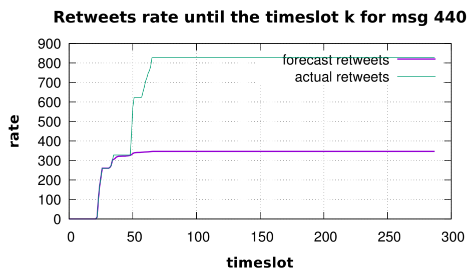


Figure D.316. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 440.

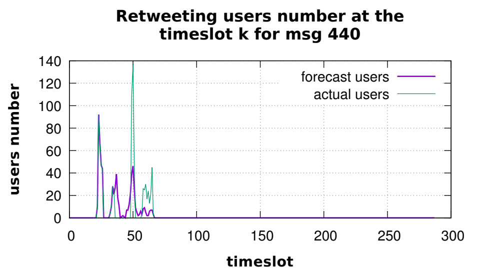


Figure D.317. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 440.

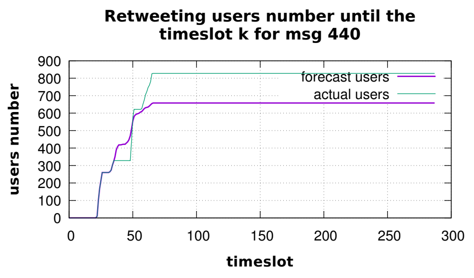


Figure D.318. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 440.

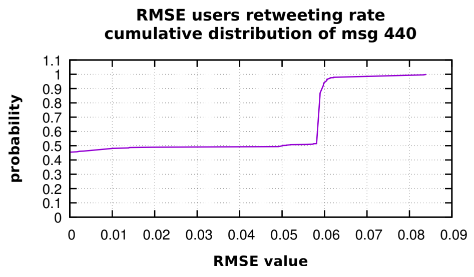


Figure D.319. Graphic showing the RMSE cumulative distribution, for message 440.

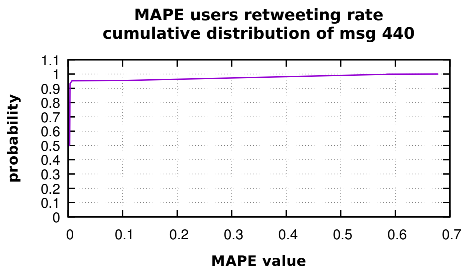


Figure D.320. Graphic showing the MAPE cumulative distribution, for message 440.

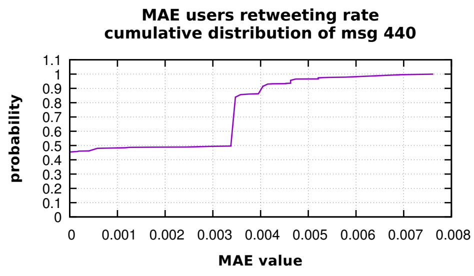


Figure D.321. Graphic showing the MAE cumulative distribution, for message 440.

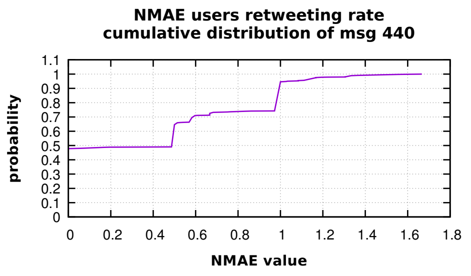


Figure D.322. Graphic showing the NMAE cumulative distribution, for message 440.

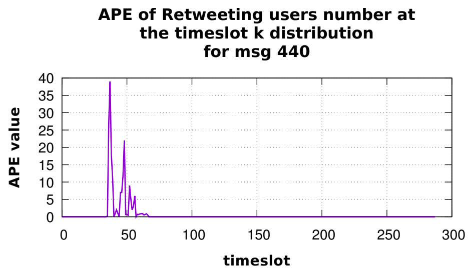


Figure D.323. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 440.

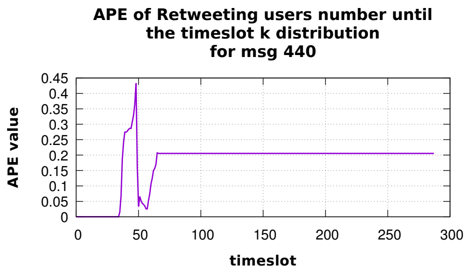


Figure D.324. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 440.

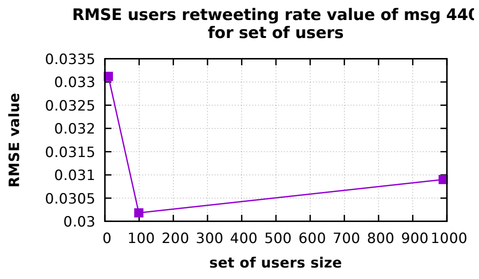


Figure D.325. Graphic showing the RMSE values distribution with respect to the set of users size, for message 440.

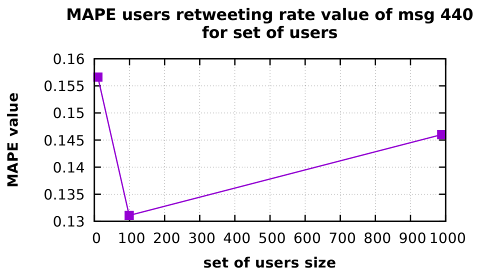


Figure D.326. Graphic showing the MAPE values distribution with respect to the set of users size, for message 440.

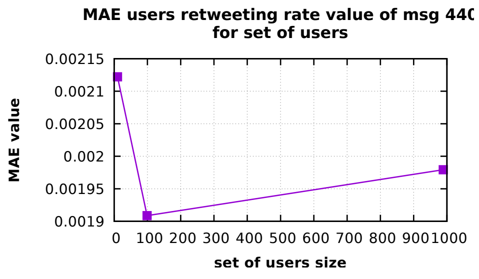


Figure D.327. Graphic showing the MAE values distribution with respect to the set of users size, for message 440.

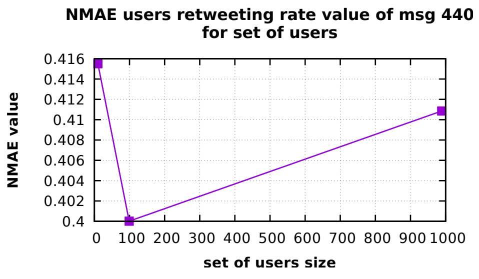


Figure D.328. Graphic showing the NMAE values distribution with respect to the set of users size, for message 440.

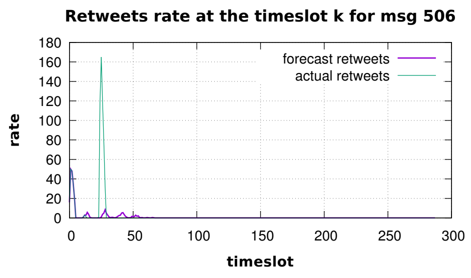


Figure D.329. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 506.

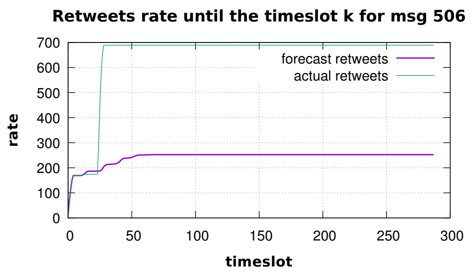


Figure D.330. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 506.

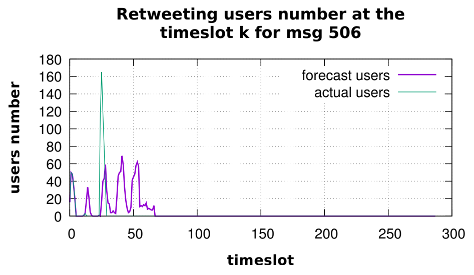


Figure D.331. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 506.

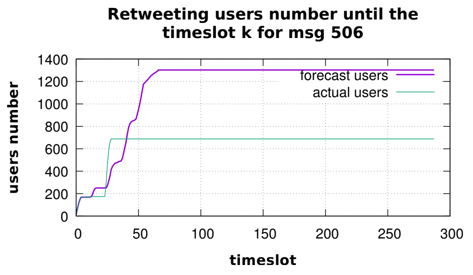


Figure D.332. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 506.

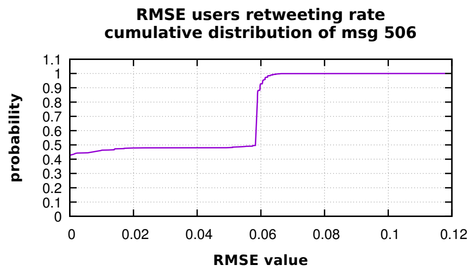


Figure D.333. Graphic showing the RMSE cumulative distribution, for message 506.

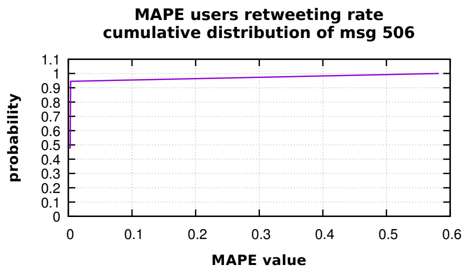


Figure D.334. Graphic showing the MAPE cumulative distribution, for message 506.

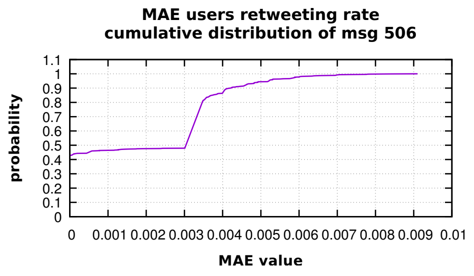


Figure D.335. Graphic showing the MAE cumulative distribution, for message 506.

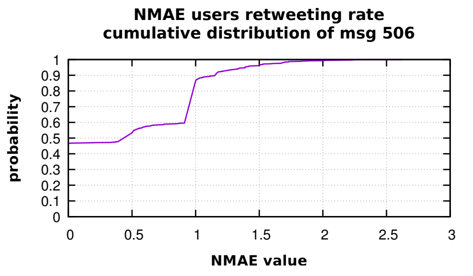


Figure D.336. Graphic showing the NMAE cumulative distribution, for message 506.

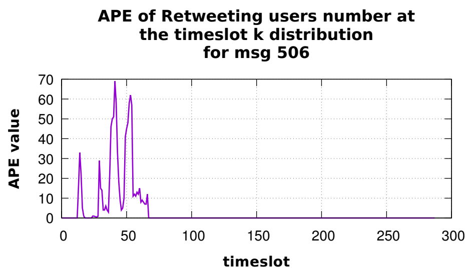


Figure D.337. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 506.

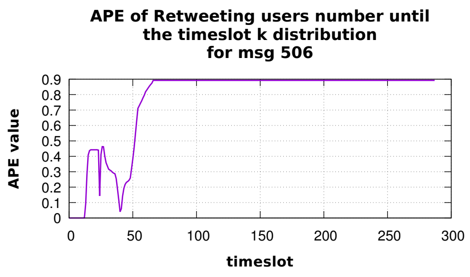


Figure D.338. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 506.

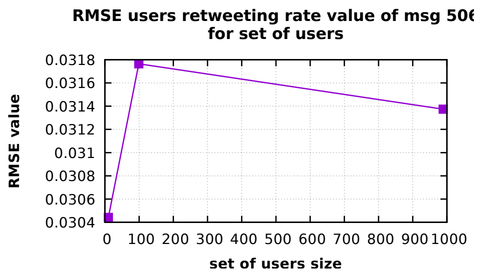


Figure D.339. Graphic showing the RMSE values distribution with respect to the set of users size, for message 506.

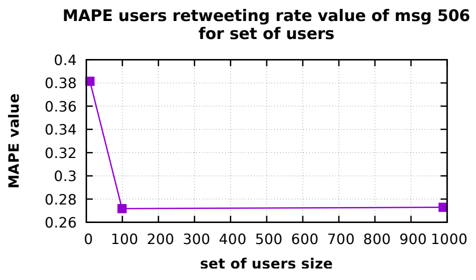


Figure D.340. Graphic showing the MAPE values distribution with respect to the set of users size, for message 506.

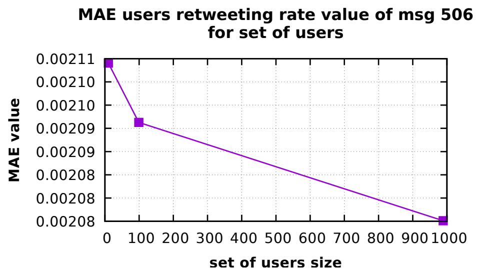


Figure D.341. Graphic showing the MAE values distribution with respect to the set of users size, for message 506.

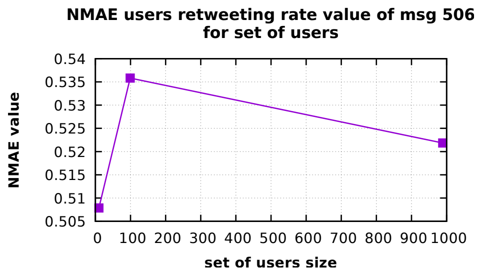


Figure D.342. Graphic showing the NMAE values distribution with respect to the set of users size, for message 506.

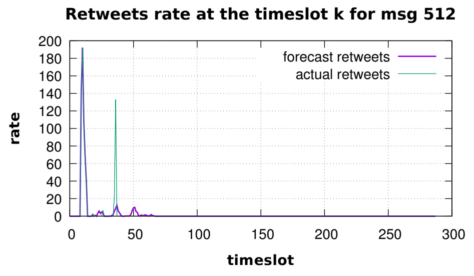


Figure D.343. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 512.

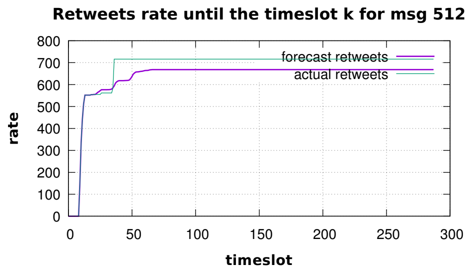


Figure D.344. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 512.

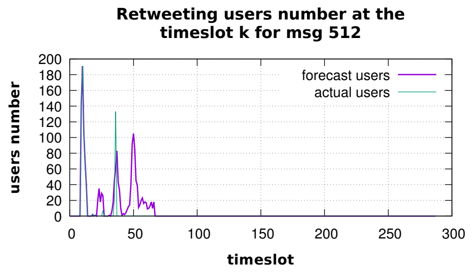


Figure D.345. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 512.

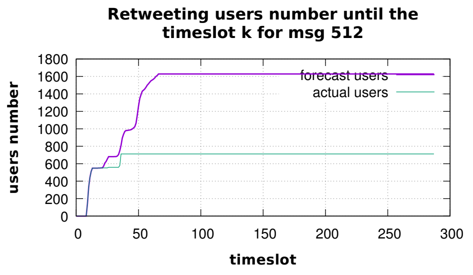


Figure D.346. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 512.

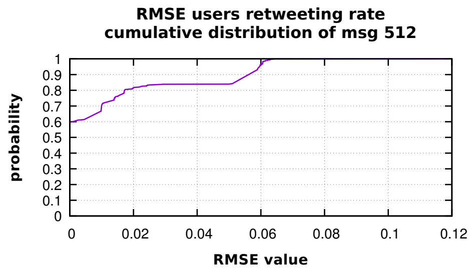


Figure D.347. Graphic showing the RMSE cumulative distribution, for message 512.

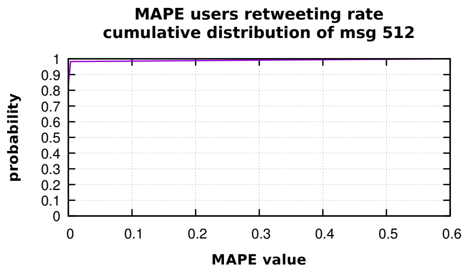


Figure D.348. Graphic showing the MAPE cumulative distribution, for message 512.

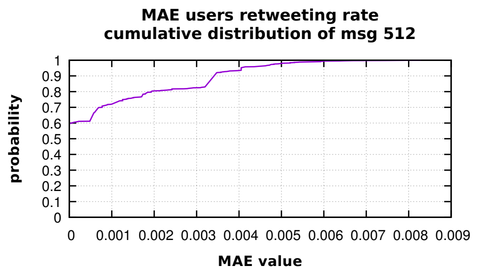


Figure D.349. Graphic showing the MAE cumulative distribution, for message 512.

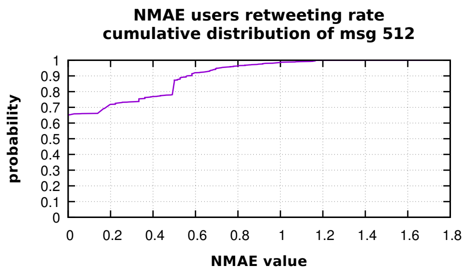


Figure D.350. Graphic showing the NMAE cumulative distribution, for message 512.

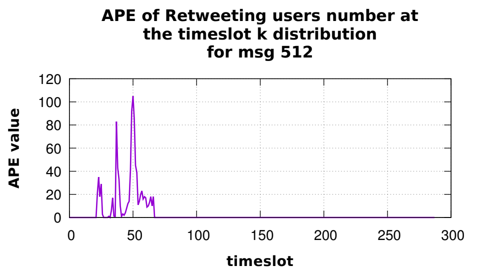


Figure D.351. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 512.

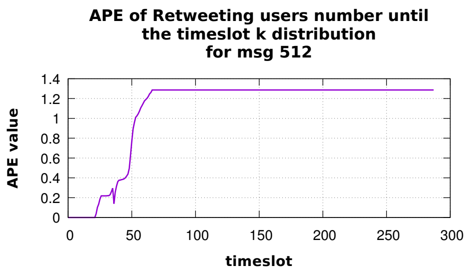


Figure D.352. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 512.

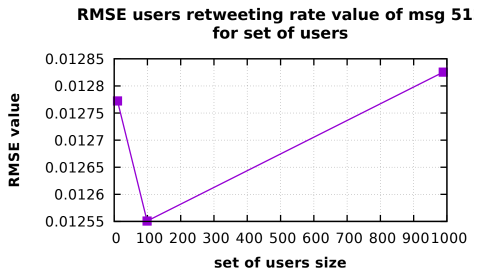


Figure D.353. Graphic showing the RMSE values distribution with respect to the set of users size, for message 512.

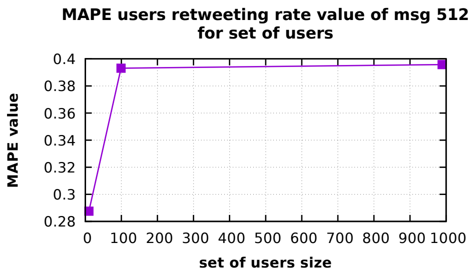


Figure D.354. Graphic showing the MAPE values distribution with respect to the set of users size, for message 512.

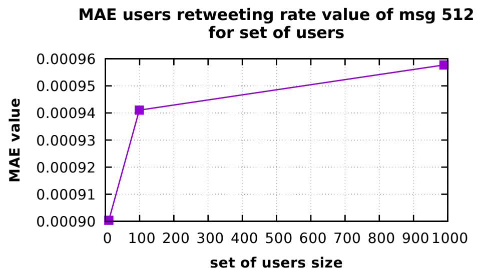


Figure D.355. Graphic showing the MAE values distribution with respect to the set of users size, for message 512.

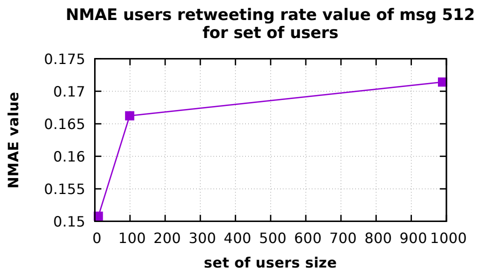


Figure D.356. Graphic showing the NMAE values distribution with respect to the set of users size, for message 512.

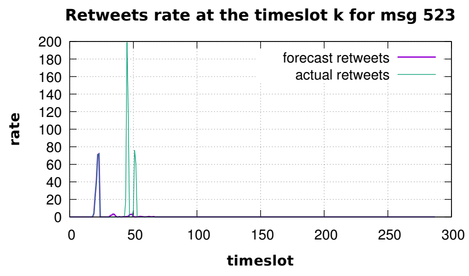


Figure D.357. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 523.

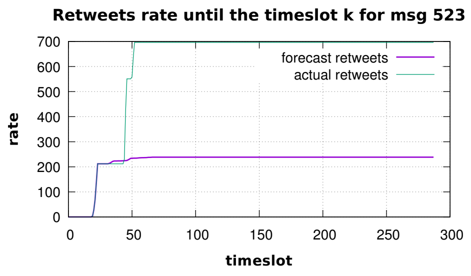


Figure D.358. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 523.

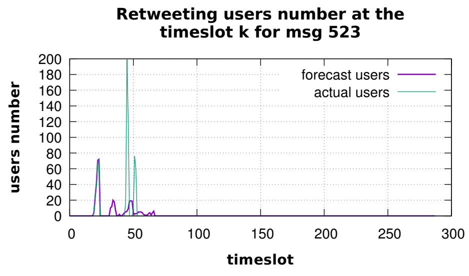


Figure D.359. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 523.

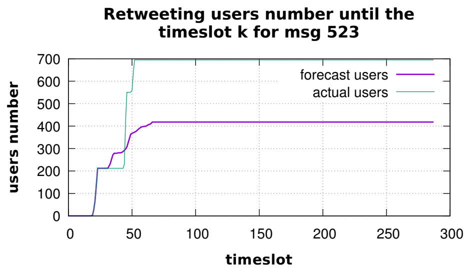


Figure D.360. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 523.

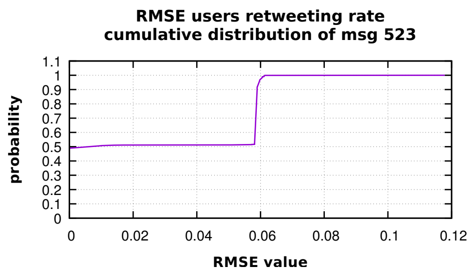


Figure D.361. Graphic showing the RMSE cumulative distribution, for message 523.

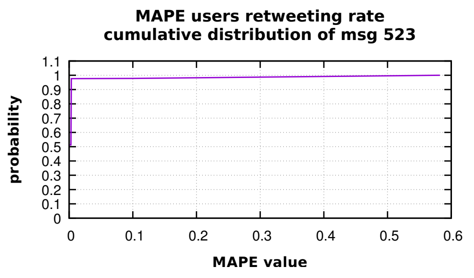


Figure D.362. Graphic showing the MAPE cumulative distribution, for message 523.

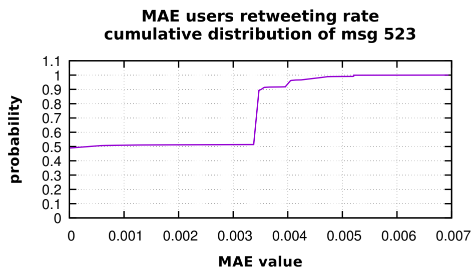


Figure D.363. Graphic showing the MAE cumulative distribution, for message 523.

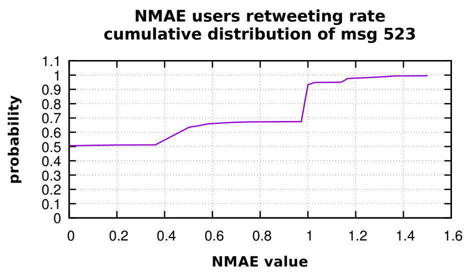


Figure D.364. Graphic showing the NMAE cumulative distribution, for message 523.

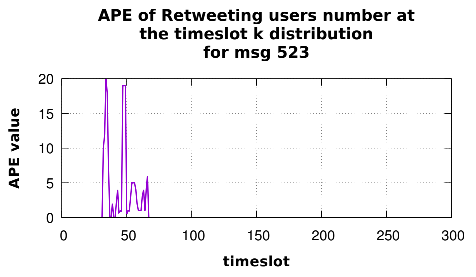


Figure D.365. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 523.

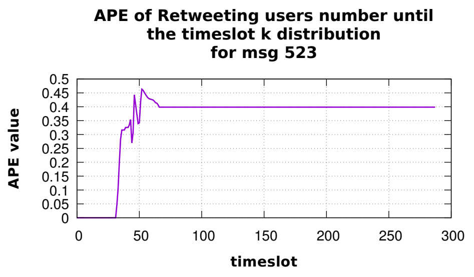


Figure D.366. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 523.

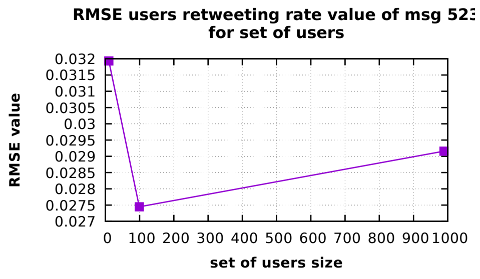


Figure D.367. Graphic showing the RMSE values distribution with respect to the set of users size, for message 523.

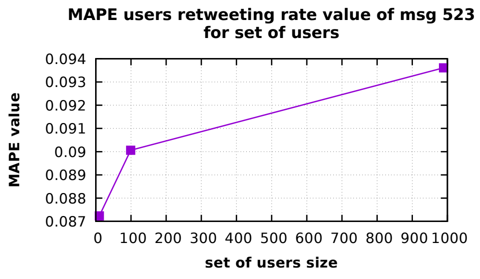


Figure D.368. Graphic showing the MAPE values distribution with respect to the set of users size, for message 523.

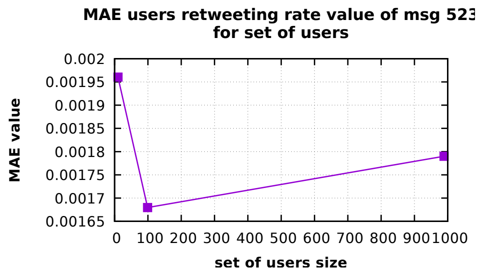


Figure D.369. Graphic showing the MAE values distribution with respect to the set of users size, for message 523.

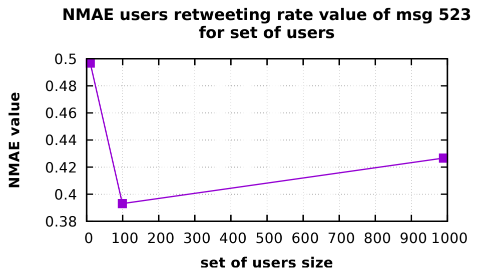


Figure D.370. Graphic showing the NMAE values distribution with respect to the set of users size, for message 523.

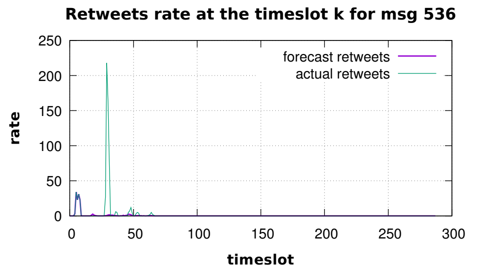


Figure D.371. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 536.

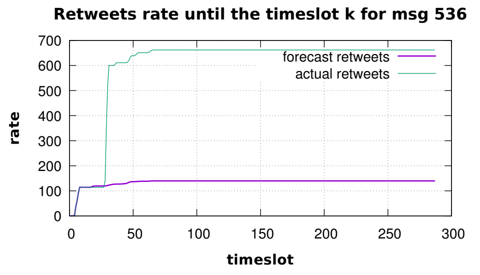


Figure D.372. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 536.

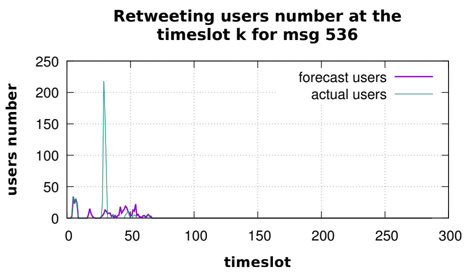


Figure D.373. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 536.

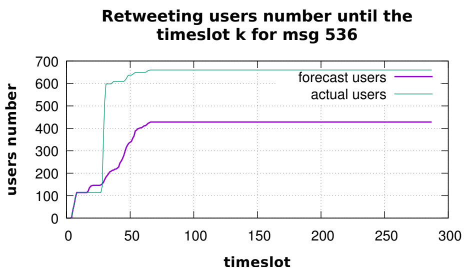


Figure D.374. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 536.

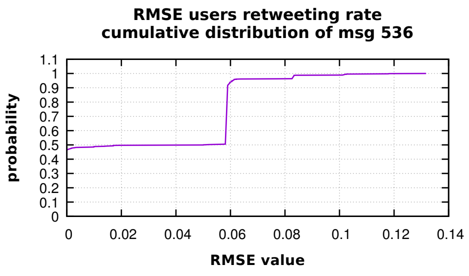


Figure D.375. Graphic showing the RMSE cumulative distribution, for message 536.

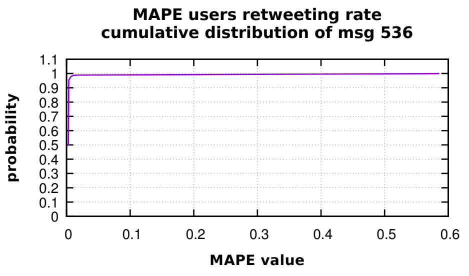


Figure D.376. Graphic showing the MAPE cumulative distribution, for message 536.

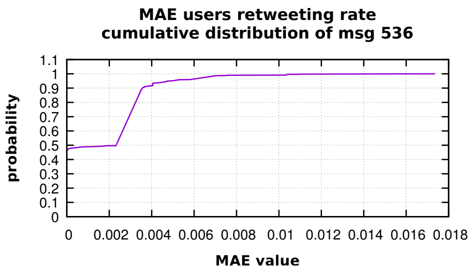


Figure D.377. Graphic showing the MAE cumulative distribution, for message 536.

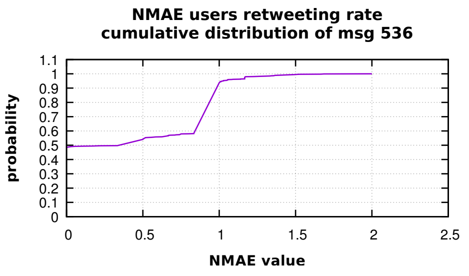


Figure D.378. Graphic showing the NMAE cumulative distribution, for message 536.

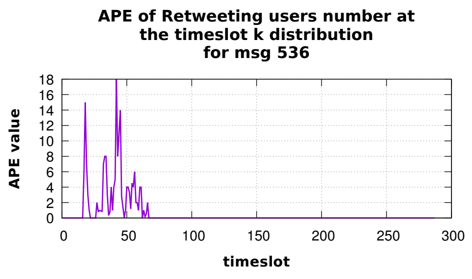


Figure D.379. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 536.

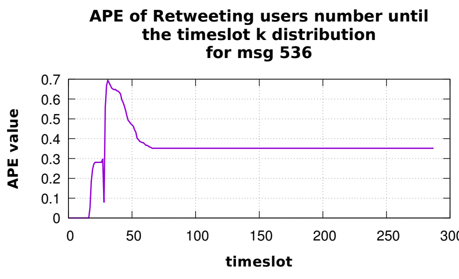


Figure D.380. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 536.

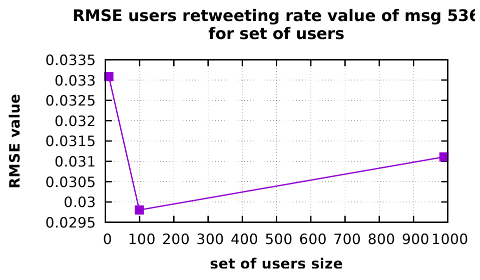


Figure D.381. Graphic showing the RMSE values distribution with respect to the set of users size, for message 536.

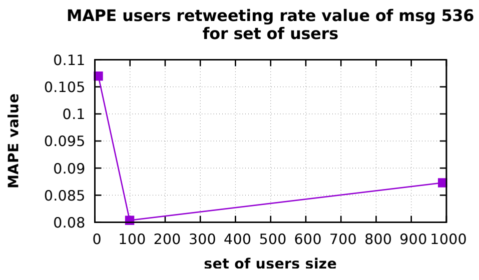


Figure D.382. Graphic showing the MAPE values distribution with respect to the set of users size, for message 536.

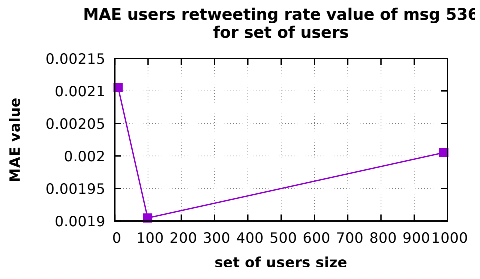


Figure D.383. Graphic showing the MAE values distribution with respect to the set of users size, for message 536.

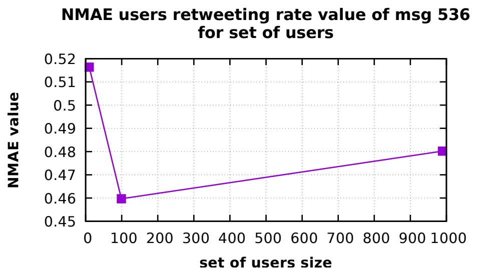


Figure D.384. Graphic showing the NMAE values distribution with respect to the set of users size, for message 536.

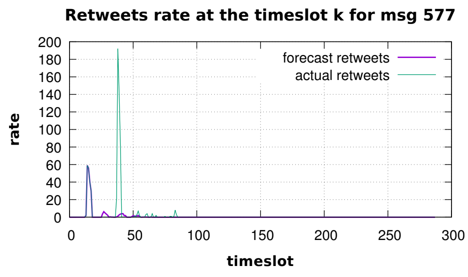


Figure D.385. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 577.

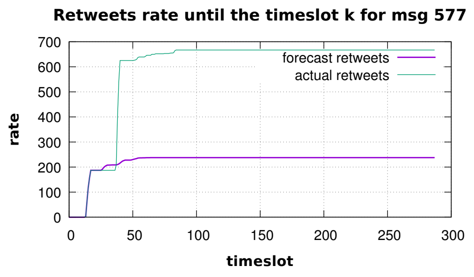


Figure D.386. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 577.

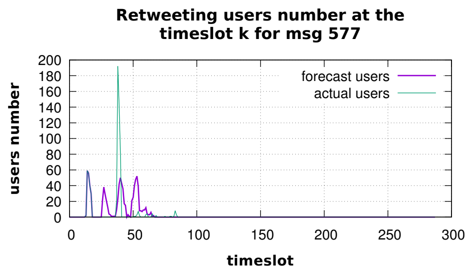


Figure D.387. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 577.

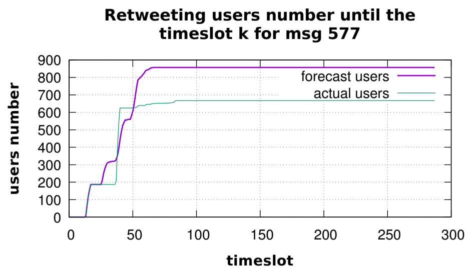


Figure D.388. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 577.

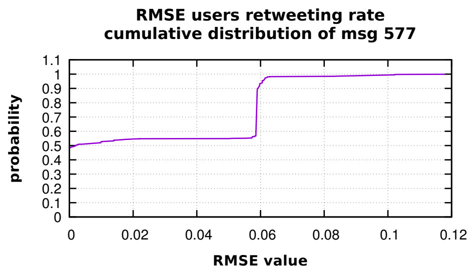


Figure D.389. Graphic showing the RMSE cumulative distribution, for message 577.

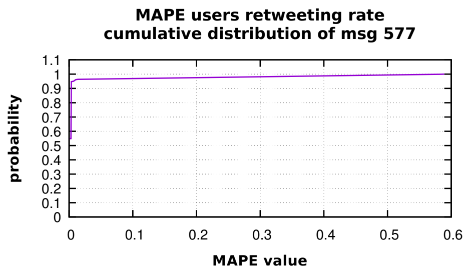


Figure D.390. Graphic showing the MAPE cumulative distribution, for message 577.

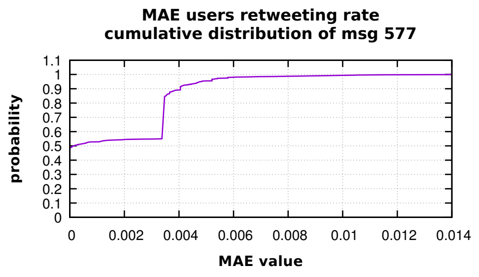


Figure D.391. Graphic showing the MAE cumulative distribution, for message 577.

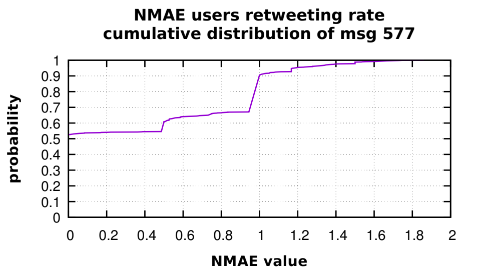


Figure D.392. Graphic showing the NMAE cumulative distribution, for message 577.

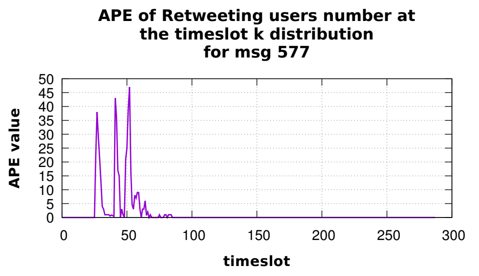


Figure D.393. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 577.

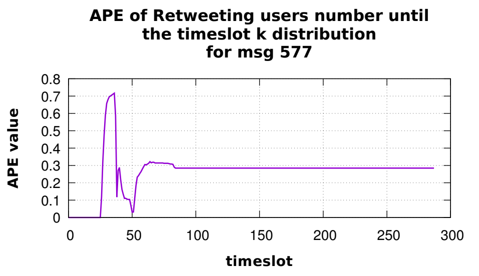


Figure D.394. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 577.

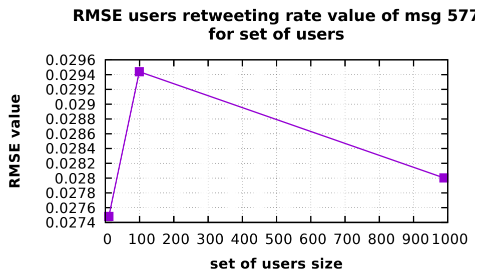


Figure D.395. Graphic showing the RMSE values distribution with respect to the set of users size, for message 577.

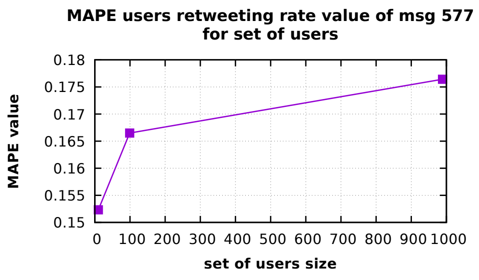


Figure D.396. Graphic showing the MAPE values distribution with respect to the set of users size, for message 577.

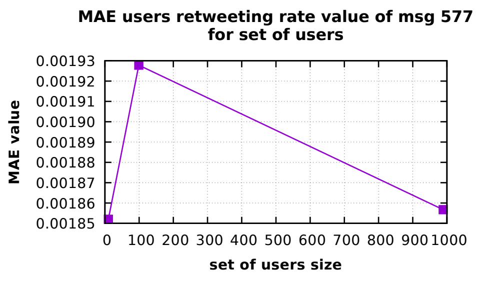


Figure D.397. Graphic showing the MAE values distribution with respect to the set of users size, for message 577.

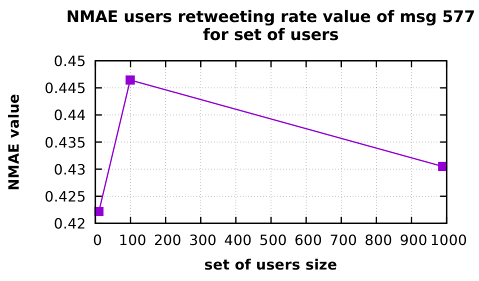


Figure D.398. Graphic showing the NMAE values distribution with respect to the set of users size, for message 577.

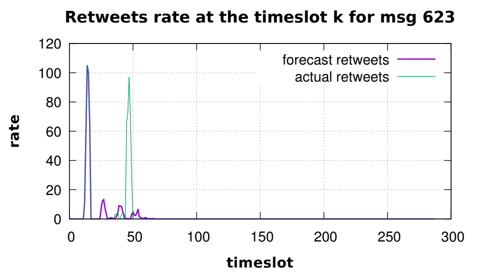


Figure D.399. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 623.

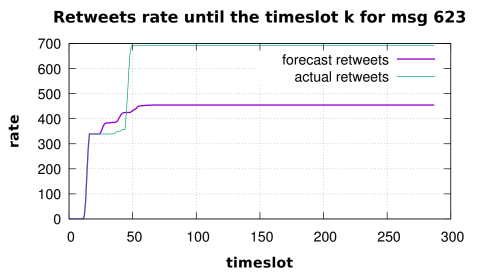


Figure D.400. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 623.

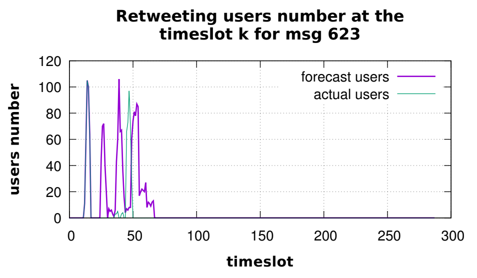


Figure D.401. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 623.

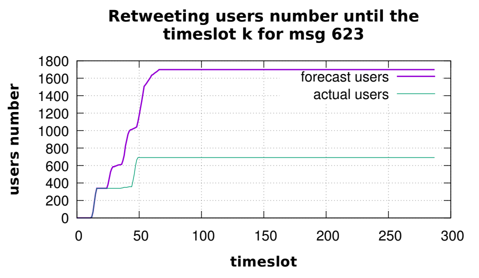


Figure D.402. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 623.

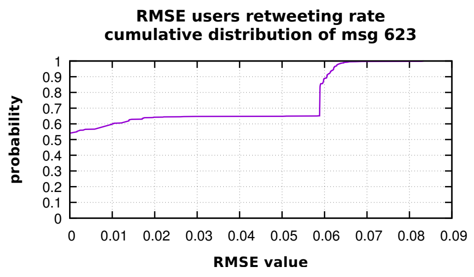


Figure D.403. Graphic showing the RMSE cumulative distribution, for message 623.

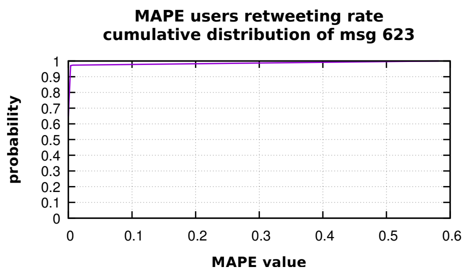


Figure D.404. Graphic showing the MAPE cumulative distribution, for message 623.

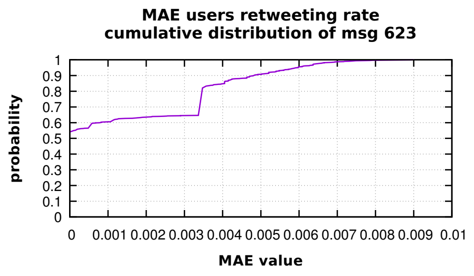


Figure D.405. Graphic showing the MAE cumulative distribution, for message 623.

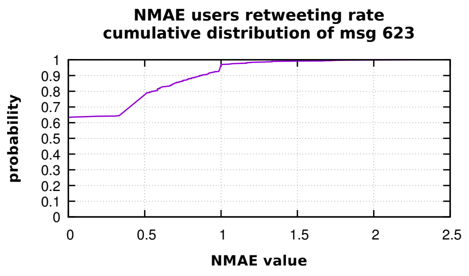


Figure D.406. Graphic showing the NMAE cumulative distribution, for message 623.

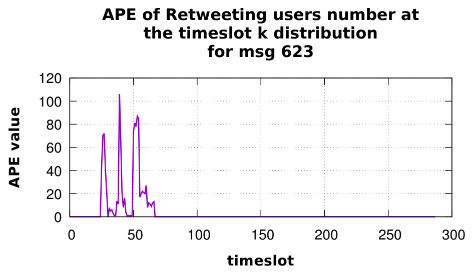


Figure D.407. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 623.

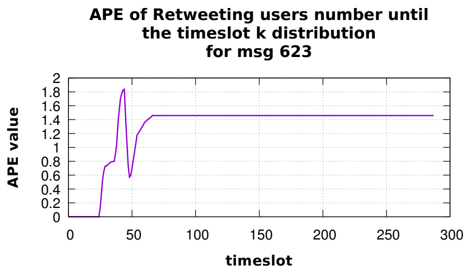


Figure D.408. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 623.

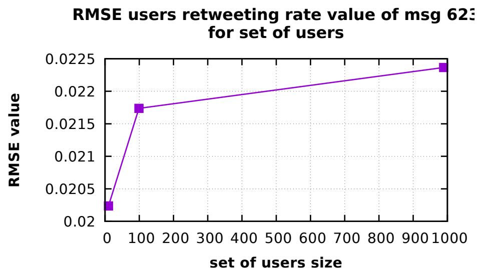


Figure D.409. Graphic showing the RMSE values distribution with respect to the set of users size, for message 623.

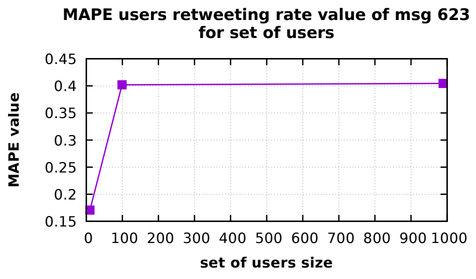


Figure D.410. Graphic showing the MAPE values distribution with respect to the set of users size, for message 623.

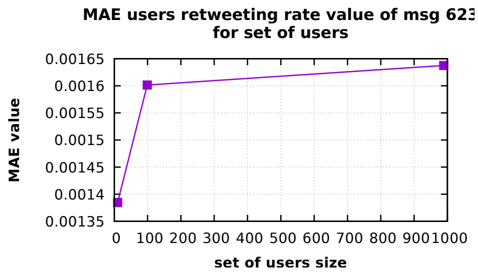


Figure D.411. Graphic showing the MAE values distribution with respect to the set of users size, for message 623.

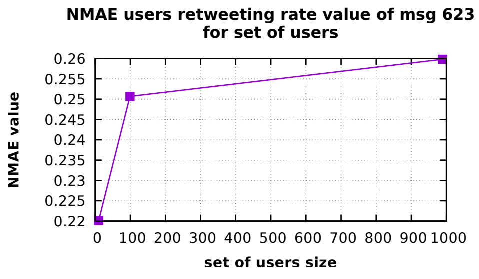


Figure D.412. Graphic showing the NMAE values distribution with respect to the set of users size, for message 623.

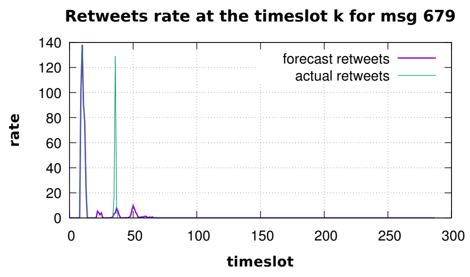


Figure D.413. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 679.

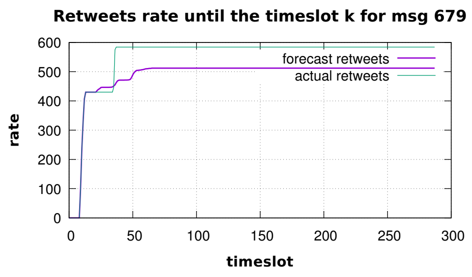


Figure D.414. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 679.

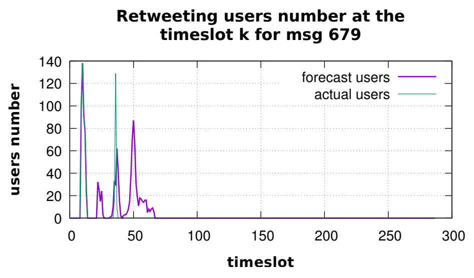


Figure D.415. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 679.

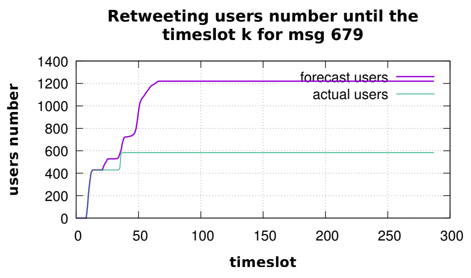


Figure D.416. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 679.

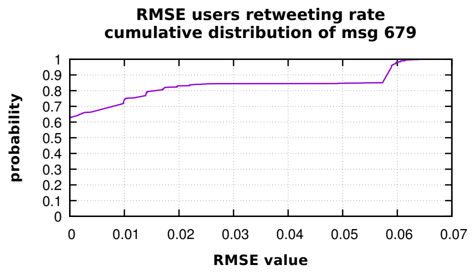


Figure D.417. Graphic showing the RMSE cumulative distribution, for message 679.

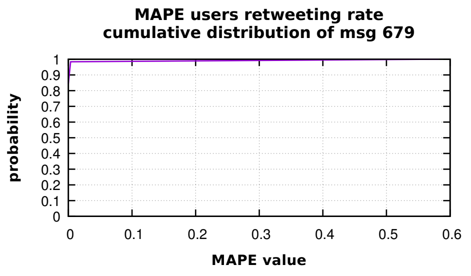


Figure D.418. Graphic showing the MAPE cumulative distribution, for message 679.

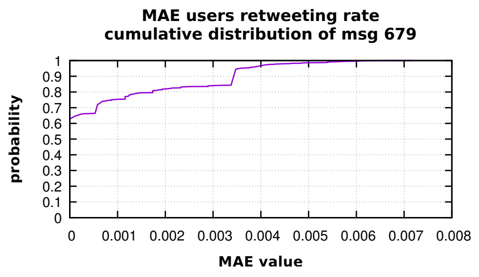


Figure D.419. Graphic showing the MAE cumulative distribution, for message 679.

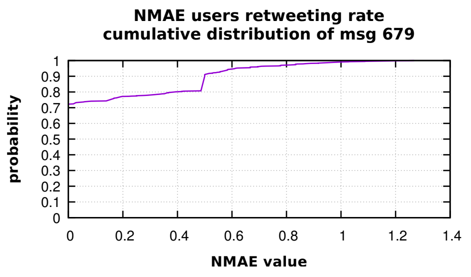


Figure D.420. Graphic showing the NMAE cumulative distribution, for message 679.

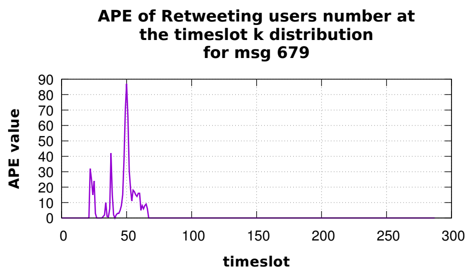


Figure D.421. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 679.

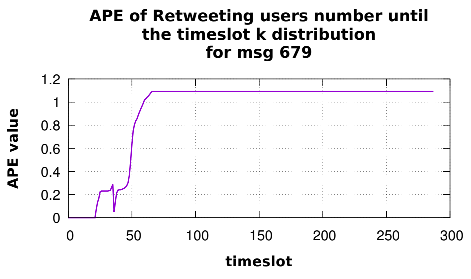


Figure D.422. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 679.

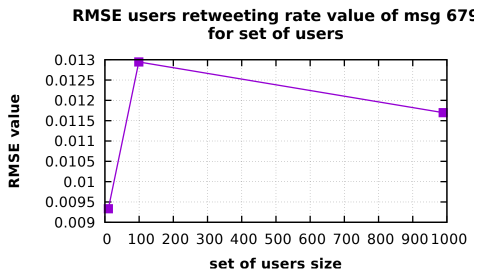


Figure D.423. Graphic showing the RMSE values distribution with respect to the set of users size, for message 679.

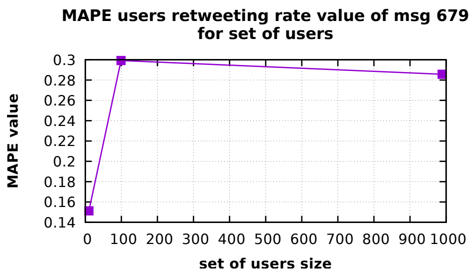


Figure D.424. Graphic showing the MAPE values distribution with respect to the set of users size, for message 679.

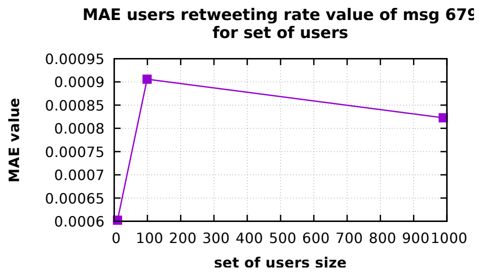


Figure D.425. Graphic showing the MAE values distribution with respect to the set of users size, for message 679.

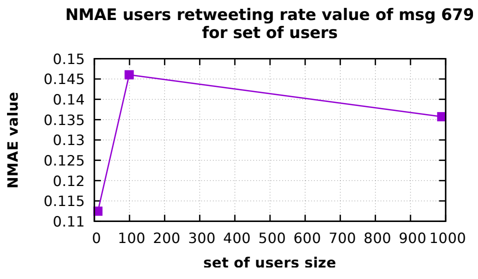


Figure D.426. Graphic showing the NMAE values distribution with respect to the set of users size, for message 679.

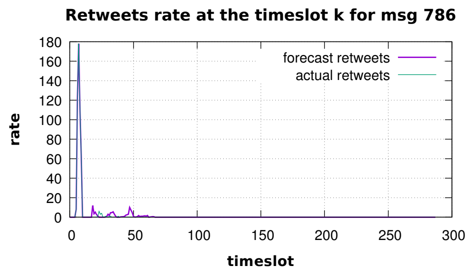


Figure D.427. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 786.

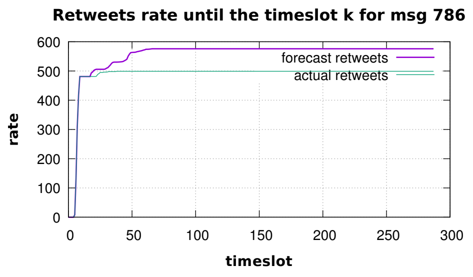


Figure D.428. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 786.

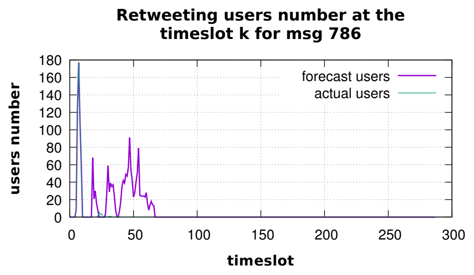


Figure D.429. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 786.

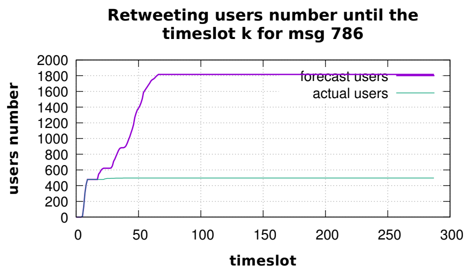


Figure D.430. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 786.

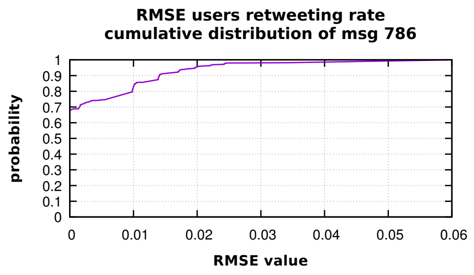


Figure D.431. Graphic showing the RMSE cumulative distribution, for message 786.

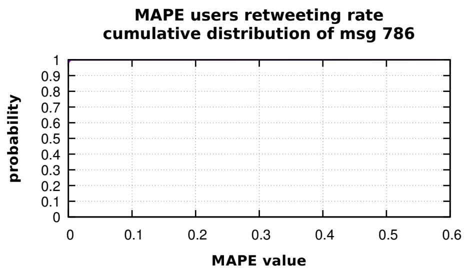


Figure D.432. Graphic showing the MAPE cumulative distribution, for message 786.

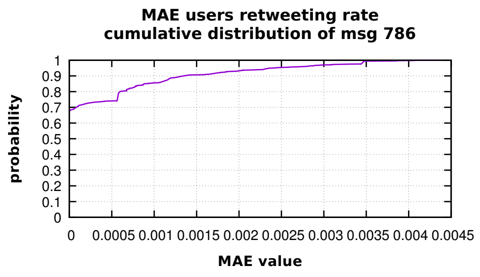


Figure D.433. Graphic showing the MAE cumulative distribution, for message 786.

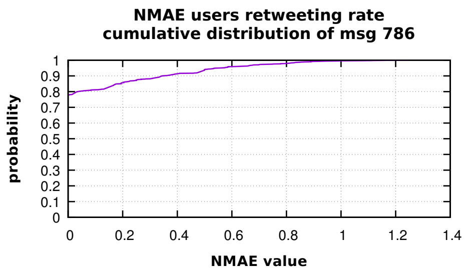


Figure D.434. Graphic showing the NMAE cumulative distribution, for message 786.

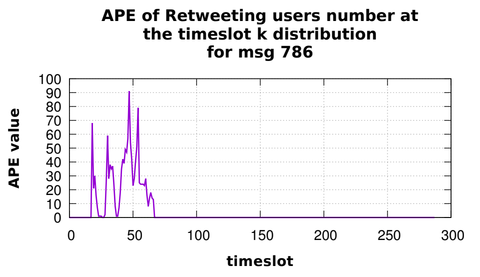


Figure D.435. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 786.

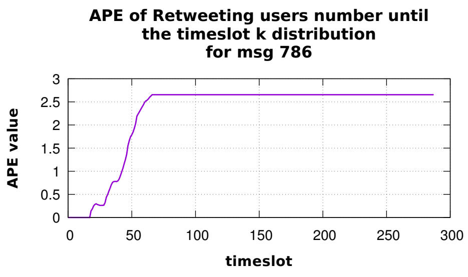


Figure D.436. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 786.

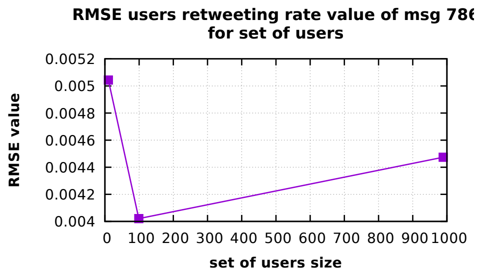


Figure D.437. Graphic showing the RMSE values distribution with respect to the set of users size, for message 786.

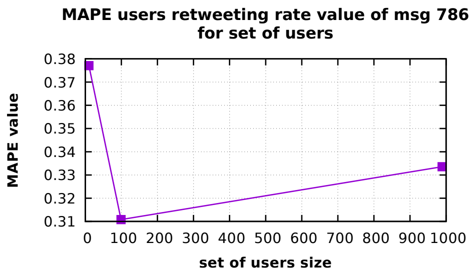


Figure D.438. Graphic showing the MAPE values distribution with respect to the set of users size, for message 786.

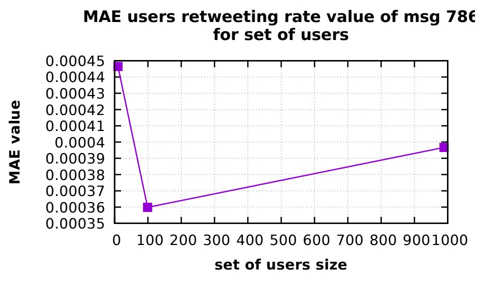


Figure D.439. Graphic showing the MAE values distribution with respect to the set of users size, for message 786.

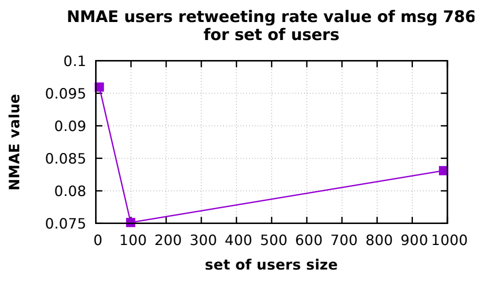


Figure D.440. Graphic showing the NMAE values distribution with respect to the set of users size, for message 786.

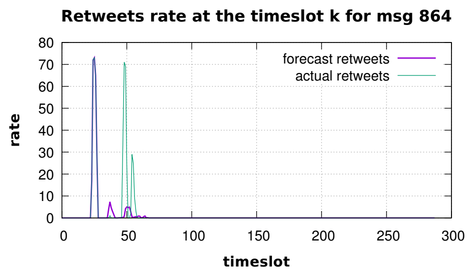


Figure D.441. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 864.

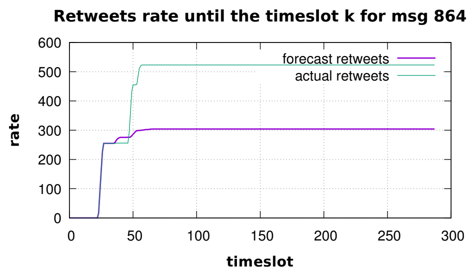


Figure D.442. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 864.

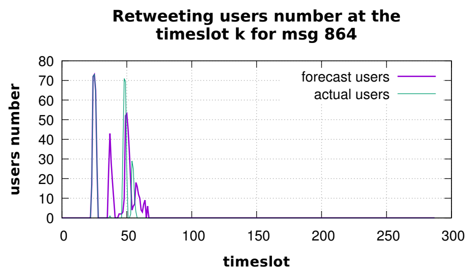


Figure D.443. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 864.

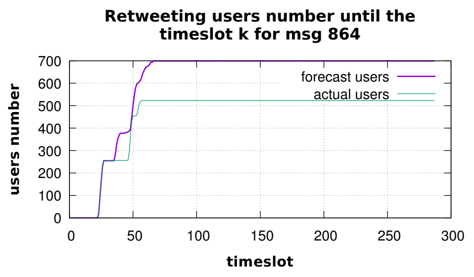


Figure D.444. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 864.

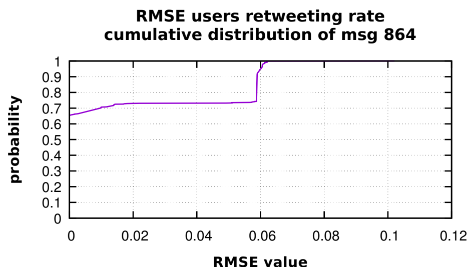


Figure D.445. Graphic showing the RMSE cumulative distribution, for message 864.

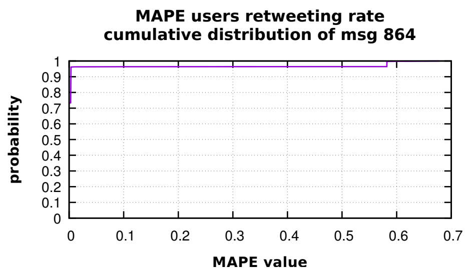


Figure D.446. Graphic showing the MAPE cumulative distribution, for message 864.

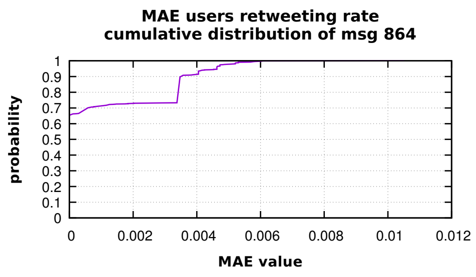


Figure D.447. Graphic showing the MAE cumulative distribution, for message 864.

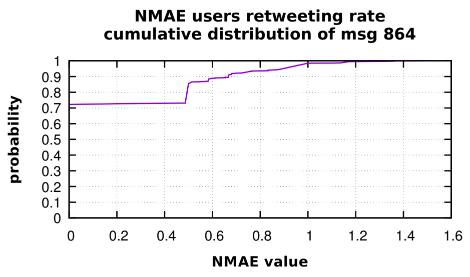


Figure D.448. Graphic showing the NMAE cumulative distribution, for message 864.

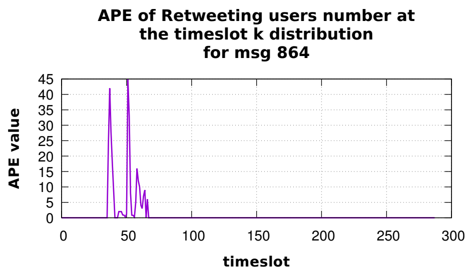


Figure D.449. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 864.

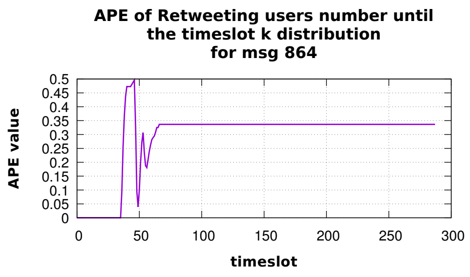


Figure D.450. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 864.

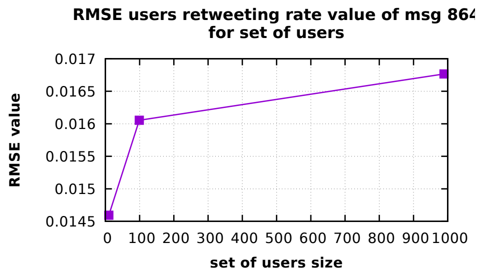


Figure D.451. Graphic showing the RMSE values distribution with respect to the set of users size, for message 864.

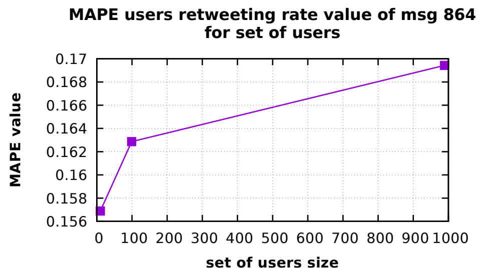


Figure D.452. Graphic showing the MAPE values distribution with respect to the set of users size, for message 864.

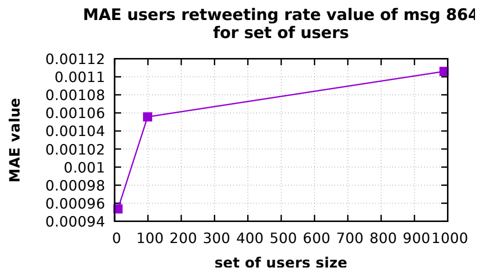


Figure D.453. Graphic showing the MAE values distribution with respect to the set of users size, for message 864.

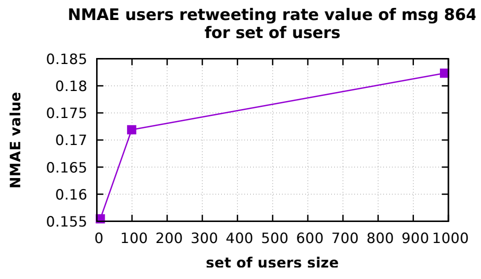


Figure D.454. Graphic showing the NMAE values distribution with respect to the set of users size, for message 864.

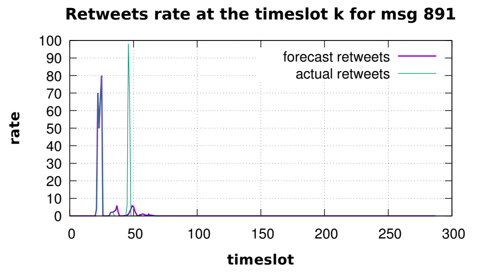


Figure D.455. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 891.

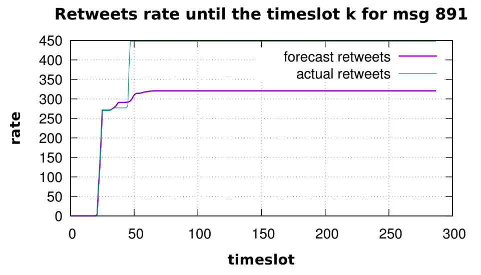


Figure D.456. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 891.

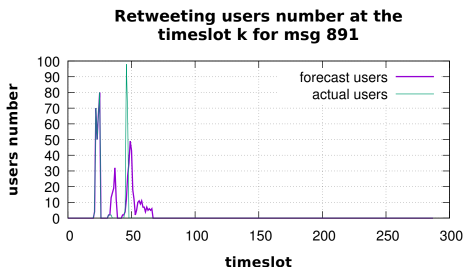


Figure D.457. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 891.

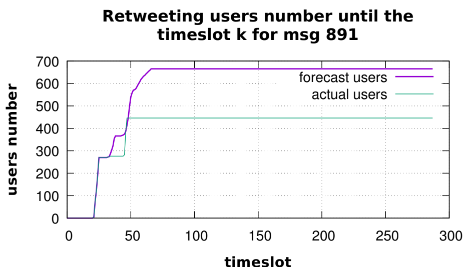


Figure D.458. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 891.

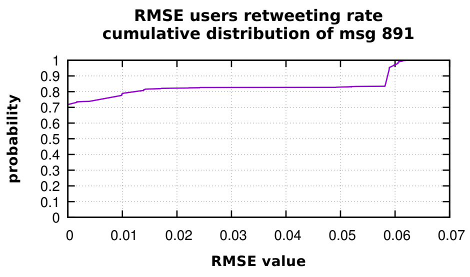


Figure D.459. Graphic showing the RMSE cumulative distribution, for message 891.

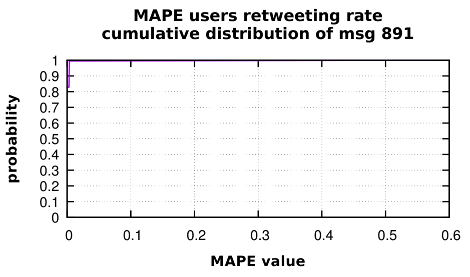


Figure D.460. Graphic showing the MAPE cumulative distribution, for message 891.

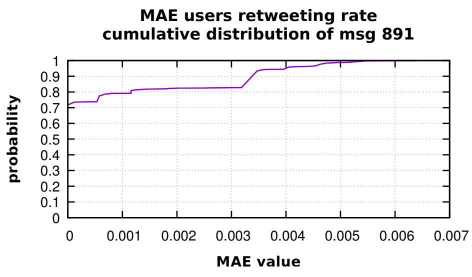


Figure D.461. Graphic showing the MAE cumulative distribution, for message 891.

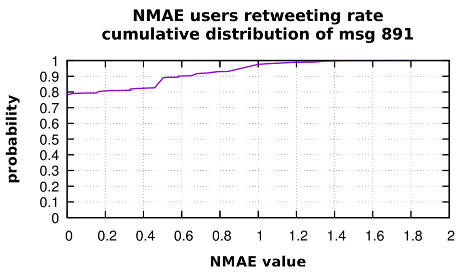


Figure D.462. Graphic showing the NMAE cumulative distribution, for message 891.

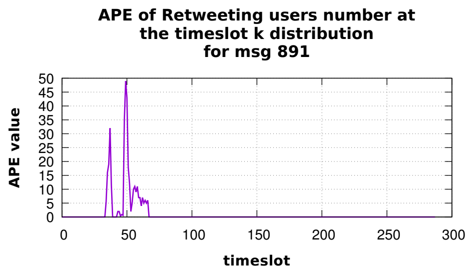


Figure D.463. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 891.

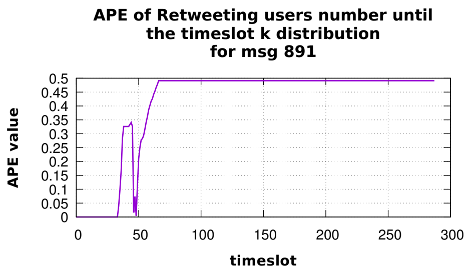


Figure D.464. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 891.

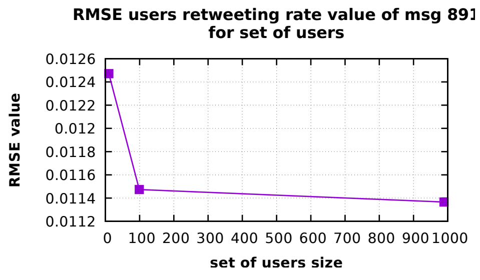


Figure D.465. Graphic showing the RMSE values distribution with respect to the set of users size, for message 891.

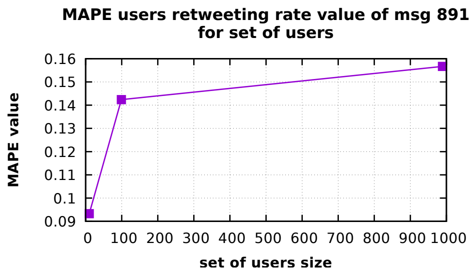


Figure D.466. Graphic showing the MAPE values distribution with respect to the set of users size, for message 891.

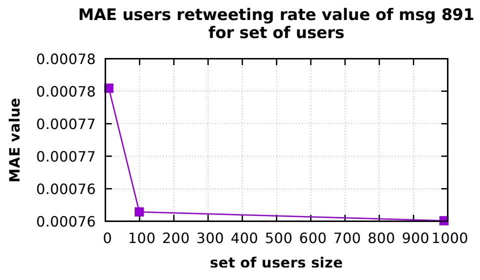


Figure D.467. Graphic showing the MAE values distribution with respect to the set of users size, for message 891.

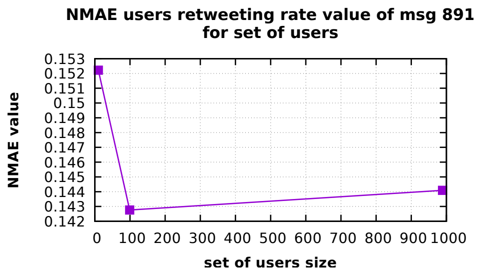


Figure D.468. Graphic showing the NMAE values distribution with respect to the set of users size, for message 891.

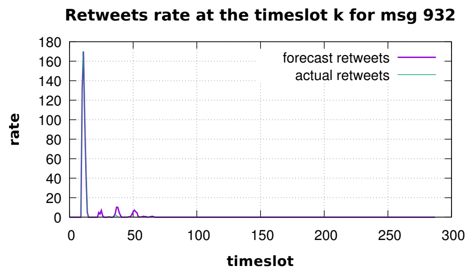


Figure D.469. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 932.

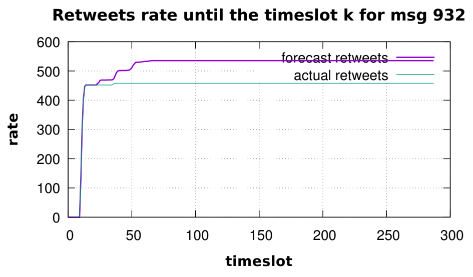


Figure D.470. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 932.

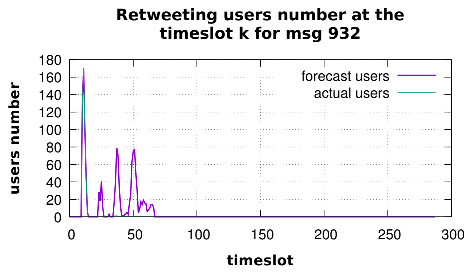


Figure D.471. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 932.

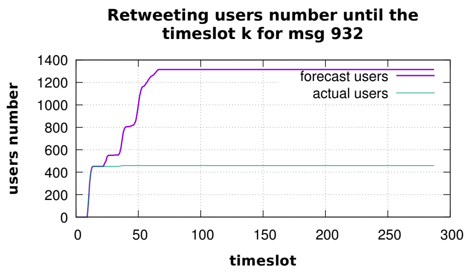


Figure D.472. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 932.

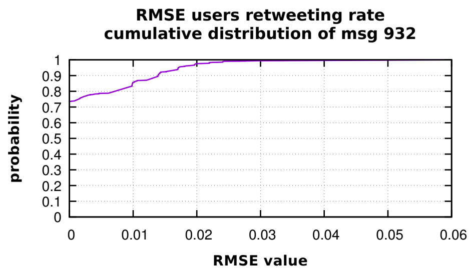


Figure D.473. Graphic showing the RMSE cumulative distribution, for message 932.

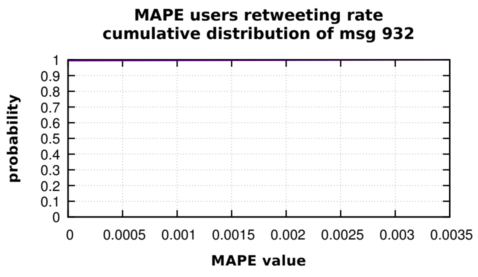


Figure D.474. Graphic showing the MAPE cumulative distribution, for message 932.

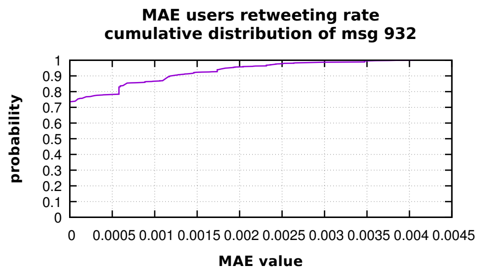


Figure D.475. Graphic showing the MAE cumulative distribution, for message 932.

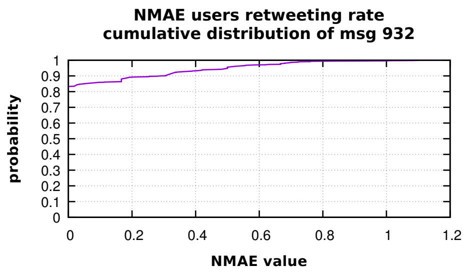


Figure D.476. Graphic showing the NMAE cumulative distribution, for message 932.

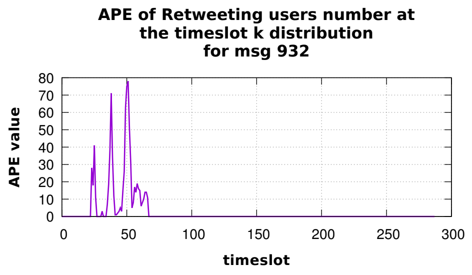


Figure D.477. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 932.

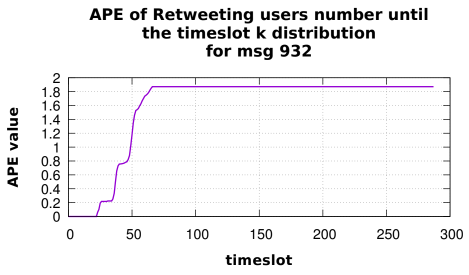


Figure D.478. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 932.

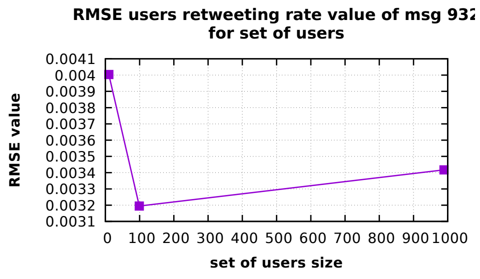


Figure D.479. Graphic showing the RMSE values distribution with respect to the set of users size, for message 932.

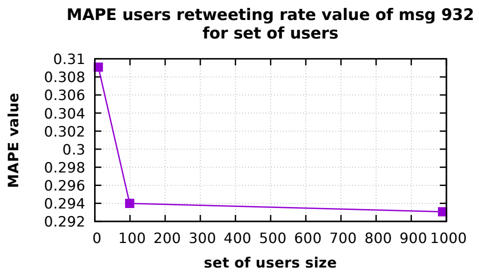


Figure D.480. Graphic showing the MAPE values distribution with respect to the set of users size, for message 932.

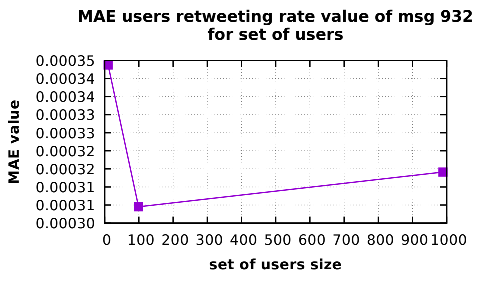


Figure D.481. Graphic showing the MAE values distribution with respect to the set of users size, for message 932.

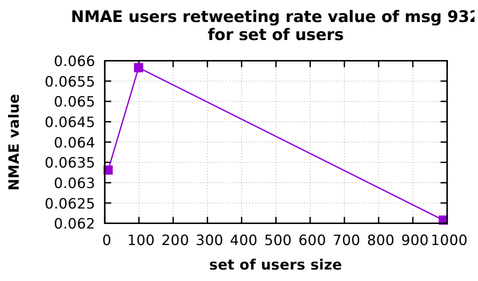


Figure D.482. Graphic showing the NMAE values distribution with respect to the set of users size, for message 932.

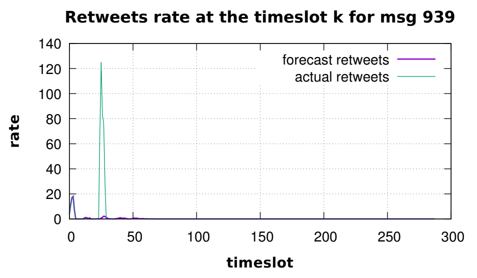


Figure D.483. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 939.

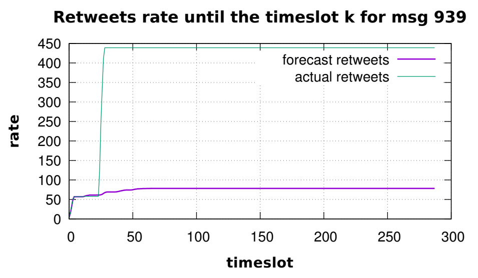


Figure D.484. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 939.

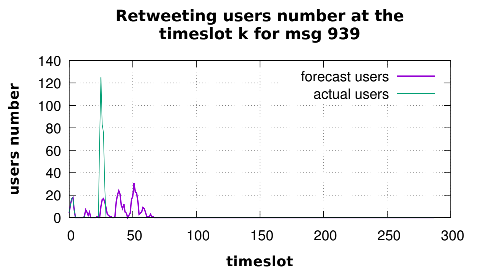


Figure D.485. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 939.

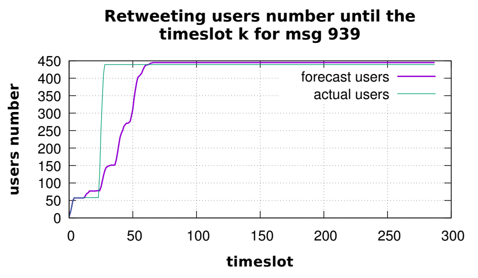


Figure D.486. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 939.

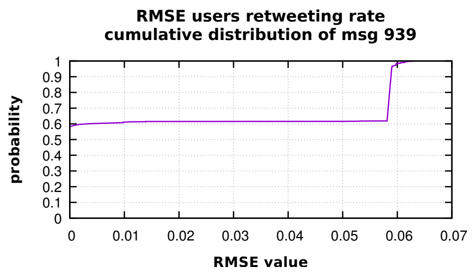


Figure D.487. Graphic showing the RMSE cumulative distribution, for message 939.

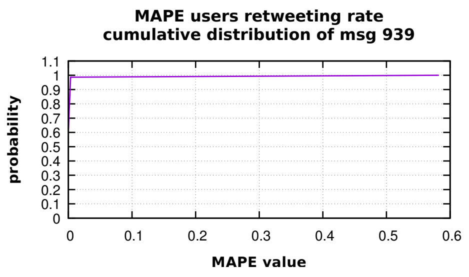


Figure D.488. Graphic showing the MAPE cumulative distribution, for message 939.

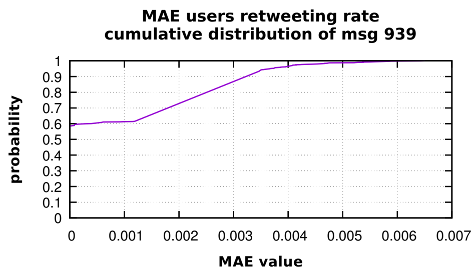


Figure D.489. Graphic showing the MAE cumulative distribution, for message 939.

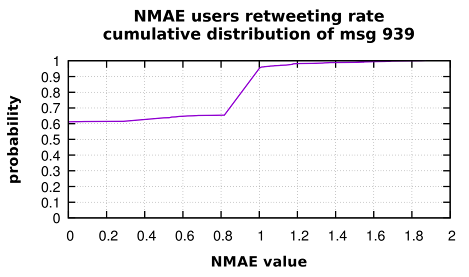


Figure D.490. Graphic showing the NMAE cumulative distribution, for message 939.

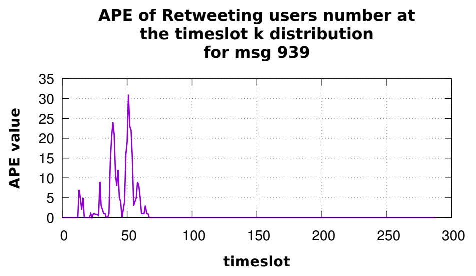


Figure D.491. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 939.

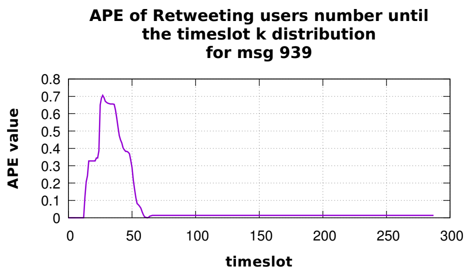


Figure D.492. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 939.

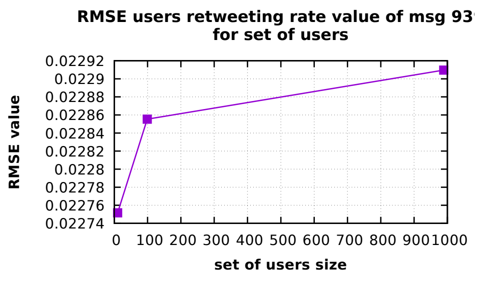


Figure D.493. Graphic showing the RMSE values distribution with respect to the set of users size, for message 939.

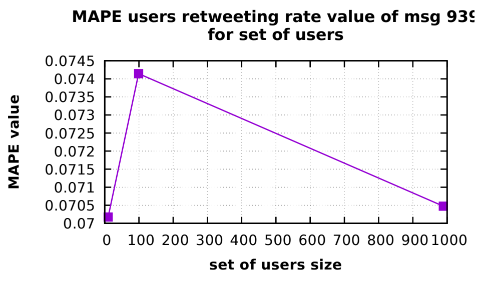


Figure D.494. Graphic showing the MAPE values distribution with respect to the set of users size, for message 939.

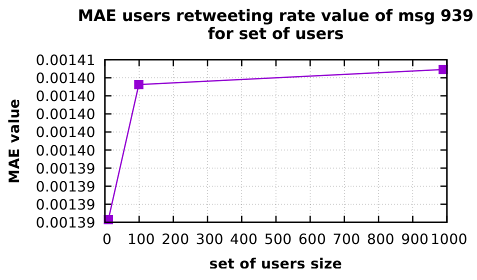


Figure D.495. Graphic showing the MAE values distribution with respect to the set of users size, for message 939.

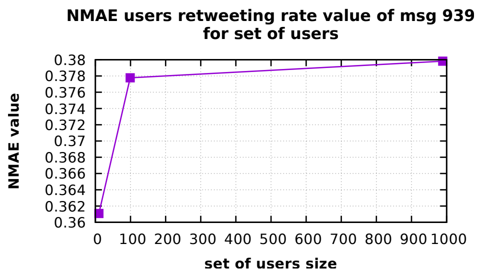


Figure D.496. Graphic showing the NMAE values distribution with respect to the set of users size, for message 939.

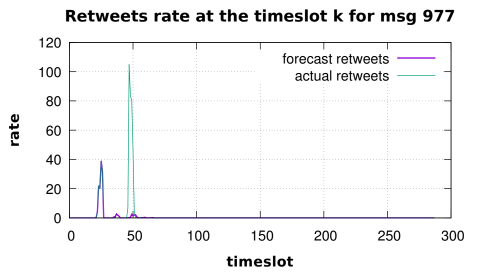


Figure D.497. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 977.

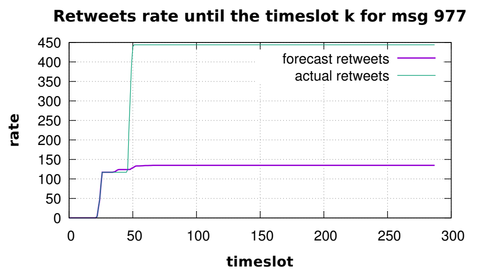


Figure D.498. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 977.

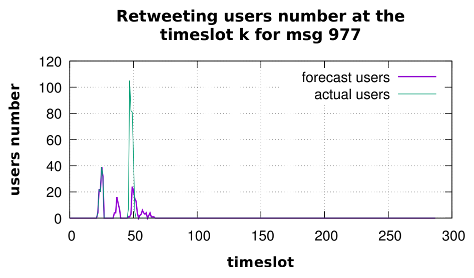


Figure D.499. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 977.

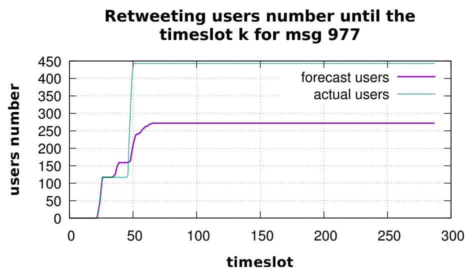


Figure D.500. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 977.

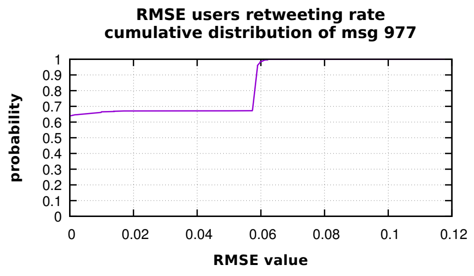


Figure D.501. Graphic showing the RMSE cumulative distribution, for message 977.

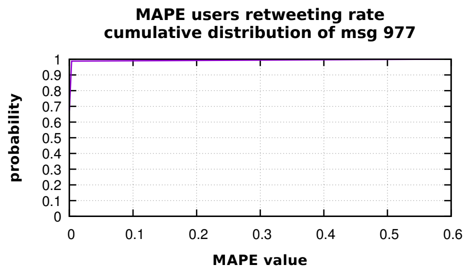


Figure D.502. Graphic showing the MAPE cumulative distribution, for message 977.

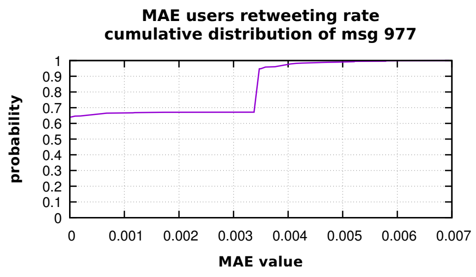


Figure D.503. Graphic showing the MAE cumulative distribution, for message 977.

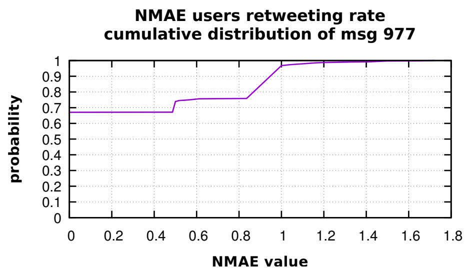


Figure D.504. Graphic showing the NMAE cumulative distribution, for message 977.

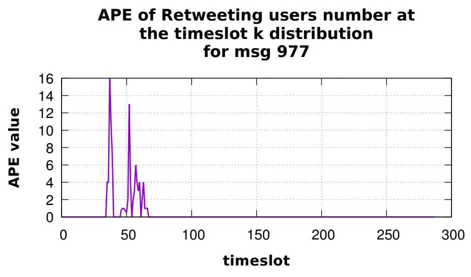


Figure D.505. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 977.

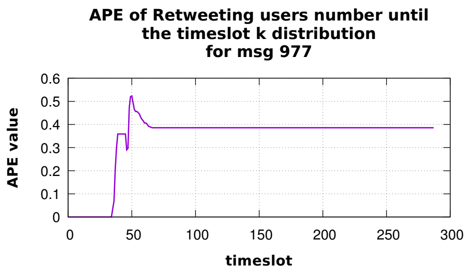


Figure D.506. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 977.

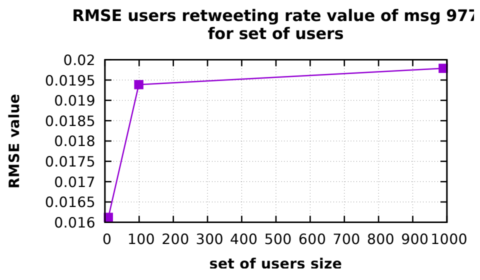


Figure D.507. Graphic showing the RMSE values distribution with respect to the set of users size, for message 977.

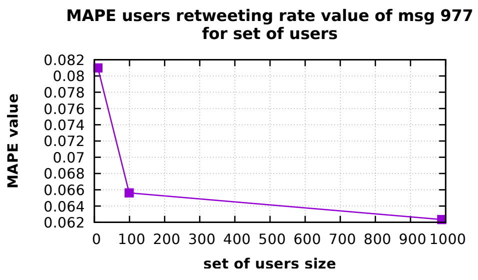


Figure D.508. Graphic showing the MAPE values distribution with respect to the set of users size, for message 977.

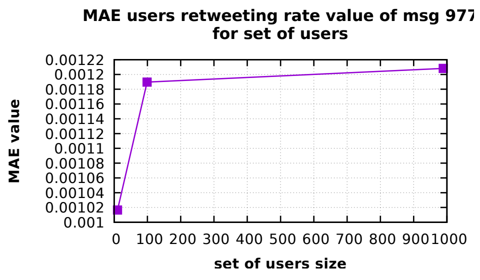


Figure D.509. Graphic showing the MAE values distribution with respect to the set of users size, for message 977.

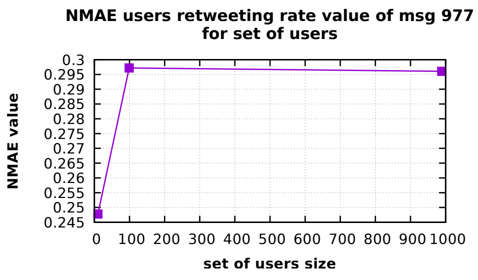


Figure D.510. Graphic showing the NMAE values distribution with respect to the set of users size, for message 977.

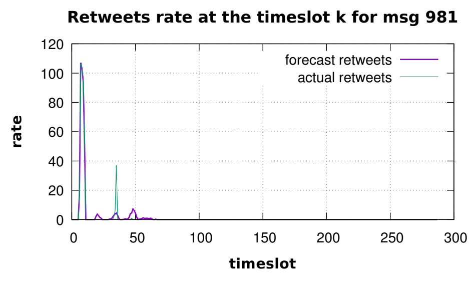


Figure D.511. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 981.

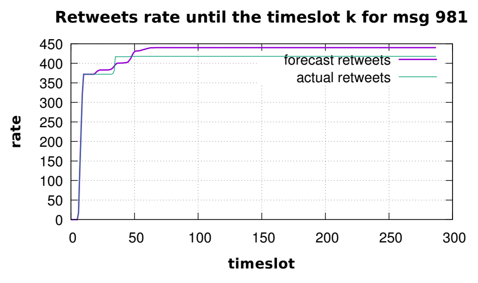


Figure D.512. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 981.

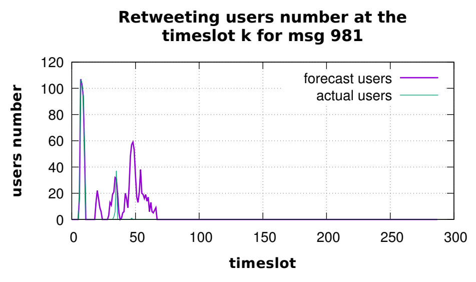


Figure D.513. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 981.

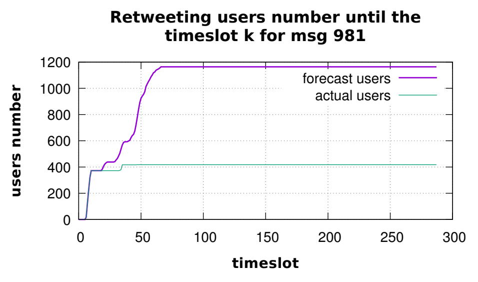


Figure D.514. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 981.

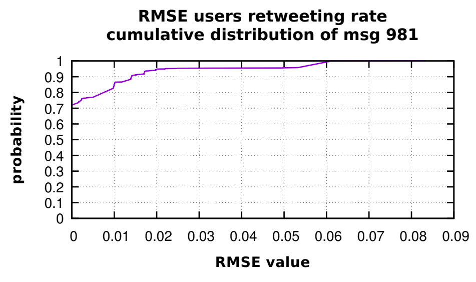


Figure D.515. Graphic showing the RMSE cumulative distribution, for message 981.

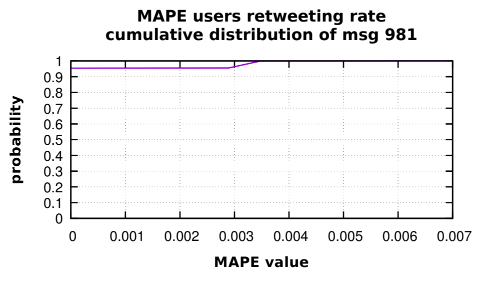


Figure D.516. Graphic showing the MAPE cumulative distribution, for message 981.

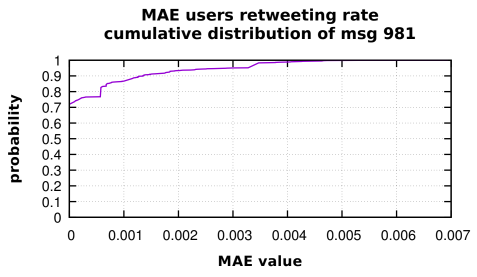


Figure D.517. Graphic showing the MAE cumulative distribution, for message 981.

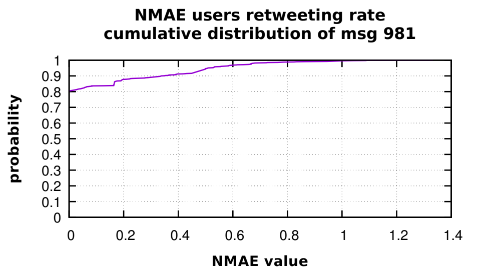


Figure D.518. Graphic showing the NMAE cumulative distribution, for message 981.

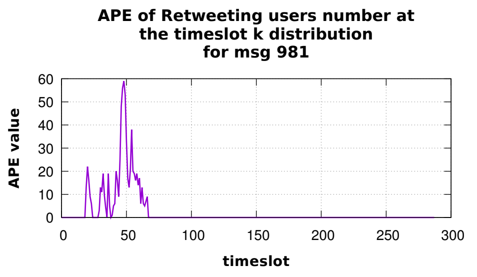


Figure D.519. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 981.

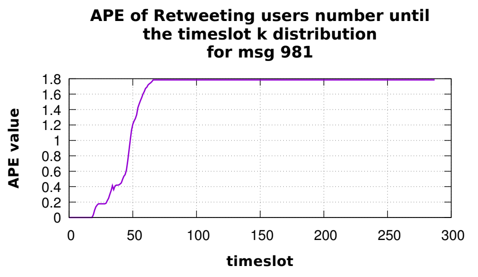


Figure D.520. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 981.

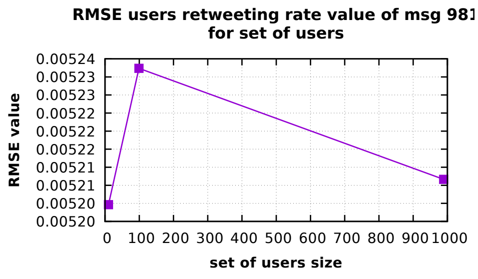


Figure D.521. Graphic showing the RMSE values distribution with respect to the set of users size, for message 981.

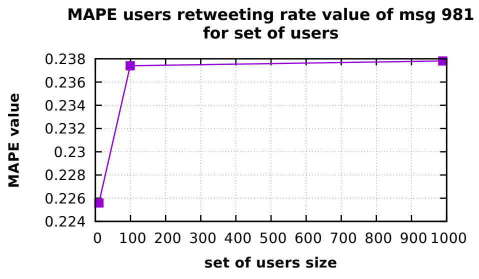


Figure D.522. Graphic showing the MAPE values distribution with respect to the set of users size, for message 981.

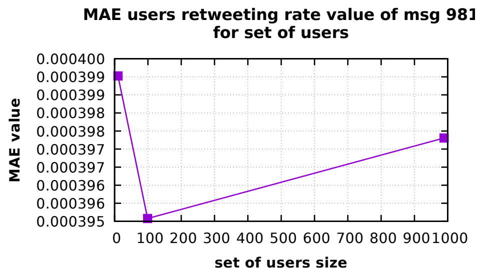


Figure D.523. Graphic showing the MAE values distribution with respect to the set of users size, for message 981.

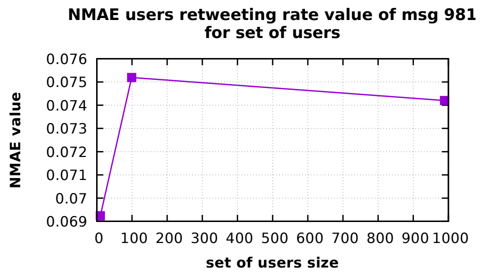


Figure D.524. Graphic showing the NMAE values distribution with respect to the set of users size, for message 981.

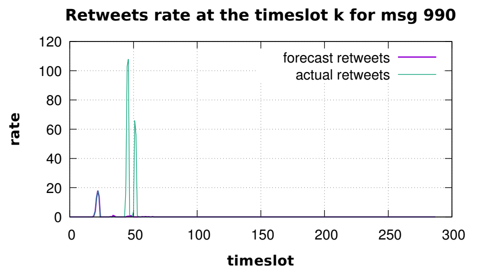


Figure D.525. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 990.

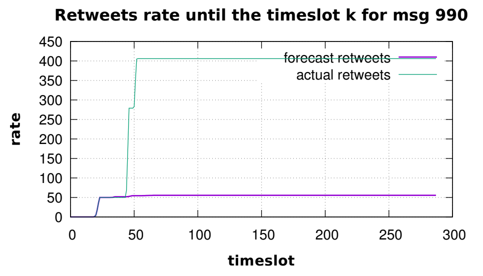


Figure D.526. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 990.

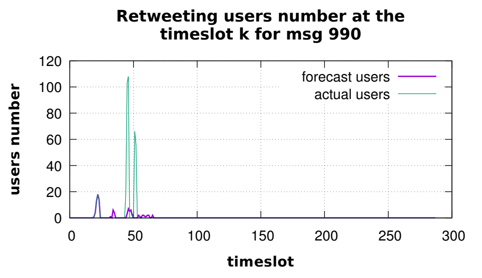


Figure D.527. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 990.

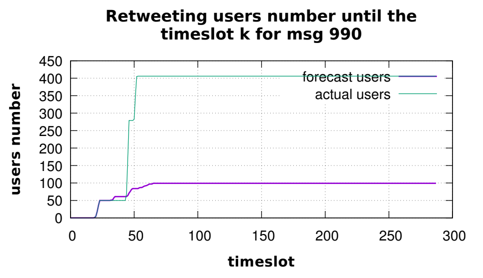


Figure D.528. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 990.

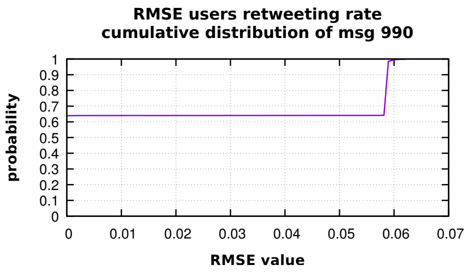


Figure D.529. Graphic showing the RMSE cumulative distribution, for message 990.

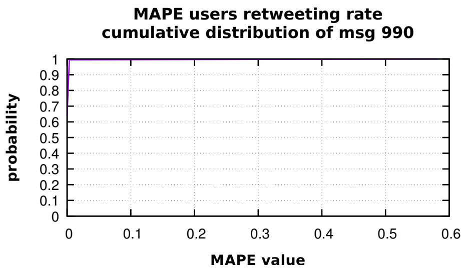


Figure D.530. Graphic showing the MAPE cumulative distribution, for message 990.

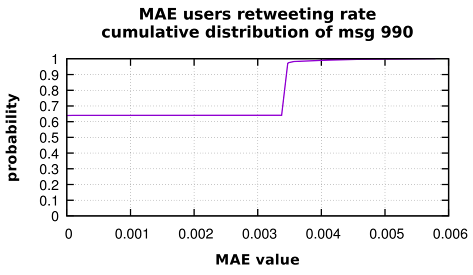


Figure D.531. Graphic showing the MAE cumulative distribution, for message 990.

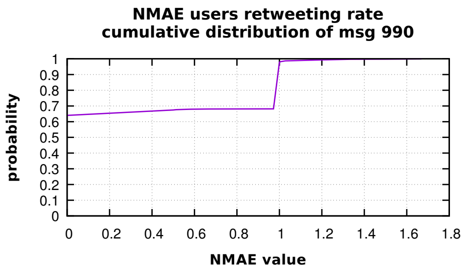


Figure D.532. Graphic showing the NMAE cumulative distribution, for message 990.

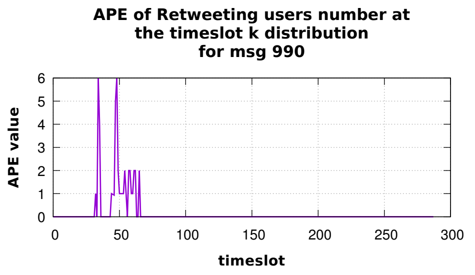


Figure D.533. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 990.

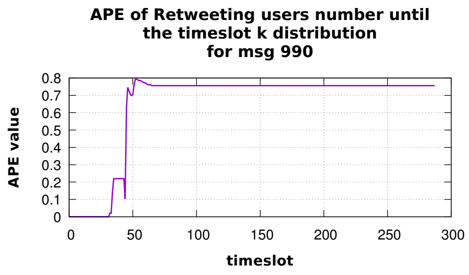


Figure D.534. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 990.

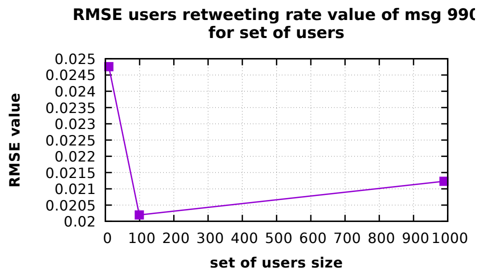


Figure D.535. Graphic showing the RMSE values distribution with respect to the set of users size, for message 990.

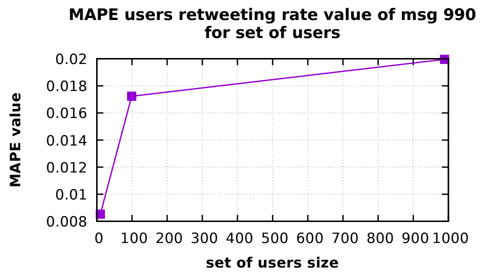


Figure D.536. Graphic showing the MAPE values distribution with respect to the set of users size, for message 990.

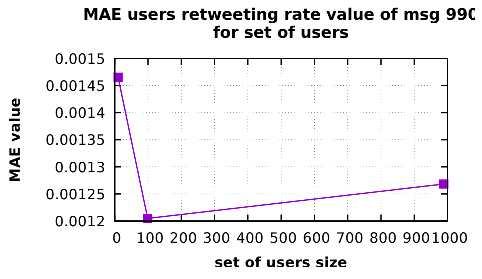


Figure D.537. Graphic showing the MAE values distribution with respect to the set of users size, for message 990.

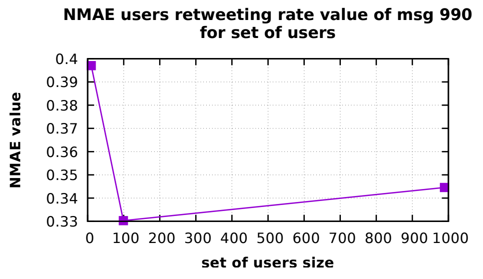


Figure D.538. Graphic showing the NMAE values distribution with respect to the set of users size, for message 990.

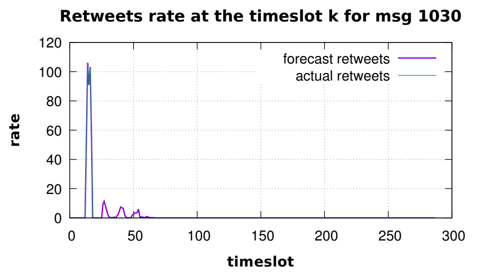


Figure D.539. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1030.

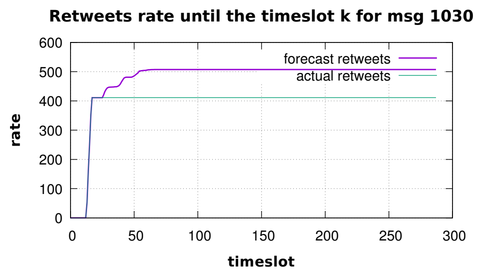


Figure D.540. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1030.

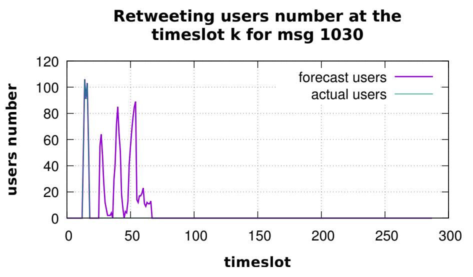


Figure D.541. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1030.

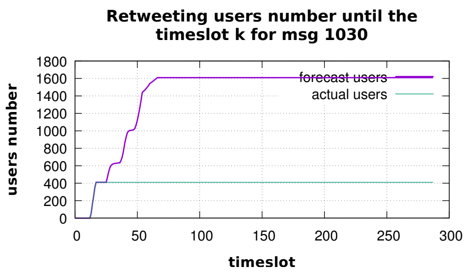


Figure D.542. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1030.

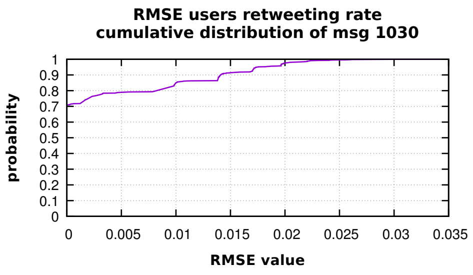


Figure D.543. Graphic showing the RMSE cumulative distribution, for message 1030.

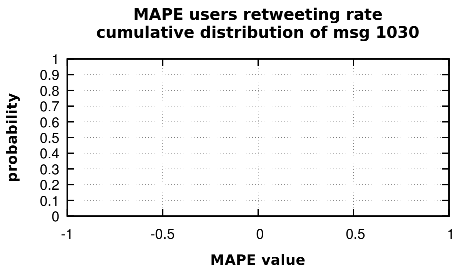


Figure D.544. Graphic showing the MAPE cumulative distribution, for message 1030.

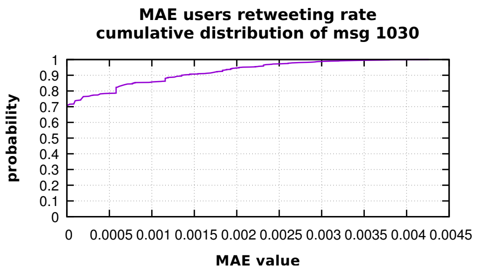


Figure D.545. Graphic showing the MAE cumulative distribution, for message 1030.

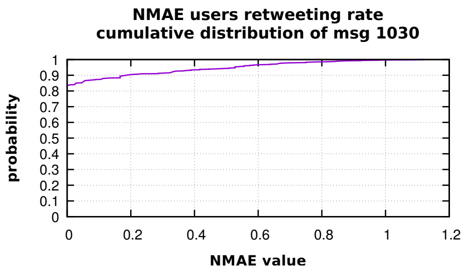


Figure D.546. Graphic showing the NMAE cumulative distribution, for message 1030.

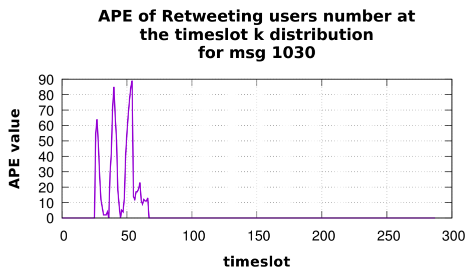


Figure D.547. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1030.

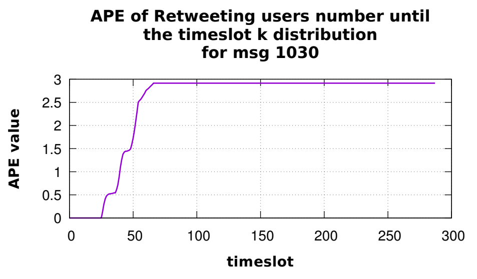


Figure D.548. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1030.

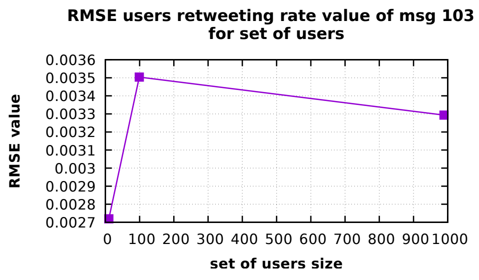


Figure D.549. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1030.

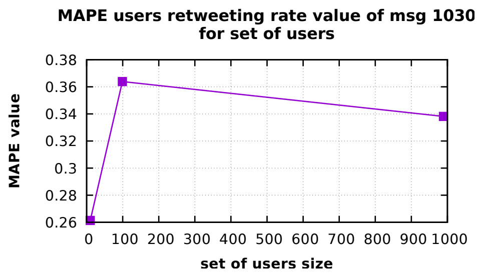


Figure D.550. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1030.

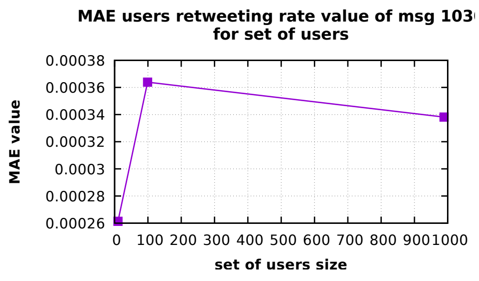


Figure D.551. Graphic showing the MAE values distribution with respect to the set of users size, for message 1030.

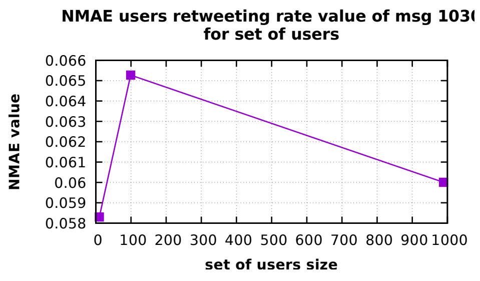


Figure D.552. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1030.

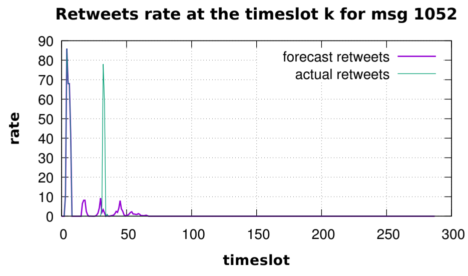


Figure D.553. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1052.

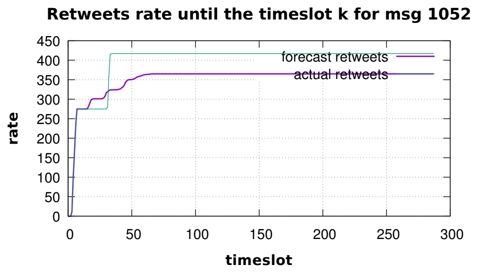


Figure D.554. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1052.

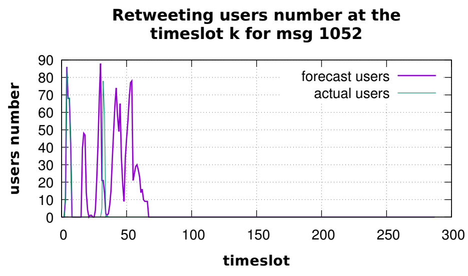


Figure D.555. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1052.

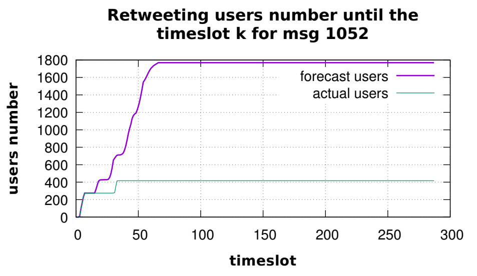


Figure D.556. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1052.

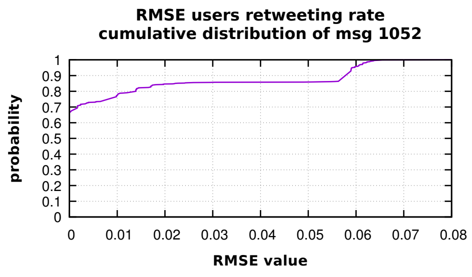


Figure D.557. Graphic showing the RMSE cumulative distribution, for message 1052.

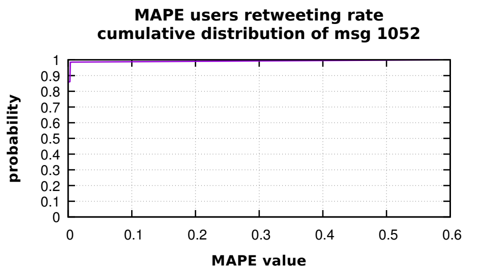


Figure D.558. Graphic showing the MAPE cumulative distribution, for message 1052.

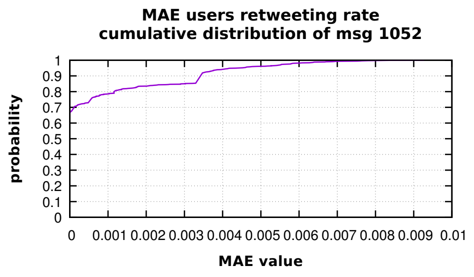


Figure D.559. Graphic showing the MAE cumulative distribution, for message 1052.

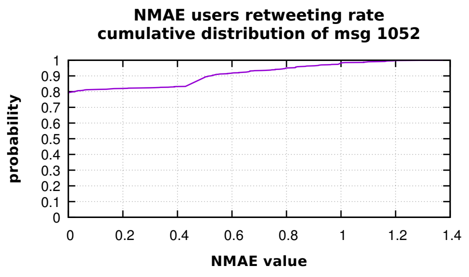


Figure D.560. Graphic showing the NMAE cumulative distribution, for message 1052.

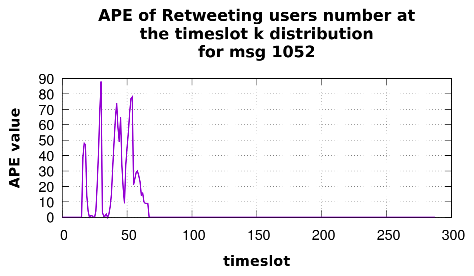


Figure D.561. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1052.

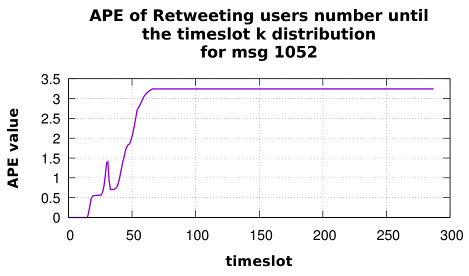


Figure D.562. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1052.

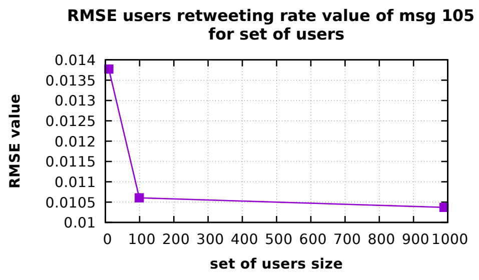


Figure D.563. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1052.

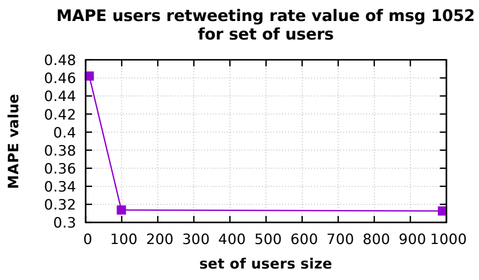


Figure D.564. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1052.

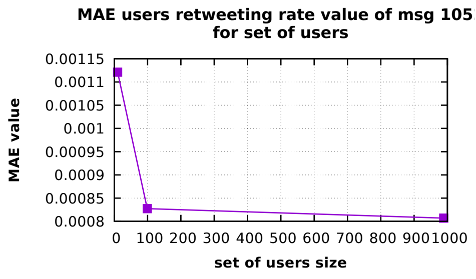


Figure D.565. Graphic showing the MAE values distribution with respect to the set of users size, for message 1052.

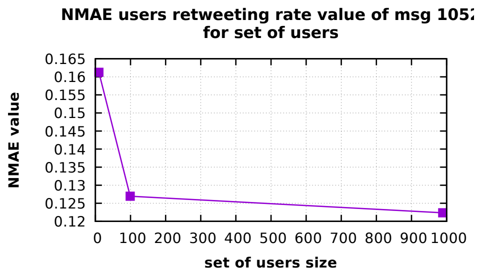


Figure D.566. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1052.

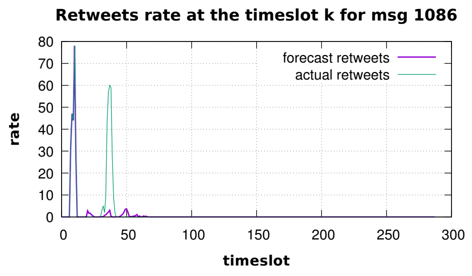


Figure D.567. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1086.

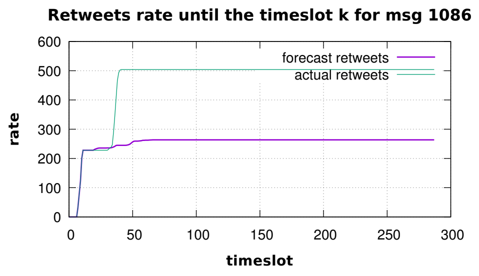


Figure D.568. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1086.

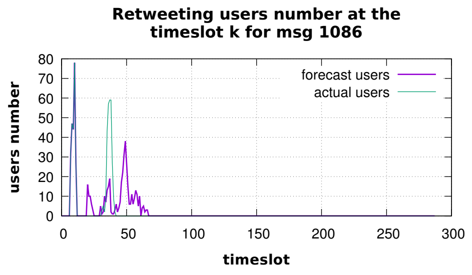


Figure D.569. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1086.

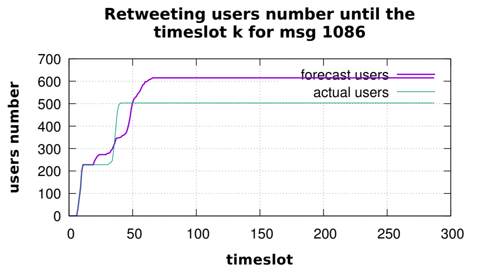


Figure D.570. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1086.

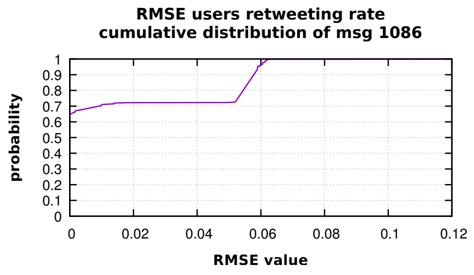


Figure D.571. Graphic showing the RMSE cumulative distribution, for message 1086.

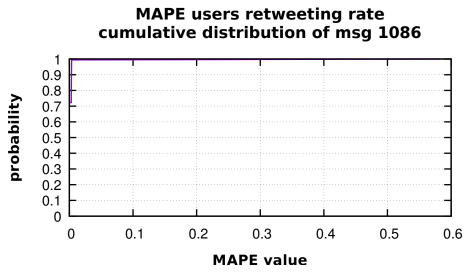


Figure D.572. Graphic showing the MAPE cumulative distribution, for message 1086.

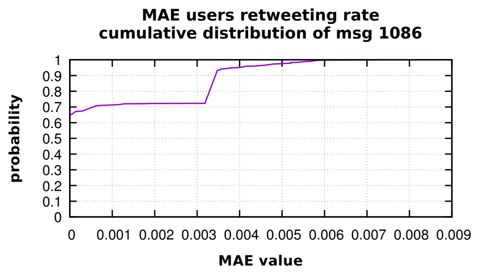


Figure D.573. Graphic showing the MAE cumulative distribution, for message 1086.

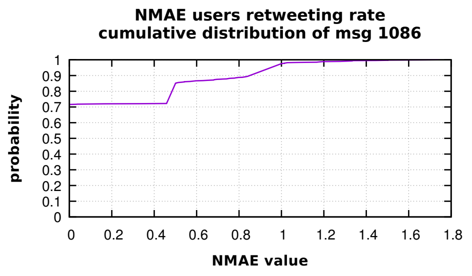


Figure D.574. Graphic showing the NMAE cumulative distribution, for message 1086.

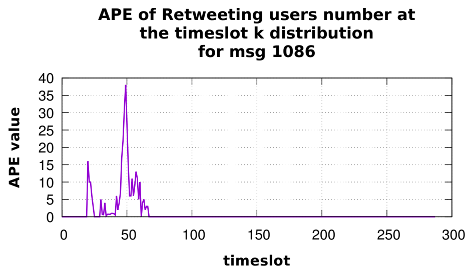


Figure D.575. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1086.

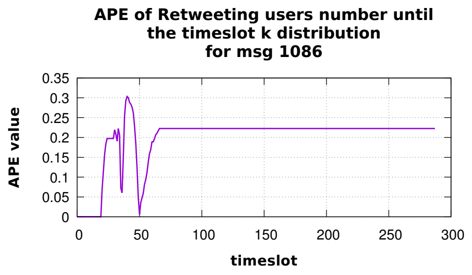


Figure D.576. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1086.

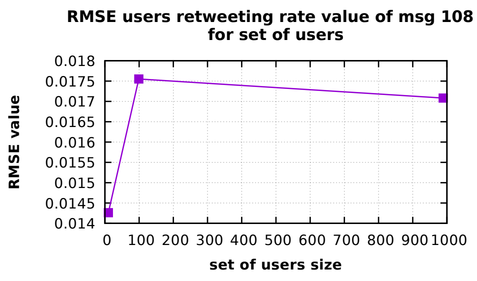


Figure D.577. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1086.

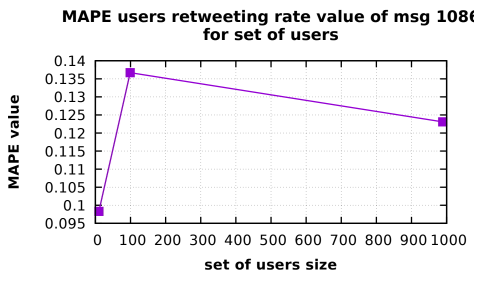


Figure D.578. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1086.

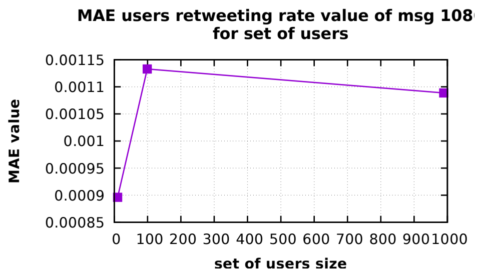


Figure D.579. Graphic showing the MAE values distribution with respect to the set of users size, for message 1086.

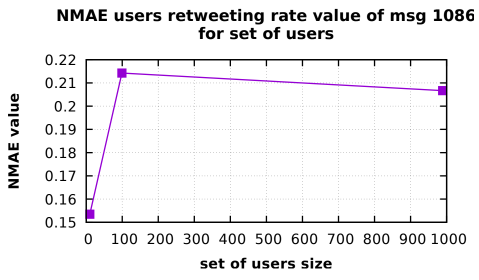


Figure D.580. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1086.

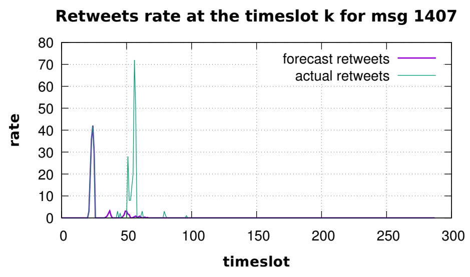


Figure D.581. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1407.

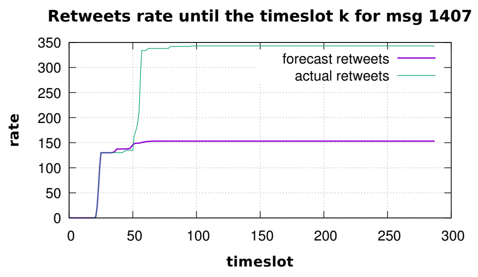


Figure D.582. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1407.

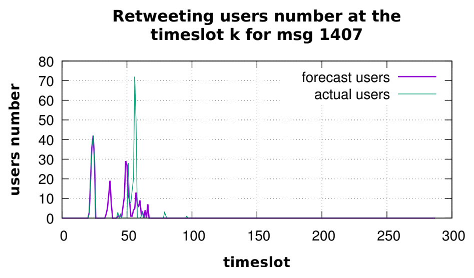


Figure D.583. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1407.

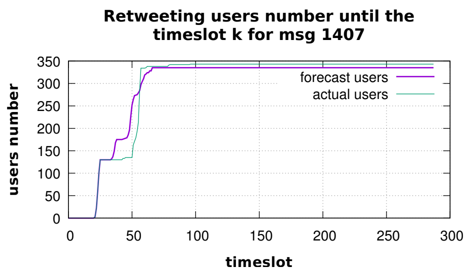


Figure D.584. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1407.

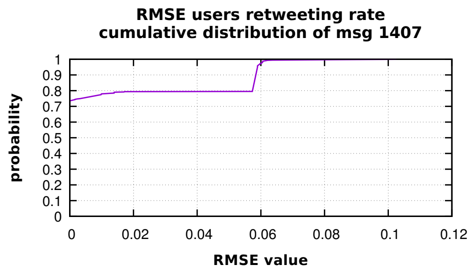


Figure D.585. Graphic showing the RMSE cumulative distribution, for message 1407.

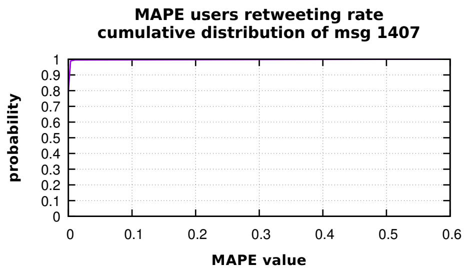


Figure D.586. Graphic showing the MAPE cumulative distribution, for message 1407.

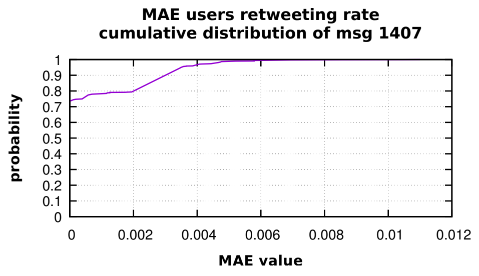


Figure D.587. Graphic showing the MAE cumulative distribution, for message 1407.

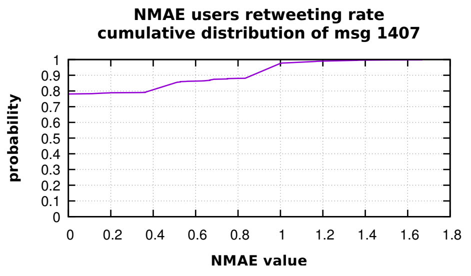


Figure D.588. Graphic showing the NMAE cumulative distribution, for message 1407.

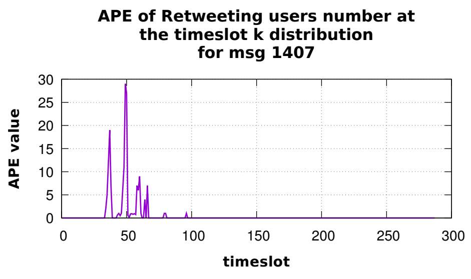


Figure D.589. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1407.

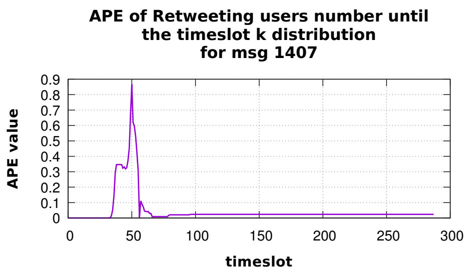


Figure D.590. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1407.

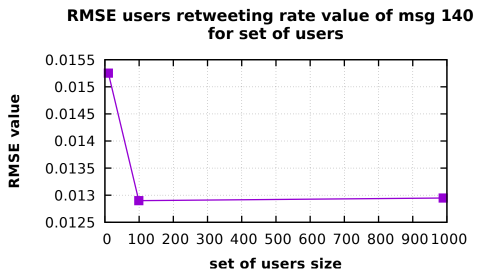


Figure D.591. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1407.

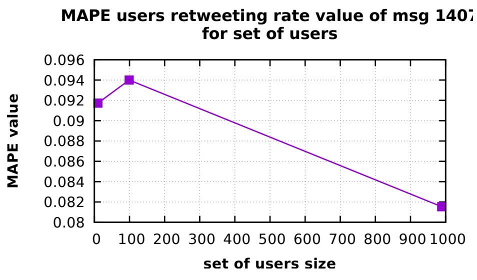


Figure D.592. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1407.

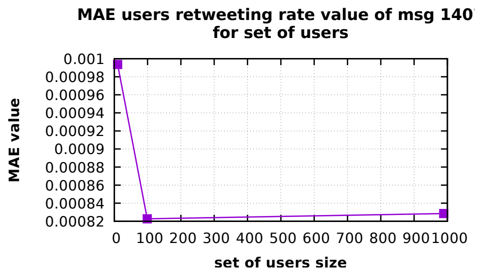


Figure D.593. Graphic showing the MAE values distribution with respect to the set of users size, for message 1407.

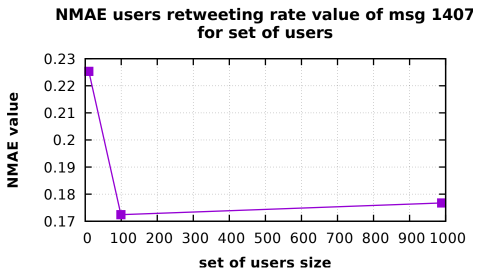


Figure D.594. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1407.

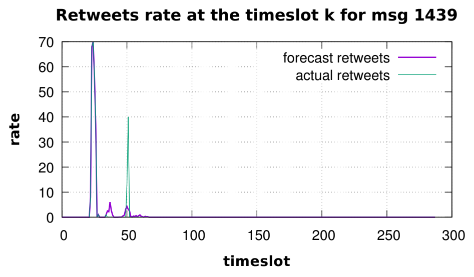


Figure D.595. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1439.

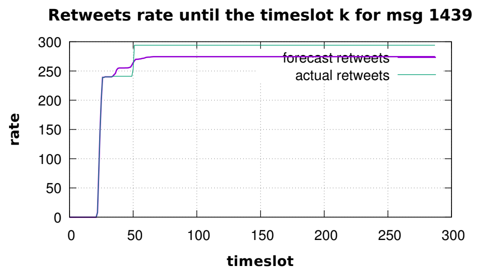


Figure D.596. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1439.

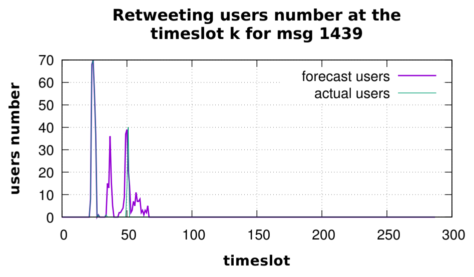


Figure D.597. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1439.

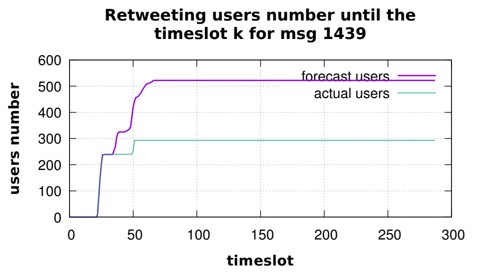


Figure D.598. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1439.

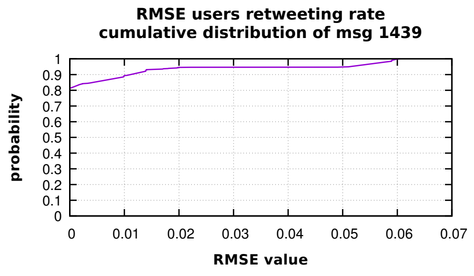


Figure D.599. Graphic showing the RMSE cumulative distribution, for message 1439.

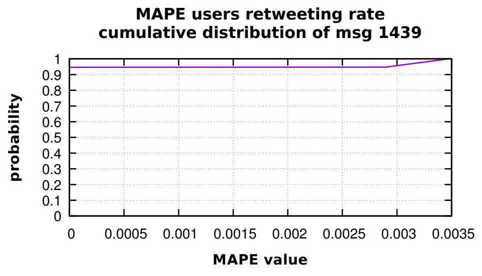


Figure D.600. Graphic showing the MAPE cumulative distribution, for message 1439.

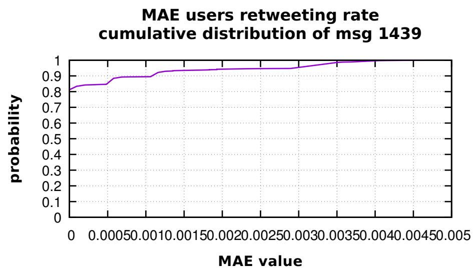


Figure D.601. Graphic showing the MAE cumulative distribution, for message 1439.

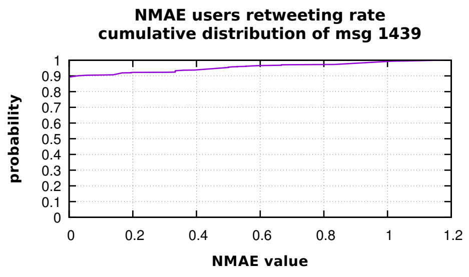


Figure D.602. Graphic showing the NMAE cumulative distribution, for message 1439.

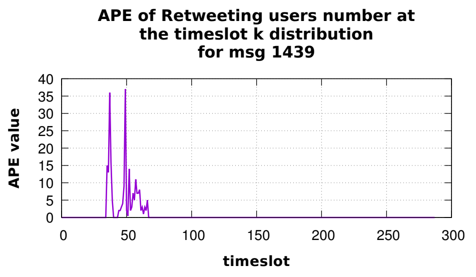


Figure D.603. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1439.

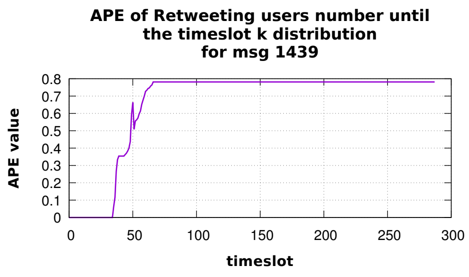


Figure D.604. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1439.

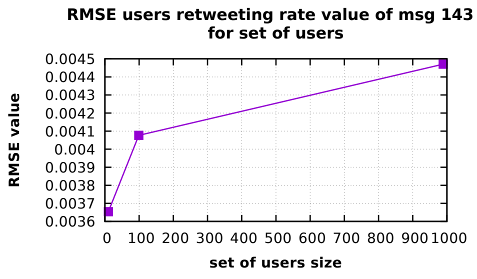


Figure D.605. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1439.

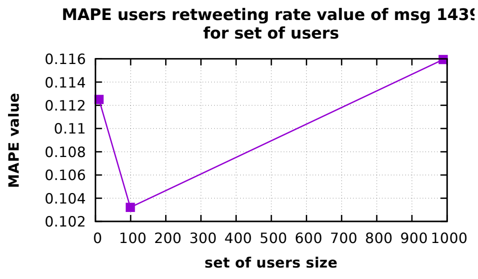


Figure D.606. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1439.

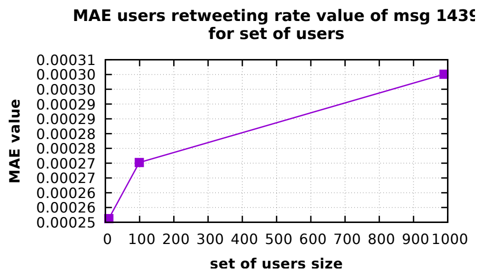


Figure D.607. Graphic showing the MAE values distribution with respect to the set of users size, for message 1439.

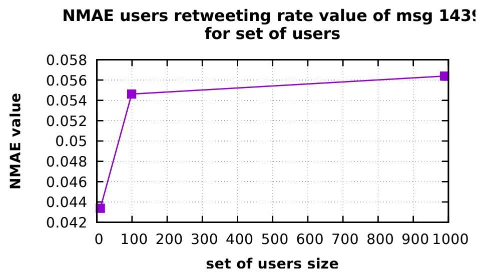


Figure D.608. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1439.

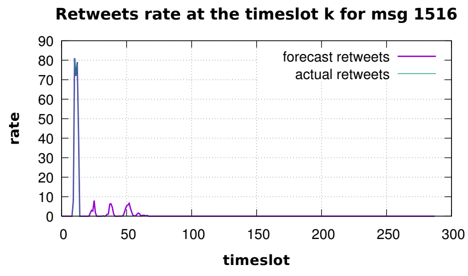


Figure D.609. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1516.

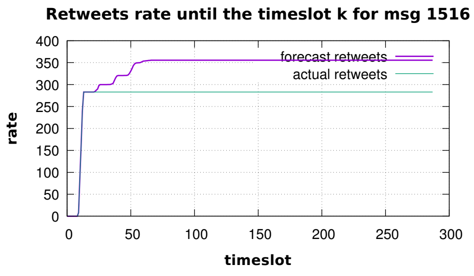


Figure D.610. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1516.

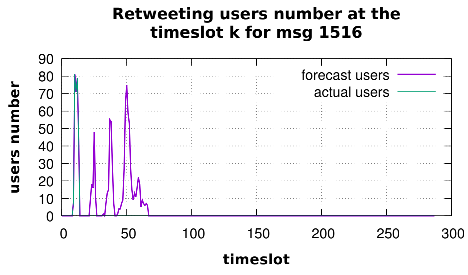


Figure D.611. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1516.

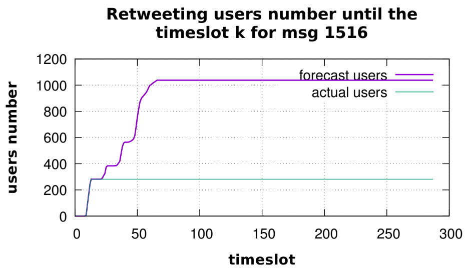


Figure D.612. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1516.

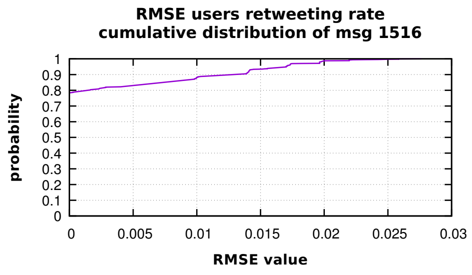


Figure D.613. Graphic showing the RMSE cumulative distribution, for message 1516.

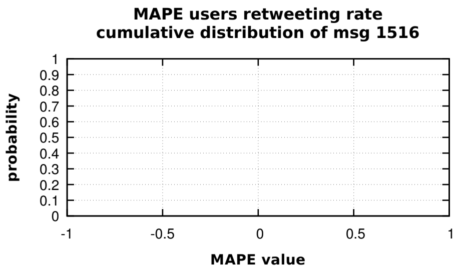


Figure D.614. Graphic showing the MAPE cumulative distribution, for message 1516.

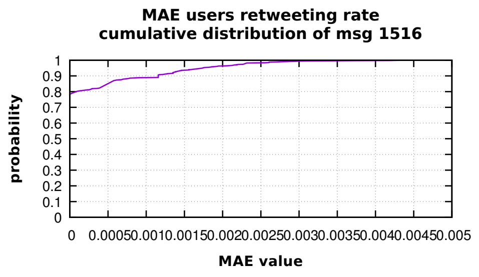


Figure D.615. Graphic showing the MAE cumulative distribution, for message 1516.

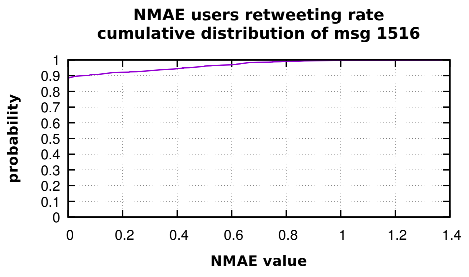


Figure D.616. Graphic showing the NMAE cumulative distribution, for message 1516.

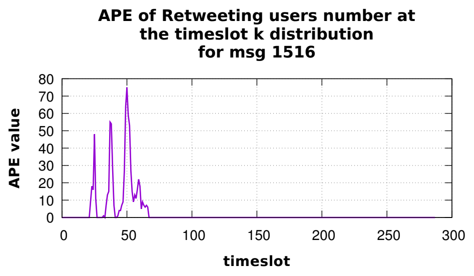


Figure D.617. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1516.

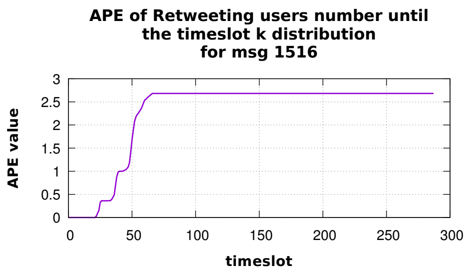


Figure D.618. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1516.

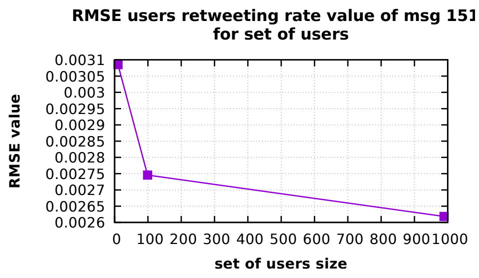


Figure D.619. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1516.

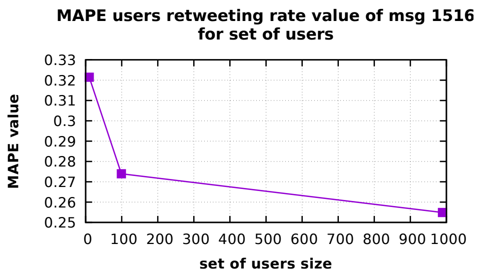


Figure D.620. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1516.

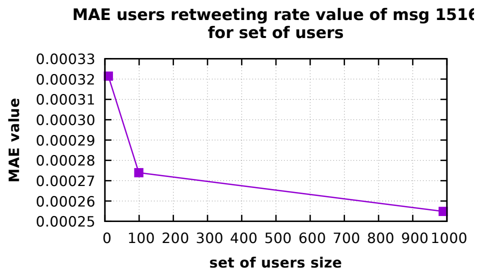


Figure D.621. Graphic showing the MAE values distribution with respect to the set of users size, for message 1516.

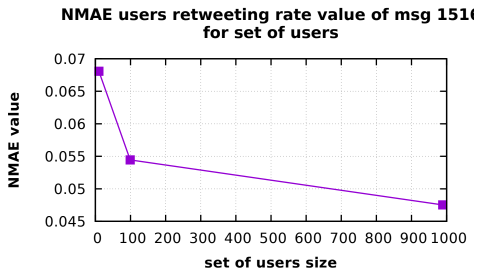


Figure D.622. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1516.

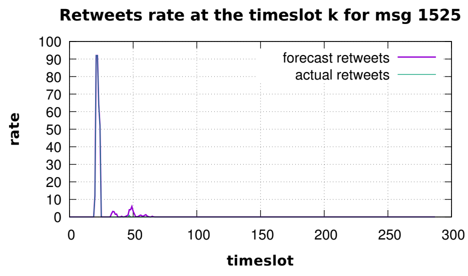


Figure D.623. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1525.

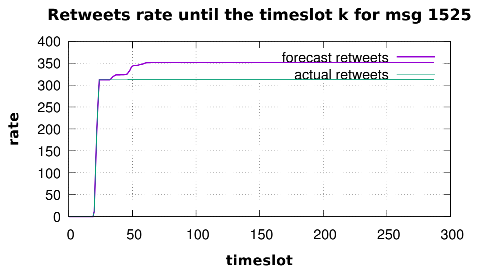


Figure D.624. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1525.

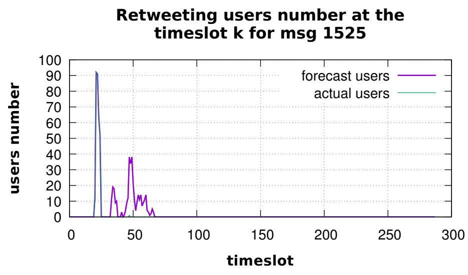


Figure D.625. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1525.

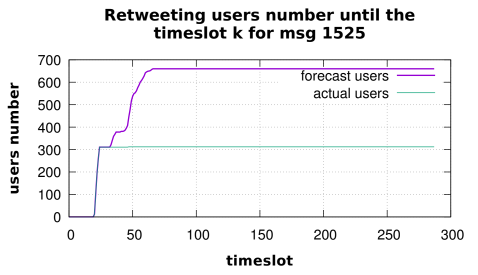


Figure D.626. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1525.

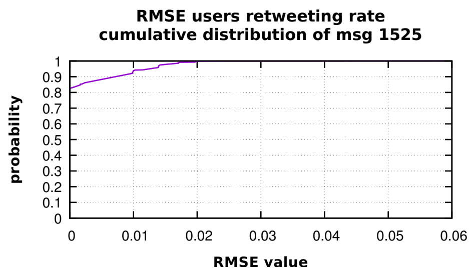


Figure D.627. Graphic showing the RMSE cumulative distribution, for message 1525.

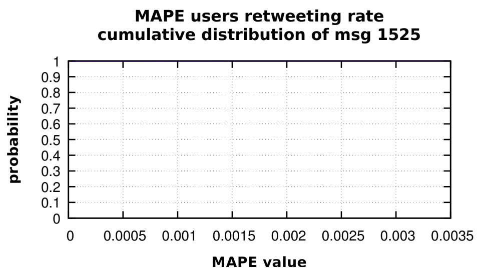


Figure D.628. Graphic showing the MAPE cumulative distribution, for message 1525.

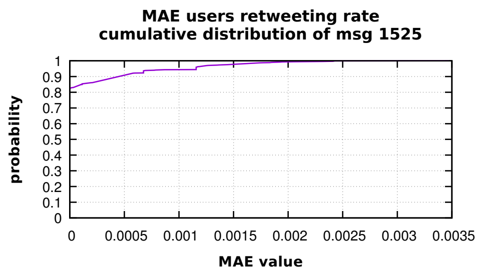


Figure D.629. Graphic showing the MAE cumulative distribution, for message 1525.

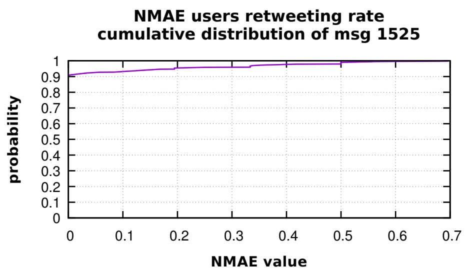


Figure D.630. Graphic showing the NMAE cumulative distribution, for message 1525.

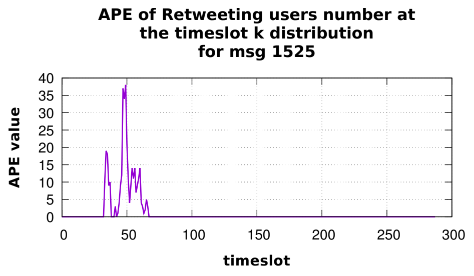


Figure D.631. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1525.

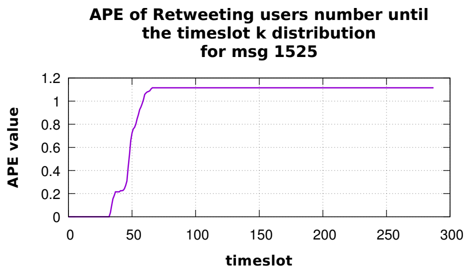


Figure D.632. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1525.

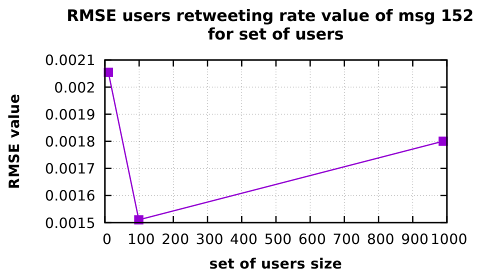


Figure D.633. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1525.

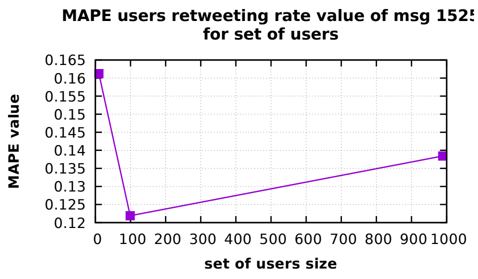


Figure D.634. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1525.

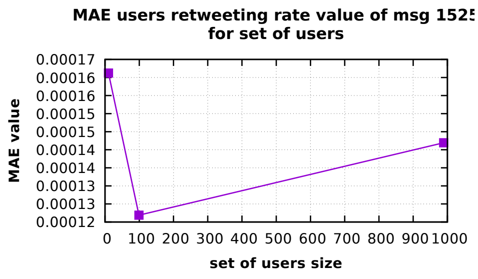


Figure D.635. Graphic showing the MAE values distribution with respect to the set of users size, for message 1525.

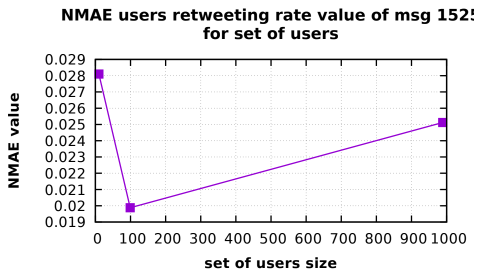


Figure D.636. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1525.

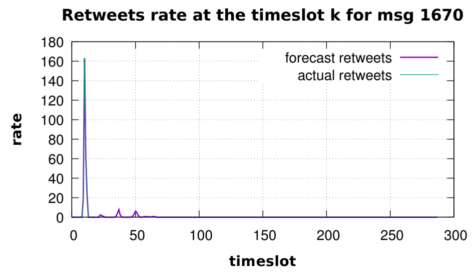


Figure D.637. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1670.

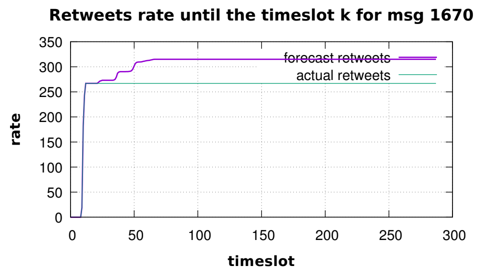


Figure D.638. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1670.

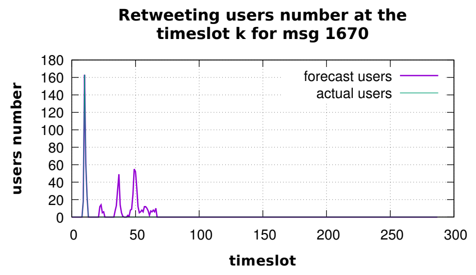


Figure D.639. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1670.

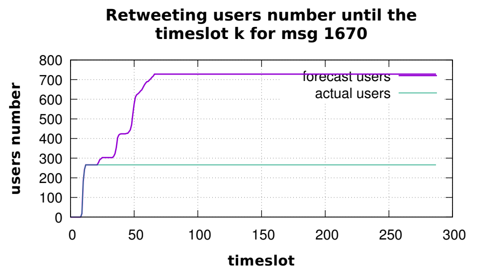


Figure D.640. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1670.

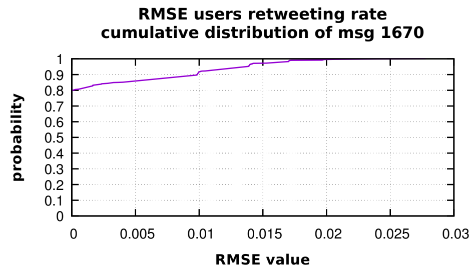


Figure D.641. Graphic showing the RMSE cumulative distribution, for message 1670.

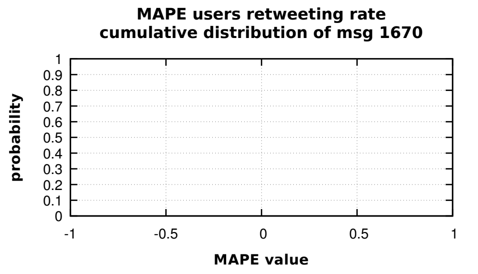


Figure D.642. Graphic showing the MAPE cumulative distribution, for message 1670.

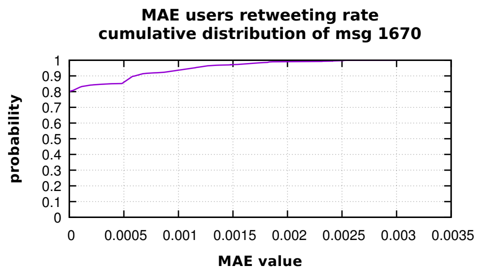


Figure D.643. Graphic showing the MAE cumulative distribution, for message 1670.

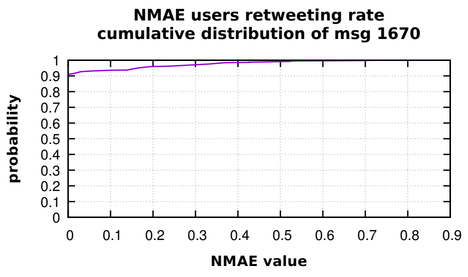


Figure D.644. Graphic showing the NMAE cumulative distribution, for message 1670.

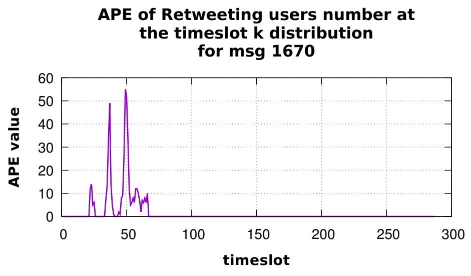


Figure D.645. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1670.

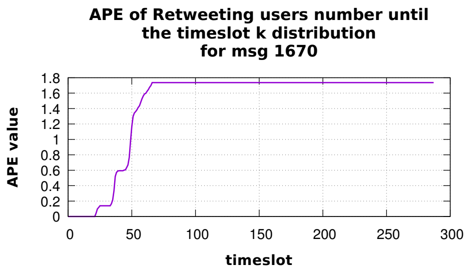


Figure D.646. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1670.

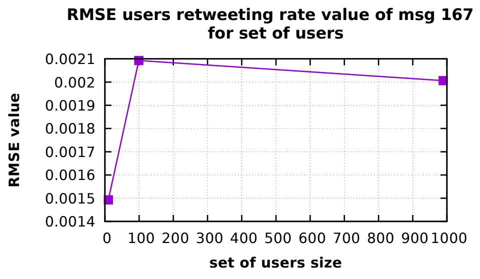


Figure D.647. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1670.

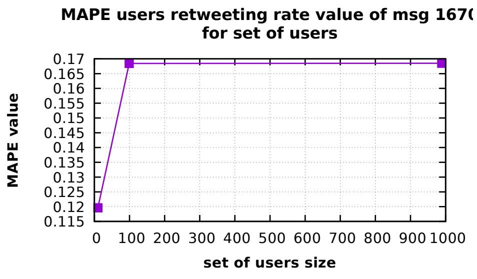


Figure D.648. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1670.

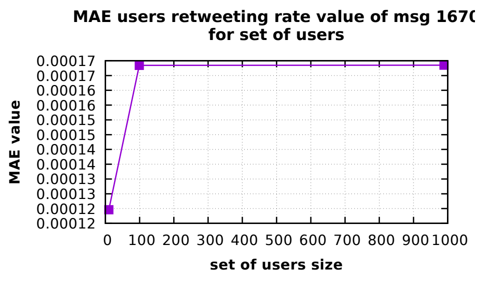


Figure D.649. Graphic showing the MAE values distribution with respect to the set of users size, for message 1670.

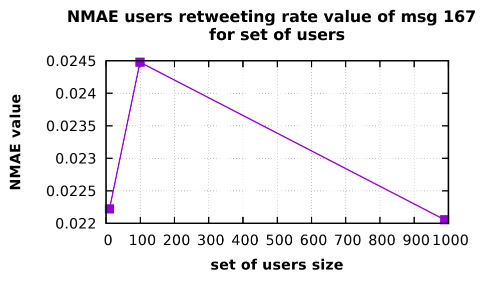


Figure D.650. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1670.

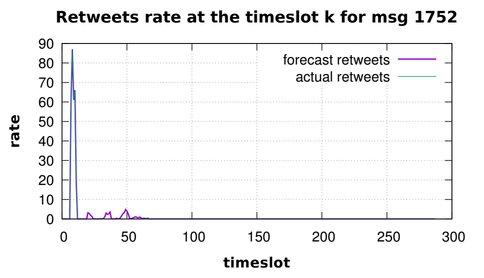


Figure D.651. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1752.

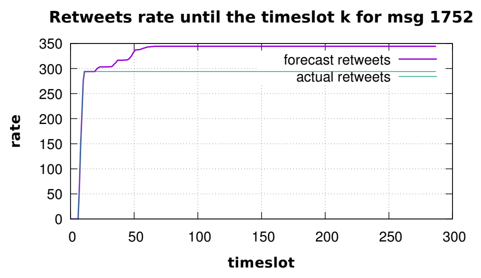


Figure D.652. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1752.

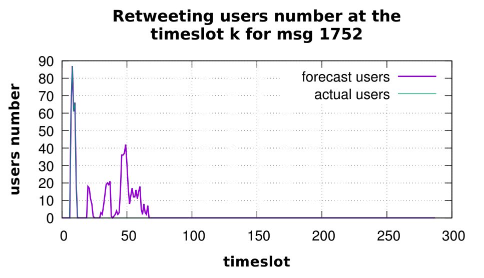


Figure D.653. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1752.

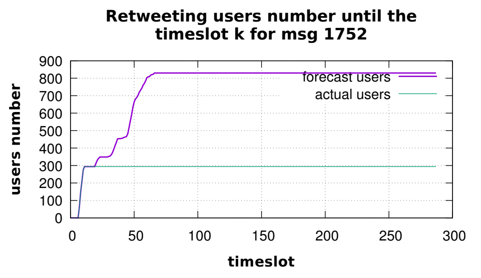


Figure D.654. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1752.

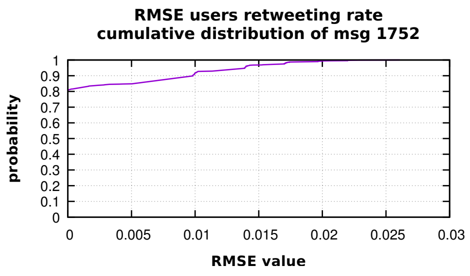


Figure D.655. Graphic showing the RMSE cumulative distribution, for message 1752.

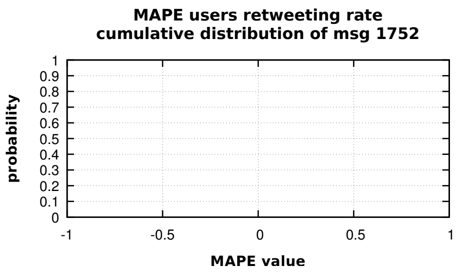


Figure D.656. Graphic showing the MAPE cumulative distribution, for message 1752.

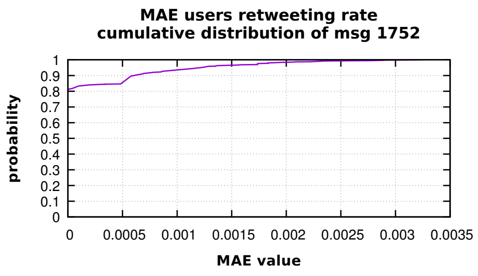


Figure D.657. Graphic showing the MAE cumulative distribution, for message 1752.

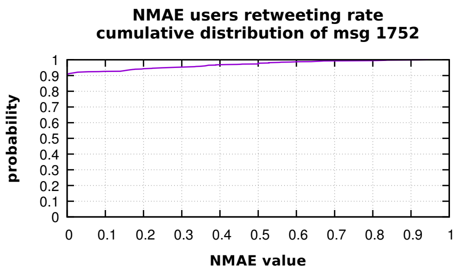


Figure D.658. Graphic showing the NMAE cumulative distribution, for message 1752.

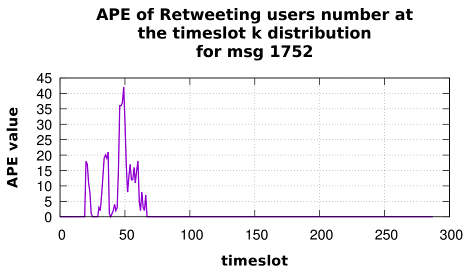


Figure D.659. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1752.

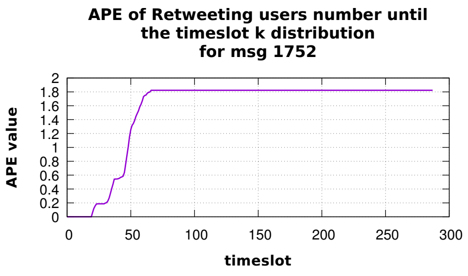


Figure D.660. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1752.

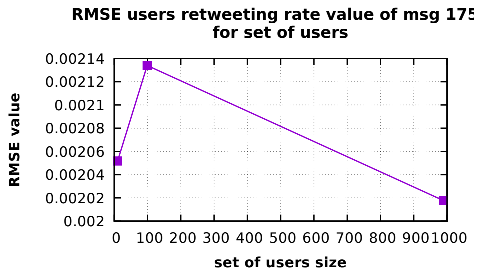


Figure D.661. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1752.

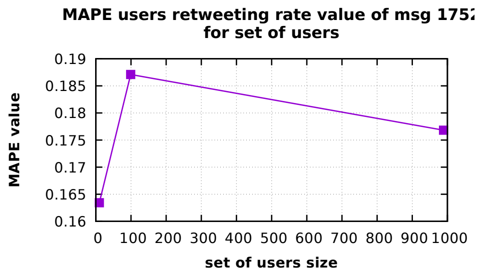


Figure D.662. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1752.

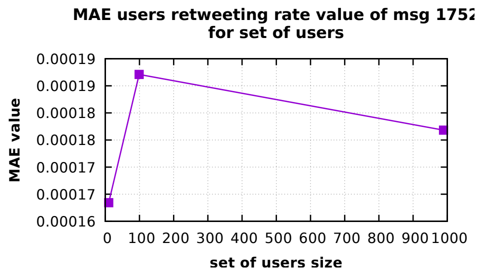


Figure D.663. Graphic showing the MAE values distribution with respect to the set of users size, for message 1752.

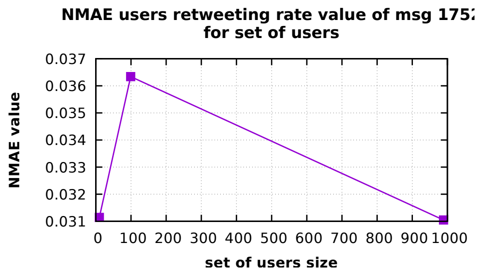


Figure D.664. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1752.

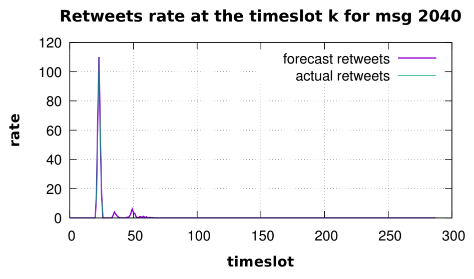


Figure D.665. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 2040.

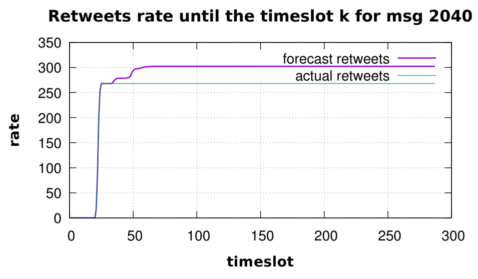


Figure D.666. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 2040.

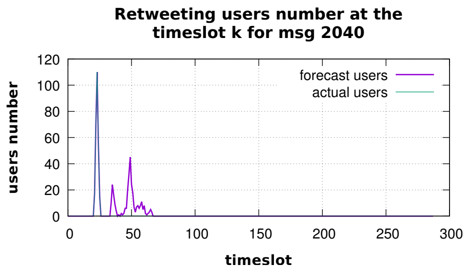


Figure D.667. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 2040.

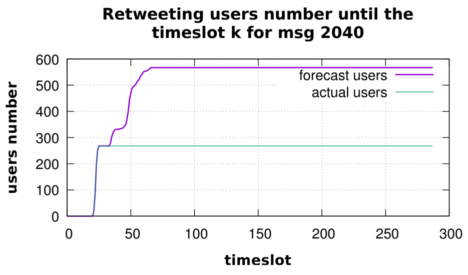


Figure D.668. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 2040.

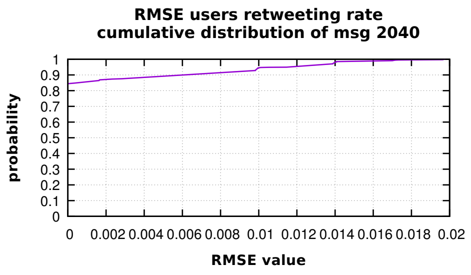


Figure D.669. Graphic showing the RMSE cumulative distribution, for message 2040.

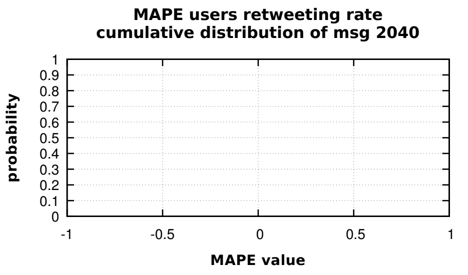


Figure D.670. Graphic showing the MAPE cumulative distribution, for message 2040.

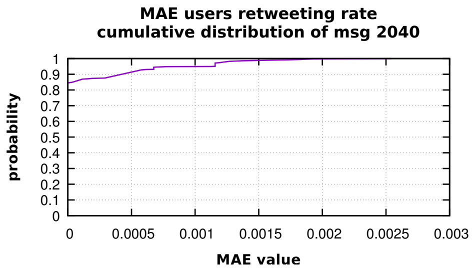


Figure D.671. Graphic showing the MAE cumulative distribution, for message 2040.

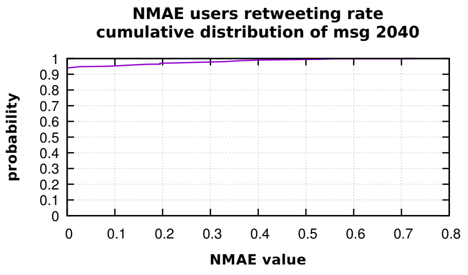


Figure D.672. Graphic showing the NMAE cumulative distribution, for message 2040.

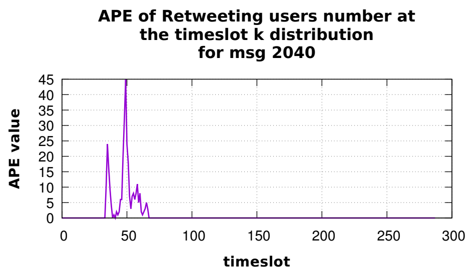


Figure D.673. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 2040.

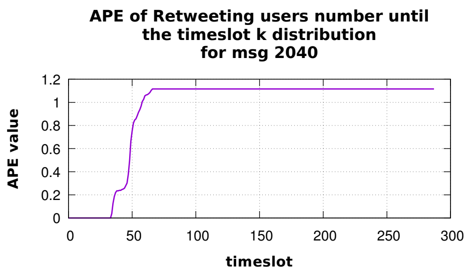


Figure D.674. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 2040.

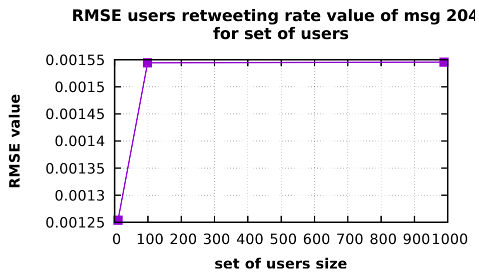


Figure D.675. Graphic showing the RMSE values distribution with respect to the set of users size, for message 2040.

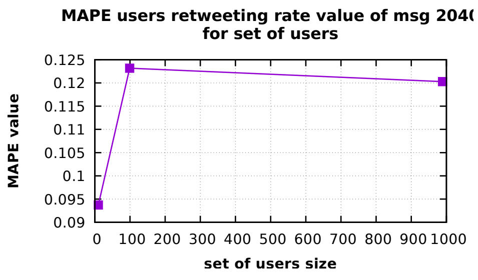


Figure D.676. Graphic showing the MAPE values distribution with respect to the set of users size, for message 2040.

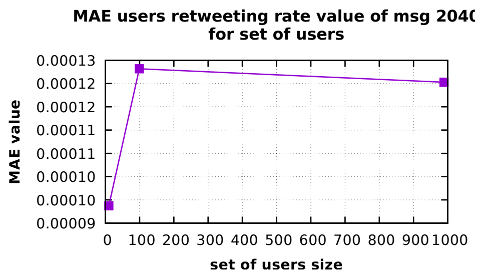


Figure D.677. Graphic showing the MAE values distribution with respect to the set of users size, for message 2040.

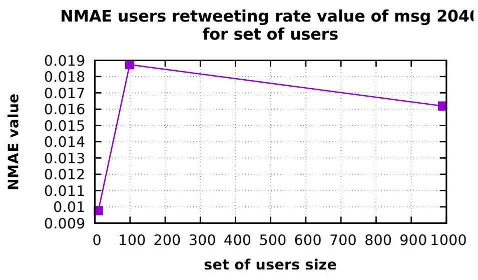


Figure D.678. Graphic showing the NMAE values distribution with respect to the set of users size, for message 2040.

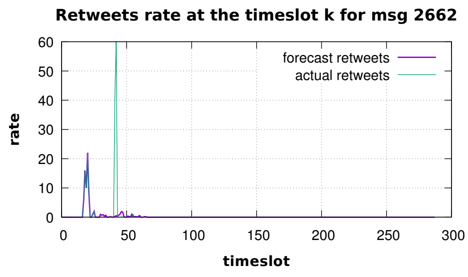


Figure D.679. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 2662.

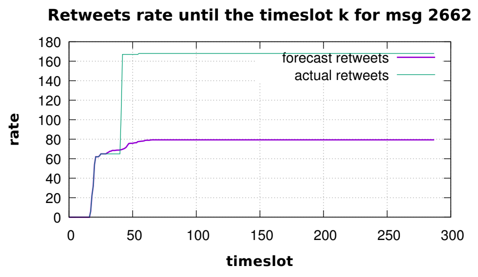


Figure D.680. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 2662.

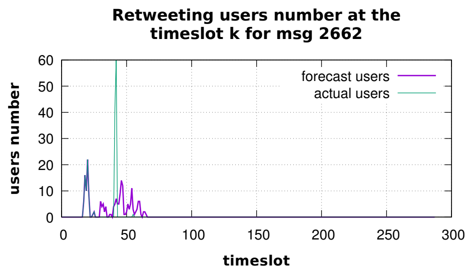


Figure D.681. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 2662.

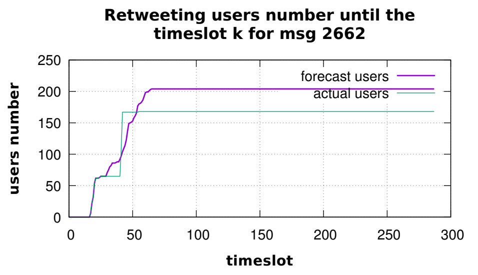


Figure D.682. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 2662.

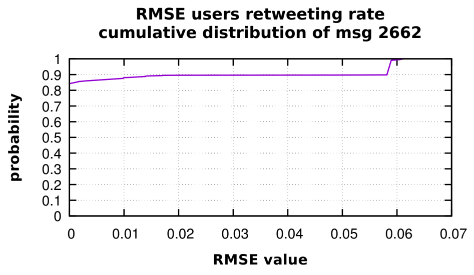


Figure D.683. Graphic showing the RMSE cumulative distribution, for message 2662.

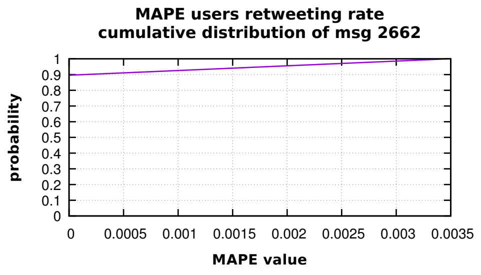


Figure D.684. Graphic showing the MAPE cumulative distribution, for message 2662.

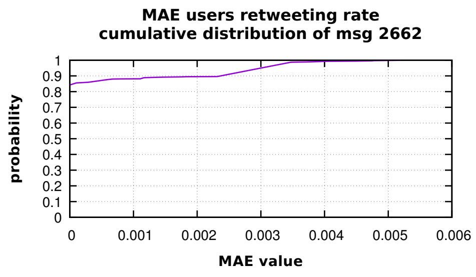


Figure D.685. Graphic showing the MAE cumulative distribution, for message 2662.

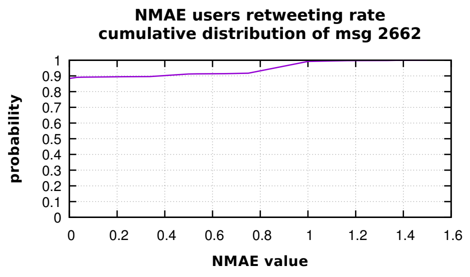


Figure D.686. Graphic showing the NMAE cumulative distribution, for message 2662.

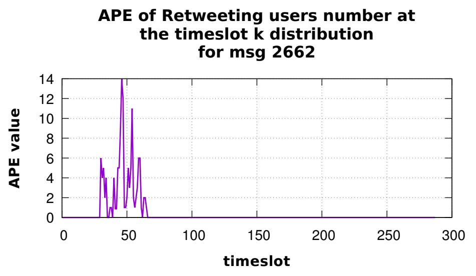


Figure D.687. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 2662.

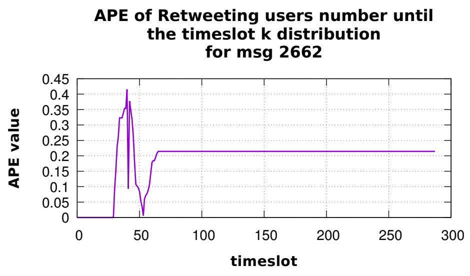


Figure D.688. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 2662.

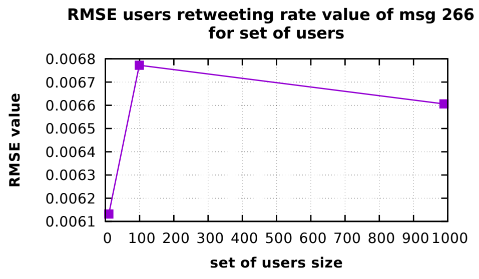


Figure D.689. Graphic showing the RMSE values distribution with respect to the set of users size, for message 2662.

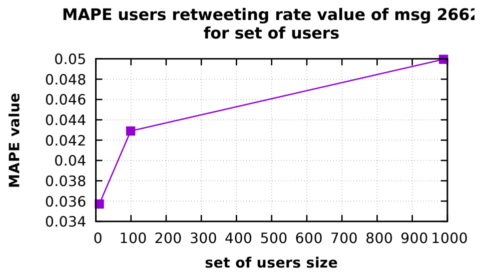


Figure D.690. Graphic showing the MAPE values distribution with respect to the set of users size, for message 2662.

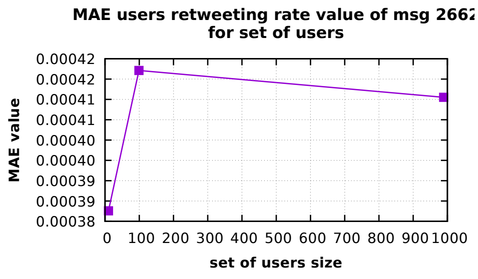


Figure D.691. Graphic showing the MAE values distribution with respect to the set of users size, for message 2662.

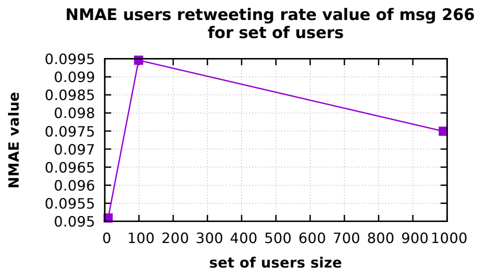


Figure D.692. Graphic showing the NMAE values distribution with respect to the set of users size, for message 2662.

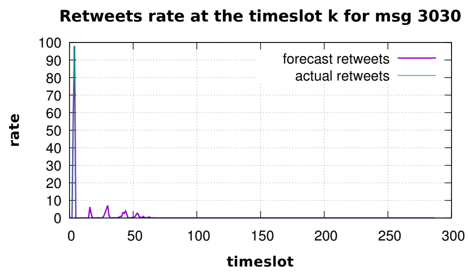


Figure D.693. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 3030.

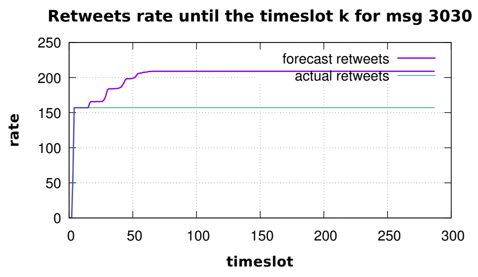


Figure D.694. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 3030.

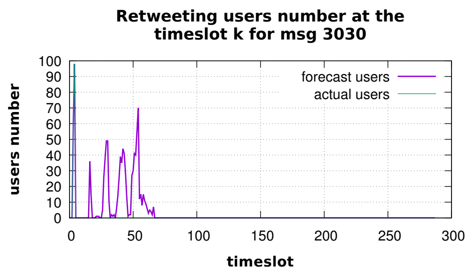


Figure D.695. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 3030.

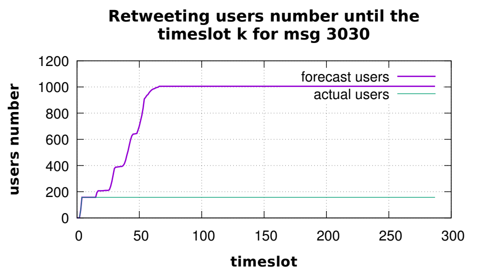


Figure D.696. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 3030.

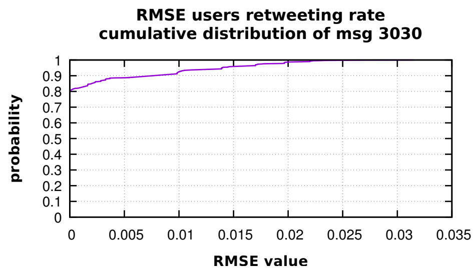


Figure D.697. Graphic showing the RMSE cumulative distribution, for message 3030.

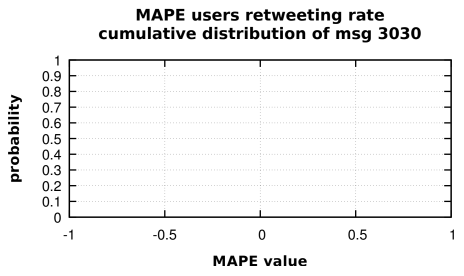


Figure D.698. Graphic showing the MAPE cumulative distribution, for message 3030.

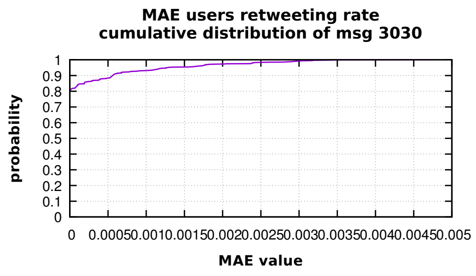


Figure D.699. Graphic showing the MAE cumulative distribution, for message 3030.

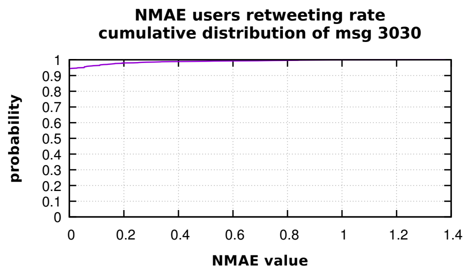


Figure D.700. Graphic showing the NMAE cumulative distribution, for message 3030.

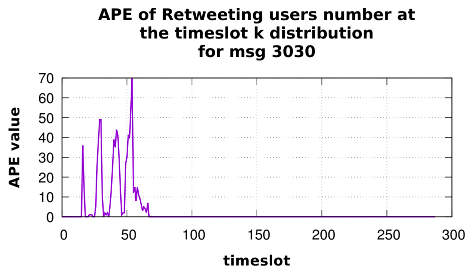


Figure D.701. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 3030.

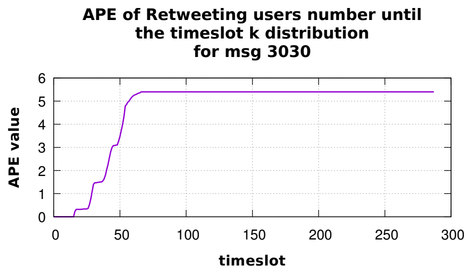


Figure D.702. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 3030.

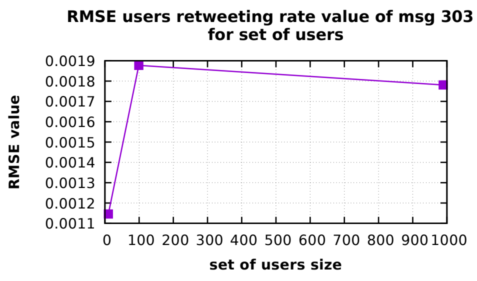


Figure D.703. Graphic showing the RMSE values distribution with respect to the set of users size, for message 3030.

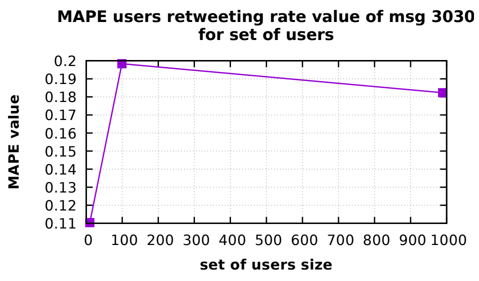


Figure D.704. Graphic showing the MAPE values distribution with respect to the set of users size, for message 3030.

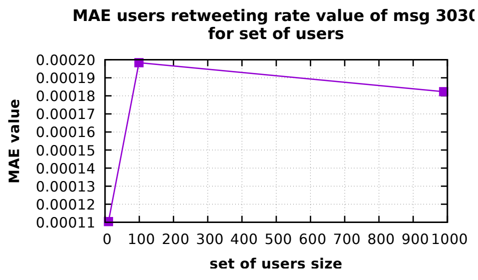


Figure D.705. Graphic showing the MAE values distribution with respect to the set of users size, for message 3030.

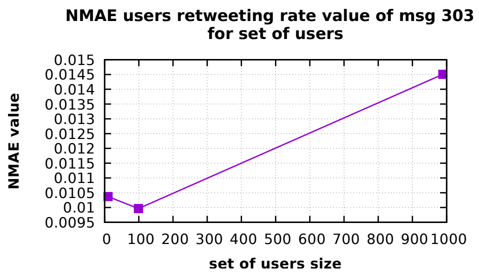


Figure D.706. Graphic showing the NMAE values distribution with respect to the set of users size, for message 3030.

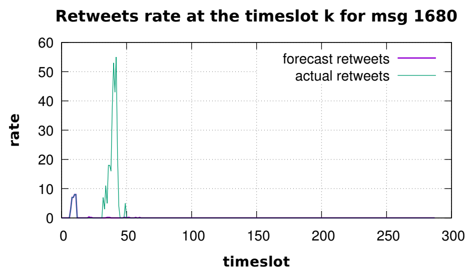


Figure D.707. Graphic showing the forecast and the observed retweets number, at the time slot k, for message 1680.

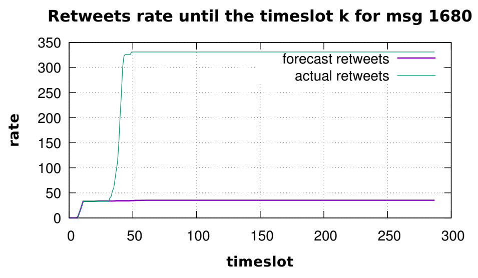


Figure D.708. Graphic showing the forecast and the observed retweets number, until the time slot k, for message 1680.

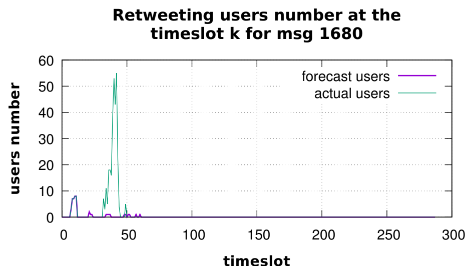


Figure D.709. Graphic showing the forecast and the observed retweeting users number, at the time slot k, for message 1680.

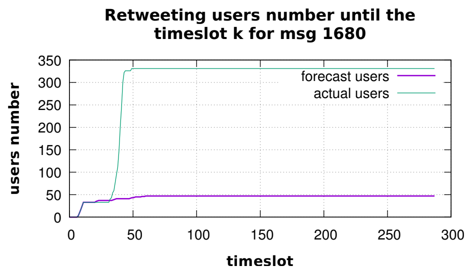


Figure D.710. Graphic showing the forecast and the observed retweeting users number, until the time slot k, for message 1680.

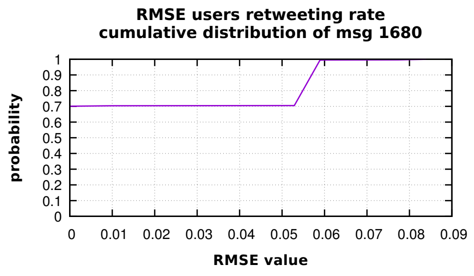


Figure D.711. Graphic showing the RMSE cumulative distribution, for message 1680.

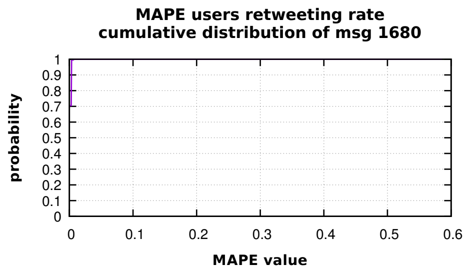


Figure D.712. Graphic showing the MAPE cumulative distribution, for message 1680.

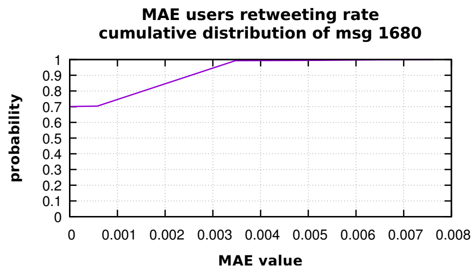


Figure D.713. Graphic showing the MAE cumulative distribution, for message 1680.

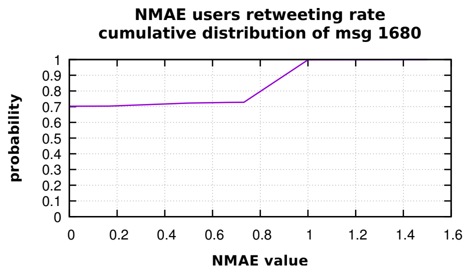


Figure D.714. Graphic showing the NMAE cumulative distribution, for message 1680.

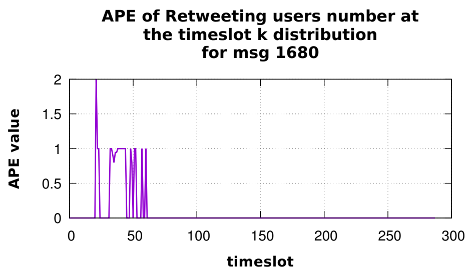


Figure D.715. Graphic showing the distribution of APE on the retweeting users number at the time slot k, for message 1680.

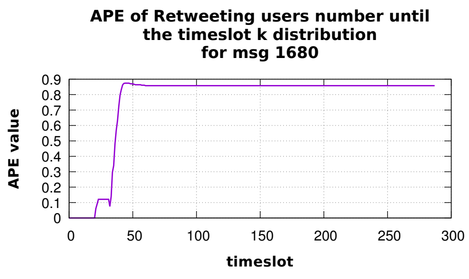


Figure D.716. Graphic showing the distribution of APE on the retweeting users number until time slot k, for message 1680.

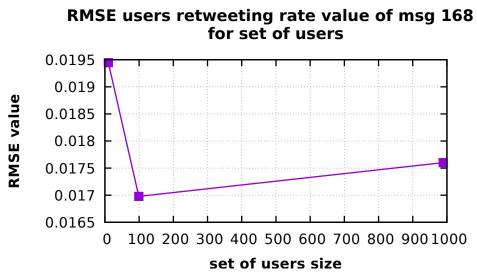


Figure D.717. Graphic showing the RMSE values distribution with respect to the set of users size, for message 1680.

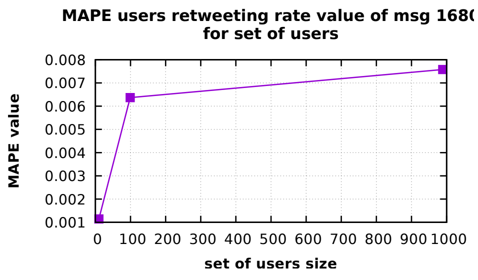


Figure D.718. Graphic showing the MAPE values distribution with respect to the set of users size, for message 1680.

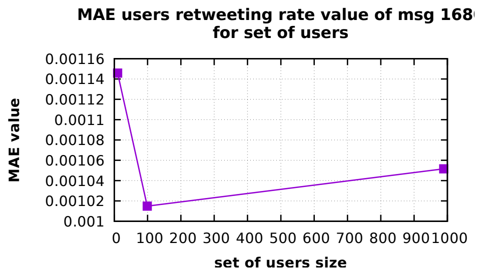


Figure D.719. Graphic showing the MAE values distribution with respect to the set of users size, for message 1680.

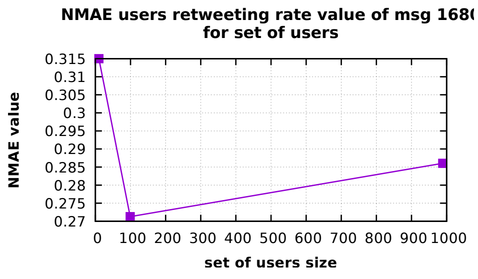


Figure D.720. Graphic showing the NMAE values distribution with respect to the set of users size, for message 1680.

Bibliography

[1] H. Abbas, G. Fainekos, S. Sankaranarayanan, F. Ivančić, and A. Gupta. Probabilistic temporal logic falsification of cyber-physical systems. *ACM Transactions on Embedded Computing Systems*, 12(2s):95:1–95:30, 2013.

[2] G. Agha, J. Meseguer, and K. Sen. Pmaude: Rewrite-based specification language for probabilistic object systems. In *Proceedings of 3rd Workshop on Quantitative Aspects of Programming Languages (QAPL 2005)*. Elsevier, 2005.

[3] G. Agha and K. Palmskog. A survey of statistical model checking. *ACM Transactions on Modeling and Computer Simulation*, 28(1):6:1–6:39, 2018.

[4] M.A. Al-garadi, K.D. Varathan, and S.D. Ravana. Identification of influential spreaders in online social networks using interaction weighted k-core decomposition method. *Physica A: Statistical Mechanics and its Applications*, 468:278–288, 2017.

[5] M. AlTurki and J. Meseguer. PVeStA: A parallel statistical model checking and quantitative analysis tool. In *Proceedings of 4th International Conference on Algebra and Coalgebra in Computer Science (CALCO 2011)*, volume 6859 of *Lecture Notes in Computer Science*, pages 386–392. Springer, 2011.

[6] R. Alur. *Principles of Cyber-Physical Systems.* MIT Press, 2015.

[7] R. Alur, T. Feder, and T.A. Henzinger. The benefits of relaxing punctuality. *Journal of the ACM*, 43(1):116–146, 1996.

[8] E. G. Amparore, M. Beccuti, and S. Donatelli. (Stochastic) model checking in GreatSPN. In *Proceedings of Applications and Theory of Petri Nets and Concurrency (PETRI NETS 2014)*, volume 8489 of *Lecture Notes in Computer Science*, pages 354–363. Springer, 2014.

[9] Y.S.R. Annpureddy, C. Liu, G. E. Fainekos, and S. Sankaranarayanan. S-TaLiRo: A tool for temporal logic falsification for hybrid systems. In *Proceedings of 17th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2011)*, volume 6605 of *Lecture Notes in Computer Science*, pages 254–257. Springer, 2011.

[10] R.B. Ash. *Basic Probability Theory.* John Wiley & Sons, 1970.

[11] C. Baier. On algorithmic verification methods for probabilistic systems, 1998.

[12] C. Baier, B.R. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continous-time markov chains. *IEEE Transactions on Software Engineering*, 29:524–541, 07 2003.

[13] C. Baier and J.-P. Katoen. *Principles of Model Checking (Representation and Mind Series)*. MIT Press, 2008.

[14] M. Bakir, M. Gheorghe, S. Konur, and M. Stannett. Comparative analysis of statistical model checking tools. In *Proceedings of Membrane Computing: 17th International Conference (CMC 2016)*, 2016.

[15] P. Ballarini, B. Barbot, M. Duflot, S. Haddad, and N. Pekergin. HASL: a new approach for performance evaluation and model checking from concepts to experimentation. *Performance Evaluation*, 90:53–77, 2015.

[16] A. Bauer, M. Leucker, and C. Schallhart. Runtime verification for LTL and TLTL. *ACM Transactions on Software Engineering and Methodology*, 20(4), 2011.

[17] J. Benesty, J. Chen, Y. Huang, and I. Cohen. *Noise Reduction in Speech Processing*, volume 2. Springer, 2009.

[18] J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, and W. Yi. UPPAAL — a tool suite for automatic verification of real-time systems. In *Proceedings of Hybrid Systems III: Verification and Control*, volume 1066 of *Lecture Notes in Computer Science*, pages 232–243. Springer, 1996.

[19] A.K. Bhowmick, M. Gueuning, J. Delvenne, R. Lambiotte, and B. Mitra. Temporal sequence of retweets help to detect influential nodes in social networks. *IEEE Trans. Comp. Soc. Sys.*, 6(3):441–455, 2019.

[20] P. Billingsley. *Probability and Mesaure (3rd Ed.)*. John Wiley & Sons, 1995.

[21] J. Bogdoll, A. Hartmanns, and H. Hermanns. Simulation and statistical model checking for modestly nondeterministic models. In *Proceedings of Measurement Modelling and Evaluation of Computing Systems and Dependability and Fault Tolerance (MMB&DFT 2012)*, Lecture Notes in Computer Science, pages 249–252. Springer, 2012.

[22] L. Bortolussi, D. Milios, and G. Sanguinetti. Smoothed model checking for uncertain continuous-time markov chains. *Information and Computation*, 247:235–253, 2016.

[23] B. Boyer, K. Corre, A. Legay, and S. Sedwards. PLASMA-lab: A flexible, distributable statistical model checking library. In *Proceedings of 10th International Conference on Quantitative Evaluation of Systems (QEST 2013)*, pages 160–164. Springer, 2013.

[24] Davide Bresolin, P. Collins, L. Geretti, R. Segala, T. Villa, and S.Ž. Gonzalez. A computable and compositional semantics for hybrid automata. In *Proceedings of 23rd International Conference on Hybrid Systems: Computation and Control (HSCC 2020)*. ACM, 2020.

[25] F.E. Cellier and E. Kofman. *Continuous System Simulation.* Springer, 2010.

[26] E.M. Clarke, T.A. Henzinger, and H. Veith. *Handbook of Model Checking.* Springer, 2016.

[27] E.M. Clarke and J. M. Wing. Formal methods: state of the art and future directions. *Computer Surveys (CSUR)*, 28(4):626–643, 1996.

[28] E.M. Clarke and P. Zuliani. Statistical model checking for cyber-physical systems. In *Proceedings of 9th International Symposium on Automated Technology for Verification and Analysis (ATVA 2011)*, volume 11, pages 1–12. Springer, 2011.

[29] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms.* MIT Press, 2001.

[30] P. Dagum, R.M. Karp, M. Luby, and S. M. Ross. An optimal algorithm for Monte Carlo estimation. *SIAM Journal on Computing*, 29(5):1484–1496, 2000.

[31] A. David, D. Du, K.G. Larsen, A. Legay, M. Mikučionis, D.B. Poulsen, and S. Sedwards. Statistical model checking for stochastic hybrid systems. *Electronic Proceedings in Theoretical Computer Science*, 92:122–136, 2012.

[32] A. David, K.G. Larsen, A. Legay, M. Mikučionis, and D.B. Poulsen. Uppaal smc tutorial. *International Journal on Software Tools for Technology Transfer*, 17(4):397–415, 2015.

[33] R. De Nicola, G. L. Ferrari, and R. Pugliese. Klaim: a kernel language for agents interaction and mobility. *IEEE Transactions on Software Engineering*, 24(5):315–330, 1998.

[34] R. De Nicola, J.P. Katoen, D. Latella, M. Loreti, and M. Massink. Model checking mobile stochastic logic. *Theoretical Computer Science*, 382(1):42–70, 2007.

[35] E. De Rubeis, J. L. Wylie, D. W. Cameron, R. C. Nair, and A. M. Jolly. Combining social network analysis and cluster analysis to identify sexual network types. *International Journal of STD & AIDS*, 18(11):754–759, 2007.

[36] L. Devroye. *Non-Uniform Random Variate Generation.* Springer, 1986.

[37] S. Donatelli, S. Haddad, and J. Sproston. Model checking timed and stochastic properties with CSL^{TA}. *IEEE Transactions on Software Engineering*, 35(2):224–240, 2009.

[38] Dymola. http://www.claytex.com/products/dymola/.

[39] A.A. Ezzat, M. Jun, and Y. Ding. Spatio-temporal asymmetry of local wind fields and its impact on short-term wind forecasting. *IEEE Transactions on Sustainable Energy*, 9(3):1437–1447, 2018.

[40] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. SpaceEx: Scalable verification of hybrid systems. In *Proceedings of 23rd International Conference on Computer Aided Verification (CAV 2011)*, volume 6806 of *Lecture Notes in Computer Science*, pages 379–395. Springer, 2011.

[41] F. Fröhlich, B. Kaltenbacher, F. J. Theis, and J. Hasenauer. Scalable parameter estimation for genome-scale biochemical reaction networks. *PLoS Computational Biology*, 13(1), 2017.

[42] M. Galassi, J. Davis, J. Theiler, B. Gough, G. Jungman, P. Alken, M. Booth, and F. Rossi. *GNU Scientific Library Reference Manual (3rd Ed.)*. Network Theory Limited, 2009.

[43] S. Gao, J. Ma, and Z. Chen. Modeling and predicting retweeting dynamics on microblogging platforms. In *Proceedings of Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (WSDM 2015)*, pages 107–116. ACM, 2015.

[44] J.P. Gleeson, K.P. O'Sullivan, R.A. Baños, and Y. Moreno. Effects of network structure, competition and memory time on social spreading phenomena. *Phys. Rev. X*, 6, 2016.

[45] D. Goldenberg, A. Sela, and E. Shmueli. Timing matters: Influence maximization in social networks through scheduled seeding. *IEEE Trans. Comp. Soc. Sys.*, 5(3):621–638, 2018.

[46] R. Grosu and S.A. Smolka. Quantitative model checking. In *Proceedings of 1st International Symposium on Leveraging Applications of Formal Method (ISoLA 2004)*, pages 165–174, 2004.

[47] R. Grosu and S.A. Smolka. Monte Carlo model checking. In *Proceedings of 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2005)*, volume 3440 of *Lecture Notes in Computer Science*, pages 271–286. Springer, 2005.

[48] G. Guirado, T. Hérault, R. Lassaigne, and S. Peyronnet. Distribution, approximation and probabilistic model checking. In *Proceedings of 4th International Workshop on Parallel and Distributed Methods in Verification (PDMC 2005)*, volume 135, pages 19–30. Elsevier, 2006.

[49] S. Hanneke, W. Fu, and E.P. Xing. Discrete temporal models of social networks. *Electronic Journal of Statistics*, 4:585–605, 2010.

[50] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[51] B.P. Hayes, I. Melatti, T. Mancini, M. Prodanovic, and E. Tronci. Residential demand management using individualised demand aware price policies. *IEEE Transactions on Smart Grid*, 8(3), 2017.

[52] M. Heiner, C. Rohr, and M. Schwarick. Marcie – model checking and reachability analysis done efficiently. In *Proceedings of Applications and Theory of Petri Nets and Concurrency (PETRI NETS 2013)*, volume 7927 of *Lecture Notes in Computer Science*, pages 389–399. Springer, 2013.

[53] D. Henriques, J.G. Martins, P. Zuliani, A. Platzer, and E.M. Clarke. Statistical model checking for markov decision processes. In *2012 Ninth International Conference on Quantitative Evaluation of Systems*, pages 84–93. IEEE, 2012.

[54] T. Hérault, R. Lassaigne, F. Magniette, and S. Peyronnet. Approximate probabilistic model checking. In *Proceedings of 5th International Workshop on Verification, Model Checking, and Abstract Interpretation (VMCAI 2004)*, pages 73–84. Springer, 2004.

[55] N.O. Hodas and K. Lerman. The simple rules of social contagion. *Scientific Reports*, 4(4343), 2014.

[56] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, pages 13–30, 1963.

[57] R.V. Hogg, J. W. McKean, and A. T. Craig. *Introduction to Mathematical Statistics*. Pearson Education, eighth edition, 2018.

[58] S. Huang, T. Lv, X. Zhang, Y. Yang, W. Zheng, and C. Wen. Identifying node role in social network based on multiple indicators. *PLoS ONE*, 9(8):1–16, 08 2014.

[59] M. Huber and B. Jones. Faster estimates of the mean of bounded random variables. *Mathematics and Computers in Simulation*, 161:93–101, 2019.

[60] N. Jagiella, D. Rickert, F.J. Theis, and J. Hasenauer. Parallelization and high-performance computing enables automated statistical inference of multi-scale models. *Cell Systems*, 4(2):194–206, 2017.

[61] C. Jegourel, A. Legay, and S. Sedwards. A platform for high performance statistical model checking–plasma. In *Proceedings of 18th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2012)*, volume 7214 of *Lecture Notes in Computer Science*, pages 498–503. Springer, 2012.

[62] C. Jegourel, A. Legay, and S. Sedwards. Importance splitting for statistical model checking rare properties. In *Proceedings of 25th International Conference on Computer Aided Verification (CAV 2013)*, volume 8044 of *Lecture Notes in Computer Science*, pages 576–591. Springer, 2013.

[63] C. Jegourel, A. Legay, and S. Sedwards. Command-based importance sampling for statistical model checking. *Theoretical Computer Science*, 649:1–24, 2016.

[64] S.K. Jha, E.M. Clarke, C.J. Langmead, A. Legay, A. Platzer, and P. Zuliani. A bayesian approach to model checking biological systems. In *Proceedings of*

7th International Conference on Computational Methods in Systems Biology (CMSB 2009), volume 5688 of *Lecture Notes in Computer Science*, pages 218–234. Springer, 2009.

[65] L. Jin, Y. Chen, T. Wang, P. Hui, and A.V. Vasilakos. Understanding user behavior in online social networks: a survey. *IEEE Communications Magazine*, 51(9):144–150, 2013.

[66] N.L. Johnson, S. Kots, and N. Balakrishnan. *Continuous Univariate Distributions, Volume 2.* John Wiley & Sons, 2 edition, 1995.

[67] K. Kalajdzic, C. Jégourel, A. Lukina, E. Bartocci, A. Legay, S.A. Smolka, and R. Grosu. Feedback control for statistical model checking of cyber-physical systems. In *Proceedings of 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques (ISoLA 2016)*, volume 9952 of *Lecture Notes in Computer Science*, pages 46–61. Springer, 2016.

[68] J. P. Katoen, I. S. Zapreev, E. M. Hahn, H. Hermanns, and D. N. Jansen. The ins and outs of the probabilistic model checker MRMC. *Performance Evaluation*, 68(2):90–104, 2011.

[69] J.P. Katoen. The probabilistic model checking landscape. In *Proceedings of 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2016)*, pages 31–45. Ass. Comp. Mach., 2016.

[70] L. Khansa, C. W. Zobel, and G. Goicochea. Creating a taxonomy for mobile commerce innovations using social network and cluster analyses. *International Journal of Electronic Commerce*, 16(4):19–52, 2012.

[71] R. Kobayashi and R. Lambiotte. TiDeH: time-dependent hawkes process for predicting retweet dynamics. In *Proceedings of 10th International AAAI Conference on Weblogs and Social Media (ICWSM 2016).* AAAI, 2016.

[72] S. Kots, N. Balakrishnan, and N.L. Johnson. *Continuous Multivariate Distributions, Volume 1: Models and Applications.* John Wiley & Sons, 2 edition, 2004.

[73] M. Kwiatkowska, G. Norman, and D. Parker. Prism 4.0: Verification of probabilistic real-time systems. In *Proceedings of 23rd International Conference on Computer Aided Verification (CAV 2011)*, volume 6806 of *Lecture Notes in Computer Science*, pages 585–591. Springer, 2011.

[74] R. Lassaigne and S. Peyronnet. Probabilistic verification and approximation. *Annals of Pure and Applied Logic*, 152(1):122–131, 2008.

[75] A. Legay, B. Delahaye, and S. Bensalem. Statistical model checking: An overview. In *Proceedings of Runtime Verification, First International Conference, RV 2010, St. Julians, Malta, November 2010. Proceedings*, volume 6418 of *Lecture Notes in Computer Science*, pages 122–135. Springer, 2010.

[76] A. Legay, A. Lukina, L.M. Traonouez, J. Yang, S.A. Smolka, and R. Grosu. Statistical model checking. *Computing and Software Science: State of the Art and Perspectives*, pages 478–504, 2019.

[77] A. Legay, S. Sedwards, and L.M. Traonouez. Scalable verification of markov decision processes. In *Proceedings of Software Engineering and Formal Methods*, pages 350–362. Springer, 2015.

[78] K. Luckow, C.S. Păsăreanu, M.B. Dwyer, A. Filieri, and W. Visser. Exact and approximate probabilistic symbolic execution for nondeterministic programs. In *Proceedings of 29th ACM/IEEE International Conference on Automated Software Engineering (ASE 2014)*. Ass. Comp. Mach., 2014.

[79] S. Ma, L. Feng, and C.H. Lai. Mechanistic modelling of viral spreading on empirical social network and popularity prediction. *Scientific Reports*, 2018.

[80] T. Mancini, F. Mari, A. Massini, I. Melatti, I. Salvo, S. Sinisi, E. Tronci, R. Ehrig, S. Röblitz, and B. Leeners. Computing personalised treatments through in silico clinical trials. A case study on downregulation in assisted reproduction. In *Proceedings of 25th RCRA International Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (RCRA 2018)*, volume 2271 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018.

[81] T. Mancini, F. Mari, A. Massini, I. Melatti, and E. Tronci. SyLVaaS: System level formal verification as a service. In *Proceedings of 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP 2015)*, pages 476–483. IEEE, 2015.

[82] T. Mancini, F. Mari, I. Melatti, I. Salvo, E. Tronci, J. Gruber, B. Hayes, M. Prodanovic, and L. Elmegaard. Demand-aware price policy synthesis and verification services for smart grids. In *Proceedings of 2014 IEEE International Conference on Smart Grid Communications (SmartGridComm 2014)*, pages 794–799. IEEE, 2014.

[83] T. Mancini, F. Mari, I. Melatti, I. Salvo, E. Tronci, J.K. Gruber, B. Hayes, and L. Elmegaard. Parallel statistical model checking for safety verification in smart grids. In *Proceedings of 2018 IEEE International Conference on Smart Grid Communications (SmartGridComm 2018)*. IEEE, 2018.

[84] T. Mancini, F. Mari, I. Melatti, I. Salvo, E. Tronci, J.K. Gruber, B. Hayes, M. Prodanovic, and L. Elmegaard. User flexibility aware price policy synthesis for smart grids. In *Proceedings of 18th Euromicro Conference on Digital System Design (DSD 2015)*, pages 478–485. IEEE, 2015.

[85] T. Mancini, E. Tronci, I. Salvo, F. Mari, A. Massini, and I. Melatti. Computing biological model parameters by parallel statistical model checking. In *Proceedings of 3rd International Conference on Bioinformatics and Biomedical Engineering (IWBBIO 2015)*, volume 9044 of *Lecture Notes in Computer Science*, pages 542–554. Springer, 2015.

[86] K.L. McMillan. *The SMV System*, pages 61–85. Springer, 1993.

[87] B.L. Mediouni, A. Nouri, M. Bozga, M. Dellabani, A. Legay, and S. Bensalem. Sbip 2.0: Statistical model checking stochastic real-time systems. In *Proceedings of 16th International Symposium on Automated Technology for Verification and Analysis (ATVA 2018)*, pages 536–542. Springer, 2018.

[88] MODEST. http://www.modestchecker.net.

[89] S.A. Myers, A. Sharma, S. Gupta, and J. Lin. Information network or social network? the structure of the twitter follow. In *Proceedings of 23rd International Conference on World Wide Web (WWW 2014)*, pages 493–498. ACM, 2014.

[90] G. Norman, D. Parker, and J. Sproston. Model checking for probabilistic timed automata. *Formal Methods in System Design*, 43(2):164–190, 2013.

[91] A. Nouri, B. Mediouni, M. Bozga, J. Combaz, S. Bensalem, and A. Legay. Performance Evaluation of Stochastic Real-Time Systems with the SBIP Framework. *International Journal of Critical Computer-Based Systems*, pages 1–33, 2018.

[92] OpenModelica. http://www.openmodelica.org.

[93] A. Pappagallo, A. Massini, and E. Tronci. An Efficient Algorithm for Multivariate Monte Carlo Estimation. *submitted for publication*, 2020.

[94] A. Pappagallo, A. Massini, and E. Tronci. Monte Carlo based Statistical Model Checking of Cyber-Physical Systems: a Review. *Information*, 11(12):588, 2020.

[95] D. Parker, G. Norman, and M. Kwiatkowska. PRISM 2017. Statistical Model Checker. https://www.prismmodelchecker.org/manual/RunningPRISM/StatisticalModelChecking.

[96] S. Petrovic, M. Osborne, and V. Lavrenko. RT to Win! predicting message propagation in twitter. In *Proceedings of 5th International AAAI Conference on Weblogs and Social Media (ICWSM 2011)*. AAAI, 2011.

[97] S. Peyronnet, R. Lassaigne, and T. Herault. Apmc 3.0: Approximate verification of discrete and continuous time markov chains. In *Proceedings of QEST 2006 - Proceedings Third International Conference on the Quantitative Evaluation of SysTems*, pages 129–130, 2006.

[98] S. Pinisetty, T. Jéron, S. Tripakis, Y. Falcone, H. Marchand, and V. Preoteasa. Predictive runtime verification of timed properties. *Journal of Systems and Software*, 132:353–365, 2017.

[99] Plasma Lab. https://project.inria.fr/plasma-lab/.

[100] M.L. Puterman. *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2005.

[101] D. Reijsbergen, W. de Boer, P.T.and Scheinhardt, and B. Haverkort. On hypothesis testing for statistical model checking. *International Journal on Software Tools for Technology Transfer*, 17(4):377–395, 2015.

[102] B.D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, 2007.

[103] K.Y. Rozier. Linear temporal logic symbolic model checking. *Computer Science Review*, 5(2):163–203, 2011.

[104] L. Schmiester, Y. Schälte, F. Fröhlich, J. Hasenauer, and D. Weindl. Efficient parameterization of large-scale dynamic models based on relative measurements. *Bioinformatics*, 36(2):594–602, 07 2019.

[105] L. Schmiester, D. Weindl, and J. Hasenauer. Statistical inference of mechanistic models from qualitative data using an efficient optimal scaling approach. *BioRxiv*, 12 2019.

[106] T.B. Schön, F. Lindsten, J. Dahlin, J. Wågberg, C.A. Naesseth, A. Svensson, and L. Dai. Sequential monte carlo methods for system identification. In *Proceedings of 17th IFAC Symposium on System Identification (SYSID 2015)*. Elsevier, 2016.

[107] S. Sebastio and A. Vandin. MultiVeStA: Statistical model checking for discrete event simulators. In *Proceedings of 7th International Conference on Performance Evaluation Methodologies and Tools (ValueTools 2013)*, pages 310–315. ICST/ACM, 2013.

[108] K. Sen, M. Viswanathan, and G. Agha. "statistical model checking of black-box probabilistic systems". In *Proceedings of 16th International Conference on Computer Aided Verification (CAV 2004)*, volume 3114 of *Lecture Notes in Computer Science*, pages 202–213. Springer, 2004.

[109] K. Sen, M. Viswanathan, and G. Agha. On statistical model checking of stochastic systems. In *Proceedings of 17th International Conference on Computer Aided Verification (CAV 2005)*, volume 3576 of *Lecture Notes in Computer Science*, pages 266–280. Springer, 2005.

[110] K. Sen, M. Viswanathan, and G. Agha. Vesta: A statistical model-checker and analyzer for probabilistic systems. In *Proceedings of QEST 2005 - Proceedings Second International Conference on the Quantitative Evaluation of SysTems*, volume 2005, pages 251–252, 2005.

[111] M.V. Shcherbakov, A. Brebels, N.L. Shcherbakova, A.P. Tyukov, T.A. Janovsky, and V.A.E. Kamaev. A survey of forecast error measures. *World Applied Sciences Journal*, 24:171–176, 2013.

[112] F. Shmarov and P. Zuliani. Probabilistic hybrid systems verification via smt and monte carlo techniques. In *Proceedings of Hardware and Software: Verification and Testing, 12nd International Haifa Verification Conference (HVC 2016)*, volume 10028 of *Lecture Notes in Computer Science*. Springer, 2016.

[113] SimulationX. http://www.simulationx.com.

[114] Simulink. http://www.mathworks.com.

[115] S. Sinisi, V. Alimguzhin, T. Mancini, E. Tronci, and B. Leeners. Complete populations of virtual patients for in silico clinical trials. *Bioinformatics*, 2020. To appear.

[116] S. Sinisi, V. Alimguzhin, T. Mancini, E. Tronci, F. Mari, and B. Leeners. Optimal personalised treatment computation through in silico clinical trials on patient digital twins. *Fundamenta Informaticae*, 174(3–4):283–310, 2020.

[117] E.D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems (2nd Ed.)*. Springer, 1998.

[118] M.H. ter Beek, A. Legay, A.L. Lafuente, and A. Vandin. Statistical model checking for product lines. In *Proceedings of 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques (ISoLA 2016)*, volume 9952 of *Lecture Notes in Computer Science*. Springer, 2016.

[119] P. Thati and G. Roşu. Monitoring algorithms for metric temporal logic specifications. In *Proceedings of Runtime Verification, Fourth Workshop on Runtime Verification 2004, RV 2004, Barcelona, Spain, April 2004. Proceedings*, volume 113 of *Electronic Notes in Theoretical Computer Science*, pages 145–162. Elsevier, 2004.

[120] N. Tracheva and S. Ukhinov. A new monte carlo method for estimation of time asymptotic parameters of polarized radiation. *Mathematics and Computers in Simulation*, 161:84–92, 2019.

[121] E. Tronci, T. Mancini, I. Salvo, S. Sinisi, F. Mari, I. Melatti, A. Massini, F. Davi', T. Dierkes, R. Ehrig, S. Röblitz, B. Leeners, T.H.C. Krüger, M. Egli, and F. Ille. Patient-specific models from inter-patient biological models and clinical records. In *Proceedings of 14th International Conference on Formal Methods in Computer-Aided Design (FMCAD 2014)*, pages 207–214. IEEE, 2014.

[122] S. Uda. Application of information theory in systems biology. *Biophysical Reviews*, 12:377–384, 2020.

[123] S. van Mourik, C. ter Braak, H. Stigter, and J. Molenaar. Prediction uncertainty assessment of a systems biology model requires a sample of the full probability distribution of its parameters. *Peer J Life and Environment*, 2, 2014.

[124] Verimag. BIP Component Framework. http://www-verimag.imag.fr/Rigorous-Design-of-Component-Based.html.

[125] A. Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.

[126] W. Whitt. Continuity of generalized Semi-Markov processes. *Mathematics of Operations Research*, 5(4):494–501, 1980.

[127] Wolfram Research, Inc. SystemModeler. http://www.wolfram.com/system-modeler.

[128] B. Xue, M. Fränzle, H. Zhao, N. Zhan, and A. Easwaran. Probably approximate safety verification of hybrid dynamical systems. In *Proceedings of Formal Methods and Software Engineering - 21st International Conference on Formal Engineering Methods (ICFEM 2019)*, volume 11852 of *Lecture Notes in Computer Science*. Springer, 2019.

[129] B. Xue, Y. Liu, L. Ma, X. Zhang, M. Sun, and X. Xie. Safe inputs approximation for black-box systems. In *Proceedings of 24th International Conference on Engineering of Complex Computer Systems (ICECCS 2019)*, pages 180–189. IEEE, 2019.

[130] H.L.S. Younes. Verification and planning for stochastic processes with asynchronous events, 2005. PhD Thesis.

[131] H.L.S. Younes. Ymer: A statistical model checker. In *Proceedings of 17th International Conference on Computer Aided Verification (CAV 2005)*, volume 3576 of *Lecture Notes in Computer Science*, pages 429–433. Springer, 2005.

[132] H.L.S. Younes, E.M. Clarke, and P. Zuliani. Statistical verification of probabilistic properties with unbounded until. In *Proceedings of 13th Brazilian Symposium on Formal Methods (SBMF 2010)*, volume 6527 of *Lecture Notes in Computer Science*. Springer, 2010.

[133] H.L.S. Younes, M.Z. Kwiatkowska, G. Norman, and D. Parker. Numerical vs. statistical probabilistic model checking. *International Journal on Software Tools for Technology Transfer*, 8(3):216–228, 2006.

[134] H.L.S. Younes and R.G. Simmons. Probabilistic verification of discrete event systems using acceptance sampling. In *Proceedings of 14th International Conference on Computer Aided Verification (CAV 2002)*, volume 2404 of *Lecture Notes in Computer Science*, pages 223–235. Springer, 2002.

[135] T.R. Zaman, R. Herbrich, J. van Gael, and D. Stern. Predicting information spreading in twitter. In *Proceedings of Advances in Neural Information Processing Systems 23 (NIPS 2010)*, 2010.

[136] Q. Zhao, M.A. Erdogdu, H.Y. He, A. Rajaraman, and J. Leskovec. SEISMIC: a self-exciting point process model for predicting tweet popularity. In *Proceedings of 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2015)*, pages 1513–1522. ACM, 2015.

[137] F. Zhou, L. Chen, Y. Wu, J. Ding, J. Zhao, and Y. Zhang. Mworks : a modern ide for modeling and simulation of multi-domain physical systems based on modelica. In *Proceedings of 5th International Modelica Conference (Modelica 2006)*, 2006.

[138] JR Zipkin, FP Schoenberg, K. Coronges, and AL. Bertozzi. Point-process models of social network interactions: Parameter estimation and missing data recovery. *European Journal of Applied Mathematics*, 27, 2016.

[139] P. Zuliani, A. Platzer, and E.M. Clarke. Bayesian statistical model checking with application to Stateflow/Simulink verification. *Formal Methods in System Design*, 43(2):338–367, 2013.