MASTER EN SCIENCES INFORMATIQUES

Prédiction de la taille des cascades dans les réseaux sociaux

D'Harveng, Jérôme

Award date:
2016

Awarding institution:
Universite de Namur

Université de Namur,

Faculté d'informatique

Année académique 2015-2016

Prédiction de la taille de Cascades dans les réseaux sociaux

Nom étudiant : Jérôme D'HARVENG

Promoteur : Benoît Frenay (signature pour approbation de dépôt)

Co-Promoteur : Renaud Lambiotte

Mémoire présenté en vue de l'obtention du grade de master en Informatique

TABLE DES MATIERES :

1. Introduction

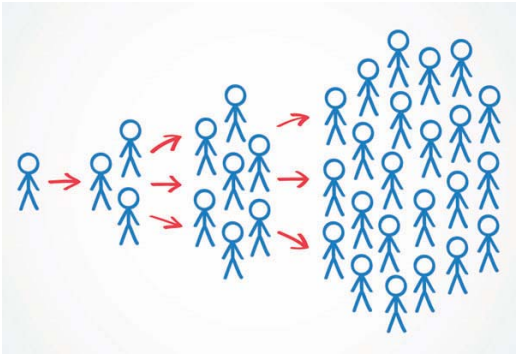
Les réseaux sociaux font de plus en plus partie intégrante de la vie de « Monsieur tout le monde » et ce pas uniquement dans le secteur des loisirs. En effet, beaucoup de professionnels principalement dans le secteur du marketing commencent à s'y intéresser également depuis quelques années. Et la raison à la base en est assez simple, il s'agit du pouvoir qu'ont les réseaux sociaux à transmettre certaines informations dans le temps et dans l'espace (càd géographiquement). Cet intérêt a suscité la curiosité de nombreux chercheurs de par le monde, comme en témoigne le nombre d'articles scientifiques parus ces dernières années sur le sujet. « Comment prévoir quelle communication sera transmise de manière virale ? » .

Dans ce travail nous nous sommes principalement focalisés sur Twitter.

Twitter permet d'envoyer gratuitement de brefs messages, appelés *tweets*, ceux-ci étant limités à 140 caractères. Un comportement intéressant qui est en train d'émerger sur Twitter, est le fait de reposter ou repartager un tweet ayant été écrit par un autre internaute. Un utilisateur trouvant un tweet publié par un autre intéressant peut alors choisir de le partager avec ses propres « *followers* ». Dans certain cas, cela donnera de très longues chaînes de retweets à partir d'un tweet original, donnant lieu à ce qu'on appelle des « cascades » de retweets. L'étude de tweets aboutissant à de telles cascades, a de nombreux intérêts, comme l'identification de « breaking news », la recommandation de message personnalisé, le marketing viral, un meilleur ordennancement du contenu, ... Donc une des questions fondamentales dans la modélisation de ces cascades est la prédiction du nombre de retweets futures. L'idée est alors pour un *tweet* donné de déterminer le nombre final de retweets que celui-ci obtiendra.

Ce travail s'articulera principalement autour de 3 axes, tout d'abord un « état de l'art » sur les différentes méthodes de prédiction, ensuite un modèle particulier, du nom de SEISMIC, sera étudié, et finalement on retournera sur une des dernières recherches en date, constituant une amélioration du modèle SEISMIC.

Comme la plupart des recherches sont publiées en anglais, toute une terminologie anglophone s'est développée dans ce secteur de recherche, et le choix a été fait de garder cette terminologie sans la traduire en français, afin de ne pas perdre son sens primaire.



2. Etat de l'art

L'étude des « cascades d'information » a déjà fait l'objet de nombreuses recherches. Les modèles de recherches étudiées, sont principalement divisées en 2 catégories, avec d'un côté les « Featured based methods » et de l'autre les « Point process based methods ».

a. *Featured based methods*

Pour ces méthodes, l'on recherche dans un premier temps à extraire une liste exhaustive de caractéristiques pertinentes. Celles-ci peuvent être de différents types, tels que des caractéristiques de contenu, des caractéristiques liées à celui qui a posté le tweet original, des caractéristiques spécifiques au réseau ou encore des caractéristiques temporelles. Nous allons maintenant nous intéresser de plus près à certaines études ayant été réalisées dans le domaine.

i. Simple regression :

Ref : Can Cascades be predicted ? J. Cheng, J. Leskovec, L. A. Adamic, P. A Dow, J. Kleinberg

Les auteurs sont partis d'une base de données de photos uploadées sur Facebook en juin 2013 en observant tout « reshare » intervenant dans les 28 jours de l'upload initial. Les cascades sont vues ici comme des objets dynamiques complexes passant lors de leur croissance au travers de différents stades successifs. Dès lors, les cascades ont été suivies au fil du temps, dans une recherche de prédiction du stade suivant. Que signifie : prédire le stade suivant ? Si l'on considère toutes les cascades atteignant au moins une taille k, alors la distribution des tailles de cascades a une valeur médiane $f(k) \geq k$. La manière la plus basique de se questionner sur le stade suivant de croissance de la cascade sachant que sa taille actuelle est k, est de se demander si elle atteindra la taille f(k). Le problème de prédiction peut ainsi être reformulé comme un problème de classification binaire : étant donné une cascade de taille k, est-ce que celle-ci doublera sa taille et atteindra au moins 2k nœuds ? Cette formulation a l'avantage d'étudier comment la popularité d'une cascade évolue au cours des différents stades de croissance. De plus, cela se rapproche de la tâche réelle qui doit être accomplie dans des applications de gestion de contenu viral, où beaucoup de cascades sont monitorées simultanément et où la question est de savoir lesquelles sont susceptibles de croître significativement dans le future. La méthodologie générale d'apprentissage utilisée est de représenter une cascade par une série de caractéristiques et d'ensuite y appliquer différentes méthodes de classifications telle que la régression linéaire. Les facteurs contribuant à la croissance et l'expansion des cascades sont ici divisées en 5 classes : propriétés du contenu, caractéristiques du *poster* original, caractéristiques de ceux qui repartagent l'information, caractéristiques structurales de la cascade ainsi que des caractéristiques temporelles de la cascade.

Caractéristiques du contenu :

Sur Twitter l'on utilise le contenu du tweet et en particulier les hashtags. Parcontre ici partant de descripteurs GIST et de caractéristiques des histogrammes de couleurs des photos uploadées sur Facebook, à l'aide d'algorithmes de d'apprentissage, des scores ont été attribués selon photo intérieure / extérieure, close-up , contenant une personne , de la nourriture, traitant de nature, d'eau ou avec du texte.

Content Features	
$score_{food/nature/...}$	The probability of the photo having a specific feature (food, overlaid text, landmark, nature, etc.)
is_en	Whether the photo was posted by an English-speaking user or page
$has\_caption$	Whether the photo was posted with a caption
$liwc_{pos/neg/soc}$	Proportion of words in the caption that expressed positive or negative emotion, or sociality, if English

Caractéristiques du poster / ceux qui repartagent :

Des études préalables liées à Twitter, ont montré que les caractéristiques de l'auteur du tweet sont plus importantes que le tweet lui-même. Dans beaucoup d'études cherchant à prédire le nombre final de retweets sur Twitter, le nombre de *followers* d'un utilisateur se classe parmi les facteurs les plus importants, sinon le plus important. Ici les auteurs on gardé l'intuition se cachant derrière ces caractéristiques, en définissant tant des caractéristiques démographiques que de réseaux concernant le poster original, ainsi que les utilisateurs repartageant l'image.

	Root (Original Poster) Features
$views_{0,\,k}$	Number of users who saw the original photo until the kth reshare was posted
$orig\_is\_page$	Whether the original poster is a page
$outdeg(v_0)$	Friend, subscriber or fan count of the original poster
age_0	Age of the original poster, if a user
$gender_0$	Gender of the original poster, if a user
fb_age_0	Time since the original poster registered on Facebook, if a user
$activity_0$	Average number of days the original poster was active in the past month, if a user
	Resharer Features
$views_{1..k-1,\,k}$	Number of users who saw the first $k-1$ reshares until the kth reshare was posted
$pages_k$	Number of pages responsible for the first k reshares, including the root, or $\sum_{i=0}^{k} \mathbb{1}\{v_i \text{ is a page}\}$
$friends_k^{avg/90p}$	Average or 90th percentile friend count of the first k resharers, or $\frac{1}{k}\sum_{i=1}^{k} outdeg_{friends}(v_i)\mathbb{1}\{v_i \text{ is a user}\}$
$fans_k^{avg/90p}$	Average or 90th percentile fan count of the first k resharers, or $\frac{1}{k}\sum_{i=1}^{k} outdeg(v_i)\mathbb{1}\{v_i \text{ is a page}\}$
$subscribers_k^{avg/90p}$	Average or 90th percentile subscriber count of the first k resharers, or $\frac{1}{k}\sum_{i=1}^{k} outdeg_{subscriber}(v_i)\mathbb{1}\{v_i \text{ is a user}\}$
$fb\_ages_k^{avg/90p}$	Average or 90th percentile time since the first k resharers registered on Facebook, or $\frac{1}{k}\sum_{i=1}^{k} fb\_age_i$
$activities_k^{avg/90p}$	Average number of days the first k resharers were active in July, or $\frac{1}{k}\sum_{i=1}^{k} activity_i$
$ages_k^{avg/90p}$	Average age of the first k resharers, or $\frac{1}{k}\sum_{i=1}^{k} age_i$
$female_k$	Number of female users among the first k resharers, or $\sum_{i=1}^{k} \mathbb{1}\{gender_i \text{ is female}\}$

Caractéristiques structurales de la cascade :

Le réseau fournit le medium par lequel l'information se propage. Comme illustré par la figure 1 , les auteurs ont généré des caractéristiques à partir du graph reprenant les k repartages ($\hat{G}$), ainsi que le sous-graphe des amis des premiers k repartages (G'). Alors que $\hat{G}$ représente la propagation actuelle de la cascade, G' fournit de l'information sur les liens sociaux entre ceux qui ont repartagé initialement.

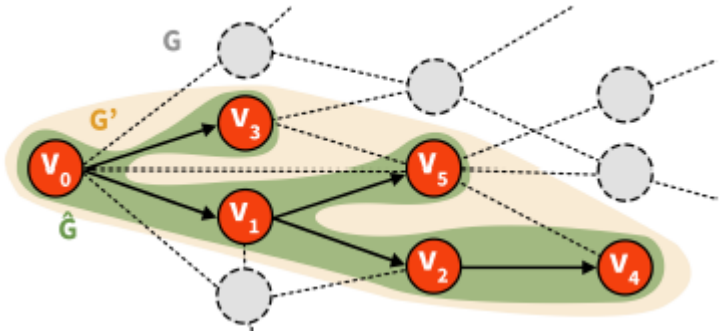


Figure 1

	Structural Features
$outdeg(v_i)$	Connection count (sum of friend, subscriber and fan counts) of the ith resharer (or out-degree of v_i on $G = (V, E)$)
$outdeg(v'_i)$	Out-degree of the ith reshare on the induced subgraph $G' = (V', E')$ of the first k resharers and the root
$outdeg(\hat{v}_i)$	Out-degree of the ith reshare on the reshare graph $\hat{G} = (\hat{V}, \hat{E})$ of the first k reshares
$orig\_connections_k$	Number of first k resharers who are friends with, or fans of the root, or $\lvert\{v_i \mid (v_0, v_i) \in E,\ 1 \le i \le k\}\rvert$
$border\_nodes_k$	Total number of users or pages reachable from the first k resharers and the root, or $\lvert\{v_i \mid (v_i, v_j) \in E,\ 0 \le i, j \le k\}\rvert$
$border\_edges_k$	Total number of first-degree connections of the first k resharers and the root, or $\lvert\{(v_i, v_j) \mid (v_i, v_j) \in E,\ 0 \le i, j \le k\}\rvert$
$subgraph'_k$	Number of edges on the induced subgraph of the first k resharers and the root, or $\lvert\{(v_i, v_j) \mid (v_i, v_j) \in E',\ 0 \le i, j \le k\}\rvert$
$depth'_k$	Change in tree depth of the first k reshares, or $\min_\beta \sum_{i=1}^{k}(depth_i - \beta i)^2$
$depths_k^{avg/90p}$	Average or 90th percentile tree depth of the first k reshares, or $\frac{1}{k}\sum_{i=1}^{k} depth_i$
did_leave	Whether any of the first k reshares are not first-degree connections of the root

Caractéristiques temporelles

Les propriétés reliées à la vitesse de propagation de la cascade sont parmis les plus importantes. De plus, la vitesse de diffusion changeant avec le temps, cela peut avoir un effet important sur la capacité de la cascade à continuer son expansion sur le réseau.

Temporal Features	
$time_i$	Time elapsed between the original post and the ith reshare
$time'_{1..k/2}$	Average time between reshares, for the first $k/2$ reshares, or $\frac{1}{k/2-1}\sum_{i=1}^{k/2-1}(time_{i+1}-time_i)$
$time'_{k/2..k}$	Average time between reshares, for the last $k/2$ reshares, or $\frac{1}{k/2-1}\sum_{i=k/2}^{k-1}(time_{i+1}-time_i)$
$time''_{1..k}$	Change in the time between reshares of the first k reshares, or $\min_\beta \sum_{i=1}^{k-1}(time_{i+1}-time_i)-\beta i)^2$
$views'_{0,k}$	Number of users who saw the original photo, until the kth reshare was posted, per unit time, or $\frac{views_{0,\,k}}{time_k}$
$views'_{1..k-1,\,k}$	Number of users who saw the first $k-1$ reshares, until the kth reshare was posted, per unit time, or $\frac{views_{1..k-1,k}}{time_k}$

Différentes observations intéressantes ont découlé de cette étude, comme le fait que l'importance du *post original* diminue quand *k* augmente, presque toutes les caractéristiques de contenu perdent en importance avec l'augmentation de k, ou encore que les cascades de taille importante reçoivent un grand nombre de vues en peu de temps. Au plus le graphe de la cascade s'étend en profondeur, au plus cette dernière est susceptible d'évoluer sur le long terme. L'importance des caractéristiques temporelles quant à elle, reste stable au cours de la croissance de la cascade.

ii. <u>Arbre de régression (Regression Tree) :</u>

Ref : Everyone's an Influencer: Quantifying Influence on Twitter E. Bakshy, J. M. Hofman, W. A. Mason, D. J. Watts

Le bouche-à-oreille a longtemps été vu comme un méchanisme de diffusion de l'information à grande échelle, influençant potentiellement l'opininion public, l'adoption d'innovations, la reconnaissance de marques, etc. Ces dernières années, les chercheurs dans le domaine du marketing, se sont intéressés de plus près à l'optimisation possible ou non de la diffusion d'une information ou d'un nouveau produit, à partir de certain types d'individus, appelés ici « *influentials* ». Ces individus disposeraient de certaines combinaisons d'attributs désirables, que ce soient des attributs personnels tels que la crédibilité, l'expertise ou l'enthousiasme, ou bien des attributs liés au réseau, comme la connectivité par exemple. Jusqu'à il y a quelques années, l'étude des mécanismes de diffusion du bouche-à-oreille, souffrait de certaines difficultés. Tout d'abord, le réseau par lequel le bouche-à-oreille se propageait étaient généralement difficilement, voire pas observable. Ensuite, les observations de diffusion sont fortement biaisées par la diffusion d'événements « à succès », qui par leur essence même sont facilement remarqués et en plus leur occurrence est fort rare. Fort heureusement, les services de micro-blogging tels que Twitter offre potentiellement de pallier à ces 2 difficultés, et offrent un laboratoire naturel pour l'étude des processus de diffusion. Avec Twitter, le réseau , « qui écoute qui ? » peut être reconstruit en reconstituant ce qu'on appelle le « follower graph ». Twitter force ses utilisateurs à communiquer plus ou moins tous de la même manière, via des *tweets* vers leurs *followers.*

Ici les auteurs sont partis d'une base de données de 2 mois de tweets, allant de septembre au 15 octobre 2009. De cette DB, ils ont extrait les tweets incluant des *bit.ly URLs*, correspondant à la diffusion d'événement distincts, où chaque événement avait un *poster* unique comme origine (« *seed* »). Pour finalement garder un sous-ensemble de 74 millions d'événements engendrés par des posters ayant été actifs sur les 2 mois, gardant le mois de septembre pour l'apprentissage de leur modèle de régression, et le second mois pour prédire la performance. Afin de déterminer le « niveau d'influence » d'un post d'URL donnée, la diffusion de l'URL a été tracquée de son origine (« seed »), via ses différents *followers*, etc jusqu'à la fin de la cascade.

La *figure 2* , visualise quelques exemples de cascades possibles sur Twitter.

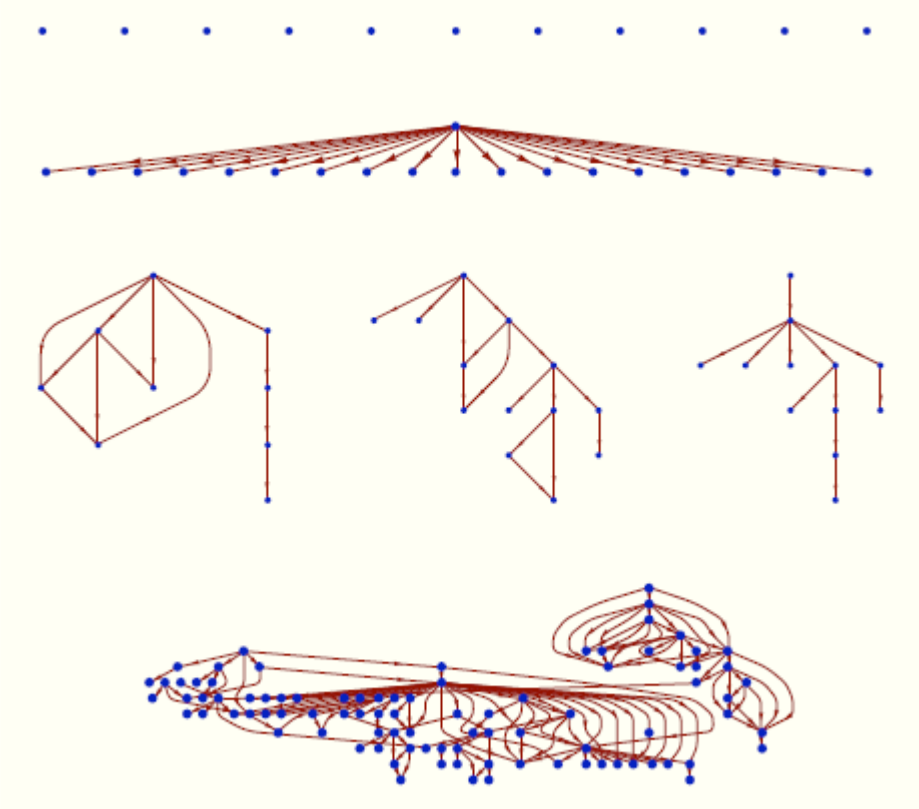
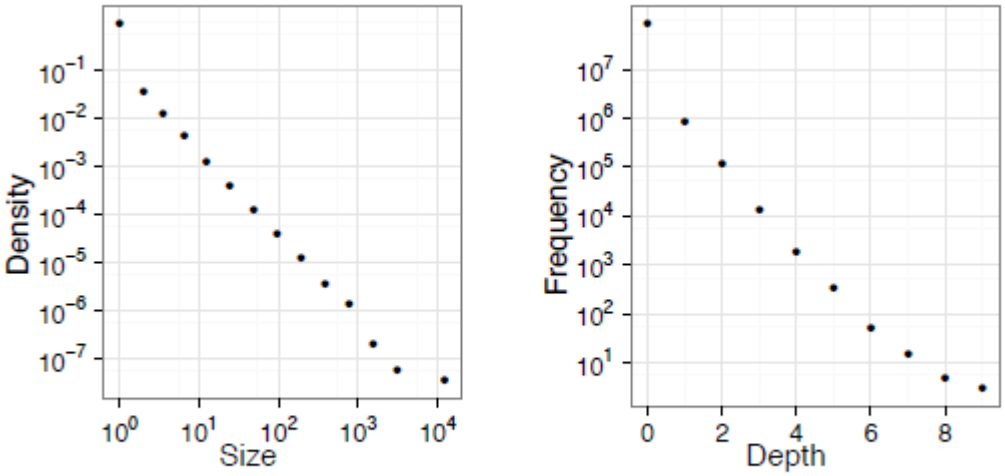


Figure 2

La *figure 3 a,* montre que la distribution de la taille des cascades peut être approximée par une loi de puissance. Ceci implique que la grande majorité des URLs postés ne sont pas diffusés du tout, alors qu'une faible portion se voit repostée des milliers de fois. La profondeur de la cascade (*figure 3b*) ressemble plus à une distribution exponentielle. Alors que les cascades les plus profondes peuvent se propager jusqu'à 9 générations à partir du *poste*r original, la majorité encore une fois n'a qu'une profondeur de 0, où le *seed* est le seul nœud de l'arbre.

Les résultats de cette étude étaient plus ou moins concluants, en ce sens que les comptes ayant un grand nombre de followers et des succès dans le passé, semblent être des critères nécessaires mais non suffisants pour conclure au succès future. Ceci replace l'intuition courante concernant les posters influants en perspective : les individus ayant été influants par le passé et ayant toujours un nombre importants de followers sont plus susceptibles d'être influants dans le futur, mais cette intuition n'est correcte qu'en moyenne.



(a) *Cascade Sizes* (b) *Cascade Depths*

Figure 3

L'on pourrait également s'attendre à ce que certain types de contenu (par exemple, vidéos YouTube) auraient une plus grande tendance à être diffusé largement que d'autres. Afin d'évaluer la puissance prédictive additionnelle du contenu, les auteurs ont répété, l'arbre de régression, en rajoutant des caractéristiques liées au contenu. Ces dernières ont été évaluées par des humains à l'aide du service Amazon's Mechanical Turk (AMT).

1. Rated interstingness
2. Perceived interestingness to an average person
3. Rated positive feeling
4. Willingness to share via Email, IM, Twitter, Facebook or Digg
5. Indicator variables for type of URL
6. Indicator variable for category content

De façon quelques peu surprenante aucune des caractéristiques de contenu, n'a apporté en précision quant à la prédiction future.

Alors que la *régression linéaire classique* tendra à fitter la grande majorité de cascades de petite taille, au dépend des grandes, les *arbres de régression* permettent aux cascades de différentes tailles d'être fittés indépendamment. Ce qui résulte en une meilleure calibration des arbres de régression.

Dans cette étude, le modèle inclue les caractéristiques suivantes comme prédicteurs :

1. Seed user attributes
    a. # followers
    b. # friends
    c. # tweets
    d. Date of joining

2. Past Influence of seed users
    a. Average, minimum and maximum total influence
    b. Average, minimum and maximum local influence

iii. Algorithme passif - agressif :

Ref : RT! Predicting Message Propagation in Twitter , S Petrovic, M. Osborne, V. Lavrenko

Dans cet article, les auteurs ont étudié des tweets sur un jeu de données datant d'octobre 2010. Cela leur a permis de rassembler un ensemble de 21 millions de tweets, qu'ils ont séparé en 2, la première partie pour le calibrage sur base de l'algorithme de machine learning, et l'autre pour le tester. L'algorithme Passif – Agressif, s'éfforce à maintenir une frontière décisionnelle linéaire et pour chaque nouvel exemple, il essaie de le classifier correctement avec une certaine marge d'erreur, tout en gardant la nouvelle frontière décisionnelle aussi proche que possible de l'ancienne.

Ils sont partis de l'idéé que chaque heure de la journée disposait de règles spécifiques quant à ce qui sera retweeté (par exemple, des tweets contenant le mot « oil » seraient peut-être plus retweetés le matin que le soir.) Raison pour laquelle, ils ont utilisé un modèle local pour chaque heure de la journée par rapport à laquelle, le tweet a été écrit. A côté de cela, ils ont distingué 2 sets de caractéristiques différents : les *caractéristiques sociales* et les *caractéristiques propres aux tweets*.

Sous les *caractéristiques sociales*, étaient repris le nombre de followers, d'amis, de status, nombre de fois que l'utilisateur était listé, si l'identité du poster était *vérifiée* et si la langue était l'anglais. Le nombre de *followers* et d'amis, s'est avéré une fois de plus être un bon indicateur de la retweetabilité. La *vérification* est majoritairement utilisée par Twitter pour confirmer l'authenticité des comptes des célébrités. Selon l'étude, 91% des tweets écrits par un utilisateur vérifié sont retweetés, contre 6% pour des comptes dont l'auteur n'est pas vérifié.

Dans les *caractéristiques propres aux tweets*, l'on retrouve le nombre de hashtags, les URLs, les mots tendances, la longueur du tweets, le contenu du tweet.

Dans ce cas-ci, le but était de déterminer si un tweet était susceptible d'être retweeté, et non de trouver le nombre final de retweets. Afin de vérifier si cette tâche était possible, une première expérience a été réalisée avec des humains. Un jeu de 202 pairs de tweets a été présenté à 2 individus, et dans chaque pair un tweet exactement avait été retweeté. Cela s'est passé en 2 étapes, premièrement uniquement le texte des tweets étaient montrés, sur base duquel les personnes devaient dire si selon elles le tweet serait retweeté, et deuxièmement les caractéristiques supplémentaires étaient également montrés. Dans le premier cas, les humains ont atteint un taux d'exactitude de 76.2% et de 73.8%, démontrant que les humains étaient capables de distinguer les tweets allant être retweetés des autres. Dans le deuxième cas, les scores atteints furent de 81.2% et 80.2%. L'algorithme quant à lui a permis d'atteindre dans le premier cas 69.3% de taux de réussite et 82.7% dans le second cas.

Comme on a pu s'en rendre compte au travers de ces diverses exemples, certains inconvénients sont liés à cette famille de méthodes. Dans un premier temps, il faut réaliser une étude très approfondie des caractéristiques, car la fiabilité des résultats est fortement liée à la qualité des caractéristiques choisies.

Ensuite vue la quantité immense d'information à traîter en suivant ces méthodes, celles-ci ne peuvent pas être utilisées en temps réel. Ce qui est des plus contreproductif, si l'on pense à l'utilité recherchée lors de la prédiction du comportement des cascades

b. *Point process based methods*

Cette catégorie de méthodes est basée sur les « point processes », modélisant de manière directe la création de cascades d'information dans un réseau. A l'origine le plus souvent ces méthodes étaient développées pour répondre au problème complémentaire d'inférence du réseau. Où en observant plusieurs cascades d'information, l'on essaie de retrouver la structure du réseau sous-jacent, à partir duquel les cascades furent propagées.

Inférence structure du réseau :

Ref : Learning Networks of Heterogeneous Influence , N. Du, L. Song, A. Smola, M. Yuan

Les graphes peuvent être des abstractions puissantes pour la modélisation d'une variété de systèmes naturels et artificiels, consistant en une large collection d'entités interagissantes. Etant donné la disponibilité grandissante de réseaux à grande échelle, la modélisation de graphes et leur analyse a été appliquée maintes fois à l'étude de la propagation et de la diffusion d'information, d'idée et même de virus dans des réseaux d'information. Toutefois il est important de se rendre compte, que le processus d'influence et de diffusion a souvent sa place dans un *graphe caché* (« *hidden network* ») complexe à observer et identifier. Par exemple, quand une maladie se propage dans une population, les épidémiologistes ne peuvent que savoir quand une personne tombe malade, mais ils peuvent rarement déterminer avec exactitude où et par qui cette personne a été infectée. De façon similaire, en marketing, quand des consommateurs se pressent d'acheter un produit particulier, on peut monitorer quand l'achat a eu lieu, mais pas d'où venait originellement la recommandation. Dans de tels cas, on peut observer l'instant auquel une information a été reçue par une entité particulière, mais le chemin exact de diffusion reste caché. D'où le challenge intéressant d'essayer de reconstituer le chemin de diffusion sur base de l'observation des différents instants d'occurrences des événements étudiés.

Ici la « survival analysis » (*basé sur les travaux de Gomez-Rodriguez*) est utilisée pour modéliser la diffusion d'information pour des entités connectées dans un graphe. L'hypothèse est faite, qu'il y a une population fixe de N nœuds connectés dans un graphe orienté G=(V,E). Des nœuds voisins peuvent s'influencer directement les uns les autres. Les nœuds se trouvant sur un chemin orienté, ne peuvent s'influencer qu'en suivant le processus de diffusion. Comme le véritable graphe sous-jacent est inconnu, seuls sont observés les instants auquels un événement se passe pour chaque nœud du réseau. Ces différents instants sont alors organisés comme des cascades, correspondant chaque fois à un événement particulier.

La *figure,4 a* , représente les cascades sur un « *hidden network* », les traîts pleins indiquent les différentes connections. Dans les *4b et 4c* , les cercles pleins donnent les sommets infectés, alors que les vides donnent les non infectés. Les nœuds a, b,c et d sont les parents du nœuds e, ayant été infectés respectivement en t0 < t1 < t2 < t3, et chechent à infecter le nœud e. Deux cas de figure peuvent se présenter soit en b) *e* est infecté, soit en c) *e* survit bien que tous ces parents aient été infectés.

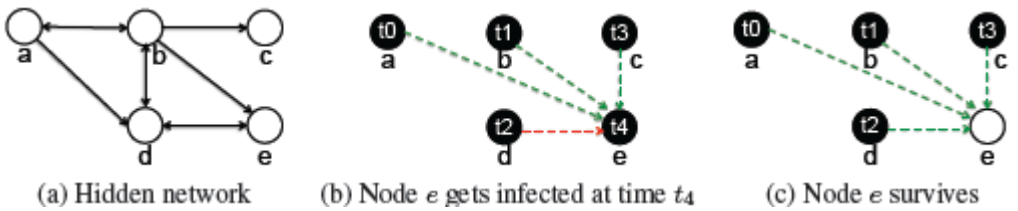


Figure 4

En pratique, les schémas de diffusion d'information parmi les entités peuvent être compliqués et différents les uns des autres, dépassant de loin ce qu'une famille de modèles paramétriques simples peut représenter. Par exemple, sur Twitter, un utilisateur actif peut être connecté plus de 12 heures par jour, et peut répondre instantanément à n'importe quel message lui semblant intéressant.

A l'opposé un utilisateur inactif, peut ne seulement se connecter qu'une fois par jour. Ce qui fait, que les schémas de diffusion des messages entre l'utilisateur actif et ses amis, peuvent être assez différents de ceux de l'utilisateur inactif. La *figure 5* , montre les histogrammes des intervalles entre le moment où un *post* arrive sur un site, et le moment où un nouveau *post* y faisant référence apparait sur un autre site.

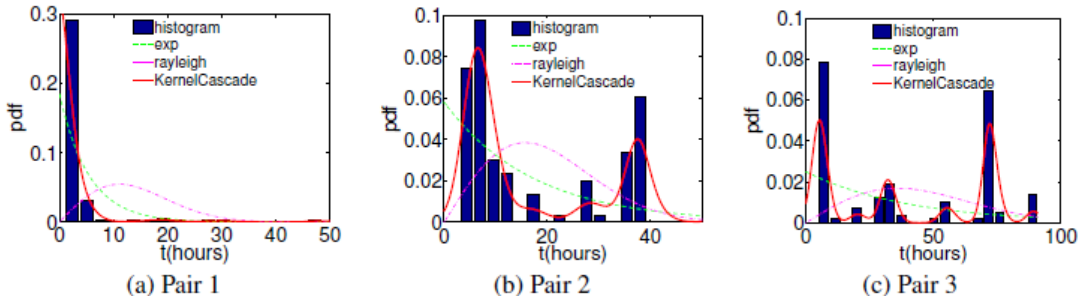


Figure 5

Dans cet article, les auteurs détaillent leur « flexible kernel method », permettant de modéliser les processus de diffusion latents et d'inférer le réseau caché avec des influences hétérogènes entre chaque paire de nœuds.

Ref : Mixture of Mutually Exciting Processes for Viral Diffusion, S.-H. Yang, H. Zha

La question de l'évolution temporelle et spatiale dans un réseau social, de signaux viraux (comportements, idée, maladies,…) est encore une fois à la base de cette étude-ci. Les chercheurs ont par le passé été challengés par 2 problèmes fondamentaux :

« Diffusion network inference » : comme expliqué précédemment le processus de diffusion se passe généralement sur un réseau dont la structure ne peut être observée ou identifiée directement.

« Tracking trending memes » : par « meme » on se réfère ici à un groupement d'événements viraux évoluant et se propageant de manière similaire les uns par rapport aux autres, mais relativement indépendemment d'événements extérieurs à ce *cluster*. Souvent on a affaire à un certain nombre de « memes » se diffusant et s'entremelant simultanément les uns avec les autres. Par exemple, plusieurs maladies peuvent se propager dans une communauté au même moment, bien que seul les conditions et les symptomes, plutôt que le type de maladie, puissent être identifiés directement quand un individu est infecté.

Généralement ces 2 questions sont étudiées séparément, alors que dans cet article les auteurs cherchent à répondre aux 2 simultanément. Selon ces derniers, ces 2 points sont des cas spéciaux d'un problème plus général, à savoir le suivi du flux des « memes » soit spatialement sur le réseau (« Network Diffusion »), soit temporellement (« Meme Evolution »). Pour ce faire, ils introduisent ce qu'ils appellent un « probalistic mixture model », sur un ensemble de processus de Hawkes multivariés (Multivariate Hawkes Process, MHP). La nature auto-exitante du processus de Hawkes, en fait un choix parfait pour la modélisation de l'évolution et de la propagation d'un « meme » unique ou de « memes » indépendants. Le modèle développé tient compte tant des effets endogènes (infections dans le réseau social) que des effets exogènes (infections externes au réseau).

Il est intéressant de mentionner que les modèles basés sur le « survival analysis » peuvent être vus comme des cas particuliers du modèle MHP utilisé ici. Et ce avec les hypothèses implicites suivantes, les événements ne sont pas récurrents, càd qu'un nœud ne peut être infecté qu'une seule fois et que le réseau étant inférré est fermé, càd que les nœuds ne propagent que des « memes » déjà existants dans le réseau, ils ne peuvent ni être influencés par un nœud hors du réseau, ni créer un nouveau « meme ». Ces hypothèses n'étant pas réalistes, elles ont été retirées du modèle MHP.

Les 2 algorithmes utilisés ici, peuvent être interprétés intuitivement. *L'algorithme de suivi* des « memes », groupe les événements en « memes » en se basant non seulement sur la sémantique du contenu viral, mais également sur les schémas d'évolution et de propagation. L'identification d'un événement faisant parti d'un « meme », est inféré en intégrant 5 aspects différents.

1. La *popularité antérieure* de chaque « meme »
2. La *sémantique du cluster* d'un contenu viral (par exemple, des virus ayant une structure génétique similaire seront groupés comme un « meme »
3. *Infection spontanée*, quelle est la probabilité qu'un « meme » ait été créé spontanément par le nœud i_n
4. *Diffusion passée*, comment le « meme » a été propagé avant d'infecter i_n
5. *Diffusion future,* comment le « meme » serait propagé après avoir infecté i_n
   L'algorithme d'inférence de réseau, retrouve le réseau de diffusion caché, en estimant la matrice d'infectivité à partir des graphes orientés acycliques (induit par la causalité temporelle) des « memes » identifiés. L'idée principale est d'approximer la relation de causalité à partir d'un jeu de causalités temporelles, par exemple si Bob est presque toujours infecté juste après l'infection d'Alice, et personne n'a autant de causalités temporelles avec Bob qu'Alice, il est fortement probable qu'Alice influe sur Bob.

   Le modèle a ensuite été appliqué à un jeu de données en provenance de Twitter de mi-juin à fin novembre 2009. Le but était de modéliser la diffusion de « memes » dans le monde réel, en

particulier identifier les « memes » (thèmes, idées, comportements) et suivre leur tendance. A côté de cela, il convenait également de déterminer le graphe caché sous-jacent.

La *figure 6* , montre le top dix des « memes » (affichés comme les termes les plus représentatifs de chaque « meme ») identifiés sur Twitter. Après analyse, il s'avère que ces « memes » représentent de façon satisfaisantes les événements tendances s'étant passés durant la période étudiée (p. exemple la réforme d'Obama sur les soins de santé, la grippe porcine,…)

1	search business deal microsoft billion yahoo pay buy google market
2	nba game lakers top season teams kobe sox howard win
3	honduras mark harriet global journey culture gilbert arts strand coles
4	oil hurricane european storm dollar china open tropical off bill
5	afghan killed pakistan taliban bomb kills iraq troops attack kabul
6	china iran obama russia minister president leader deal myanmar korea
7	fire ny killed nj ave dead plane crash injured hudson
8	sales profit uk loss rise prices london economy quarter june
9	obama medical health care house politics bill government plan reform
10	man police flu woman death swine murder charged court arrested

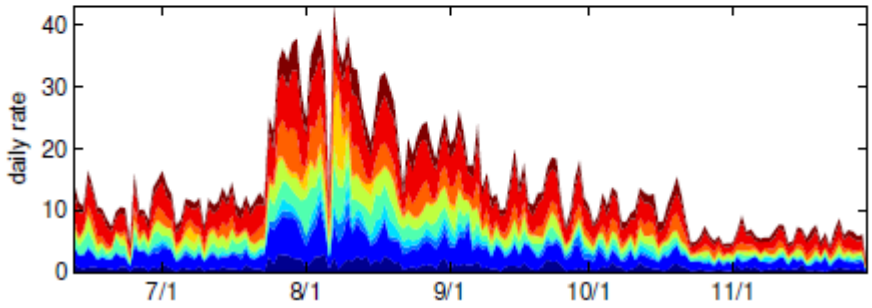


Figure 6

3. Modèle SEISMIC

Ref : SEISMIC : a Self-Exciting Point Process Model for Predicting Tweet Popularity, Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, J. Leskovec

a. Jeu de donées

Les données utilisées dans ce travail sont les mêmes que celles recueillies pour l'article SEISMIC, à savoir un jeu complet de 3.2 milliards de tweets et retweets apparus sur Twitter du 7 octobre 2011 au 7 novembre 2011. Chaque retweet est caractérisé par l'id du tweet original, le moment du post, le moment relatif de retweet ainsi que le nombre de followers de l'utilisateur qui tweete/retweete. Sur ce jeu complet, seulement ont été retenus les tweets ayant au moins 50 retweets. Il y a 166.076 tweets satisfaisant ce critère les 15 premiers jours. R_∞ est approximé par R_{14}, le nombre de retweets après 14 jours. L'on remarquera que la seule information disponible sur le réseau, concerne le nombre de followers à chaque nœud. Ces données sont téléchargeables sous la forme de data.csv et index.csv.
(http://snap.stanford.edu/seismic/)

Download:

- **data.csv** (34,784,489 lines of tweets/retweets, 285Mb)
- **index.csv** (166,077 lines of tweets, 7.9Mb)

Data format:

- data.csv (with header)

```
<relative_time_second>,<number_of_followers>

 <relative_time_second>: relative post time of the tweet/retweet (in second)
 <number_of_followers>: number of followers of the user who tweets/retweets
```

- index.csv (with header)

```
<tweet_id>,<post_time_day>,<start_ind>,<end_ind>

 <tweet_id>: id of the original tweet
 <post_time_day>: post time (UTC) of the original tweet (in day)
 <start_ind>: the first row in data.csv of this tweet
 <end_ind>: the last row in data.csv of this tweet
```

Des paramètres importants dans ce modèle sont :

- R_t représentant le nombre total de retweets pour une publication donnée au temps t
- λ_t la vitesse d'expansion de la cascade, qui repose sur la « post infectiousness p_t » et le temps de réaction humain
- $\phi(s)$, le « memory kernel », quantifiant le délai entre un *post* arrivant au *feed* d'un utilisateur et le moment où celui-ci le partage à son tour. Cela représente intuitivement le temps de réaction de l'utilisateur.

Le but étant ici de déterminer R_∞ , le nombre final de retweets.

b. *Memory Kernel*

Ref 1: The origin of bursts and heavy tials in human dynamics, A. L. Barabasi

Ref 2: Robust dynamic classes revealed by measuring the response function of a social system. R. Crane , D. Sornette

Afin de prédire la taille de la cascade, il nous faut connaître combien de temps il faut à une personne pour que celle-ci partage un tweet. Le temps de réaction humain a été étudié dans de nombreuses recherches, ici nous nous attarderons sur 2 d'entre elles. La première réalisée par A. L Barabasi et la deuxième par R. Crane et D. Sornette.

A. L Barabasi, s'est intéressé au temps de réaction humain intervenant dans différentes tâches du quotidien, allant des communications électroniques (telles que l'envoi de mails ou les appels téléphoniques, la navigation sur le web ou encore des transactions financières). La plupart des modèles de l'activité humaine se basaient sur les processus de Poisson et assumaient que dans un intervalle de temps dt, un individu (appelé un *agent*) s'engagerait dans une action spécifique et ce avec une probabilité *qdt,* où *q* est la fréquence globale de l'acitivité monitorée. Ce modèle prédit par conséquent que l'intervalle de temps entre 2 actions consécutives effectuées par le même individu suit une distribution exponentielle. *(figure 7)*

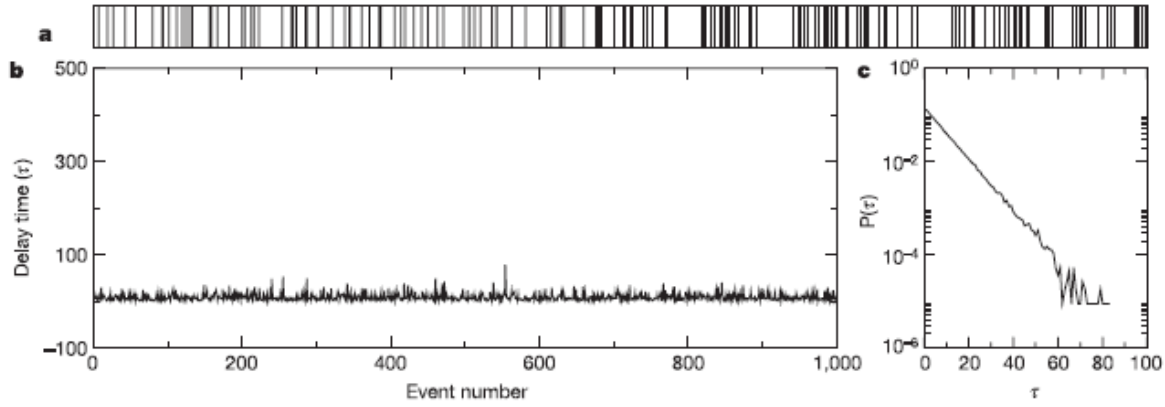


Figure 7

Bien que les processus de Poisson soient largement répandus pour quantifier les conséquences d'actions humaines (par exemple, modélisant les flux de la circulation, la fréquence des accidents, le contrôle de stock, le trafic d'appels dans un call-center), beaucoup de prises de mesures récentes du timing de nombreuses activités humaines dévient systématiquement de la distribution de Poisson. Les intervalles de temps entre 2 actions successives semblent mieux approximimées par une « *Heavy tailed » distribution.*

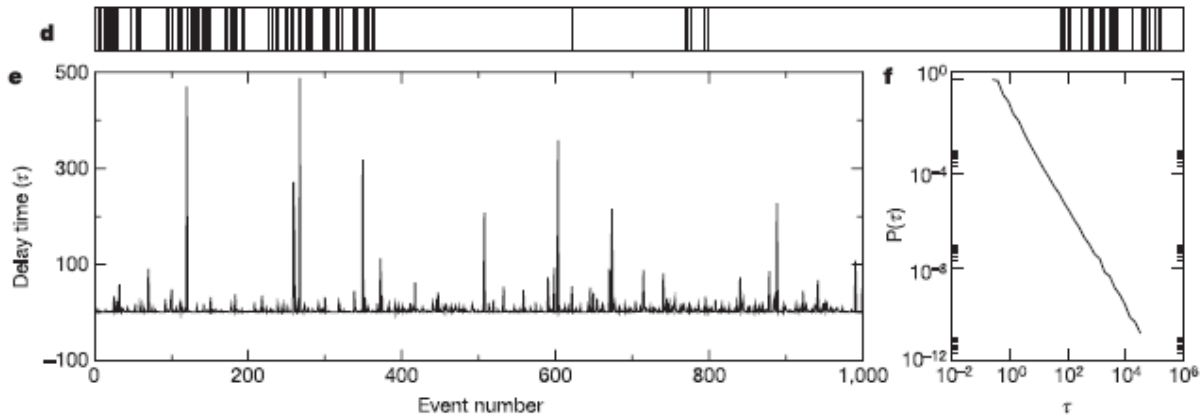


Figure 8

Alors qu'une distribution de Poisson diminue de façon exponentielle, forçant les événements consécutifs à se suivre de manière relativement régulière, une distribution heavy-tailed à décroissance lente autorise de très longs temps d'inactivité séparant des périodes hautement réactives.

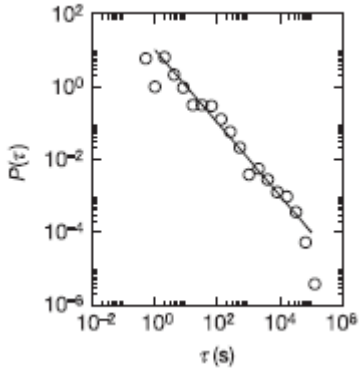


Figure 9

Afin de démontrer ce caractère non-Poisson dans le comportement humain, A. L. Barabasi a étudié plusieurs milliers de communication par mails reposant sur une base de données reprenant l'identifiant de celui à l'origine du mail, de celui qui le réceptionne, le temps et la taille de chaque mail. La *figure 9* montrant les intervalles de temps entre 2 mails consécutifs se voit le mieux modélisé par $P(\tau) \sim \tau^{-\alpha}$, où $\alpha \sim 1$ indiquant que le schéma d'activité de mails d'un individu a un caractère non-Poisson. Pendant une session unique un utilisateur envoie plusieurs mails se succédant rapidement, avec ensuite une longue période d'inactivité.

Ce comportement se retrouve également dans d'autres situations, comme par exemple la distribution du temps d'attente lors de discussions online sous forme de messages instantanés.
Selon A. L. Barabasi, la nature de cette dynamique humaine est une conséquence du processus de files d'attente, engendré par la prise de décision humaine. A chaque fois qu'un individu est face à différentes tâches et choisit entre elles selon une certaine perception de priorité, les temps d'attente des différentes tâches suivront une distribution *heavy tailed*. Parcontre, lorsqu'il s'agit de classement aléatoire des tâches ou FIFO, cela mènera le plus souvent à une distribution Poisson.

R. Crane and D. Sornette, ont étudié les fonctions de réponses de systèmes sociaux, ils ont basé leur étude sur une base de données reprenant une sélection de 8 mois d'activités sur Youtube donnant lieu à plus de 5 millions de séries temporelles d'activités humaines. Différents facteurs peuvent conduire au visionnage d'une vidéo sur YouTube, déclenchement via un mail, un link via un siteweb extérieur, discussions sur un blog, journaux et télévision et autres influences sociales. Un des ingrédients constitutifs du modèle épidémique utilisé, est une distribution selon une loi de puissance, décrivant l'activité humaine. L'hypothèse est faite (sur base notamment des études de A. L. Barabasi), que cette distribution prend la forme d'un processus à mémoire du type :

$$\varphi(t) \sim t^{-(1+\vartheta)} \quad avec\ 0 < \vartheta < 1$$

Par définition, ce « memory kernel $\varphi(t)$» représente la distribution des temps d'attente entre une « cause » et « l'action » pour un individu.

Dans le modèle SEISMIC, on suit l'hypothèse suivant laquelle, le temps *s*, entre l'arrivée d'un *tweet* sur l'échelle de temps de l'utilisateur et son partage par ce dernier est distribué selon une densité $\phi(s)$.

Généralement, on assume que la queue de la distribution *heavy-tailed* de $\phi(s)$ suit une loi de puissance avec un exposant entre 1 et 2 ou une distribution log-normal. Cependant étant donné la nature très réactive des tweets et retweets sur Twitter, il est également naturel de s'attendre à beaucoup de temps de réaction instantantés. D'après l'étude des données Twitter, $\phi(s)$ serait de la forme suivante :

$$\phi(s) = \begin{cases} c & 0 < s \leq s_0, \\ c(s/s_0)^{-(1+\theta)} & s > s_0. \end{cases} \quad (1)$$

Pour l'estimation des paramètres de $\phi(s)$, les auteurs de l'article ont choisi 15 tweets dans le « training set » et ont utilisé la distribution de tous leur temps de retweets comme $\phi(s)$. Les *posters* originaux de chacun de ces 15 tweets ont tous un nombre impressionnant de *followers*, et par conséquent la grande majorité des retweets devraient venir de followers directs du « poster » original.

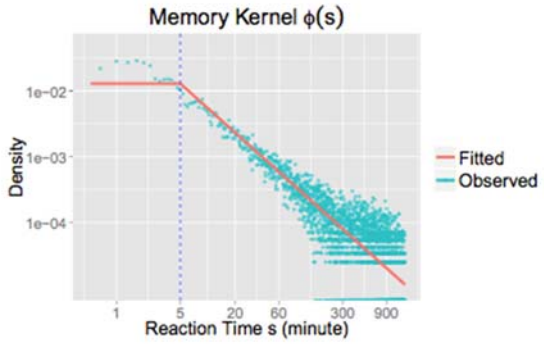


Figure 10

L'observation de la distribution des temps de réaction jointe à l'équation (1) montre que $\phi(s)$ est plus ou moins constant les 5 premières minutes (s_0 =5), suivi ensuite d'une loi puissance décroissante *(figure 10).* Différents réseaux sociaux peuvent avoir différentes distributions de temps de réaction humain. Toutefois, $\phi(s)$ ne doit être déterminé qu'une fois par réseau.

c. *Post infectiousness*

L'hypothèse est faite par les auteurs que chaque *post* w est associé à un paramètre dépendant du temps $p_t(w)$, indiquant la probablité d'être retweeté à l'instant *t*. L'infectiousness d'un post peut dépendre d'un certain nombre de facteurs, incluant la qualité du contenu du post, la structure du réseau social, l'heure locale ou encore la localisation géographique. Avec SEISMIC, plutôt que de partir sur une forme paramétrique de p_t, les chercheurs ont préféré le modéliser de façon plus flexible sous forme non-paramétrique prenant toutefois les différents paramètres en compte de manière implicite. La plupart des modèles étudiant les « self-exciting point processes » prennent un p_t constant dans le temps. Un concepte important en découlant est le caractère critique du processus R_t. Dans un « self-exciting point process » ayant un infectiousness constant, $p_t = p$, l'on observe une phase de transition à une certaine limite critique p* tel que :

- si p>p* alors R_t tend vers l'infini quand t tend vers l'infini et ce de façon exponentielle. C'est ce qui est appelé le régime *supercritique*.
- Si p <p*, alors $\sup_t R_t < \infty$, soit le régime *subcritique*.

En réalité R_t est toujours borné, étant donné la taille finie du réseau. Cela signifie qu'aucune cascade en régime supercritique pourrait exister lorsque p_t est supposé être constant. Ce qui va à l'encontre de l'idée de modélisation de tweets hautement contagieux. Pour résoudre ce problème dans SEISMIC, p_t n'est pas

pris constant. De plus, à mesure qu'une publication devient plus ancienne, l'information qu'elle transmet peut être dépassée, et sa probabilité d'être retweeté peut du coup diminuer. Cette décroissance peut également s'observer à mesure que l'on s'éloigne de celui qui a poster le tweet original. A l'opposé l'on pourrait voir cette probabilité augmenter lors d'un retweet par un utilisateur ayant un grand nombre de followers.

Avec le modèle SEISMIC, les temps de réaction humain sont combinés à la *post infectiousness*. Dans un premier temps, R_t est modélisé comme un « doubly stochastic self-exciting point process ».

Définissons l'intensité de λ_t , indiquant le taux d'apparition d'un retweet additionel à l'instant t :

$$\lambda_t = \lim_{\Delta \downarrow 0} \frac{\mathbb{P}\left(R_{t+\Delta} - R_t = 1\right)}{\Delta}$$

Dans le modèle utilisé par SEISMIC, λ_t dépend de l'infectiousness p_t, du moment de retweet t, du degré du noeud n_i et de la distribution de la réaction humaine $\phi(s)$. La relation en elle-même découle de la théorie de processus de Hawkes:

$$\lambda_t = p_t \cdot \sum_{t_i \leq t,\ i \geq 0} n_i \phi(t - t_i), \quad t \geq t_0$$

Intuitivement $\sum_{t_i \leq t,\ i \geq 0} n_i \phi(t - t_i)$ représente l'intensité d'arrivée d'utilisateurs nouvellement exposés à l'instant t. Ce *point process,* est appelé *self-exciting,* car chaque observation antérieure *i* telle que $t_i < t$ contribue à l'intensité λ_t, ce qui signifie que chaque observation augmente l'intensité future. A côté de cela ce processus est également doublement stochastique étant donné le caractère stochastique de p_t à l'origine. Finalement, l'on assume que les degrées des noeuds n_i sont indépendants et identiquement distribués de moyenne n*. Cette moyenne est reliée au seuil critique p* de la façon suivante, p*=1/n*.

Estimation des différents paramètres selon le modèle SEISMIC

Dans ce qui suit, l'hypothèse est faite que les *followers* de tous ceux qui retweetent sont disjoints, ainsi une structure arborescente peut être utilisée afin de décrire la diffusion de l'information.

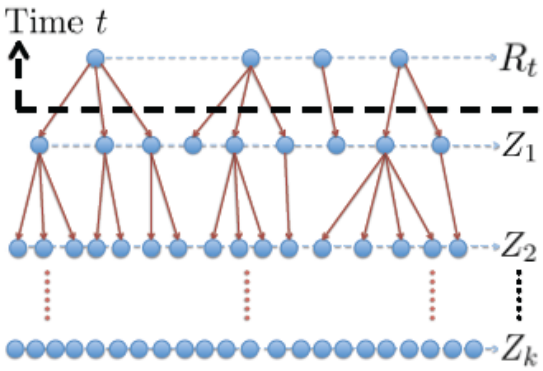


Figure 11

L'article définit le Maximum Likelihood Estimate (MLE) de p_t comme suit :

$$\hat{p}_t = \frac{R_t}{\sum_{i=0}^{R_t} n_i \int_{t_i}^{t} \phi(s - t_i) ds}$$

Cette équation est à la base de SEISMIC, permettant d'estimer l'infectiousness p_t à un instant t donné. Afin de lisser le MLE, uniquement les observations proches de t sont utilisées pour estimer p_t. Et pour se faire, on se base sur $K_t(s)$ (*one-side kernels*), s>0 indexé par le temps t. Ces noyaux sont utilisés pour pondérer les retweets. Ce qui donne :

$$\hat{p}_t = \frac{\sum_{i=1}^{R_t} K_t(t - t_i)}{\sum_{i=0}^{R_t} n_i \int_{t_i}^{t} K_t(t - s)\phi(s - t_i) ds}$$

On observe que lorque K(s)=1, on retombe sur l'expression pour MLE. Dans SEISMIC des noyaux triangulaires à fenêtre croissante t/2 sont utilisés pour $K_t(s)$:

$$K_t(s) = \max\left\{1 - \frac{2s}{t}, 0\right\}, \quad s > 0$$

Ce type de noyaux permet de ne prendre en compte que les posts plus récents que t/2, grâce à cela le modèle néglige rapidement la période initiale potentiellement instable. A côté de cela, avec le temps qui augmente une fenêtre plus large de posts est prise en compte, aidant par rapport à une fenêtre de taille constante, ainsi à stabiliser le calcul de *p(t).* De plus, pour des retweets à l'intérieur de la fenêtre, le noyau pondère plus fortement les posts les plus récents, et diminue graduellement les plus anciens. Cela maintient l'estimateur *p(t)* proche de la valeur réelle fort fluctuante de p_t.

La *figure 11,* illustre le concepte d'arbre de diffusion de l'information. La cascade étant observée jusqu'au temps t, on cherche à modéliser comment l'arbre va croître dans le future. Les variables Z_k indiquent le nombre de retweets engendrés par la génération k. Donc la première génération de déscendants Z_1 se réfère au nombre de nouveaux retweets générés par les posts créés avant t, alors que la seconde génération de déscendants Z_2 se réfèrent au retweets des posts des premièrs déscendants, etc. A partir de cela le nombre final de retweets serait indiqués par

$$R_t + \sum_{k=1}^{\infty} Z_k$$

Partant de cette constatation, et supposant que les degrées du réseau ont une moyenne égale à n*, et que l'i*nfectiousness* p_s est une constante *p* pour s >t, alors les auteurs de l'article établissent la proposition suivante:

$$\mathbb{E}|R_\infty| = \begin{cases} R_t + \frac{p(N_t - N_t^e)}{1 - pn_*}, & p < \frac{1}{n_*}, \\ \infty, & p \geq \frac{1}{n_*}. \end{cases}$$

Cette formule peut devenir instable quand *p* s'approche de la phase de transition 1/n*. Voilà pourquoi dans le modèle SEISMIC cette dernière est stabilisée de la façon suivante:

$$\hat{R}_\infty(t) = R_t + \alpha_t \frac{\hat{p}_t(N_t - N_t^e)}{1 - \gamma_t \hat{p}_t n_*},\ 0 < \alpha_t, \gamma_t < 1$$

Avec

$$N_t = \sum_{i:t_i \le t} n_i$$

$$N_t^e = \sum_{i=0}^{R_t} n_i \int_{t_i}^{t} \phi(s - t_i) ds$$

Les facteurs de corrections α_t et γ_t ont été introduits sur base de l'intuition suivante:

On s'attend à ce que α_t décroisse avec le temps t, afin qu'il diminue l'estimation future de l'*infectiousness*, ce qui permet ainsi de tenir compte des posts devenant hors propos avec le temps. Le coefficient γ_t quant à lui, est sensé augmenter avec le temps, alors que plus de noeuds sont exposés de façon multiple càd que le taux d'arrivée de nouveaux noeuds (non exposés précédemment) décroit avec le temps. Les mêmes valeurs de α_t et γ_t sont utilisées pour tous les posts mais ils peuvent varier avec le temps. Ces valeurs sont sélectionnées afin de minimiser la Absolute Percentage Error (APE).

Tout ceci a mené à l'élaboration de l'agorithme SEISMIC:

Algorithm 1 SEISMIC: Predict final cascade size

Purpose: For a given post at time t, predict its final reshare count
Input: Post resharing information: t_i and n_i for $i = 0, \ldots, R_t$.
Algorithm:
$N_t = 0$, $N_t^e = 0$
for $i = 0, \ldots, R_t$ **do**
  N_t += n_i
  N_t^e += $n_i \int_{t_i}^{t} \phi(s - t_i) ds$
end for
$\hat{R}_\infty(t) = R_t + \alpha_t \hat{p}_t (N_t - N_t^e)/(1 - \gamma_t \hat{p}_t n_*)$
Deliver: $\hat{R}_\infty(t)$

Algorithm 2 Compute real-time infectiousness $\hat{p}(t)$

Purpose: For a given post w, calculate infectiousness p_t with information about w prior to time t
Input: Post resharing information: t_i and n_i for $i = 0, \ldots, R_t$.
Algorithm:
$\tilde{R}_t = 0$, $\tilde{N}_t^e = 0$
for $i = 0, \ldots, R_t$ **do**
  $\tilde{R}_t$ += $K_t(t - t_i)$
end for
for $i = 0, \ldots, R_t$ **do**
  $\tilde{N}_t^e$ += $n_i \int_{t_i}^{t} K_t(t - s)\phi(s - t_i) ds$
end for
$p_t = \tilde{R}_t / \tilde{N}_t^e$
Deliver: p_t

En étudiant ces 2 algorithmes, on se rend compte que la complexité du modèle peu importe le choix de $\phi(s)$ et $K_t(s)$ est de $O(R_t)$ pour le calcul de $p(t)$ et R_∞.

Les avantages de SEISMIC

Ce modèle de prediction identifiant la popularité de tweets présente plusieurs avantages:

- il s'agit d'un *generative model* qui est non-paramétrique et suppose une connaissance minimale du réseau. Les seules informations nécessaires sont l'historique temporelle des retweets ainsi que le degré des noeuds impliqués dans ces retweets.
- Le modèle s'interprète de façon intuitive. Pour une cascade individuelle le modèle synthétise toute l'historique passée dans l'*infectiousness*.
- De plus SEISMIC a une complexité de calcul linéaire (O(# retweets))

Application de SEISMIC sur une cascade

Pour cette section, l'on s'intéresse au tweet original avec l'ID 127001313513967616, et à la cascade en découlant résultant en un jeu de données de 15563 lignes et 2 colonnes (reprenant le temps relatif de retweet en secondes ainsi que nombre de followers de celui qui retweete).

Le 20 octobre 2011 à la mort du Colonel Kadhafi, *@mottbollomy* publie le tweet suivant:

Chevrolet, le fabricant de voiture, qui pour quelconque raison suit ce compte, le retweete et s'attire les foudres des fans de Justin Bieber, et se voit ensuite obligé de lui présenter ses excuses.

La *figure 12* représente l'histogramme des 6 premières heures de retweets. Il est intéressant de remarquer que le fait que Chevrolet ait retweeté le post original dans les 30 minutes de sa publication a probablement entretenu sa popularité.

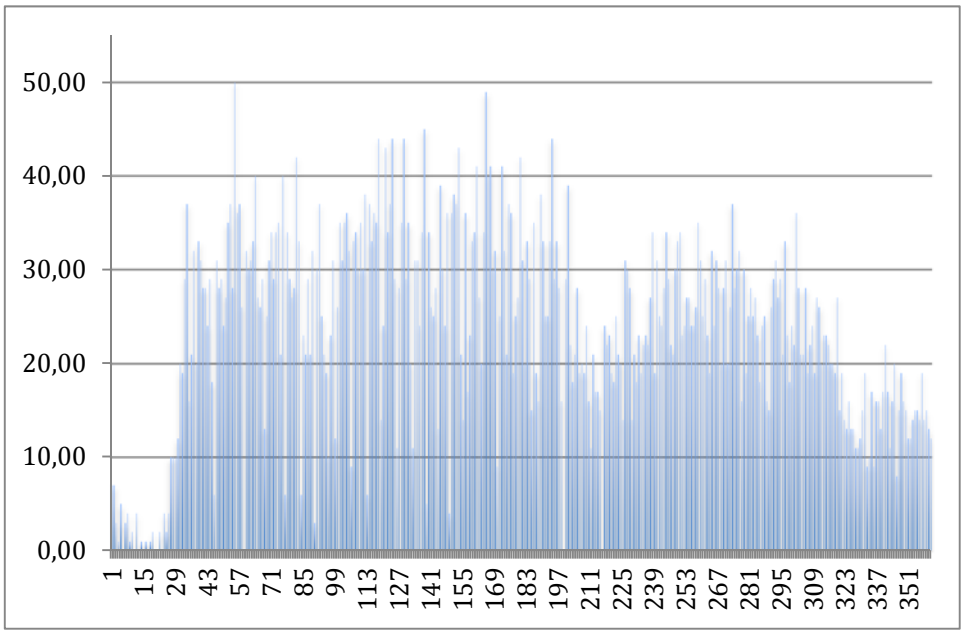
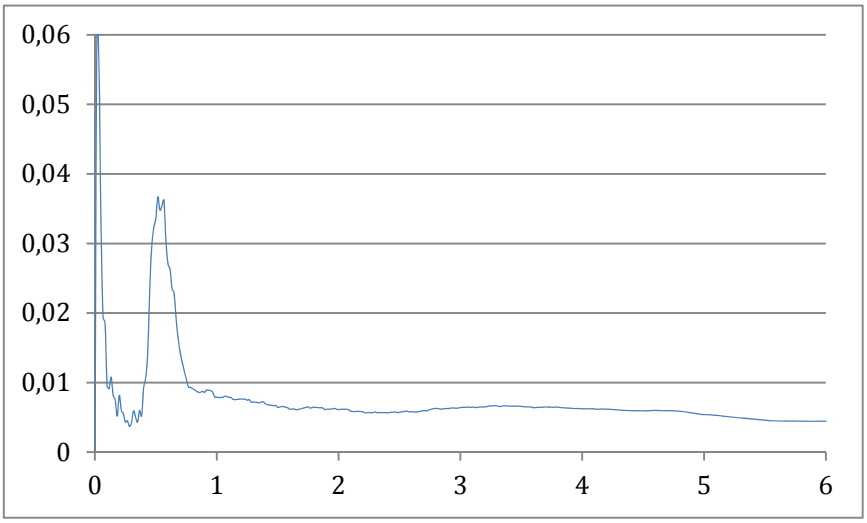


Figure 12

Ci-dessous l'infectiousness est mis en carte pour les 6 premières heures également.



Le résultat de l'infectiousness est ensuite utilisé comme décrit précédemment afin de prédire le nombre de retweet final (ligne horizontale). On observe que SEISMIC arrive rapidement à une estimation cohérente du nombre final de retweets. (*figure 13)*

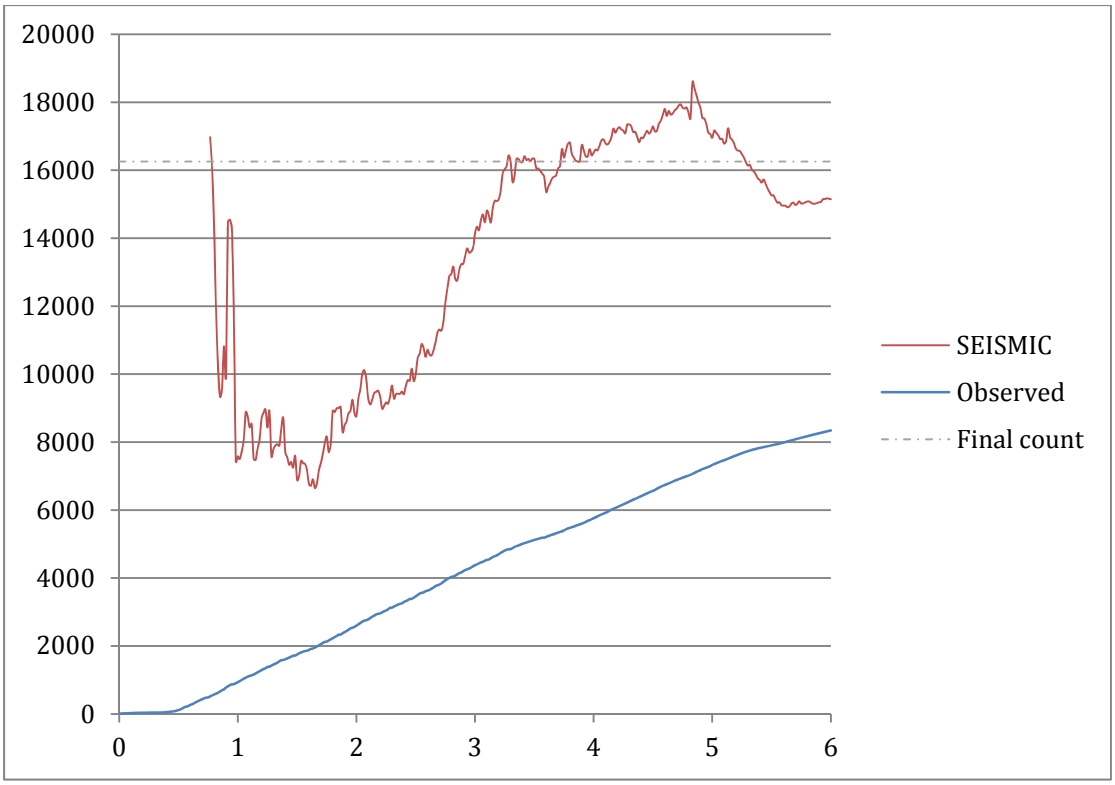

20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
0
1
2
3
4
5
6
SEISMIC
Observed
Final count

Figure 13

4. Etude de performance SEISMIC selon taille des cascades

Dans cette partie, nous allons nous intéresser de plus près à l'impact de la taille des cascades, sur le résultat de la prédiction à l'aide du modèle SEISMIC. Pour ce faire, le code en R (cfr Annexes) a été utilisé pour estimer le nombre de retweets. Afin de juger de la performance de l'estimation, nous utiliserons l'Absolute Percentage Error (APE) comme unité de mesure. Cette dernière est définie pour un tweet donné *w* et un instant de prédiction *t*, de la façon suivante :

$$\mathrm{APE}(w,t) = \frac{|\hat{R}_\infty(w,t) - R_\infty(w)|}{R_\infty(w)}$$

Comme mentioné précédemment, l'on suivra l'article SEISMIC quant à la valeur mesurée du nombre de retweets final, en prenant R_{14}, soit le nombre de retweets après 14 jours.

Dans un premier temps, nous regardons la distribution des tailles des cascades sur la bases de données, en prenant celles de minimum 1000 retweets. Un sous-échantillons de 3140 cascades est alors obtenu, tel que représenté sur la *figure 14*. L'on remarque que la majorité des cascades sur ce sous-échantillon, ont une taille proche des 1000 retweets. Ce qui correspond plutôt bien à l'intuition et à ce que l'on retrouve dans la littérature, à savoir que très peu de cascades atteignent un grand nombre de retweets.

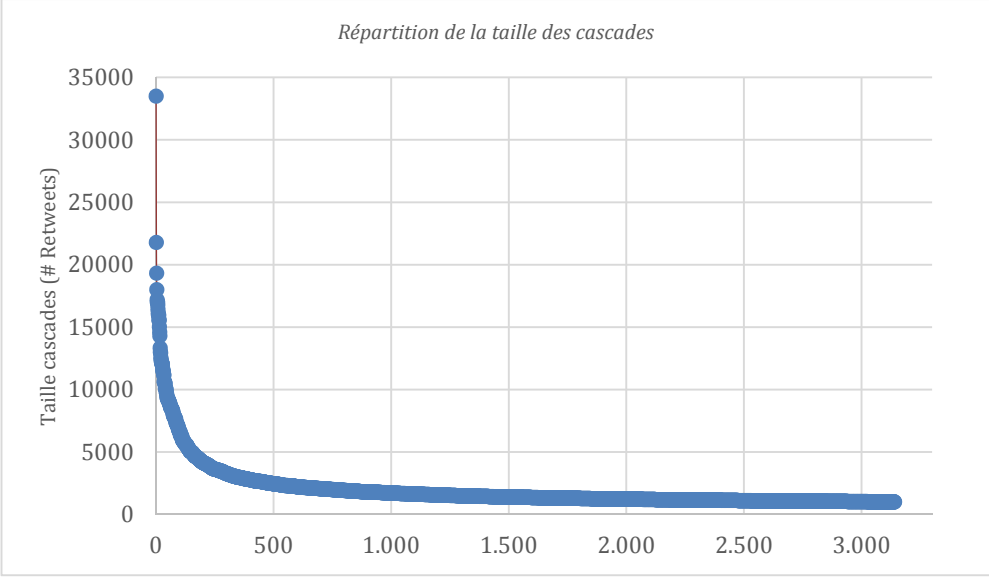


Figure 14

Pour la suite de l'étude, nous nous sommes basés sur cette répartition des cascades, afin de déterminer une répartition des tailles de cascades permettant d'obtenir 100 échantillons par catégorie de taille. Cela dans le but de pouvoir justifier une étude statistique pour chaque catégorie.

La *figure 15* représente cette idée schématiquement, et le choix des différents catégories, à savoir les cascades ayant une taille entre :

- 1000 et 2000 Retweets
- 2001 et 4000 Retweets
- 4001 et 6000 Retweets
- De plus de 6001 Retweets

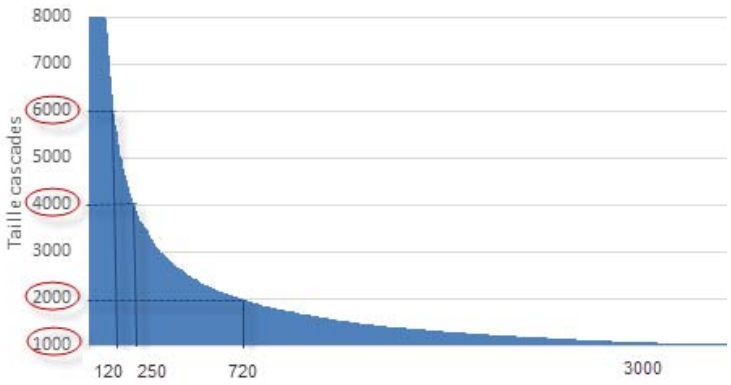


Figure 15

Selon ce jeu de données, nous avons pu déterminer qu'en moyenne une cascade recevait 70% de son nombre total de retweets les 10 premières heures après le post original. *(cfr figure 16)* Nous avons par conséquent décidé, de baser la suite de l'étude sur les 10 premières heures des cascades, en prenant l'instant du post original comme t_0.

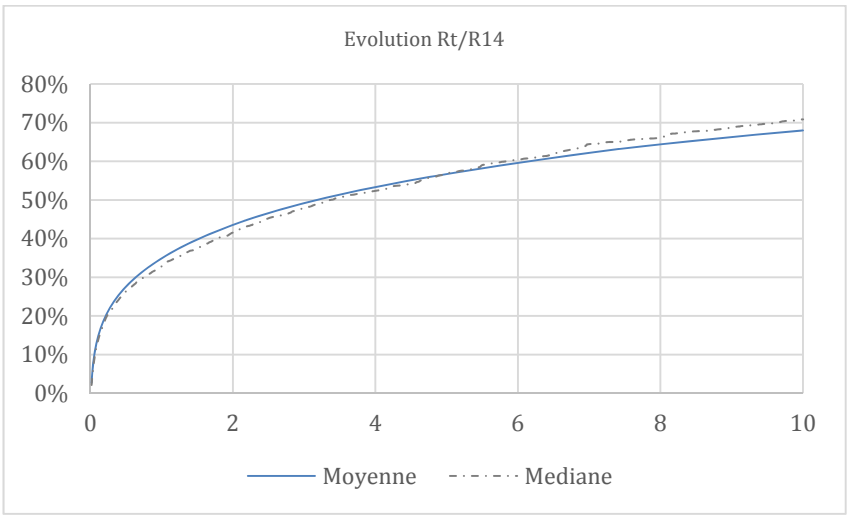


Figure 16

Etude APE selon les catégories de taille

La *figure 17* représente les moyennes des APE pour les différentes catégories de taille. Plusieurs observations peuvent être faites. Premièrement, la période iniale après le post original est instable et l'on retrouve parfois un caractère assez explosif des résultats. Comme mentionné précédemment, le fait d'utiliser un noyau triangulaire à fenêtre croissante de taille t/2 comme pondération de $K_t(s)$, permet au modèle de rapidement négliger cet effet. Deuxièmement, après la période de turbulence initiale, on remarque une tendance décroissante de l'APE pour les différentes catégories. Ici encore cela confirme, l'intuition selon laquelle le fait d'utiliser ce type de noyaux triangulaires permet de prendre en compte des retweets dans une plus large fenêtre au fur et à mesure que le temps *t* s'écoule, et d'ainsi mieux stabiliser les résultats. Finalement, on observe (en ne prenant pas en compte la période instable du début), que l'APE pour les cascades de grandes tailles (supérieure à 6000 retweets) est généralement meilleure que pour les autres catégories. Alors que celle de petites tailles(1000 à 2000 retweets) est généralement moins bonne. Parcontre, les catégories intermédiaires s'entremêlent, n'indiquant pas de réelle distinction entre l'APE, pour les estimations des catégories 2001-4000 et 4001-6000.

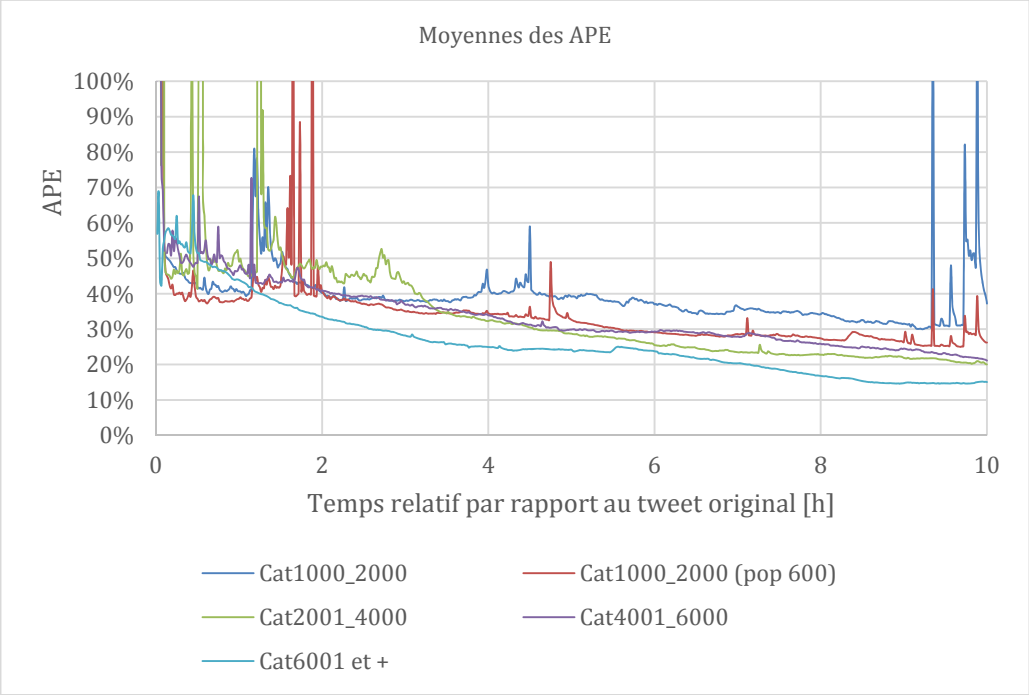


Figure 17

Si l'on juge par les moyennes, on voit un caractère généralement plus instable pour les cascades de petites taille (ex. entre 4 et 5 h, et à partir de 9 h). Pour analyser, cela l'on retrouve également la courbe *Cat1000_2000 (pop 600),* où les APE ont été déterminées sur une population de 600 cascades au lieu de 100. Même si les piques se trouvent diminués, l'on garde ce caractère plus instable. Afin d'être moins influencé par les points extrêmes, un graphe a également été fait pour les médianes de l'APE. *(figure 18)*

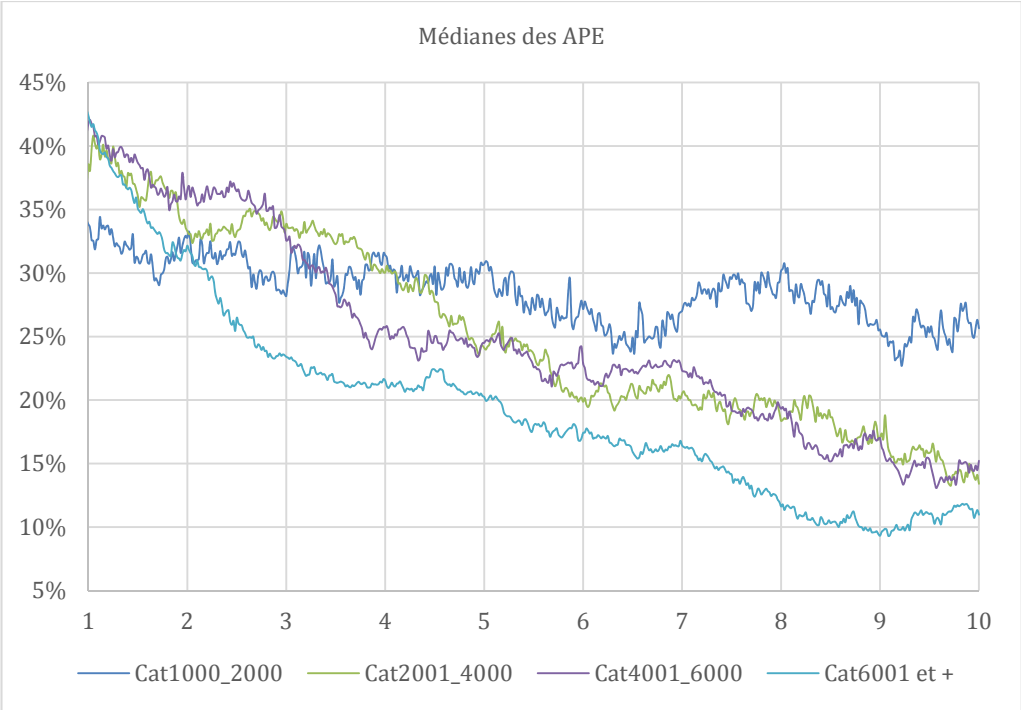


Figure 18

En reprenant la *figure 18* avec les médianes , on retrouve encore logiquement la décroissance de l'APE, mais quand même toujours un caractère assez fluctuant pour les cascade de 1000 à 2000. Après 6h, on observe la médiane de l'APE des cascades supérieures à 6001 retweets, aux alentours de 17% et 11% après 10h, contre respectivement 25% et 15% pour la moyenne.

Afin d'avoir une meilleure idée de la dispersion des données selon les catégories, la *figure 19* nous donne les écart-types respectifs hors zone instable pour (cat1000_2000). On peut constater que la dispersion de l'erreur est effectivement plus stable (plus constante), pour les cascades de grande taille, (à partir de 4000 retweets, alors qu'elle est assez fluctuante en-dessous.

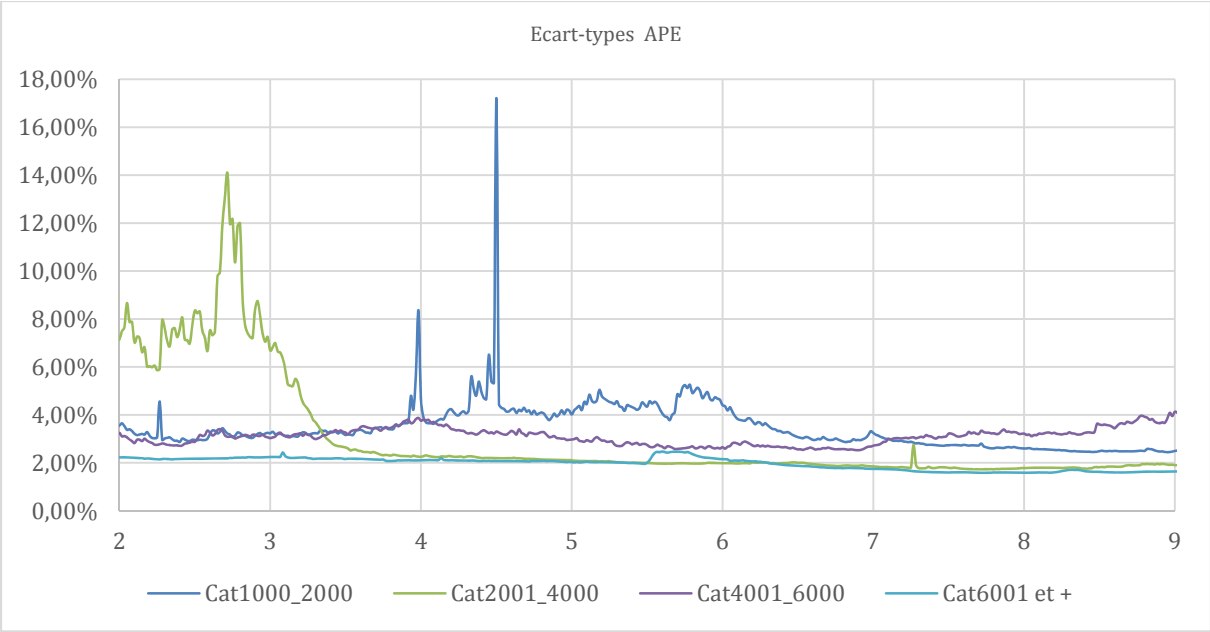
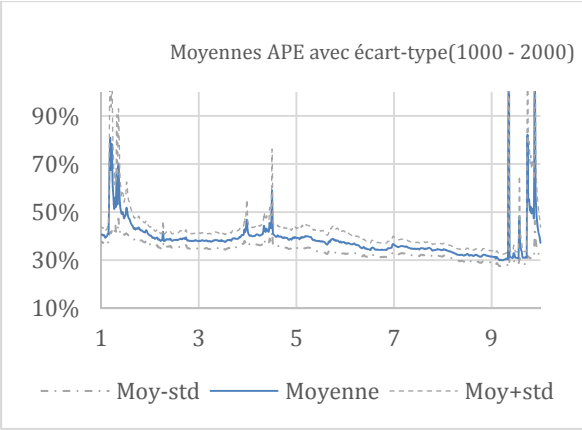
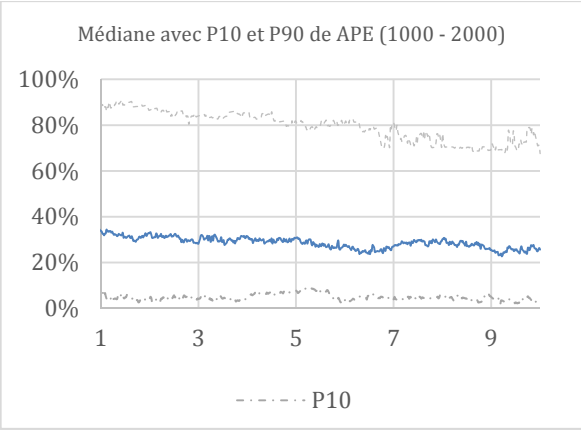
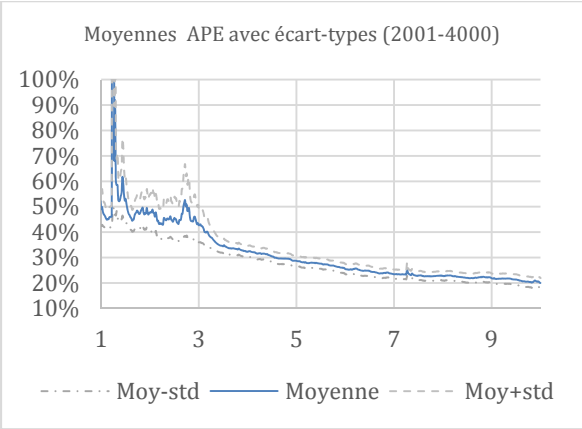
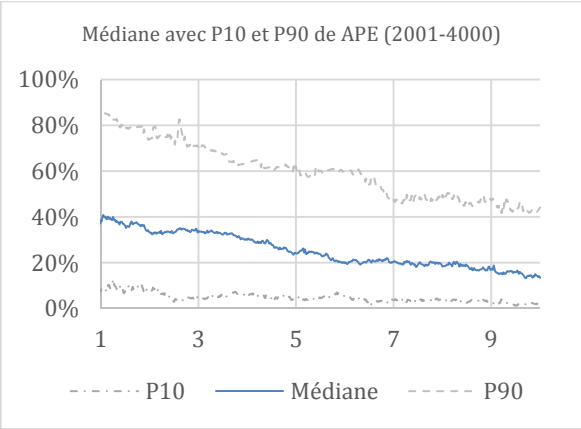

Ecart-types APE
18,00%
16,00%
14,00%
12,00%
10,00%
8,00%
6,00%
4,00%
2,00%
0,00%
2
3
4
5
6
7
8
9
Cat1000_2000
Cat2001_4000
Cat4001_6000
Cat6001 et +

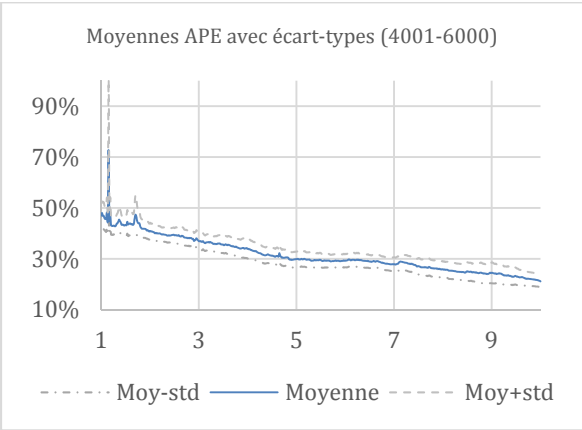
Figure 19

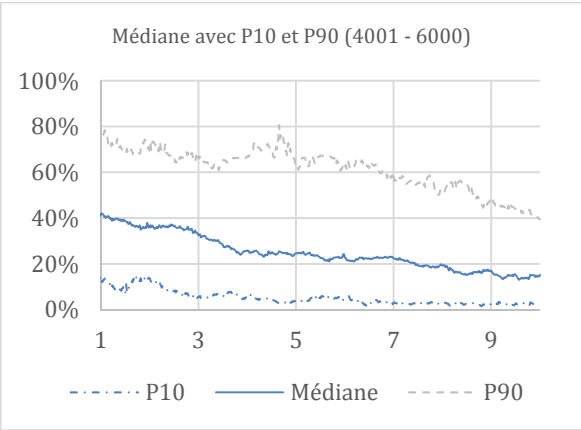

Moyennes APE avec écart-type(1000 - 2000)
90%
70%
50%
30%
10%
1
3
5
7
9
Moy-std
Moyenne
Moy+std

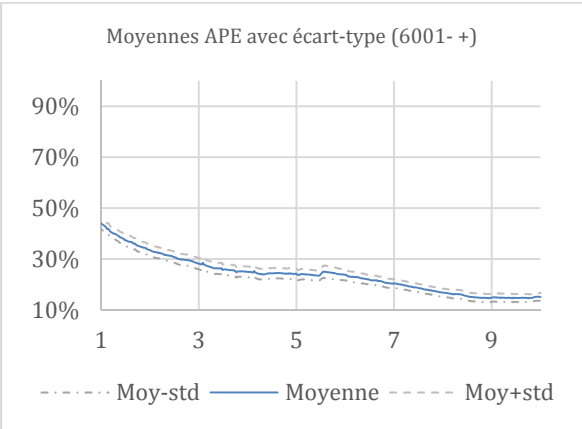

Médiane avec P10 et P90 de APE (1000 - 2000)
100%
80%
60%
40%
20%
0%
1
3
5
7
9
P10

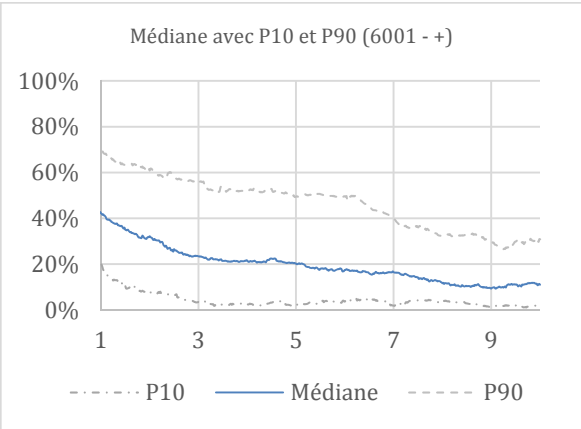

Moyennes APE avec écart-types (2001-4000)
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
1
3
5
7
9
Moy-std
Moyenne
Moy+std


Médiane avec P10 et P90 de APE (2001-4000)
100%
80%
60%
40%
20%
0%
1
3
5
7
9
P10
Médiane
P90


Moyennes APE avec écart-types (4001-6000)
90%
70%
50%
30%
10%
1
3
5
7
9
Moy-std
Moyenne
Moy+std


Médiane avec P10 et P90 (4001 - 6000)
100%
80%
60%
40%
20%
0%
1
3
5
7
9
P10
Médiane
P90

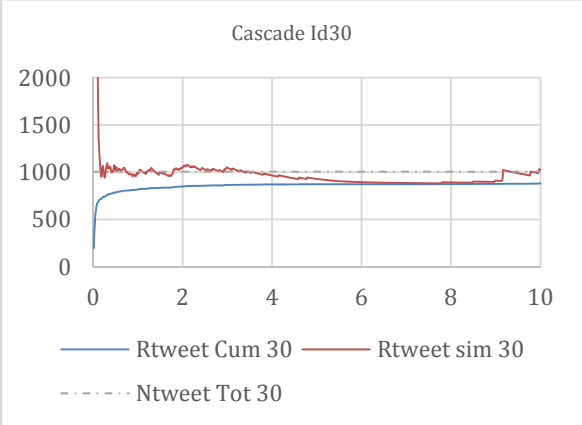

Moyennes APE avec écart-type (6001- +)
90%
70%
50%
30%
10%
1
3
5
7
9
Moy-std
Moyenne
Moy+std

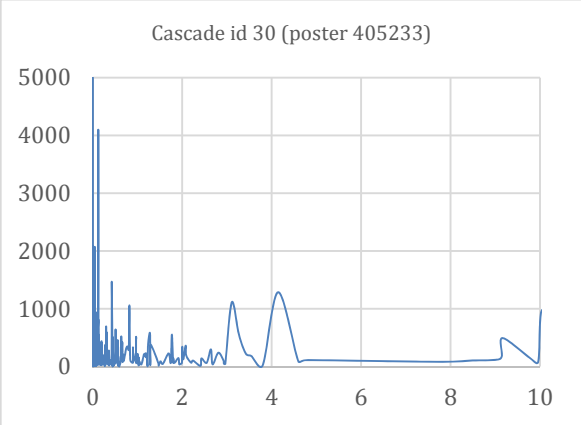

Médiane avec P10 et P90 (6001 - +)
100%
80%
60%
40%
20%
0%
1
3
5
7
9
P10
Médiane
P90

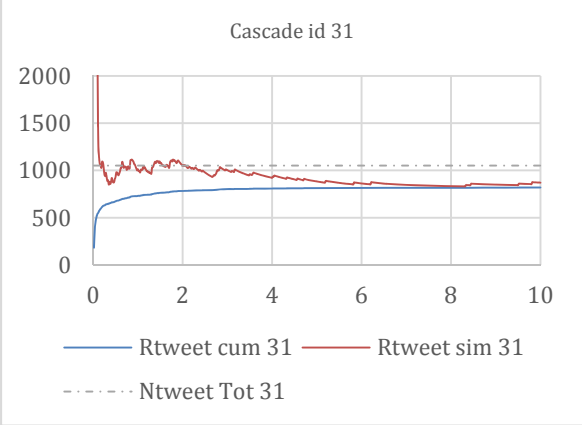
Ci-dessous suivent pour chaque catégorie de taille des exemples de cascades avec une APE stable (vues comme bon) et d'autres avec une mauvaise APE. Comme légende nous avons, *Rtweet Cum*, indiquant le nombre réelle de retweets cumulés, *Rtweet sim*, donnant le résultat de la simulation SEISMIC, *et Ntweet Tot,* le nombre de retweets total. Egalement representé dans les graphes *Cascade id… (poster…),* sont le nombre de *followers* des personnes retweetant le post original. Et finalement, le nombre de retweets relatifs par rapport au nombre de retweets total.

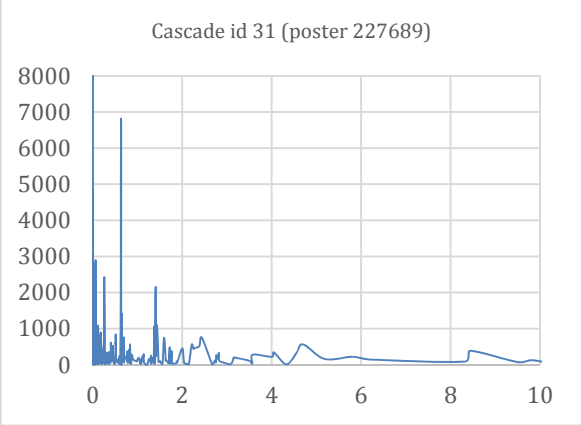
Cascades 1000 – 2000

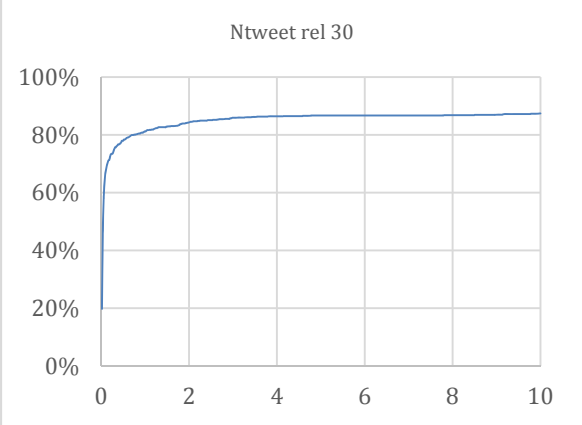
Exemples cascades avec bon APE

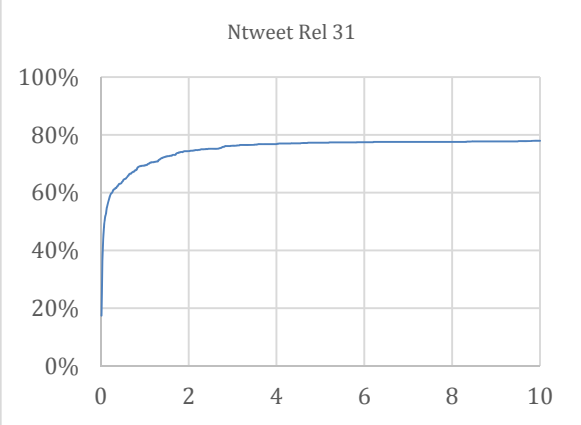




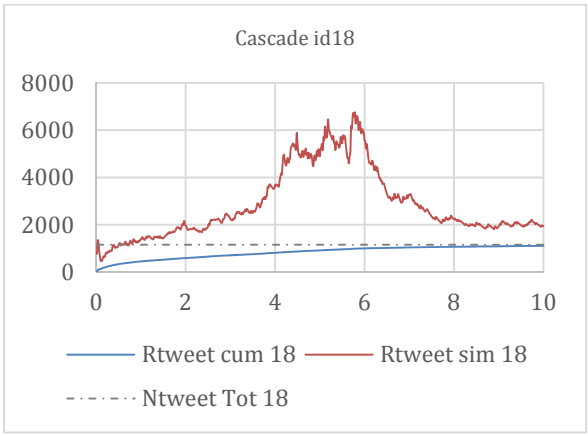


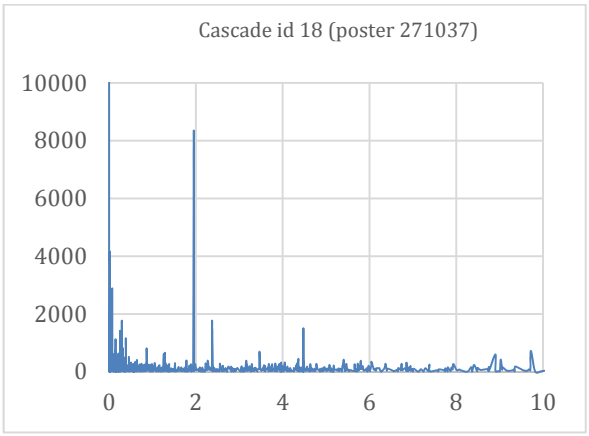


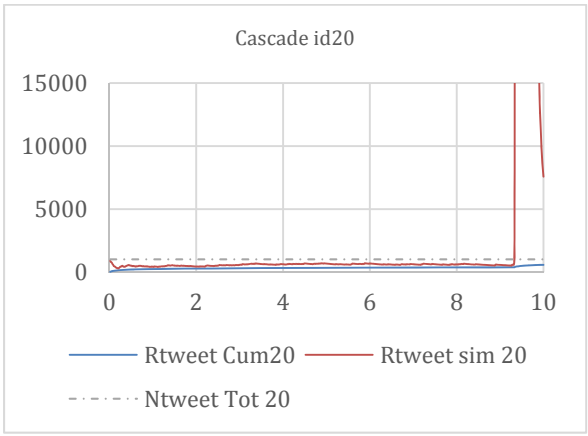


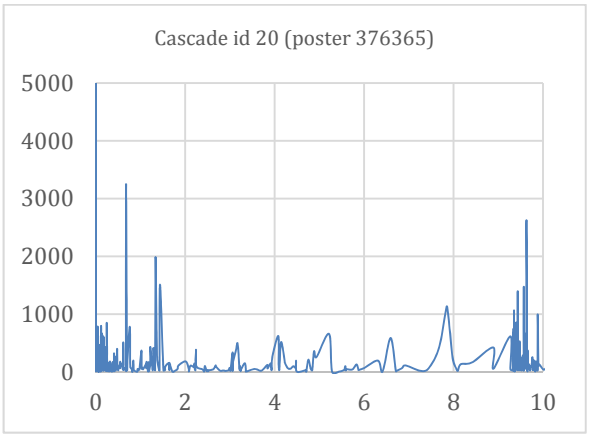


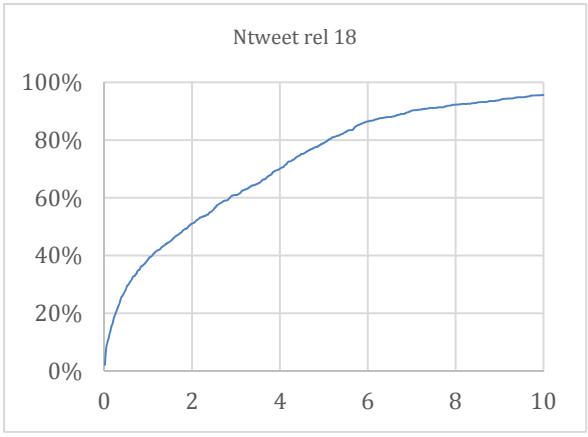
Exemples cascades avec mauvais APE

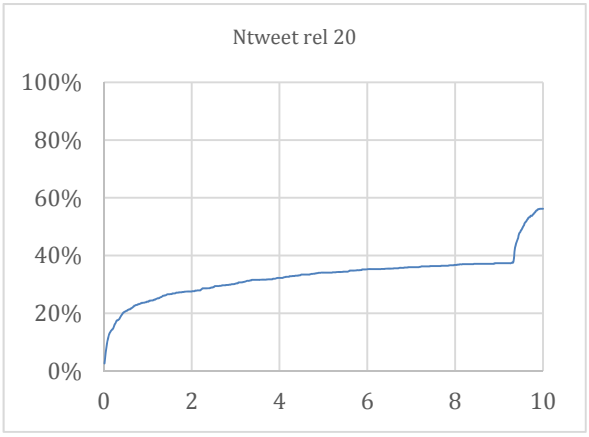






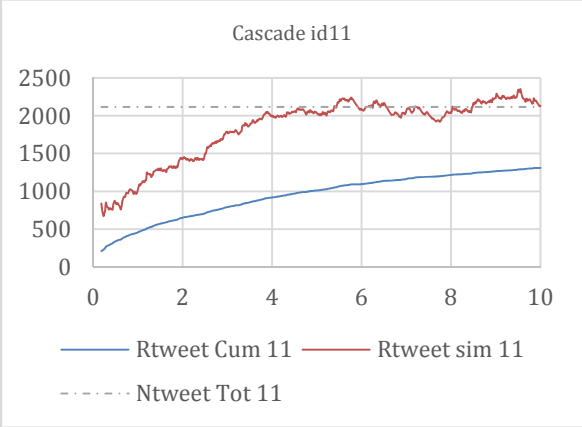


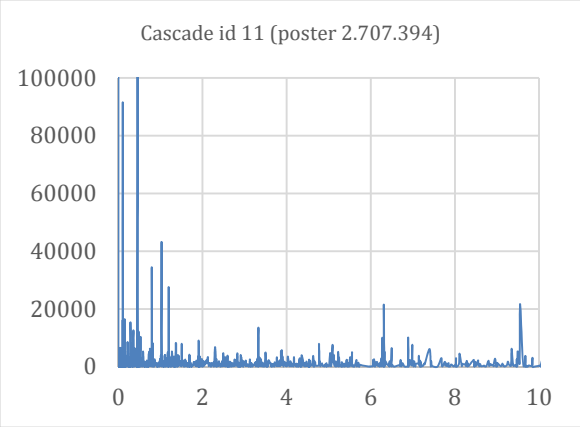


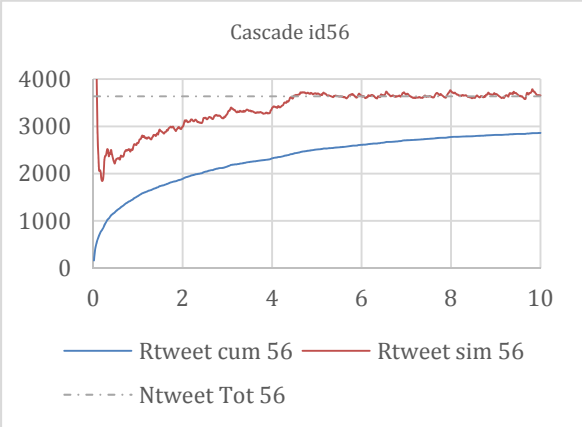


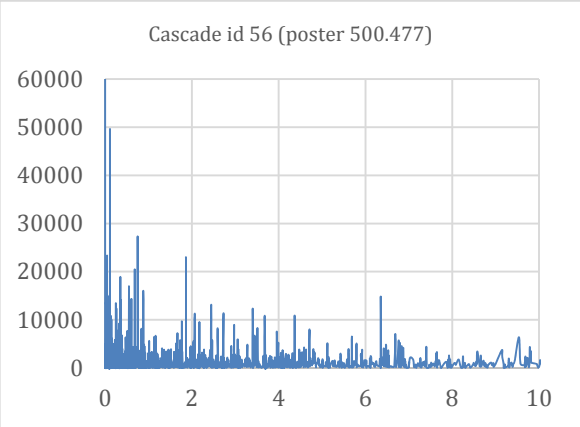
Cascades 2001 - 4000

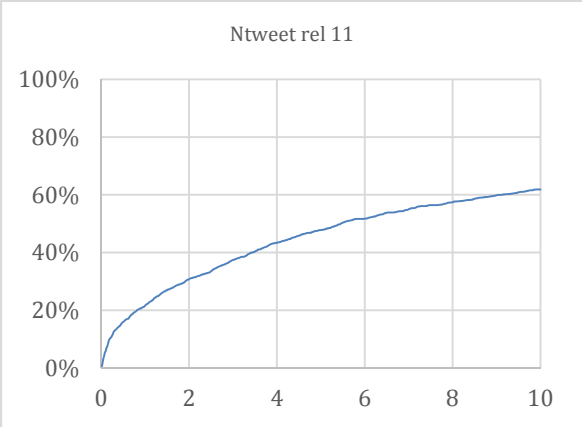
Exemples cascades avec bon APE

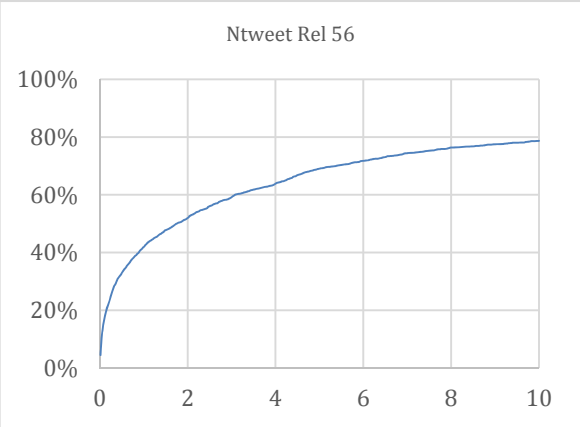




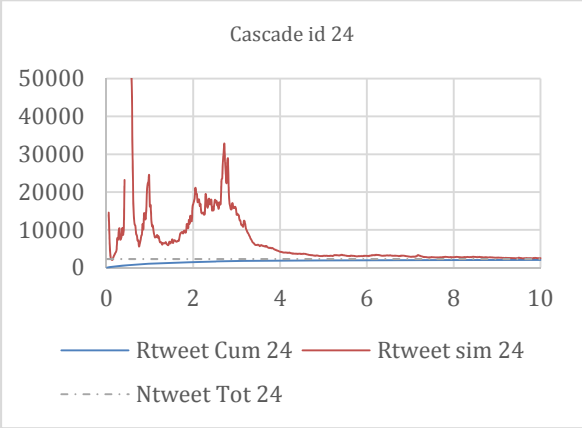


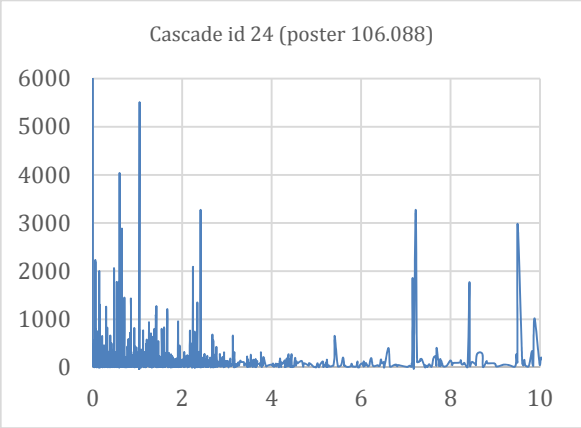


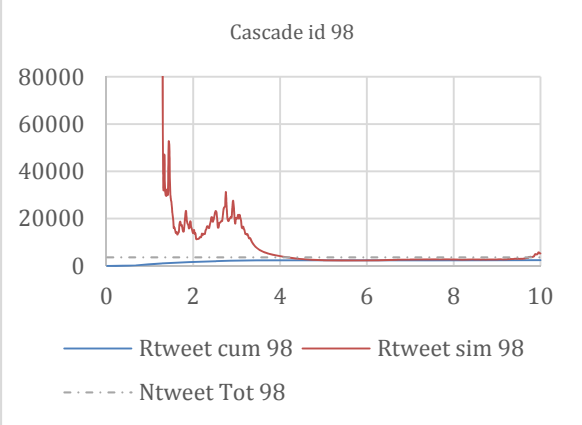


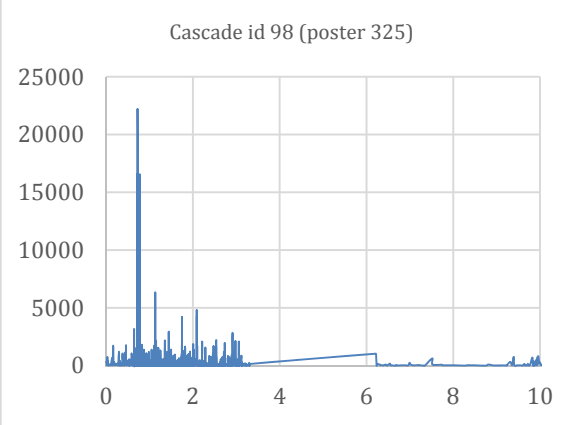


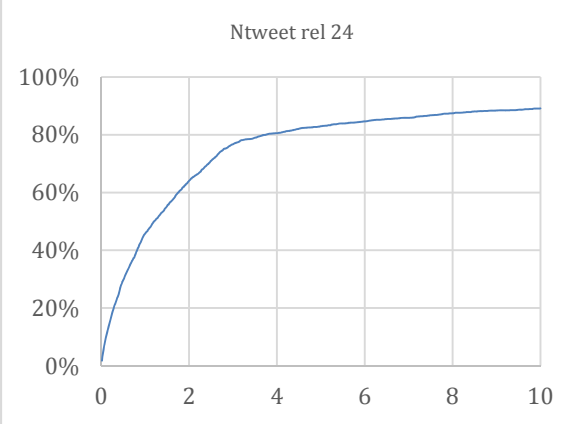
Exemples cascades avec mauvais APE

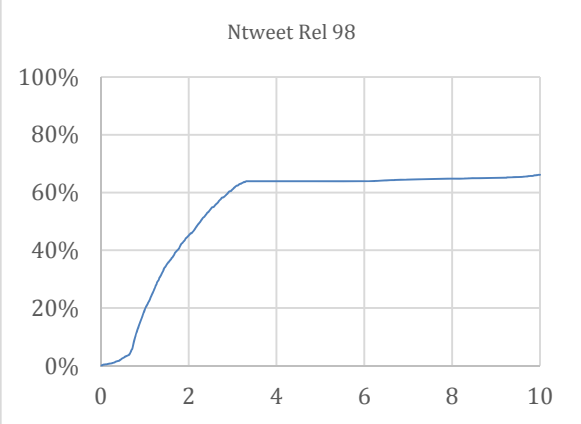






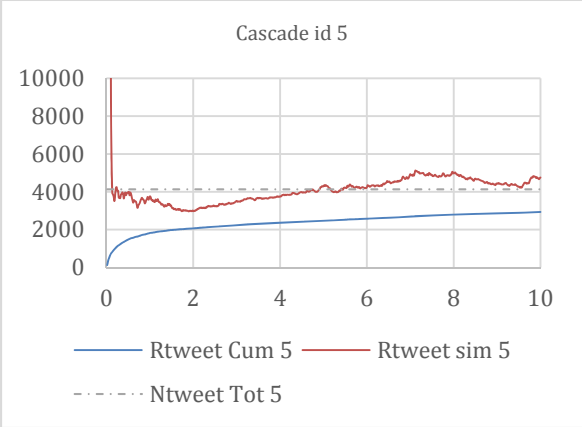


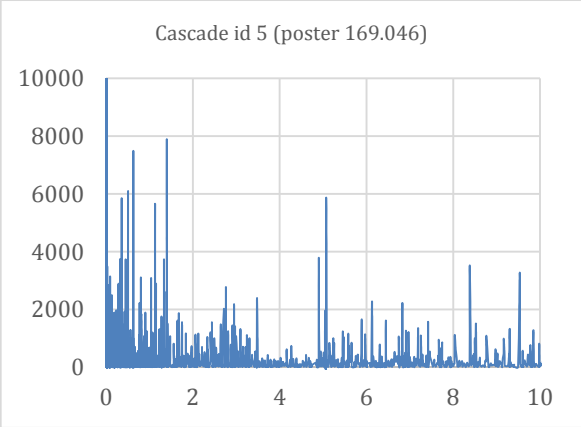


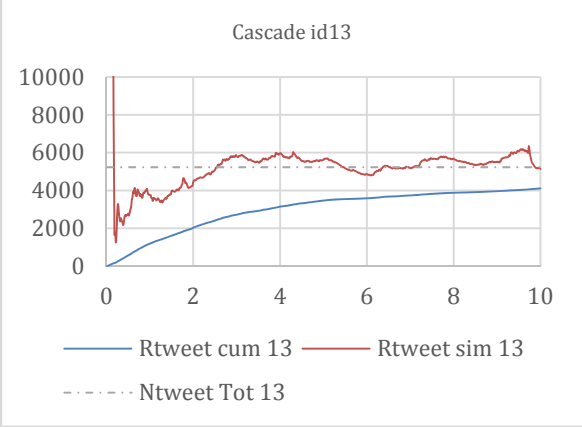


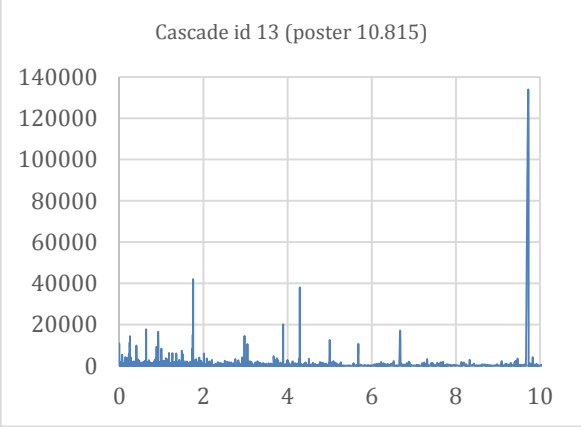
Cascades 4001 - 6000

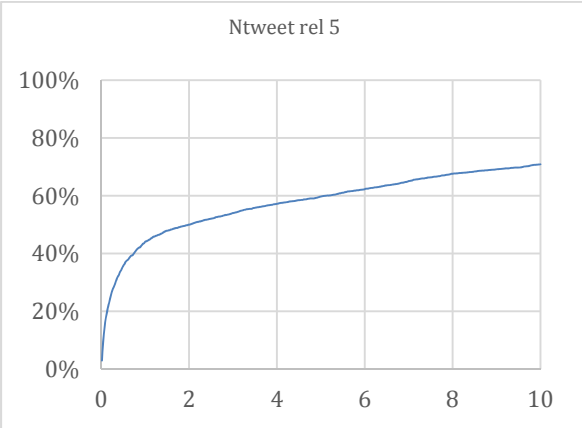
Exemples cascades avec bon APE

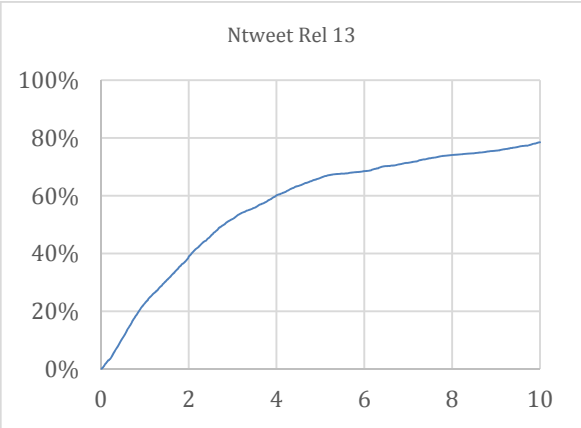




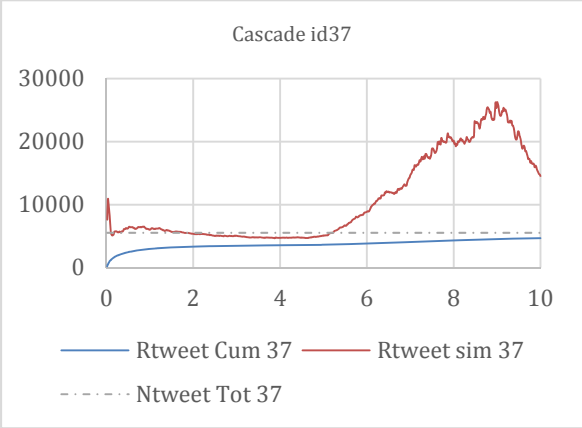


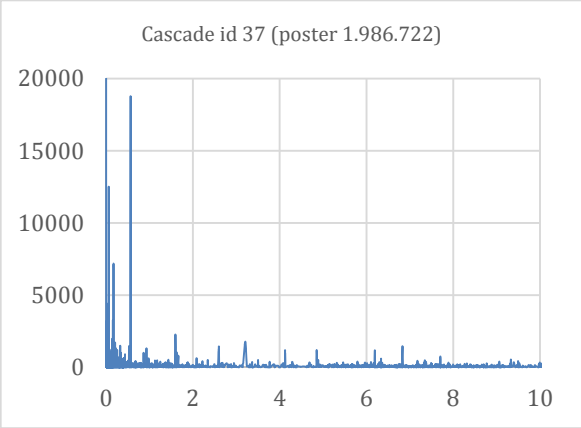


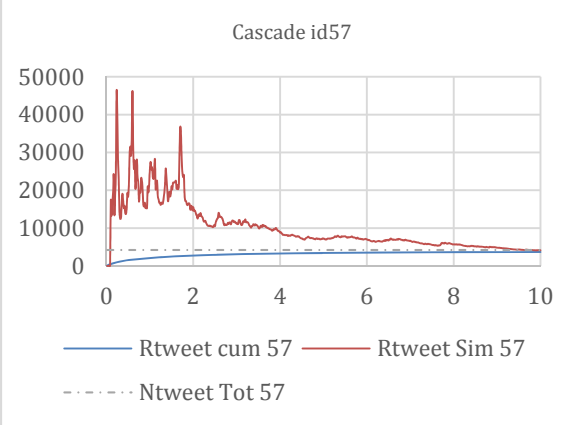


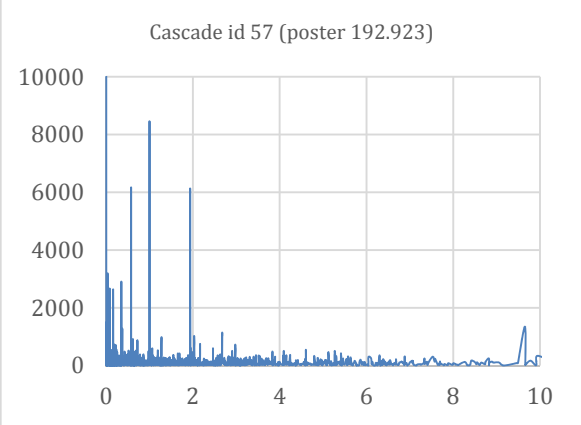


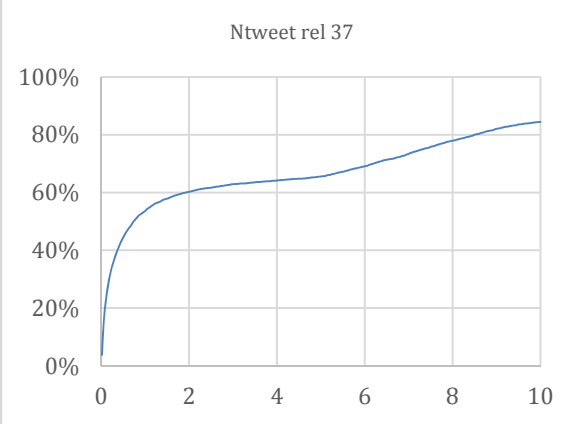
Exemples cascades avec mauvais APE

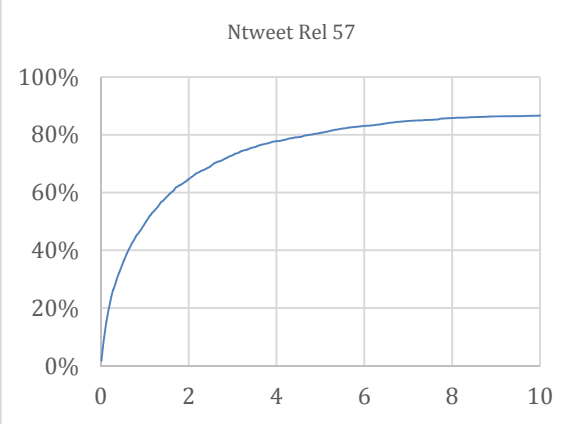






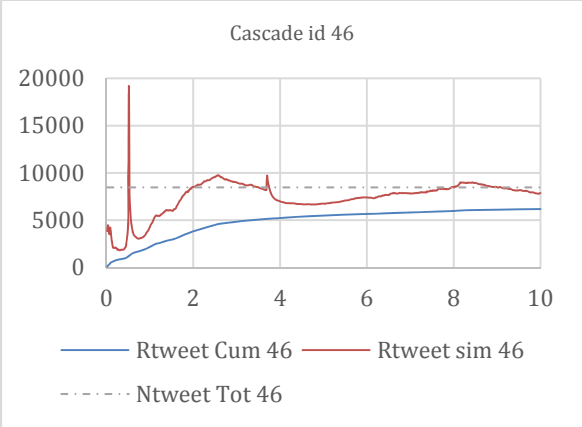


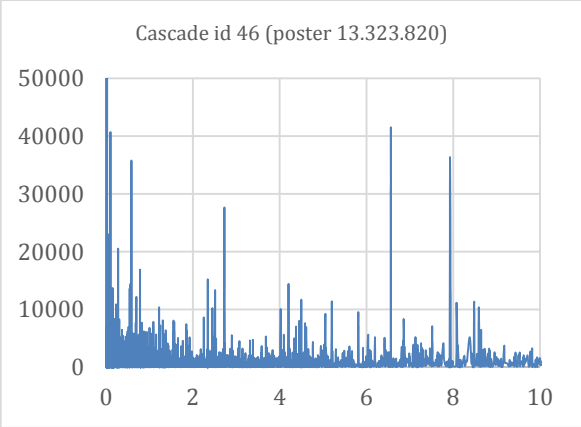


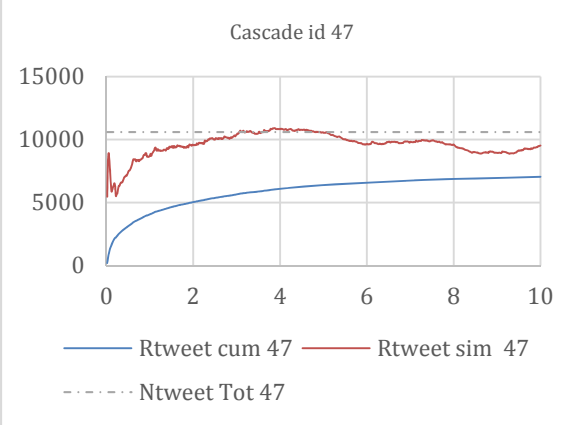


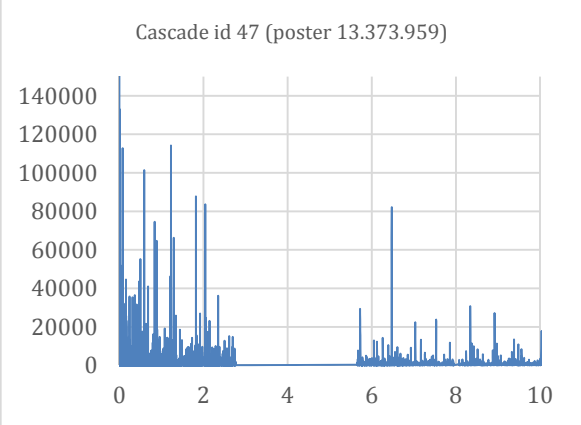
Cascades 6000 - +

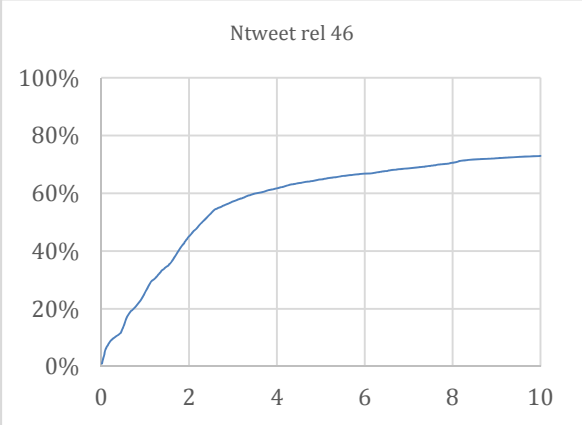
Exemples cascades avec bon APE

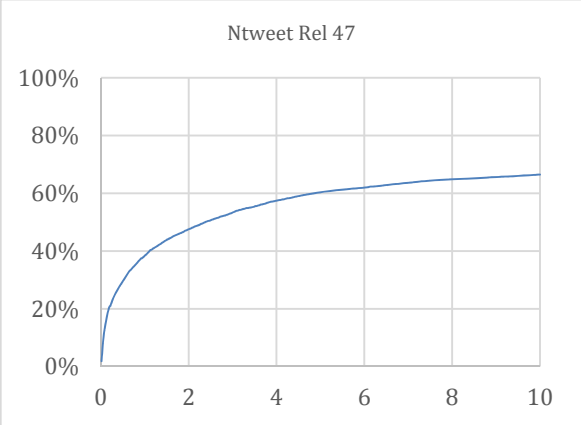




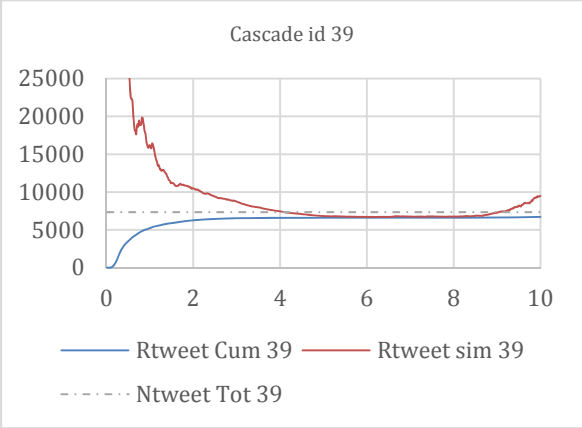


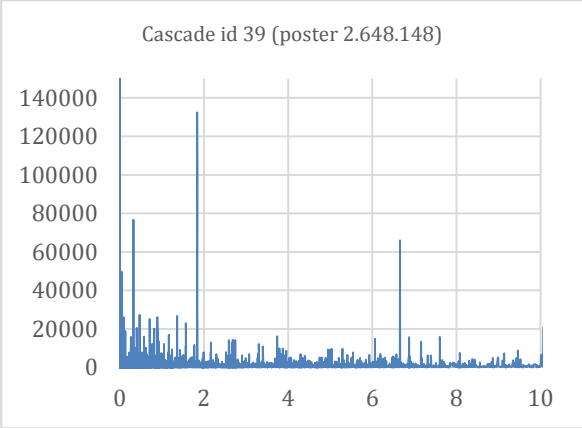


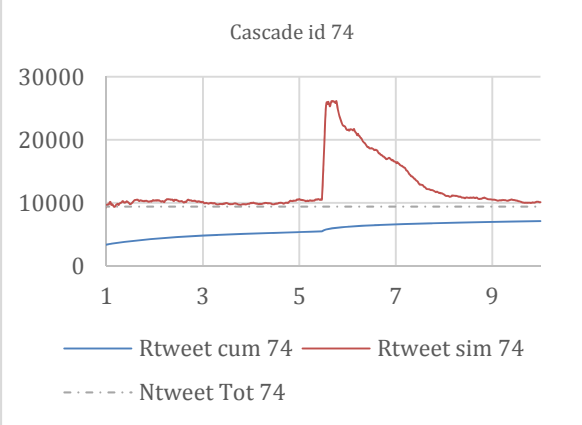


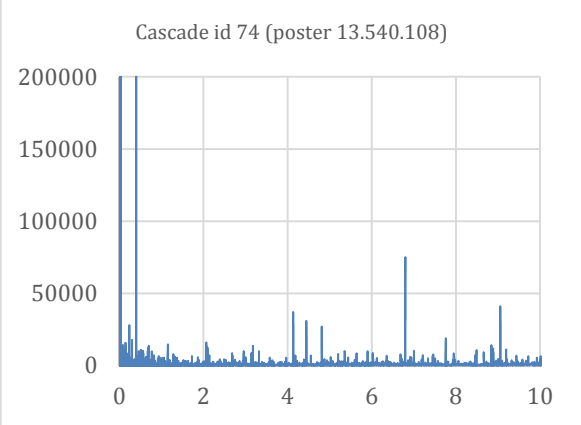


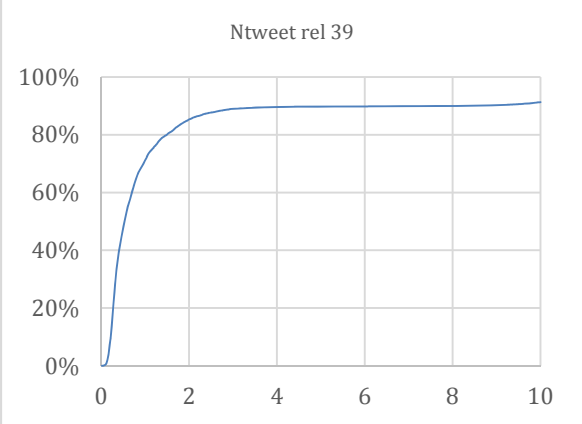
Exemples cascades avec mauvais APE

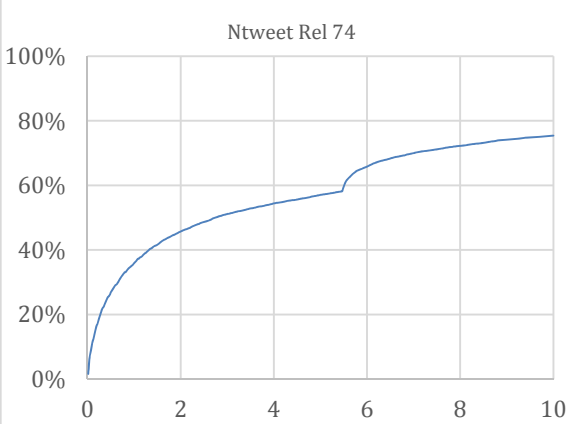












Bien que différents paramètres aient été étudiées (répartitions du nombre de followers, nombre de followers du poster original, pente de l'évolution du nombre de retweets relatifs), aucun n'a pu réellement mettre un profil de cascade en avant donnant lieu soit à une bonne ou mauvaise APE.

5. Amélioration du modèle SEISMIC

Time-Dependent Hawkes process:

Ref : TiDeH: Time-Dependent Hawkes process for predicting retweet dynamics , Ryota Kobayashi/ R. Lambiotte

Après s'être penché sur le modèle SEISMIC, intéressons-nous maintenant de plus près à une amélioration de celui-ci. R. Kobayashi et R. Lambiotte, n'ont pas seulement étudié la question du nombre final de retweets, mais également celle de l'évolution de la popularité des tweets dans le temps, en se basant sur une fenêtre d'observation. Pour ce faire, les séries temporelles sont modélisées par un « Time-Dependent Hawkes process ». TiDeH, généralise le modèle classique des « self-exciting point processes ». Un des avantages du processus de Hawkes dans ce contexte, par rapport aux processus de Poisson sans mémoire, est que le futur degré d'activité est *stimulé* par l'occurrence d'événements dans le passé. Ils peuvent être vus comme une généralisation des modèles de prédiction épidémiologique où l'on ajoute un « memory kernel », déterminant le temps entre une cause (par exemple un tweet) et son effet (un retweet). La *nature contagieuse* des processus de Hawkes, traduit bien le fait qu'un retweet supplémentaire expose de nouveaux followers et peut ainsi mener à de *nouveaux retweets* dans le futur. Ici les auteurs, ont rendu le processus de Hawkes dépendant du temps, en permettant le paramètre du modèle de varier quotidiennement.

La méthodologie établie est la suivante : Pour un tweet particulier, l'on observe sa séquence de retweets $\{t_i,d_i\}$ jusqu'au temps $t_0 + T$, où t_i est le $i^{ième}$ temps retweeté, d_i le nombre de followers de la $i^{ième}$ personne retweetant, t_0 l'instant du post original, d_0 le nombre de followers du *poster* original et T la période de l'observation. Les paramètres de TiDeH sont tout d'abord ajustés en fonction de la série temporelle des retweets *(figure 20)* ainsi que de la répartition du nombre de followers.

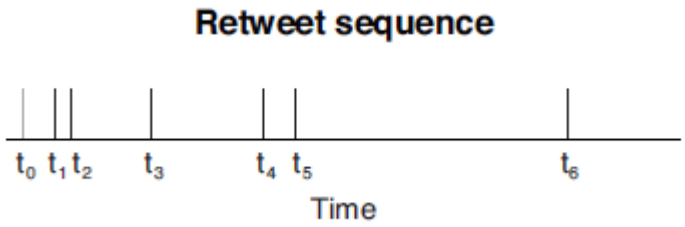
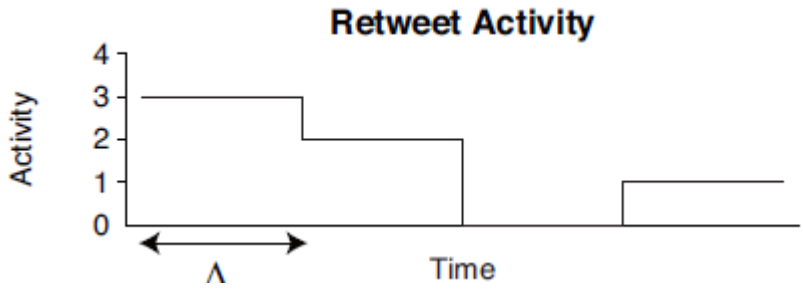


Figure 20

Ensuite l'activité de reweet (*retweet activity*), définie comme le nombre de retweets dans la $k^{ième}$ fenêtre temporelle $t \in [(k-1)\Phi_{pred}, k\Phi_{pred}[$, est estimée, où Φ_{pred} représente la largeur de la fenêtre temporelle et par conséquent la résolution temporelle de la prédiction.



Le but étant de prédire l'activité de retweet future à partir des instants de retweet observés et des nombres de followers jusqu'à un temps $t_0 + T$.

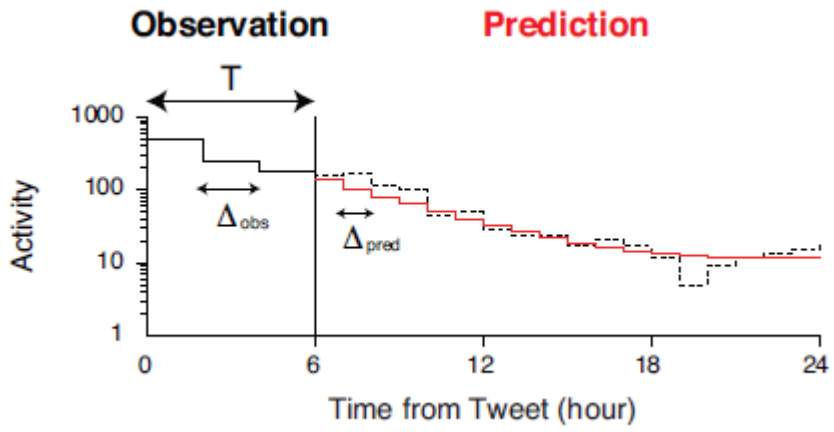


Figure 21

Une distinction importante entre TiDeH et les modèles existants, et sa dépendance au temps, comme il tient compte des rythmes circadiens (oscillation de périodicité d'environ 24 heures) de la popularité en ligne ainsi que de l'obsolescence de l'information avec le temps. Il est important de constater que l'*infectiousness* du tweet original dépend tout naturellement de l'instant auquel il a été posté et cet effet se retrouve également pour les retweets ultérieurs. Comme illustré par la *figure 22* représentant l'estimation de l'*infectiousness* de séquences temporelles de retweets, on observe 2 types de dynamiques, une décroissance (A) et une décroissance à oscillations circadiennes (B). Les courbes noires sont les taux estimés avec la fenêtre de temps se déplaçant dans le temps alors que celles en rouge sont un fit à l'aide du modèle proposé.

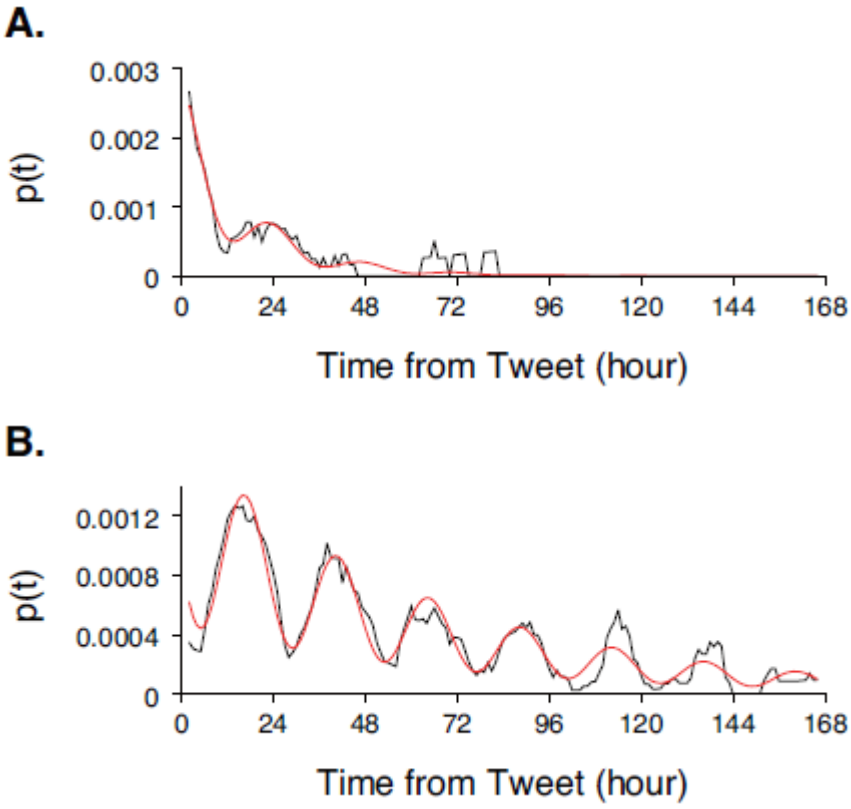


Figure 22

Le modèle TiDeH, peut être considéré comme une extension de SEISMIC avec une dépendance au temps, comme il incorpore une information partielle sur la structure du réseau. Et le but ici est de prédire l'évolution temporelle du nombre de retweets dans le temps, et pas uniquement le nombre de retweets final.

Dans cet article, la même base de données a été utilisée que pour SEISMIC à savoir 166 076 tweets apparus sur Twitter du 7 octobre au 7 novembre 2011.

Les auteurs ont examiné la dépendance de la performance de prédiction par rapport au temps T d'observation. La *figure 23* nous donne le résultat de la comparaison de la performance de prédiction pour différents modèles (LR : Linear Regression , LR-N : Linear regression with the number of followers , RPP : Reinforcement poisson process et TiDeH) avec un Tmax = 168 heures par rapport au tweet original, et une fenêtre de taille Φ_{pred} = 4 heures. On observe que TiDeH donne de meilleurs résultats dans tous les régimes, tant après 1 heure qu'après 48 heures. Ce qui est également important de remarquer, c'est que l'erreur augmente lorsque le temps d'observation diminue. Parcontre cette augmentation de l'erreur est minimale pour TiDeH.

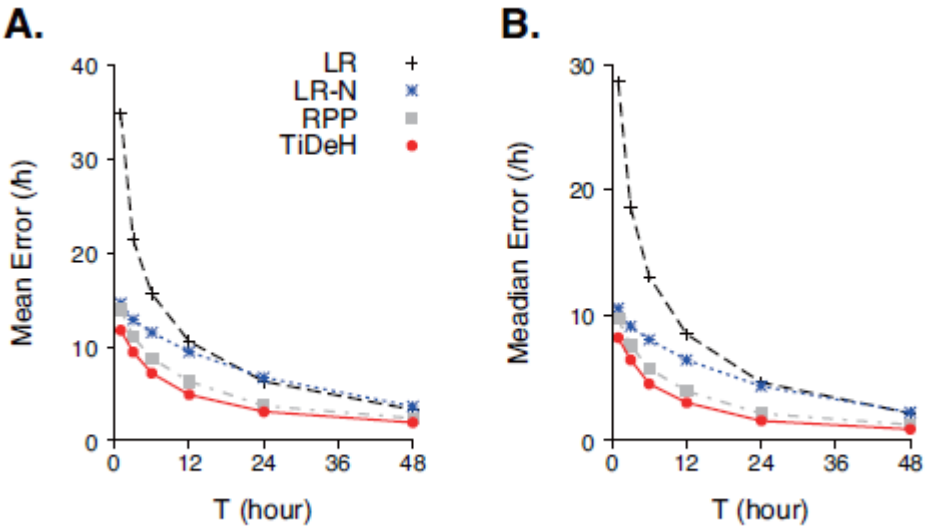
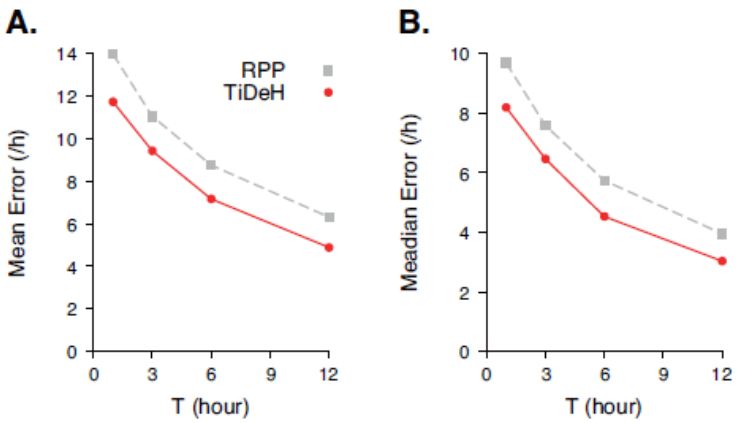


Figure 23



A côté de cela une comparaison a également été faite par rapport au nombre final de retweets, et cette fois en incorporant SEISMIC. La *figure 24,* nous montre cette comparaison, et l'on observe qu'à nouveau TiDeH est plus précis, en améliorant les résultats par rapport à RPP et SEISMIC d'environ 30%.

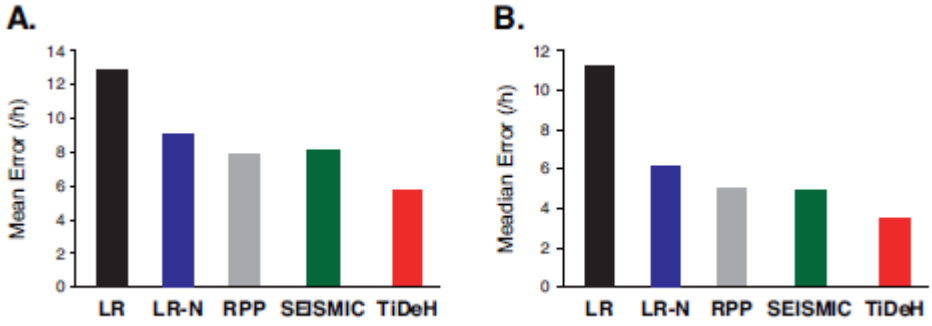


Figure 24

6. Conclusion

Au cours de ce travail, nous nous sommes tout d'abord intéressés aux différentes recherches réalisées dans le domaine de la pédiction de cascades d'information. Nous avons pu nous rendre compte que 2 grandes familles d'algorithmes co-existaient, avec d'un côté les « Featured Based Methods » et de l'autre les « Point Based Process Methods ».

La première catégorie, requiérant l'identification de *caractéristiques clefs* des cascades étudiées, dépend fortement du choix de ces paramètres quant à ses performances prédictives. De plus, comme elle nécessite le traîtement d'une quantité massive de données, elle ne peut être utilisée en temps réel. Ce qui rend cette catégorie d'algorithmes beaucoup moins adéquate, surtout lorsqu'on souhaite utiliser ses résultats en marketing viral par exemple. Toutefois, certaines observations intéressantes découlent de l'étude de ces *caractéristiques clefs* à savoir que le contenu d'un message, n'a qu'un faible effet sur le caractère viral ou non de celui-ci. Ce qui par contre ressort régulièrement, est le fait que le degré du nœud, c'est-à-dire le nombre de *followers* dans le cas de Twitter, est quant à lui un paramètre déterminant.

La deuxième catégorie, dont fait partie SEISMIC, cherche à modéliser directement la formation d'une cascade d'information dans un réseau. Cela résulte dès lors en une équation calculable en temps réel, sans toute une étude appronfondie des caractéristiques. Nous avons ensuite détaillé le modèle SEISMIC, qui a un « post infectiousness » variable en fonction du temps. Ce modèle, n'utilise que très peu d'informations concernant le réseau, vu qu'il se base principalement sur le degré des nœuds (nombre de *followers* des individus retweetant le post original en fonction du temps).

Le modèle SEISMIC a été appliqué à différentes tailles de cascades (taille minimale de 1000 retweets), ce qui a permis de réaliser une étude statistique de l'Absolute Percentage Error (APE) selon la taille des cascades. Il s'est avéré que la performance du modèle, serait meilleure pour des cascades atteignant des tailles plus importantes (supérieures à 6000 retweets). La performance du modèle appliquée à des cascades de tailles inférieures à 2000 retweets montrait également une plus grande dispersion. L'idée ensuite pour chaque catégorie de taille, était d'étudier des cascades donnant lieu d'une part à une bonne APE et d'autres part à une mauvaise APE, et d'essayer de retrouver un certain schéma, quant aux types de cascades favorisant ou non la performance du modèle. Malheureusement, cet aspect de l'étude n'a pas aboutit comme souhaité, et serait une des pistes possibles d'approfondissement, tout comme l'affinement éventuel des bornes des catégories de tailles.

Finalement, nous avons souhaité terminer ce travail par l'étude d'une amélioration du modèle SEISMIC, du nom de TiDeH. Celui-ci ne cherchait pas uniquement à déterminer le nombre final de retweets, mais également l'évolution temporelle de la cascade.

7. ANNEXES

```
#' Predicting information cascade by self-exciting point process model
#'
#' This package implements a self-exciting point process model for information cascades.
#' An information cascade occurs when many people engage in the same acts after observing
#' the actions of others. Typical examples are post/photo resharings on Facebook and retweets
#' on Twitter. The package provides functions to estimate the infectiousness of an
#' information cascade and predict its popularity given the observed history.
#' For more information, see
#' \url{http://snap.stanford.edu/seismic/}.
#'
#' @docType package
#' @name seismic
#' @references SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity by Q. Zhao, M. Erdogdu, H. He, A. Rajaraman, J. Leskovec, ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2015.
NULL


#' Memory kernel
#'
#' Probability density function and complementary cumulative distribution function
#' for the human reaction time.
#' @keywords internal
#'
#' @param t time
#' @param theta exponent of the power law
#' @param cutoff the cutoff value where the density changes from constant to power law
#' @param c the constant density when t is less than the cutoff
#' @return the density at t
#' @details default values are measured from a real Twitter data set.
#' @return \code{memory.pdf} returns the density function at t.
#' \code{memory.ccdf} returns the ccdf (probabilty of greater than t).
#'
memory.pdf <- function(t, theta=0.2314843, cutoff=300, c=0.0006265725) {
 if (t < cutoff)
  return(c)
 else
  return(c*exp((log(t) - log(cutoff))*(-(1+theta))))
```

```
}

#' @describeIn memory.pdf
memory.ccdf <- function(t, theta=0.2314843, cutoff=300, c=0.0006265725) {
 t[t<0] <- 0
 index1 <- which(t <= cutoff)
 index2 <- which(t > cutoff)
 ccdf <- rep(0, length(t))
 ccdf[index1] <- 1 - c*t[index1]
 ccdf[index2] <-  c*cutoff^(1+theta)/theta*(t[index2]^(-theta))
 ccdf
}

#' Integration with respect to locally weighted kernel
#'
#' @keywords internal
#'
#' @param t1 a vector of integral lower limit
#' @param t2 a vector of integral upper limit
#' @param ptime the time (a scalar) to estimate infectiousness and predict for popularity
#' @param slope slope of the linear kernel
#' @param window size of the linear kernel
#' @inheritParams memory.pdf
#' @inheritParams get.infectiousness
#' @return \code{linear.kernel} returns the integral from vector t1 to vector t2 of
#' c*[slope(t-ptime) + 1];
#' \code{power.kernel} returns the integral from vector t1 to vector 2 of c*((t-share.time)/cutoff)^(-(1+theta))[slope(t-ptime) + 1];
#' \code{integral.memory.kernel} returns the vector with ith entry being integral_-inf^inf phi_share.time[i]*kernel(t-p.time)
#' @seealso \code{\link{memory.pdf}}
linear.kernel <- function(t1, t2, ptime, slope, c=0.0006265725){
 ## indefinite integral is c*(t-ptime*slope*t+(slope*t^2)/2)
 return(c*(t2-ptime*slope*t2+slope*t2^2/2) - c*(t1-ptime*slope*t1+slope*t1^2/2))
}

#' @describeIn linear.kernel
power.kernel <- function(t1, t2, ptime, share.time, slope, theta=0.2314843, cutoff=300, c=0.0006265725){
 return(c*cutoff^(1+theta)*(t2-share.time)^(-theta)*(share.time*slope-theta+(theta-1)*ptime*slope-theta*slope*t2+1)/((theta-1)*theta) - c*cutoff^(1+theta)*(t1-share.time)^(-theta)*(share.time*slope-theta+(theta-1)*ptime*slope-theta*slope*t1+1)/((theta-1)*theta))
}
```

```
#' @describeIn linear.kernel

integral.memory.kernel <- function(p.time, share.time, slope, window, theta=0.2314843, cutoff=300, c=0.0006265725){

 index1 <- which(p.time <= share.time)

 index2 <- which(p.time > share.time & p.time <= share.time + cutoff)

 index3 <- which(p.time > share.time + cutoff & p.time <= share.time + window)

 index4 <- which(p.time > share.time + window & p.time <= share.time + window + cutoff)

 index5 <- which(p.time > share.time + window + cutoff)

 integral <- rep(NA, length(share.time))

 integral[index1] <- 0

 integral[index2] <- linear.kernel(share.time[index2], p.time, p.time, slope)

 integral[index3] <- linear.kernel(share.time[index3], share.time[index3] + cutoff, p.time, slope) +
power.kernel(share.time[index3]+cutoff, p.time, p.time, share.time[index3], slope)

 integral[index4] <- linear.kernel(p.time-window, share.time[index4]+cutoff, p.time, slope) +
power.kernel(share.time[index4]+cutoff, p.time, p.time, share.time[index4], slope)

 integral[index5] <- power.kernel(p.time-window, p.time, p.time, share.time[index5], slope)

 return(integral)

}


#' Estimate the infectiousness of an information cascade

#'

#' @param share.time observed resharing times, sorted, share.time[1] =0

#' @param degree observed node degrees

#' @param p.time equally spaced vector of time to estimate the infectiousness, p.time[1]=0

#' @param max.window maximum span of the locally weight kernel

#' @param min.window minimum span of the locally weight kernel

#' @param min.count the minimum number of resharings included in the window

#' @details Use a triangular kernel with shape changing over time. At time p.time, use a triangluer kernel with slope =
min(max(1/(\code{p.time}/2), 1/\code{min.window}), \code{max.window}).

#' @return a list of three vectors: \itemize{

#' \item infectiousness. the estimated infectiousness

#' \item p.up. the upper 95 percent approximate confidence interval

#' \item p.low. the lower 95 percent approximate confidence interval

#' }

#' @export

#' @examples

#' data(tweet)

#' pred.time <- seq(0, 6 * 60 * 60, by = 60)

#' infectiousness <- get.infectiousness(tweet[, 1], tweet[, 2], pred.time)

#' plot(pred.time, infectiousness$infectiousness)
```

```
get.infectiousness <- function(share.time,
                  degree,
                  p.time,
                  max.window = 2 * 60 * 60,
                  min.window = 300,
                  min.count = 5) {

 ix <- sort(share.time, index.return=TRUE)$ix
 share.time <- share.time[ix]

 slopes <- 1/(p.time/2)
 slopes[slopes < 1/max.window] <- 1/max.window
 slopes[slopes > 1/min.window] <- 1/min.window

 windows <- p.time/2
 windows[windows > max.window] <- max.window
 windows[windows < min.window] <- min.window

 for(j in c(1:length(p.time))) {
  ind <- which(share.time >= p.time[j] - windows[j] & share.time < p.time[j])
  if(length(ind) < min.count) {
   ind2 <- which(share.time < p.time[j])
   lcv <- length(ind2)
   ind <- ind2[max((lcv-min.count),1):lcv]
   slopes[j] <- 1/(p.time[j] - share.time[ind[1]])
   windows[j] <- p.time[j] - share.time[ind[1]]
  }
 }

 M.I <- matrix(0,nrow=length(share.time),ncol=length(p.time))
 for(j in 1:length(p.time)){
  M.I[,j] <- degree*integral.memory.kernel(p.time[j], share.time, slopes[j], windows[j])
 }
 infectiousness.seq <- rep(0, length(p.time))
 p.low.seq <- rep(0, length(p.time))
 p.up.seq <- rep(0, length(p.time))
 share.time <- share.time[-1]        #removes the original tweet from retweet
 for(j in c(1:length(p.time))) {
  share.time.tri <- share.time[which(share.time >= p.time[j] - windows[j] & share.time < p.time[j])]
```

```
  rt.count.weighted <- sum(slopes[j]*(share.time.tri - p.time[j]) + 1)
  #print(paste("p.time[i]", p.time[j], "rt.num", length(share.time.tri)))
  I <- sum(M.I[,j])
  rt.num <- length(share.time.tri)
  if (rt.count.weighted==0)
    next
  else {
    infectiousness.seq[j] <- (rt.count.weighted)/I
    p.low.seq[j] <- infectiousness.seq[j] * qchisq(0.05, 2*rt.num) / (2*rt.num)
    p.up.seq[j] <- infectiousness.seq[j] * qchisq(0.95, 2*rt.num) / (2*rt.num)
  }
 }
 ## p.low.seq[is.nan(p.low.seq)] <- 0
 ## p.up.seq[is.nan(p.up.seq)] <- 0
 list(infectiousness = infectiousness.seq, p.up = p.up.seq, p.low = p.low.seq)
}

#' Predict the popularity of information cascade
#'
#' @param infectiousness a vector of estimated infectiousness, returned by \code{\link{get.infectiousness}}
#' @param n.star the average node degree in the social network
#' @param features.return if TRUE, returns a matrix of features to be used to further calibrate the prediction
#' @inheritParams get.infectiousness
#' @return a vector of predicted populatiry at each time in \code{p.time}.
#' @export
#' @examples
#' data(tweet)
#' pred.time <- seq(0, 6 * 60 * 60, by = 60)
#' infectiousness <- get.infectiousness(tweet[, 1], tweet[, 2], pred.time)
#' pred <- pred.cascade(pred.time, infectiousness$infectiousness, tweet[, 1], tweet[, 2], n.star = 100)
#' plot(pred.time, pred)
pred.cascade <- function(p.time, infectiousness, share.time, degree, n.star=100, features.return = FALSE){

 # n.star should a vector of the same length as p.time
 if (length(n.star) == 1) {
  n.star <- rep(n.star, length(p.time))
 }

 # to train for best n.star, we get feature matrices
```

```
 features <- matrix(0, length(p.time), 3)


 prediction <- matrix(0, length(p.time), 1)
 for (i in 1:length(p.time)) {
  share.time.now <- share.time[share.time <= p.time[i]]
  nf.now <- degree[share.time <= p.time[i]]
  rt0 <- sum(share.time <= p.time[i]) - 1
  rt1 <- sum(nf.now * infectiousness[i] * memory.ccdf(p.time[i] - share.time.now))
  prediction[i] <- rt0 + rt1 / (1 - infectiousness[i]*n.star[i])
  features[i, ] <- c(rt0, rt1, infectiousness[i])
  if (infectiousness[i] > 1/n.star[i]) {
     prediction[i] <- Inf
  }
 }


 colnames(features) <- c("current.rt", "numerator", "infectiousness")


 if (!features.return) {
  prediction
 } else {
  list(prediction = prediction, features = features)
 }
}


#' An example information cascade
#'
#' A dataset containing all the (relative) resharing time and node degree of a tweet. The original Twitter ID is
127001313513967616.
#'
#' \itemize{
#'   \item relative_time_second. resharing time in seconds
#'   \item number_of_followers. number of followers
#' }
#'
#' @format A data frame with 15563 rows and 2 columns
#' @source \url{http://board.muse.mu/archive/index.php/t-85075.html}
#' @name tweet
NULL
```