PhD Dissertation

Epidemiology of Representations: An Empirical Approach

—original title may change—

Sébastien Lerique[1]

Supervisor: Jean-Pierre Nadal[2]
Co-supervisor: Camille Roth[3]

[1]Centre d'Analyse et de Mathématique Sociales (CAMS, UMR 8557, CNRS-EHESS, Paris). Email: sebastien.lerique@normalesup.org.

[2]CAMS, and Laboratoire de Physique Statistique (LPS, UMR 8550, CNRS-ENS-UPMC-Univ. Paris Diderot, Paris). Email: nadal@lps.ens.fr

[3]CAMS, Centre Marc Bloch (CMB, UMIFRE 14, CNRS-MAEE-HU, Berlin), and Sciences Po, médialab (Paris). Email: camille.roth@sciencespo.fr

Contents

Chapter 1

Introduction

Chapter 2

Brains Copy Paste

Chapter 3

Gistr

3.1 Introduction

TBD

3.2 Related work

TBD

3.3 Methods

3.3.1 Experiment design principles

Advantages and challenges of transmission chains

An obvious way to address the questions raised in the previous chapter is to use transmission chains in the laboratory to study the evolution of online quotations in a controlled setting: each subject reads, retains, and rewrites sentences that are then passed on to the next subject in a chain of reformulations. Such a setup can reproduce an idealised version of the read-remember-rewrite process which, we hypothesised, participates in the evolution of quotations in blogspace and media outlets. It also provides the information that our previous data set lacked in order to analyse the complete transformations of quotations, as well as the long-term effect of those changes: the links between parent and child sentences are naturally encoded in the data, such that the transformations undergone by each sentence can be studied in full detail. There is no need to restrict ourselves to simpler changes as was necessary for the inference procedure used with digital traces from blogspace. By creating an artificial setting, the experiment design also lets us control the reading and writing conditions as well as the context in which sentences appear, which further removes one of the inevitable uncertainties of the previous protocol (albeit at the cost of less ecological conditions).

The laboratory transmission chain paradigm is not a good fit for our exploratory approach however: we aim to collect data that will allow us to study both the complete set of transformations undergone by short utterances such as online quotations, and the interactions and cumulative effect of such changes; yet we do not know in advance the types of changes that subjects will make, or the extent to which such changes vary according to the type of linguistic content. Transmission chain studies typically start with an a priori hypothesis focused on a well-identified type of content, which is then empirically tested by contrasting the evolutionary outcome of two classes of sentences. Instead, our goal here is to provide first steps in characterising the process by which such evolution of linguistic content arises, and observe how it accumulates in the long term. The setup must thus allow us to collect enough data to extract regularities in successive transformations operated by different subjects on different sentences, and provide a resolving power similar to that of substitutions in online quotations so that we can compare results with the previous chapter. Since our main target is the set of detailed transformations and their interactions, a phenomenan of higher dimensionality than the contrast of accumulated outcomes, it is also crucial to fine-tune the difficulty of the read-write task and the complexity of the source sentences, in order to trigger a set of transformations varied enough that it could approach some of the changes encountered in real life situations. Our progress therefore involves a non-trivial trial-and-error component: indeed, a task made too easy or too difficult, and more so a set of source sentences that are too complex or too straightforward, will lead to either mass deletions or perfect conservation (or the former followed by the latter), none of which can help characterise the more intricate changes that linguistic content undergoes in the ecological setting we aim to simulate.

Web and smartphone experiments

Complementary to laboratory studies and to approaches using online digital traces, a new empirical approach based on Web browsers and mobile computing is striking a different balance in the trade-offs of experimental work which seems very promising in addressing the problems outlined above. Indeed, browsers (both on desktop and mobile) and smartphones have evolved into powerful, ubiquitous application environments for which one can develop any kind of experiment involving text, graphics, and human interactions. At the cost of increased engineering requirements and a different approach to subject recruitment, Web and smartphone experiments give the designer full control over what data is collected and the way interactions are framed (similar to laboratory experiments), and make it possible to quickly collect data sets at scales comparable to what filtered and cleaned digital traces provide.

This approach makes a number of unusual trade-offs, the benefits of which can be summarised as follows:

- *Control*: similar to laboratory experiments, and unlike digital trace analysis, it is possible to use complex designs where all the interactions of the subjects are framed and observed by the experimenter. This includes for instance the presentation of the experiment (e.g. as a game or a self-improvement aid, aside from being a scientific study) and, more importantly, the ways in which the system mediates the interactions between the subjects.
- *Scale*: if and when needed, the technical platform can scale the number of subjects to the tens of thousands at low marginal cost. Interactions between subjects can also scale to involve synchronous or asynchronous contact between hundreds of people, without having to manage per-subject scheduling.
- *Speed of data collection*: once the initial development is completed (see costs below), the data collection cycle is short. One day can be enough to collect 1000–10,000 usable data points, a

size comparable to the final substitutions set extracted and analysed in the previous chapter. This is especially relevant for exploratory work which is made much easier with shorter trial-and-error cycles.
- *Flexible recruitment*: while also a challenge (see costs below), subject recruitment is more flexible than in the laboratory: services like Prolific Academic[1] let the experimenter recruit at reasonable costs in pools of tens of thousands of subjects with fine-grained demographic filters. Wider audiences can be achieved by offering non-financial rewards, framing the experiment as a self-improvement application, or turning it into a game.

The corresponding costs are the following:

- *Technical challenge*: developing Web and smartphone experiments involves a substantial amount of engineering, and makes use of an array of technologies that most researchers, even technical, are not familiar with. While a couple of all-in-one kits exist,[2] creating an experiment that meets one's research questions requires learning average skills in most of the various technologies at play: a native or cross-platform smartphone development environment, Web application development, backend server programming, and some server administration skills. Most importantly, the paradigms and problems encountered are new to researchers: programming is asynchronous due to network communication and the user interface, and technicalities such as user management or email validation can grow into difficult engineering challenges.
- *Spam-control*: subjects are not constrained or encouraged by the face-to-face interaction of a laboratory experiment, neither are they (in most experiments) in the course of an interaction with friends that provides natural incentives for what they write, as can be the case with digital traces. Participants must have an incentive to perform the experiment's tasks well. If the spam introduced by one subject can be isolated in the design of the experiment, one possibility is to filter it once the data is collected and make payment depend on its prevalence. But if the spam introduced by one subject naturally propagates to data seen by other subjects in the experiment, as is the case for transmission chains, effective anti-spam pressures and motivations need to be factored into the design.
- *Recruitment cost*: while recruiting up to a few hundred subjects is cheaper than the equivalent for a laboratory experiment (not counting the development cost),[3] and is easy to manage for fast prototyping and pilot tests, recruitment cost rises linearly with the number of subjects and the time they spend on the experiment, unless a different strategy is used. Turning an experiment into a playful application or an application useful to the subjects (effectively making them users) involves yet another set of skills, can prove challenging, and must be factored into the development cost.

General setup for Web-based transmission chains

The balance achieved by the Web-based approach is well adapted to the experimental requirements we outlined above. Since no existing system would fit our needs, we chose to develop an in-house Web-based platform that could run all our transmission chains as Web experiments. Once ready, the platform would allow us to gather sufficient amounts of quality data in short cycles. We further decided to implement the simplest possible version of the transmission chain paradigm that is still viable, and leave the exploration of more complex setups for future research: the task we used asks

[1] https://www.prolific.ac/.
[2] See e.g. http://funf.org/ and http://www.epicollect.net/.
[3] Global competition on online platforms like Prolific Academic drive subject payments down.

subjects to read and memorise a short utterance, wait a few seconds, then rewrite what they have read as accurately as possible. We ran three main experiments using this evolving platform, and many smaller pilots in between to test improvements from the larger runs and adjust task parameters with source complexity, such that the overall quality of the data we gathered gradually increased. In what follows we present the general setup of the experiments, the data quality evaluation along with the changes implemented to improve it, and finally the adjustment of task complexity. Let us start with the architecture common to all experiments.

Subjects' productions are arranged in chains, such that what a given subject produces is used as the source utterance for the subject appearing next in a chain. In particular, the utterances to memorise are presented with no surrounding context, no distraction task is used between the reading and writing phases, and the material incentive for the task is purely monetary (although as we describe below we fine-tuned the interface to strongly encourage subjects to be conscientious). This simple setup lets us quickly gather data sets of several thousand utterance transformations, ensuring our results will be comparable to those from the set of 6177 substitutions we extracted in the previous chapter. Two parameters are left to vary: the reading time for the source utterances, computed as the number of words in an utterance multiplied by a reading factor that is to be adjusted (and may or may not be shortened by the subject), and the set of initial source utterances.

The experiment is available to subjects as a website, and passing it involves the following steps:

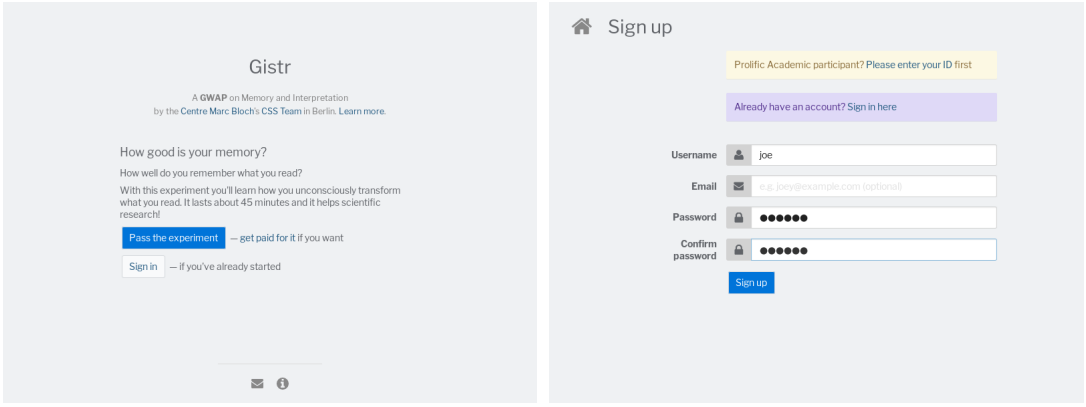
- Welcome and sign up (Figs. 3.1a and 3.1b),
- Answering a preliminary questionnaire (Fig. 3.2),[4]
- Subjects then start training for the main task, where they are asked to repeatedly memorise and rewrite short utterances as accurately as possible. As the instructions in Fig. 3.3 indicate, an utterance is presented to the subject and after a short pause they are asked to rewrite it as remembered. The process loops until the subject has completed all the utterances assigned to them (calibrated so that completing the experiment lasts at most one hour). The real trials start after 3 to 5 training trials, depending on the overall experiment length.

Each utterance from the initial set is used to create several parallel chains in order to allow for comparisons across chains with the same initial utterance. The final data thus consists in a set of reformulation trees, where each tree branch is a transmission chain started from the tree root, and continuing until it reaches a target depth defined for the experiment.[5] The number of branches in a tree is also adjusted for each run of the experiment. Except for those who drop out before finishing the experiment, all subjects are exposed exactly once to each tree in random order, such that all the reformulations in a given tree are made by distinct subjects, and nearly all subjects (excluding dropouts) are present in each tree. Satisfying this constraint means that we must always have at least as many subjects as there are reformulations in a tree. Finally, note that when exposed to a tree, subjects are always randomly assigned to the tip of one of the branches that have not yet reached the target depth: subjects are thus randomly distributed across branches, but their depth-ordering loosely corresponds to time of arrival on the tree. In particular, if a subject starts the experiment after most other subjects have completed it, he or she will be mostly exposed to utterances deep in the branches. Due to the chained nature of the data, there is no economical way of countering this ordering bias.[6] Fig. 3.4 shows a representation of the shape of the final trees.

[4] An early version of the experiment also included a word span test at this stage. However, similarly to the age of subjects that we collect in the questionnaire, this data turned out to not be relevant in the analyses. The magnitude of transformations depends far more on the conscientiousness of subjects, and this non-trivial test was later removed during one of the frontend rewrites.

[5] We therefore use the terms "chain" and "branch" interchangeably in what follows.

[6] The following three approaches could be combined to counter ordering bias. (1) Have each subject do a single trial, that is, use as many subjects as there are reformulations in the full experiment; this is extremely expensive as there is a fixed



(a) Welcome screen (b) Signup form

Figure 3.1: Initial steps for a subject entering the experiment.

Technically, the platform is a complete Web application based on current technologies, with accompanying backend server to collect and distribute utterances.[7] The experiment is available at https://gistr.io and subject recruitment was done using Prolific Academic, a service analogous to Amazon Mechanical Turk and geared towards academic research.[8]

Using the Prolific Academic service allowed us to select among a pool of over 26,000 subjects, for which we used the following criteria:

- First language English speaker
- At least 18 years old
- Current country of residence and place of most time spent before turning 18 must both be the UK
- Normal or corrected to normal vision
- No diagnosed literary difficulties
- Completed secondary school
- Not having participated in any of the preceding experiments

Only the first two constraints were enforced for the first experiment, and the full set was used for all subsequent runs. The full filter provided over 2300 eligible subjects, from which the service

minimal price for each subject, in order to give time to explore the interface, answer the initial questionnaire, and train for the task. (2) Have each subject wait an adjustable amount of time between each trial, to open the possibility of ordering subjects differently from their time of arrival; this is also expensive, as it means paying subjects for waiting most of the time they spend on the experiment. (3) Optimise the order of tree presentations of each subject so as to spread subjects across depths; while this approach could achieve some level of spread when combined with (2), it is contingent on the starting times of subjects and their synchronisation, which we do not control (subjects find the experiment through Prolific Academic notifications and are free to start whenever they want).

[7] The frontend first used the Ember.js framework (Ember.js contributors 2017), and was later rewritten and extended using the Elm programming language (Czaplicki and Elm contributors 2017). Indeed, the assurance of no runtime exceptions that Elm provides was a strong argument in favour of switching, as was made clear by the trying "customer support" experience of a bug hitting 40 to 50 subjects at once during Experiment 1. The backend is a Python application written on top of the Django REST framework (Christie and Django REST framewor contributors 2017). Most of the critical logic in the software is verified using automated tests, and the full source code is available under a Free Software licence at https://github.com/interpretation-experiment/gistr-app (frontend), and https://github.com/interpretation-experiment/spreadr (backend).

[8] The public url of the experiment was not advertised anywhere else, and checking the subjects' Prolific Academic ID confirmed that only people from that platform participated in each experiment.

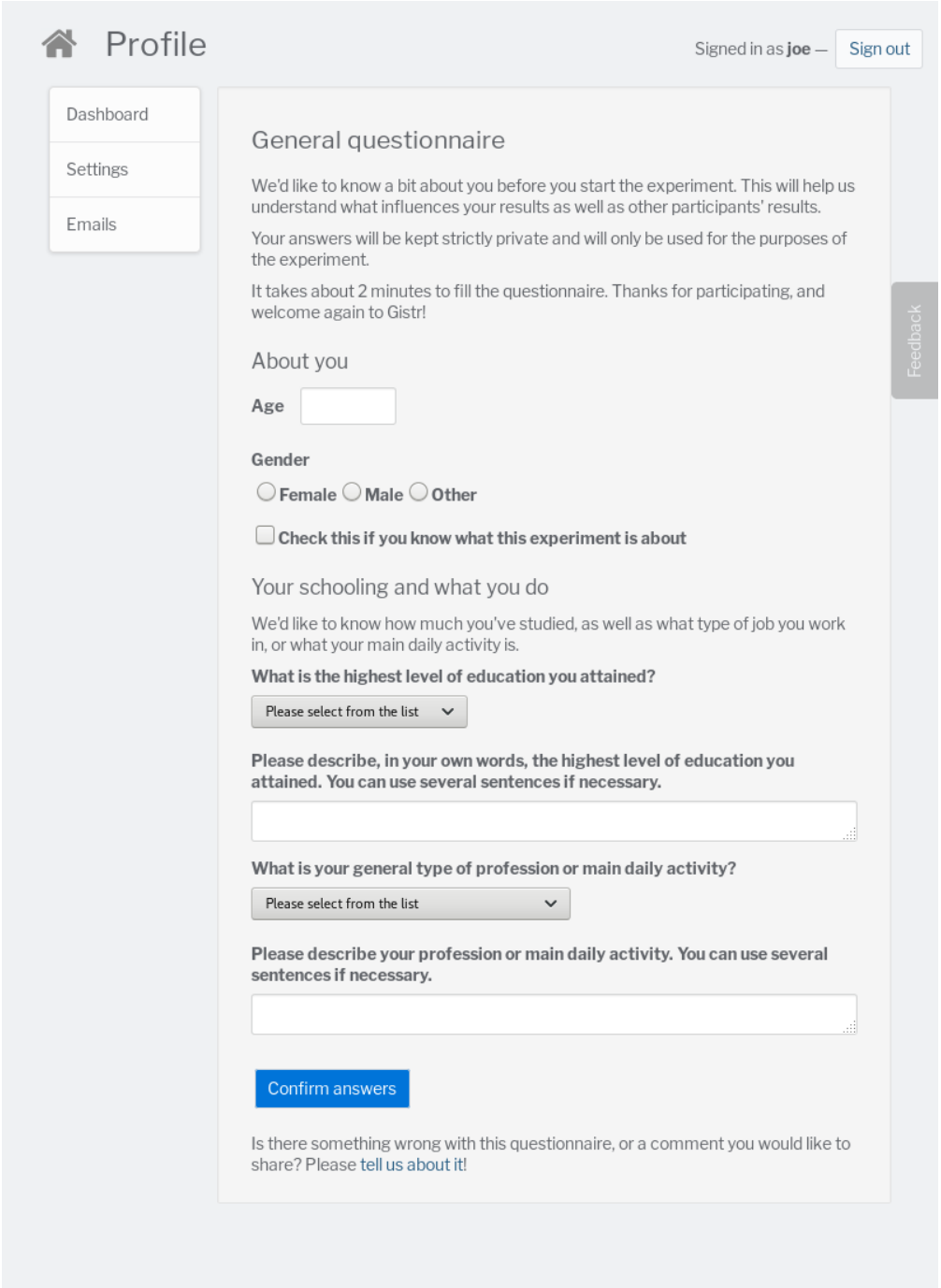


Figure 3.2: Initial questionnaire. Subjects can additionally submit feedback on the questionnaire or any other aspect of the experiment on most screens of the website.

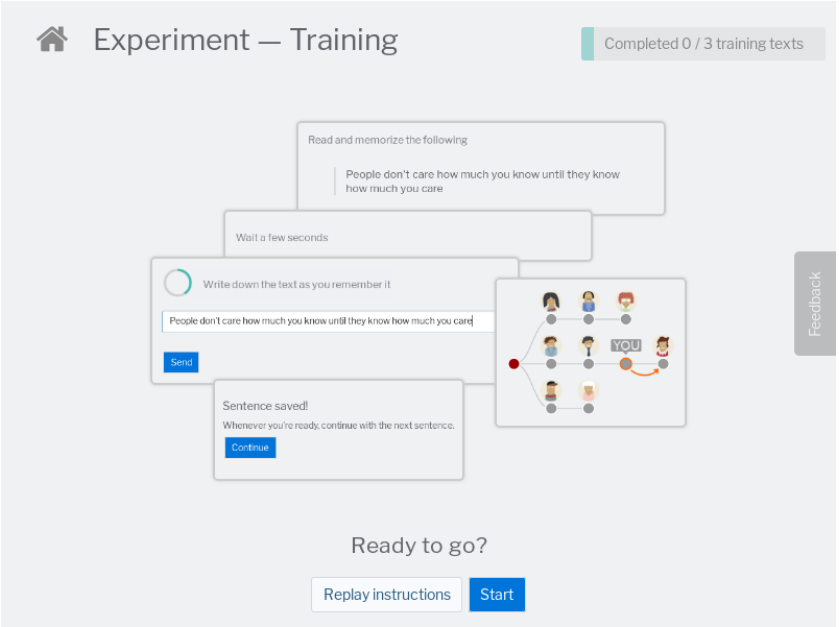


Figure 3.3: Instructions for the main task

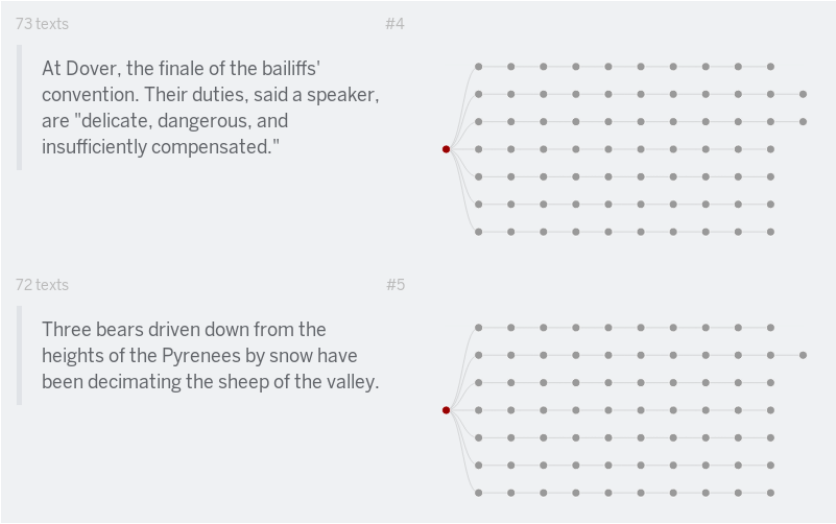


Figure 3.4: Two example reformulation trees generated by the setup, targeted at 7 branches of depth 10: the text on the left is the initial utterance for all branches of a tree, represented by a red dot in the right-hand graph; each grey dot in the graph represents an utterance produced by a subject on the basis of the preceding dot. Most subjects created one reformulation in each tree; however, since subjects from Prolific Academic do not always complete the whole set of utterances assigned to them, we recruit additional subjects to fill the trees that were left incomplete. This leads the other, already complete trees to receive more reformulations than needed, making some of their branches run deeper than the target depth (as is the case for the trees shown here). All branches are cropped to the target depth for analysis.

automatically sampled the number of subjects we requested.

Experiment 1 was the first non-trivial launch of the platform, with an initial 48 subjects, 54 root utterances, and trees targeted for 6 branches of depth 8. Subjects took an average 64 minutes to complete the experiment, and were rewarded with £6.5. A software bug that appeared and had to be fixed halfway through the experiment led the Prolific Academic service to recruit more subjects than was originally asked for, and the final number of participants was 53,[9] gathering a total 2695 utterance reformulations (above the planned $54 \times 6 \times 8 = 2592$ reformulations). Manual inspection showed that large portions of the data were of poor quality, both linguistically and because of technical inefficiencies leading to badly shaped trees; the sections below provide further details on these questions. Pilots following Experiment 1 were therefore aimed at improving data quality and solving tree shaping issues. Experiment 2 was launched with 49 subjects, 50 root utterances, and trees targeted for 7 branches of depth 7, gathering a total 2450 utterance transformations. Subjects took an average 43 minutes to complete the experiment, and were rewarded with £6. Quality issues in this data set were solved, but the choice of source utterances proved too easy to trigger varied transformations. After pilots exploring different fits of task parameters with source complexity, Experiment 3 took advantage of a more complex set of source utterances. It was launched with two batches of 70 subjects each receiving 25 root utterances, and trees targeted for 7 branches of depth 10, gathering a total 3546 utterances transformations. Subjects took an average 37 minutes to complete the 25 transformations, and were rewarded with £4.25 on average.

We now detail the evaluation of data quality and the measures that were taken to improve it. The section after this will focus on the fit of task parameters and source complexity, before moving on to the analysis and results.

3.3.2 Data quality

The choice of a Web-based setup sets the requirements of the interface much higher than for a laboratory experiment. There is no opportunity for a face-to-face walk-through of the experiment or for questions, and subtle changes in the way the interface reacts to actions can lead subjects to interpret a signal where none was intended, or conversely to not notice an important message. The time of the subjects is not booked, and not having to travel to the laboratory or to talk to someone renders the interaction free of any commitment and generally more consumable: subjects can leave whenever they want, without having to feel bad about it (the only cost being the loss of their reward). The lack of human interaction with the experimenter also removes a natural incentive for subjects to take their time and perform according to what the experimenter in front of them explained. Combined together, these factors mean that if the interface is strenuous or ambiguous in any way, subjects will often pick the interpretations that make the process faster and either complete the experiment with minimal engagement or drop out. Redacting detailed textual instructions often makes matters worse. Instead, the interface must lead the subjects through the necessary explanations while remaining enjoyable, and must be unambiguous while still hinting towards the expected behaviour at the right moment, either through subtle interface reactions or through explicit contextual aids.

[9] The bug appeared only once a large proportion of trees had reached their target depth, and then affected all the subjects nearing completion of the experiment. The time taken to respond to complaints and realise that the experiment had to be paused led some subjects to exceed the maximum allowed time on Prolific Academic, and the service then sent the experiment out to new subjects. After fixing the bug, most subjects who had started the experiment accepted to finish it, leading the final subject count to be higher than originally requested.

Manual spam-coding

Failure to properly encourage and wherever possible enforce the experiment's expectations led to data riddled with spurious transformations. Indeed, manual inspection of the data collected through Experiment 1, for which a substantial effort on instructions and for the overall interface had already been made, showed that large portions of the data was not usable as such. We therefore spam-coded all the utterances from this and subsequent experiments by hand. An utterance with any of the following properties was coded as spam:

- An ellipsis ("…") or other special characters (e.g. ">", "<") are present
- The utterance is partly or completely addressed to the experimenter (e.g. "Sorry, I can't remember")
- Over half the words are misspelled
- The utterance has no relationship to its parent utterance (i.e. it is an entirely new utterance)
- The utterance doesn't stand as an autonomous sentence, either because it is truncated or because so many words are garbled it becomes nonsense

Note that the last two criteria are not sharp, and several borderline cases had to be decided for the last one in particular. In Experiment 1 for instance, a subtle misunderstanding allowed by the interface led subjects to submit some sentences truncated at exactly 10 words, without regard to their meaning (see the details below); such utterances were unambiguously incomplete, and were thus coded as spam. In subsequent experiments however, utterances that could be made complete with the addition or the deletion of a single, sometimes unimportant word, were questioned by the same criterion. For instance the simple sentence "Mr Jones was robbed during" can be completed by adding the word "dinner" at the end, or by removing the word "during". Such sentences do not seem tied to a misunderstanding of the task, and are arguably attributable to temporary distraction whose effects are relevant to our analysis. The benefit of the doubt was given to such utterances, and they were not coded as spam.

Spam in transmission chains has the additional property of invalidating all the utterances that are made after it, such that the total number of utterances to discard is more than the spam introduced by subjects. Coded this way, Experiment 1 showed an accumulated spam rate of 22.4%. Combined with an initial technical oversight that led a small portion of utterances to be misplaced in the chains,[10] a total of 25.9% of the utterances generated by Experiment 1 were discarded. Apart from reducing the size of the usable data, spam also leads to uneven chains across trees, a heterogeneity that complicates the analysis. Accepting this level of spam was therefore not an option.

The main tool we used to reduce the level of spam is the user interface. As explained above, minor changes in the way the interface reacts to the subjects' actions combined with relevant context-dependent information can have a comparatively large impact on the spam rate.

[10] Ensuring that no two subjects are creating reformulations for the same chain tip at the same time, while not blocking other subjects from moving on with the experiment, is a non-trivial technical hurdle. Not solving it leads the chains to have "forks", that is, utterances with several children (possibly sub-branches) instead of a single one. One of the children must then be chosen to form the main chain, and the others discarded. Solutions to the problem are difficult to test in practice, as they involve simulating dozens of subjects concurrently sending utterances to the platform. The approach adopted in Experiment 1 relied on client-side randomisation, but proved insufficient: 3.5% of the utterances posted by subjects were forks deep in the chains. Experiments 2 and 3 relied on a mix of client-side randomisation and server-side locking to solve the problem.

User interface improvements

The situation is similar to that of surveys, where much effort is put into mitigating the risk of users engaging the minimum possible effort to complete the survey (Krosnick 2000). Successfully tuning the user interface is therefore a crucial factor in the quality of the data collected: what the interface might lead subjects to see as acceptable can easily be spam for the experimenter, and both perspectives must be aligned as much as possible. Interface design problems appeared repeatedly throughout the development of the platform and the pilots. The most important points can be summed up as follows:

- *Preventing digital copy-paste*: an obvious workaround to the task that most subjects will try in the first few trials.
- *Constraining the input*: a well-known behaviour in transmission chains of linguistic content is the rapid reduction in size of the content that is transmitted (Maxwell 1936; Bangerter 2000; Mesoudi and Whiten 2004). In order to encourage subjects to rely on what they remember, and prevent them from quickly reaching empty sentences, an early version of the experiment would disable the "send" button if the subject's input was shorter than 10 words (Experiments 2 and 3 later relaxed this constraint to 5 words). However, some subjects interpreted the button becoming active after 10 words as a signal that their input was ready to be sent as is, even if it was only a partial sentence. This ambiguity, corrected in later versions, is responsible for a large part of the spam found in Experiment 1.
- *Improving input quality*: Experiment 1 and subsequent pilots showed the need for strong incentives to write well-edited meaningful text. Indeed when pressed for time, some subjects will tend to write misspelled, poorly punctuated, or even meaningless utterances, which invalidate all the sentences that follow in the branch. Countering this tendency involved several changes to Experiments 2 and 3: emphasis was added to the fact that subjects' productions are later sent to other subjects, encouraging a more conscientious behaviour; a bonus was associated with high-fidelity trials, and the top 5 subjects with lowest transformation rates (as defined below in the analysis) received increased payment; most importantly, input from the subjects was also checked for repeated or inadequate punctuation, and for correct spelling against a combined British and American English dictionary. The interface asked subjects to correct any input that failed those tests, and presented them with a short explanation that emphasised the faulty behaviour and recalled the chain structure of the experiment. Inspecting the platform logs showed that this last measure led subjects to often correct their utterances, a fact that was also confirmed by the increased average writing time.
- *Relaxing the time pressure*: the interface of Experiment 1 made several mistakes that worsened the inherent pressure on subjects to complete the study as fast as possible (indeed, payment on Prolific Academic is per experiment, not per time spent – which, conversely, would encourage subjects to be very slow). First, subjects could terminate the reading time of an utterance at will: while this provided a measure of the effective reading time used by subjects, it also opened the possibility of speeding through the trials. Indeed over a third of the transformations of Experiment 1 were done by using less than half the allotted reading time. This pressure was increased by the presence of a "remaining time" clock in the reading and waiting phases, similar to the green clock shown on Fig. 3.3 for the writing phase. By removing superfluous clocks, keeping the reading time fixed, and proposing a break after each utterance, Experiments 2 and 3 relaxed the time pressure on the subjects and improved the final data quality.
- *Feedback channel*: survey design handbooks regularly insist on the importance of providing a channel for subjects to comment on the questions they were asked, and encourage the use of debriefing sessions to deepen that understanding (Leeuw, Hox, and Dillman 2008). Such

feedback channels have also become a norm in online services, and we therefore chose to give the possibility for subjects to comment on most screens of the experiment (excluding the read-write screens) through a side-ribbon which, when clicked, would overlay a comment box (see Fig. 3.5). It seems, however, that a more interactive option would be more effective, as only a handful of subjects entered comments over the course of Experiments 2 and 3.

- *Instructions*: finally, a continuous effort was invested into fine-tuning the exact phrasing of instructions, and making the interface for instructions palatable using a now common pattern: for the instructions pictured in Fig. 3.3 for instance, different elements or images are successively foregrounded and highlighted, and a tooltip with short explanations appears next to the active element.[11] Here too, Experiment 1 and subsequent pilots allowed users to skip these instructions, leading a portion of the subjects to effectively never read them. Experiments 2 and 3 made navigating the complete list of instructions mandatory in order to start the trials.

These changes reduced the spam rate drastically. On the same criteria as Experiment 1, Experiment 2 showed an accumulated spam rate of .8%, which combined with misplaced utterances led to a total 1.4% of utterances discarded. Experiment 3 showed an accumulated spam rate of 1.0%, and with misplaced utterances had to discard a total 1.1% of the data.

TODO: Am I certain that I coded spam the same way in all three experiments?

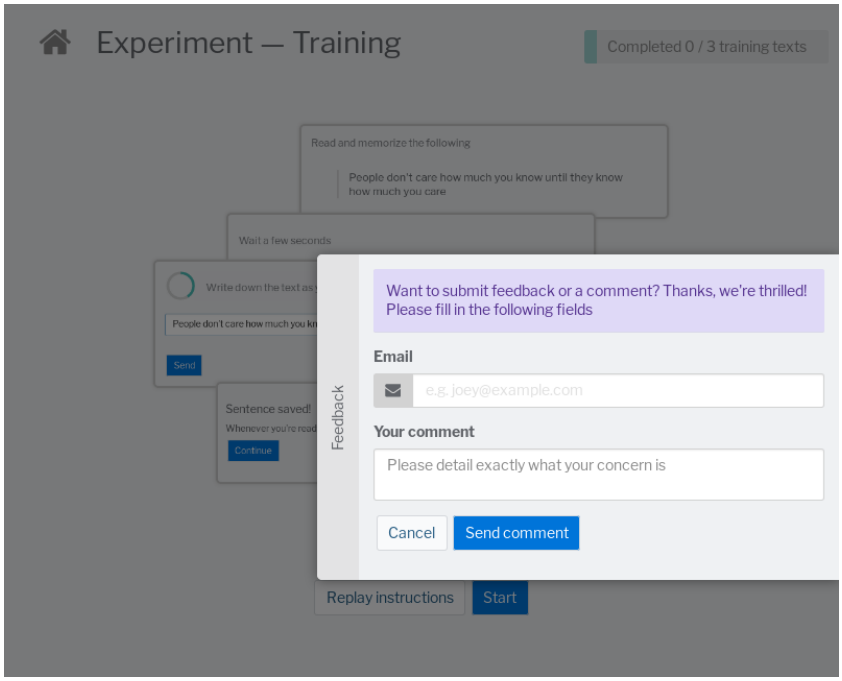


Figure 3.5: Overlay feedback box opened in the instructions screen from Fig. 3.3. The box is available in most screens of Experiments 2 and 3.

[11] The pattern was popularised by software libraries such as Intro.js (http://introjs.com/).

3.3.3 Task difficulty and source complexity fit

Given the simple task we used, difficulty is controlled by the reading time allotted to subjects and by the selection of source utterances. In order to unify reading times across utterances we decided that reading time would be proportional to the number of words in the utterance presented: for an utterance u, its number of words is noted $|u|_w$ and its reading time is defined as $|u|_w \cdot r$. We call r the *reading factor*. Average reading speeds for university students are usually between 200 and 300 words per minute, that is between 3.33 and 5 words per second (see Rayner, Slattery, and Bélanger 2010, where fast readers average at 330 wpm and slow readers average at 207 wpm). A reading factor of $r = 1$ therefore gives fast readers the time to read utterances more than 5 times, and slow readers about 3 to 4 times. A reading factor of $r = .3$ gives fast readers one or two readings, and slow readers at least one. $r = .2$ gives some readers one reading and others less than one, and $r = .1$ lets readers simply glance at the utterances without being able to read them completely.

Pilots and manual exploration indicated that the difficulty of the task is not linear with r. Whatever the value of r, longer utterances (more than 25 words) are often more transformed, relative to their length, than shorter utterances; longer utterances also give more space for the subjects to reformulate, leading to more changes in style and permutations in the words. Changing r has less effect on shorter utterances than on longer utterances, and on utterances in oral versus written style. For short utterances in an oral style, pilots indicated that there is an abrupt transition between a low transformation regime when subjects can read the sentence at least once, and an extremely noisy regime when the subjects do not have the time to read the utterances entirely at their normal speed. Conversely, the transition is smoother for longer utterances or utterances with a more formal written style. Choosing an adequate set of source utterances is therefore an integral factor in adjusting the difficulty of the task.

Changing the source utterances also affects the sampling bias in ways that are difficult to measure given the multidimensionality of text. Contrasting minimally different utterances in different domains has resulted in domain-specific outcomes on for instance stereotypes, information hierarchy, and counter-intuitiveness (Kashima 2000; Mesoudi, Whiten, and Dunbar 2006; Barrett and Nyhof 2001; Mesoudi and Whiten 2008), and authors have suggested that these outcomes are related to domain-specific biases in transformations (although to our knowledge these effects have not yet been studied jointly). In spite of this, we hypothesise that the low-level cognitive mechanisms underlying utterance transformation, that is the mechanisms that give rise to such accumulated outcomes, do not fundamentally change because of the type or the style of an utterance. If using news quotes instead of movie quotes or stories is likely to affect parameters of the observed transformations, it is less likely to affect the structure of the underlying cognitive mechanism, and therefore the general structure of transformations. Making this hypothesis lets us use utterance selection as an exploratory tool: by altering both the sampling of the transformations and the task difficulty, the exploration of different styles and types can help (1) improve data quality and (2) make general structure more visible, thus easier to measure and characterise. If this exploration yields insights about the structure of transformations and their effects in the long term, and if such insights are consistent with the previous chapter, then it will make sense to ask to what extent the uncovered structure is applicable to or varies with other types of utterances. Throughout pilots and experiments, our goal was therefore to find a set of utterances which would trigger varied transformations whose structure we could analyse, while at the same time helping the subjects to produce quality data by not creating too much pressure with reading time.

The set of sources used in Experiment 1 covered a broad spectrum of utterance types sampled from the following categories:

- Quotes from the MemeTracker data set used in the previous chapter,
- Famous compelling quotes from Wikisource[12] such as "Never doubt that a small group of thoughtful committed citizens can change the world, it is the only thing that ever has"
- Quotes extracted from the movie *12 Angry Men* such as "If you ask me I'd slap those tough kids down before they start any trouble, it saves a lot of time and money",
- Excerpts from news stories on controversial subjects (such as "How will the cultural and religious aspects of so many migrants impact E.U. society?") or risk-related subjects such as stories about the risks of Triclosan (used by Moussaïd, Brighton, and Gaissmaier 2015 in their study of the amplification of risk perception),
- The tale "War of the Ghosts" used by Bartlett (1995) in his original studies,[13] as well as excerpts from other tales,
- A small number of hand-crafted sentences such as surprising statements (e.g. "Don't forget to leave the door open when you leave the office") or stereotype incongruent statements (e.g. "The young boy was suddenly hit by the little girl").

Each of these categories, we thought, could encourage the triggering of transformations. The spam level of Experiment 1, and especially the amount of misspelled words, made the exploration of the detailed transformations impossible and shifted the focus towards improving data quality through the interface. Nonetheless, it became clear that using such a heterogeneous set of utterances could surprise subjects, and was not be the best approach to elicit regularities in transformations. Experiments 2 and 3 relied on a more thorough exploration of possible source data sets. Pilots explored utterances extracted from previous studies (Bangerter 2000 on personification and increased stereotypes; Heath, Bell, and Sternberg 2001 on the role of disgust; Maxwell 1936 on incoherent stories; Mesoudi, Whiten, and Dunbar 2006 for the role of social information).

Two larger and more homogeneous sets of utterances were also reconstituted and finally used in Experiments 2 and 3. First, a set of movie quotes provided by Danescu-Niculescu-Mizil et al. (2012). This data set contains about 2200 pairs of quotes extracted from 1000 movie scripts; each pair is made of a quote that was marked as memorable by users of the Internet Movie Database, coupled with the closest quote in the same movie script that is spoken by the same character, has the same number of words, but is not marked as memorable on the Internet Movie Database. The 2200 pairs of quotes were filtered to keep only those which passed the spelling and punctuation quality tests from the previous section, and for which the number of words was strictly matched when excluding punctuation (this left 505 pairs). Second, a set of short stories from Fénéon and Sante (2007) was used. These stories are productions from Félix Fénéon originally anonymously published in the French newspaper *Le Matin* in 1906. They describe facts from everyday life such as accidents, suicides, or trials, in a terse and sometimes humorous style. A sample of 60 stories was extracted from the English version, for which French names and places were replaced with names and places more familiar to British subjects. Pilots explored these sets of utterances with reading factors of .1, .2, .3, .75 and 1. Finally, tests were also made using these utterances with content words replaced with pseudo-words, in order to restrict effects to the grammatical dimension only.[14] The pseudo-word tests were inconclusive, as the task became too confusing and subjects often replaced unknown words with real words.

Experiment 2 used 25 of the 27 pairs of movie quotes that had exactly 15 or 16 words, providing a homogeneous set of 50 utterances in oral style, with a reading factor of .75. Experiment 3 used 43 of the 60 short stories by Fénéon (average number of words 21.2) coupled with 4 utterances extracted

[12] https://en.wikisource.org/.
[13] Available online at http://penta.ufrgs.br/edu/telelab/2/war-of-t.htm.
[14] Pseudo-words were generate using the Wuggy library (Keuleers and Brysbaert 2010).

from Mesoudi, Whiten, and Dunbar (2006) (average number of words 60.3) and 3 utterances extracted from the story used by Maxwell (1936) (average number of words 40.7), with a reading factor of 1.

ADD: clean counts.

ADD: mispelling proportion in exp 1

ADD: a few example sentences for each experiment

3.4 Results

ADD: a clearer outline of the analysis and the steps we're going to follow: overview of aggregate trends, then wanting to achieve a simple representations of the process through transformation-breakdown/exploration-plots/quantification-of-operations-and-deps

3.4.1 Descriptive observations

We begin the analysis of our three data sets by examining the evolution of aggregate measures as a function of depth in the trees. Here and in what follows, the analyses are made on the data cleaned of spam, and chains truncated at their target depth: the data from Experiment 1 is truncated at depth 8, Experiment 2 at depth 7, and Experiment 3 at depth 10, and all plots are aligned to the same depth axis to facilitate comparisons. The goal of this section is to give an overview of the shape of the data and highlight a few important trends to keep in mind in the rest of the analysis. We will come back to the details of the trends further down. # TODO: do that.

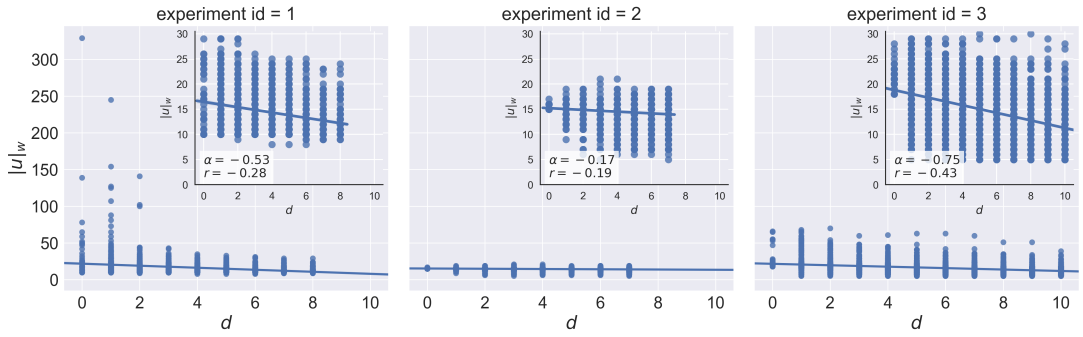
Utterance length

A well-known effect in transmission chains with linguistic content is the quick reduction of utterance length.[15] These experiments are no exception: Fig. 3.6a shows a scatter plot of the number of words of an utterance $|u|_w$ versus depth in a tree. The insets show the data restricted to trees for which root utterances have 30 words or less (thus most utterances in those trees also have 30 words or less); this boundary keeps all the Fénéon root utterances in Experiment 3, and we use it to separate longer from shorter utterances for the purposes of this figure. The plots confirm that word length quickly decreases as subjects read and rewrite utterances, and indicate that the reduction depends on the size of what is being transformed: very long utterances (above 100 words) are reduced to less than 100 words in 2 reformulations or less, whereas root utterances with up to 28 content words can maintain their size until the end of the branches of Experiment 3. Note that the differences in the speed of size reduction across the experiments are tied to surface features of the root utterances. Word count and average word frequency in particular, which we will later show are strongly related to transformation rate # TODO: do that, have different distributions in the set of root utterances of each experiment: word counts are disjoint between Experiments 2 and 3, and root utterances from Experiment 2 are in an oral style, with a higher proportion of stopwords than in Experiments 1 and 3 (stopwords are very frequent words, and make up 67% of the root utterances of Experiment 2, versus 58% in Experiment 1 and 48% in Experiment 3). The steeper slopes in Experiments 1 and 3 compared

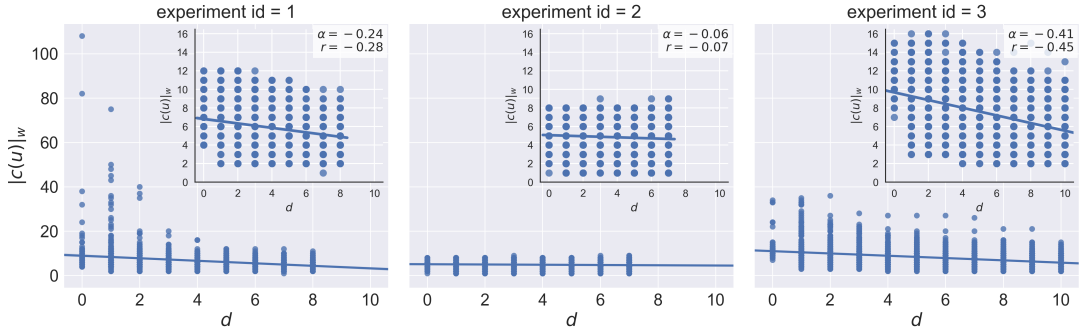
[15] All NLP computations in this chapter are performed using the spaCy library for Python, version 1.9.0, available at http://spacy.io/.

to Experiment 2 are thus tied to the higher word counts and lower proportions of stopwords in their root utterances.

Next, we eliminate stopwords from the utterances and focus on the reduction in number of content words (notice however that stopword recognition is less reliable in Experiment 1 than in Experiments 2 and 3, because of spelling mistakes). For a given utterance u, the list of its content words is noted $c(u)$, and the number of content words is therefore $|c(u)|_w$, where $|\cdot|_w$ is extended to provide the length of a list of words (aside from counting words in a string). Fig. 3.6b plots the content word counts for the same utterances as those in Fig. 3.6a (in particular, insets show the same utterances in both figures), and shows a similar reduction in counts across all experiments. Here too, the differences in regression values across experiments and between the two figures are tied to the differences in distributions of word count and proportion of stopwords in the roots. In other words, the size reduction is sampled differently by each experiment: these figures show how the effect acts on each set of root utterances, but do not indicate that the mechanism is any different across experiments nor that it depends on the actual meaning of the utterances (versus primarily on surface features such as word count and average word frequency).



(a) Numbers of words, $|u|_w$



(b) Numbers of content words, $|c(u)|_w$

Figure 3.6: Reduction in utterance word count and content word count as a function of depth (d) in the three experiments. Each blue dot represents an utterance. For a given experiment, the same utterances appear in panels (a) and (b). The insets show the utterances for which the tree root has 30 or less words ($|u_{\text{root}}|_w \leq 30$), with the numerical values of the linear regression slope and correlation coefficient. All regression slopes are non-zero with $p < .001$.

Utterance to utterance distance

As a first approximation of the magnitude of transformations we introduce a measure of the distance $\lambda(u, u')$ between two utterances u and u', defined as the Levenshtein distance[16] between the lemmas of the content words of u and u':

$$\lambda(u, u') = \text{levenshtein}\left(\text{lemmatize}(c(u)), \text{lemmatize}(c(u'))\right)$$

For example, consider the three following utterances taken from Experiment 1 (in a tree whose root is from the MemeTracker data set):

u_a: "This crisis did not develop overnight and it will not be solved overnight"

u_b: "the crisis did not developed overnight, and it will be not solved overnight"

u_c: "The crisis didn't happen today won't be solved by midnight."

After removing the punctuation and converting all words to lowercase, the lemmas of the content words of these utterances are as follows:

lemmas($c(u_a)$): "crisis", "develop", "overnight", "solve", "overnight"

lemmas($c(u_b)$): "crisis", "develop", "overnight", "solve", "overnight"

lemmas($c(u_c)$): "crisis", "happen", "today", "solve", "midnight"

Such that $\lambda(u_a, u_b) = 0$ and $\lambda(u_a, u_c) = \lambda(u_b, u_c) = 3$. λ thus measures differences in content words and is blind to minor transformations, such as changes of stopwords or word inflexions. In order to have a uniform quantity across utterances of different lengths, we define the *transformation rate* ρ as the normalised distance between two utterances:

$$\rho(u, u') = \frac{\lambda(u, u')}{\max\left(|c(u)|_w, |c(u')|_w\right)}$$

ρ thus measures the magnitude of the transformation between the contents of u and u', relative to the size of the contents of those utterances. It takes its values between 0 and 1: $\rho(u, u') = 0$ if and only if u and u' have exactly the same content words in the same order, and $\rho(u, u') = 1$ means that the content words of u and u' have so little in common that rewriting from scratch is quicker than changing one into the other with word insertions, deletions, or replacements. Here, $\rho(u_a, u_b) = 0$ and $\rho(u_a, u_c) = \rho(u_b, u_c) = .6$. A major caveat of this measure is that it does not know about synonyms or expressions with similar meaning, such that two sentences separated by a transformation rate of 1 can have the same meaning at a higher level. For instance with the following sentences,

u_d: "Will you investigate the gravest crimes of the Bush administration, including torture and warrantless wiretapping?" (from Experiment 1, originally from the MemeTracker data set)

u_e: "Will you research the worst problems of the 2004 mandate, like its surveillance?" (hand-crafted for this comparison)

[16] The Levenshtein distance (also known as the edit distance) is defined between two lists of items, and counts the minimal number of insertions, deletions, and replacements that are needed to transform the first list of items into the second. It has all the properties of a metric (non-negativity, identity of indiscernibles, symmetry, and subadditivity).

u_f: "Don't forget to leave the door open when you leave the office" (from Experiment 1)

we have $\rho(u_d, u_e) = \rho(u_d, u_f) = 1$. The measure misrepresents the changes between these utterances, as u_d and u_e can easily be considered to have similar meanings at a high level, and their difference is far less important than the difference between u_d and u_f. The measure performs reasonably well on utterances inside the same tree however: in that context all utterances come from the same source and have therefore some level of meaning in common, and there is no need to differentiate between the types of transformations that $u_d \to u_e$ and $u_d \to u_f$ represent.

Transmissibility and transformation rate

Together with transformation rate, we examine a measure derived from it: the *transmissibility* of utterances, defined as the proportion of utterances at a given depth whose content words are perfectly transmitted to their child, computed over all the branches of all the trees of an experiment (this measure was introduced as 'average success' in Claidière et al. 2014). If we note $\mathbb{1}_{\lambda(u,u')=0}$ the success of a subject in transmitting an utterance's content (it equals 1 if the content words of u and u' match perfectly, and 0 if there was any change in content words), the transmissibility of utterances at depth d can be expressed as:

$$\eta(d) = \left\langle \mathbb{1}_{\lambda(u,u')=0} \right\rangle_{u \text{ at depth } d,\ u' \text{ child of } u}$$

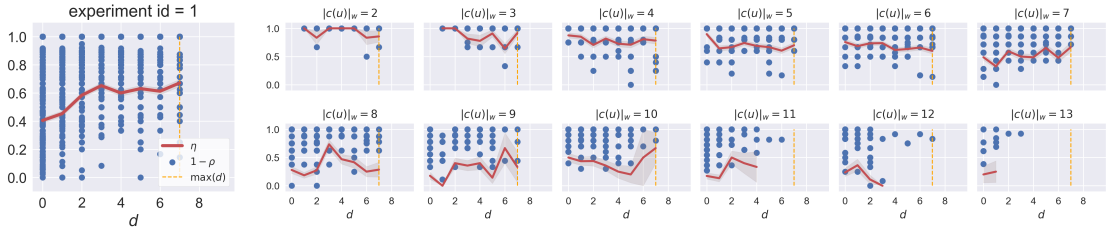
Transmissibility provides a coarser measure of the evolution of transmission success than transformation rate (a change in transmissibility implies a change of transformation rates), but lets us better differentiate between the two important alternatives: perfect transmission, and transformation. A classic effect of transmission chains for various types of content is that transmissibility increases with depth in the chains.

Fig. 3.7 shows the transmissibility and one minus the transformation rates ($1 - \rho$) for the three experiments, both overall and grouped by content length of the utterances. Fig. 3.7a shows an increase in transmissibility with respect to depth (from .40 to .67), when considering the whole data set from Experiment 1. However, the plots on the right show only a slight increase in transmissibility (or even no increase at all for $|c(u)|_w \notin [7, 10]$) for utterances of a given content length. The right-hand side also indicates that transmissibility depends on content length, as the transmissibility lines become gradually lower when content length increases (average .92 for 2 content words, .20 for 12 content words). Together, these trends indicate that the overall increase in transmissibility with respect to depth could be mostly due to the rarefaction of utterances with long content length: as depth increases, the proportion of shorter utterances increases; shorter utterances are better transmitted, and as consequence global transmissibility increases too.

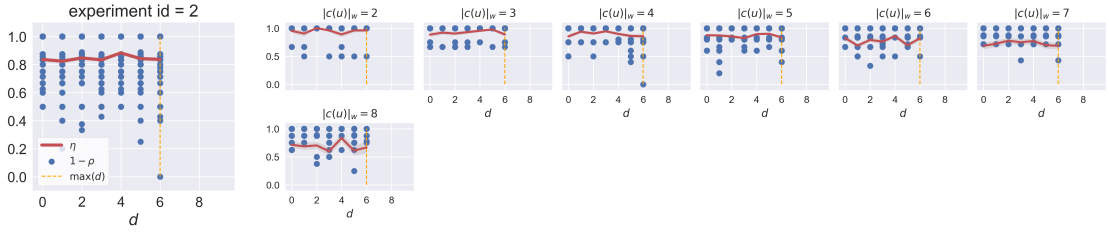
Fig. 3.7b shows the same analysis for Experiment 2. Contrary to the previous case, transmissibility here is stable at .82-.88 with respect to depth, both for the whole data set and at fixed content length. It also depends less on content length than in Experiment 1, as utterances with 2 content words have an average transmissibility of .95, and utterances with 8 content words an average transmissibility of .69.

Experiment 3 (Fig. 3.7c) features an increase in transmissibility with respect to depth both globally (from .18 to .71) and at long fixed content length. This effect is stronger than in Experiment 1, and indicates that long utterances in the data set become slightly easier to transmit as they are transformed. As noted previously, utterances found at the end of a chain will often come from much

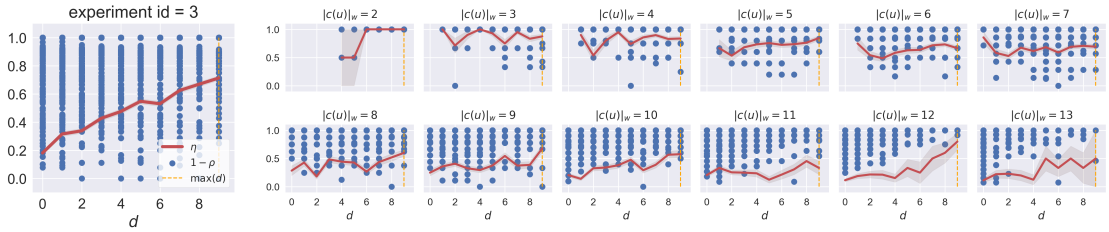
longer utterances at the start, such that improved transmission success along a single branch is always mixed with the shortening of content. However, for long utterances (content lengths 8 and above), utterances found at the end of all chains are on average better transmitted than utterances *of the same content length* at the start of all chains, meaning that transmission along the chains has an effect on transmissibility of long utterances beyond the shortening of the content. Finally, the different behaviours across experiments are here again tied to the differences in word count and stopword proportion distributions in the root utterances.



(a) Experiment 1



(b) Experiment 2



(c) Experiment 3

Figure 3.7: Transmissibility and conservation rate for each experiment. Each individual graph shows both transmissibility (red line, with 95% confidence intervals) and one minus the transformation rate (blue dots) for a subset of all utterances. A blue dot at $y = 1$ is an instance of perfect transmission ($\rho = 0$), and pulls transmissibility upwards; a blue dot anywhere below is a transformation instance ($\rho > 0$), and pulls transmissibility downwards. The large plot on the left shows both measures for all the sentences of an experiment. The small plots on the right show both measures for utterances that have a given number of content words (up to 13 content words, after which the data is nonexistent or very sparse in all experiments). The orange dashed line marks the maximum depth in the experiment, so as to differentiate content lengths reaching the limit from content lengths disappearing before the limit.

Variability

TODO: remove this?

Let us close this overview of the aggregate evolution in all experiments with a final measure: the variability of utterances at a given tree depth. For a given tree t, the variability $\kappa(t, d)$ measures the average transformation rate between all pairs of utterances at depth d in t (also called the *slice* of t at depth d):

$$\kappa(t, d) = \langle \rho(u, u') \rangle_{\{u,u'\} \subset \{u \text{ at depth } d \text{ in } t\}}$$

This measure gives a sense of how fast branches diverge between each other. For each experiment, Fig. 3.8 plots the variability of all slices of all trees, and the average variability averaged across trees. All three increase significantly, meaning that utterances from different branches in a tree become more and more different as the chains progress. The different divergence rates correspond loosely to the transformation rates observed in Fig. 3.7 (Experiment 2 has lower transformation rates, and diverges slower), and are therefore again tied to the differences in root utterances.

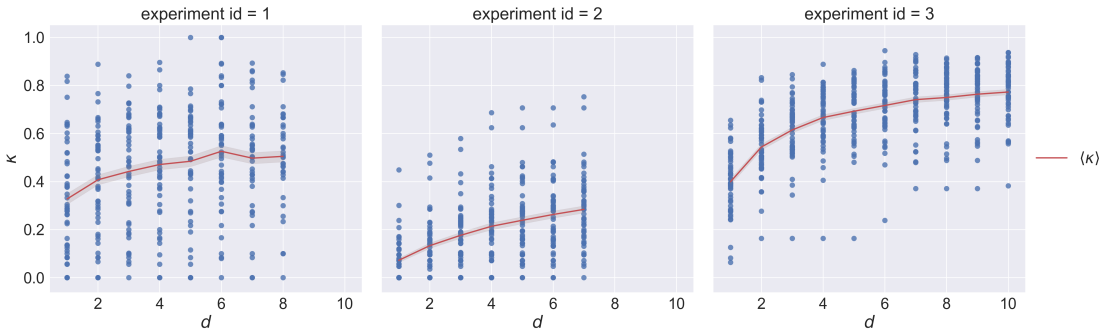


Figure 3.8: Slice variabilities in the three experiments. Each plot shows the variabilities of each slice of each tree (blue dots), as well as the average variability across slices of all trees at a given depth (red line with 95% confidence interval).

We now wish to achieve a much finer view of the process at work in this evolution, by decomposing the transformation operations into smaller blocks. Our goal from now on is to achieve a more synthetic understanding of the process at work, beyond the simple description of the outer shape of the evolution. In what follows we will focus primarily on the data set from Experiment 3, which provides the best overall quality of data and sampling of transformations. The procedure we develop is of course applicable to the other two experiments, but we will not discuss those applications in any detail. # TODO: check that.

3.4.2 Transformation breakdown

Sequence alignments

Our first step towards this finer view of transformations is to take advantage of existing generalisations of the Levenshtein distance underlying our measure of transformation rate ρ. Recall that the Levenshtein distance between two sequences of items s and s' computes the minimal number of

insertions, deletions, and replacements necessary to turn s into s' (and vice-versa). This problem can equally be formulated as that of aligning the items of s and s': each item of s can be paired either with an item from s' (signifying a conservation if both items match, or a replacement if the two items are different), or with nothing (signifying a deletion in the transformation of s into s'). Symmetrically, items from s' can also be paired with nothing (aside from being paired with items from s), signifying an insertion in the transformation of s into s'. In this formulation, insertions and deletions are merged into the same operation: a "gap" (or "indel"), found either in s or in s'. As we will now see, this problem has become extremely important over the last 50 years in the subfield of bioinformatics known as sequence alignment.

Sequence alignment is in the business of looking for similarities between sequences of DNA, RNA, or amino acids in proteins that could indicate evolutionary or structural relationships between two or more species. Research on this task has led to the development of several generalisations of the algorithm underlying the Levenshtein distance; these are geared towards assigning different weights or costs to the individual operations transforming one sequence into the other, finding optimal alignments of subparts of the two sequences (a task known as local alignment, in contrast to global alignment), or aligning more than two sequences simultaneously (multiple sequence alignment, in contrast to pairwise alignment).

The problems tackled are strikingly similar to our present task: we aim to decompose the transformation from a parent utterance to a child utterance into small basic operations. In sequence alignment terms, this task is a pairwise global alignment of lists of words, for which the Needleman-Wunsch algorithm (Needleman and Wunsch 1970, henceforth NW) provides a flexible generalisation of the Levenshtein distance. For two sequences of items of any type, s and s', the NW algorithm assigns different scores to each basic operation (gap, mismatch a.k.a. replacement, and match, which is considered a scored operation like the first two), and returns the list of alignments between s and s' with maximal total score (there can be several such alignments that achieve the maximal total score). Each alignment can be directly interpreted as a list of operations to transform s into s' (or vice-versa).

More precisely, let us note $s = (s_1, ..., s_n)$ and $s' = (s'_1, ..., s'_{n'})$ the items in both sequences, with n and n' the lengths of the sequences. The NW algorithm returns pairs of sequences a and a' of lengths $m = \max(n, n')$, made of the items from s and s' (respectively), in the same order, but interspersed with a "gap item" which we note g. Noting $a = (a_1, ..., a_m)$ and $a' = (a'_1, ..., a'_m)$, each pair (a_i, a'_i) represents the pairing of an item from s with an item from s' (either match or mismatch), or with a gap if $a'_i = g$ (and vice versa if $a_i = g$). Considered as a transformation from s to s', gaps in a' represent deletions, and gaps in a represent insertions. a and a' are such that the sum of scores of the operations they represent is maximal.

Take for instance the DNA sequences $s =$ AGAACT and $s' =$ GACG. An example alignment between the two sequences can be represented as follows (with the gap item represented as "-", and matches shown with vertical bars):

$$
\begin{aligned}
a &= \texttt{AGAACT-} \\
&\quad\ \texttt{| ||} \\
a' &= \texttt{-G-AC-G}
\end{aligned}
$$

The power of the NW algorithm is that gap, mismatch and match scores can be defined at compute time, knowing what items are being compared (or evaluated for a gap), and what operations would have been made up to that point if this operation were to be part of an optimal alignment. This

flexibility has been used in biological sequence alignment to account for the fact that, in a DNA sequence for instance, the deletion of a base in the middle of an otherwise intact portion of DNA is less probable than the continuation of a gap that has already started. In other words, in biological sequence alignment opening a new gap is more costly than extending an existing gap, and the compute-time scores of gaps can reflect that. The same goes for mismatches: not all bases are equally likely to replace another base, and mismatch scores can factor that into the evaluation of alignments. As is hopefully clear by now, the situation is strikingly similar for sequences of words. In the next sections we detail our application and extension of the NW algorithm to decomposing utterance transformations.

Application to utterance alignment

The NW algorithm can be straightforwardly applied to sequences of any kind, provided we define scores for opening and extending gaps and a function to evaluate the comparison of two items (henceforth the match scoring function). We chose to use the algorithm on sequences of words, excluding punctuation, with a match scoring function that takes into account the semantic distance between the two word compared. For a given pair of utterances u and u', we start by tokenising them and removing all punctuation. We then apply the NW algorithm[17] on the resulting sequences of tokens, with a match scoring function computed as an affine transformation of the similarity between two words w and w':

$$\text{similarity}(w, w') = \begin{cases} S_C\left(\boldsymbol{w}, \boldsymbol{w}'\right) & \text{if we have word vectors for both } w \text{ and } w' \\ \delta_{\text{lemma}(w),\text{lemma}(w')} & \text{otherwise} \end{cases}$$

where S_C is the cosine similarity function (one minus cosine distance) and $\boldsymbol{w}$ is a 300-dimensional vector representation of w encoding the word's semantics,[18] such that the $S_C(\boldsymbol{w}, \boldsymbol{w}')$ provides a measure of semantic similarity between w and w'. Finally, $\delta_{i,j}$ is Kronecker's delta which equals 1 if and only if $i = j$, and 0 otherwise. This function thus provides a "best effort" similarity measure which depends on whether we have detailed information about the words being compared or not.

Adding an affine transformation to similarity lets us adjust its importance with respect to gap scores, for which we only differentiate opening and extension scores. This definition thus uses an initial 4 scalar parameters (two gap scores, two affine parameters) that define the way each operation is scored against the others. Since the final score of an alignment is computed as the sum of the scores of its individual operations, a linear scaling of all the parameters by the same amount does not change the choice of best-scoring alignments, such that we can further reduce the number of parameters by one. We choose to set the slope of the affine transformation of similarity to 1, and are then left with 3 alignment parameters:

- $\theta_{mismatch}$, the base score for the match scoring function, such that

$$\text{score}(w, w') = \text{similarity}(w, w') + \theta_{mismatch}$$

- θ_{open}, the score for opening a gap; θ_{open} is negative since it is a cost,
- θ_{extend}, the score for extending a gap; θ_{extend} is also negative.

[17] We used Biopython's implementation of the NW algorithm (Cock et al. 2009).

[18] The standard spaCy English language model includes "vectors for one million vocabulary entries, using the 300-dimensional vectors trained on the Common Crawl corpus using the GloVe algorithm" (https://alpha.spacy.io/docs/usage/word-vectors-similarities). The GloVe algorithm was developed by Pennington, Socher, and Manning (2014).

Given the right set of parameters, this tool lets us align two utterances and obtain the minimal set of operations that transform one into the other. Take for instance the following two utterances from Experiment 3:

> "Finding her son, Alvin, 69, hanged, Mrs Hunt, of Brighton, was so depressed she could not cut him down."
>
> "Finding her son Arthur 69 hanged Mrs Brown from Brighton was so upset she could not cut him down"

TODO: use labelled lists for quotes

With the set of parameters that we obtain through training as explained below, the algorithm aligns these two utterances as follows (noting any gaps with "-", and emphasising replacements):

```
Finding her son Alvin  69 hanged Mrs Hunt of -     -    Brighton was so depressed
Finding her son Arthur 69 hanged Mrs -    -  Brown from Brighton was so upset

she could not cut him down
she could not cut him down
```

Detecting exchanges

The application of the NW algorithm developed up to this point works rather well for simple transformations such as the one exemplified above. However, more complicated transformations include operations that the algorithm as is cannot detect or represent. Hand inspection of the data showed that exchanging sub-parts of an utterance, in particular, is a relatively common operation for which our current tool has no representation. Consider the following two utterances from Experiment 3 for instance:

> u_a: "At Dover, the finale of the bailiffs convention, their duty said a speaker are delicate, dangerous and detailed"
>
> u_b: "At Dover, at a Bailiffs convention. a speaker said that their duty was to patience, and determination"

The current alignment algorithm, with parameters trained according to a procedure outlined below, produces the following:

```
At Dover the finale of the -  - bailiffs convention - -       -    -    their duty
At Dover -   -      -  -   at a Bailiffs convention a speaker said that their duty

said a speaker are delicate dangerous -   -  -        and detailed -
-    - -       -   -        -         was to patience and -         determination
```

This alignment misses the fact that the deleted part "said a speaker" is found as "a speaker said" earlier in the reformulated utterance. The general idea to detect such exchanges is that blocks of insertions and blocks of deletions can be matched against one another with the same alignment algorithm, and the resulting deep (recursive) alignment can be scored and compared to the initial shallow alignment. If the final deep score $\chi_{deep}(u_a, u_b)$ is higher than the initial shallow score $\chi_{shallow}(u_a, u_b)$, then we adopt the deep alignment with exchange as the best solution. Suppose that for the alignment of the deletion block u_- "said a speaker are delicate dangerous" with the insertion block u_+ "a speaker said that", we are able to compute an optimal deep alignment with associated score $\chi_{deep}(u_-, u_+)$; then the deep score for the top level $\chi_{deep}(u_a, u_b)$ is as follows:

$$
\begin{aligned}
\chi_{deep}(u_a, u_b) = \chi_{shallow}(u_a, u_b) & \quad \text{start from the initial shallow score} \\
+ \theta_{exchange} & \quad \text{score the addition of an exchange operation} \\
- \text{score(deletion of } u_-) & \quad \text{recover the cost of the deletion block} \\
- \text{score(insertion of } u_+) & \quad \text{recover the cost of the insertion block} \\
+ \chi_{deep}(u_-, u_+) & \quad \text{add the deep alignment score of the exchange} \qquad (3.1)
\end{aligned}
$$

where $\theta_{exchange}$ is a new negative parameter that defines the cost of creating an exchange, to be added to the existing three shallow alignment parameters. The deep alignment extension we implemented follows exactly that recursive principle, but accommodates for the possibility of multiple exchanges at each level of the recursion. Algorithm 3.1 provides an overview of the way this tree of alignments can be constructed. Note that for long utterances, the size of the deep alignment tree can grow very fast:

- For a given deep alignment, there is a list of mappings between deletion and insertion blocks,
- Each mapping is a set of (deletion block, insertion block) pairs,
- Under each such pair, there is a list of deep alignments; and from there on recursively.

Also, our implementation of the exploration of that tree is mostly brute force, and does not try to be smart in predicting which branches are dead-ends. In spite of this, we did not need to optimise the computation any further (aside from obvious gains in caching repeated computations), as most of the time of a deep alignment is spent computing shallow alignments, and most alignments of utterances are very shallow anyway. Finally, note that this approach provides no guarantee of finding the globally optimal deep alignment. Indeed, it starts from optimal shallow alignments, and explores the tree of possibles from there on. But the initial shallow alignments it extends may not be the best starting point, such that the exploration may return locally optimal deep alignments.

Nonetheless, given a good set of parameters (see the next section where we derive those), this deep alignment algorithm produces surprisingly satisfying results given the simplicity of its underlying principles. In the case of the two utterances exemplified at the beginning of this section, the algorithm produces the following deep alignment tree. First, the top-level alignment:

```
At Dover the finale of the -  - bailiffs convention |-Exchange-1------| their duty
At Dover -   -      -  -   at a Bailiffs convention a speaker said that their duty

said a speaker are delicate dangerous -   -  -        and detailed -
|-Exchange-1-----------------------| was to patience and -       determination
```

For which $\chi_{shallow} = -2.93$ and $\chi_{deep} = -2.89$. Then the alignment of Exchange 1:

```
said a speaker are delicate dangerous |E2-----|
|E2| a speaker -   -        -         said that
```

For which $\chi_{shallow} = -1.01$ and $\chi_{deep} = -0.99$. And finally the alignment of Exchange 2, from inside Exchange 1:

```
said -
said that
```

For which $\chi_{shallow} = \chi_{deep} = -0.18$.

Notice how in this deep alignment the phrase "are delicate and dangerous" was initially included in Exchange 1, only later to be recognised as a deletion in the alignment of Exchange 1. The same

Algorithm 3.1 An implementation of the deep alignment extension for detecting exchanges in NW alignments. Note that this implementation is grossly inefficient, but presentationally clearer than the implementation we made.

function SHALLOWALIGN(u, u')
  (Implemented by Biopython)
  return List of optimal shallow alignments of u and u'
end function

function MAPPINGS($a_{shallow}$)
  $\mathcal{D} \leftarrow \{d | d$ deletion block in $a_{shallow}\}$
  $\mathcal{I} \leftarrow \{i | i$ deletion block in $a_{shallow}\}$
  return $\mathcal{D}^{P(\mathcal{I})}$ ▷ $P(\Omega)$ is the power set of Ω
end function

function SCOREMAPPING($a_{shallow}$, D_M)
  $s \leftarrow 0$
  for $((u_{e,-}, u_{e,+}), D_e)$ in D_M **do**
    $s \leftarrow s + \theta_{exchange}$
    $s \leftarrow s -$ SCORE(deletion of $u_{e,-}$) $-$ SCORE(insertion of $u_{e,+}$)
    $s \leftarrow s + \max\{\chi_{deep}(a_{deep}) | a_{deep} \in D_e\}$
  end for
  return s
end function

function DEEPALIGN(u, u')
  $D \leftarrow []$ ▷ D is the list of deep alignments trees we have explored
  for $a_{shallow}$ shallow alignment in SHALLOWALIGN(u, u') **do**
    if $a_{shallow}$ has no gaps or has only gaps **then**
      $D \leftarrow (a_{shallow}, [], \chi_{shallow}(a_{shallow}))$
    else
      for M mapping in MAPPINGS($a_{shallow}$) **do**
        $D_M \leftarrow []$
        for $(u_{e,-}, u_{e,+})$ exchange in M **do**
          $D_M \leftarrow D_M + ((u_{e,-}, u_{e,+}),$ DEEPALIGN$(u_{e,-}, u_{e,+}))$
        end for
        $D \leftarrow D + (a_{shallow}, D_M, \chi_{shallow}(a_{shallow}) +$ SCOREMAPPING$(a_{shallow}, D_M))$
      end for
    end if
  end for
  return Recursively pruned version of D with only maximally scoring deep alignments
end function

happened for "that", initially included in Exchange 1 and finally recognised as an insertion in the alignment of Exchange 2. Most cases of deep alignments look like this one, where a single path exists in the tree of recursive alignments. For longer utterances however, there can be several exchanges at each level, and the tree of alignments becomes much larger.

Training alignment parameters

To finish the development of our alignment tool, it is necessary to determine a set of alignment parameters that produce plausible results. Recall that the parameters are:

- $\theta_{mismatch}$, the base score for the match scoring function,
- θ_{open} and θ_{extend}, the scores for opening and extending a gap,
- $\theta_{exchange}$, the score for creating an exchange.

In order to make the problem of finding usable parameters tractable, we decided to restrict parameter training to the shallow alignment parameters only (henceforth noted $\boldsymbol{\theta} = (\theta_{mismatch}, \theta_{open}, \theta_{extend})$), and fine-tune $\theta_{exchange}$ by hand after the first three were defined (this also corresponds to the fact that deep alignments are made on the basis of optimal shallow alignments). Our general approach for this task is therefore to hand-code shallow alignments for a random set of utterance transformations in Experiment 3, train the shallow alignment parameters to that standard, before adjusting the exchange parameter by hand. Since the there are only three dimensions to explore, the training step is easiest to accomplish by brute force.

We thus start by evaluating the size of the training set that is necessary to obtain a set of parameters that extrapolates well to untrained data. Indeed, a training set too small in size might provide too weak a constraint on the set of parameters, such that brute forcing would find many parameter sets that do not extrapolate well. On the other hand, manually coding alignments is time-consuming and we do not wish to code more than necessary. We used the following procedure to decide this trade-off:

1. Uniformly sample a random parameter set $\boldsymbol{\theta}^0 \in [-1, 0]^3$ and use it to generate artificial alignments for all the non-trivial transformations in Experiment 3 (i.e. for which the transformation rate ρ was positive, which amounts to 2159 transformations); note these alignments $\mathcal{A}^0$.
2. Sample a training set of size n from the artificial alignments; note the training set $\mathcal{A}^0_t$, and the remaining evaluation set $\mathcal{A}^0_e = \mathcal{A}^0 \setminus \mathcal{A}^0_t$.
3. Brute-force the sets of parameters $\hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_m$ that best reproduce the training set $\mathcal{A}^0_t$, by exploring the sampling space $[-1, 0]^3$ with a discretisation step of.1; parameters that perfectly reproduced the training set were always found, such that no finer-grained exploration was needed.
4. Evaluate the worst fit $\hat{f}_n$ between the evaluation alignments $\mathcal{A}^0_e$ and the alignments produced by each of the $\hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_m$ on the same transformations.

For a given set $\mathcal{T}$ of transformations, the alignments generated by parameters $\boldsymbol{\theta}$ can be written:

$$A(\mathcal{T}, \boldsymbol{\theta}) = \{\text{aln}(u, u', \boldsymbol{\theta}) | (u, u') \in \mathcal{T}\}$$

Where $\text{aln}(u, u', \boldsymbol{\theta})$ is the set of alignments between u and u' produced by $\boldsymbol{\theta}$. $A(\mathcal{T}, \boldsymbol{\theta})$ is thus a set of sets of individual shallow alignments (indeed each pair of utterances generates its own set of shallow alignments). The fit between two such sets of sets of alignments $A(\mathcal{T}, \boldsymbol{\theta}_1)$ and $A(\mathcal{T}, \boldsymbol{\theta}_2)$ is then computed as:

$$f(\mathcal{T}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \frac{1}{2} \sum_{(u,u') \in \mathcal{T}} \max_{((a_1,a_1'),(a_2,a_2')) \in \text{aln}(u,u',\boldsymbol{\theta_1}) \times \text{aln}(u,u',\boldsymbol{\theta_2})} \{\text{lev}(a_1, a_2) + \text{lev}(a_1', a_2')\}$$

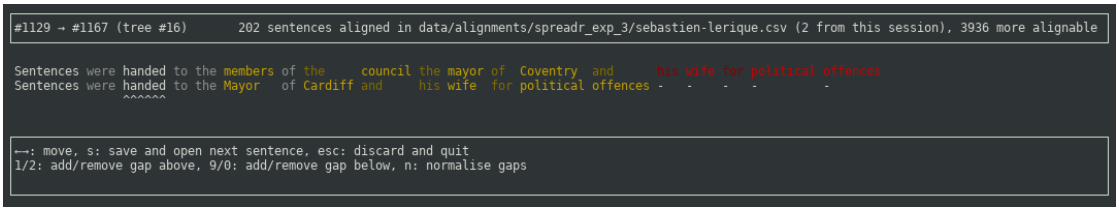
TODO: unify mathematics notations

The value of the fit thus loosely corresponds to the total number of words whose alignments would need to be changed in order to go from one set of alignments to the other. Divided by the number of transformations $|\mathcal{T}|$, it tells us the average number of word alignment errors per transformation. The worst fit $\hat{f}_n$ then gives us an upper bound estimation of the error that can be produced by training on a set of size n. One caveat in this evaluation approach is that there is no guarantee that the hand-coded alignments on which we will train could be produced by this parametrisation of alignments. We have no workaround for this caveat, other than hand-evaluation of the parameters after the training step.

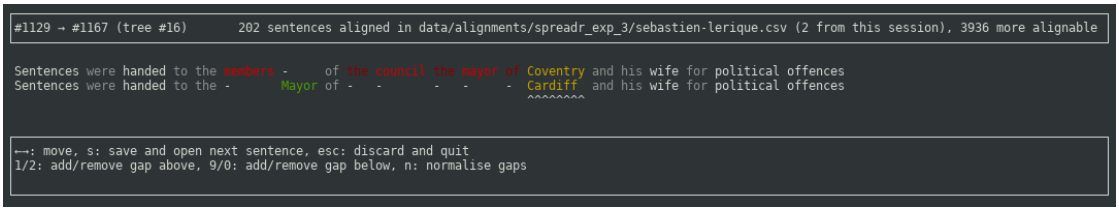
After having sampled a parameter set for step 1, we used $n = 20, 50, 100, 200$ and ran steps 2-4 ten times for each value of n. The worst values of the ten runs were $\hat{f}_{20} = 3652.5$ (1.71 errors per transformation), $\hat{f}_{50} = 1377.5$ (.65 errors per transformation), $\hat{f}_{100} = 847$ (.41 errors per transformation), and $\hat{f}_{200} = 636.5$ (.32 errors per transformation). For $n = 100$, we further resampled $\boldsymbol{\theta}^0$ ten times (step 1) and ran steps 2-4 ten times for each of those 10 parameter sets, yielding an overall $\hat{f}_{100} = 1437.5$, that is .70 errors per transformation. We conclude from this evaluation that a training set between 100 and 200 alignments is enough to reduce the final error below one word per transformation.

We thus developed a small console interface to hand-code 200 alignments of non-trivial transformations (see Fig. 3.9 for how this is done). The manual alignments were then used as a parameter training set, on which brute forcing the best $\boldsymbol{\theta} \in [-1, 0]^3$ with discretisation step up to .025 achieved a training fit of 240 (i.e., 1.2 errors per transformation), confirming that our hand-coded alignments are most likely not possible to reproduce perfectly with this parametrisation. The final parameters obtained with this approach are $\theta_{mismatch} = -.89$, $\theta_{open} = -.29$, and $\theta_{extend} = -.12$. $\theta_{exchange}$ was then set to $-.5$ after manual trial and error.

Finally, we manually evaluated the overall quality of these parameters by hand-coding the number of errors in a random set of 100 non-trivial alignments generated by the parameters. Errors were counted as the number of words whose alignment would have to be changed in order to obtain a perfect alignment. Of those 100 alignments, 79 were perfect, 12 had 1 error, 4 had 2 errors, and the remaining 5 had between 3 and 6 errors. Counting 1 error as acceptable, this method yields a successful alignment in 91% percent of the cases. To make sure this is also the case for deep alignments, we hand-coded errors in 100 random alignments for which the algorithm had explored exchanges (though not all of them are deep alignments, as it may be that the shallow alignment was the best). An error was counted for each exchange that was missing, mistaken, or should not have been present at all. Of the 100 alignments, 81 had no errors, 17 had 1, and 2 had 2 errors. The parameters obtained here were thus used for all further analyses. They are also the ones used in the example alignments discussed in the previous sections.



(a) Before manual alignment



(b) After manual alignment

Figure 3.9: Console interface for manual transformation alignment. The user moves their cursor (underline below "handed" and "Cardiff") along the word sequences to insert or remove gaps and align the two utterances by hand.

3.4.3 Streamlining the representation of transformations

Word filiations

Having developed a method to reliably break down individual transformations into simpler operations, we can come back to our initial goal of synthesising the overall phenomenon with intelligible building blocks. Ideally, we would like to know how often each basic operation of utterance alignment (deletion, replacement, exchange, conservation) is actually used by subjects, how those basic operations get grouped together, and how they depend on each other both inside a single utterance transformation and across successive transformations in branches. In more abstract terms, we are looking for the best way to dice the complete data into its internal components; such components need not be restricted to the level of individual utterance-to-utterance transformations, but can also span along branches and in a tree.

To do so we use the alignment tool we just developed to build a more synthetic representation of the branches. Indeed with the information provided by each alignment it is now possible to follow the ancestry and descent of individual words through parent and child transformations in a branch. Consider for instance a toy branch $u \to u' \to u''$. Any word $w' \in u'$ can be identified as a new insertion or affiliated to a parent word $w \in u$ that was conserved, replaced, and/or moved in an exchange. On the child side, w' can also be linked to its child w'' (if it wasn't deleted), thus continuing the lineage of this specific word along the branch.[19]

Fig. 3.10 gathers this information and plots the lineages of the branches for an example tree from Experiment 3 (tree #4, also shown in Fig. 3.4). The root of this tree is the following utterance:

[19] Note that the alignment tool produces a tree of possible deep alignments for each pair of utterances, such that a word in u could be assigned to different words in u' for different choices of deep alignments. To decide this uncertainty for a given word w we first determine if it is conserved (either exactly or through replacement) in at least of all the deep alignments; if so, we select the child word in u' which appears in most deep alignments (i.e. the majority child); if not, the operation is considered a deletion. Any word in u' that has no assigned parent word is considered an insertion.

> « At Dover, the finale of the bailiffs' convention. Their duties, said a speaker, are "delicate, dangerous, and insufficiently compensated." »

and the example transformation discussed in our introduction of deep alignments (# TODO: add refs for sentences) corresponds to the transition from depth 1 to depth 2 in branch #49 of the figure. This representation of branches provides our most synthetic view yet of the process at work. Stepping back, we can now examine which trends are more salient in the evolution along the branches. First observations are that the plots reflect the rapid shortening of utterances, and that transformations are less important on shorter utterances. We also see here that word replacements, the process studied in the previous chapter with data from blogspace, is quite speckled, and both less frequent and of smaller magnitude than word deletions and word insertions. This obvious caveat was one of the motivations for our current experimental approach (indeed, replacements were the only process that could be extracted from the blogspace data set).

Branch and utterance axes

The plots show noticeable regularities in the way transformations vary. For a given branch, we distinguish between the two axes of Fig. 3.10 as two scales of analysis, each of which corresponds to a different set of events. First the horizontal axis: an event at this level is the bulk transformation or conservation of an utterance by a subject, without going into the detail of the way an utterance changes. This yields a series of conservation and transformation events, one at each depth in the branch. We call this the branch dimension, pictured in Fig. 3.11a. A salient feature on this dimension is the apparent burstiness of transformations. Since successive subjects perform transformations independently, confirming this trend would indicate a behaviour reminiscent of punctuated equilibria: a transformation occurring after a period of stability would result in a new utterance that is more likely to be transformed again (we quantify this trend below).

Second, the vertical axis where we delve into the detail of a transformation seen as a set of word insertions, deletions, conservations and replacements with or without exchange. Call this the utterance dimension. At this stage of our analysis, an ideal tool to quantify the properties of transformations in the utterance dimension would be a generative model of the transformative process. Unlike in the branch dimension however, the transformation of an utterance is a multi-level process that does not reduce easily to an ordered series of events. Indeed, manually inspecting the branches' transformations indicated several trends that complicate potential generative models (several of these are visible in Fig. 3.10):

- Deletions, exchanges, and insertions seem bursty (that is they appear in large chunks – a behaviour that replacements do not seem to have), and the bursts seem often longer if the utterance they appear in is longer,
- Insertions seem to rarely occur without deletions; when they appear with deletions, the two tend to be close to each other and of similar magnitude,
- All operations are less frequent at the very beginning of utterances.

Note that burstiness at the word level is no surprise, as words are not processed independently and transforming parts of an utterance is likely to depend on syntactic and semantic boundaries. However, combined with the dependencies between the sizes and positions of insertions and deletions, and the possibility for exchanges, it means that a full generative model capable of reproducing and predicting future transformations will most likely involve memory and attention span mechanisms that are beyond the scope of this chapter. We therefore wish to extract the regularities of individual utterance transformations without modelling their genesis.

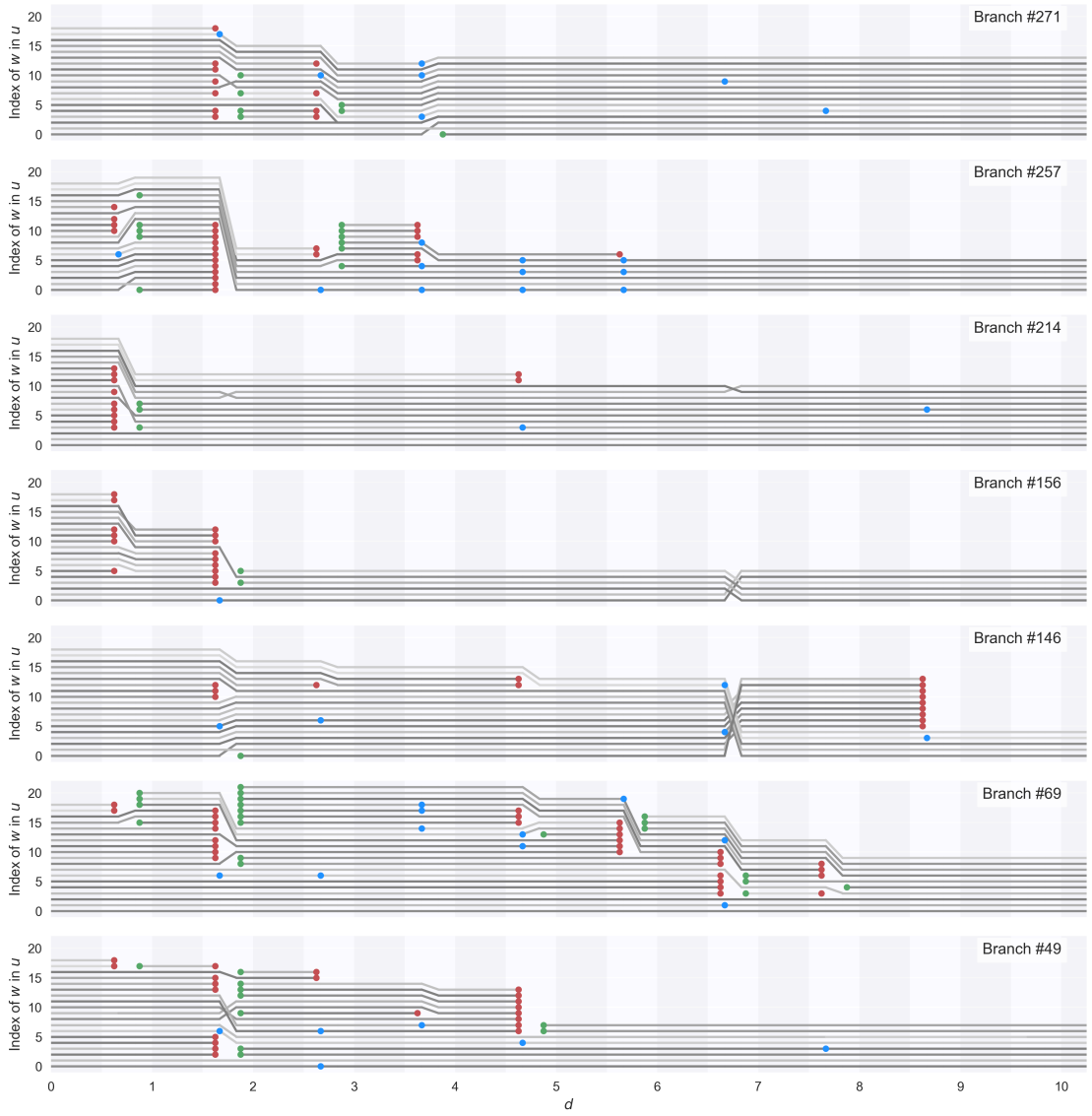


Figure 3.10: Example lineages for all the branches of tree #4 from Experiment 3. Each subplot corresponds to a different branch. The horizontal axis is the depth in the branch, and the vertical axis is the index of each word in its utterance. A grey line represents a word lineage along the branch, and the darkness of the line corresponds to that word's frequency (darker lines represent more frequent words). At each depth, the darker background band indicates what the subject sees, and the lighter band indicates the transformation that the subject made. Inside lighter bands: red dots are word deletions, green dots are word insertions, blue dots are word replacements, and exchanges can be seen when bundles of lines cross each other. Dots inside each light band are spread out on the horizontal axis so as make them easier to distinguish visually, but the horizontal position of a dot inside its band has no further meaning.

To do so we still need a canonical representation of transformations whose properties we can examine without being biased by our choice of representation (such a representation could be the output of a generative model). Recall that the alignment tool we developed encodes a transformation as a pair of word sequences with gaps, an encoding that is not unique: insertions and deletions that happen together can be reordered (putting insertions before deletions instead of the other way around, or alternating an insertion with a deletion), and the exchange of two parts around a stable chunk can be re-encoded by inverting the roles of stable and exchanged chunks, all without changing the transformation represented by the encoding. Running statistics on this representation is thus not an option. However, compressing the gaps in this representation merges all the variations together and produces an encoding that is in bijection with transformations. This compressed representation of a transformation is precisely what the lineage plots of Fig. 3.10 achieve, through another path.[20] We picture this correspondence between the transformation diagram produced by lineage plots and the compressed form of alignment sequences in Fig. 3.11b. We thus take these diagrams as our canonical representation of a transformation.

In what follows we further set aside part of the information provided by exchanges. The natural way of analysing an exchange in a transformation diagram is to look at it as the permutation of the words affected, with possible replacements, insertions and deletions added afterwards. We chose to leave the analysis of this aspect of transformations to further research, focusing instead on insertions, deletions, and replacements only. We therefore simplify exchanges to keep only the information on whether an exchanged word was conserved or replaced.[21]

From a given transformation diagram we then extract two arrays of word-level operations,[22] one for the parent utterance and one for the child utterance. The parent array contains conservation, replacement and deletion operations, and the child array conservation, replacement and insertion operations. The diagram further provides us with the correspondence of conservation and replacement operations between the two arrays (except for operations that were involved in an exchange, for which we lose position information), such that we can measure the distance between two blocks of insertion and deletion operations (except if the two blocks are separated by operations involved in an exchange). This final representation of transformations is pictured in Fig. 3.11c.

Bursty behaviours

We now wish to quantify the regularities of transformations on both dimensions. Our first step is to measure the extent to which each dimension does indeed feature bursty behaviour. Following Jo et al. (2012; who rely on Goh and Barabási 2008), we measure the burstiness of a series of events through the parameter B defined as

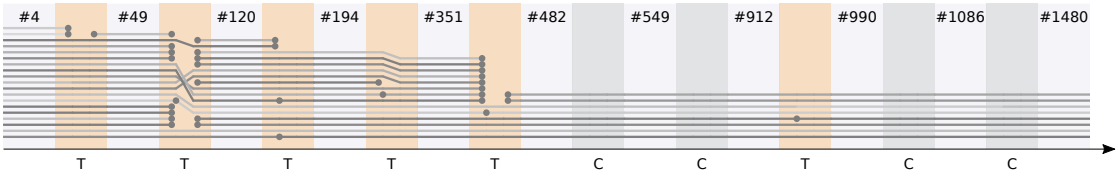
$$B = \frac{\sigma - \mu}{\sigma + \mu}$$

where σ and μ are respectively the standard deviation and mean of the distribution of inter-event times in the series of events. The same computation applies to arrays of operations (the two have

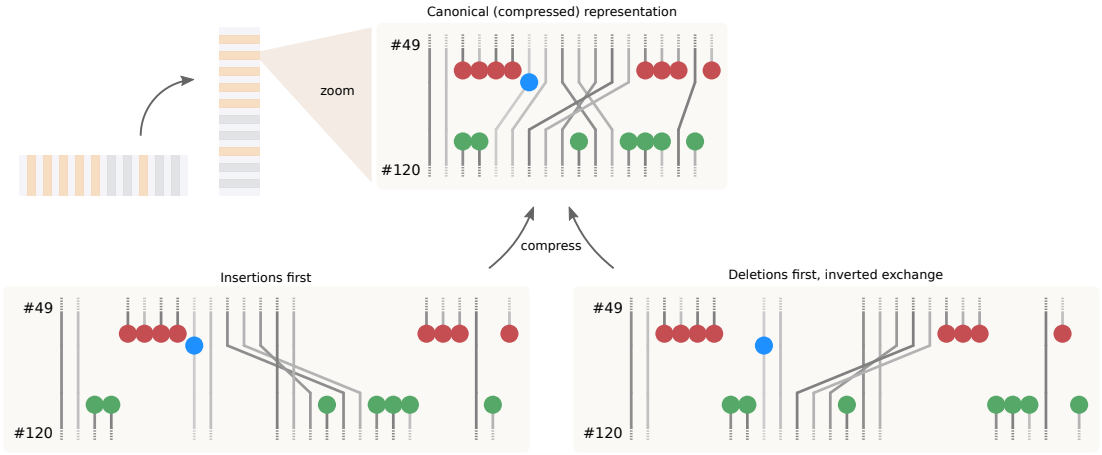
[20] The lineage plots are built horizontal line by horizontal line, by following the path of each word along the branch, not by compressing utterance transformations.

[21] Note that while we do not analyse exchanges per se, we still benefit from the deep alignment tool developed: the insertions and deletions we are left with correspond to real appearances and disappearances, not undetected exchanges.

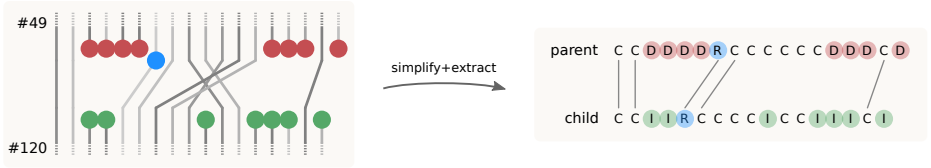
[22] We use the phrase "array of operations", and not "series of events", to emphasise that these operations appear linearly in the utterances, but do not necessarily come from a one-dimensional generation process. The two terms refer to the same mathematical object, and simply change the interpretation of the index: for a series of events the index represents time, for an array of operations it does not.



(a) Branch dimension. This level looks at whether or not an utterance is transformed, without going into the detail of changes (hence the greyed out dots). Similar to Fig. 3.10, light grey bands are what subjects see, and the bands between those represent what the subjects do with what they read. An orange band indicates that an utterance was transformed, that is a T event, and a dark grey band indicates that an utterance was perfectly conserved, that is a C event. The corresponding ordered series of events is shown underneath the axis' arrow.



(b) Utterance dimension. This level looks at the detail of a transformation, and represents it with a diagram that compresses the pair of sequences produced by aligning parent and child utterances. This diagram uniquely represents the transformation, and merges any variations in encoding that can exist in pairs of sequences with gaps. The top level of the figure shows how the canonical representation comes from the lineage plots. The bottom level shows two equivalent encodings of the same transformation (as would be produced by the alignment tool), which compress to the same canonical representation.



(c) Parent and child arrays of operations. The canonical representation is further simplified by discarding the change in position encoded by word exchanges, and only keeping the information on whether a word is conserved or replaced. This procedure results in two arrays of word operations: a parent array made of conservations (C), replacements (R) and deletions (D), and a child array made of conservations, replacements and insertions (I). Conservations and replacements in the parent array, if not involved in exchanges, are linked to their corresponding operation in the child array, such that we can compute the distance between a block of insertions in the child and a block of deletions in the parent (except when exchanges separated the blocks).

Figure 3.11: Analysis dimensions. Transformations are analysed along two dimensions. The branch dimension only looks at whether utterances are transformed or not, thus sees a series of T (transformed) and C (conserved) events. The utterance level looks at the detail of the transformations, and after simplification represents them with two arrays of operations, one for the parent utterance (made of C, R and D operations) and one for the child utterance (made of C, R and I operations). This example is built on branch #49 from Fig. 3.10. # TODO: Make this work in grayscale

the same mathematical description). B has values between -1 and 1; $B = -1$ corresponds to a perfectly regular process ($\sigma = 0$, and $\mu > 0$ is the constant period between events), $B = 0$ indicates a burstiness equivalent to that of a Poisson process, where the occurrence of a new event does not depend on the presence of previous events (and $\sigma = \mu$), and $B = 1$ corresponds to an asymptotically perfectly bursty process (it is the limit $\mu/\sigma \to 0$). In other words, B measures the intuition according to which a process with average inter-event time shorter than its standard deviation will often have events close to each other with a few long periods without events, and a process with an average inter-event time longer than its standard deviation will have events more evenly spaced (relative to their mean spacing).

In the branch dimension, we consider a transformation to be an event and a conservation to be the absence of an event. Note that our data in this dimension is truncated due to branches not being infinite. When the last subject in a branch does not transform the utterance they reproduce, we do not observe the actual duration of that stability: had the branch continued, the stability could have been interrupted immediately, or could have lasted for many more reproductions of the utterance. Including these truncated spans in the distribution of inter-event times artificially inflates the burstiness (because it adds underestimated spans to the distribution), but removing them biases our sample towards inter-event times for longer utterances (earlier in the branch), which could also inflate burstiness. We thus present measures for both distributions, with and without the truncated spans.

Burstiness in the branch dimension with truncated spans is $B_{branch,all} = 0.251 \pm 0.029$, and burstiness without the truncated spans is $B_{branch,observed} = 0.305 \pm 0.031$ (both error estimates correspond to the 95% confidence interval). Both measures show that the transformation process in the branch dimension is significantly bursty. This is consistent with our intuition that when a transformation appears after a period of stability, it is likely to trigger other transformations following it until a new stable (often much shorter) utterance is reached.

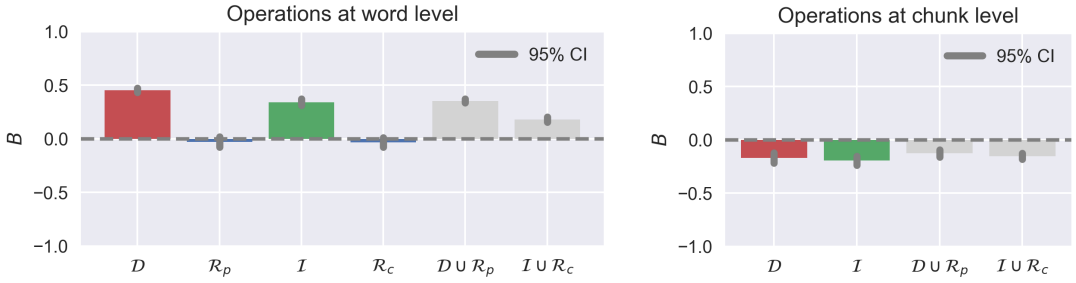
The situation in the utterance dimension involves more event types. In the parent array, we note the series of deletion events $\mathcal{D}$ and the series of replacements $\mathcal{R}_p$. In the child array, we note the series of insertion events $\mathcal{I}$ and the series of replacements $\mathcal{R}_c$. A conserved word is considered an absence of event. Note that because of inserted and deleted words, replacements may not appear with the same distributions in the parent and child arrays. As a consequence, $\mathcal{R}_p$ and $\mathcal{R}_c$ may not have the same distribution of inter-event times. The burstiness measures for each of these series are shown in Fig. 3.12a, along with the burstiness of the series made of all parent or child events without distinguishing their type. The plots show that deletions and insertions are both bursty, while replacements are undistinguishable from a non-bursty process such as a Poisson process. When all event types are joined together, the process is also bursty, albeit slightly less.

Given the strength of this behaviour for deletions and insertions, we further look at these series by collapsing each contiguous chunk of deleted or inserted words into a single event. This leads to a series of deletion and insertion chunks separated by word replacements and word conservations (non-events). For inter-event times, it corresponds to removing the null values in the previous distributions of inter-event times (which separated words in the same chunk); computing the burstiness of the chunk process is therefore straightforward. The values plotted in Fig. 3.12b show that none of the chunk processes are bursty; rather, they are slightly more regular than a Poisson process would be.

Although this behaviour is consistent with our intuition of the way an utterance is reformulated, there is a question as to whether the alignment tool we developed does not favour burstiness. In other words, are these measures not due to artefacts? Indeed, the scores of operations are parametrised

in such a way that insertion and deletion gaps are assigned different costs for initial opening and extension. Let us answer this concern. While the parametrisation of costs does make it possible for burstiness to be more easily identified, it does not make it a necessity: setting the gap opening cost to the same value as the gap extension cost would make the alignment tool neutral with respect to burstiness (setting it lower would be biased against burstiness, and the alignment algorithm would favour word mismatches over gaps to encode differences). In our case, the parameters we trained set the gap opening cost to a much higher value than the extension cost (.29 vs. .12 in absolute values), such that the alignment tool does find bursty insertions and deletions more easily. However, these parameters are learned from hand-coded alignments and their output has been validated on test samples: any bursty insertions or deletions detected by the alignments is therefore the product of the data itself.

We now move on to further characterising the size, position, and dependencies between replacements and insertion and deletion chunks.



(a) Burstiness of word operations

(b) Burstiness of chunks

Figure 3.12: Burstiness of operations in the utterance dimension. The left pane shows the burstiness of each type of word-level operation (in parent and child arrays), as well as the burstiness of the series made of all operations joined regardless of their type. The right pane shows the burstiness for deletions, insertions, and joined events, where contiguous blocks of operations are collapsed into single events. This corresponds to the burstiness of *chunks* of deletions, insertions, and joined events (i.e. only considering strictly positive inter-event times). Grey lines are the 95% confidence intervals.

3.4.4 Inner structure

3.4.5 High-level case studies

3.4.6 Meaning change

3.5 Discussion

TBD

Chapter 4

Discussion

Chapter 5

Conclusion

References

Bangerter, Adrian. 2000. 'Transformation Between Scientific and Social Representations of Conception: The Method of Serial Reproduction'. *British Journal of Social Psychology* 39 (4): 521–35. doi:10.1348/014466600164615.

Barrett, Justin L., and Melanie A. Nyhof. 2001. 'Spreading Non-Natural Concepts: The Role of Intuitive Conceptual Structures in Memory and Transmission of Cultural Materials'. *Journal of Cognition and Culture* 1 (1): 69–100. doi:10.1163/156853701300063589.

Bartlett, Sir Frederic Charles. 1995. *Remembering: A Study in Experimental and Social Psychology*. Cambridge University Press.

Christie, Tom, and Django REST framewor contributors. 2017. 'Django REST Framework'. http://www.django-rest-framework.org/.

Claidière, Nicolas, Kenny Smith, Simon Kirby, and Joël Fagot. 2014. 'Cultural Evolution of Systematically Structured Behaviour in a Non-Human Primate'. *Proceedings of the Royal Society of London B: Biological Sciences* 281 (1797): 20141541. doi:10.1098/rspb.2014.1541.

Cock, Peter J. A., Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, et al. 2009. 'Biopython: Freely Available Python Tools for Computational Molecular Biology and Bioinformatics'. *Bioinformatics* 25 (11): 1422–3. doi:10.1093/bioinformatics/btp163.

Czaplicki, Evan, and Elm contributors. 2017. 'Elm: A Delightful Language for Reliable Webapps'. http://elm-lang.org/.

Danescu-Niculescu-Mizil, Cristian, Justin Cheng, Jon Kleinberg, and Lillian Lee. 2012. 'You Had Me at Hello: How Phrasing Affects Memorability'. In *ACL '12 Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers*, edited by Haizhou Li, Chin-Yew Lin, and Miles Osborne, 1:892–901. Stroudsburg, PA: ACM. http://arxiv.org/abs/1203.6360.

Ember.js contributors. 2017. 'Ember.js: A Framework for Creating Ambitious Web Applications.' https://emberjs.com/.

Fénéon, Félix, and Luc Sante. 2007. *Novels in Three Lines*. New York, NY, USA: New York Review Books.

Goh, K.-I., and A.-L. Barabási. 2008. 'Burstiness and Memory in Complex Systems'. *EPL (Europhysics Letters)* 81 (4): 48002. doi:10.1209/0295-5075/81/48002.

Heath, Chip, Chris Bell, and Emily Sternberg. 2001. 'Emotional Selection in Memes: The Case of Urban Legends'. *Journal of Personality and Social Psychology* 81 (6): 1028–41. doi:10.1037/0022-

3514.81.6.1028.

Jo, Hang-Hyun, Márton Karsai, János Kertész, and Kimmo Kaski. 2012. 'Circadian Pattern and Burstiness in Mobile Phone Communication'. *New Journal of Physics* 14 (1): 013055. doi:10.1088/1367-2630/14/1/013055.

Kashima, Yoshihisa. 2000. 'Maintaining Cultural Stereotypes in the Serial Reproduction of Narratives'. *Personality and Social Psychology Bulletin* 26 (5): 594–604. doi:10.1177/0146167200267007.

Keuleers, Emmanuel, and Marc Brysbaert. 2010. 'Wuggy: A Multilingual Pseudoword Generator'. *Behavior Research Methods* 42 (3): 627–33. doi:10.3758/BRM.42.3.627.

Krosnick, Jon A. 2000. 'The Threat of Satisficing in Surveys: The Shortcuts Respondents Take in Answering Questions'. *Survey Methods Newsletter* 20 (1): 4–8.

Leeuw, Edith D. de, Joop J. Hox, and Don A. Dillman. 2008. *International Handbook of Survey Methodology*. New York, NY, USA ; London, UK: Taylor & Francis.

Maxwell, R. S. 1936. 'Remembering in Different Social Groups'. *British Journal of Psychology. General Section* 27 (1): 30–40. doi:10.1111/j.2044-8295.1936.tb00814.x.

Mesoudi, Alex, and Andrew Whiten. 2004. 'The Hierarchical Transformation of Event Knowledge in Human Cultural Transmission'. *Journal of Cognition and Culture* 4 (1): 1–24. doi:10.1163/156853704323074732.

———. 2008. 'The Multiple Roles of Cultural Transmission Experiments in Understanding Human Cultural Evolution'. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 363 (1509): 3489–3501. doi:10.1098/rstb.2008.0129.

Mesoudi, Alex, Andrew Whiten, and Robin Dunbar. 2006. 'A Bias for Social Information in Human Cultural Transmission'. *British Journal of Psychology* 97 (3): 405–23. doi:10.1348/000712605X85871.

Moussaïd, Mehdi, Henry Brighton, and Wolfgang Gaissmaier. 2015. 'The Amplification of Risk in Experimental Diffusion Chains'. *Proceedings of the National Academy of Sciences* 112 (18): 5631–6. doi:10.1073/pnas.1421883112.

Needleman, Saul B., and Christian D. Wunsch. 1970. 'A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins'. *Journal of Molecular Biology* 48 (3): 443–53. doi:10.1016/0022-2836(70)90057-4.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. 2014. 'GloVe: Global Vectors for Word Representation'. In, 1532–43. http://www.aclweb.org/anthology/D14-1162.

Rayner, Keith, Timothy J. Slattery, and Nathalie N. Bélanger. 2010. 'Eye Movements, the Perceptual Span, and Reading Speed'. *Psychonomic Bulletin & Review* 17 (6): 834–39. doi:10.3758/PBR.17.6.834.