# UROP International Summer 2019

## Automatic Assessment of Text Complexity (1001)
## Keystroke Logging in Second Language Writing Research (1002)

Project leader: PD Dr. Elma Kerz

May 20, 2019

# Welcome!

- All relevant information can be found in our shared repositories:
  - https://git.rwth-aachen.de/
  - Automatic Assessment of Text Complexity (keystroke)
  - Keystroke Logging in Second Language Writing Research (cocogen)
- Timetable: Details in UROP.Schedule.2019.pdf
- We have 8 weeks together
- Week 9: Presentation & system demo
  - Automatic analysis of complexity of raw text and keystroke logging data

# Schedule

- Week 1: Familiarization with relevant background
  - Key readings in repository (point of departure)
- Week 2:
  - Tuesday: Visit Marcus Ströbel
- Week 3:
  - TBA
- Week 4:
  - TBA

# Automatic assessment of linguistic complexity

- Advances in natural language processing have paved the way for the development of computational tools designed to automatically assess the complexity of spoken and written language samples. There are a variety of computational tools available that afford speed, flexibility and reliability and permit the direct comparison of numerous indices of language complexity. Considerable gains have been made from the use of such tools with respect to identification of reliable and valid measures of linguistic complexity to predict text readability, writing quality or speaking proficiency. However, all these available tools only generate for each complex measure a single score (summary statistics) that represents the global complexity of a language sample. We have developed a computational tool that implements a sliding-window approach to track changes in complexity within a language sample derived from a series of measurements obtained by the tool.

# Automatic assessment of linguistic complexity

- An increased interest in the combined use of NLP techniques and machine learning algorithms for automatic text categorization tasks, e.g.
  - author recognition and verification (Van Halteren, 2004)
  - native language identification e.g. (Malmasi et al., 2017)
  - readability assessment (Francois and Miltsakaki, 2012)
  - text genre identification (Xu et al., 2017, Ströbel et al., 2019)

# Automatic assessment of linguistic complexity

- Research on linguistic complexity has benefitted greatly from the use of computational tools that automatically measure complexity of a text.
  - Coh-Metrix (McNamara et al., 2014)
  - L2 Syntactic Complexity Analyzer (Lu, 2010)
  - Lexical Complexity Analyzer (Lu, 2012)
  - TAALES (Kyle & Crossley, 2015)
  - TAASSC (Kyle, 2016)
- These tools provide comprehensive assessments of text complexity at a global level
  - produce for each measure a single score that represents the complexity of a text (a summary statistics)

# Complexity Contour Generator (CoCoGen)

- Novel approach to the automatic assessment of of complexity
  - **Adopts a sliding-window approach to track the changes in complexity within a text (*complexity contours*)**

# Implemented Complexity Measures

- Currently, 32 measures from four classes:
  - Syntactic complexity
  - Lexical complexity
    - Lexical diversity
    - Lexical density
    - Lexical sophistication
      (frequency lists, NGSL, NAWL, LFP, AFL)
  - Morphological complexity
    - mean length of word
    - mean syllables per word
  - Information theoretic measures (Kolmogorov complexity)

# Complexity per window

- measures are represented as fractions for each sentence
- calculating window $n$ for $measure_m = \frac{num_m}{den_m}$

$$window_n = \frac{num_n + num_{n+1} + ... + num_{n+ws}}{den_n + den_{n+1} + ... + den_{n+ws}}$$

- Why?
  - fractions: correct overall value
  - counting measures: fixed denominator of 1 $\rightarrow$ arithmetic mean
  - different window sizes remain comparable
  - caching, efficiency

# Assessment of text complexity

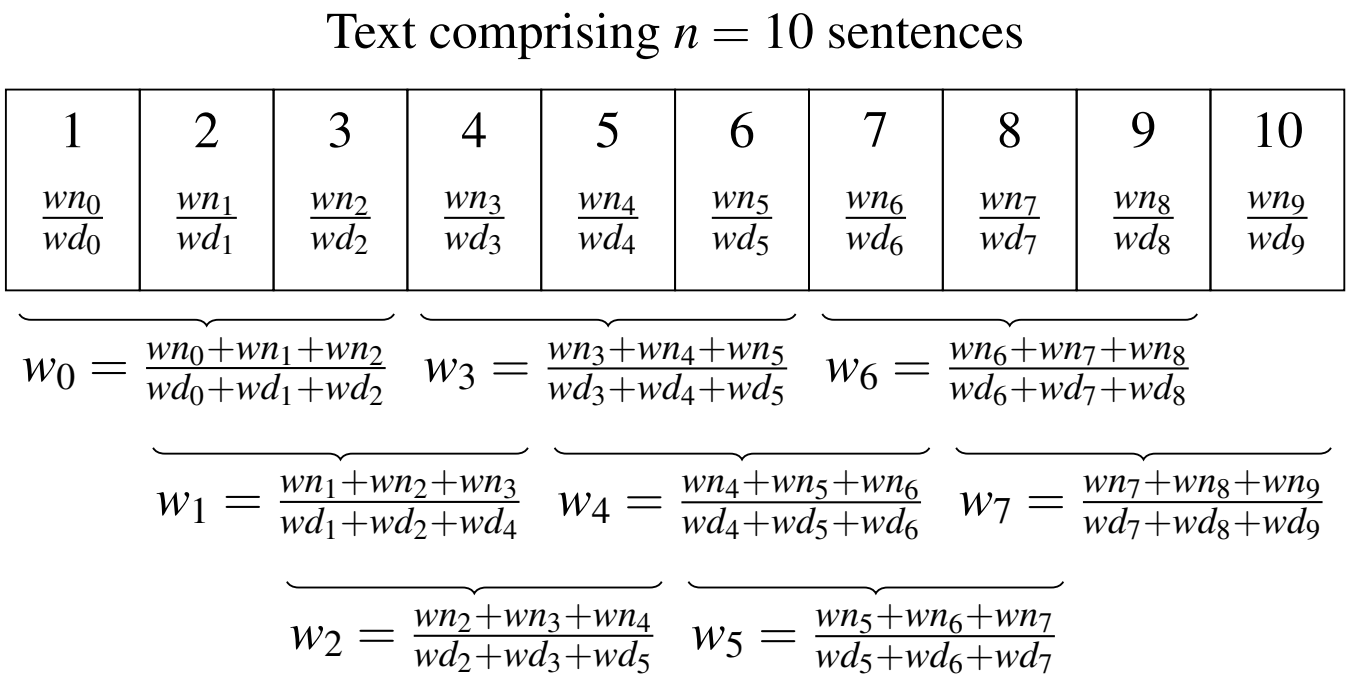



Figure: Schematic illustration of how complexity measurements are obtained in CoCoGen for a text comprising ten sentences with a window size of three sentences.

# Assessment of text complexity: Scaled output

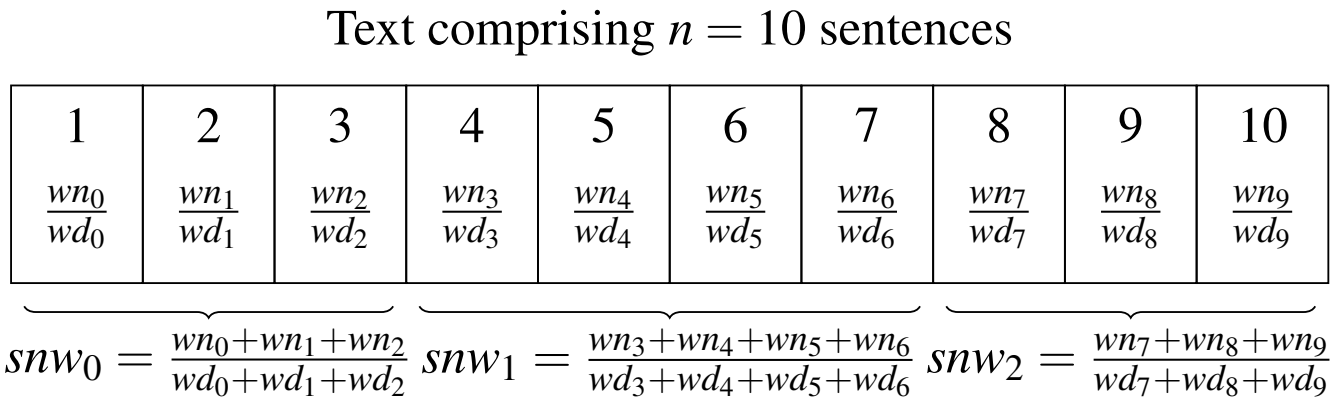



Figure: Schematic Illustration of the complexity measurements obtained by CoCoGen for a text comprising ten sentences with the number of scaled windows set to 3.

# Complexity Contours

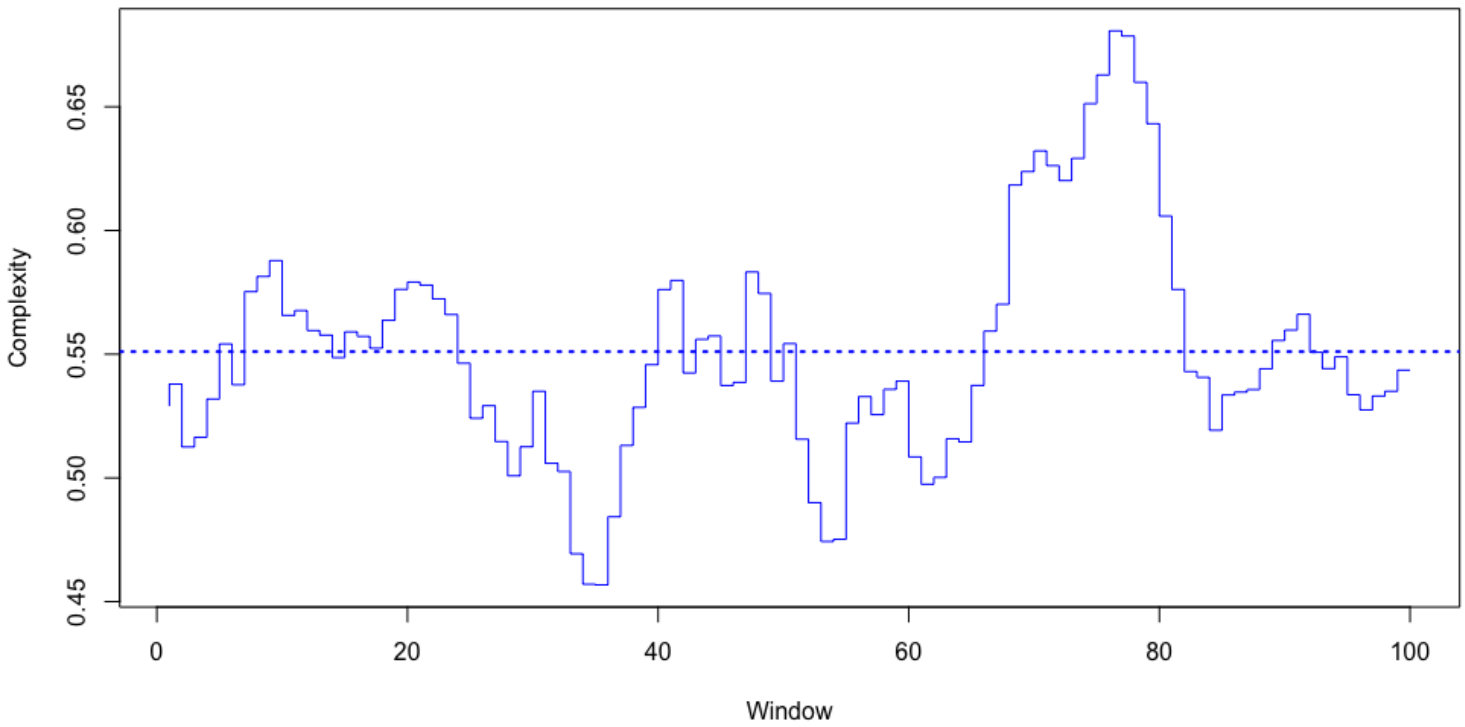



Figure: Complexity contour of a text (measure: Lexical Sophistication (BNC))

# Complexity Contours

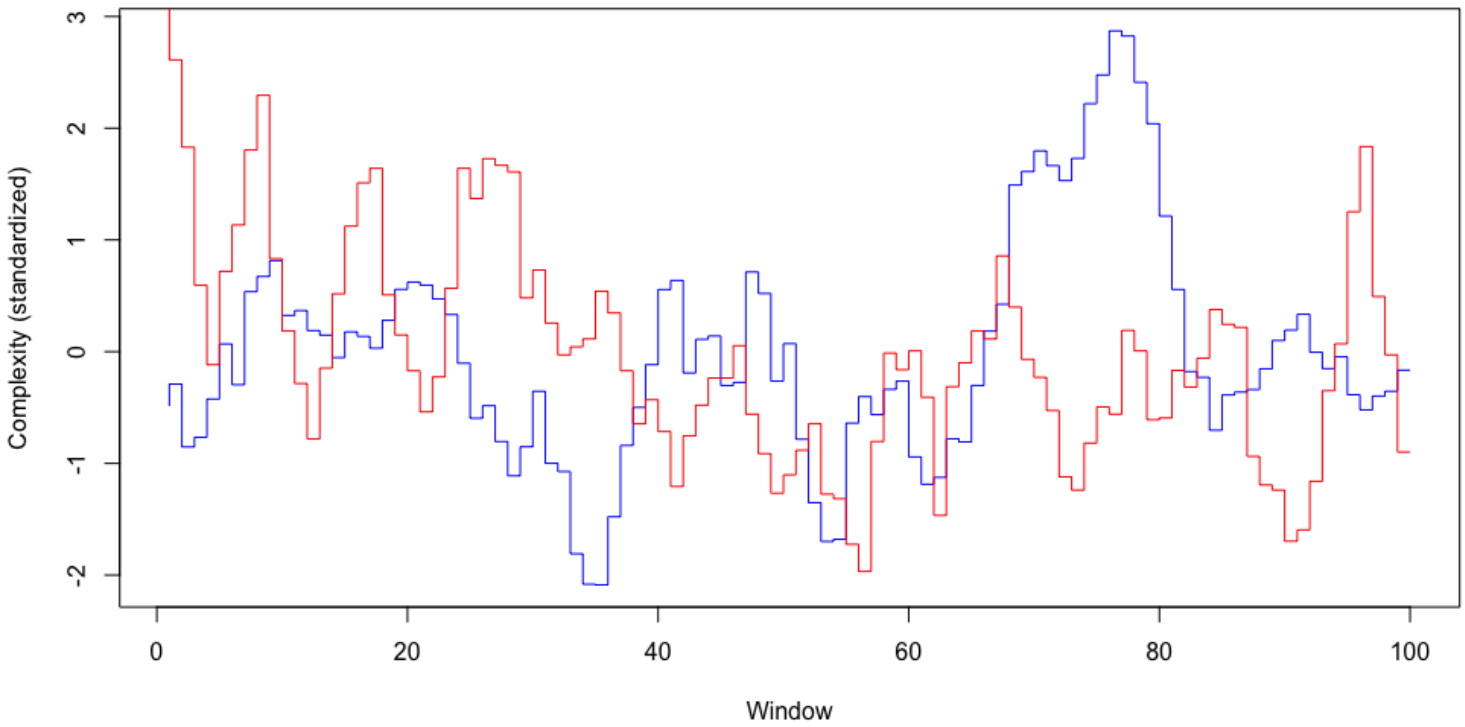



Figure: Complexity contours of a text (measures: Lexical Sophistication (BNC), Nominals per T-Unit)

# Complexity Contours: Comparing corpora

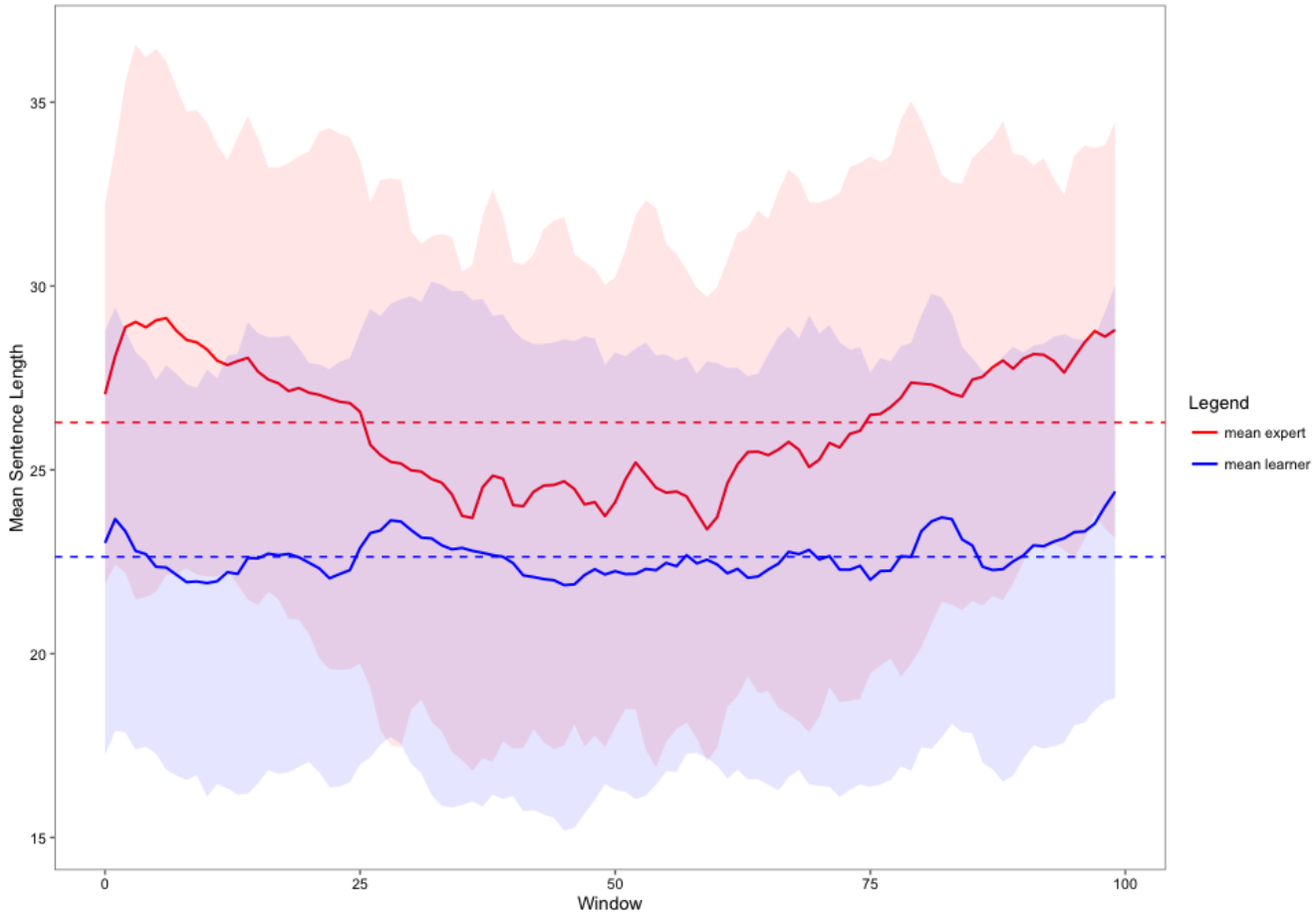
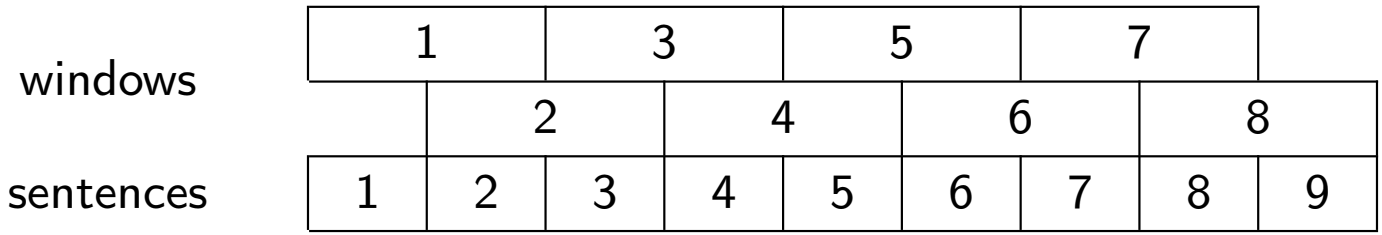


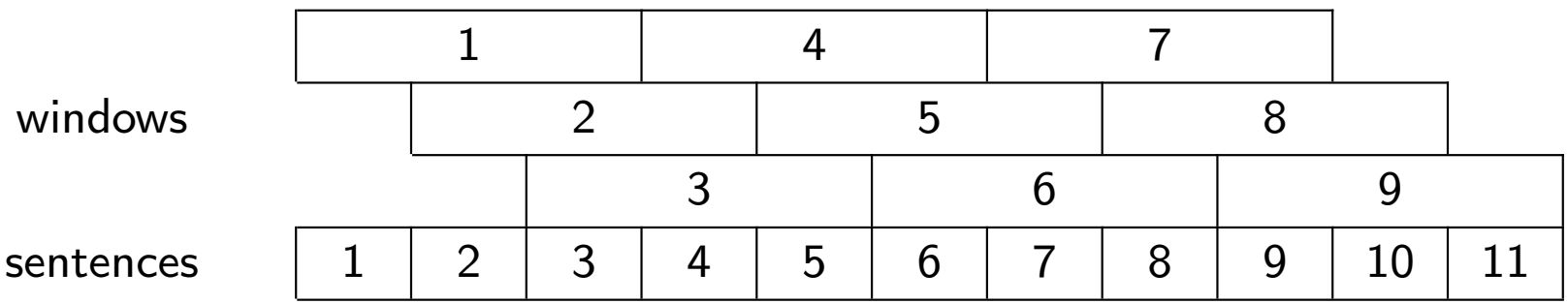

Figure: Distribution of complexity for a single measure in two corpora

# Variable Window Size

- movable window with fixed "window size" ($ws$) in sentences




$ws = 2$




$ws = 3$

- for $n$ sentences, we get $n - ws + 1$ windows

# Window Size & Smoothing

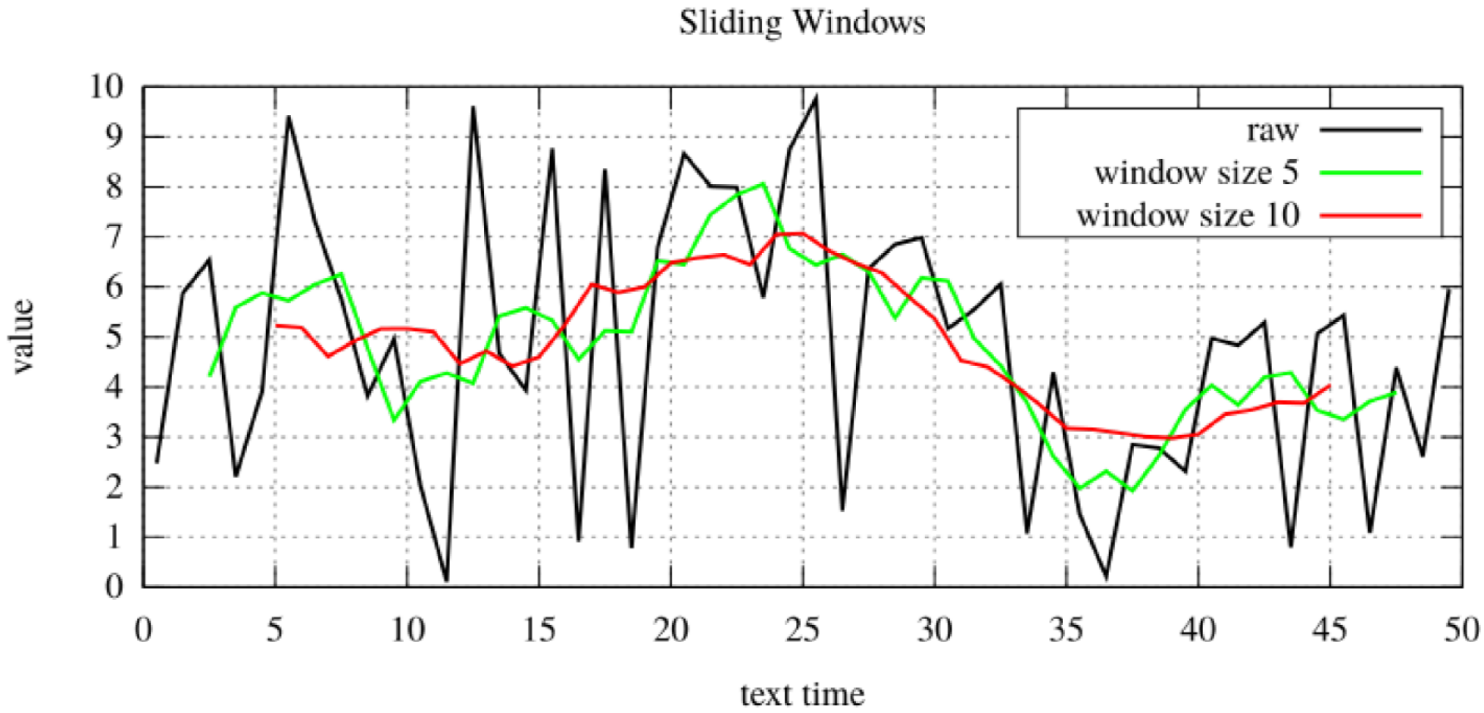



Figure: Sliding windows for window sizes 5 and 10 compared to raw data

# Application - Example 1: Text Genre Classification Based on Linguistic Complexity Contours Using A Recurrent Neural Network (Ströbel et al., 2019)

- Study set out to determine whether and to what extent genre classification can benefit from inclusion of sequence information, i.e. by the incorporation of the information regarding the progresssion of complexity within a text.

# Data

1. Balanced subsample form Corpus of Contemporary American English (COCA) (Davies, 2008)
   - 1.1 balanced corpus of American English
   - 1.2 560 million words of text (20 million words each year 1990-2017)
   - 1.3 equally divided among five general genres:
     - 1.3.1 spoken, fiction, popular magazines, newspapers, academic texts
2. Sample: 10,500 texts (random sampling of 2,100 texts from each of these five genres)

# Feature extraction

- Complexity contours were obtained using CoCoGen with a window size of 10 sentences over all 10,500 texts
- For each text, we extracted a feature sequence:
  - a series of length n − 10 + 1 32 dimensional feature vectors generated at each window position, where n is the number the sentences in a text.

# Training & test sets

- After normalization and padding of the feature sequences, data were divided into
  - a balanced training set of 10,000 feature sequences (2,000 texts per genre)
  - a balanced test set of 500 feature sequences (100 texts per genre)

# Comparison dataset

- Is classification accuracy improved by inclusion sequence information, rather than average text complexity only?
- Comparison dataset: collapsed each unnormalized feature sequences to its mean vector
- We used the same COCA subset of 10,500 texts described above to train and test the global classification model.

# Results: Complexity Contours from a single sample of Academic Writing (scaled output)

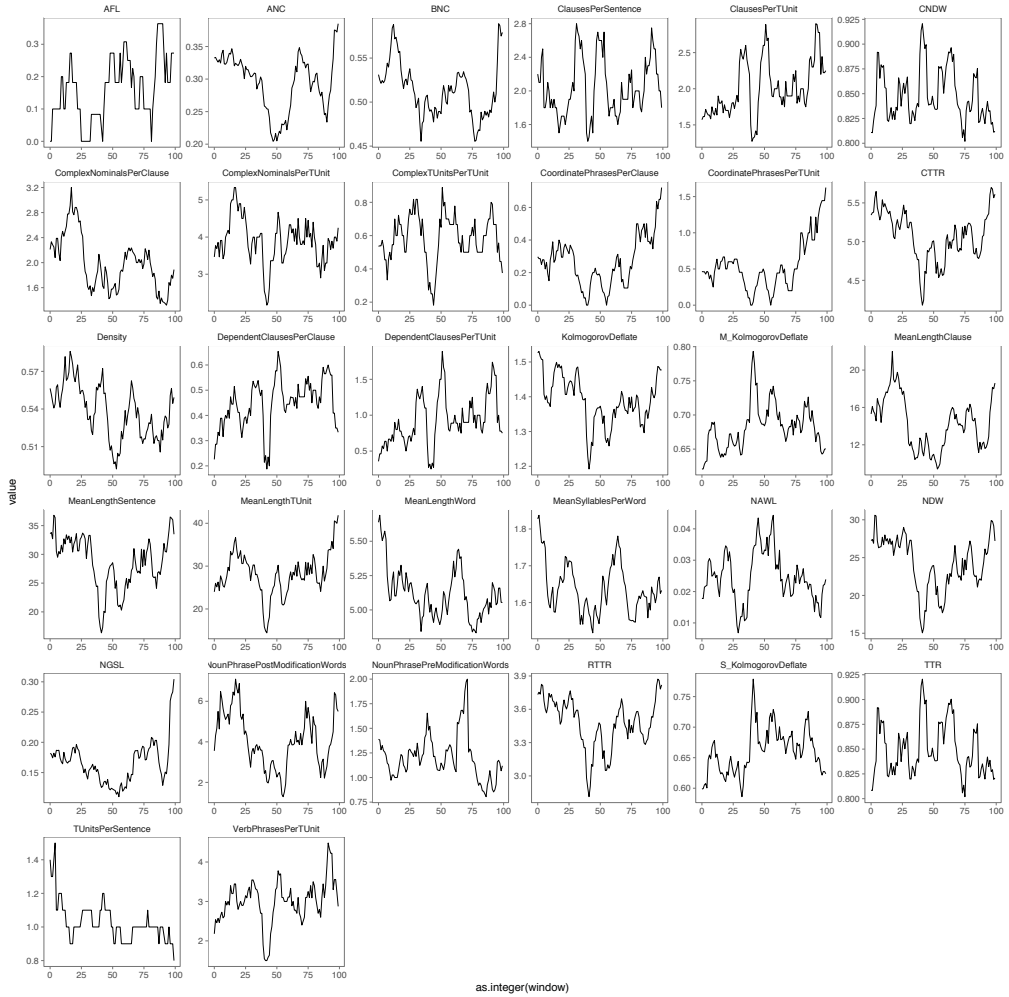



Figure: Progression of complexity within a single text from the genre of academic writing across all measures)

# Results: Compensatory effects between lexical and syntactic cpomplexity (scaled output)

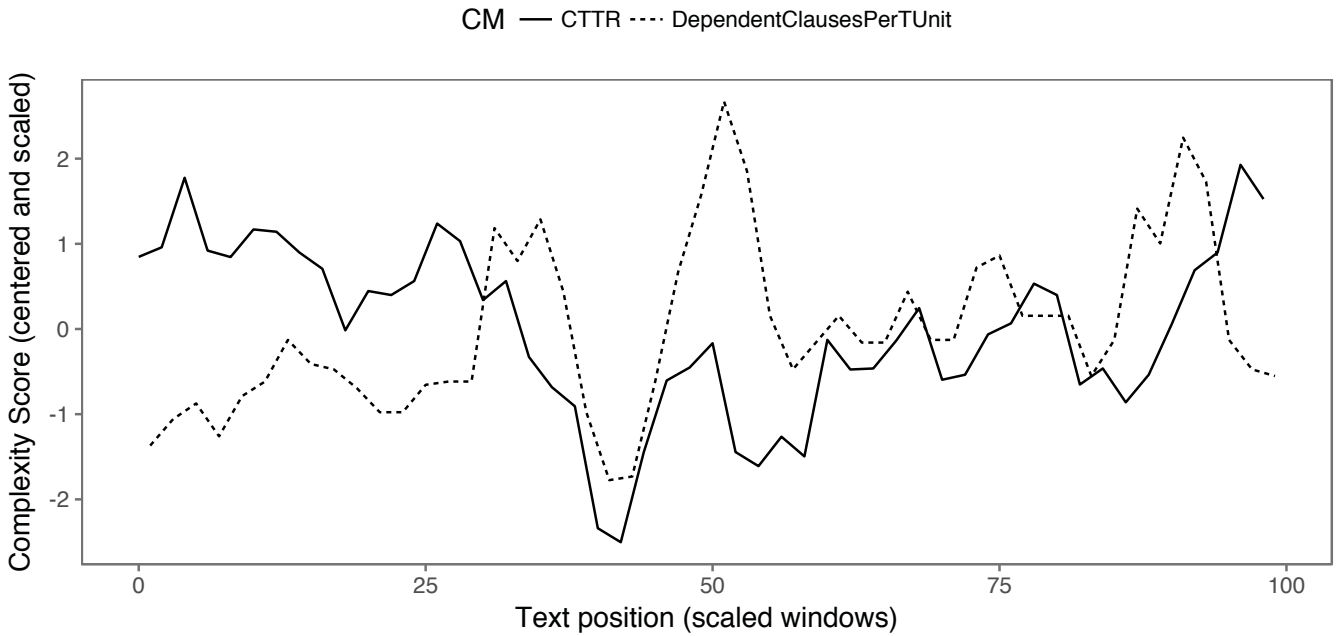



Figure: Progression of complexity within a single text from the genre of academic writing for two measures of complexity (corrected~type token ratio and Dependent~Clauses per T-Unit )

# Results: Average complexity contours across genres and measures (scaled output)

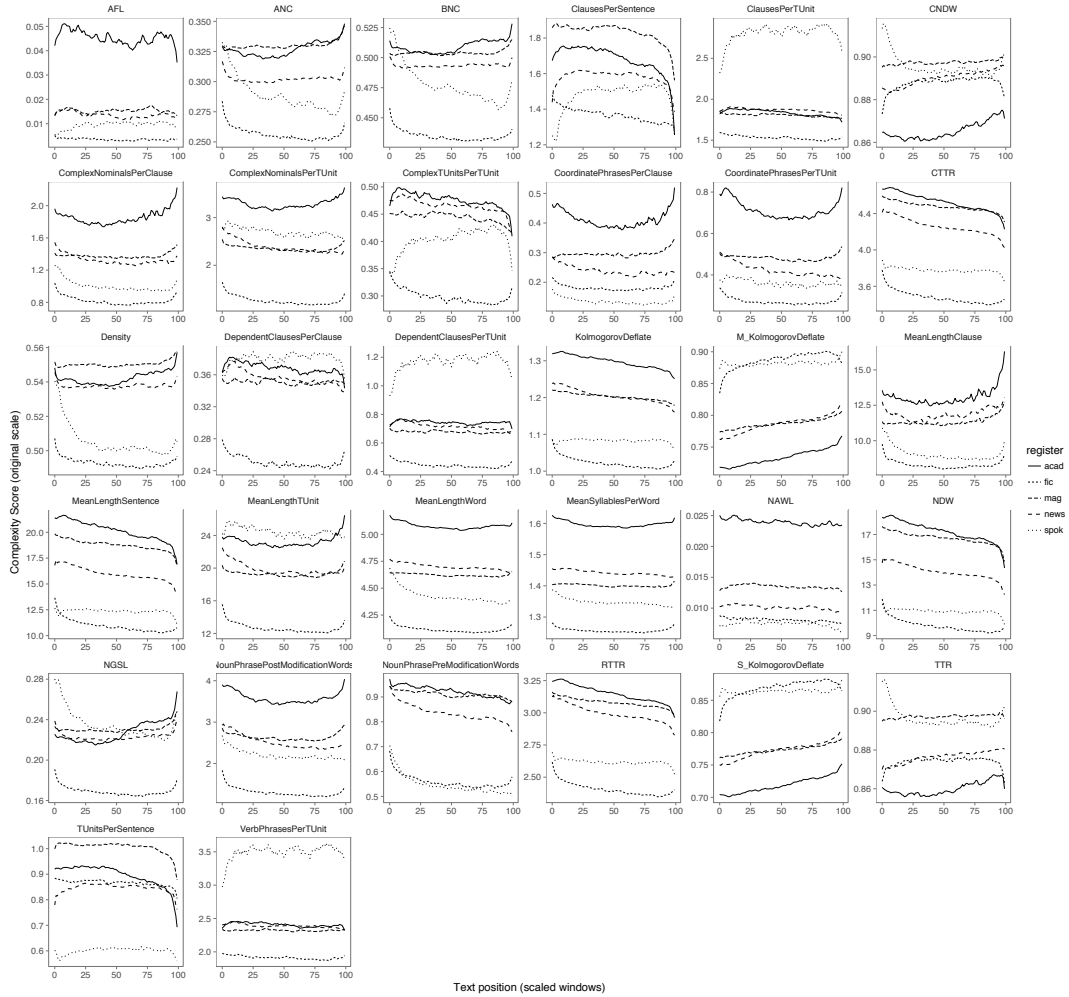



Figure: Average distribution of complexity across all texts from a given genre for all measures of complexity

# Classification Model

1. Recurrent Neural Network classifier (model specification adopted from Hafner, 2018)
   - 1.1 Dynamic RNN model that can handle sequences of variable length
   - 1.2 Gated Recurrent Unit (GRU) cells: better performance on smaller datasets (Chung et al., 2014)

# Classification Model

- Assume an input sequence $X = (x_1, x_2, \ldots, x_l, x_{l+1}, \ldots, x_n)$ , where each of $x_i$ is a 32 dimensional vector, $l$ is the length of the sequence, $n \in \mathbb{Z}$ is a number, which is greater or equal to the length of the longest sequence in the dataset and $x_{l+1}, \cdots, x_n$ are padded $\mathbf{0}$-vectors.
- GRU cells with 200 hidden units.
- To predict the classification, softmax was applied to the output of a fully-connected layer, where the output of the last GRU cell, i.e. whose input is $x_l$, are transformed from a 200 dimensional vector to a 5 dimensional vector.
- same model was used for comparison to the average-complexity approach
  - trained on vectors of average-complexities, i.e. the roll-out of the model consist of only one GRU cell.

# Results: Classification Performance

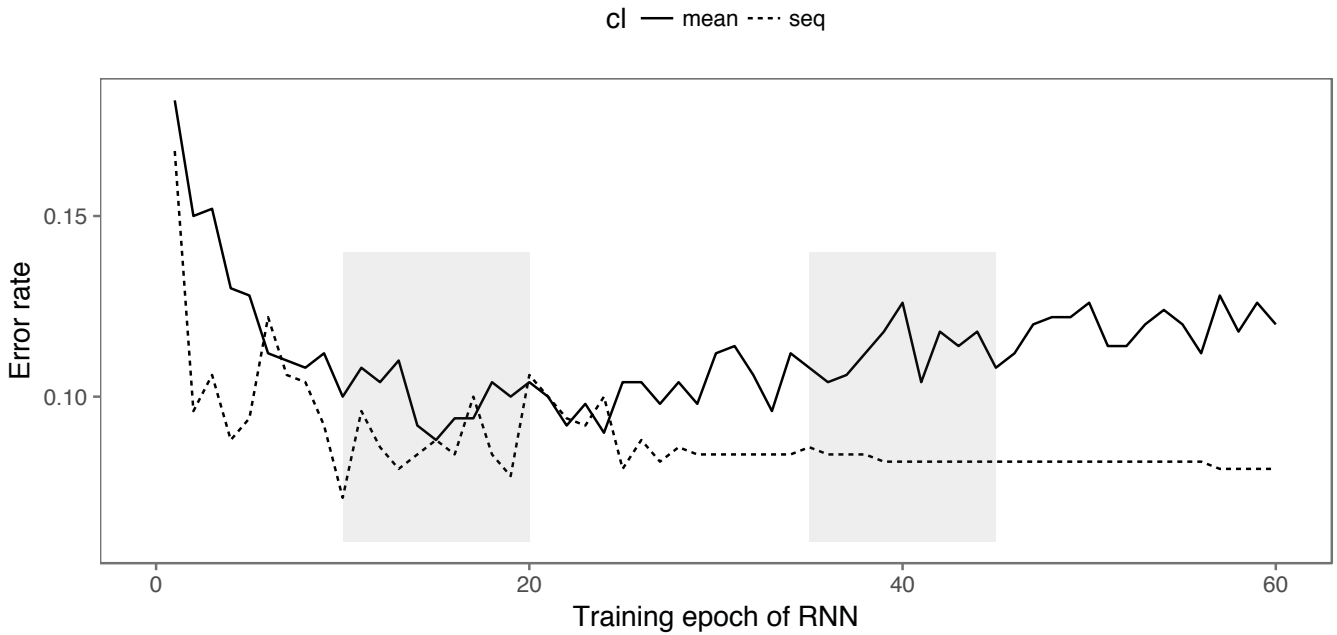



Figure: Performance of our RNN classifiers over 60 training epochs: sequence-based RNN displays consistently lower error rates than the average-based RNN: The average-based RNN reached a maximal performance of 91.2\% at epoch 16 with an mean performance of 90\% accuracy in the surrounding epochs (epochs 10-20).

# Example: Kerz, Ströbel, Wiechmann, in progress

- Investigate the development of writing at advanced levels of L2 learning over the time course of one semester (ten weeks) across a large number indices of linguistic complexity
  - **RQ1**: Are there significant changes in the degree of lexical sophistication of L2 writing over a short period of time, and if yes, ...
    1. can it be observed across different types of indices?
    2. does it follow the developmental stages of academic writing proposed by Biber, Grey & Poonpon (2011)?
  - **RQ2**: What is the shape of the developmental trajectory (growth curves)?
  - **RQ3**: To what extent are the developmental trajectories subject to individual differences?

# Data

- 152 L1 German advanced L2 learners of English
- University students recruited from the RWTH Aachen University
- Samples were elicited on a weekly basis (10 time points)
- All samples were analyzed with regard to syntactic complexity and lexical sophistication using
  - 32 measures implemented in Complexity Contour Generator (CoCoGen, Ströbel, 2014; Ströbel, Kerz, Wiechmann, & Neumann, 2016)

# Data: An example

```
Learning Journal 13:¬
I) Summary of the topics: ¬
    The lecture started with an introduction to two research areas, the first one is involved in
the comparison of two different languages. The outcome of this study is able to judge about the
difficulty of learning a language. The thesis is that the similar a language is the easier it is
to learn it. The second area is second language acquisition, what is concerned with the learning
process of native speaker or non-native speakers ¬
    The next step we took was to learn something about the different Quantitative research
methods, we discussed how to divide the research in a certain structure which is named IMRAD and
we discussed how you should deal with the results of a research. We continued with another kind
of research, that is named Corpus-Based Research, it is defined by the use of a corpora to
acquaint data and to use the data for your research hypothesis and to create a result with the
help of it. The results of those researches are mostly illustrated with the use of statistics,
one kind of statistics is descriptive statistics. As the name is telling you it is describing the
distribution of data on a individually created scale. An example for the use of descriptive
statistics is normal distribution. These kinds of distributions are easy to detect, because of
its distinctive appearance. The most data points are situated around the mean, which creates a
parable like shape of the distribution. Another kind of statistics is the so called inferential
statistics, that is used to compare two different populations that went through the same treating
and therefore should create the same results. While descriptive statistics only enables you to
illustrate your founding, inferential statistics gives you the possibility to use the results of
two different tests and bring them in correlation, to find out what differs between the two
populations. To acquire this data are several methods useful. ¬
```

Figure: Excerpt of an writing sample (final raw text version)

# Data: An example


Figure: Excerpt of the same writing sample (changelog)

# Complexity Measures

| Complexity Measure | Label | Definition |
|---|---|---|
| `Mean Length of Words` | $MLWc$ | $N_{Characters}/Word$ |
| `Lexical Density` | $LD$ | $N_{Lexical\ Words}/N_{Words}$ |
| `Number of Different Words` | $NDW$ | $N_{Type}$ |
| `Type-Token Ratio` | $TTR$ | $N_{Type}/N_{Words}$ |
| `Logarithmic Type-Token Ratio` | $logTTR$ | $log(N_{Type})/log(N_{Type})$ |
| `Corrected Type-Token Ratio` | $cTTR$ | $N_{Type}/\sqrt{2N_{Type}}$ |
| `Root Type-Token Ratio` | $rTTR$ | $N_{Type}/\sqrt{N_{Type}}$ |
| `Mean Length Clause` | $MLC$ | $N_{Words}/Clause$ |
| `Mean Length Sentence` | $MLS$ | $N_{Words}/Sentence$ |
| `Clauses per Sentence` | $ClS$ | $N_{Clauses}/Sentence$ |
| `Coordinate Phrases per Clause` | $CoordCl$ | $N_{Coord.Phrases}/Clause$ |
| `KolmogorovDeflate` | $Kol$ | $len(w_{compressed})/len(w_{original})$ |

Figure: Complexity Measures investigated in this study

# Change in syntactic complexity over time (12 random students)

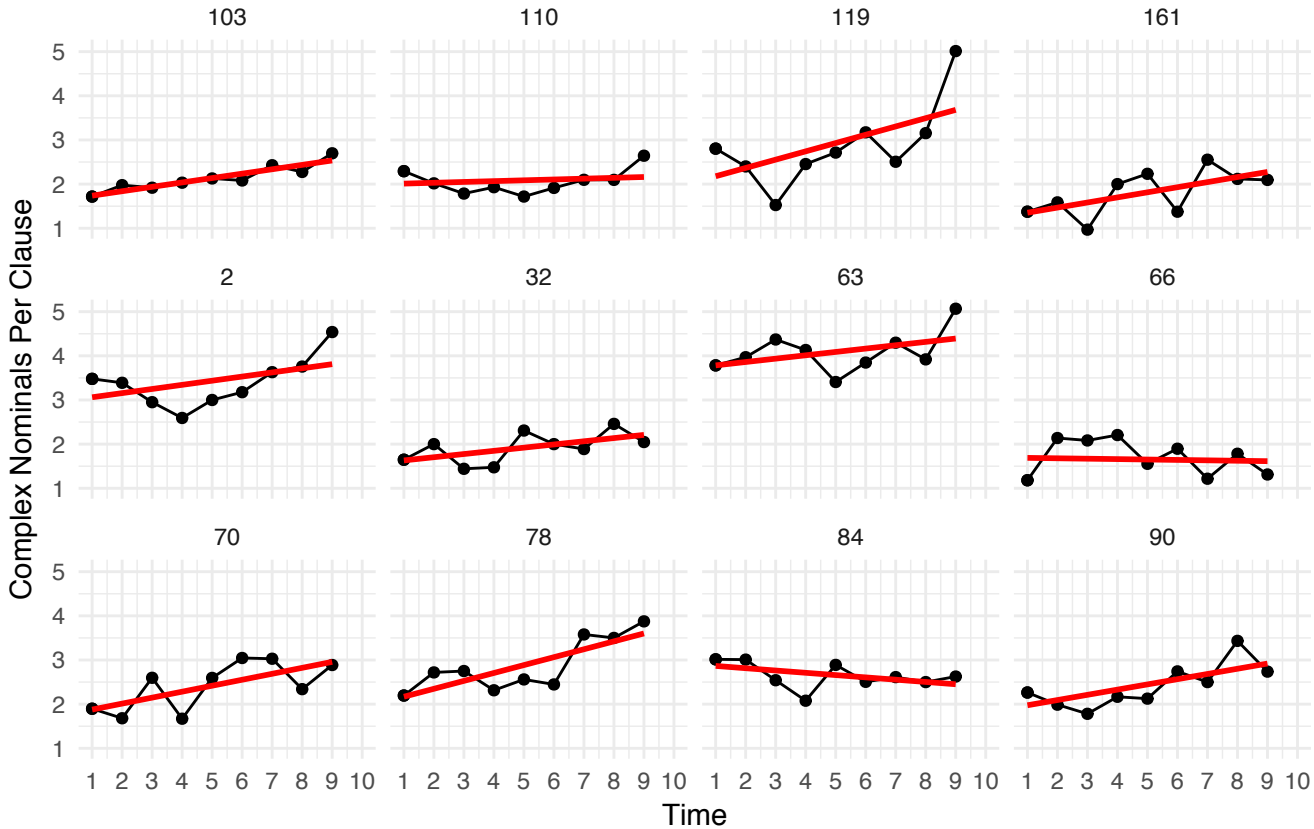



Figure: Development of lexical sophistication in L2 writing for twelve students.
**Index**: 'Complex Nominals Per T-Unit'

# Statistical Modeling

- Growth Curve Analysis: Developmental trajectories were analyzed using a combination of generalized linear mixed model (GLMM; cf. Mirman, 2014; using `lme4` 1.1-17) and generalized additive mixed models (GAMM; Baayen et al, 2017; using `mgcv` 1.8-24) in `R` 3.5.0
  - Separate models for each complexity measure (N = 32)
  - Treatment of TIME: GLMM
    - up to fourth-order orthogonal polynomial
    - participant random effects on all time terms
    - examine the change in the goodness of fit (log likelihood) through model comparisons
  - Treatment of TIME: GAMM:
    - TIME variable was entered as *thin plate regression spline*
    - random variability in nonlinear patterns was modeled using *factor smooths*
    - inclusion of autoregressive model for serially correlated errors in all models (Box et al., 2013)

# Development of syntactic complexity: Illustration: 'Complex Nominals per T-Unit'

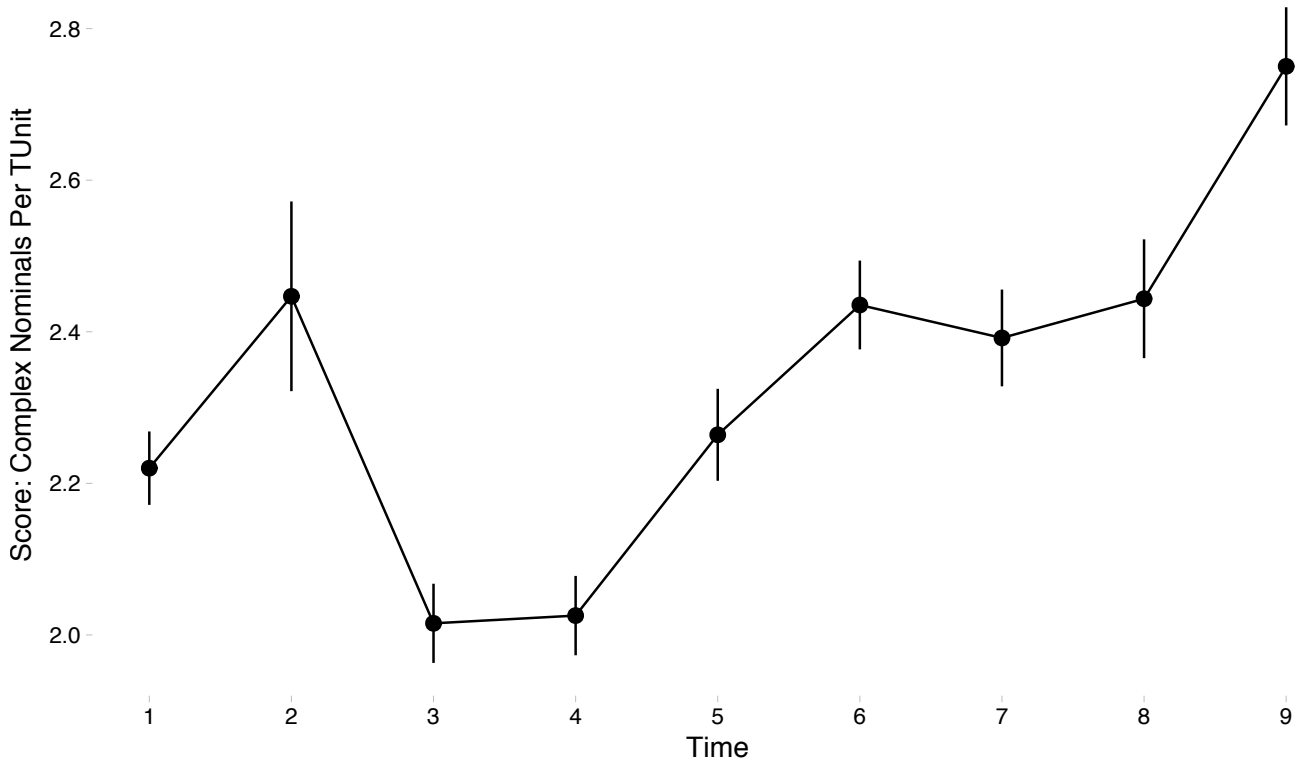



Figure: Mean syntactic sophistication (with standard errors) across 10 points of measurement (weekly assignments).

# Development of syntactic complexity: Illustration: 'Complex Nominals per T-Unit'

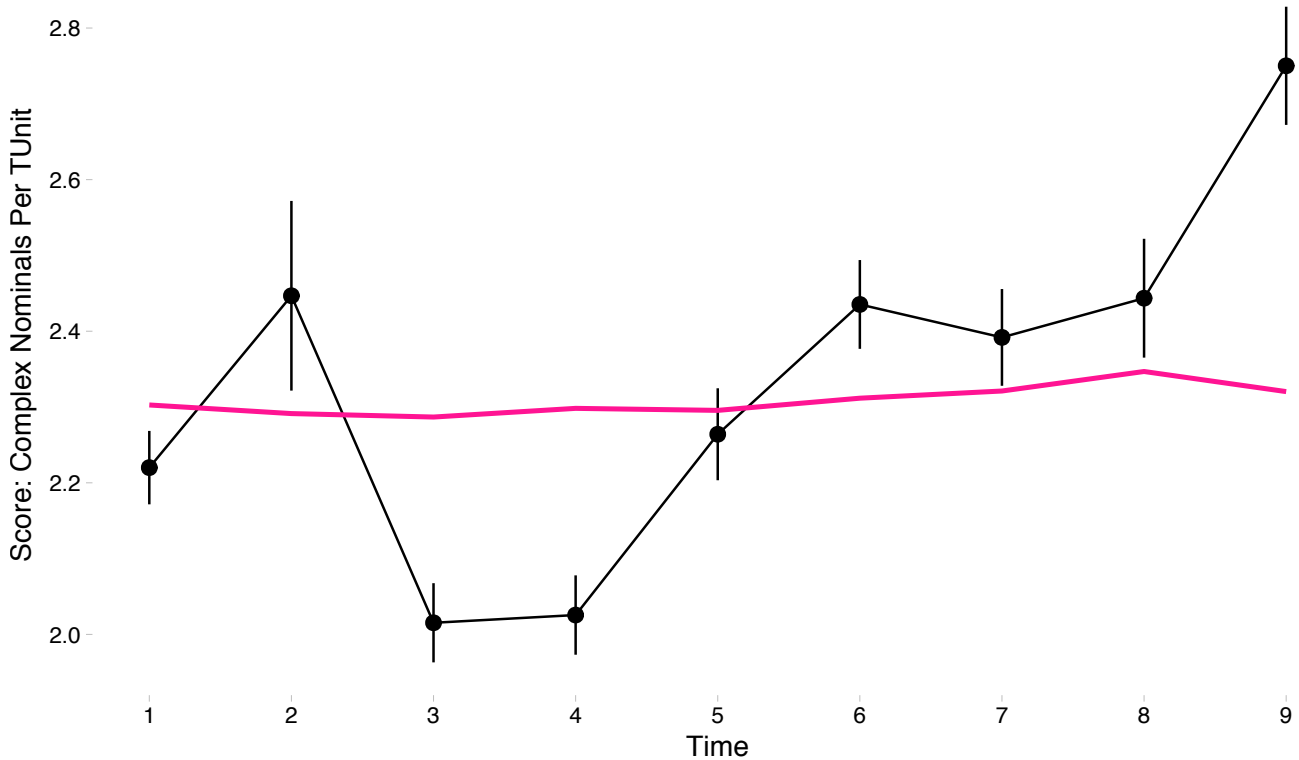



Figure: Mean syn. complexity (with standard errors) across 10 points of measurement (weekly assignments). UM (a random intercept for student)

# Development of syntactic complexity: Illustration: 'Complex Nominals per T-Unit'

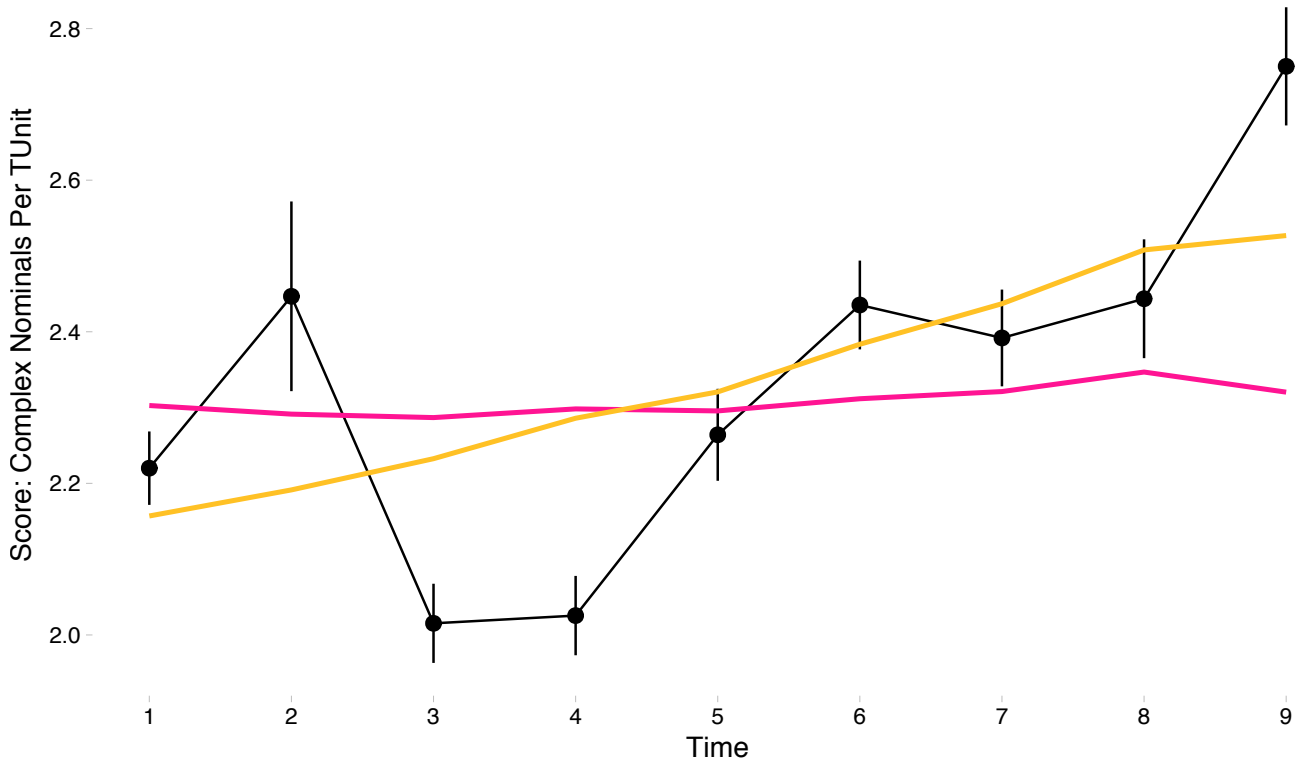



Figure: Mean syn. complexity (with standard errors) across 10 points of measurement (weekly assignments). **ADDED:** Linear term for Time

# Development of syntactic complexity: Illustration: 'Complex Nominals per T-Unit'

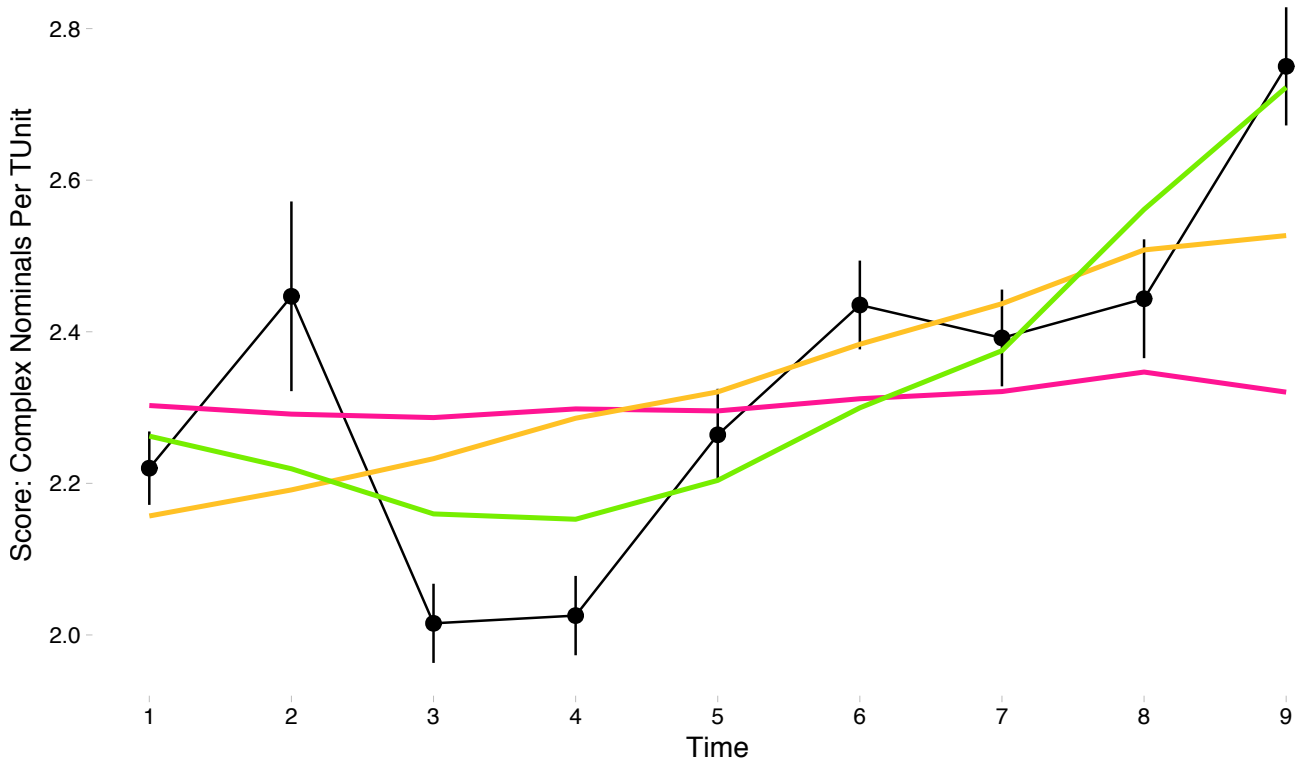



Figure: Mean syn. complexity (with standard errors) across 10 points of measurement (weekly assignments). **ADDED:** Quadratic term for Time

# Development of syntactic complexity: Illustration: 'Complex Nominals per T-Unit'

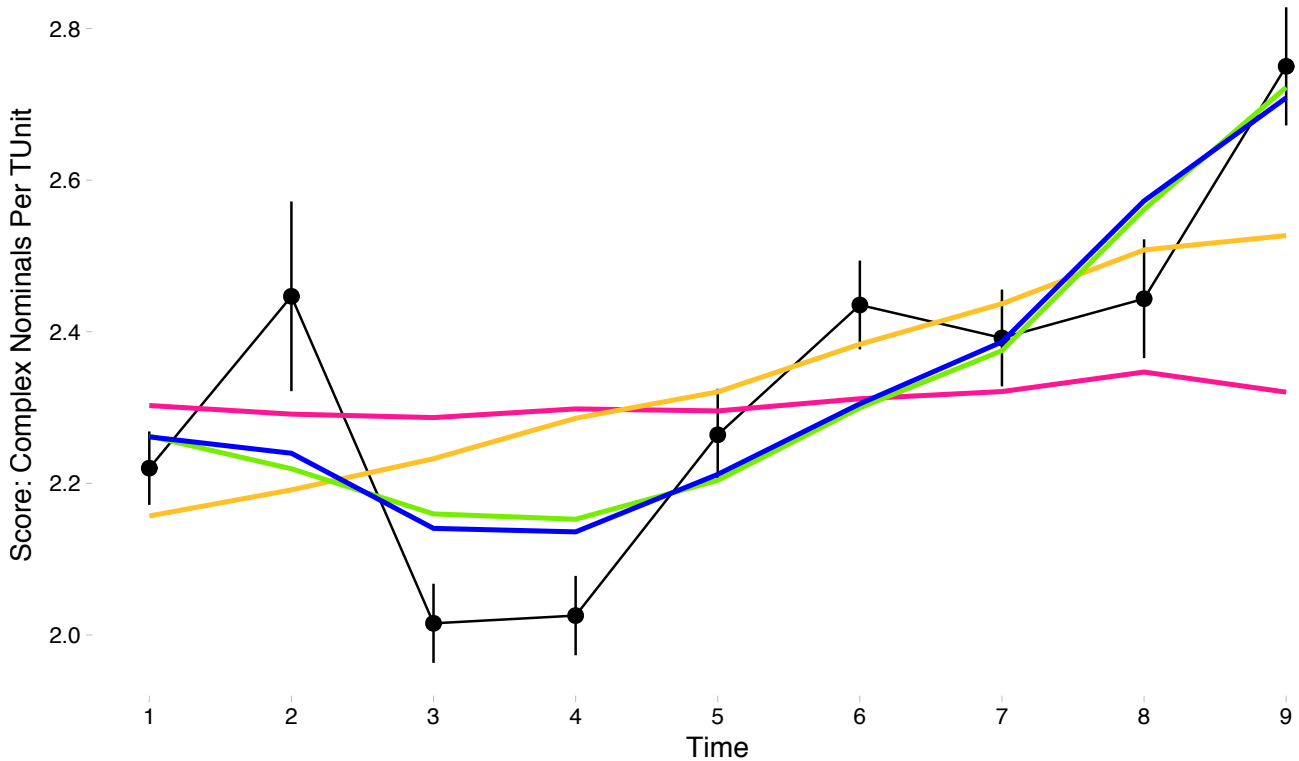



Figure: Mean syn. complexity (with standard errors) across 10 points of measurement (weekly assignments). **ADDED:** Cubic term for Time

# Development of syntactic complexity: Illustration: 'Complex Nominals per T-Unit'

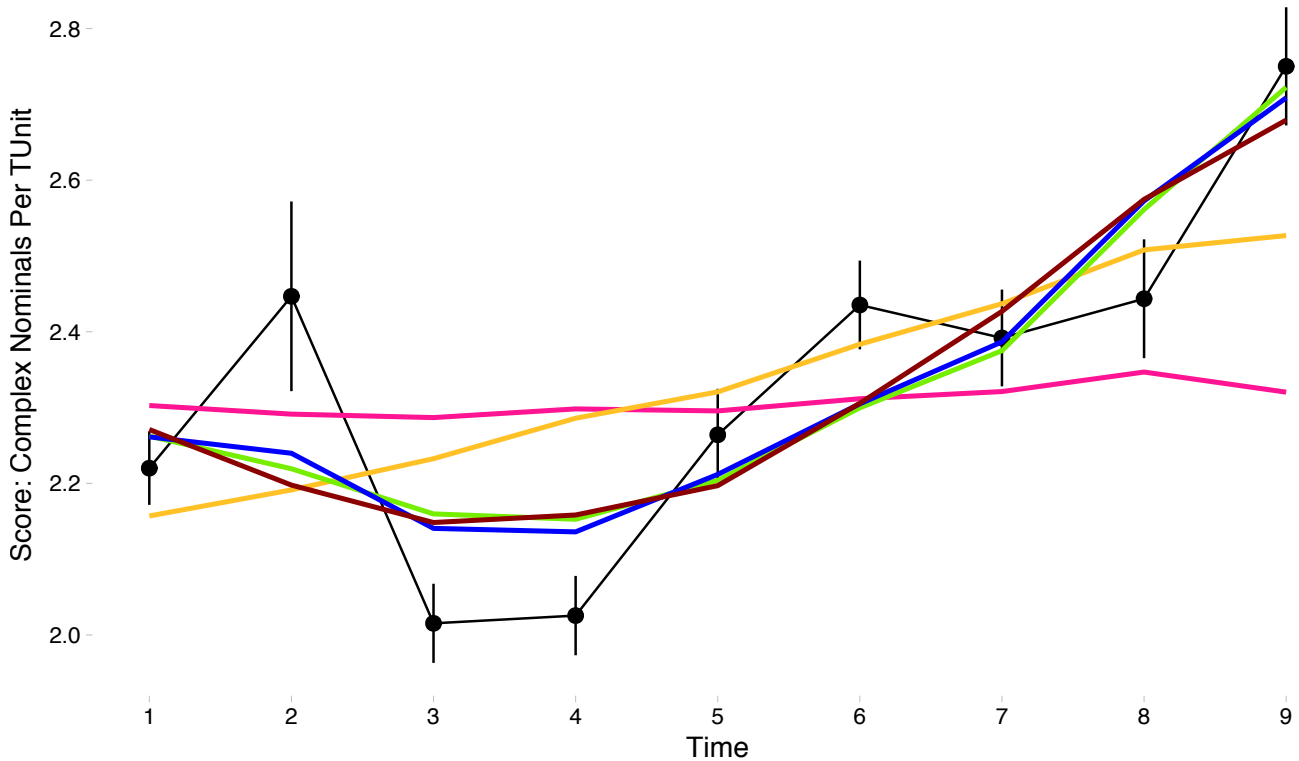



Figure: Mean syn. complexity (with standard errors) across 10 points of measurement (weekly assignments). **ADDED:** Quartic term for Time

# Development of syntactic complexity: Illustration: 'Complex Nominals per T-Unit'

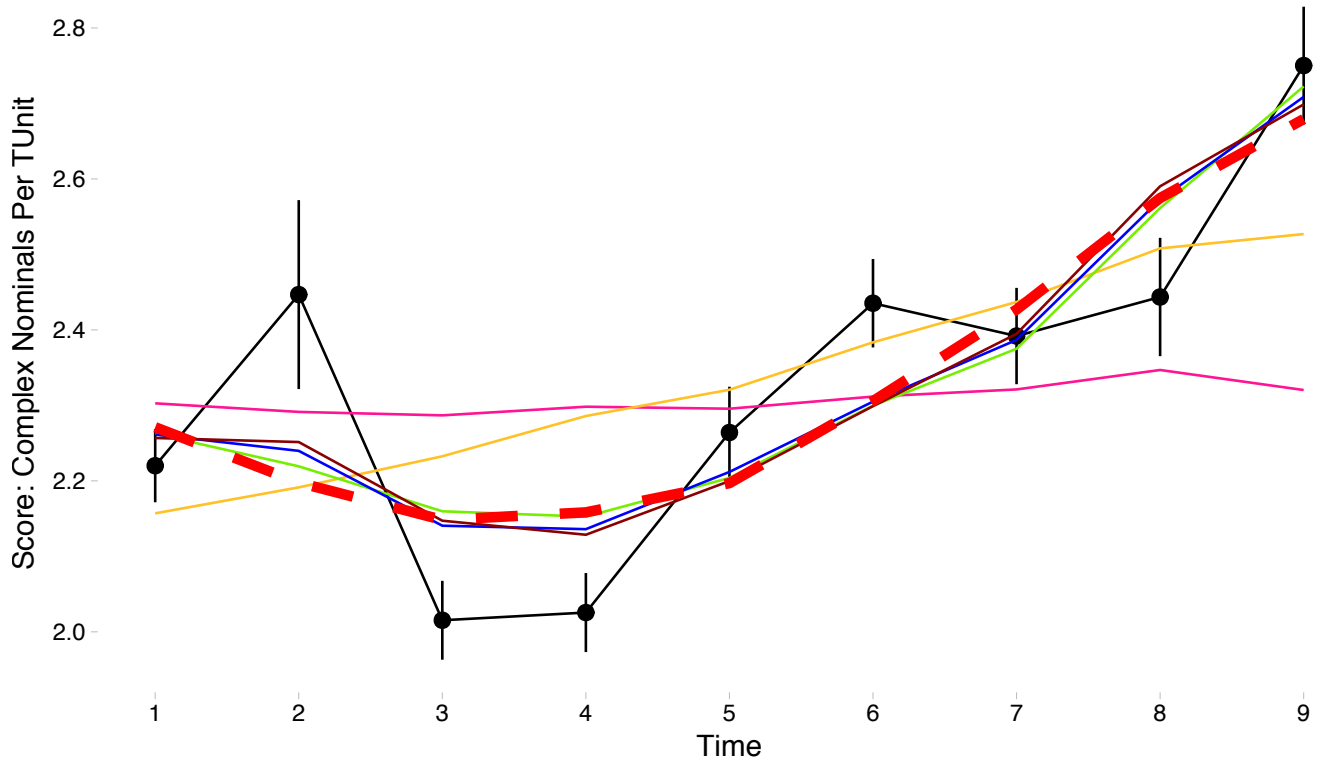



Figure: Mean syn. complexity (with standard errors) across 10 points of measurement (weekly assignments). **ADDED:** GAMM fitted (dashed line)

# Shape of development across complexity measures

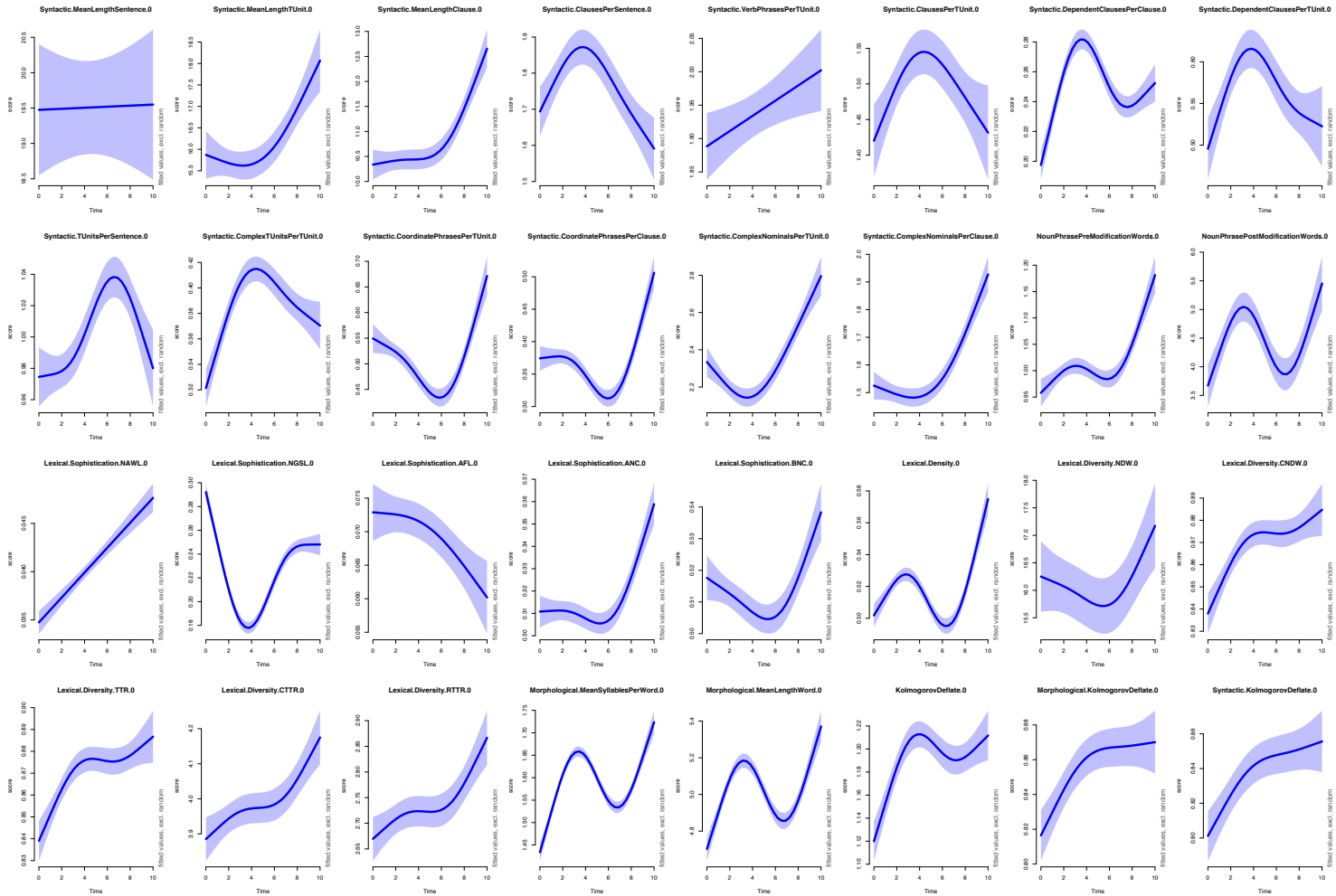



Figure: Results GAMM: Overview (32 complexity measures)

# Individual differences in developmental trajectory

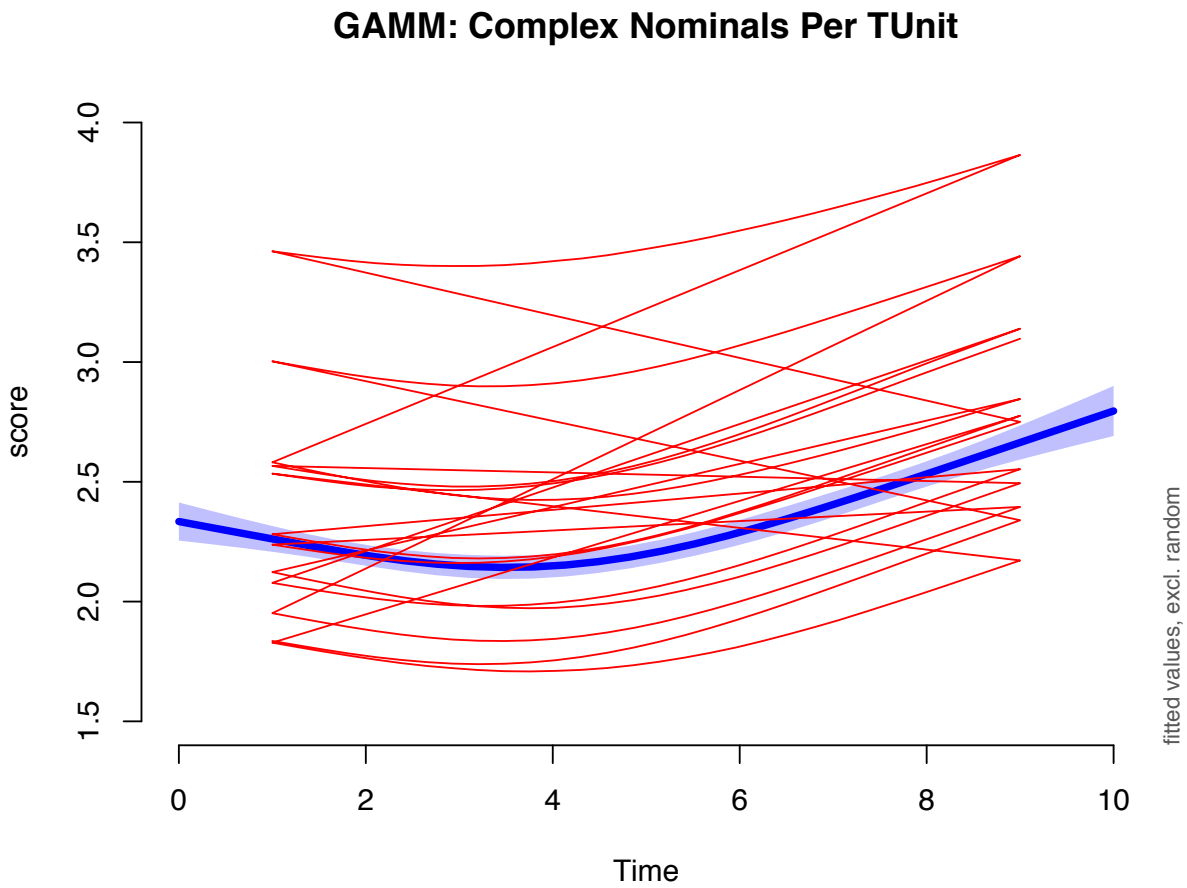



Figure: Group-level fitted values for Complex Nominals Per T-Unit plus estimates of shape of development for 12 random subjects.

# Development of register-adequate language use

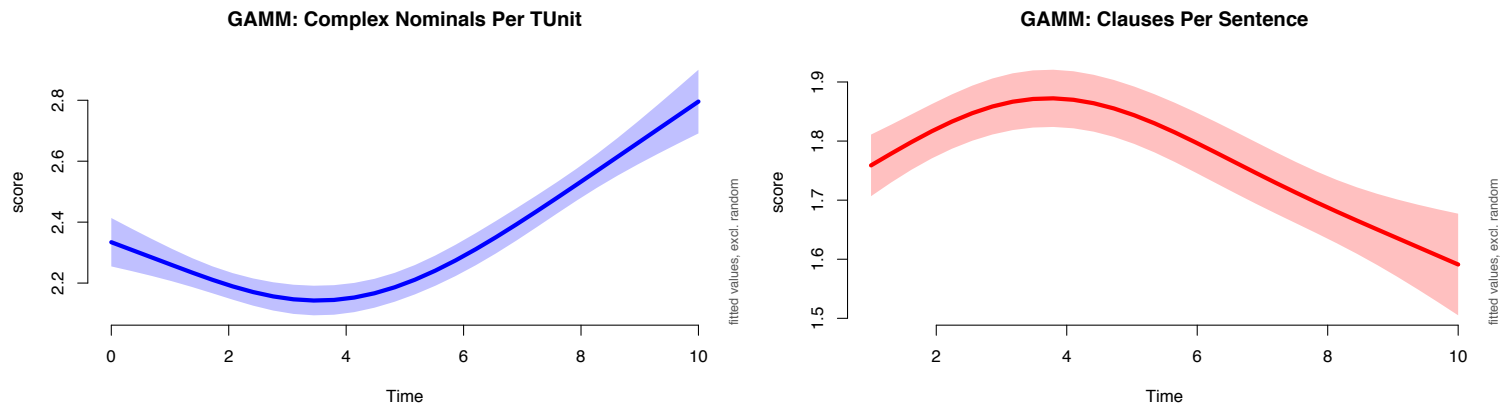



Figure: Group-level fitted values for 'Complex Nominals per T-Unit' (left) and 'Clauses per Sentence' (right).

# Goals

- **Task**: In its current version, the tool supports 32 complexity measures derived from language acquisition and processing research. The tool was designed with extensibility in mind, so that additional complexity measures can easily be integrated. Your task is to work on the further development of the tool including the integration of new complexity measures and the development of a platform independent desktop application.
- **Requirements**: Interest in natural language processing, good programming skills ideally in JavaScript or Python

# Goals

- **Task**: In its current version, the tool supports 32 complexity measures derived from language acquisition and processing research. The tool was designed with extensibility in mind, so that additional complexity measures can easily be integrated. Your task is to work on the further development of the tool including the integration of new complexity measures and the development of a platform independent desktop application.
- **Requirements**: Interest in natural language processing, good programming skills ideally in JavaScript or Python

- Development and implementation of new complexity measures that concern the use of multiword units (n-grams) and psycholinguistics norms
- Development of front-end
- New application domains (fake news detection, grade level identification, author recognition, ...)

# Keystroke Logging in Second Language Writing Research

Recent years have seen a growing interest in the use of keystroke logging for identifying writing strategies and underlying cognitive processes (see, e.g., Leijten and Van Waes, 2013 for an overview). Keystroke logging programs time stamp keystroke activity making it possible to reconstruct text production processes. The basic assumption is that the analyses of pauses (length, number, distribution, location, etc.) and revisions (number, type, operation, emeddeddness, location, etc.) are indicative of cognitive effort involved. One of the main challenges is to link the logging data to the underlying cognitive processes. The goal of the project is to analyze a combination of cross-sectional and longitudinal keystroke logging data collected at our university to gain insights into the writing processes and strategies of second language learners of English.

# Writing

- Writing – composing a text – is the result of several operations that have considerable motor and cognitive overlap in time
  - Gentner, 1988; Salthouse 1986, Hayes and Flower; 1980, 1986

# Writing

- Writing – composing a text – is the result of several operations that have considerable motor and cognitive overlap in time
  - Gentner, 1988; Salthouse 1986, Hayes and Flower; 1980, 1986
- High-level writing processes
  - **planning**: setting rhetorical goals, generating ideas, and organizing them into a writing plan.
  - **translating:** converting ideas into language, and transcribing them in written form (notice that Salthouse's model considers only this last transcription step).
  - **revising**: reading, evaluating, and editing the text produced, but can also operate upon a mental plan.

# Writing

- Writing – composing a text – is the result of several operations that have considerable motor and cognitive overlap in time
  - Gentner, 1988; Salthouse 1986, Hayes and Flower; 1980, 1986
- High-level writing processes
  - **planning**: setting rhetorical goals, generating ideas, and organizing them into a writing plan.
  - **translating:** converting ideas into language, and transcribing them in written form (notice that Salthouse's model considers only this last transcription step).
  - **revising**: reading, evaluating, and editing the text produced, but can also operate upon a mental plan.
- Low-level processing
  - holding a to-betranscribed chunk in working memory (WM)
  - parse chunk inro discrete characters
  - translate result into motor programs that specify the characteristics of the appropriate keystrokes
  - execution: perform ballistic typing movements

# Writing

- Thus when typing, a writer must keep active in WM the chunk being transcribed while parsing, programming, and motor execution take place.
- Skilled motor output frees WM capacity, which can then be allocated to higher level processes
  - Fayol, 1999; Kellogg, 1996; McCutchen, 1996

# Methods in writing research

| | Direct research methods | Indirect research methods |
|---|---|---|
| Synchronous | Concurrent think aloud protocols<br>Prompted pauses | Keystroke logging<br>Video observation<br>Double task method<br>Eyetracking<br>EVP or fMRI |
| Asynchronous | Retrospective protocols | Text analysis<br>Versioning |

Source: Based on Janssen, Van Waes, and Van den Bergh (1996).
EVP = evoked potential; fMRI = functional magnetic resonance imaging.

Figure: Elaboration of Classification of Writing Observation Methods (from: Leijten and Van Waes, 2013)

# Methods in writing research: keystroke logging

- Keystroke logging programs are designed to observe writing processes on a computer
  - for a review, see Latif, 2009; Van Waes, Leijten, Wengelin, et al., 2012
- These programs log and time stamp keystroke activity to reconstruct and describe text production processes
- The logfiles contain a wealth of information concerning operations on the text, and this information can be used to replay the writing event revision by revision

# Methods in writing research: keystroke logging

- Research applications for keystroke logging in writing include a wide range of areas:
  - writing strategies in professional writing or creative writing
  - the writing development of children—with and without writing difficulties
  - spelling
  - writing of expert and novice writers in professional contexts and in specialist skill areas such as translation and subtitling
  - first and second language writing
    - insights into the writer's cognitive activity / **cognitive writing processes**

# Keystroke logging and cognive effort

- The main rationale behind keystroke logging:
  - writing fluency and flow reveal traces of the underlying cognitive processes
- This explains the analytical focus on pause (length, number, distribution, location, etc.) and revision (number, type, operation, embeddedness, location, etc.) characteristics

# Keystroke logging and cognive effort

- As in speech, **pause times are seen as indexical of cognitive effort** / pauses are indicators of cognitive processing
  - Foulin 1995; Schilperoord 1996; Cenoz 2000
- Increased cognitive load correspond to increased **pause activity; high cognitive load appears to be associated with both long pauses and clusters of short pauses**
  - e.g., Krings 2001; O'Brien 2004; O'Brien 2005; O'Brien 2006; Dragsted and Hansen 2009; Carl et al. 2011; Lacruz et al., 2012
- About 50% of pauses (threshold of 3 seconds) are followed by revisions
  - Van Waes and Schellens, 2003

# Keystroke logging: Measures (selection)

- Fluency: the number of words (or characters) written per minute
- Burst: number of words (or characters) written between pauses or revisions
- Total number of revisions: sum deletions and/or insertions
- Total pause time: sum pauses
- Average pause length
- Pause time: percent pauses of total writing time
- Pause ratio: total time in pause divided by total time in segment
- Average pause ratio: average time per pause in a segment divided by average time per word in the segment
- Total post-editing time
- Pause ratio in post-editing: pause time over total task time

# Goals

- **Task**: Your task will involve the preprocessing of keystroke logging data and subsequent data analysis. For each individual text, you will get lists of Etherpad changeset and the corresponding timestamps (see, https://github.com/ether/etherpad-lite/wiki/Changeset-Library).
- **Requirements**: Ideally interest in natural language processing and experience working with language data

# Goals

- **Task**: Your task will involve the preprocessing of keystroke logging data and subsequent data analysis. For each individual text, you will get lists of Etherpad changeset and the corresponding timestamps (see, https://github.com/ether/etherpad-lite/wiki/Changeset-Library).
- **Requirements**: Ideally interest in natural language processing and experience working with language data

- Development of data processing pipeline
- Development of new complexity measures to be used is CoCoGen
  - interface with CoCoGen

# MISCELLANEOUS

- ....

# Factors determining the length and position of pauses

- Position of pauses is in part determined by rhetorical features (Foulin, 1998; Schilperoord, 1996)
- The lengths of these pauses vary with features of the text that is being composed; pause length increases with text unit level (cf. Spelman Miller, 2000; Wengelin, 2006)
  - In general, pauses between letters within a word are shorter than those preceding a word
    - pauses between sentences are shorter than those between paragraphs
    - grammatical, discourse, and morphological boundaries affect pause length (Nottbusch, Grimm, Weingarten, & Will, 2005; Spelman Miller, 2006).
  - Pausological research of written language production has focused on quantifying the frequencies of pauses at various linguistically relevant locations. Such locations have been primarily defined in terms of surface-level and grammatical features, such as paragraphs, sentences, T-units, clauses, and phrases

# Functions of pauses

- Pauses can signal the pursuit of high-level writing processes that for some reason could not be carried out at the same time as typing.
- The function of pauses is poorly specified in writing research (see Torrance & Galbraith, 2006) but research in other domains of cognitive psychology provides plausible interpretations
  - pauses are due to the competition for limited capacity (Just & Carpenter, 1992)
  - typing and high-level processes compete for a common processing component (Pashler, 1994)
  - pauses result from cross-talk between products and processing of ongoing activities (Navon & Miller, 1987)
  - pauses arise as consequence of memory decay, and are used to reinstate the intended message (Torrance & Galbraith, 2006)

# Pipeline

1. Stanford CoreNLP (Manning et al., 2014)
   - tokenization, sentence splitting, tagging, lemmatization, parsing
   - caching of results implemented
2. Calculate measures for each sentence
3. Apply sliding windows and scaling
4. Write output to .csv files
   - windows: one line per window
     corpus, text, filename, window, measureA, measureB, ...
   - scaled output: one line per text
     corpus, text, filename, measureA$_0$, ..., measureA$_{swc-1}$, ...