Formation CorLi, Paris, 16 mai 2023
Atelier TXM avancé : potentiel de fonctionnalités non statistiques
Déroulé

Avertissements :

- *Prérequis de la formation : la formation TXM avancé ne s'adresse qu'à des participants ayant déjà une pratique de TXM sur leurs corpus.*
- *Ce document est basé sur les notes de la formatrice, comme trame pour la séance. Il n'a pas été conçu pour être utilisé sans avoir assisté à la séance de formation.*

Version de référence = dernière version stable de TXM = 0.8.2

Généralités

- Adresse de référence du **site** Textometrie : https://textometrie.org
- Le **manuel** utilisateur 0.8 en ligne, plus récent que le PDF : https://txm.gitpages.huma-num.fr/txm-manual/
- Comment éliminer les **zones vides** « fantômes » (en glissant-déposant un résultat dans une zone vide).

Concordance, et CQL

Régler les **références**

- Avec le menu contextuel (clic-droit) sur la colonne des références. Considérer non seulement l'affichage des références, mais aussi leur tri (plus clair si les deux sont analogues). Exemple sur VOEUX, `paix`, affichage des propriétés `text_loc` et `text_annee`
- Trucs&Astuces : garder un réglage en utilisant la commande « Conserver »
- Réglages à l'import XTZ : remarquer le nouveau paramètre pour composer les références ; plus de possibilités encore avec `$TXMHOME/xsl/txm-posttok-addRef.xsl`

La simplification des **requêtes en plusieurs mots**, ex. :

```
pour l'avenir
?
```

Remarque : ces facilités sont limitées aux graphies (pour interroger sur les lemmes, il faut utiliser le langage CQL ou recourir à l'assistant de requête, la « baguette magique » à gauche du champ requête)

Se construire un **Sommaire** de son corpus (une liste des textes) avec en concordance la requête `<text>[]`

La concordance est ici utilisée surtout pour la colonne des références (désignation de chaque texte) et pour l'accès direct à n'importe quel texte en double-cliquant sur la ligne de concordance correspondante (a priori on ne s'intéresse pas vraiment au contenu de la concordance elle-même, pivot et contextes).

Vu qu'ici on s'intéresse peu au contenu de la concordance elle-même sinon à sa première colonne, on peut placer le retour au texte à droite plutôt qu'au-dessus, en glissant déposant l'onglet.

Si d'une façon générale on préfère avoir l'affichage de l'édition à droite de la concordance plutôt qu'au-dessus, on peut régler cela comme nouveau comportement par défaut via une préférence (Menu principal Edition > Préférences > TXM > Utilisateur > Édition (tout en bas) : New editor position → remplacer ABOVE par RIGHT_OF).

Contrainte sur une structure (métadonnée)
```
[_.text_loc="chirac" & frlemma="crise"]
```

Passage au **sous-corpus** correspondant (**expand to**)
Trucs&Astuces : pour un corpus comme pour un sous-corpus, on peut consulter et vérifier la liste de toutes les valeurs de `loc` et `annee` présentes dans l'interface de **partition** assistée.
```
[frlemma="crise"] expand to text
<text>[_.text_annee="196."] expand to text
```

- cet exemple un peu artificiel est inspiré d'un cas réel, dans un corpus de presse avec de très nombreuses dates au jour près, où l'on a pu ainsi construire une partition sur les mois pour mettre en évidence les « marronniers » au sens journalistique. Mais ici dans le cas des décennies dans VOEUX, l'interface de sous-corpus simple suffisait (avec réglage de l'affichage sur `text/annee` puis une sélection multiple).
- Noter la différence avec `[_.text_annee="196."]` (sous-corpus émietté → aucune étude possible sur les suites de mots, l'ordre des mots est perdu)

Pour s'exercer :
sauriez-vous chercher toutes les dernières phrases des textes de VOEUX ?
```
[]</text> expand to s
```
et si l'on ne voulait que celles qui ne se terminent pas par « vive la France ! » ?
```
[frlemma!="France"]{2,2}</text> expand to s
```
(ce n'est qu'une solution parmi d'autres – on s'est contenté de ne pas avoir « France » sur les deux dernières positions, sachant que la ponctuation compte, et cela a suffi)

Index, et CQL

Élastique **within**
L'exemple de la recherche de « je ... souhaiter … année » dans la même phrase, en utilisant le formulaire de requête

qui génère
```
[frlemma = "je"] []{0,50} [frlemma = "souhaiter"] []{0,50} [frlemma = "année"] within s
```

Remarque : Le choix de `[]{0,50}` plutôt que `[]*` pour les élastiques : pour faire une requête pas trop « dangereuse » par défaut, car le `[]*` peut traverser des milliers de mots, et c'est un besoin a priori rare.

Recherche distributionnelle et opérateur @
Pour illustrer, on se propose de lister et compter les verbes qui sont utilisés dans une construction négative.
On commence par mettre au point sa requête en vérifiant qu'on trouve bien les contextes qu'on vise, qu'on les décrit bien avec la requête CQL :

```
[frlemma="ne"][frpos="PRO:PER|ADV|VER.*"]{0,4}[frpos="VER.*"]
[frpos!="VER.*"]
```

Puis une fois la requête trouvant les contextes qui nous intéressent, on peut se focaliser sur le verbe seul (à l'infinitif → régler la propriété d'affichage sur frlemma) en le « ciblant » avec l'opérateur @ placé devant le mot voulu (un seul mot possible par requête) :

```
[frlemma="ne"][frpos="PRO:PER|ADV|VER.*"]{0,4}@[frpos="VER.*"]
[frpos!="VER.*"]
```

Nouveau comportement du **lien INDEX -> CONCORDANCE**
Quand on double-clique sur une ligne d'INDEX, on a une CONCORDANCE pour visualiser en contexte les occurrences correspondantes.
Avant TXM 0.8.2, il fallait se méfier que dans certains cas, certaines contraintes de sélection qui avaient été actives pour l'INDEX sont perdues pour la CONCORDANCE (quand cela arrive, c'est simple à repérer : il y a plus de lignes de concordance que la fréquence annoncée dans l'index). Le nouveau comportement règle ce problème : les lignes de CONCORDANCE correspondent bien aux occurrences dénombrées dans l'INDEX, comme attendu.
Cela se fait via un sous-corpus caché lié à la requête de l'index. L'effet de bord, c'est que si on change la requête d'INDEX, alors les concordances précédemment calculées sous l'index ne « fonctionnent » plus (elles continuent d'être filtrées sur l'ancienne requête). Donc quand on réutilise un INDEX, c'est plus clair de supprimer toutes les concordances qui ont été créées dessous.

Vers les segments répétés (Salem) **SR** (avec `{}`, négation `!=` et `SetMatchingStrategy`)
Progressivement :

```
[][][][][]
[]{5,5}
[word!="\p{P}"]{5,5}
[word!="\p{P}"]{3,8}
```

en stratégie `longest`
Passer le moteur de recherche en stratégie `longest` (i.e., il cherche la réalisation la plus longue possible de la requête) se fait en lançant la macro `cqp/SetMatchingStrategy` qui est livrée avec TXM (il suffit d'aller la chercher au bon endroit...)
cf. Manuel :
https://txm.gitpages.huma-num.fr/txm-manual/piloter-la-plateforme-par-scripts.html#piloter-par-macros-groovy
voir en particulier §8.1.1 Exécuter une macro
Attention : dès qu'on n'a plus le besoin particulier de la stratégie `longest`, **revenir en stratégie `standard`** – car si on oublie qu'on est en stratégie `longest`, un certain nombre de requêtes peuvent devenir « dangereuses » et lancer des recherches excessives.
Ce retour à la stratégie standard s'opère de toutes façons à chaque redémarrage de TXM. (Quand on lance TXM, on est par défaut en stratégie standard, quelque soit l'état dans lequel on l'a quitté.)

Progression

Exemple sur VOEUX, pour les requêtes `[frlemma="crise"]` et `[frlemma="emploi"]`

- Rappel de lecture : non pas la position respective des courbes mais leurs pentes
- Cumulatif vs densité : observer le passage Mitterrand-Chirac pour : `[frlemma="crise"]`
- Affichage des repères, et filtrage

exemple sur `text_annee`, puis exp rég. `...0` pour les décennies

Est-ce la meilleure façon d'approcher la chronologie ? Tout dépend de la façon dont on peut/veut considérer le corpus, avons-nous plutôt affaire à une évolution continue au fil des mots (avec pas vraiment de « saut » entre les textes notamment), ou plutôt à différentes périodes qu'il s'agit de caractériser ?

- Évolution continue → Progression : continuité et détail interne
- Différentes périodes → S+ : tranches (globales, séparées)

Donc souvent il y a une approche plus adaptée selon le type d'évolution auquel on a affaire.

En Progression : attention à l'ordre du corpus

- en général donné par l'ordre alphabétique des fichiers sources des textes
- possibilité de le régler dans le fichier `metadata` avec une colonne `textorder`, au moins pour l'import XML-TEI Zero (mais à tenter pour d'autres imports aussi ?), cf. manuel https://txm.gitpages.huma-num.fr/txm-manual/importer-un-corpus-dans-txm.html#module-xml-tei-zerocsv-dit-aussi-xtzcsv-ou-xtz-import-de-xml-tei-g%C3%A9n%C3%A9rique-.xml
voir section §4.9.5.5 Ordre des textes

Pour les Spécificités :

- Trucs&Astuces : La construction de la partition en mode assisté (même si le mode simple aurait suffi) permet de choisir l'ordre (et la désignation) des parties.
- Permet de traiter des sous-corpus discontinus (refusés par la Progression).
- Donne une évaluation statistique :
  - attention à ne pas lire un fort emploi mais un sur-emploi, ou un non-emploi au lieu d'un sous-emploi.
  - Influence aussi de la taille des tranches sur la comparaison.

Annotation par requête

Cette partie n'a pas été vue en séance, mais je laisse les indications sur le contenu prévu au cas où cela puisse intéresser certains de poursuivre en mode auto-formation (sachant cependant que les indications restent succinctes, le support pour cette partie n'étant pas plus détaillé que pour celles vues en séance).

Rappel : 3 types d'annotation dans TXM (diapo 39 de ma présentation à la journée Club Corpus en Sorbonne, juin 2022, https://shs.hal.science/halshs-03763765) :

Type	*Projets*	*Interface*	*Nature technique*	*Intérêts*	*Limites*
CQP sur Mot	PaLaFra, BFM	Concordance	Propriétés lexicales	Simple à comprendre, peut répondre à beaucoup de besoins	Unités définies par la tokenisation
CQP sur Séquence de mots	BHE / SyMoGIH	Concordance	Structures et propriété	Délimitation de l'unité	Une seule annotation par structure (ref) ; complexité de la gestion des chevauchements
URS (Unité-Relation-Schéma)	Democrat, DTH	Edition, puis Concordance	Annotation déportée	Délimitation de l'unité, pas de contraintes structurelles, et modèle d'annotation structuré	Pas d'exploitation directe en CQP donc macros dédiées (ou projection vers CQP)

Exemple de cas d'application : recodage des POS en vue de calculer des spécificités en distinguant bien les lemmes par leur catégorie (ex. « pouvoir » en tant que verbe ≠ en tant que nom). (Cet exemple correspond à un cas réel, rencontré sur un corpus en anglais, où les homographies nom/verbe sont fréquentes.) La difficulté sans recodage, c'est que le jeu d'étiquettes répartit les verbes dans plusieurs catégories (au moins selon le temps verbal pour les modèles TreeTagger les plus courants en français et en anglais), on ne les a pas en bloc.

Ressources pour l'expérimentation : `https://bit.ly/3NR4FUt`
Celle-ci fournit deux fichiers :

- `VOEUX-TEST-230512.txm` : un clône du corpus VOEUX, avec un nom différent, qui nous servira de « bac à sable » (ainsi, vous gardez en l'état le corpus VOEUX que vous avez déjà dans TXM). En effet, l'annotation va modifier le corpus.
- `cql_frpos33a20.csv` : un fichier tout prêt à passer en paramètre de la macro. Il donne un modèle pour construire d'autres fichiers analogues pour d'autres annotations par requêtes : on repart du fichier lui-même (pour avoir le bon format) en l'enregistrant sous un nouveau nom et en remplaçant le contenu des cellules par ses propres catégories et requêtes.

Remarque :
Le lien pour les ressources du cours n'est pas pérenne. Ce qui est surtout utile est d'avoir un exemple de fichier paramètre pour la macro. On peut en trouver un autre ici (lien pérenne) : référence HAL https://shs.hal.science/halshs-03667319, fichier annexe `pincemin_al_jadt22_TXM_CQLList2WordProperties_parameter_file.zip`

Charger le corpus VOEUX-TEST-230512

Sélectionner le corpus VOEUX-TEST-230512 et lancer la macro `CQLList2WordProperties`, avec les paramètres :

- queries_file : ...`cql_frpos33a20.tsv`
- word_property : `frposgrp`
- update_corpus_indexes_and_editions : <coché>

Le temps de traitement peut être long (ici VOEUX-TEST-230512 = petit corpus)

Exploitation pour un calcul de spécificités organisé par POS :

- Partition simple sur `text_loc`
- INDEX sur [] en `frlemma_frposgrp`
- TABLE LEXICALE seulement sur la sélection
- SPECIFICITES

Si on veut aller plus loin :
= pour obtenir directement les lemmes les plus spécifiques de chaque partie, pour chaque catégorie grammaticale séparément

exporter le tableau de spécificités (fichier .csv)

ouvrir dans l'éditeur de TXM (menu Fichier > Éditer) pour faire quelques modifications dans le fichier .csv : on voudrait « éclater » la première colonne, frlemma_frposgrp, en deux colonnes, l'une pour le lemme et l'autre pour la POS. Pour cela :

1) ajustement de la 1ère ligne :
remplacement de Unité par frlemma tab frposgrp
chercher _ remplacer par -

2) à partir de la 2e ligne, remplacement du 1er souligné de la ligne (=séparateur) par une tab avec expression régulière cochée :
rechercher : `(^[^\n_]*)_`
remplacer par : `\1\t`

→ enregistrer dans
`..._col1split.csv`

Ouvrir dans LibreOffice, calc, typer « texte » les mots, et « Anglais US » les nombres (ce type Anglais US ne s'explique pas vraiment mais c'est un Trucs&Astuces général pour que les exports se comportent bien !)
enregistrer en `.ods`

Ensuite, on a tout ce qu'il faut dans le tableur, il n'y a plus qu'à filtrer et trier pour récupérer les résultats :

Filtrer sur la ou les valeurs frposgrp qu'on étudie, par ex. :
- Noms communs : NOM
- Adjectifs qualificatifs : ADJ
- Verbes hors auxiliaires : VER
- Modaux : MOD
- Pronoms personnels, possessifs, déictiques : PRO:PER, POS, DEM, certains ADV (ici,...)
- Adverbes, Interjections : ADV|INT

Trier par score-XXX décroissant et vérifier qu'à la 10e ligne on est au-dessus du seuil, sinon sélectionner la dernière ligne à considérer.