quanteda Aide-Mémoire

Quantitative Analysis of Textual Data

Syntaxe générale

- **corpus_*** collections/metadonnées de textes
- **tokens_*** créer/modifier textes tokenisés
- **dfm_*** gérer matrices doc/caractéristiques
- **fcm_*** manipuler les matrices de co-occurrence
- **textstat_*** Calculer des statistiques de textes
- **textmodel_*** ajuster modèles (non)supervisés
- **textplot_*** créer des visualisations de textes

Grammaire cohérente:
- **object()** constructeur pour le type d'objet
- **object_verb()** prends & retourne le type d'objet

Extensions

quanteda marche bien avec les packages compagnons suivants:
- **readtext**: une facon simple de lire les données textuelles
- **spacyr**: NLP en utilisant spaCy
- **quanteda.corpora**: corpus additionnel
- **stopwords**: Listes de mots vides multilingues en R

Créer un corpus à partir de textes (corpus_*)

Lire des fichiers textuelles (txt, pdf, csv, doc, docx, json, xml)
```
my_texts <- readtext::readtext("~/link/to/path/*")
```

Construit un corpus à partir d'un vecteur de chaîne de caractères
```
x <- corpus(data_char_ukimmig2010, text_field = "text")
```

Explorer un corpus
```
summary(data_corpus_inaugural, n = 2)
# Corpus consisting of 58 documents, showing 2 documents:
#            Text Types Tokens Sentences Year  President FirstName
# 1789-Washington   625   1538        23 1789 Washington    George
# 1793-Washington    96    147         4 1793 Washington    George
#
# Source:  Gerhard Peters and John T. Woolley. The American Presidency Project.
# Created: Tue Jun 13 14:51:47 2017
# Notes:   http://www.presidency.ucsb.edu/inaugurals.php
```

Extraire ou ajouter des variables au niveau document
```
party <- docvars(data_corpus_inaugural, "Party")
docvars(x, "serial_number") <- 1:ndoc(x)
```

Joindre ou créer des sous-ensembles de corpora
```
corpus(x[1:5]) + corpus(x[7:9])
corpus_subset(x, Year > 1990)
```

Changer l'unité d'un corpus
```
corpus_reshape(x, to = c("sentences", "paragraphs"))
```

Segmenter des textes selon des motifs
```
corpus_segment(x, pattern, valuetype, extract_pattern = TRUE)
```

Tirer un échantillon aléatoire de textes du corpus
```
corpus_sample(x, size = 10, replace = FALSE)
```

Extraction de caractéristiques (dfm_*; fcm_*)

Créer une matrice document-caractéristique (dfm) à partir d'un corpus
```
x <- dfm(data_corpus_inaugural,
         tolower = TRUE, stem = FALSE, remove_punct = TRUE,
         remove = stopwords("english"))

head(x, n = 2, nf = 4)
## Document-feature matrix of: 2 documents, 4 features (41.7% sparse).
##                  features
## docs              fellow-citizens senate house representatives
##   1789-Washington               1      1     2               2
##   1793-Washington               0      0     0               0
```

Créer un dictionnaire
```
dictionary(list(negative = c("bad", "awful", "sad"),
                positive = c("good", "wonderful", "happy")))
```

Appliquer un dictionnaire
```
dfm_lookup(x, dictionary = data_dictionary_LSD2015)
```

Choisir des caractéristiques
```
dfm_select(x, dictionary = data_dictionary_LSD2015)
```

Selection aléatoire de documents ou de caractéristiques
```
dfm_sample(x, what = c("documents", "features"))
```

Poids ou lissage des fréquences de caractéristiques
```
dfm_weight(x, type = "prop") | dfm_smooth(x, smoothing = 0.5)
```

Trier ou grouper un dfm
```
dfm_sort(x, margin = c("features", "documents", "both"))
dfm_group(x, groups = "President")
```

Combiner les éléments de dimensions identiques d'un dfm
```
dfm_compress(x, margin = c("both", "documents", "features"))
```

Créer une matrice de co-occurrence des caracteristiques (fcm)
```
x <- fcm(data_corpus_inaugural, context = "window", size = 5)
```
`fcm_compress`/`remove`/`select`/`toupper`/`tolower` sont aussi disponible

Fonctions additionnelles utiles

Localiser des mots-clés dans leur contexte
```
kwic(data_corpus_inaugural, "america*")
```

Fonctions utilitaires

`texts(corpus)`	*Montrer le texte d'un corpus*
`ndoc(corpus/dfm/tokens)`	*Compte les documents/caracteristiques*
`nfeat(corpus/dfm/tokens)`	*Compte les caracteristiques*
`summary(corpus/dfm)`	*Afficher résumé*
`head(corpus/dfm)`	*Retourne la première partie*
`tail(corpus/dfm)`	*Retourne la dernière partie*

Tokenizer un ensemble de textes (tokens_*)

Tokenizer des textes à partir de chaîne de caractères ou de corpus

```
x <- tokens("Powerful tool for text analysis.",
            remove_punct = TRUE, stem = TRUE)
```

Convertir des sequences en tokens composés

```
myseqs <- phrase(c("powerful", "tool", "text analysis"))
tokens_compound(x, myseqs)
```

Choisir des tokens

```
tokens_select(x, c("powerful", "text"), selection = "keep")
```

Créer des ngrams et des skipgrams à partir de tokens

```
tokens_ngrams(x, n = 1:3)
tokens_skipgrams(toks, n = 2, skip = 0:1)
```

Convertir la casse de tokens

```
tokens_tolower(x) | tokens_topupper(x)
```

Raciniser les termes dans un object

```
tokens_wordstem(x)
```

Calculer des statistiques de textes (textstat_*)

Tabuler les fréquences des caractéristiques à partir d'un dfm

```
textstat_frequency(x) | topfeatures(x)
```

Identifier et mesurer des collocations à partir de textes tokenisés

```
toks <- tokens(c("quanteda is a pkg for quant text analysis",
                 "quant text analysis is a growing field"))
textstat_collocations(toks, size = 3, min_count = 2)
```

Calculer la lisibilité d'un corpus

```
textstat_readability(data_corpus_inaugural, measure = "Flesch")
```

Calculer la diversité lexicale d'un dfm

```
textstat_lexdiv(x, measure = "TTR")
```

Mesurer des distances ou similarités à partir d'un dfm

```
textstat_simil(x, "2017-Trump", method = "cosine")
textstat_dist(x, "2017-Trump", margin = "features")
```

Calculer des statistiques de keyness

```
textstat_keyness(x, target = "2017-Trump")
```

Ajuster des modèles basés sur un dfm (textmodel_*)

Analyse des Correspondences (CA)

```
textmodel_ca(x, threads = 2, sparse = TRUE, residual_floor = 0.1)
```

Classification Naïve Bayesienne pour textes

```
textmodel_nb(x, y = training_labels, distribution = "multinomial")
```

Modèle Wordscores pour texte

```
refscores <- c(seq(-1.5, 1.5, .75), NA))
textmodel_wordscores(data_dfm_lbgexample, refscores)
```

Modèle de mise à l'échelle Wordfish Poisson

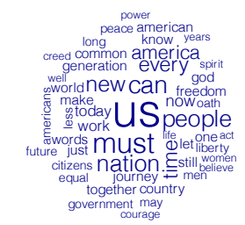
```
textmodel_wordfish(dfm(data_corpus_irishbudget2010), dir = c(6,5))
```

Méthodes sur objet textmodel: `predict()`, `coef()`, `summary()`, `print()`

Représenter caractéristiques ou modèles (textplot_*)

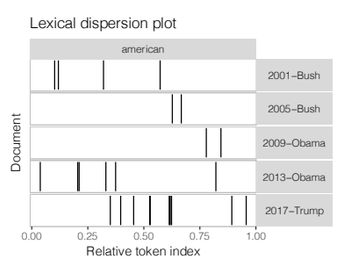
Représenter les caractéristiques comme nuage de mots

```
data_corpus_inaugural %>%
  corpus_subset(President == "Obama") %>%
  dfm(remove = stopwords("english")) %>%
  textplot_wordcloud()
```



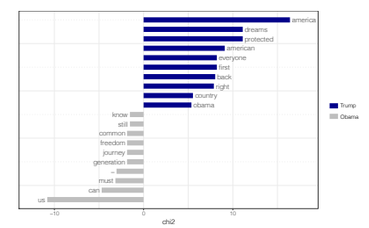
Représenter la dispersion de mot(s)-clé(s)

```
data_corpus_inaugural %>%
  corpus_subset(Year > 1945) %>%
  kwic("american") %>%
  textplot_xray()
```



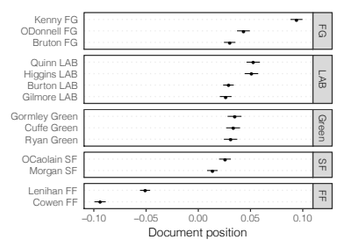
Représenter la keyness des mots

```
data_corpus_inaugural %>%
  corpus_subset(President %in%
                  c("Obama", "Trump")) %>%
  dfm(groups = "President",
      remove = stopwords("english")) %>%
  textstat_keyness(target = "Trump") %>%
  textplot_keyness()
```



Représenter les modèles Wordfish, Wordscores ou CA

```
textplot_scale1d(scaling_model,
  groups = party,
  margin = "documents")
```



Convertir un dfm à un format non-quanteda

```
convert(x, to = c("lda", "tm", "stm", "austin", "topicmodels",
                  "lsa", "matrix", "data.frame"))
```


par **Stefan Müller** et **Kenneth Benoit** • *mullers@tcd.ie*, *kbenoit@lse.ac.uk*

Apprenez-en plus à: http://quanteda.io • updated: 11/18