テキストマイニングツール
RMeCab と RCaBoCha について

石田 基広

徳島大学総合科学部

RMeCab とは何か

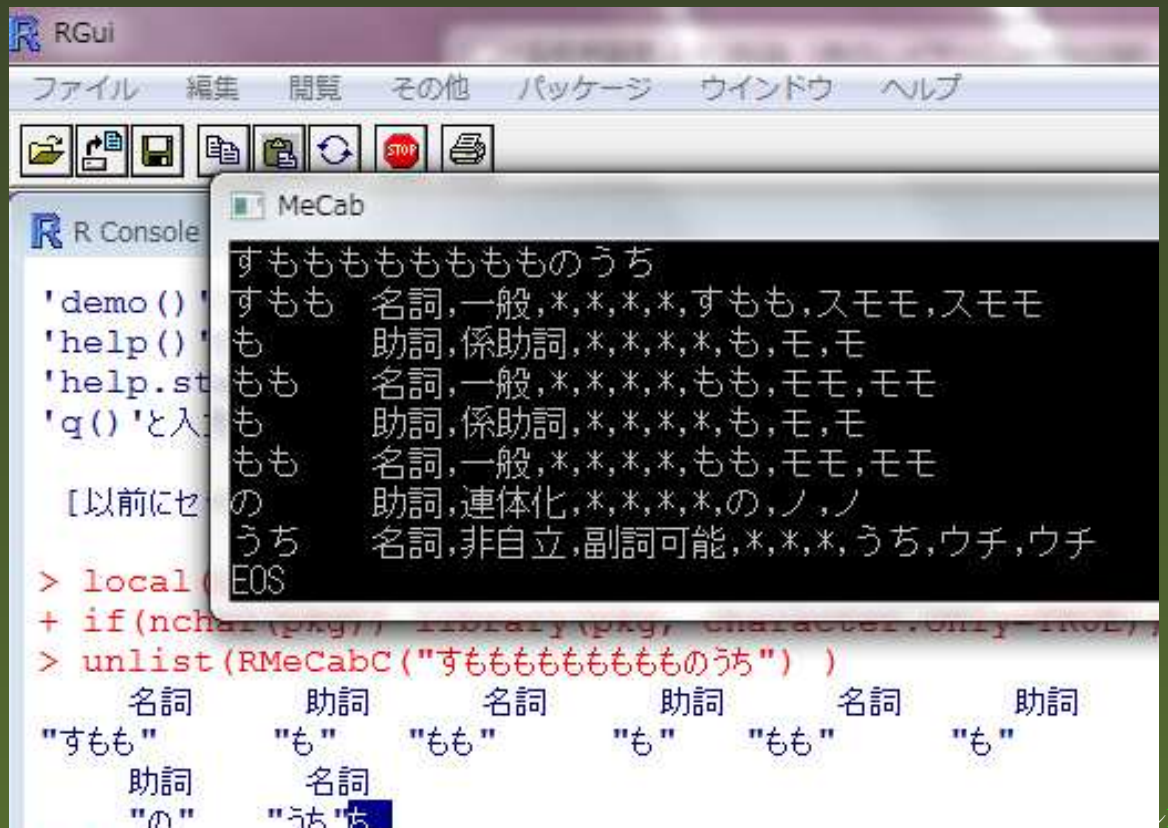
R と MeCab をつなぐインターフェイス

- R フリーのデータ解析・グラフィックス作成ソフトウェア
- MeCab 工藤 拓 氏開発の日本語形態素解析器

RMeCab とは何か

R と MeCab をつなぐインターフェイス

- R フリーのデータ解析・グラフィックス作成ソフトウェア
- MeCab 工藤 拓 氏開発の日本語形態素解析器



footnote2

RMeCab は，名前そのままに，R から MeCab を利用するインターフェイスです．R については，改めて紹介するまでもなく，ここ数年，統計解析やグラフィックスの分野で世界中で普及の進んでいるソフトウェアです．
MeCab の方は，あるいは馴染みのない方もおられるかもしれませんが，現在 Google で活躍されている工藤拓さんが開発された日本語形態素解析器です．
下の画面は，Windows 上で MeCab を操作している画面と，R から RMeCab を使って MeCab を操作しているところです．
テキストマイニングを実行する場合，日本語テキストを文字や文の単位に分割する必要がありますが，MeCab は日本語の文章を単語，正確には形態素単位に分割してくれるソフトです．

和布蕪による日本語形態素解析

```
mecab          # 形態素解析を行う
暑い日が続きます。
```

```
暑い    形容詞, 自立,*,*, 形容詞・アウオ段, 基本形, 暑い, アツイ, アツイ
日      名詞, 非自立, 副詞可能,*,*,*, 日, ヒ, ヒ
が      助詞, 格助詞, 一般,*,*,*, が, ガ, ガ
続き    動詞, 自立,*,*, 五段・カ行イ音便, 連用形, 続く, ツヅキ, ツズキ
ます    助動詞,*,*,*, 特殊・マス, 基本形, ます, マス, マス
.       記号, 句点,*,*,*,*,.  ,.  ,.
EOS
```

和布蕪による日本語形態素解析

```
mecab          # 形態素解析を行う
暑い日が続きます。
```

```
暑い    形容詞, 自立,*,*, 形容詞・アウオ段, 基本形, 暑い, アツイ, アツイ
日      名詞, 非自立, 副詞可能,*,*,*, 日, ヒ, ヒ
が      助詞, 格助詞, 一般,*,*,*, が, ガ, ガ
続き    動詞, 自立,*,*, 五段・カ行イ音便, 連用形, 続く, ツヅキ, ツズキ
ます    助動詞,*,*,*, 特殊・マス, 基本形, ます, マス, マス
.       記号, 句点,*,*,*,*,.  ,.  ,.
EOS
```

- テキスト（集合）を MeCab で解析
- 出力の抽出
- 頻度表作成

footnote3

もう少し MeCab について述べると，この画面にありますように，入力された文章を，形態素に分けるだけではなく，品詞などの情報を抽出してくれます．精度も高く非常に優れたソフトウェアです．ただ，テキストマイニングでは，文章を一つ一つ入力するわけではなく，かなりまとまった分量のテキストを分析対象としますので，このウィンドウでの操作は非効率です．実際にはファイルなどを指定してバックグランドで実行することになります．

たただし MeCab の出力はそのままでは使えません．一番左に単語があり，タブで区切りがあって，品詞の詳細な情報が続き，その後に，活用語であれば，原型，そして読み方が続きます．テキストマイニングでは，このすべてが必要になるわけではなく，多くの場合は，左端の表層形か，あるいは活用語であれば，その原型でしょう．したがって，この出力から適当な要素を取出す必要があります．これはそれほど面倒な処理ではないので，ちょっとしたプログラミングの知識，あるいは Excel でもできると思います．

必要な要素が抽出できれば，そこから頻度表を作成できます．頻度表あるいは分割表が作成できれば，これは数値表ですから，R を使ってデータマイニングを実行できます．

逆にいえば，テキストデータを R で分析するためには，あらかじめ何らかの方法で，MeCab による解析とその結果の抽出を，別のソフトで行う必要があるわけです．これが 2 度手間で面倒だというのが，RMeCab を開発したきっかけです．

この前処理も R で行うことができれば，というのが，RMeCab を作成した動機です．

出力の順番

表層型品詞、品詞細分類１、再分類２、再分類３、活用形、活用型、原形、読み、発音

お正月の読みはオショウガツ発音はオショーガツ

RMeCab パッケージ

- R の拡張機能(パッケージ)
- C++
- MeCab ライブラリ (libmecab.{so,dll})

RMeCab パッケージ

- R の拡張機能(パッケージ)
- C++
- MeCab ライブラリ (libmecab.{so,dll})
- 2008 年明けから実装
- CRAN 未登録
  野良パッケージとして公開

RMeCab パッケージ

- R の拡張機能(パッケージ)
- C++
- MeCab ライブラリ (libmecab.{so,dll})
- 2008 年明けから実装
- CRAN 未登録
  野良パッケージとして公開
- 現在の ver. RMeCab_0.91
- ユーザ環境の設定 (辞書あるいは.mecabrc の読み込み)

RMeCab パッケージ

- R の拡張機能(パッケージ)
- C++
- MeCab ライブラリ (libmecab.{so,dll})
- 2008 年明けから実装
- CRAN 未登録
  野良パッケージとして公開
- 現在の ver. RMeCab_0.91
- ユーザ環境の設定 (辞書あるいは.mecabrc の読み込み)
- 石田基広『Rによるテキストマイニング入門』森北出版

footnote4

RMeCab は R の拡張機能として開発を行なっています．R はユーザーが独自に機能を追加できるようになっており，これをパッケージと読んでいます．
R は，多言語とのインターフェイスがよく整備されていて，拡張がそれほど難しくありません．RMeCab の場合，C++で実装を行ない，MeCab のライブラリと通信を行なっています．
最初のバージョンを作成したのは 2008 年の 3 月頃です．実は，R でパッケージを作成すると，これを CRAN という R のサイトで世界中に公開することが可能です．ここにはすでに 1000 を遥かにこえる数のパッケージが公開されてます．
当然，このサイトへの登録を考えたのですが，登録にはマニュアルを整備するなど，それなりに手間が必要で，結論からいうと，時間的余裕がなく，先のばしにして，いまもそのままです．
その後，改良や拡張を進め，現在のバージョンは 0.91 です．一番，最後に行なった改良では，ユーザーが独自に追加した辞書を RMeCab の関数の引数として指定できるようにしました．
また RMeCab を使った本なども出してもらっています．

RMeCab とは何か

R から MeCab に文字列あるいはテキスト（集合）を送り，その結果を受けとる

```
> library(RMeCab)
> unlist (RMeCabC("まだまだ暑い日が続きます．", 1) )
副詞            形容詞     名詞     助詞     動詞     助動詞     記号
"まだまだ"     "暑い"     "日"     "が"     "続く"     "ます"     "．"


> res < - RMeCabText("yukiguni.txt")
file = yukiguni.txt
> res
[[1]]
[1] "国境"     "名詞"     "一般"     "*"     "*"
[6] "*"     "*"     "国境"     "コッキョウ"     "コッキョー"
[[2]]
[1] "の"     "助詞"     "格助詞"     "一般"     "*"
[6] "*"     "*"     "の"     "ノ"     "ノ"
```

footnote5

ここに示したのは RMeCab のもっともシンプルな関数です．RMeCabC の方は，R で日本語をそのまま入力すると，その文章を形態素に分け，品詞をラベルとして付加して返す関数です．ここでは第 2 引数に 1 を与えているので，活用語は原型に戻されていますが，これを指定しないと表層形のまま出力されます．

一方，RMeCabText は，指定されたテキストを MeCab に送り，さきほどの MeCab を出力をほぼそのまま，ただし R のリストという形式に直して出力する関数です．リストというデータ形式にすることで，例えば，ある単語の発音だけを取りだすなどの処理が可能です．

ただ，この二つの関数は RMeCab でもデモンストレーションとして実装しています．

MeCab の出力を整形する

単語の頻度行列：デフォルトは全ての品詞を対象

```
> docDF("doc",pos = c("名詞","形容詞"),type = 1)
```

TERM	POS1	POS2	doc1	doc2	doc3
彼女	名詞	代名詞	0	1	1
数学	名詞	一般	0	1	1
真面目	名詞	形容動詞語幹	1	0	0

MeCab の出力を整形する

単語の頻度行列：デフォルトは全ての品詞を対象

> docDF("doc",pos = c("名詞","形容詞"),type = 1)

TERM	POS1	POS2	doc1	doc2	doc3
彼女	名詞	代名詞	0	1	1
数学	名詞	一般	0	1	1
真面目	名詞	形容動詞語幹	1	0	0

文字の行列

> docDF("doc")

Ngram	doc1	doc2	doc3
数	1	1	1
学	0	1	1
生	1	1	0

MeCab の出力を整形する

単語の頻度行列：デフォルトは全ての品詞を対象

```
> docDF("doc",pos = c("名詞","形容詞"),type = 1)
```

TERM	POS1	POS2	doc1	doc2	doc3
彼女	名詞	代名詞	0	1	1
数学	名詞	一般	0	1	1
真面目	名詞	形容動詞語幹	1	0	0

文字の行列

```
> docDF("doc")
```

Ngram	doc1	doc2	doc3
数	1	1	1
学	0	1	1
生	1	1	0

単語の共起行列

```
> docDF("doc", pos = c("名詞","形容詞"), co = 1)
```

TERM	POS1	POS2	学生	彼女	数学
学生	名詞	一般	2	1	1
彼女	名詞	代名詞	1	2	2
数学	名詞	一般	1	2	2

MeCab の出力を整形する

単語の頻度行列：デフォルトは全ての品詞を対象

> docDF("doc",pos = c("名詞","形容詞"),type = 1)

TERM	POS1	POS2	doc1	doc2	doc3
彼女	名詞	代名詞	0	1	1
数学	名詞	一般	0	1	1
真面目	名詞	形容動詞語幹	1	0	0

文字の行列

> docDF("doc")

Ngram	doc1	doc2	doc3
数	1	1	1
学	0	1	1
生	1	1	0

単語の共起行列

> docDF("doc", pos = c("名詞","形容詞"), co = 1)

TERM	POS1	POS2	学生	彼女	数学
学生	名詞	一般	2	1	1
彼女	名詞	代名詞	1	2	2
数学	名詞	一般	1	2	2

node 前後の共起頻度

> collocate("kumo", node = "極楽", span =3)

Term	Before	After	Span	Total
蓮池	0	4	4	4
蜘蛛	0	2	2	14
行く	1	0	1	4
釈迦	2	0	2	7
MORPHEMS	18	13	31	413
TOKENS	40	30	70	1808

footnote6

RMeCab のメインの関数は，MeCab の出力を，R の統計関数で処理できるような形，具体的には分割表や行列に変換して出力することです．
これは docDF という，実は最初に紹介させて頂いた拙著が出版された後で加えた関数の出力です．たとえば，この出力では，表の最初の 3 列は語とその品詞情報二つがあり，これ以降の列はファイル名になります．ここでは doc で始まる三つのファイルを指定して実行しています．
なお，出力はすべてその一部のみを示しています．
これは単語を抽出し，各文書での頻度を示す行列です．抽出する品詞は，この関数ではデフォルトではすべての品詞を抽出しますが，たとえば名詞と形容詞だけという指定も可能です．
同じ関数で文字だけを抽出するともできます．docDF 関数はデフォルトでは文字単位でテキストを分割するようになっています．この場合は mecab の処理ではなく，RMeCab が内部で文字分割しています．
こちらは共起行列といわれるもので，行と列に同じ品詞が対応して，交差するセルには，この場合には，二つの単語のどちらもが利用されてる文書数がでます．
こちらはあるキワードを中心として，ある範囲内に出現した単語とその頻度を示しています．
Before が node の前での出現数，After は後，Span は二つの合計，そして Total というのは，対象テキストでそのワードが出現した総頻度です．

Ngram 行列

国境の長いトンネルを抜けると雪国であった。夜の底が白くなった。信号所に汽車が止まった。

単語の bigram

```
> docDF("yukiguni", type = 1,N = 2)
```

TERM	POS1	POS2	yukiguni
。-信号	記号-名詞	句点-一般	1
。-夜	記号-名詞	句点-副詞可能	1
ある-た	助動詞-助動詞	*-*	1
が-止まる	助詞-動詞	格助詞-自立	1
が-白い	助詞-形容詞	格助詞-自立	1
た-。	助動詞-記号	*-句点	3

Ngram 行列

国境の長いトンネルを抜けると雪国であった。夜の底が白くなった。信号所に汽車が止まった。

単語の bigram

```
> docDF("yukiguni", type = 1,N = 2)
```

TERM	POS1	POS2	yukiguni
。-信号	記号-名詞	句点-一般	1
。-夜	記号-名詞	句点-副詞可能	1
ある-た	助動詞-助動詞	*-*	1
が-止まる	助詞-動詞	格助詞-自立	1
が-白い	助詞-形容詞	格助詞-自立	1
た-。	助動詞-記号	*-句点	3

文字の trigram

```
> docDF("yukiguni", N = 3)
```

Ngram	yukiguni
。- 信 - 号	1
。- 夜 - の	1
あ - っ - た	1
い - ト - ン	1
が - 止 - ま	1

footnote

一方，テキストマイニングで求められる頻度表としては，ほかに NGram があります．すなわちある語なり文字の繋がりとその頻度を表す行列です．たてえば，これは川端康成の『雪国』の冒頭部分ですが左は単語あるいは品詞のつながりを示し，右は文字の場合です，デフォルトでは二つの繋がりをカウントします．これをバイグラムといいます．引数で N を指定すると，三つ以上の繋がりの頻度をカウントすることも可能です．右の文字の出力の方は，文字がみっつ連なるパターンで，トリグラムと呼びます．

Ngram 行列 2

ハイフンでまとめた trigram

> docDF("morikita", pos = c("名詞","形容詞"), type = 1, N =3)

TERM	POS1	POS2	doc1	doc2	doc3
これ-知識-科学	名詞-名詞-名詞	代名詞-一般-一般	0	0	1
その道-スペシャリスト-方々	名詞-名詞-名詞	一般-一般-一般	0	1	0
づくり-私-達	名詞-名詞-名詞	接尾-代名詞-接尾	0	0	1
スペシャリスト-方々-編集	名詞-名詞-名詞	一般-一般-サ変接続	0	1	0
フィールド-これ-知識	名詞-名詞-名詞	一般-代名詞-一般	0	0	1
世の中-こと-使命	名詞-名詞-名詞	一般-非自立-一般	1	0	0

Ngram 行列 2

ハイフンでまとめた trigram

> docDF("morikita", pos = c("名詞","形容詞"), type = 1, N =3)

TERM	POS1	POS2	doc1	doc2	doc3
これ-知識-科学	名詞-名詞-名詞	代名詞-一般-一般	0	0	1
その道-スペシャリスト-方々	名詞-名詞-名詞	一般-一般-一般	0	1	0
づくり-私-達	名詞-名詞-名詞	接尾-代名詞-接尾	0	0	1
スペシャリスト-方々-編集	名詞-名詞-名詞	一般-一般-サ変接続	0	1	0
フィールド-これ-知識	名詞-名詞-名詞	一般-代名詞-一般	0	0	1
世の中-こと-使命	名詞-名詞-名詞	一般-非自立-一般	1	0	0

語ごとに列を別にした trigram

> docDF("morikita", pos = c("名詞","形容詞"), type = 1, N =3, nDF = 1)

N1	N2	N3	POS1	POS2	doc1	doc2	doc3
これ	知識	科学	名詞-名詞-名詞	代名詞-一般-一般	0	0	1
その道	スペシャリスト	方々	名詞-名詞-名詞	一般-一般-一般	0	1	0
づくり	私	達	名詞-名詞-名詞	接尾-代名詞-接尾	0	0	1
スペシャリスト	方々	編集	名詞-名詞-名詞	一般-一般-サ変接続	0	1	0
フィールド	これ	知識	名詞-名詞-名詞	一般-代名詞-一般	0	0	1
世の中	こと	使命	名詞-名詞-名詞	一般-非自立-一般	1	0	0

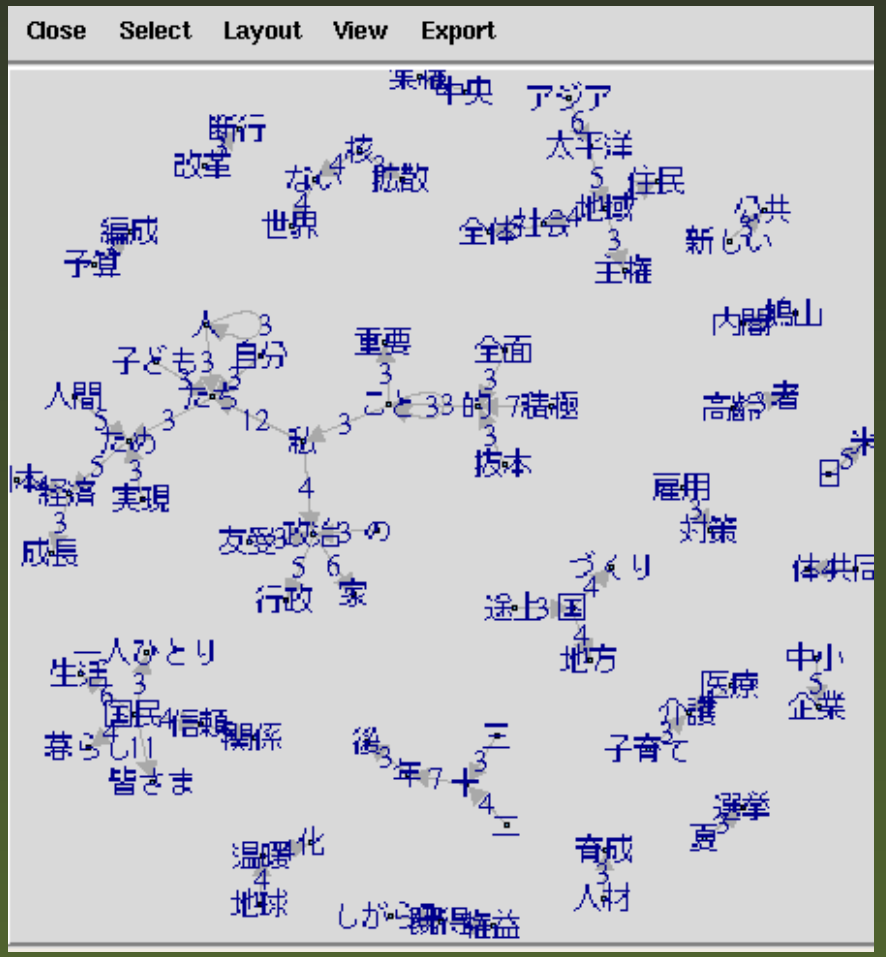
footnote

こちらも Ngram の続きなのですが，RMeCab では Ngram の出力に二つのパターンを用意しています．これは単語が三つ続くパターン，つまり trigram を出力したものですが，上では，最初の列にみっつの繋がりをまとめていますが，下ではみっつの単語それぞれを個別に列にとっています．これは．たとえば R でネットワーク分析などを行なう場合，それれの列が独立していた方が便利だからです．

ネットワーク分析

金明哲著『テキストデータの統計科学入門』より

```
hato <- docDF("Hato.txt", type =
1, N = 2, nDF = 1)
hato  <-  hato[hato$Hato.thatot  >
2,]
hato <- hato[,c(1,2,5)]
library (igraph)
hato.g <- graph.data.frame(hato)
E(hato.g)$weight <- hato[,3]
tkplot(hato.g,      vertex.label     =
V(hato.g)$name,
vertex.size = 1,
edge.label = E(hato.g)$weight)
```



footnote

これは R の igraph パッケージを使って単語の繋がりをビジュアル化したものですが，ここで，たとえば，ここの関数で，単語ごとに列がとってあることが必要になります.

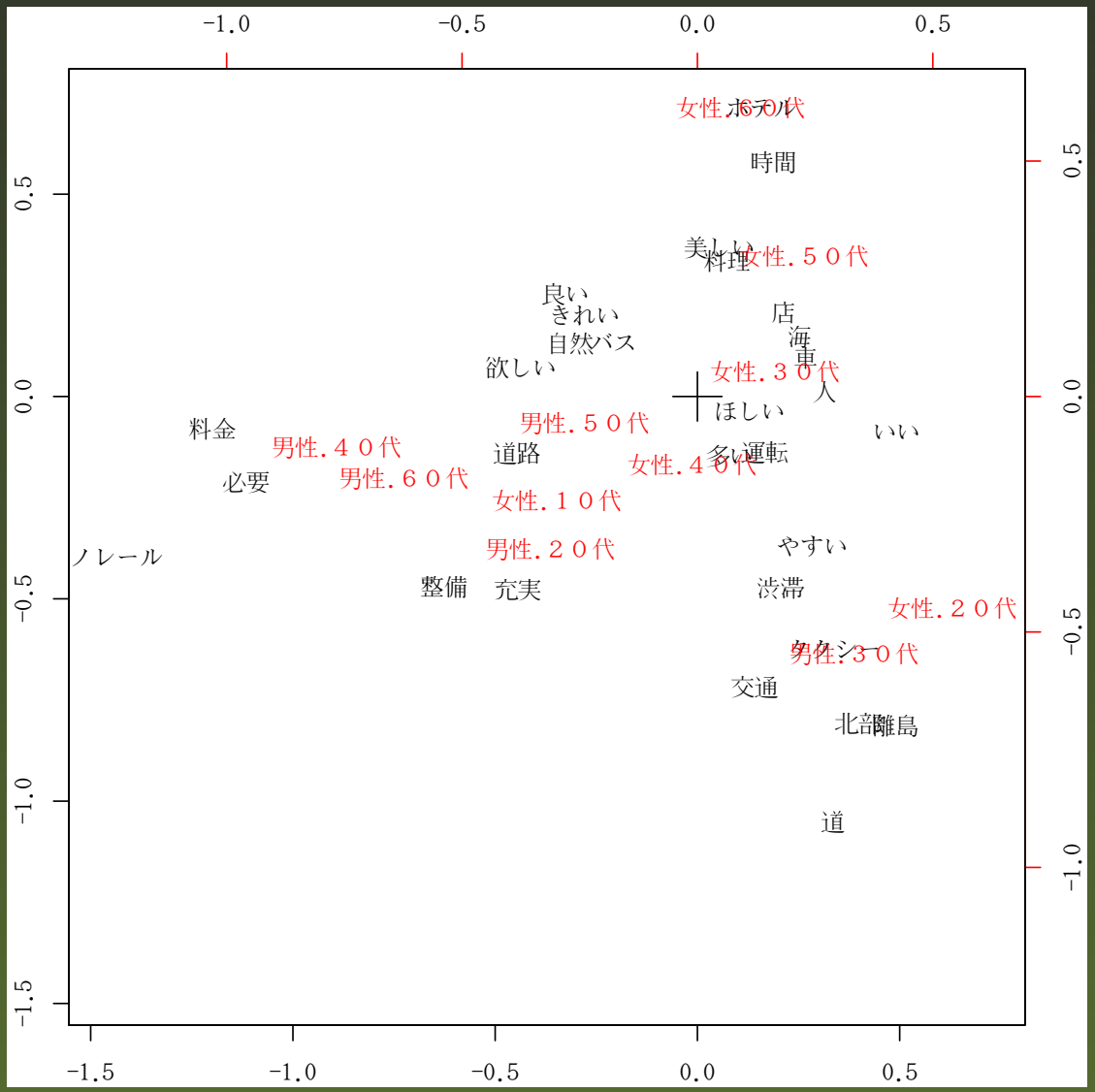
その他の応用事例1

沖縄観光分析

性別	年代	意見
女性	60 代	花粉で悩んでいますので、今回過去に訪れて思い切り美味しい
		クシャミの出ない空気を吸えて幸せでした。
		今後花粉の時季に 3ヶ月くらいを目途に滞在を考えております。
		そういう宿泊施設の宣伝等ありましたら、観光情報とともに
		インターネットで流していただくと利用しやすく有難く思っています。
男性	50 代	空港→国際通りまでタクシーを利用。
		乗車したらどちらまで、下車したらありがとうございますの基本動作が全く無い。
		沖縄の観光のためにもお客様に気持ちよく帰ってもらう為にも教育をした方が良い。
女性	20 代	気候が暖かくて住みやすく、沖縄の人たちは名古屋人と違ってのんびりしていて楽しい。
		車もせかす人もいなく運転しやすかった。
		今回友達と来たけど、次回は母と一緒に遊びに行きたいです。

沖縄観光アンケート

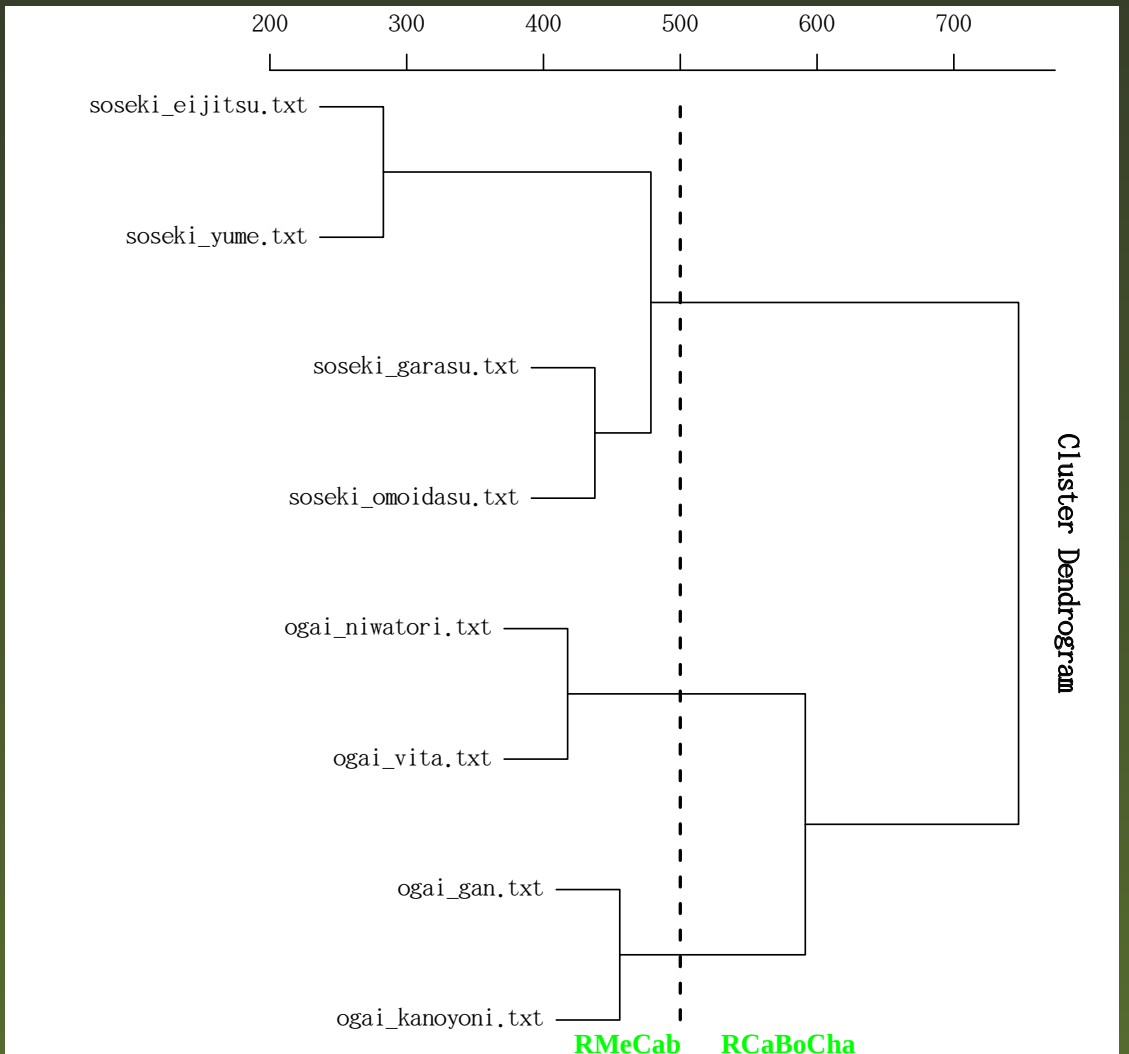
対応分析



応用事例

書き手の判別

term	鴎外１	鴎外２	漱石３	漱石４
が、	107	79	34	31
て、	251	245	155	81
で、	86	69	41	40
と、	63	42	99	27
に、	76	101	45	45



RMeCab の主な関数

- データファイル (csv などで自由記述を含む列がある)
  - 形態素解析のリスト： RMeCabDF
  - 単語行列の作成： docMatrix, docDF
  - Ngram（語，文字，品詞）行列： docNgramDF， docDF
- テキストファイル（ファイル全体がテキスト）
  - 形態素解析のリスト： RMeCabText
  - 頻度表： MeCabFreq , docDF
  - 単語行列の作成：docMatrix， docMatrix2, docDF
  - Ngram（語，文字，品詞）行列：Ngram， NgramDF, NgramDF2, docNgram, docNgram2, docDF

RMeCab の主な関数

- データファイル (csv などで自由記述を含む列がある)
  - 形態素解析のリスト： RMeCabDF
  - 単語行列の作成： docMatrix, docDF
  - Ngram（語，文字，品詞）行列： docNgramDF， docDF
- テキストファイル（ファイル全体がテキスト）
  - 形態素解析のリスト： RMeCabText
  - 頻度表： MeCabFreq , docDF
  - 単語行列の作成：docMatrix， docMatrix2, docDF
  - Ngram（語，文字，品詞）行列：Ngram， NgramDF, NgramDF2, docNgram, docNgram2, docDF

実は docDF だけですべてできます

footnote

RMeCab で実装している主な関数をあげてみます．この他に，頻度に tfidf と言われる重み付けする関数もありますが，これらは単独では使わないので，省略します．
RMeCab に実装した関数は主に二つありまして，一つは CSV 形式のファイルの特定の列に，アンケートの自由記述文が記録されている場合を想定した関数です．
もう一つは，ファイルに日本語の文章だけが書かれていることを想定したファイルですね．

実は docDF だけですべてできます．

RMeCab に欠けているもの

- 否定や係り受け → RCaBoCha パッケージ
  - 易しい ⇔ 易しくない
  - 太郎 は、家に 帰った。

RMeCab に欠けているもの

- 否定や係り受け → RCaBoCha パッケージ
  - 易しい ⇔ 易しくない
  - 太郎 は、家に 帰った。
- 語義分類の自動化
  - いぬ → 犬 → イヌ
  - 犬 ⇒ スパイ ⇒ ドッグ

RMeCab に欠けているもの

- 否定や係り受け → RCaBoCha パッケージ
  - 易しい ⇔ 易しくない
  - 太郎 は、家に 帰った。
- 語義分類の自動化
  - いぬ → 犬 → イヌ
  - 犬 ⇒ スパイ ⇒ ドッグ
- CaBoCha 工藤拓氏開発 (係り受けを判定)

RMeCab に欠けているもの

- 否定や係り受け → RCaBoCha パッケージ
  - 易しい ⇔ 易しくない
  - 太郎 は、家に 帰った。
- 語義分類の自動化
  - いぬ → 犬 → イヌ
  - 犬 ⇒ スパイ ⇒ ドッグ
- CaBoCha 工藤拓氏開発 (係り受けを判定)
- 日本語 Wordnet 情報通信研究機構の日本語の意味辞書

footnote

形態素解析の結果を集計することで，もとのテキストデータが，構造化された頻度表あるいは行列として十分に利用可能になるわけですが，ただ，もう少しデータを細かく調整したいという場面はあります．
この二つの文には，どちらも「面白い」という語がありますが，文の意味内容はまったく反対です．これを形態素解析して単純に「面白い」の頻度だけを比べてしまうと，もとの文の意味の違いが分らなくなります．
もう一つ，重要なのは，日本語のばあい，表記が多数あり，これをまとめることが必要になります．さらには語義をまとめる必要もあるはずです．
前者の係り受けについては市販のテキストマイニングソフトウェアでも，ユーザー辞書定義機能として実装されてます．
後者の類義語統一については，ある程度サポートしているソフトもあるようですが，ただ，基本的には，ユーザーの側で独自に辞書を作って対応することになると思いますが，これはかなりの手間です．

RCaBoCha パッケージ

CaBocha の出力から係り受けを抽出

```
> RCaBoChaFreq("それは面白い本であった。
しかし、この本に比べると面白くはない。")
```

Term	Pos	Freq
比べる	動詞	1
面白い	形容詞	1
面白い+ない	形容詞+形容詞	1

RCaBoCha パッケージ

CaBocha の出力から係り受けを抽出

```
> RCaBoChaFreq("それは面白い本であった。
しかし、この本に比べると面白くはない。")
```

Term	Pos	Freq
比べる	動詞	1
面白い	形容詞	1
面白い+ない	形容詞+形容詞	1

```
> res <- RCaBoChaMx("morikita")
> res[res$POS1 %in% c("名詞+動詞"), 1:5]
```

TERM	POS1	POS2	doc1	doc2	doc3
技術+支える	名詞+動詞	一般+動詞	0	0	1
役割+担う	名詞+動詞	一般+動詞	0	0	1
知識+活かす	名詞+動詞	一般+動詞	1	0	0

footnote

で，RMeCab と平行して，私は CaBoCha の出力を取り込む RCaBoCha パッケージ を開発中です．ここでは「ない」に係る単語を別にカウントすることをしています．デフォルトでは「ない」との係り受けのみ抽出しますが，これは変更することができます．
こちらの出力は，特定のキーワードではなく，係り受け関係をすべて出力する関数です．デフォルトでは名詞と形容詞，動詞の係り受け関係を出力しますが，これも引数で変更可能です．この下は，結果から，R の検索用演算子を利用して，名詞と動詞の組み合わせを抽出したところです．

今後の実装に向けて

類義語の統合

日本語 WordNet (SQLite の DB で配布) の取り込み

```
> RWordNet( "犬" )
```

犬	犬	犬	犬	犬	犬
"いぬ"	"まわし者"	"イヌ"	"スパイ"	"ドッグ"	"回し者"
犬	犬	犬	犬	犬	犬
"回者"	"密偵"	"工作員"	"廻し者"	"廻者"	"探"
犬	犬	犬	犬	犬	犬
"探り"	"洋犬"	"犬"	"秘密捜査員"	"諜報員"	"諜者"
犬	犬	犬	犬	犬	
"間者"	"間諜"	"隠密"	"飼い犬"	"飼犬"	

footnote

一方，こちらは日本語 Wordnet のデータベースを R で検索するパッケージを作ってみまして，利用したところです．こうすることで，同義語を抽出し，ある程度自動的に類義語を一つにまとめることができるようになると思います．
あとは実際にテキストデータを分析するうえで，こうした類義語の統一などのためにあると便利だと思われる機能を整理したうえで， RCaBoCha には取り込む予定でいます．

最後に

GUI 化

やはりコマンドラインでの実行は厳しい

- Rcommander