

Rでグラフつくるの！

7/27 第80回R勉強会@東京

@wkwk\_soprano

# 自己紹介

- 名前 :

wkwk\_soprano

右のアイコンで生息してます

- やっていること :

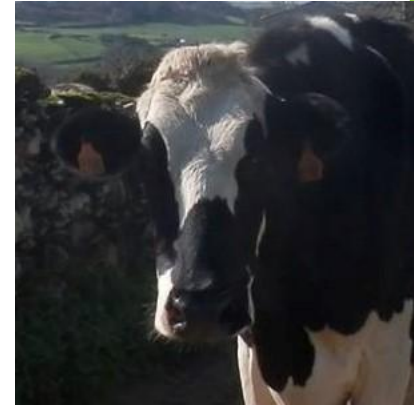
レコメンドしてます

基本的にPython使ってます

(しかしエンジニアリング苦手)

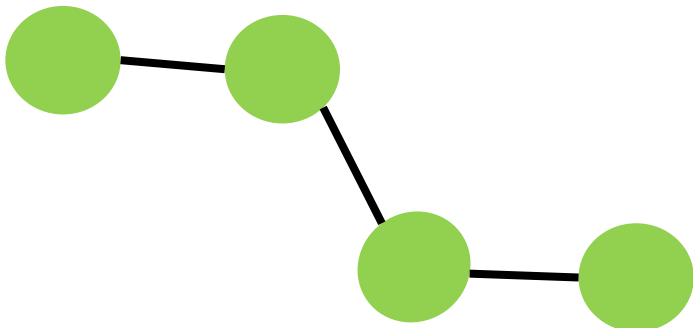
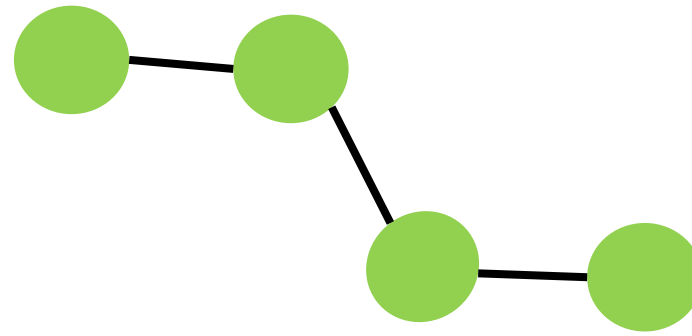
- R歴 :

1年あるかないか。R触るの5年ぶりぐらいです



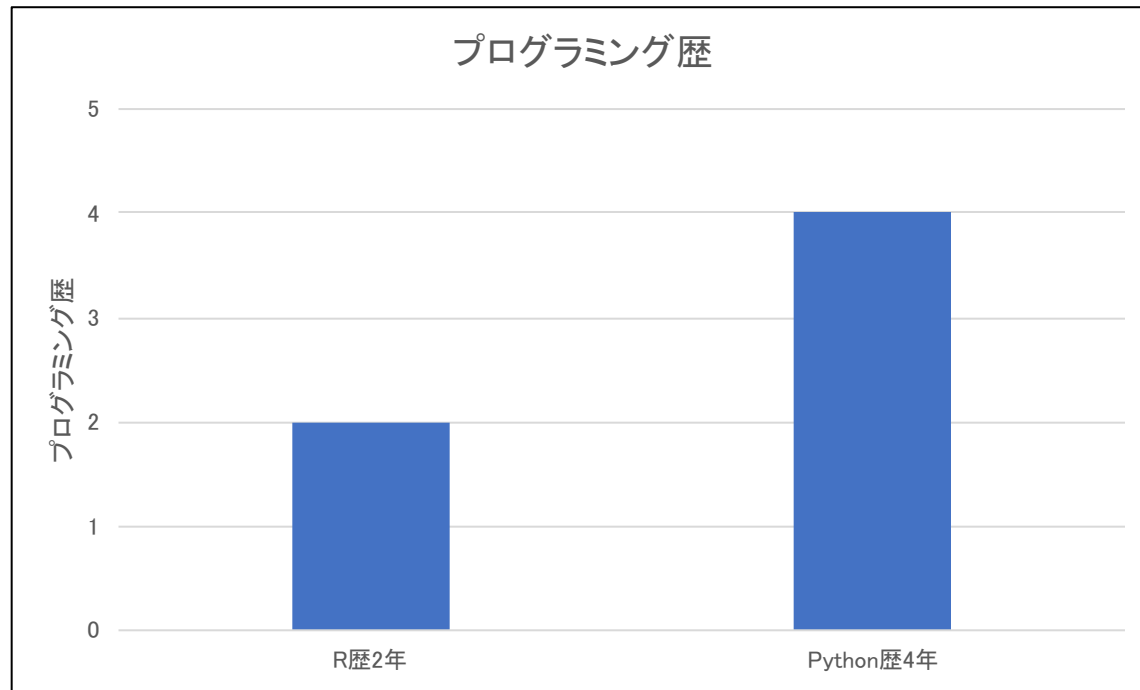
# 本日の目的

1. 徐々にRで何かしたかった
2. グラフのデータセット作った報告



# 「グラフ」

今日扱う「グラフ」は



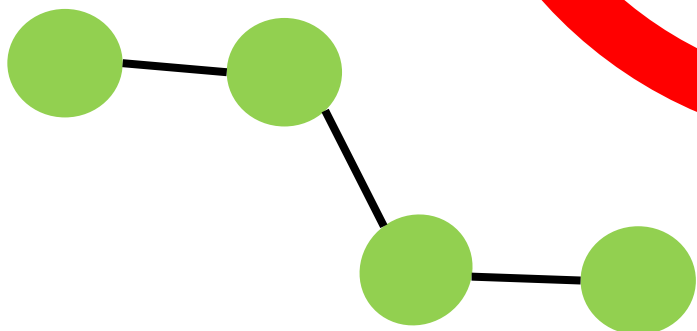
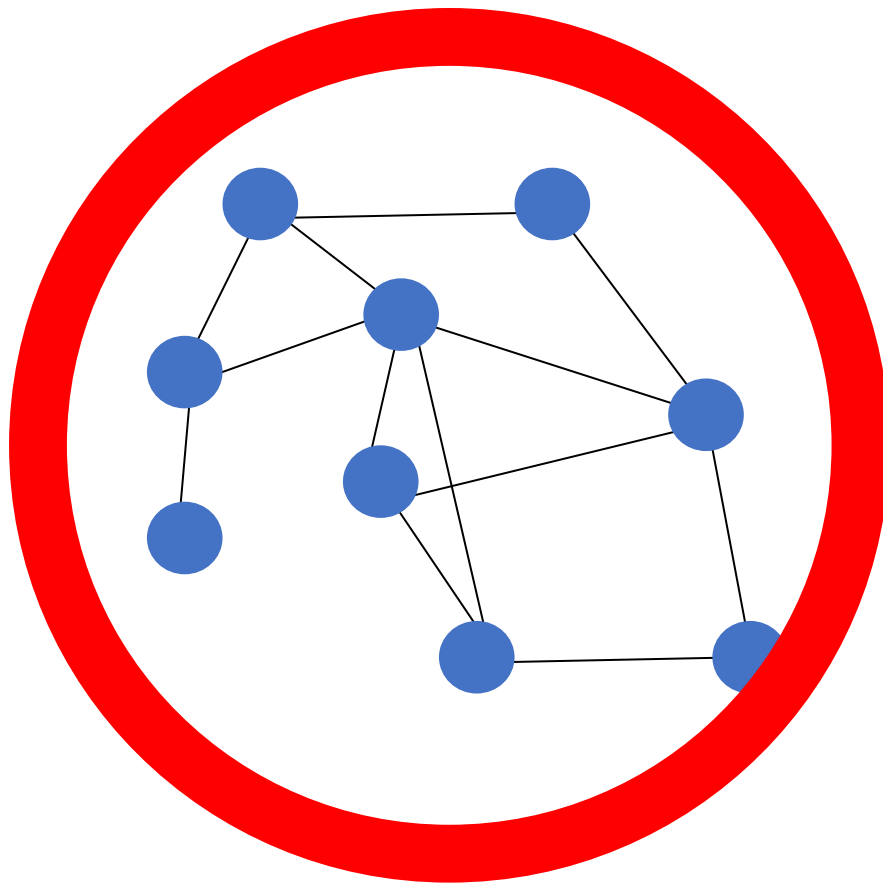
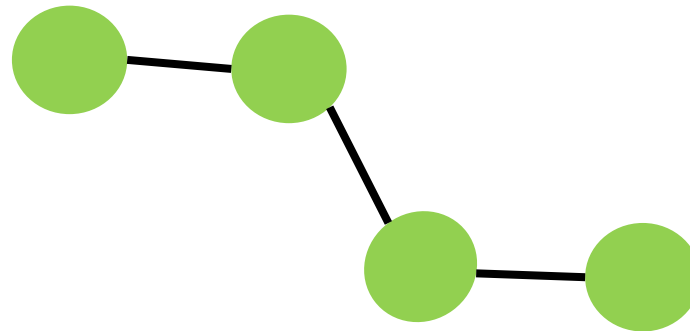
「グラフ」

今日扱う「グラフ」は



「グラフ」

こっち



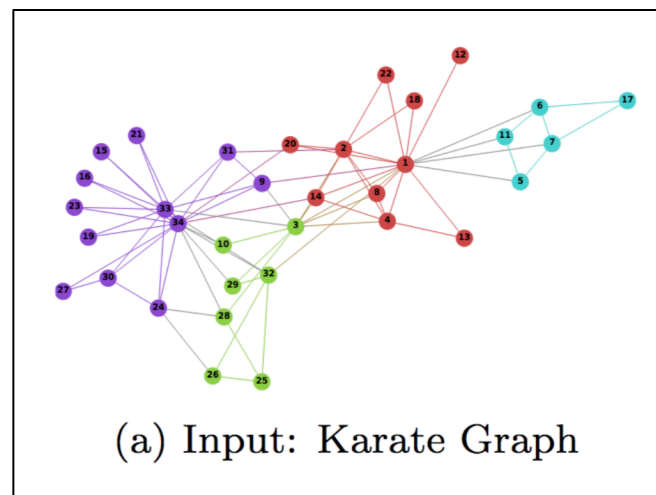
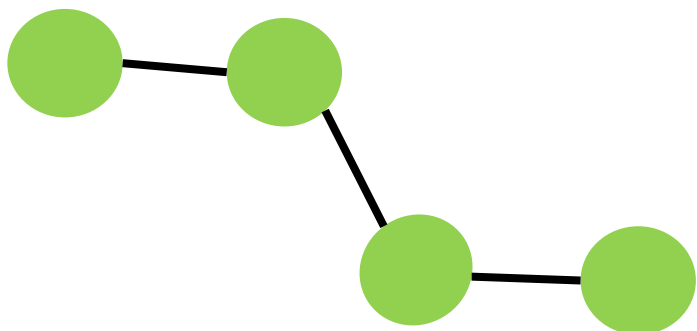
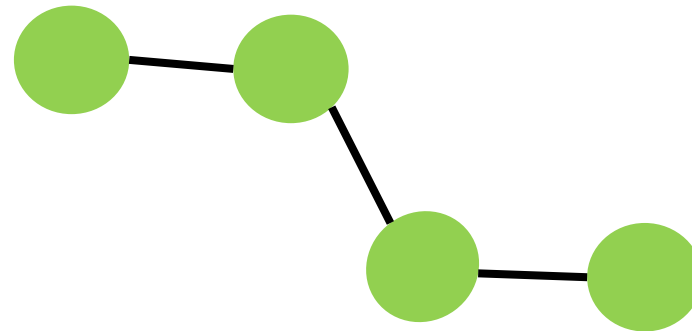
# 使われ方

- 人間関係

ex.) DeepWalk論文のKarate Club (右下図)

- レコメンド

ex.) ユーザとアイテムのネットワーク



Perozzi et al., *DeepWalk: Online Learning of Social Representations* より

# グラフ面白い

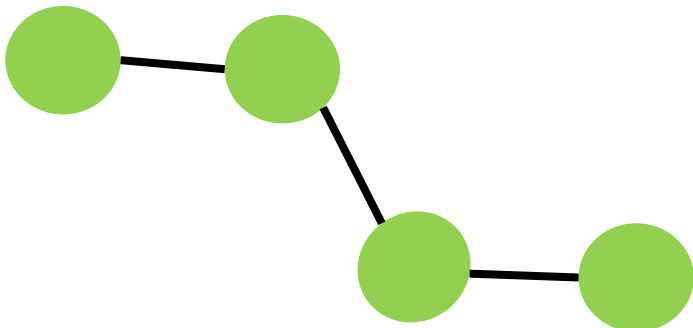
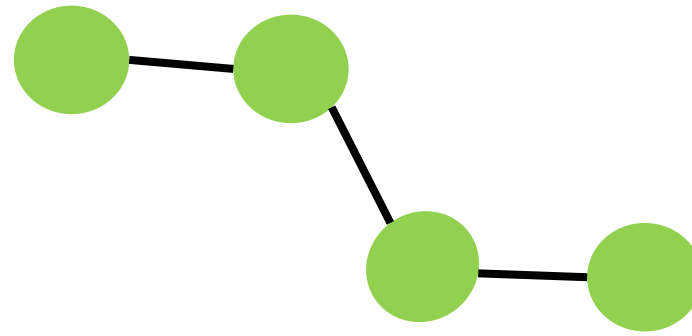
- やってみたい

←当初実務でも試してみたがデータの性質上結局使えなかった

- データセットない

←「ちょっとだけ試してみたい」と思ったが手頃なグラフ用データセットは案外ない。

→作ればええやん（作ってみた）





# ワンピースデータセット

- 概要

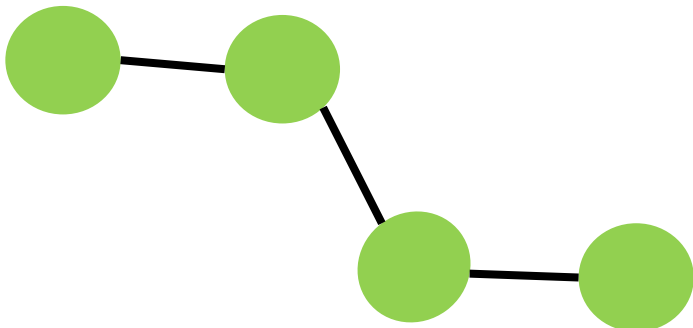
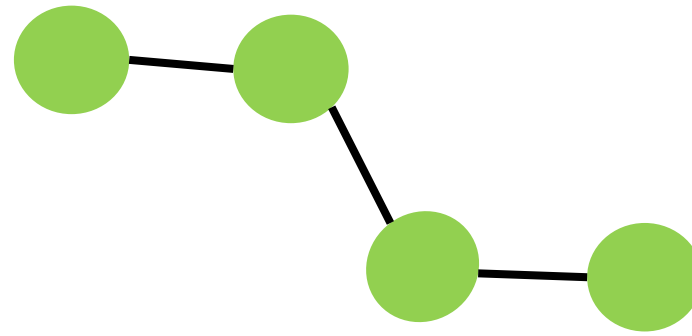
ワンピースのキャラクターの人間関係で  
グラフを作るためのデータセット

- 作成方法

キャラクター同士の共起回数をカウント  
無向グラフとして作成

- 主な作成のルール：

1. 1コマ内の共起で1カウント
2. 扉絵は除外、コマからはみ出しも除外
3. 目視できる程度ならばOK←



# ワンピースデータセット (つづき)

- 対象巻 :

1巻-23巻

- このデータセットは公開しています

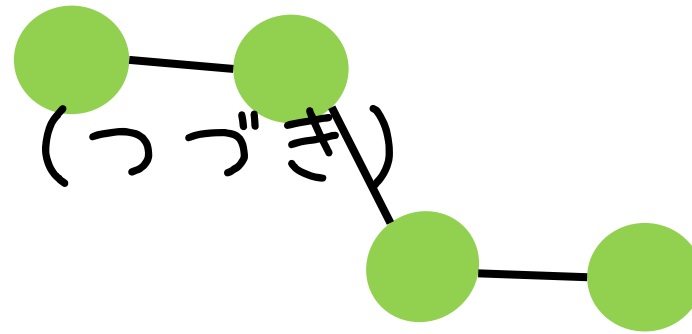
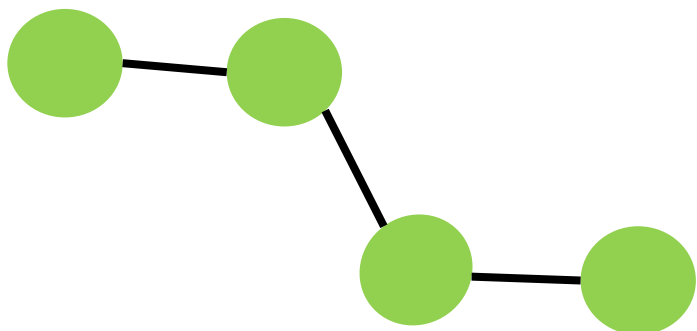
→グラフの練習したい方はご自由にお使い下さい

リンク : <https://drive.google.com/open?id=1y0uDbPLsMB0C5KpjT9CDDmQLAu0mZK2N>

- 余談:

全編手作業 (1冊あたり1時間かかった)

実はまだ20巻までしかできていない (時間切れ)

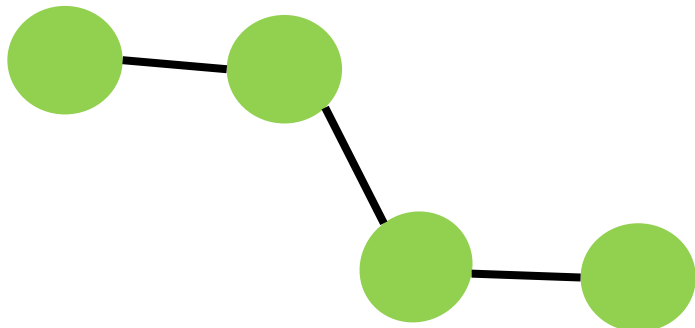
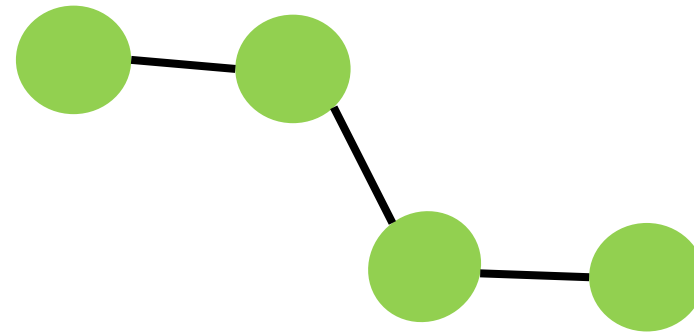


# データセットの見た目

繋がっているノード名がV1, V2


V3には共起回数（今回は正規化なし）

無向グラフのためV1とV2が逆になったものも後半に現れる（V3は同じ）



```
> df[1:10,]
```

	V1	V2	V3
2	Luffy	Shanks	27
3	Shanks	Ben_Beckman	6
4	Ben_Beckman	Lucky_Roux	4
5	Luffy	Lucky_Roux	4
6	Luffy	Ben_Beckman	3
7	Luffy	Makino	9
8	Shanks	Makino	10
9	Shanks	Higuma	12
10	Makino	Higuma	7
11	Luffy	Higuma	7

- 
- gist:  
<https://>



# Graph Embedding

- キャラクターを分散表現に直したい

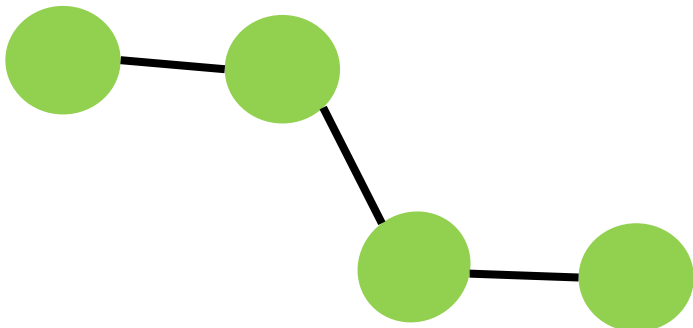
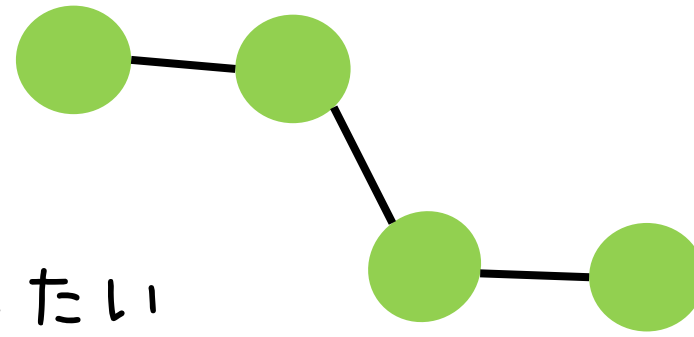
←せっかくキャラの人間関係をグラフにできたので  
Embeddingもやってみたい

←というか最初調べたときはこれをやってた

- 手法

←DeepWalkやLINEなどが候補

RではSpectral Embeddingもあるようだが馴染みない  
→今回はLINEを採用（実装上の都合から）



# 今回使用する手法

- LINE

Large-scale Information Network Embeddingの略  
緑色の会社は無関係

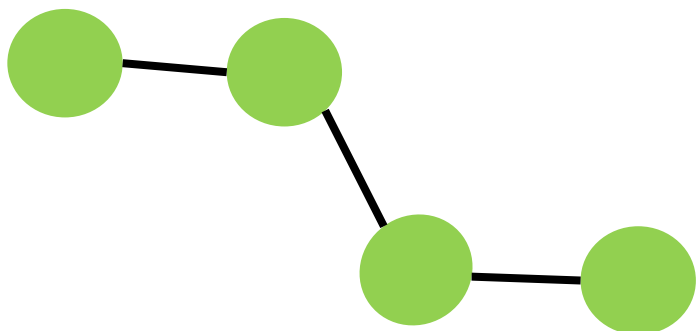
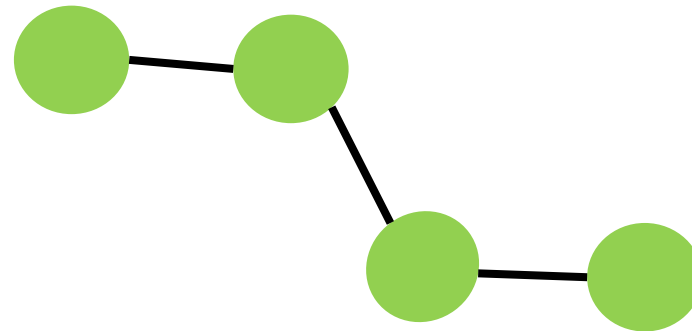
- Rでの実装も一応あるが...

なんだかインストールがうまく行かず断念  
最終更新も1年前なのでちょっと期待薄

リンク : <https://github.com/YosefLab/Rline>

- **今回は元のC++の実装を使っています**

著者のGithubリポジトリ : <https://github.com/tangjianpku/LINE>



# 手法の情報（概要だけ）

## • 論文

Tang, Jian, et al. “[LINE: Large-scale Information Network Embedding](#)”, *Proceedings of the 24th International Conference on World Wide Web*, 2015

## • 概要

- ローカルな構造 (first order proximity) もグローバルな構造 (second order proximity) も保持しながら最適化を行う。
- alias samplingによる高速化

## • 詳しくはWebで

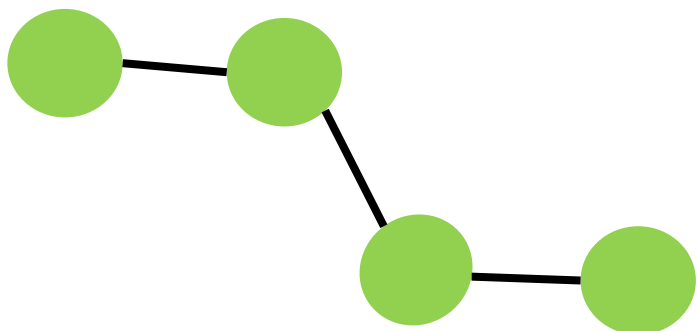
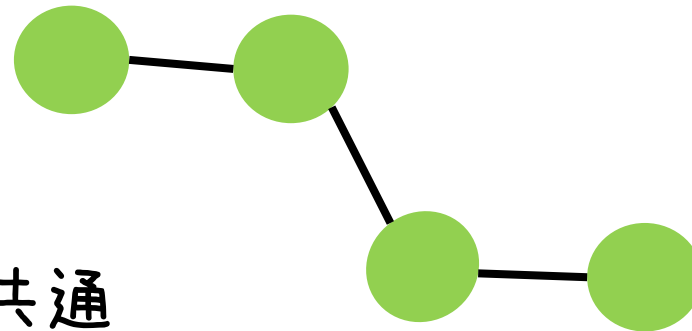
[https://qiita.com/michi\\_wkwk/items/32def413fa0bdd6394f4](https://qiita.com/michi_wkwk/items/32def413fa0bdd6394f4)

手前味噌ですみませんmm

# 設定

基本的な設定はfirst/secondで共通

- size: 32
- negative: 5
- samples: 100
- rho: 0.025
- threads: 4





# 出力結果

各キャラクターが分散表現になっている

```
> df_first[2:11,]
```

		V1	V2	V3	V4	V5	
2	Shanks	-0.084545	0.122299	-0.211796	-0.492336		
3	Lucky_Roux	0.209949	0.108707	-0.187592	0.247794		
4	Luffy	0.056101	-0.099746	-0.171953	-0.164704		
5	Ben_Beckman	0.201852	0.111298	-0.166728	0.276586		
6	Makino	-0.242693	0.344847	0.394989	-0.676295		
7	Higuma	-0.210919	0.298823	0.259405	-0.673945		
8	Woop_Slap	-0.387561	0.410900	0.620744	-0.630082		
9	Yasopp	0.278230	0.183053	-0.109657	0.358208		
10	Lord_of_the_Coast	0.070120	0.330455	0.223572	-0.542739		
11	Coby	0.229623	-0.184727	-0.011546	-0.232342		
		V6	V7	V8	V9	V10	V11
2		-0.208380	-0.420596	-0.321037	-0.176387	-0.452561	-0.122574
3		-0.016941	-0.686511	0.065582	-0.335426	-0.324184	-0.023516

# キャラ同士の類似度を測る

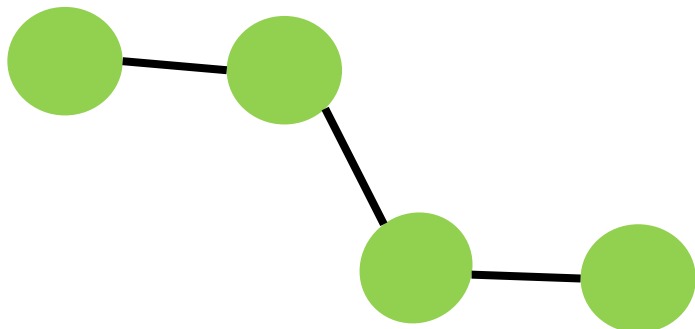
- LINEによりキャラクターを分散表現に直した

- せっかくだから類似度でも出してみる

- 類似度の測り方：

今回はコサイン類似度を採用

gist: <https://gist.github.com/wmichi/6b60b12543bf3205cff32d6adc3995>



# ケースその1: クロオビさん

- 上位には魚人海賊団のメンバーがランクイン  
類似度もなかなか良き  
無駄に一味のメンバーと会っていないからか類似度  
上位の結果はあまり悪くない

```
> coSimTopk(embs, names, 49,10)
```

```
[1] "Kuroobi"
```

	[,1]	[,2]	[,3]
[1,]	"51"	"Nezumi"	"0.813013875430128"
[2,]	"52"	"Hatchan"	"0.653644280179701"
[3,]	"50"	"Chew"	"0.644189616974385"
[4,]	"53"	"Nojiko"	"0.611759614114654"
[5,]	"48"	"Arlong"	"0.580808399701821"
[6,]	"55"	"Mohmoo"	"0.49564903987859"
[7,]	"37"	"Yosaku"	"0.469638086077603"
[8,]	"36"	"Johnny"	"0.458163161507584"
[9,]	"54"	"Genzo"	"0.39981101002403"
[10,]	"56"	"Pudding_Pudding"	"0.381723950990527"

## ケースその2: クロコダイルさん

- クロオビさんほどくっきり分かれていない  
second orderの影響かあってないはずのエースや  
ペルが上位に来ておりやや不満  
ニコ・ロビンやオフィサーエージェントたちには納得

```
> coSimTopk(embs, names, 81,10)
```

```
[1] "Crocodile"
```

	[,1]	[,2]	[,3]
[1,]	"76"	"Nico_Robin"	"0.593823741479889"
[2,]	"89"	"Mr_2"	"0.578630504079855"
[3,]	"93"	"Miss_Double_Finger"	"0.529070610372077"
[4,]	"94"	"Mr_1"	"0.526406298961487"
[5,]	"98"	"Pell"	"0.468265181109235"
[6,]	"90"	"Portgas_D_Ace"	"0.428433709291843"
[7,]	"91"	"Mr_4"	"0.400016522013519"
[8,]	"92"	"Miss_Merry_Christmas"	"0.397764971680522"
[9,]	"29"	"Gaimon"	"0.344942438683463"
[10,]	"79"	"Mr_3"	"0.325342632936231"

## ケースその3: ルフィさん

- 基本的に全員類似度高くない  
色々なキャラとつながっているほど  
うまく当てにくい様子  
麦わら一味が全くいないのはつらい

```
> coSimTopk(embs, names, 3,10)
```

```
[1] "Luffy"
```

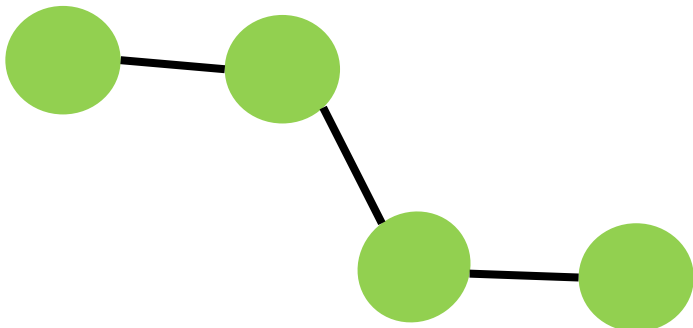
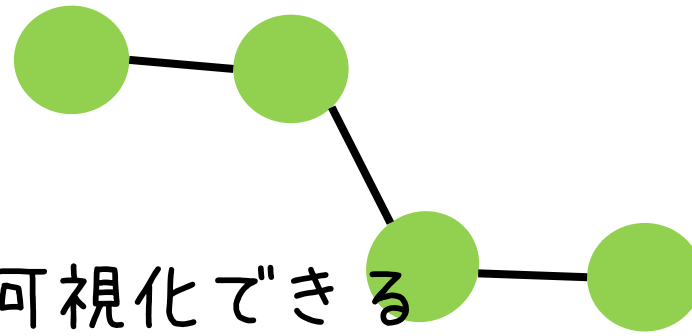
	[,1]	[,2]	[,3]
[1,]	"49"	"Kuroobi"	"0.165657995948109"
[2,]	"34"	"Sham"	"0.164361139258638"
[3,]	"68"	"Ms_Monday"	"0.144310741641242"
[4,]	"52"	"Hatchan"	"0.0746150081010144"
[5,]	"16"	"Kuina"	"0.0574612897896843"
[6,]	"57"	"Bell-mère"	"0.0546663534235397"
[7,]	"17"	"Koushirou"	"0.0485651988296239"
[8,]	"53"	"Nojiko"	"0.0417105970044746"
[9,]	"87"	"Mr_4"	"0.0399578911861013"
[10,]	"56"	"Pudding_Pudding"	"0.0375727427496495"

## まとめ

- statnet使ってグラフの作成・可視化できる  
しかしまだまだうまくいかないことが多い

- グラフ面白い（知ってた）

- やっぱりR楽しい！  
もっと実務で使いたい  
というかもっとRらしいことしたい



enjoy!