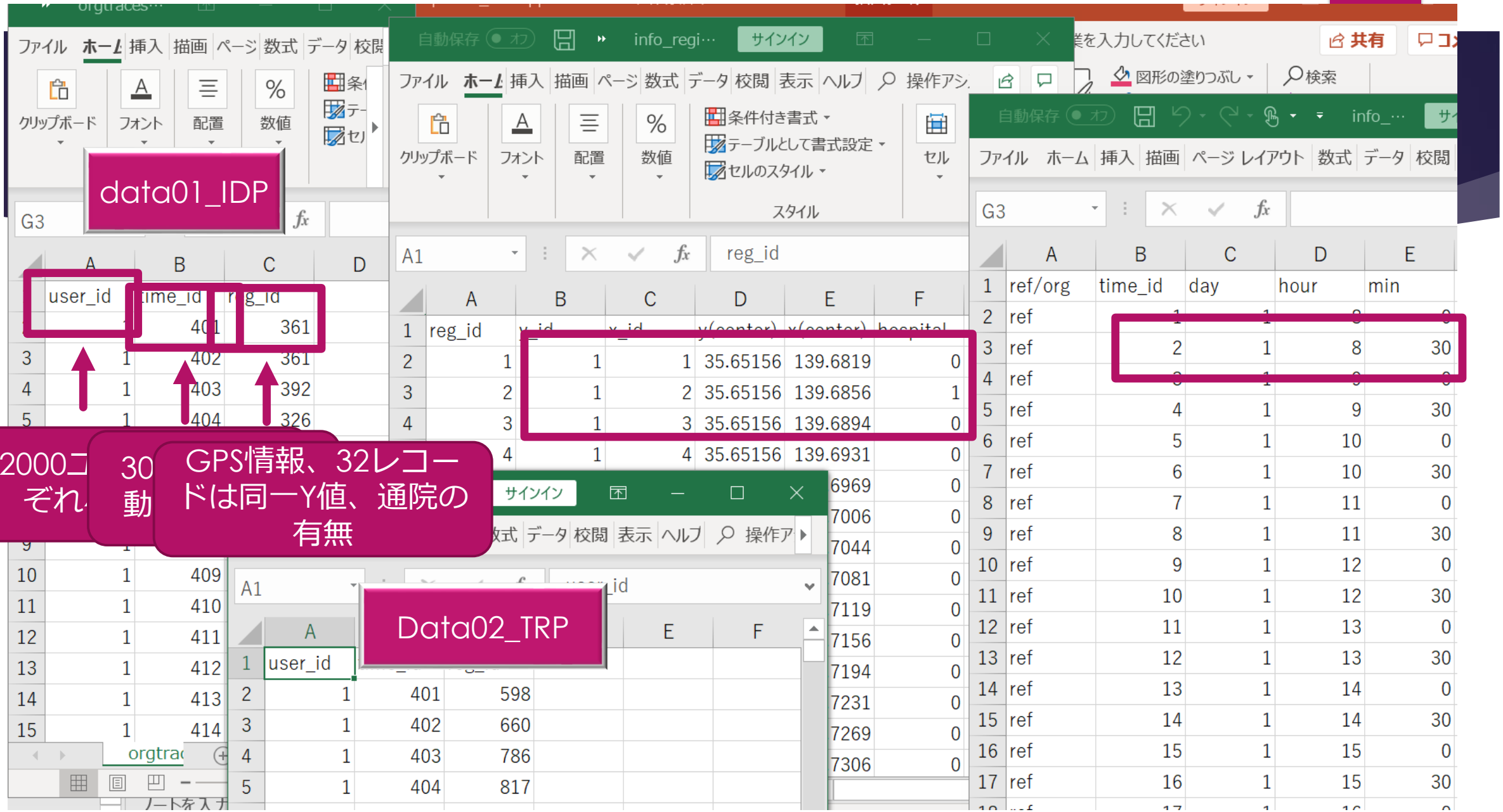
PWS CUP 2019@CSS 2019

チームNO：17

チーム名：東方求敗　　所属：Galaxy



2019/12/2

データの整形と可視化

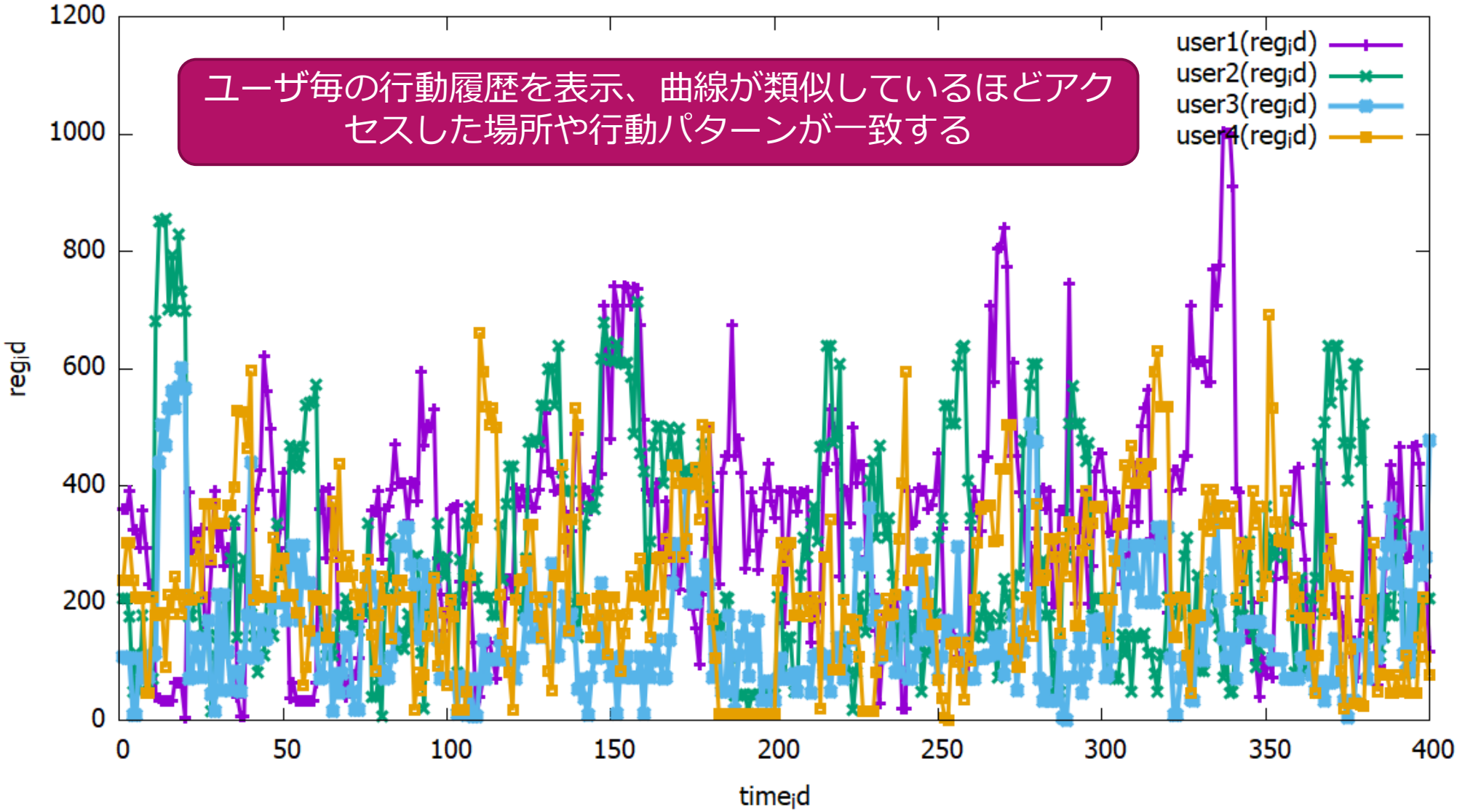
- re

一人400レコードの履歴情報が表示

1	user1	user2	user3	user4	user5	user6	user7	user8	user9	user10	user11	user
3	361	209	107	303	168	70	593	252	471	164	542	
4	392	178	107	303	202	235	584	216	471	134	640	
5	326	50	9	240	233	70	647	187	505	195	443	
6	326	115	9	211	238	66	777	152	534	273	505	
7	294	50	108	209	309	2	905	154	532	176	508	
8	359	179	72	210	238	38	967	249	534	243	507	
		82	108	48	275	38	997	346	504	466	795	
		50	73	48	270	70	874	255	534	431	701	
		71	111	211	304	66	810	285	533	422	703	
		682	116	181	70	66	626	345	504	422	859	
13	38	851	441	182	278	70	811	286	504	70	830	
14	35	851	505	184	281	72	656	224	634	195	700	
15	33	856	469	91	153	72	598	316	830	166	700	
16	33	702	532	216	153	67	599	248	635	390	734	
17	33	794	562	183	151	2	562	117	472	810	734	
18	66	698	532	246	215	70	621	211	246	810	605	
19	66	830	568	215	84	70	685	149	185	779	604	

reg_id2_temp

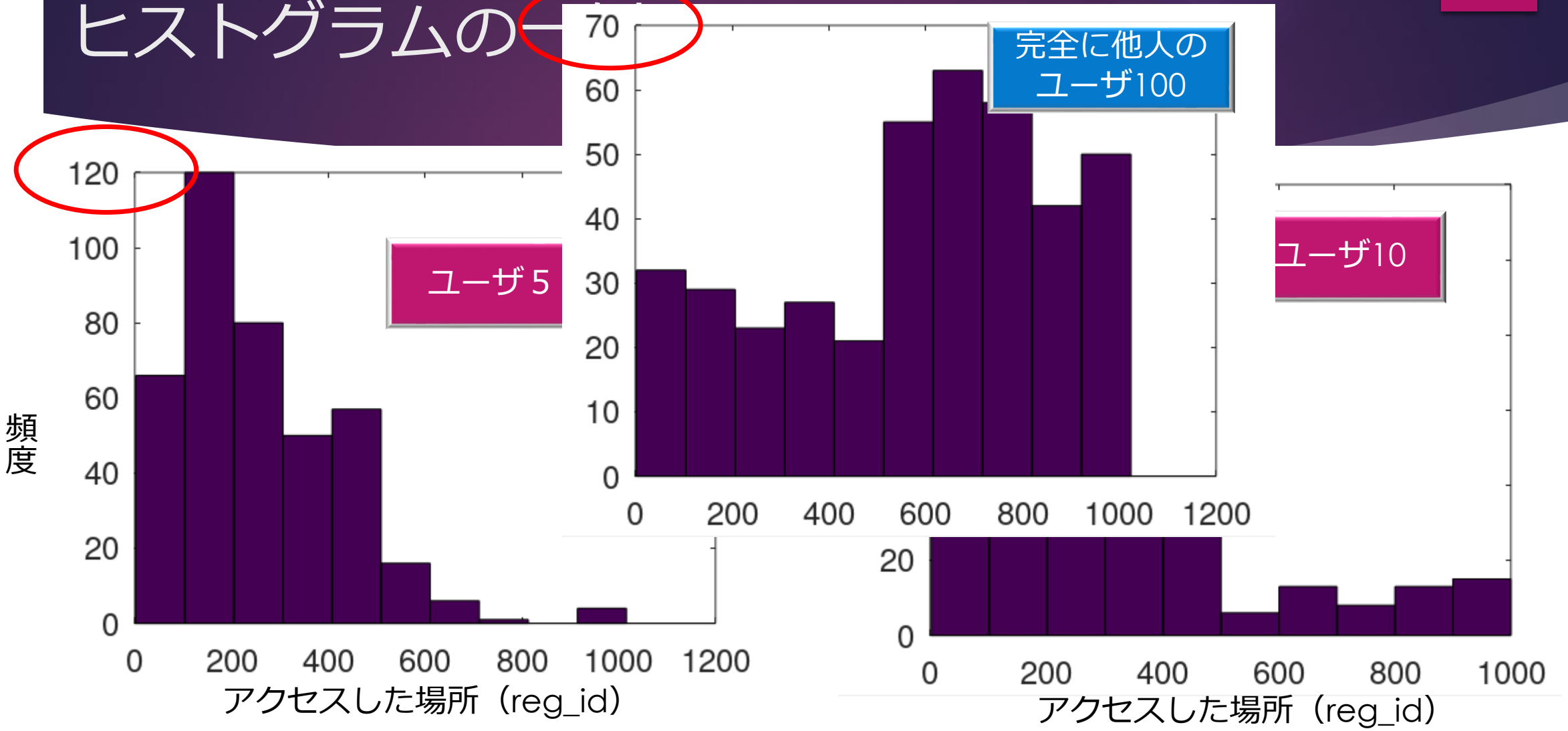
データの可視化（一部のみ図示）



データの分析
（トレース推定に対抗したユーザ類似度推定）

- ヒストグラム分布の類似度を用いたユーザ類似度の算出
  - アクセス履歴が一目瞭然
    - どこにアクセスしたか（reg_id）
    - 同一地点での滞在時間
      - より長時間滞在のユーザがいる
- トレース推定に対抗した匿名に有効だと思われる
- 自己相関及び相互相関も分析したが、使えなかった

ヒストグラムの一



id識別に対抗したユーザ類似度推定（最頻値）

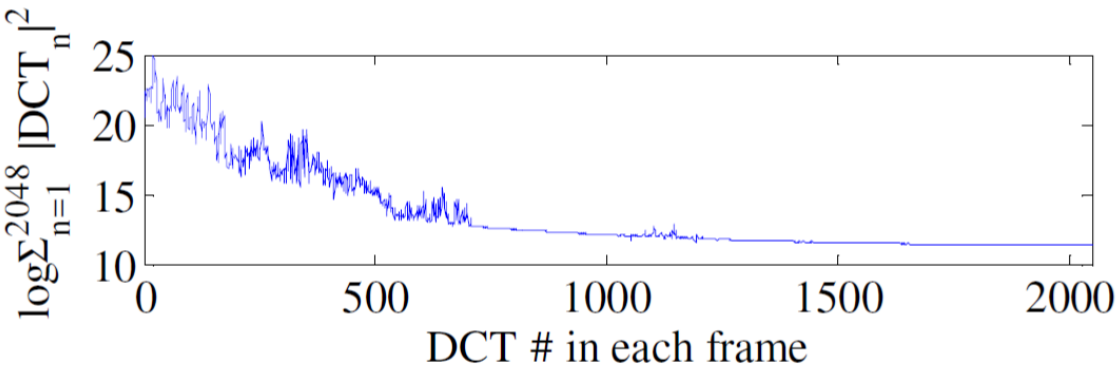
- 同一時点によくアクセスするか、一番長時間に滞在
  - 家族、カップル、趣味が似ているユーザ同志の可能性が高い
  - 病院の有無は考慮していない
  - 今回のデータの一例(reg_idの最頻値)：
    - mode(user5)=169, mode(user10)=167
    - mode(user3)=72, mode(user6)=70
- 距離は半径32以内であれば類似ユーザとみなす
  - Info_region表では32レコードのreg_idが同一Y値を共有しているため
  - 半径指標以外にはマンハッタン距離やユークリッド距離なども有効

匿名加工のメカニズム

- トレース最頻値(mod(reg_id))を用いてユーザの類似度を求める
- 類似ユーザを対象に下記処理を行う
  - メソッド１（予備戦）：reg_idを一般化
    - 利点：ユーザを特定しにくい（安全性が高い）
    - 弱点：有用性が落ちる
    - 予備選で複数の列を使いましたが、有用性が通ってなかった
    - 他のチームも単独列で処理
  - メソッド２（本戦）：トレースreg_idの複製
    - 類似ユーザのreg_idをコピペ
    - 余計な処理をしてしまった（DCT成分に基づいた差分プライバシーLaplace雑音付与の予備処理）

修正整数離散コサイン変換（Modified integerDCT）成分における雑音付加の前処理

- reg_idにノイズ付与
  - ceiling()にて整数への丸め処理
  - ノイズ生成
    - reg_idのDCT成分に基づいて生成
    - Laplace分布に従い、$\lambda = (\log_2^n)/\epsilon$
- 雑音をより小さな値に抑えるのが目的
  - 今回のコンテストでは不適切（reg_idの差が小さい）
  - 結果はA3_kRR.pyに似ている



ノイズ生成アルゴリズム

- xをオリジナルreg_idとし、
- XをDCT成分とする

$$x = \{(x_1), (x_2), \ldots (x_n)\}^T$$
$$X = \{(X_1), (X_2), \ldots (X_n)\}^T$$

$$C_N^{DCT-IV}(i,t) = \sqrt{\frac{2}{N}}\left[\cos\left(\frac{(t+\frac{1}{2})(i+\frac{1}{2})\pi}{N}\right)\right]$$

修正整数DCTの行列分解処理は下記論文を参照：

Xuping Huang, "Watermarking Based Data Spoofing Detection Against Speech Synthesis and Impersonation with Spectral Noise Perturbation", Proc. of IEEE International Conference on Big Data (co-located workshop), pp. 4587-4591, USA, Dec 2018

Input: $D_i = \{x_1, x_2, \ldots, x_n\}, (1 \le i \le n)$
Output:$D_i' = \{x_1', x_2', \ldots, x_n'\}$ in the time domain.
step 1 Transform D_i to generate $dct(D_i)$ to get X_1, $X_2, \ldots, X_n$ in the frequency domain using intDCT.
step 2 Specify privacy utility ϵ (ϵ = 0.1,0.15,0.2...) to generate Laplace noise mechanism $\lambda = \Delta_{1,q}/\varepsilon$ to generate the noise according to Laplace distribution as $r_i = \frac{1}{2\lambda}e^{\frac{-|x|}{\lambda}}$, here $\Delta_{1,q}$ is the mechanism which is related to integer DCT transform that $\Delta_{1,q} = \log_2^n$, here n is the length of data; and then specify δ=0.05. δ here means the error ratio, that 5% of noise value r_i is allowed to be larger then the estimated upper bound.
step 3 Perturbation: $\widetilde{D_i} = X_i + r_i$
step 4 Perform invise DCT $idct(\widetilde{D_i})$ to get $\{x_1', x_2', \ldots, x_n'\}$.