耐巨大性を備えた
表データ分析用コマンド群

(株)ウフル 下野寿之

I. 背景, 概要, どう耐巨大か

5スライド

背景/目的

- 昨今、大規模なデータの分析の需要はあるが、
  未だに十分なツールがあるとは言いがたい(cf.次頁)。
  - Unix, R, Pandas, Excel, SQL, Hadoop を使ってもまだ不十分。
- 新規開発のコマンド群でこの状況を打開する。

- さらには、このソフトウェアの開発の経験により、
  理論的普遍性を持った新規概念も得たい。

既存ソフトの問題点

- ソフトウェア固有の制約により、ハードウェアの性能が活かされることは、現状なかなか難しい。
  - Excelは104.9万行以上を扱えず、black-box 性が高い。
  - Rと Pandas は、データロードに相応の時間がかかる。
  - DB(SQL等)は、テーブル設計の負担が最初に大きい。
  - Unix/Linux のコマンド群でも、実は不十分。
- 現状の既存コマンドを駆使して、データ分析をしても、
  - 簡単と思ったことでも、意外と複雑になる。数個のステップで済まない。
  - デバグ/テストに大きな手間を要し、再現性も損なわれる。
  - 素人目に数時間か数日で終わることが、数週間も数ヶ月もかかる。

新規ソフトで何ができるのか

■ 大規模なデータに対し、下記が容易かつ極めて迅速になる。

1. 初期的なデータの解読 (ファイル/各属性値の様子、足りない情報の発見)
2. 前処理用の分析操作 (要除去/要加工の値の発見、必要な計算機環境の見積もり)
3. 数理モデリングにて適合性の検証 (説明変数に意味があるかなど)
4. 提出レポートの検証 (数が合っているかなど、不自然な点が無いか検証)

■ 実績:

- ✓ 10行程度から数十億行以上の { csv, tsv, 単純な行の} データファイル
- ✓ Perl言語(5.14以上) がインストール可能な普通のPC～大型サーバー

このコマンド群は CUI (Character User Interface) である。

どう ”耐巨大” であるか

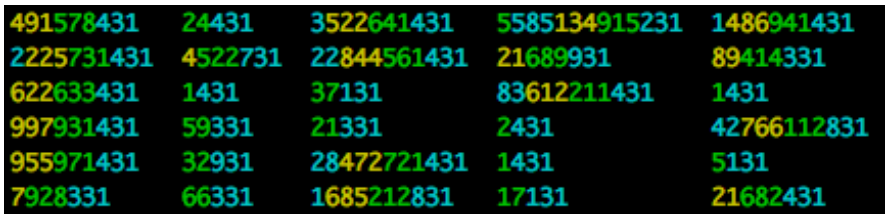
✓ 小規模データは相応に早い。インターフェースは同一で。

✓ 巨大な表データファイルをうまく扱う。

- 大きなデータを操作する上で必要であろう機能は網羅。
- Ctrl + C で途中結果を出力する、などの親切設計。

  長時間起動していて、何も起きず不安になった時に有用。
- Unixコマンドを駆使しても困難なことを実現。

  Unix は、大きなファイルの基礎的なハンドリングが得意であった。



何桁かも分かりにくい数も、
3桁ごとに着色することで、
昨今ありがちな大きな数を
読みやすくする機能もある。

どのように使うのか

0. まず、データを受領/アクセスする。

1. 必要に応じて、ファイル形式を変換。
   - ✓ TSV(タブ文字区切り)が基本、CSVでも一応可能。
   - ✓ 圧縮形式もそのまま使えるようにしている。(読取り/書込みが早くなるので。)

2. 下記のいろいろな操作をする。

   全列の様子を一覧、クロス集計表、値変換(参照/丸め/アフィン)
   列抽出操作(名前や範囲指定)、行抽出操作 (10^N 行目だけ、確率抽出)
   集合4ヶのベン図、列指定のgrep、似た列の検出、
   .. など

3. 得られた知見を保管する。

   → たとえば、エクセルにコピペして、プレゼン用に加工。

どのように作られているか

- 数十個のコマンドがそれぞれの機能を持つ。
  - 利用者の必要に応じて、1個～数個を組み合わせて使う。
  - 各コマンドは Perl で書かれている。各コマンドは単体でも動く。
  - Perl 5.1 場合により Perl 5.14 及び CPAN上モジュールを使う。
- 現場で必要な派生機能の搭載
  - ✓データの先頭行が、列名の並びの場合は適切に考慮。
    - 計数時などは除外する。列ごとの集計時は列名として利用する。
    - 列の指定の際に、列番号で無くて列名指定ができるようにする。
  - ✓ターミナル画面から Excel に容易にコピペが出来る。
  - ✓15桁の数や 50列のデータも見やすくする着色表示機能。
  - ✓アフィン変換/数値丸め/分位点/各種統計的検定/分類器

II.
どのようなコマンドで構成されるか

7スライド

現状のコマンド一覧 (2016-08-18 ; 88個)

```
~/bin4tsv % hg ma | grep -ve '\.' | colorplus -s "^.*/" | column -c150 | column -t
combinator/colkeep   flux_v/sampler      producer/randexp      tabulate/repeater
combinator/colop     flux_v/watching     producer/saikoro      tabulate/shuffler
combinator/headkeep  map/affine          statistics/meanvar    tabulate/spacer
combinator/keypole   map/emptycell       statistics/minmax     tabulate/splitlines
combinator/xcol      map/rounding        statistics/pareto     tabulate/transpose
diff/atime2mtime     map/widths          statistics/quantiles  test/poisson_verynaive
diff/daydiff         map/zeropad         statistics/silhouette uniq11equiv/alluniq
flux_h/fluxsort      primary/checkeof    subtotal/Lcount       uniq11equiv/chunkSHA
flux_h/rightditto    primary/colorplus   subtotal/crosstable   uniq11equiv/kvcmp
flux_h/wideline      primary/cols        subtotal/freq         uniq11equiv/leftcut
flux_time/finch      primary/colsummary  subtotal/keyvalues    uniq11equiv/linedigest
flux_time/madeafter  primary/csv2tsv     subtotal/marginsum    uniq11equiv/similarcols
flux_time/ticktack   primary/dirhier     subtotal/summing      uniq11equiv/uniq-c
flux_time/timeput    primary/dirsizes    subtotal/venn2        uniq11equiv/uniqm
flux_time/usec       primary/expskip     subtotal/venn2-3      util/alterline
flux_v/Lsubstr       primary/listcol     subtotal/venn4        util/denomfind
flux_v/bendline      primary/tsv2csv     subtotal/wc-l         util/diggrp3
flux_v/cat-n         producer/binom      tabulate/colsort      util/koala
flux_v/colgrep       producer/boxmuller  tabulate/headadd      util/maxrss
flux_v/idmaker       producer/cauchy     tabulate/headcolon    util/memlogger
flux_v/inarow        producer/nums       tabulate/pack         util/new-processes
flux_v/leftdrop      producer/poisson    tabulate/poolsort     util/topinfo2
```

提供可能なコマンドを取りだし、コマンド名を目立たせるべく、作成したコマンドcolorplus で正規表現指定で赤く着色を行い、Unixコマンドの column で表形式に整理して、出力をしている。

コマンドの分類

＜開発者視点での分類＞

- ファイル形式の変換
  csv2tsv tsv2csv transpose checkeof
- 行の羅列として処理
  freq **sampler** venn{2,3,**4**} alluniq
  expskip cat-n uniq-c
  linedigest wc-l shuffler
- 各列を意識する処理
  cols listcol **colsummary** colgrep colop
  crosstable similarcols pack
- キーとバリューの2列を操作
  xcol kvcmp keyvalues koala
- 実時間の流れを意識した処理
  ticktack timeput madeafter
  usec memlogger
- 値を変換
  widths affine emptycell rounding zeropad
- R言語を利用する
  silhouette Rscan Rmatrix

＜利用者視点での分類＞

- テーブルの様子を把握する処理
  colsummary **expskip** silhouette wc-l widths
  sampler splitlines similarcols inarow
- 他コマンドと組み合わせて使う
  headkeep memlogger **usec** **colop**
- 複数のファイルが必要なコマンド
  venn{2,3,4} kvcmp **koala** xcol
- 関係データベースとしての正規化
  idmaker rightditto leftcut
- 乱数など数値を生成する
  nums **saikoro** boxmuller poisson
- 統計的集計処理をする
  quantiles minmax poolsort silhouette pareto
  meanvar venn{2,3,4} **denomfind**
- その他
  colorplus dirhier watching

colsummary : 各列の様子を把握する

```
 ~ % cat olympic.tsv | expskip -g | column -t
1   西暦  大会名              金  銀  銅  計  順位
2   1912  ストックホルム      0   0   0   0
3   1920  アントワープ        0   2   0   2   17
5   1928  アムステルダム      2   2   1   5   15
10  1960  ローマ              4   7   7   18  8
20  2004  アテネ              16  9   12  37  5
21  2008  北京                9   6   10  25  8
22  2012  ロンドン            7   14  17  38  11
23  2016  リオデジャネイロ    12  8   21  41  6
 ~ % colsummary -= -n2 olympic.tsv | sed 1d
1   22  1968.182    西暦    1912..2016  1996, 1932  , 1(x22)
2   21  0.000   大会名  アテネ..東京    ロサンゼルス, シドニー  , 2, 1(x20)
3   14  6.455   金  0..16   4, 0    3(x2), 2(x4), 1(x8)
4   11  6.091   銀  0..14   8, 6    5, 4..2(x2), 1(x6)
5   13  7.409   銅  0..21   7, 8    3(x3), 2(x3), 1(x7)
6   15  19.955  計  0..41   18, 25  4, 3, 2(x2), 1(x11)
7   12  10.227  順位    , 3..23 5, 8    4, 3(x2), 2(x3), 1(x6)
```

olympic.tsv の内容：
オリンピックの日本のメダルの個数(1912-2016年) の情報。

各列はタブ文字で区切られている。
マウスカーソルを使って、
Excel に表のまま、コピペが可能。

何通りの異なる値を含むか

平均値

列名

値の範囲

高頻出値

高頻出値と低頻出値の出現回数
同じ出現回数の重なりは x に続けて表す。

表データ解読上の最初の大きなハードルが、ほぼ即座に解決する

crosstable：クロス集計

データから3,4列目を抽出

2列受け取り、頻度をクロス表に

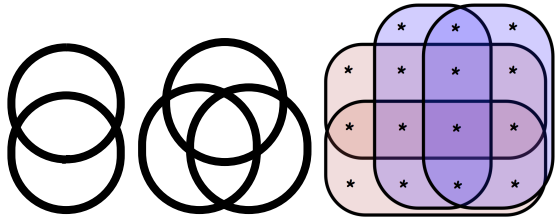
読みやすくするため、0を青く。

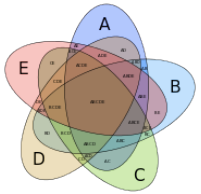
```
 ~ % cols -p3,4 olympic.tsv | crosstable -=q | colorplus -s "0" -b blue
金*銀 0     2     3     4     5     6     7     8     9     10    14
0     2     1     0     0     0     0     0     0     0     0     0
1     0     0     0     0     0     1     0     0     0     0     0
2     0     1     0     0     0     0     0     0     0     0     0
3     0     0     0     0     0     1     0     1     0     0     0
4     0     0     1     0     0     0     1     0     0     1     0
5     0     0     0     0     0     0     0     1     0     0     0
6     0     0     0     1     0     0     0     0     0     0     0
7     0     0     0     0     0     0     1     0     0     0     1
9     0     0     0     0     0     2     0     0     0     0     0
10    0     0     0     0     0     0     0     1     0     0     0
11    0     0     0     0     0     0     1     0     0     0     0
12    0     0     0     0     0     0     0     1     0     0     0
13    0     0     0     0     0     0     0     1     0     0     0
16    0     0     0     0     1     0     0     0     1     0     0
 ~ %
```

crosstable のコマンドで、2列のデータの頻度表を作り、クロス表で表示する。
2列の値の関係や相関の関係を知ることができる。エクセルにコピペ可能。

クロス集計表は、ひとつのデータテーブルで多数作ることになる。

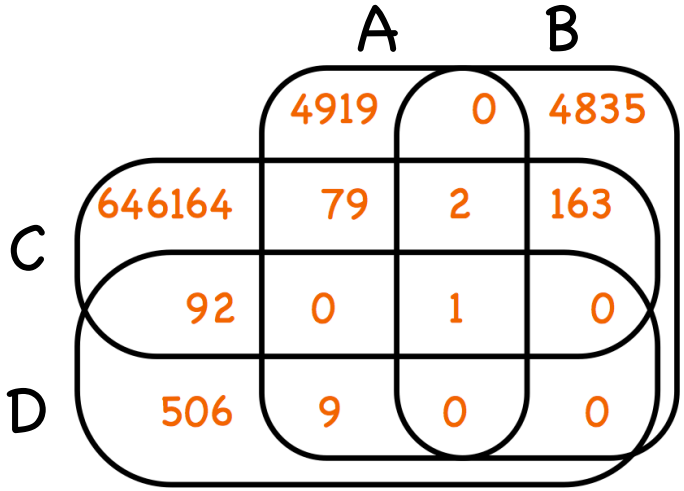
venn4：
ベン図の領域サイズ算出





▲Wikipedia からの引用

- 本格的な作業前に、ベン図で要素数を算出することは、重要。
  どのファイルの組合せで、何個の共通 id があるかを知ると、
  作業上、どんな条件/対象を分析しているのについて、迷いが減る。




```
$ cmp4.pl <( cut -f2 Dove.g19 ) <( cut -f2 TeslaMotors.g19 ) <( cut -f2 nissan.g19 ) <( cut
          AB=00     AB=10     AB=11     AB=01
CD=00         0      4919         0      4835
CD=10    646164        79         2       163
CD=11        92         0         1         0
CD=01       506         0         0         0
          AB=00     AB=10     AB=11     AB=01       A=0       A=1       B=0       B=1     A+B>0       any
CD=00         0      4919         0      4835      4835      4919      4919      4835      9754      9754
CD=10    646164        79         2       163    646327        81    646243       165       244    646408
CD=11        92         0         1         0        92         1        92         1         1        93
CD=01       506         0         0         0       506         0       506         0         0       506
C=0         506      4919         0      4835      5341      4919      5425      4835      9754     10260
C=1      646256        79         3       163    646419        82    646335       166       245    646501
D=0      646164      4998         2      4998    651162      5000    651162      5000      9998    656162
D=1         598         0         1         0       598         1       598         1         1       599
C+D>0    646762        79         3       163    646925        82    646841       166       245    647007
any      646762      4998         3      4998    651760      5001    651760      5001      9999    656761
```

- 既存ソフトだと、上記は、案外とても煩雑であった。
- 空行/先頭行を要素と見なさないオプションなど実装し、利用性を高めた。

インターフェース設計方針

■ コマンド(命令)名は、原則10文字以内の英単語2個。

∵ 英単語1個で言い表される命令は、ほぼ作成され尽くされている。

∵ 単語3個以上からなる命令は、必要な時に思い出しにくい。

■ オプションスイッチをうまく活用する。

-a -b .. -z ：さまざまな付加機能　**-A -B .. -Z** ：動作モードの大きな変更

-= ：先頭行を列名の並びと見なす　**-~**：何かの機能の反転　**-,**：列の区切り文字の指定

■ “迅速性” を特に重要視。

✓ 必要な場面で、存在と名前を思い出しやすいコマンド名。

✓ 初めてでも、うろ覚えでも、何が起こるかすぐ理解できる設計。

✓ Unix/Linux の既存のコマンドと比べても、処理速度が遜色ない。

オプションスイッチの例 [cols : 列選択]

cols は、awk や cut コマンドの列選択機能を高めたモノ。
-d, -p, -h, -t で柔軟に機能が利用可能(組合せも可能)。

- cols –d 6..9 ⇒ 6列目から9列目以外を表示 (delete)
- cols –p 5,9..7 ⇒ 5,9,8,7 列目をこの順に表示 (print)
- cols –h 3 ⇒ 3列目を最左列に移動して全列表示 (head)
- cols –t -2 ⇒ 右から2列目を右に移動して全列を表示 (tail)
- cols -= ～ ⇒ 列の指定を列名に変更。列名は1行目を利用。
- cols -,, ～ ⇒ 区切り文字をタブ文字からコンマに変更。
- cols --help opt ⇒ 各オプションの解説を画面に出力。

III. コマンドの活用例

& 最後のまとめ

2 スライド + 1スライド

活用例-1 : クロス集計 crosstable

1. Twitter の発言で特定の文字列を含むものをインターネット上で収集。
2. ログから日付と時間の2列を取り出し、**crosstableコマンドで集計**。
3. Excelにコピペし条件付き書式で色を付けた。
4. 午前と午後の違い、データ収集サーバーのダウンの日時が明らかとなった。

日付/時間	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2013-12-10												32	157	153	156	139	161	170	182	315	356	276	221	256
2013-12-11	127	93	64	58	39	72	90	144	200	149	170	116	177	144	170	149	156	268	318	253	216	187	203	235
2013-12-12	136	86	68	63	46	61	64	115	177	134	130	121	463	582	423	442	703	1285	1394	1446	1444	1611	2023	1974
2013-12-13	1532	958	634	478	396	403	526	837	1122	1097	1457	1386	1988	1816	2037	3025	3662	4125	4551	5062	4655	4993	5576	2969
2013-12-14	2873	1626	1139	1018	529	682	807	1164	1655	2115	2536	2715	2868	3454	3720	3812	3611	3438	4202	5005	2557	3133	5031	4274
2013-12-15	3617	2232	1694	1330	1051	974	1173	1732	2590	3549	4942	5871	5306	3056	3571	5247	6666	6798	6964	2597	1710	3382	7879	6425
2013-12-16	4623	2741	1631	1304	927	1094	1185	2053	2597	2474	3415	3614	4580	4836	4187	4999	4115	4122	5957	6427	6328	6793	6545	5963
2013-12-17	4045	2405	1545	1189	1007	946	1237	2692	3532	3435	4178	4925	6158	5049	4756	4375	3890	4138	5090	6265	7628	7077	6639	6325
2013-12-18	4458	2807	1713	1404	1131	1051	1347	1954	2299	2624	3412	2707	1962	1773	1910	3851	5263	6359	7177	6583	6936	6644	6234	6119
2013-12-19	4637	3064	1739	1509	1141	1100	1394	2362	2994	3897	3999	4344	5260	6844	5092	4459	5706	6004	7407	3817	1672	4132	6727	5514
2013-12-20	5393	3082	1977	1973	1321	1462	1667	974	1222	1423	1818	2954	5261	4082	4889	6194	6355	6755	9276	6215	6275	6152	5849	5959
2013-12-21	4924	3247	1859	1637	1331	1414	1617	1249	1327	2072	2385	2784	2927	2630	2740	2916	2871	3139	6222	5042	4680	4828	4829	4633
2013-12-22	3838	2702	2073	1836	1417	1186	1427	1442	1633	2043	3376	4120	4346	4159	4473	4759	4450	4773	4962	6876	7718	7327	7472	6768
2013-12-23	5260	3316	2134	1776	1258	1231	1672	1299	894	1023	1195	2402	3516	3584	5172	7778	8700	8331	8030	6035	7712	7669	8084	7450
2013-12-24	7098	4150	2293	1774	1205	1189	1589	1476	1145	1733	2023	2678	4647	3835	4571	6430	6943	7864	8863	5181	6080	7125	8609	6496
2013-12-25	4905	2734	1695	1604	1377	1360	1580	1200	1289	1335	2341	4287	5532	4332	5077	6474	6952	7888	7702	3655	6388	9370	9115	8470
2013-12-26	7468	3877	2269	1959	1370	1398	1908	467			39	140	155	143	131	418	1081	2148	8863	8775	8934	8498	9086	8356
2013-12-27	6419	4015	1833	1627	1421	1361	1590	2511	3768	3996	5245	6196	7894	8098	7373	7497	7296	7324	8437	8120	8059	8384	8435	8277
2013-12-28	7678	4543	2619	2041	1587	1568	1726	2539	3578	4911	5945	7082	7563	7343	7916	7529	7863	8234	8165	8076	8528	8763	8670	7892
2013-12-29	6661	4372	2621	1823	1336	1224	1433	2290	3223	4477	5873	7296	7842	7614	7975	8060	8639	8402	8889	8398	8457	8320	9454	8700
2013-12-30	7607	4656	2765	2100	1528	1440	1635	2511	3997	5140	7044	7658	9435	9147	9147	10058	9956	10529	9852	9662	10644	11448	12848	12136
2013-12-31	10349	6155	3885	2703	1900	1803	2110	3586	5362	8051	10713	12249	13035	13001	13298	15048	15950	17235	17573	15644	16229	15365	14898	17461
2014-01-01	22849	15263	4149	3157	4218	4828	7216	15086	7594	12145	27222	2067	2511	2540	2478	4938	9088	14816	21263	7463	1861	4411	10241	9638
2014-01-02	8627	4964	2749	4744	3729	4168	6328	1915	1342	1889	2848	3233	3197	3612	4528	4449	7701	12565	20139	6596	4637	6998	8677	8949
2014-01-03	6857	5227	6151	3906	2678	2656	2899	2227	1378	2470	2706	3105	3573	3287	3754	7378	10992	11090	10936	5549	6432	8894	9887	8318
2014-01-04	6833	4113	2371	2687	2248	2016	2138	1689	1079	1437	1921	2448	2451	4279	7620	8104	8364	8326	7957	6164	5549	5233	5850	8926
2014-01-05	9757	6208	3823	2627	2073	1782	1854	695	419	511	661	1073	1976	2815	3173	3884	9464	11131	11139	9671	9997	10924	9716	10967
2014-01-06	7623	4818	2873	2159	1656	1480	1733	2948	3538	4550	5638	6605	9118	7973	7552	7691	7392	8170	7714	8308	8974	9368	10272	9641
2014-01-07	7025	4124	2604	1800	1488	1417	1647	1235	1434	2369	3146	3886	5686	6811	6426	6926	7317	8297	8453	6858	9441	9961	9958	8910
2014-01-08	7551	3997	2606	1837	1379	1248	1872	886	647	1022	1465	1828	2463	2138	1909	2163	2298	2832	2944	5583	7953	9266	10316	10737
2014-01-09	9161	4801	2999	2361	1871	1734	2094	3674	5135	5177	6758	7622	11370	10181	10548	12154	13235	13557	11017	4679	5650	6648	7858	7154
2014-01-10	5273	3243	1795	1349	1206	1020	1556	673	526	505	698	827	1039	996	1127	1232	2925	5981	10609	1481				

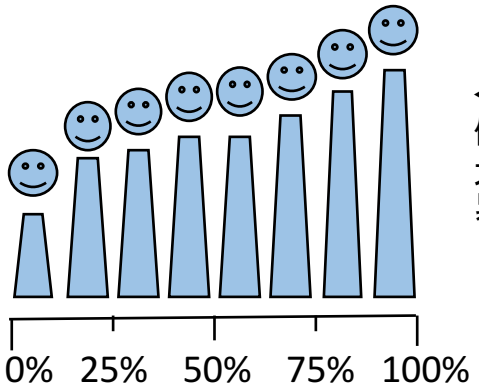
Excel の ピボットは、手間がかかる。さらにこの場合は、300万件強でエクセルの処理上限を超過している。

活用例-2: 累積ヒストグラム silhouette

1. 同じデータの130万超の、各アカウントの follower 数と follow 数の比について、分布を知りたいとする。

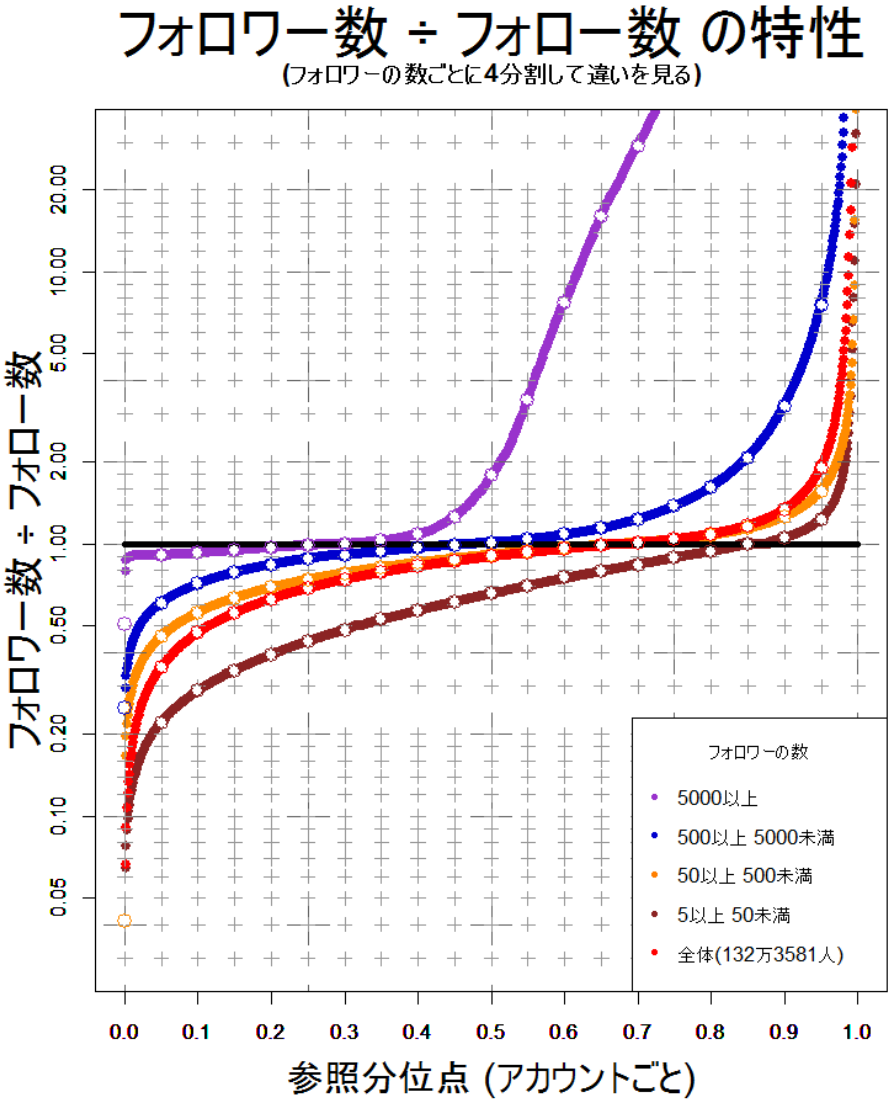
2. ただし、follower の数で4層に分けた。
   5000以上, 残りの内500以上, 残りの内50以上と,その他

3. 各層のデータをsilhouetteコマンドに渡す
   と、各層を色分けした累積ヒストグラムが、pdf ファイルで出力される。



値を棒グラフ化して、
大きさの順に並べると
累積ヒストグラムになる。

4. 5000以上のアカウントにfollowされていると、約半数のアカウントは、follow数の何倍もfollowされる。
   (48%のアカウントは followしている数の2倍以上、他の人からfollowされる。平均的には、95%そうならない。)



特徴や工夫点のまとめ

✓TSVファイルに対する有用な機能群である。

✓データの解読、前処理用の分析に大きな強み。

✓既存のソフトウェアと遜色ない、むしろ凌駕する。

✓Perl があればどこでも動く。古いPCでも。

✓1行目が列名の並びで、2行目からデータの場合も対応。

✓Ctrl+C で途中結果を表示して、続行/停止が選択可能。

■ 一部をGitHub などで GPLライセンス付きで公開中。

■ 使って頂いた感想、専門家の意見は大いに歓迎します。

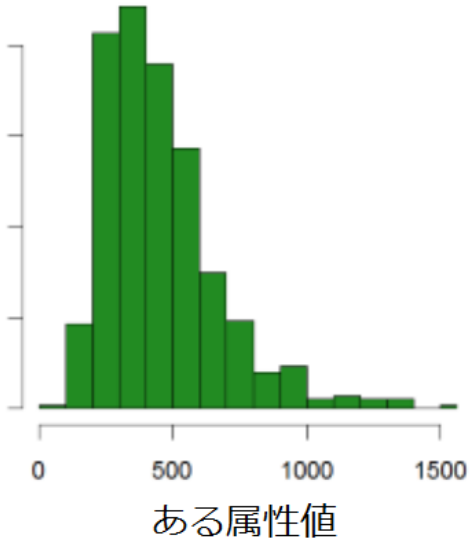
IV.
予備スライド (質疑応答用)

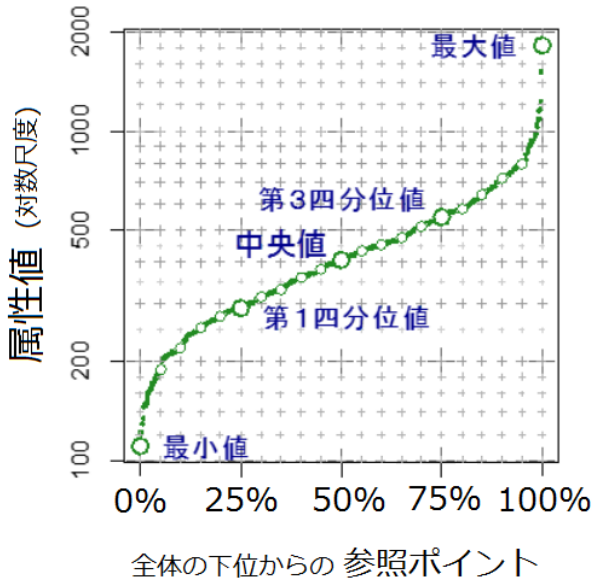
13スライド

補足: CSV 形式と TSV 形式に関して

- CSVファイルの形式について
  - 様々な使われ方が長い間乱立した。
  - 2005年10月に RFC4180 で初めて公式に文書化がされた。
  - CSVファイルは、各列はコンマ(,) で区切られている
  - フィールド(値)の文字列がコンマを含む場合は、フィールドの初めと終わりは、必ずダブルクォーテーションを付加する(☆)。
  - A,”B,C,D”,E の文字列は、A と B,C,D と E の3個のフィールドと見なされる。
- 作成したソフトウェアのCSVファイルに対する対応:
  - 私の提供するコマンド(命令) 群の大部分は、 (☆)の場合に対応してはいない。
  - -,, というオプションスイッチにより、単純に,を区切り文字と見なす。
  - コマンド csv2tsv により TSV形式に変換出来る。
    - ただし、元の CSVファイル中に、タブ文字が出現した場合は厄介である。

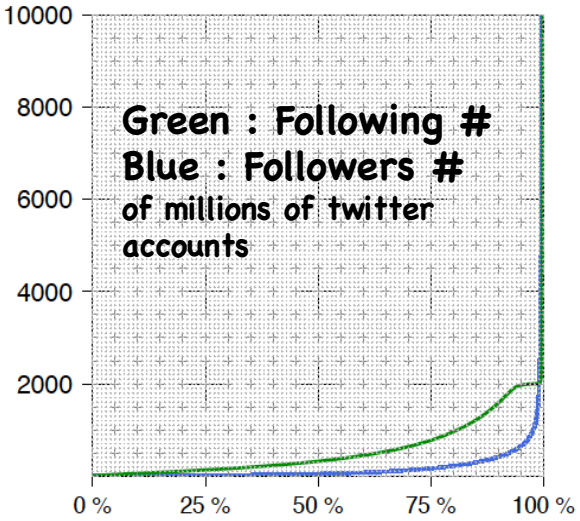
累積ヒストグラムについて

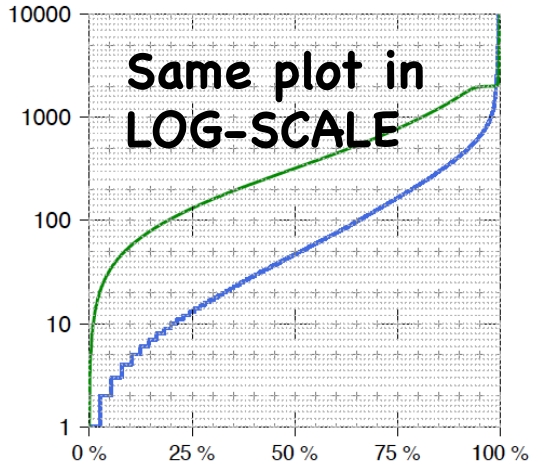


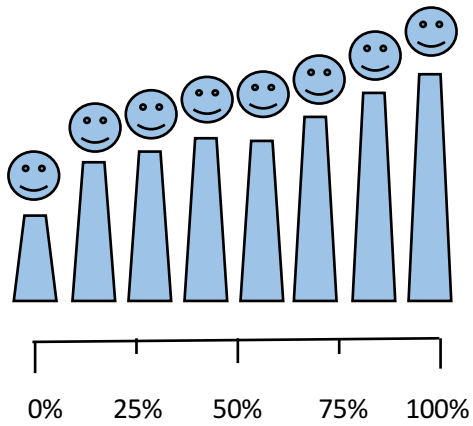


上の段：
ある分布に対する
ヒストグラムと
累積ヒストグラム

下の段：
あるTwitterのアカウントについてのfollow数(緑) と follower 数(青) の累積ヒストグラム。
左が線形スケール、右が対数スケール。







活用例: 累積ヒストグラム silhouette

silhouette コマンド：

- ”背の低い順に左から並べた生徒の様子”から命名。
- 数値データの分布の視覚化に有用である。
- グラフ出力は、R言語に依存。pdfに保存。

ヒストグラムに対する優位性：

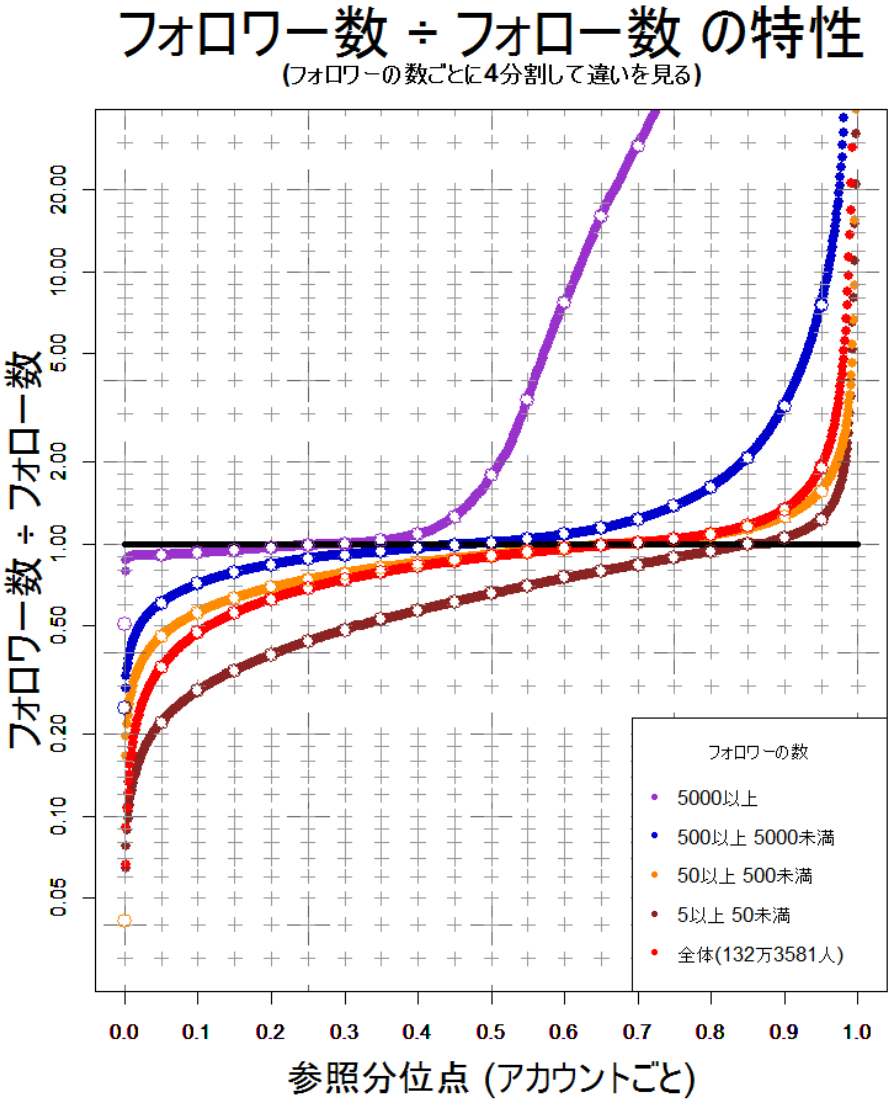
1. 階級幅決定アルゴリズムが不要。対数化も簡単。
2. 中央値や四分位値など分位値の読取りが容易。

右の図について:

- ツイッター130万超のアカウントのデータを分析。
- フォロー数とフォロワー数の関係を調べる目的で作成。
- “ follow返しに頼らないfollowされやすさ “ を調べた。ツイッターでマーケティングをする場面で重要。

- 5千人以上にfollowされていると、48%のアカウントはfollowしている数の2倍以上、他の人からfollowされることが分かった。平均的には、95%そうならない。

補足:
左の図は、分析例を示すために、silhouette コマンドが即座に出来ることとほぼ同じ出力を載せたものである。
この図自体は2年前に、R言語で手間暇を掛けて作った。



なぜ高速動作するのか

- 多くのソフトは、ファイルの先頭から読み始めて最後まで読むことをしてしまう。
- しかし、新作ソフトはファイルを(1行ずつ)逐次必要な所まで読んだら、すぐに処理を行う。
- パイプ/プロセス置換により、前のコマンドの出力を処理中にすぐ次のコマンドで処理可能。
- Perl が十分に早い。RubyやPython よりも。
  ただし、CやJavaなどのコンパイラ言語よりは遅い。

パイプとプロセス置換について

Unix の bash 環境などで、下記の様に使う。

コマンド1 | コマンド2 # ←パイプの使い方
コマンド2 <(コマンド1) # 例 wc <(date)

パイプは、左側のコマンドの実行結果が右側のコマンドに入力として渡される。
プロセス置換で <(..) の場合は、括弧内のコマンドの実行結果を格納したファイルが存在して、そのファイル名が <(..) 全体を置換したかのように振る舞う。

▼ Ctrl + C を押下した時の違い:

パイプの場合は、| の前後の両方のコマンドが、interrupt シグナルを受け取る。 <(〜) の場合は、interrupt シグナルは括弧内に送出されない。

Ctrl+Cを押下した時の動作を設計したコマンドを使う場合に、他のコマンドは Ctrl+C の影響を受けないようにする場合に、このプロセス置換は便利。

開発経験から何を得たか?

- 2単語命名作戦はとても有効だった。
  - 命名候補をいろいろ考える。
  - 機能も広がる。
  - 直接機能を表現しない “奥ゆかしさ” も必要となる。
- 既存のソフト/手法の不足分を多数発見できた。

有用と考えられるコマンド

コマンド名	機能	詳細
freq	頻度集計	何が何個あるかまとめる
crosstable	クロス表	2次元表で頻度,集計などをまとめる
venn2,3,4	ベン図で要素数	複数データのどんな組み合わせで何個キーがあるか
xcol	複数表の組合せ	join(unix)を改良して、直感的に使い易くした
colsummary	各列を要約する	異なる値の個数、値範囲、頻出値、出現回数分布
sampler	確率的抽出	一定値または指定列に比例した値の確率抽出
keyvalues	キー重複度	各キーが何個の異なるバリューを持つかを出力
kvcmp	KV関係の異同	2個のデータ間でキーバリューの関係の異同を見る
colgrep	列指定の検索	Unix の grep では出来ないことをする。
idmaker	識別子の付与	関係DB第二正規化が出来る
denomfind	共通分母の探索	%の値から母数の推定
shuffler	行の順序を乱雑化	sampler とよく一緒に用いる
alluniq	全行が異なるか	重なり数と該当行例も出力が可能
expskip	"対数的に抽出"	指数関数的に間隔を空けて行を抽出
eofcheck	終末文字の検査	改行で終わらないファイルの検出
dirhier	目録構造の解読	目録の階層がどうなっているかが分かる
timeput	各行に時刻記載	一行読み取る毎に現在時刻を付加して出力
usec	実行時間測定	uはマイクロを表す
madeafter	文件の作成後時刻	日時よりも経過時間の方が便利な場合がある
memlogger	使用メモリ測定	指定命令がどれだけのメモリを消費したか知る

有用なコマンド例 [sampler 確率抽出]

```
 ~ % seq 1e6 | sampler -r 3e-5  | paste - - - - - - - - - - - - ; # 1~1e6 の数をそれぞれ 0.00003 の確率で抽出
printed lines: 32/1000000 ; a priori expected line number : 30.00 ; used random seed: 1518477206 (sampler)
16250   37528   68163   84509   90788   114158  139172  150521  157093  212662  241785  312454
315279  348212  366986  421402  442268  475056  576551  577883  730788  751874  771881  799299
804342  814684  851649  863356  913054  954610  962435  991489
 ~ % seq 1e6 | sampler -r 3e-5  | paste - - - - - - - - - - - - ; # 1~1e6 の数をそれぞれ 0.00003 の確率で抽出
printed lines: 31/1000000 ; a priori expected line number : 30.00 ; used random seed: 2352713188 (sampler)
10379   27821   31921   43942   111961  145868  156033  187620  308516  319710  340381  391910
408896  421413  463410  497429  515490  534938  549264  557272  582199  609334  611322  677601
678589  747433  764885  810672  859873  921316  955742
 ~ % seq 1e6 | sampler -r 3e-5 -s 2352713188 | paste - - - - - - - - - - - - ;
printed lines: 31/1000000 ; a priori expected line number : 30.00 ; used random seed: 2352713188 (sampler)
10379   27821   31921   43942   111961  145868  156033  187620  308516  319710  340381  391910
408896  421413  463410  497429  515490  534938  549264  557272  582199  609334  611322  677601
678589  747433  764885  810672  859873  921316  955742
```

sampler で各行の確率抽出を行う。

-r で抽出確率を指定する。再現性確保の為、-s でシードの設定が可能。

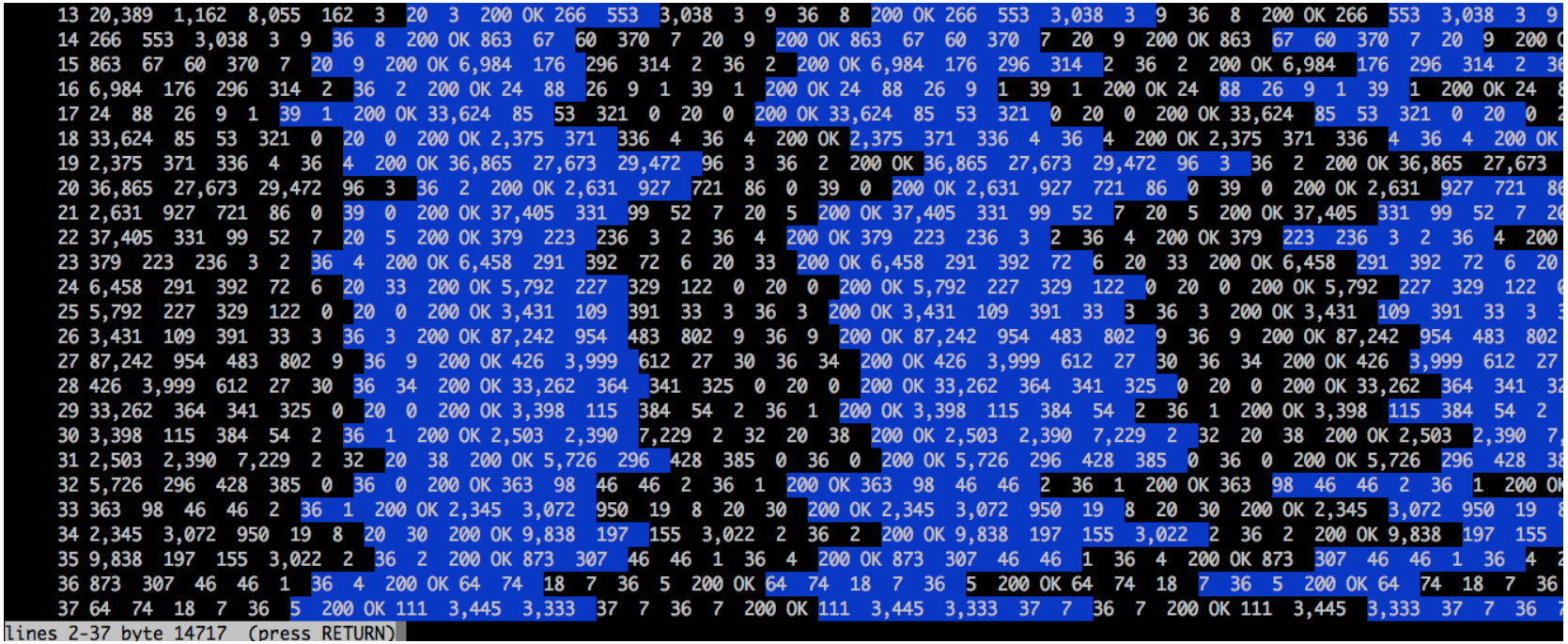
(統計的な処理で必要と考えられる他の機能をいくつか搭載している。)

入力行数と出力行数と抽出行数の事前期待値を、標準エラー出力に出す。
なお、Perlのsrand関数がPerl 5.14 (2011年) を要求する。

上記は、1〜100万の各数を1万分の0.3で確率抽出し、横に12個ずつ並べた。

他、shuffle コマンドで順序も乱雑化できるが、これは行を減らしてから使うこと。

有用なコマンド例 [colorplus 着色]



colorplus で様々な着色を可能とした。

オプション指定で、数値のみ3桁または4桁ごとに着色、
列を5列ごとに背景を着色、指定した正規表現に着色、
さらには、着色の除去などができる。

上記は、5列ごとに背景を青で塗り、less で閲覧している。
詳細未知のデータを与えられて、最初に眺め、意味を把握するときに有用。

有用なコマンド例 [freq 何が何行あるか]

```
$ for i in *.g19 ; do sed 1d $i ; done | gawk -F'\t' '{print $13}' | freqL -nr
661916  200 OK
65      404 Not Found
41      500 read timeout
40      503 Service Temporarily Unavailable
21      500 Can't connect to twitter.com:443
11      500 Can't connect to twitter.com:443 (timeout)
4       500 Internal Server Error
1       500 Server closed connection without sending any data back
```

機能は Unix/Linuxの “ sort l uniq -c ” とほぼ等価。
ただし、高速動作する。
また、度数(頻度)を出力するコマンドの出力順を指定可能。

有用なコマンド例 [xcol：ルックアップ, 別表参照]

xcol –x 列指定 tablefile < datafile

- 2個のファイル table と data がある場合に、
  table を変換表(変換前と変換先の2列から成る)と見なし、
  data の指定列を変換表に従って変換する。

  SQLの join句 と同じ意図を持った機能を持つ。
  Excel の vlookup 関数 にもほぼ近い。
  Unix/Linux にも join コマンドがあるが、使いにくい。

- 下記のオプションがある。
  - 変換前の値を残す指定。それをどの列に残すかの指定。
  - data に対し該当変換が無い行の処理法の指定(残すか消すか)。

他のソフトとの比較

	Excel	R	Pandas	SQL	Hadoop	Unix	新作ソフト
主要用途	表計算	統計解析	行列操作など	DB操作	分散処理のDB	ファイル操作など	初期分析,前処理
利用の手軽さ	◎	○	installに手間?	△	知識要求度高	○だと思う	Unixと同様
透明性	×	高	○?	ベンダー依存	成長途上?	○	○
経年的互換性	△?	△	不明	やや高	不明	やや高	高くしたい
高速性	低	要工夫	○	△	ある用途で とても高速	○	○
巨大データ 親和性	×?	要工夫	高いのかも	要調整	特に高	小範囲のみ	高
データファイルが 多くても使えるか	×	ロードに時間がかかる		--	--	ある範囲で	高いはず
小データで実演 しやすいか	◎	◎	○?	要SQL文理解	意味なし？	◎	◎
列指定の方法	多種	複数あり	Rと同様	列名が必要	SQLと同様	cutかawk ならば 列番号を並べる	最も容易

この表については、本発表者の主観が多分に含まれます。訂正情報等を歓迎します。

特徴や工夫点についてのまとめ (補足スライド用)

特徴

- ✓ TSVファイルに対する有用な機能群である。
- ✓ データの解読、前処理用の分析などに大きな強み。
- ✓ 本格的な統計分析/機械学習の前段階でも役立つ。
- ✓ 既存のソフトウェアと遜色ない、むしろ凌駕する。
- ✓ 各コマンドが1個の (完結する) Perl プログラム。
- ✓ Perl があればどこでも動く。古い小さなPCでも。

工夫点

- ■ TSVデータに対応。1行目が列名の並びでも対応。
- ■ 必要な場面でとにかく使い易くしてある。
  - ➢ コマンド命名の工夫、オプションの活用。
    ← 汎用性のある設計手法であろう。
  - ➢ 途中の計算結果の表示。

現状

- ■ 一部をGitHub などで GPLライセンス付きで公開中。
- ■ CPANで公開もしたいが、どうするかは検討中。
- ■ 応用範囲を広げるべく、機能をさらに拡充したい。
- ■ 目標とする機能群が、一体として完成するのは、
  あと数ヶ月かかる見込み。
- ■ 使って頂いたフィードバック、専門家の意見は
  大いに歓迎します。
- ■ 今後の拡張について
  - 画像・動画・音声は対象とする予定はなし。
  - 分類器・ネットワーク・統計学的検定は検討中。
  - メモリ制約、CPU使用率制約は検討するかも知れない。