Literate programming contoh implementasinya dalam pembelajaran sains

- Author: Dasapta Erwin Irawan[1], Cut Novianti Rachmi[2], dan Sandi Herho[1]
- Affiliasi:

    1. Institut Teknologi Bandung
    2. Universitas Padjadjaran
- Email: dasaptaerwin@outlook.co.id

- Description: This document is a manuscript for the SNIPS 2018 ITB
- Keywords: literate programming, reproducible research

Abstrak

Kode sering diajarkan dalam pembelajaran sains. Selain dapat menuntun alur pikir, kode juga melatih ingatan serta kreativitas. Saat menulis laporan, sering kali kita menggabungkan narasi dengan kode dan luarannya menggunakan teknik salin tempel. Cara ini tidak praktis. *Literate programming* dapat membantu Anda menulis laporan (atau artikel) dengan menggabungkan narasi-kode-luaran secara otomatis. Dalam artikel ini, kami menayangkan aplikasi literate programming menggunakan Bahasa Python dengan Jupyter Notebook untuk melakukan analisis statistik sederhana terhadap data kualitas air tanah di Bandung. Dari hasil yang didapatkan, metode ini dapat digunakan untuk menjelaskan tahapan analisis sejak membuka data, memanipulasi data untuk menyiapkan data, visualisasi, hingga analisisnya secara naratif yang menyatu dengan kode perintah dan luaran prosesnya.

Abstract

[translate] Kode sering diajarkan dalam pembelajaran sains. Selain dapat menuntun alur pikir, kode juga melatih ingatan serta kreativitas. Saat menulis laporan, sering kali kita menggabungkan narasi dengan kode dan luarannya menggunakan teknik salin tempel. Cara ini tidak praktis. *Literate programming* dapat membantu Anda menulis laporan (atau artikel) dengan menggabungkan narasi-kode-luaran secara otomatis. Dalam artikel ini, saya menayangkan aplikasi literate programming menggunakan Bahasa Python dengan Jupyter Notebook untuk melakukan analisis statistik sederhana terhadap data kualitas air tanah di Bandung. Dari hasil yang didapatkan, metode ini dapat digunakan untuk menjelaskan tahapan analisis sejak membuka data, memanipulasi data untuk menyiapkan data, visualisasi, hingga analisisnya secara naratif yang menyatu dengan kode perintah dan luaran prosesnya.

Pendahuluan

Kode sering diajarkan dalam pembelajaran sains. Selain dapat menuntun alur pikir, kode juga melatih ingatan serta kreativitas (Filiz, 2015 (https://www.sciencedirect.com/science/article/pii/S0747563215004288)). Saat menulis laporan, sering kali kita menggabungkan narasi dengan kode dan luarannya menggunakan teknik salin tempel. Cara ini tidak praktis. *Literate programming* (LP) dapat membantu Anda menulis laporan (atau artikel) dengan menggabungkan narasi-kode-luaran secara otomatis. Tujuan dari artikel ini adalah untuk menjelaskan konsep LP, bagaimana metodenya, serta aplikasinya dengan studi kasus analisis statistik sederhana untuk data kualitas air tanah di Kota Bandung.

Sekilas tentang literate programming

Literate programming (LP) dikenalkan oleh David Knuth. Bila dilacak dokumentasinya, maka ide ini pertama kali terbit sebagai makalah dalam jurnal (Knuth, 1984 (https://academic.oup.com/comjnl/article/27/2/97/343244)). Dalam dokumen itu, Knuth menyampaikan bahwa dunia pemrograman secara umum memerlukan suatu cara agar kode program yang sama dapat diulang oleh orang lain, sekarang konsep ini diberi nama *reproducible research*/RR (riset yang dapat diulang). Definisi dari RR dijelaskan dengan sangat baik oleh ROpensci (2018) (http://ropensci.github.io/reproducibility-guide/sections/introduction/), yakni suatu upaya yang bertujuan agar pihak lain dapat mengulang setiap prosedur riset yang telah kita lakukan. Tidak hanya mengulang, tapi lebih jauh lagi, yakni dapat menggunakan ulang (*reuse*) dan memodifikasinya untuk keperluan lain (*remix*), atau bahkan mengoreksi alur yang kita buat (*contribute*) (Peng, 2011 (http://science.sciencemag.org/content/334/6060/1226)),(Sandve et al., 2013 (https://www.ncbi.nlm.nih.gov/pubmed/24204232)).

Metode

Dalam artikel ini kami akan menggunakan Jupyter Notebook (JN) sebuah aplikasi LP yang awalnya dikembangkan untuk Bahasa Pemrograman Python. Bagi pengguna Linux dan MacOSX, Python (https://www.python.org/) adalah bawaan sistem operasi (SO), walaupun demikian, direkomendasikan untuk memeriksa versi Pythonnya dan memperbaruinya. JN berjalan dengan baik pada Python versi 2.7 atau 3.x. Bagi pengguna SO Windows, Anda perlu menginstalasi Python secara terpisah. Distribusi Continuum Anaconda (https://anaconda.org/) adalah yang kami rekomendasikan untuk SO Linux, MacOSX, dan Windows karena kemudahannya dan kelengkapan panduan instalasinya. Seluruh hasil komputasi di sini tidak dilakukan secara salin-tempel (*copy-paste*) tetapi adalah hasil dari proses pengkodean (*coding*).

Kemudian kami akan membuat analisis statistik sederhana berdasarkan data terbuka kualitas air sumur di Semarang dari penelitian sebelumnya (Triadi et al. 2016 (https://doi.pangaea.de/10.1594/PANGAEA.862987)). Data, kode, dan narasi akan dikombinasikan ke dalam artikel ini secara langsung untuk mendemonstrasikan LP. Analisis statistik yang dilakukan: mendeskripsikan data, membuat histogram untuk melihat distribusi data, dan membuat beberapa grafik x-y untuk melihat beberapa korelasi yang mungkin muncul diantara parameter yang diukur.

Contoh aplikasi dalam analisis statistik sederhana

Deskripsi data

Kami akan mendeskripsikan data menggunakan fungsi dalam *library* `Pandas`. Langkah-langkahnya adalah: (1) memuat `Pandas` ke memori, (2) membuka data menggunakan fungsi `pd.read_csv` dan menyimpannya sebagai `dataframe` bernama `data`, (3) kemudian menampilkannya sebagai tabel (10 baris pertama). Lihat Tabel 1 di bawah ini.

In [64]:

```
import pandas as pd # langkah 1
data = pd.read_csv('data.csv', sep='\t') # langkah 2
print('Tabel 1 Data')
data.head(n=10) # langkah 3
```

```
Tabel 1 Data
```

Out[64]:

	Event	Area	date_time	lat	long	utm_east	utm_north	utm_
0	SB_185	PT. Ny.Meneer-1	1992	-6.95854	110.45625	439936	9230800	49M
1	SB_273	PT. INAN	1992	-6.98295	110.44322	438500	9228100	49M
2	SB_283	Obs. SD Kuningan	1992	-6.96437	110.41608	435500	9230150	49M
3	SB_271	PT. Sango Keramik	1992	-6.98006	110.31332	424150	9228400	49M
4	SB_270	Dolog Mangkang	1992	-6.97099	110.29341	421950	9229400	49M
5	SB_278	Hotel Santika	1992	-6.99333	110.42918	436950	9226950	49M
6	SB_325	PT Wahyu Utomo	1992	-6.99323	110.34770	427950	9226950	49M
7	SB_190	PT. Gentong Gotri	1992	-6.98338	110.42874	436900	9228050	49M
8	SB_256	Tambakharjo, Tugu	1992	-6.97832	110.36447	429800	9228600	49M
9	SB_206	Tambak Udang, Mangkang	1992	-6.95020	110.30385	423100	9231700	49M

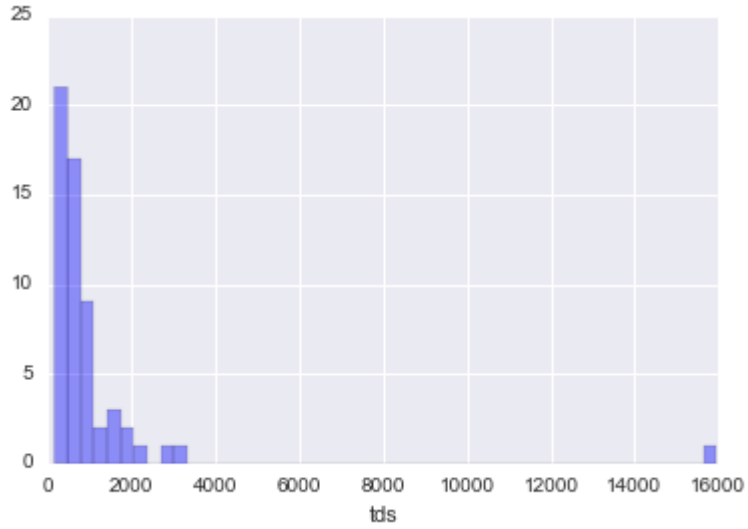
10 rows × 25 columns

Kemudian kita akan membuat grafik histogram parameter zat padat terlarut (*total dissolved solids*, kolom `tds`), kandungan klor (kolom `cl`), kalsium (kolom `ca`) dan elevasi sumur (kolom `elevation`). Untuk membuat grafik, kami menggunakan *library* `Seaborn` dan memuatnya ke memori dengan nama `sns`.

In [47]:

```
%matplotlib inline # perintah untuk memuat grafik langsung ke dalam notebook ini
 (inline plotting)
import numpy as np # memuat library numpy (numeric python) sebagai np
import seaborn as sns # memuat library seaborn sebagai sns
sns.distplot(data['tds'], kde=False, color='b')
print('Gambar 1 Plot histogram TDS (ppm)')
```

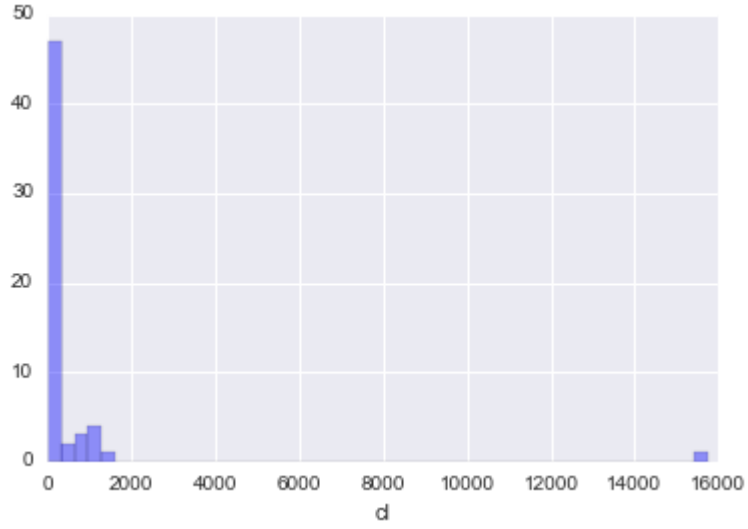
Gambar 1 Plot histogram TDS



In [53]:

```
sns.distplot(data['cl'], kde=False, color='b')
print('Gambar 2 Plot histogram klor (ppm)')
```

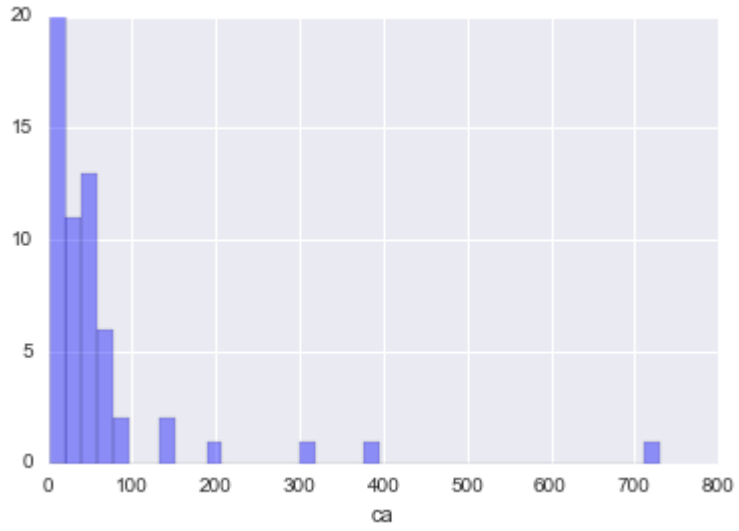
Gambar 2 Plot histogram klor



In [54]:

```
sns.distplot(data['ca'], kde=False, color='b')
print('Gambar 2 Plot histogram kalsium (ppm)')
```

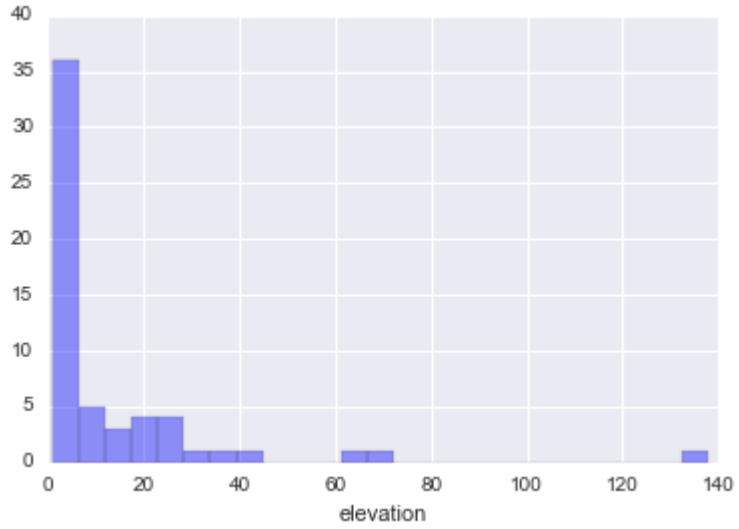
```
Gambar 2 Plot histogram kalsium
```



In [56]:

```
sns.distplot(data['elevation'], kde=False, color='b')
print('Gambar 3 Plot histogram elevasi posisi sumur (ppm)')
```

```
Gambar 3 Plot histogram elevasi posisi sumur
```



Korelasi

Selanjutnya kami akan mencoba membuat tabel matriks korelasi. Sebelumnya kolom dalam `dataframe` yang berisi teks (string) perlu dikeluarkan dan disimpan sebagai `dataframe` baru bernama d dengan fungsi `dataframe.loc()` (lihat Tabel 2). Kemudian kami buat tabel berikutnya berisi beberapa ukuran statistik dasarnya menggunakan fungsi `dataframe.describe()`(lihat Tabel 3). Pada baris berikutnya, kami menayangkan matriks korelasi dengan fungsi `dataframe.corr()` (lihat Tabel 4).

In [67]:

```
d = data.loc[:,'tds':'hco3']
d.head(n=10)
print('Tabel 2 Dataframe yang baru')
```

Out[67]:

	tds	ph	ec	k	ca	mg	na	so4	cl	hco3
0	424	7.85	704	6.5	10.0	6.0	120.0	92.5	35.7	222.0
1	964	7.31	1372	5.0	8.7	8.0	150.0	65.5	136.4	171.4
2	531	7.24	759	10.0	3.7	29.0	148.0	71.7	37.2	379.1
3	279	6.94	408	6.0	41.2	12.0	30.0	10.9	13.9	261.7
4	381	7.23	557	9.0	45.0	16.4	50.0	38.3	57.0	231.9
5	901	7.15	1341	10.0	53.7	29.4	160.0	14.9	224.0	355.9
6	262	6.87	373	6.0	46.2	13.6	50.0	70.4	12.4	268.4
7	669	7.24	1020	15.0	36.2	34.6	134.0	15.5	173.6	301.3
8	1767	7.64	2790	19.0	10.5	60.0	320.0	22.0	664.0	117.2
9	541	7.48	790	12.0	2.5	9.0	164.0	123.5	111.6	169.6

In [71]:

```
print('Tabel 3 Statistik dasar dataframe d' )
d.describe()
```

Tabel 3 Statistik dasar dataframe d

Out[71]:

	tds	ph	ec	k	ca	mg
count	58.000000	58.000000	58.000000	58.000000	58.000000	58.000000
mean	1040.517241	7.453103	1561.189655	10.650000	63.577759	53.807241
std	2080.304636	0.735015	3116.152286	8.477343	111.128017	236.734501
min	152.000000	6.420000	226.000000	1.500000	2.500000	0.500000
25%	425.000000	7.142500	672.250000	6.100000	17.600000	7.125000
50%	556.000000	7.305000	875.000000	8.250000	37.000000	15.550000
75%	934.000000	7.670000	1347.750000	11.900000	54.300000	28.425000
max	15947.000000	11.600000	23900.000000	49.000000	730.400000	1811.700000

In [72]:

```
print('Tabel 4 Matriks korelasi dataframe d' )
d.corr()
```

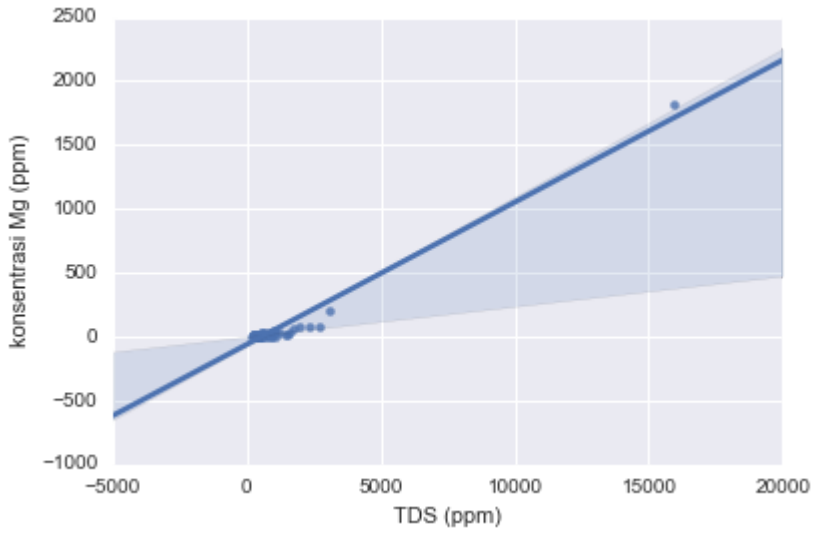
Out[72]:

	tds	ph	ec	k	ca	mg	na	sc
tds	1.000000	-0.020906	0.999770	0.268030	0.865850	0.978942	0.987034	0.(
ph	-0.020906	1.000000	-0.021676	-0.129942	-0.112700	-0.043160	-0.001383	0.(
ec	0.999770	-0.021676	1.000000	0.269267	0.865724	0.979582	0.986649	0.(
k	0.268030	-0.129942	0.269267	1.000000	0.370504	0.161970	0.178290	-0
ca	0.865850	-0.112700	0.865724	0.370504	1.000000	0.828171	0.830466	-0
mg	0.978942	-0.043160	0.979582	0.161970	0.828171	1.000000	0.991248	-0
na	0.987034	-0.001383	0.986649	0.178290	0.830466	0.991248	1.000000	0.(
so4	0.026610	0.034096	0.022284	-0.054717	-0.052327	-0.028558	0.012856	1.(
cl	0.989866	-0.029729	0.989664	0.195601	0.851196	0.994212	0.997764	-0
hco3	0.890621	0.018041	0.892535	0.227762	0.840162	0.906087	0.897149	-0

Dari tabel 4 di atas, dapat Anda lihat bahwa nilai TDS memiliki korelasi yang kuat dengan kandungan Ca, Mg, Na, Cl, dan HCO3, tapi memiliki korelasi lemah dengan K. Untuk memperlihatkan korelasi tersebut, kami buatkan grafik x-y antara TDS dengan Mg dan TDS dengan K (lihat Gambar 4 dan 5 berikut ini). Kedua plot menggunakan fungsi `sns.regplot()` dengan beberapa kode tambahan untuk pengaturan penamaan sumbu.

In [82]:

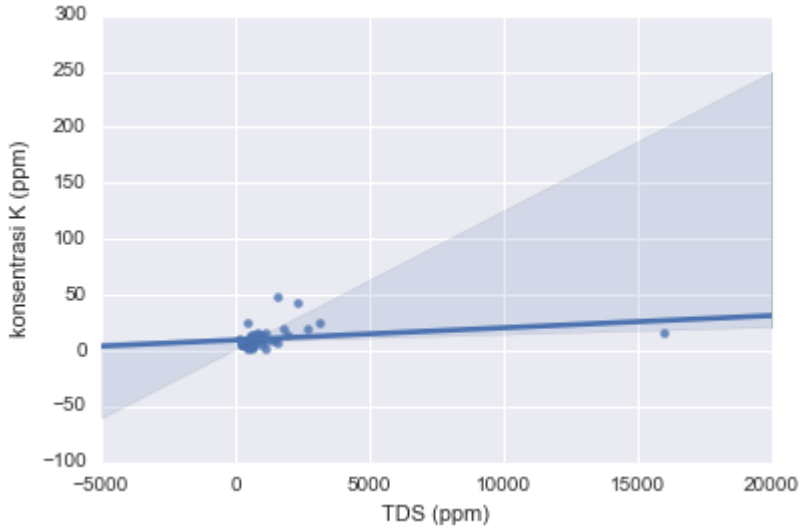
```
ax = sns.regplot(x='tds', y='mg', data=d)
ax.set_xlabel("TDS (ppm)")
ax.set_ylabel("konsentrasi Mg (ppm)")
plt.show()
print('Gambar 4 Plot TDS terhadap Mg')
```



Gambar 4 Plot TDS terhadap Mg

In [83]:

```
ax = sns.regplot(x='tds', y='k', data=d)
ax.set_xlabel("TDS (ppm)")
ax.set_ylabel("konsentrasi K (ppm)")
plt.show()
print('Gambar 5 Plot TDS terhadap K')
```



Gambar 5 Plot TDS terhadap K

Beberapa catatan

Jika diperhatikan demo di atas, dapat kita lihat bahwa tahapan-tahapan dalam analisis statistik dapat dijelaskan secara naratif, bukan dengan komentar pada baris kode yang pendek-pendek. Biasanya kita memberikan komentar atau penjelasan dengan diawali karakter #. Dokumen JN ini, pada waktunya, dapat diekspor sebagai dokumen berformat PDF atau HTML hanya dengan memilih opsi pada menu `File` di atas. Hasilnya adalah satu file PDF atau HTML yang siap tayang berisi perintah kode, luarannya berupa tabel atau grafik, serta penjelasannya. Bahkan makalah ini pun dikonsep secara langsung dalam JN.

Dengan menggunakan JN ini, proses belajar mengajar dapat menjadi lebih mudah. Pengajar hanya perlu memberikan file JN ini, disertai data mentahnya. Para siswa akan menyalin file-file yang diperlukan ke dalam folder kerjanya, maka mereka akan dapat menjalankan perintah dan menghasilkan luaran yang persis sama dengan tayangan pengajar. Piranti lunak yang diperlukanpun sangat fungsional, serta seluruhnya gratis dan *open source*, sehingga dapat menghemat biaya penyelenggaraan pendidikan.

Daftar pustaka

1. D. E. Knuth (1984) Literate Programming, The Computer Journal, Volume 27, Issue 2, Pages 97–111, url: https://doi.org/10.1093/comjnl/27.2.97 (https://doi.org/10.1093/comjnl/27.2.97).
2. ROpensci (2018) Reproducibility guide, ROpensci blog, url: http://ropensci.github.io/reproducibility-guide/sections/introduction/ (http://ropensci.github.io/reproducibility-guide/sections/introduction/).
3. Peng, R.D. (2011) Reproducible Research in Computational Science, Sciencemag blog, url: http://science.sciencemag.org/content/334/6060/1226 (http://science.sciencemag.org/content/334/6060/1226).
4. Sandve GK, Nekrutenko A, Taylor J, Hovig E. (2013) Ten simple rules for reproducible computational research, PLOS Computational Biology, url:https://www.ncbi.nlm.nih.gov/pubmed/24204232 (https://www.ncbi.nlm.nih.gov/pubmed/24204232).
5. Kalelioglu, F (2015) A new way of teaching programming skills to K-12 students, Computers in Human Behavior, Volume 52, November 2015, Pages 200-210, url: https://www.sciencedirect.com/science/article/pii/S0747563215004288 (https://www.sciencedirect.com/science/article/pii/S0747563215004288).
6. Putranto, Thomas Triadi; Rüde, Thomas; Irawan, Dasapta Erwin (2016): Hydrochemical properties of groundwater samples in Semarang area, Java Island, Indonesia (1992, 1993, 2003, 2006, and 2007). PANGAEA, https://doi.org/10.1594/PANGAEA.862987 (https://doi.org/10.1594/PANGAEA.862987).