Amelia

Jan Borowski, Piotr Fic, Filip Chrzuszcz

29 03 2020

Podstawowe działanie pakietu

Z jednego datasetu zawierającego braki tworzymy kilka innych gdzie dane, które znajdowały się w pierwotnej ramce pozostają takie same. A brakujące wartości zostają uzupełnione zgodnie z rozkładem danych, różne datasety zapewniają oddanie niepewności.

Amelia II's EMB algorytm

Założenia:
1.Dane (D) pochodzą z wielowymiarowego rozkładu normalnego:
$D \sim N_k(\mu, \Sigma)$
Dotyczy to istniejących obserwacji (D^{obs}) ale i brakujących (D^{obs})
2.Braki danych są typu *MAR* (missing at random) lub *MCAR* (missing completely at random) co można sformalizować jako:
Niech M bedzie macierzą gdzie $m_{ij} = 1$ jeżeli $D_{ij} \in D^{obs}$ lub $m_{ij} = 0$ w przeciwnym przypadku, wtedy:
$p(M \mid D^{obs}) = p(M \mid D)$

Podstway działania algroytmu EM

Interesuje nas parametr $\theta = (\mu, \Sigma)$ odpowiadjącemu rozkładowi D, naszymi danym są D^{obs} i M.
Można więc zapisać gęstość $p(M, D^{obs} \mid \theta)$ korzystając z założenia drugiego mamy:
$p(M, D^{obs} \mid \theta) = p(M \mid D^{obs})p(D^{obs} \mid \theta)$.
Omijając przekształcenia otrzymujemy:
$L(\theta; D^{obs}) = p(D^{obs} \mid \theta) = \int p(D^{obs}, D^{mis} \mid \theta) dD^{mis}$
Gdzie $L(\theta)$ - "likelihood". Zdefiniujmy jescze $l = log(L(\theta))$.

Algorytm EM

Zdefinujmy:
$Q(\theta \mid \theta^{(t)}) = E_{D^{mis} \mid D^{obs}, \theta^{(t)}}[log(L(\theta; D^{obs}, D^{mis})]$
którę można przedstawić jako:
$Q(\theta \mid \theta^{(t)}) = \int l(\theta; D^{obs}, D^{mis}) p(D^{mis} \mid D^{obs}; \theta_t) dD^{mis}$.
Działanie algorytmu:
Zaczynamy od losowego wyznaczenia θ_0 potem w t kroku:

Maksymalizujemy $Q(\theta \mid \theta_t)$ i nadpisujemy parametr:
$\theta_{t+1} = arg \max_{\theta} Q(\theta \mid \theta_t)$
Powtarzamy aż do zbiegania.

Idea Bootstrap

Polega na estymowaniu rozkładu zmiennych w następujący sposób.
1.Wybieramy n elemetnową próbkę ze zwracaniem M razy.

2.Dystrybuante empiryczną liczymy jako: $F_n(y) = \frac{1}{n}\sum_{i=1}^{n} I(Y_i \leq y)$ (gdzie I - indykator)
Po dokładną impelemntację bootstrapu do algortymu EM odsyłam do: https://www.nstac.go.jp/services/society_paper/27_06_01_Paper.pdf.

Podstawowa funkcja pakietu Amelia

amelia(x, m = 5, p2s = 1, frontend = FALSE, idvars = NULL, ts = NULL, cs = NULL, polytime = NULL, splinetime = NULL, intercs = FALSE, lags = NULL, leads = NULL, startvals = 0, tolerance = 1e-04, logs = NULL, sqrts = NULL, lgstc = NULL, noms = NULL, ords = NULL, incheck = TRUE, collect = FALSE, arglist = NULL, empri = NULL, priors = NULL, autopri = 0.05, emburn = c(0, 0), bounds = NULL, max.resample = 100, overimp = NULL, boot.type = "ordinary", parallel = c("no", "multicore", "snow"), ncpus = getOption("amelia.ncpus", 1L), cl = NULL, ...).

Postaram się omówi istotne parametry tej funkcij:
x-ramka danych lub macierz do imputacij (możliwe jest też użycie obiektów typu "amelia" lub "molist")
m-ilość datesetów które chcemy otrzymać
p2s-sposób wyświetlania(0-brak,1-podstawowe,2-szczegółowe)

Parametry CD.

idvars-nazwy lub numery niezdefiniowanych zmiennych
ts- nazwy lub numery kolumn z szeregami czasowymi
splinetime- im wyższa wartość parametru tym szybciej zbiega algorytm kosztem dokładności
polytime - jak wyżej (mniejszy wpływ na dokładnoś ale pozwala ustawić tylko wartości 0-3)
startvals- macierz reprezntujaca które obserwacjie mają być usunietę (startvals=1 nie usuwa obserwacij)
tolerence- dopuszczalna toleracija zbierzności w algorytmie EM
logs - nazwy lub numery kolunm do transformacij logarytmicznej

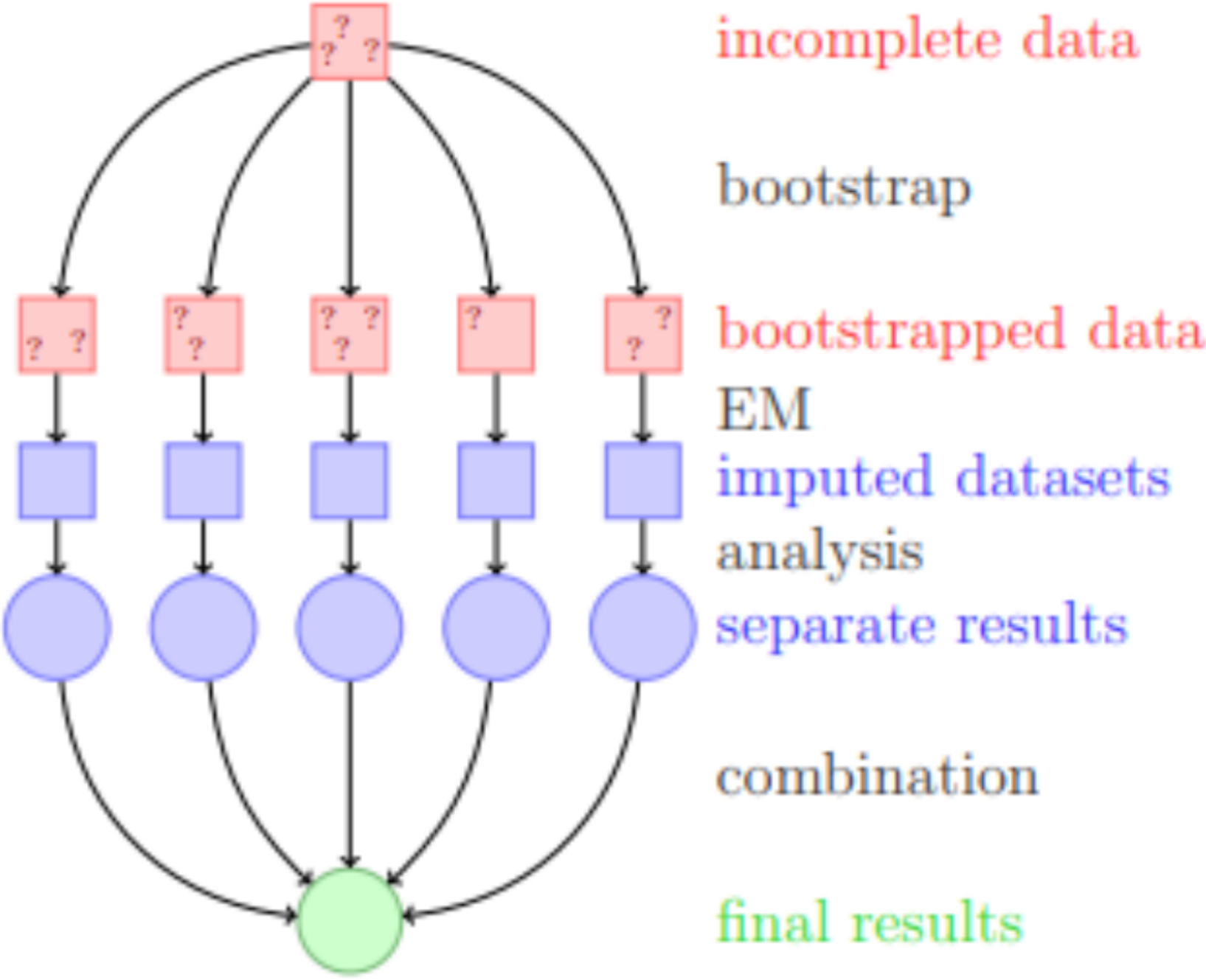
CD

sqrt- nazwy lub numery kolumn do transformacij przez pierwiastek (kolumny muszą zawierać wartowści dodatnie)
lgstc - nazwy lub numery kolumn które powinny zostać przekształcone przez regresję logistyczną (z 0-1 do danych proporcjonalnych)
noms - nazwy lub numery kolumn z danaymi kategorycznymi ords - jak wyżej ale zmienne z oczywistą kolejnością
empir - wartośc decydująca o kowarjancij zmiennych powinna mieć wartość około 0.5-1 górny ograniczenime jest 10
priors - pozwala podać informacjie o brakujących zmiennych w następującym formacie:
one.prior <-c(row,column,mean,standard deviation)
lub
one.prior <-c(row,column,minimum,maximum,confidence).

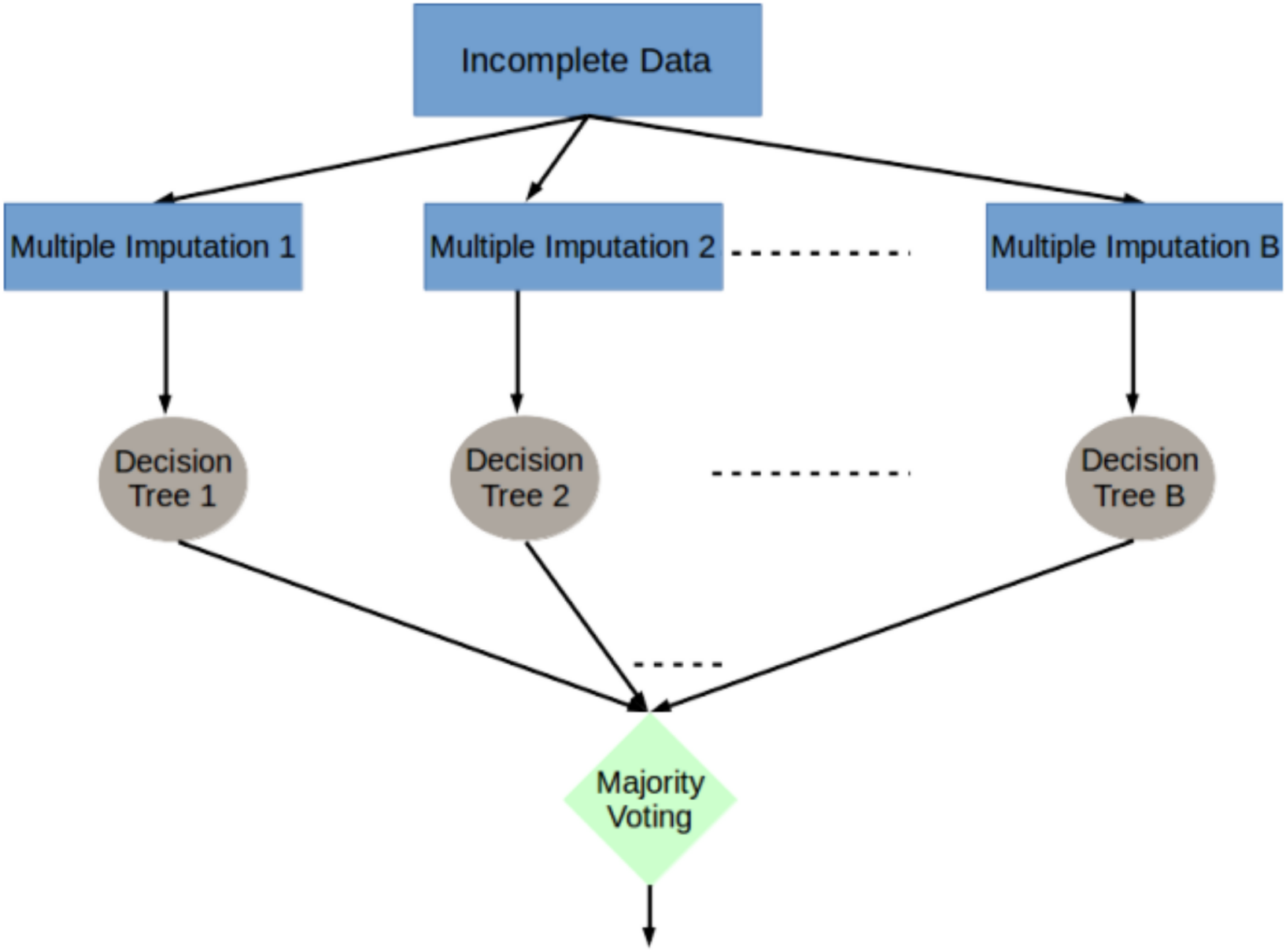
CD

autopri - automatycznie tworzy powyższą macież (wartości 0-1) gdzie 0 oznacza wyłaczenie tej funkji
bounds - pozwala wyznaczyć grancę implementowanych warotści w następującej formie:
c(column.number,lower.bound,upper.bound)
overimp - pozwala zdecydować które wartości powinny być nadpisane pomimio tego ,że znajdują się w ramce w następującej fromie:
c(row,column)

Schemat imputacji algorytmem EMB



Trenowanie modeli po imputacji



Modelowanie i predykcja

b - liczba imputacji

1. Przygotowanie zbioru treningowego i testowego
2. Wytrenowanie ***b*** modeli na ***b*** uzupełnionych zbiorach treningowych
3. Predykcja ***b*** wytrenowanymi modelami na ***b*** uzupełnionych zbiorach testowych
4. Finalna predykcja
   - klasyfikacja: głosowanie większościowe
   - regresja: średnia z predykcji

Pozostałe funkcje dostępne w pakiecie

a.out - output funkcji **amelia**()

1. Zapis output-u do plików csv
   - **write.amelia**(obj=a.out, file.stem="outdata")
2. Połączenie kilku output-ów
   - **ameliabind**(a.out1, a.out2, ...)
3. Wizualizacja braków
   - **missmap**(a.out)

Analiza imputacji

1. Porównanie gęstości zmiennych po imputacji
   - **plot**(a.out, which.vars=1:5)
   - **compare.density**(a.out, var="")
2. Overimputing: przyjęcie kolejno wartości za brakujące, następnie porównanie imputowanych wartości do rzeczywistych
   - **overimpute**(a.out, var="")
3. Wizualizacja zbieżności algorytmu EM
   - **disperse**(a.out)

Transformacje i modele

1. Transformacje zmiennych
   - **transform**(a.out, new_col_name=log(col_name))
2. Pakiet Zelig umożliwia automatyczną implementację niektórych modeli na output-cie funkcji **amelia** np. regresję logistyczną
   - **zelig**(vote ~ age+race, model="logit", data=a.out)
3. Połączenie wyników z modeli za pomocą średniej
   - **mi.meld**()

Przyklady

Zdecydowaliśmy się przedstawić działanie pakietu na przykładzie zbioru danych Titanic. Celem jest przewidzenie, która z osób przeżyła słynna katastrofę.

NA

Dane zawierają dość sporo NA, więc Amelia będzie miała na czym pracować

```
summary(data)
```

```
##      pclass      survived      sex                 age
##  Min.   :1.000   0:809    Min.   :0.000   Min.   : 0.166
##  1st Qu.:2.000   1:500    1st Qu.:0.000   1st Qu.:21.000
##  Median :3.000            Median :1.000   Median :28.000
##  Mean   :2.295            Mean   :0.644   Mean   :29.881
##  3rd Qu.:3.000            3rd Qu.:1.000   3rd Qu.:39.000
##  Max.   :3.000            Max.   :1.000   Max.   :80.000
##                                           NA's   :263
##      sibsp            parch            fare              e
##  Min.   :0.0000   Min.   :0.000   Min.   :  0.000   Min.
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:  7.896   1st
##  Median :0.0000   Median :0.000   Median : 14.454   Medi
##  Mean   :0.4989   Mean   :0.385   Mean   : 33.295   Mean
##  3rd Qu.:1.0000   3rd Qu.:0.000   3rd Qu.: 31.275   3rd
```

CD

Po standardowym podzieleniu danych na część treningową oraz testową pora na pokazanie faktycznego działania pakietu. Tak jak pisaliśmy wcześniej Amelia tworzy kilka zbiorów z zaimputowanymi danymi i na nich nalezy wytrenować algorytmy, a wyniki zestackować.

Użycie funkcji

Przykład konkretnego użycia i wykorzystania głównej funkcji z pakietu

```
imputed_train<- amelia(train,m=5,noms = noms,ords=ords,idva
```

```
## -- Imputation 1 --
##
##    1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 1
##   21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 3
##   41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 5
##   61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 7
##   81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 9
##   101 102 103 104 105 106 107 108 109 110 111 112 113
##
## -- Imputation 2 --
##
##    1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 1
##   21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 3
##   41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 5
```

CD

Do wytrenowania algorytmów użyliśmy drzew losowych z pakietu mlr3. Na kazdym z zestwów danych otrzymaliśmy podobny wynik, jednakże dzięki połączeniu ich poprzez tak zwany Voting Classifier powinno udać się jeszcze poprawić wyniki.

```
## classif.auc classif.auc classif.auc classif.auc classif
##   0.8110871   0.8091631   0.8093735   0.8094336   0.810
```

CD

Zbieramy nasze wyniki i wybieramy te przewidziane wartości które większość z naszych algorytmów przewidziała.

```
##            Reference
## Prediction  0  1
##          0 98 74
##          1 57 33
```

Podsumowanie

Jak widać imputacja z użyciem tego pakietu zajmuje trochę czasu, ze względu na konieczność stackowania wyników. Jednakże sama imputacja przebiegała bardzo szybko i bezproblemowo, warto rozwazyć użycie tego pakietu w przyszłych projektach.

Żródła

[1]https:
//www.nstac.go.jp/services/society_paper/27_06_01_Paper.pdf
[2]https://r.iq.harvard.edu/docs/amelia/amelia.pdf?fbclid=
IwAR2HBnifAAs8EfX9WL6tPyD3mvQorI-
IZWWIOG2YKyXSMLph9Tdvob-c_bc
[3]https://cran.r-project.org/web/packages/Amelia/Amelia.pdf/
[4]https://arxiv.org/pdf/1802.00154.pdf