Rを利用した経済統計データの取得(1)

資　料

Rを利用した経済統計データの取得 (1)

遠　山　弘　徳

Ⅰ　はじめに

「経済学研究にとってどのプログラミング言語がベストか―Julia，Matlab，PythonそれともRか？」．本タイトルのエッセイにおいてAguirre氏とDanielsson氏は経済学研究に一般的に利用されているJulia，Matlab，Pyhon，およびRの４言語を比較しています．評価の基準は「言語の特質」,「実行速度」,「扱えるデータ形式」,「ライブラリー」,「グラフィックス」および「使いやすさ」の６点にもとづいています．結論としては，両氏は総合評価でJuliaを推奨しています．Juliaの若さ（わずか８年の歴史）と将来性を買ってのことと思われます．負の遺産もなく，コードは他のものよりもクリーンで高速で，エラーが出にくいと評価しています[1]．他方，もっともネガティブな評価はMatlabに与えられています．

しかし，個別の評価項目をみると，扱えるデータ形式，ライブラリーおよびグラフィックスの３点においてもっとも高い評価が与えられているのはRです．グラフ作成については，４つの言語すべてが高品質のグラフィックスを出力することができるものの，Rは他の３つの言語よりも頭一つ抜きん出ていると評価されています．実際，ニューヨークタイムズやBBCもグラフィックスの作成にRを使っています．また，ライブラリの豊富さについても，Rを強く支持しています．

データ処理を行うさい，テキストファイル，CSVファイル，Excel，SQLデータベース，独自のデータフォーマットなど，多くの異なるフォーマットから大規模なデータを読み込んだり，書き込んだりしなければならないことがよくあります．Aguirre氏とDanielsson氏は，そうした異なるデータ形式を扱うにはRが最も適していると評価しています．つまり，Rがあれば，ほぼすべてのデータ形式を読み込むことができます．今では多くの国際機関や研究組織によって多くのデータベースが提供されていますが，Rを利用すれば，データ分析までシームレスに―つまりRを離れることなしに―そうしたデータを簡単に読み込むことができます[2]．

[1] たとえば，こうした理由からPerla et al.（2020）の数量経済学プロジェクト（https://julia.quantecon.org）でもJuliaが採用されているようです．

[2] タディ（2020）も，Rの強みを豊富なパッケージを提供するそのエコシステムと，多くのデータ形式が読み込み可能であることを評価しています．

実証研究においてデータの収集と編集はとても重要です．分析データは個人もしくは組織で一次データを収集することが望ましいことは言うまでもありません．しかし，これには時間も費用もかかりますし，マンパワーも必要となります．また，収集したデータが，かならずしも意図する分析に適したものとは言えないかもしれません．むしろ，そうしたケースが多いと言えるでしょう．データの収集と編集はじっさいの分析よりも，骨の折れる作業かもしれません．

Rには，国際機関，研究組織の提供するデータベースからデータを取得するパッケージが多く開発されています．そうしたパッケージを利用すれば，骨の折れる作業がかなり軽減されますし，シームレスかつ再現可能な形でのデータの読み込みが可能となります．そこで本資料では，Rパッケージを使った，経済分析において頻繁に利用される国際機関のデータの取得方法と簡単な利用方法を紹介します．

本資料では，最初に，さまざまな形式のデータの読み込み方法を紹介します（第Ⅱ節）．次に，マクロおよびミクロ経済統計データベースとそのデータ取得に開発されたRパッケージを紹介します（第Ⅲ節以降）．

Ⅱ　RStudioのProject管理とさまざまなデータ形式の読み込み方法

Rを利用し，レポートを作成するとき，データ，スクリプト，グラフなどが増えて行きます．そうしたデータ等が増えていくと，ファイルを整理し，まとめていくのが大変になって行きます．RStudioにはデータ管理はもちろんのことプロジェクト全体を管理するのに便利なProjectと呼ばれる機能が用意されています．そこで最初に，データを扱う上で有益なRStudioのProject機能を紹介します．

その上で，さまざまな形式のデータの読み込み方法を解説します．上述のように，Rはほぼすべてのデータ形式を読み込むことができます．以下で紹介するデータベースの提供するデータ形式はほとんどがcsv形式ですが，パネルデータの扱い易さからStata形式，あるいはもっとも一般的なExcel形式で提供されている場合もあります．そこでここではさまざまなデータ形式をRに読み込む方法も紹介します．

Ⅱ．1　RStudioのProjectの利用

Project機能を利用すると，プロジェクトごとに，作業ディレクトリの場所，実行の履歴やオブジェクト（変数や関数），スクリプトなどが記憶されます．作業の再開時に，プロジェクトを開くとすぐにプロジェクトごとの以前の環境が利用でき便利です．

それではRStudioでProjectを作成してみましょう．Projectの作成は，図１のように，メニュー

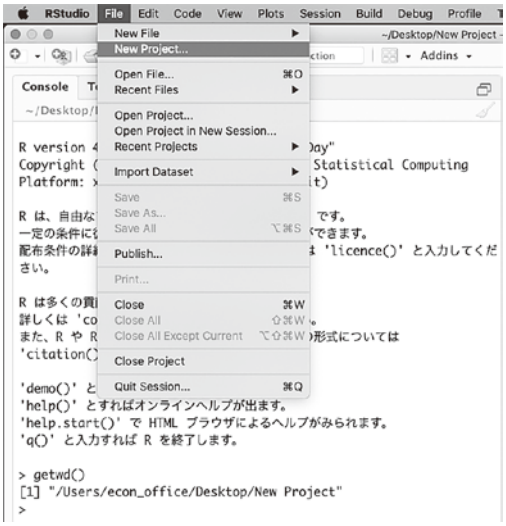


図１　プロジェクト作成手順１

から，File > New Project…をクリックします．

　すると，図２のようなダイアログが現れます．既存のディレクトリを利用する場合はExisting Directoryを選択しますが，ここで新しいディレクトリを作成しますので，New Directoryを選択します．

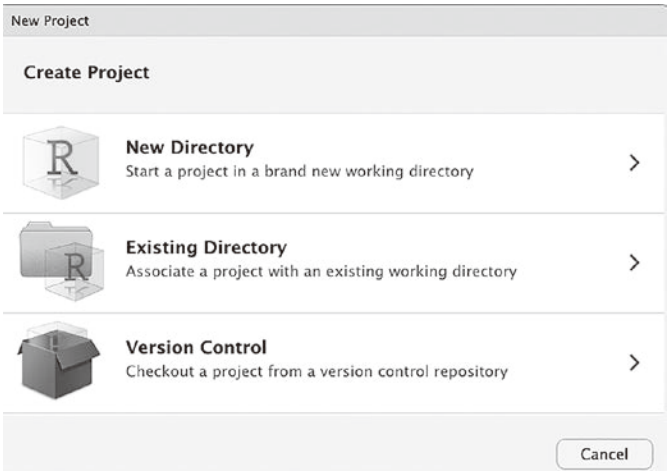


図２　プロジェクト作成手順２

　ついで図３のダイアログが開きますので，New Projectを選択してください．

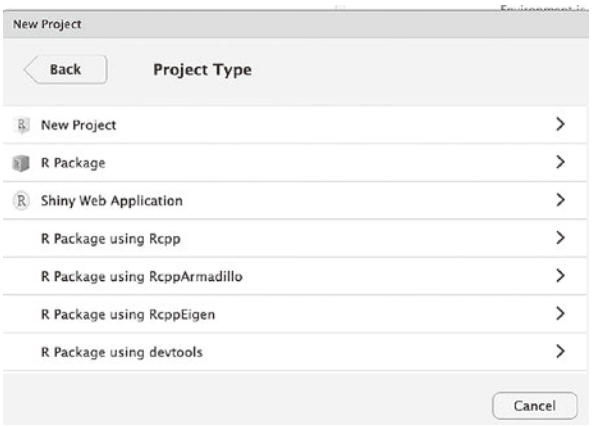


図3　プロジェクト作成手順3

最後に，新しいプロジェクトNew Projectのためのディレクトリ名を尋ねるダイアログが開きます．

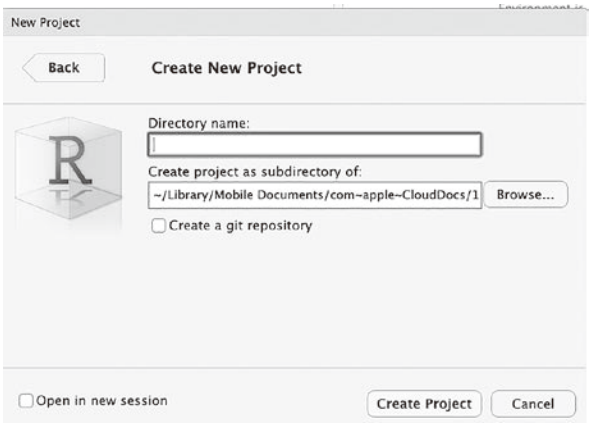


図4　プロジェクト作成手順4

［Directory name］の空欄のボックスに，新規プロジェクトを作成するディレクトリ名を入力します．ここではとりあえずProject_１という名前にしておきます．［Browse…］ボタンをクリックして，プロジェクトを置きたいディレクトリの場所を指定します．すると，新たに作ったプロジェクト用のディレクトリの下に，ディレクトリと同じ名前のプロジェクトができます．この例では“~/Project_1/”下に“Project_1.Rproj”ができます．次回からは，RStudioのメニューからFile > Open Project…または，直接，ディレクトリを開いて，プロジェクト［Project_1.Rproj］をダブルクリックすれば，Project_１の環境が再現されます．

ちなみに，Project_1.Rprojをクリックし，RStuidoを起動させた後，コンソール画面にgetwd()―get working directoryを意味します―と入力し，現在の作業ディレクトリを確認してみてください．“~/Project_1/”と表示されるはずです．

```
getwd()        ＃現在の作業ディレクトリの場所を確認
```

これでプロジェクトを管理するディレクトリができました．それではさまざまなデータ形式の読み方を紹介しておきましょう．ここでは作業ディレクトリProject_１にch04_wage.csvファイル[3]が入っているという想定です．

Ⅱ．２　多様なデータ形式の読み込み方法

データを読み込む方法としては大きくは２つあります．１つは関数を利用する方法，もう１つはRStudioの［Import Dataset］プルダウンメニューを利用する方法です．

Ⅱ．２．１　Import Datasetを利用する

最初に，後者のRStudioの［Import Dataset］を使ってデータを読み込むとしましょう．Environmentタブのところにあるimport Datasetをクリックしてみてください．図５のようなメニューが開きます．

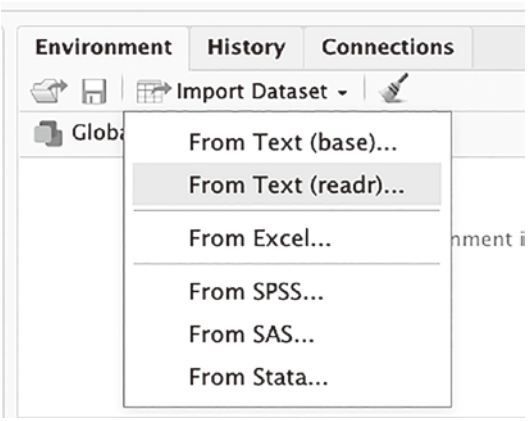


図５　Import Datasetメニュー

[3] 練習用ファイルとして西山慶彦他著『計量経済学』のウェブサポートで提供されているcsvファイルを利用します．サポートサイト（http://www.yuhikaku.co.jp/books/detail/9784641053854）から第４章のch04_wage.csvファイルをダウンロードし，作業ディレクトリProject_１の下に保存してください．

この図5から理解されるように，RStudioのImport Datasetを使えば，csvフィアルは言うまでもなく，主なファイル形式―エクセルファイル，SPSSファイル，SASファイル，Stataファイル―を読み込むことができます．

たとえば，今持っているファイルがStata形式だとしましょう．そのときはこのメニューから［From Stata…］を選択するだけです．［Browse］ボタンを押し，Stataファイルを選択すると，図6のようにデータが読み込まれます．右下の［Import］をクリックすると，ファイルがRStudioに読み込まれ，View()関数によってViewウィンドウが開きます．

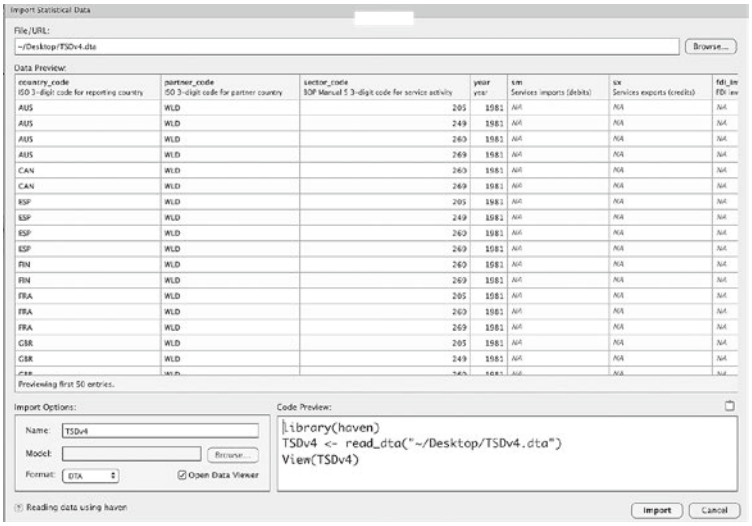


図6　Stata形式のファイルの読み込み

csv形式のフィルを読み込む場合は，図5において［From Text(readr)…］を選択し，ディレクトリ内のcsvファイルを指定するだけです．その結果，図6と同じウィンドウが開きます．あとは同じく右下の［Import］をクリックするだけです．

以上がRStudioのメニュー［Import Dataset］を利用したデータの読み込み方法ですが，再現性の観点からはこれはお奨めできません．データの読み込み手順も，スクリプトに書き込んだ方が良いでしょう．実は図6のウィンドウの右下にStataファイルを読み込むためのスクリプトが自動的に生成されます．関数を利用したデータの読み込みのさいには，これをコピーし，スクリプトにペーストすれば，作業が再現可能となります．

Ⅱ．2．2　関数を利用する

それでは次に関数を利用したデータの読み込み方法を紹介しましょう．

(1) csvファイルの読み込み

csv（comma-separated values）ファイルとはカンマ区切りで値が並べられているプレーンテキストファイルです．csv形式のファイルを読み込むためにはさまざまな関数がありますが，readrの中のread_csv()関数を利用します．readrはtidyverseがインストールされていれば，利用可能です．パッケージtidyverseをインストールし，library()で呼び出し，利用可能にしておきます．

```
install.packages("tidyverse")
library(tidyverse)
```

これでreadrも利用可能となります．それではreadrのread_csv()関数を使ってcsvファイルを読み込むとしましょう．read_csvのもっともシンプルなコードは次のようになります．

```
read_csv("csvファイルの名前")
```

練習用として作業ディレクトリProject_１の下に保存したch04_wage.csvファイルを読み込み，それをwageというオブジェクトに容れてみましょう．

```
wage <- read_csv("ch04_wage.csv")
```

スクリプトに上のように入力し，実行［Run］してみてください．RStudioの右上のEnvironmentウィンドウをみると，［Data］の下に21の観察値，３変数からなるwageデータセットが出来上がったことが分かります．head(wage)，str(wage)でデータ構造を確認してみてください．また小さなデータセットですからView(wage)で確認してみても良いでしょう．次に，read_csv()関数がとるいくつかの有益な引数を紹介しておきます．

－ 変数（列）タイプの指定：col_types = NULL

これはNULLの場合（もしくはこの引数が省略された場合），すべての変数（列）のタイプが最初の1,000行にもとづき指定されます．

```
> wage <- read_csv("ch04_wage.csv")

── Column specification ────────────────────
cols(
  year = col_double(),
  productivity = col_double(),
  wage = col_double()
)
```

図7　read_csv()によるcsvファイルの読み込み

read_csv(“ch04_wage.csv”)を実行したさいに，図7のような結果がコンソール画面に表示されます．たとえば，year = col_double()は変数列yearが倍精度浮動小数点数の数値タイプであることを示しています[4]．この表示は列が期待通りの変数タイプで読み込まれたかどうかをチェックするのに有益です．

「年」を表現するyear変数の型に倍精度浮動小数点数を利用することは計算資源の無駄遣いです．資源節約のため「整数」に変更するとしましょう．つまり変数yearを整数integerとして読み込むことにします．この場合，col_types=を利用します．図7の結果をそのままコピーし，yearの部分だけをyear = col_integerに変更し，read_csv()関数の引数col_typesに次のように指定します．

```
wage <- read_csv(“ch04_wage”,col_types =
  cols(
    year = col_integer(),                    ＃year変数の型を「整数」に変更
    productivity = col_double(),
    wage = col_double()
      )
```

str(wage)で変数の型を確認すると，変数yearがyear = col_integer()と整数に変わっていることが分かります．

－　列名の指定：col_names = TRUE

col_names = TRUEの場合，読み込むデータセットの最初の行が列名として利用され，その行

[4] Rは，倍精度浮動小数点数，整数，文字列，論理値，複素数,バイナリの6つのデータ型を認識します．Rの扱うデータの型については，たとえば，グロールマンド（2015）の第3章を参照してください．

はデータフレームには含まれません．他方，col_names = FALSEの場合，列の名前は，たとえばX1，X2，X3のように自動的に生成されます．また，col_names =c(“country”,“wage”,“GDP”)のように文字列のベクトルからなる場合，それが列名として利用されます．

－　欠損値の扱い：na= “NA”

これによって欠損値と解釈すべき文字列を指定します．Rでは欠損値は“NA”で表示されますので，na = “NA”と指定することができます．また“NA”に加えて“.”でも欠損値が表示されているような場合，na= c(“.”,“NA”)と指定します．

最後に，データフレームをcsvファイルとして保存する方法を紹介しておきましょう．

```
write_csv(データフレーム名, file = “保存データの名前.csv”, append=FALSE,col_names=FALSE)
```

append,col_namesの引数は省略しても構いませんが，append=は論理値をとり，FALSEの場合，新たな保存用ファイルを作成し，append=TRUEの場合には，既存のファイルに追加されます．col_names=FALSEの場合，ファイルのトップに列名は含まれません．TRUEの場合は列名が含まれます．

以上のcsvファイルの読み込み・保存手順までをスクリプトにまとめて書くと次のようになります．

```
library(tidyverse)
    wage <- read_csv(“ch04_wage.csv”)
    str(wage)
    head(wage)
    View(wage)
wage <- read_csv(“ch04_wage”,col_names = TRUE,col_types =
    cols(
    year = col_integer(),
    productivity = col_double(),
    wage = col_double()
)
write_csv(wage, “wage.csv”, append = FALSE, col_names = FALSE)
```

⑵　Excelファイルの読み込み

エクセルファイル，つまり拡張子が.xls，.xlsxのファイルを読み込むにはreadxlパッケージが必要となります．これもパッケージtidyverseに含まれていますのでtidyverseが組み込まれているのであれば，改めてインストールする必要はありません．

.xls，.xlsxのファイルを読み込むためには，readxlの中のread_excel()関数を利用します．ここでもサンプルファイルとして『計量経済学』提供の練習ファイルを利用します．第４章のdata for chap 4 exercise 10.xlsxファイルをダウンロードし，作業ディレクトリProject_１の下に保存してください．このエクセルファイルを読み込み，それをsampleというオブジェクトに容れます．

```
library(readxl)　　＃readxlの呼び出し
sample <- read_excel(“data for chap 4 exercise 2.xlsx”)
```

このスクリプトを実行［Run］すると，Excel形式のファイルが読み込まれます．View(sample)でデータ内容を表示させると，５変数の47観察値が確認されます．

これはもっともシンプルな書式ですが，read_excel()もread_csvと同様の引数を取りますが，いくつか違いがあります．もっとも大きな違いはエクセルファイルが複数のワークシートを持つ点にあります．このサンプルのエクセルファイルは１つのワークシートだけからなるファイルですが，エクセルファイルには複数のワークシートからなるものもあります．これを扱うためにread_excel()の引数sheet=シート番号によって特定のワークシートを指定できます．たとえば，次のように指定します．

```
sample <- read_excel(“data for chap 4 exercise 2.xlsx”,sheet=1)
```

⑶　Stata，SAS，SPSSファイルの読み込み

SPSS，SAS，およびStataのファイルを読み込むためには，havenパッケージが必要です．事前にインストールしておく必要がありますが，これもパッケージtidyverseがインストールされていれば，havenもインストールされています．SPSS，SAS，およびStataのファイルを読み込むために，最初にhavenを呼び出します．

```
library(haven)
```

Stataファイルを読み込むにはread_dta()，SASファイルの読み込みはread_sas()，SPSSファイ

ルを読み込むにはread_spss()を使います．たとえば，拡張子.savもしくは.zsavを持つSPSSファイルを読むとしましょう．かりにsample.savというファイルを持っていたとします．これをRで読み込み，sampleというオブジェクトに容れることを想定しましょう．次のように入力するだけです．

```
sample <- read_spss(“sample.sav”)
```

read_dta(),read_sas()についても同じです．

Ⅲ　Rパッケージを利用したマクロデータの取得

今では国際機関，研究組織によって多くのデータが提供されています．そうしたデータを取得するには，その機関・組織にアクセスし，csv，EXCEL等のファイルをダウンロードすることももちろん可能です．しかし，すでに述べたように，Rには，国際機関，研究組織の提供するデータベースからデータを取得するパッケージが多く開発されています．そうしたパッケージを利用すれば，シームレスかつ再現可能な形でのデータの読み込みが可能となります．そこで本資料では，Rパッケージを使った国際機関のデータの取得方法と簡単な利用方法を紹介します．

本資料では最初に，たとえば，penn world table，Word Development Indicatorsのようなマクロ経済統計データベースをとりあげます．

Ⅲ．1　Penn World Table ― pwt9

Penn World Tableは所得，産出高，投入および生産性に関する情報を提供するデータベースです．これは1950年から2017年の182ヵ国をカバーしています（基準年は2011年）[5]．pwt9はデータパッケージです．country（国名），isocode（３桁の国コード），year（年），currency（各国の通貨単位）の変数の他に，rgdpe（支出面の実質GDP），rgdpo（生産面の実質GDP），hc（１人あたり人的資本指数）などを含む48の時系列データが提供されています．それではinstall.packages()を使ってRにインストールしましょう．

[5] 詳細については，http://www.ggdc.net/pwt/ で提供されているFeenstra, Inklaar, and Timmer（2016, 2015）を参照されたい．

```
install.packages(“pwt9”)
```

Rパッケージptw9はデータだけのパッケージですので，data()関数で読み込みます．スクリプトに次のように入力し，実行［Run］してください．

```
data(“pwt9. 1”)
```

データ自体は更新され，現在，バージョン9. 1となっています．これを実行することにより，12, 376の観察値×52の変数のデータフレームが取得されます．

head()と反対のtail()関数―データセットの最終数行を表示させる関数―を使ってこのデータセットの最後の部分を見てみましょう．

```
tail(pwt9. 1)
```

これを実行［Run］すると，データセットの終わりの6行が表示されます．図8ではそのうち1列目のcountry変数から7列目のpop変数までを表示しています．

```
> tail(pwt9.1)
          country isocode year  currency    rgdpe    rgdpo      pop
ZWE-2012 Zimbabwe     ZWE 2012 US Dollar 26144.06 26444.04 14.71083
ZWE-2013 Zimbabwe     ZWE 2013 US Dollar 28086.94 28329.81 15.05451
ZWE-2014 Zimbabwe     ZWE 2014 US Dollar 29217.55 29355.76 15.41168
ZWE-2015 Zimbabwe     ZWE 2015 US Dollar 30091.92 29150.75 15.77745
ZWE-2016 Zimbabwe     ZWE 2016 US Dollar 30974.29 29420.45 16.15036
ZWE-2017 Zimbabwe     ZWE 2017 US Dollar 32693.47 30940.82 16.52990
```

図8　Penn World Table, pwt9. 1

ここでpwt9. 1データを使ってジンバブエ経済を観察してみましょう．そのためにジンバブエ経済だけ取り出し，zweというオブジェクトに容れます．

```
zwe <- pwt9. 1 %>%
filter(isocode == “ZWE”)          ＃isoコードでジンバブエを抽出
```

ここではジンバブエのマクロ状態―GDP，雇用，価格の成長率・上昇率―をグラフにし観察し

てみましょう．それぞれの変数名はrgdpe（実質GDP），emp（就業者数），pl_c（家計消費の価格水準）です．

グラフ作成までの手順は次のようになります．

手順１：３つの変数の時系列的推移を描くために，zweからこの３つの変数とyear変数を抽出します．

手順２：３つの変数を成長率に変換します．

手順３：最後に，３つの変数を表示するグラフを作成します．

手順１：特定の変数の取出し

select()関数を使ってyearと３つの変数―rgdpe（実質GDP），emp（就業者数），pl_c（家計消費の価格水準）―を抽出します．

```
zwe <- zwe %>%
        select(year, rgdpe, emp, pl_c)
```

手順２：成長率の計算

成長率の計算にあたっては対数差分を利用します．変数を対数に変換するにはlog()を使います．たとえば，log(rgdpe)と入力し，実行［Run］すれば，Rはrgdpeの対数を返します．さらに，１期前の対数の値との差をとる必要があります．１期前に変換するためにlag()関数[6]を利用します．lag(rgdpe)で１期前の実質GDPの値を指定できます．以上を利用すると，対数差分の計算式は以下のように表現されます．

```
log(rgdpe) - log(lag(rgdpe))          ＃成長率（対数差分）の計算
```

この計算式を利用し，mutate()関数を使って成長率変数を作成します．mutate()は既存の変数（列）に関数を適用し，新たな変数を作成する関数です．新たな変数の名前は，この例では既存の変数の冒頭にg_をつけたものにしています．

[6] Rにはdiff()という差分をとる関数がありますが，diff()は差分計算できない最初の値を除いた値を返すため，mutate()で変数を追加しようとすると，エラーになります．

```
zwe <-zwe %>%
  mutate(
      g_rgdpe = log(rgdpe) -log(lag(rgdpe)),
      g_emp = log(emp) -log(lag(emp)),
      g_pl_c =  log(pl_c) -log(lag(pl_c))
      )
```

以上をまとめてスクリプトに書くとつぎのようになります.

```
zwe <- pwt9. 1 %>%
        filter(isocode == “ZWE”)    %>%
        select(year, rgepe, emp, pl_c)    %>%
        mutate(
                g_rgdpe = log(rgdpe) -log(lag(rgdpe)),
                g_emp = log(emp) -log(lag(emp)),
                g_pl_c = log(pl_c) -log(lag(pl_c))
                )
```

これを実行［Run］すると，３つの変数とその成長率変数をもつジンバブエデータが作成されます．View(zwe)関数で確認してみてください.

手順３：グラフの作成

グラフの作成にはggplot2パッケージを利用します.

```
zwe_growth <-ggplot(zwe) +                    # GDP成長率のグラフ作成
      geom_point(aes(year, g_rgdpe))+
      geom_line(aes(year,g_rgdpe))+
      geom_abline(intercept = 0,slope = 0)+
      labs(x=“year”, y= “Growth rate”)+
      theme_bw()
```

```
zwe_inflation <- ggplot(zwe) +                    ＃インフレ（価格上昇率）のグラフ作成
        geom_point(aes(year, g_pl_c))+
        geom_line(aes(year,g_pl_c))+
        geom_abline(intercept = 0,slope = 0)+
          labs(x="year",y="Inflation")+
          theme_bw()
```

```
zwe_emp <- ggplot(zwe) +                          ＃雇用成長率のグラフ作成
        geom_point(aes(year, g_emp))+
        geom_line(aes(year,g_emp),lty="dashed")+
        labs(x="year",y="Employment")+
        geom_abline(intercept = 0,slope = 0) + theme_bw()
```

グラフの作成にあたってはggplot2を利用していますが，このパッケージもtidyverseをインストールすることによってインストールされます．すべてのグラフが同じ構成要素―データセット，座標システムおよびデータ点を表す描画の印―で作成されます．

使い方は構成要素を＋で重ねることが基本です．

```
ggplot(データフレーム名)+
geom_point(aes(x軸変数,y軸変数))+     ＃データを点で表現します．
geom_line(aes(x軸変数,y軸変数))+      ＃データを線で結びつけます．
```

この例ではlabs()を使ってx軸とy軸のタイトルを追加しています．labs()の書式は以下のとおりです．

```
labs(x="x軸タイトル", y="y軸タイトル")
```

さらに，geom_abline()を使って成長率ゼロの水準に直線を追加しています．geom_abline()の使用法は次のとおりです．

```
geom_abline(intercept = “直線の切片の位置”,slope =“直線の傾き”)
```

最後に，テーマを指定するtheme_でtheme_bw()を選択し，白黒のテーマを選んでいます．

それぞれ3つのグラフを描きますが，ここではRパッケージpachwork[7]を使ってグラフのレイアウトを操作します．

```
install.package(“patchwork”)
library(patchwork)
```

library(patchwork)で呼び出したのちに，次のように入力することでグラフを縦に配置します．

```
zwe_growth/zwe_inflation/zwe_emp
```

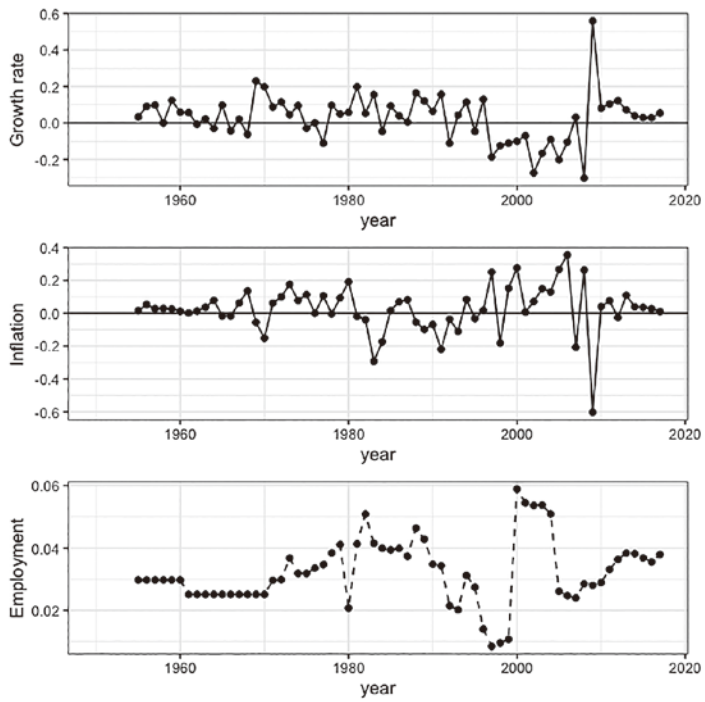


図9　ジンバブエのマクロ経済状態の推移

[7] patchworkパッケージは操作対象の図を足し算記号（+）や割り算記号（/）等を使って配置します。詳細についてはhttps://cran.r-project.org/web/packages/patchwork/vignettes/patchwork.htmlを参照してください．

３つの変数の変化率をグラフにしただけですが，これだけでも2000年代初頭に経済が大きな混乱の中にあったとことが伺われます．

Ⅲ．２　World Bank ― WDI

世界銀行が提供する世界開発指標WDI（World Development Indicators）は，グローバルな開発状況と貧困について国際的に比較可能な統計をまとめたものです．このデータベースには，217の経済と40以上の国グループの1,600の時系列指標が収録されています．またその多くの指標のデータは50年以上前までさかのぼることができます．

WDIデータは，さまざまな方法で取得できるようにされています．詳しくは同ホームページ（http://datatopics.worldbank.org/world-development-indicators/）を参照してください．多少時間がかかりますが，すべてのデータ（ExcelおよびCSV形式ファイル）をまとめてダウンロードすることも可能です．

このWDIデータを利用するためのRパッケージWDIが開発されています[8]．これ以外にもwbstatsというパッケージもありますが，ここではWDIを紹介します．

RパッケージWDIは，世界銀行によって運営される40以上のデータベースからデータを検索・ダウンロードすることを可能にしています．そうしたデータベースには世界開発指標（WDI）はもちろんのこと，国際債務統計，Doing Business，人的資本指数，サブナショナルな貧困指標も含まれます．

⑴　インストール方法

RパッケージWDIはCRAN上で公開されていますのでインストールにはinstall.packages()を利用します．コンソール画面に次のように入力し，エンターキーを押してください．

```
install.packages(“WDI”)
```

WDIを利用するために，スクリプト画面にlibrary(WDI)と入力し，実行［Run］しておきます．

```
library(WDI)
```

[8] https://www.youtube.com/watch?v=uSGfrUnn4YA

⑵　データを探す

データを探すためにはWDIパッケージに用意されたWDIsearch()関数を使います．この関数は利用可能なWDIデータ系列のコード名，名前，説明，およびデータソースから成る行列を返します．

基本的な書式は以下の通りです．

```
WDIsearch(string = “検索語”, field = “name”, short = TRUE, cache = NULL)
```

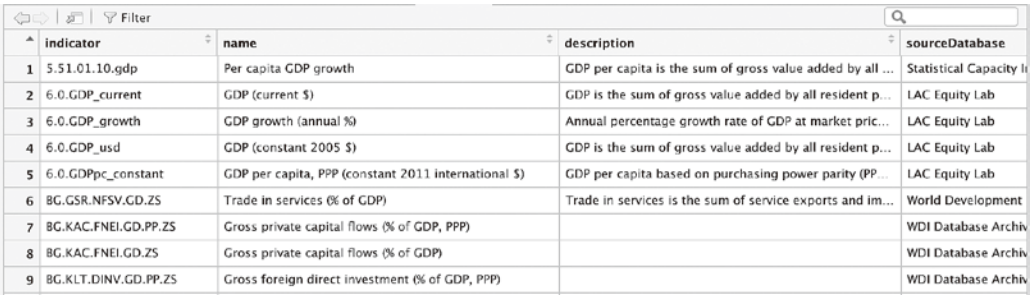
それぞれの引数を説明しましょう．

- string = “” : “” に検索語（文字列）を入力します．WDIsearchは文字列マッチング関数grepを使い，“検索語” を探します．また，caseを無視します（igunore.case=TRUE）ので，正規表現―簡単に言えば，通常の文字―が利用可能です．
- field = “” : “” に検索するフィールドを指定します．利用可能なフィールドは “indicator”, “name”, “description”, “sourceDatabase”, “sourceOrganization” です．たとえば “name” と入力すれば，データのnameの中を検索します．
- short = : 既定値はshort=TRUEです．この場合，指標コードと名前だけを返します．short=FALSEの場合，指標コード，名前，説明およびデータソースを返します．
- cache: WDIcache関数によって作成されるデータリストを返します．省略された場合（あるいはNULLの場合），WDIseachはデータ系列のローカルリストを探します．

たとえば，GDPに関するデータを探すとしましょう．この例では検索文字列を “gdp” とし，探すフィールドを “name” にしています．また，詳しい説明を得るためにshort=FALSEと設定しています．検索結果はオブジェクトgdpに格納されます．

```
gdp <- WDIsearch(string = “gdp”,field = “name”,short = FALSE, cache = NULL)
```

この結果をView(gdp)でみると，図10のような，539× 5 の行列が返されます．ここで重要なのはindicatorです．これは指標コードであり，ダウンロードのさいに利用されます．



Filter

	indicator	name	description	sourceDatabase
1	5.51.01.10.gdp	Per capita GDP growth	GDP per capita is the sum of gross value added by all ...	Statistical Capacity I
2	6.0.GDP_current	GDP (current $)	GDP is the sum of gross value added by all resident p...	LAC Equity Lab
3	6.0.GDP_growth	GDP growth (annual %)	Annual percentage growth rate of GDP at market pric...	LAC Equity Lab
4	6.0.GDP_usd	GDP (constant 2005 $)	GDP is the sum of gross value added by all resident p...	LAC Equity Lab
5	6.0.GDPpc_constant	GDP per capita, PPP (constant 2011 international $)	GDP per capita based on purchasing power parity (PP...	LAC Equity Lab
6	BG.GSR.NFSV.CD.ZS	Trade in services (% of GDP)	Trade in services is the sum of service exports and im...	World Development
7	BG.KAC.FNEI.GD.PP.ZS	Gross private capital flows (% of GDP, PPP)		WDI Database Archiv
8	BG.KAC.FNEI.GD.ZS	Gross private capital flows (% of GDP)		WDI Database Archiv
9	BG.KLT.DINV.GD.PP.ZS	Gross foreign direct investment (% of GDP, PPP)		WDI Database Archiv

図10　WDIsearch()の検索結果
注．4列目まで表示．

絞り込みが不十分なため，539行のデータ系列が表示されてしまっていますが，スクロールダウンして行くと，448行めにGDP（constant 2010 US$）が見つかります．以下の例においては，これをダウンロードします．

⑶　データをダウンロードする

データをダウンロードするためにはWDI()関数を利用しますが，利用方法は次のようになります．

```
WDI(
        country = “all”,
        indicator = “NY.GDP.MKTP.KD”,
        start = 1960, end = 2020, extra = FALSE, cache = NULL)
```

この関数は6つの引数をとります．それぞれを簡単に説明していきましょう．

- country = “” : “” の中に，ダウンロード対象の国名（ISO-2文字コードで表現された国名）を入力します．たとえば，“US”，“CA”，“JP” です．複数の国を指定したい場合，c()を利用します．たとえば，country = c(“US”,“CA”,“JP”)と入力します．なお，“all” と入力すると，すべての利用可能な国のデータがダウンロードされます．
- indicator = : 指標のコード名を入力します．これは図10のindicator列に表示されているものになります．
- start = : データの開始年です．通常整数フォーマットの年です（ただし，1960以上）．
- end = : データの終了年です．言うまでもなく，start引数に指定した値より大きくなければなりません．

- extra = : TRUEの場合，地域，iso3コード，所得水準といった追加的な変数を返します．
- WDIcache()によって作成されるリストで，extra = TRUEのとき利用されます．

それではWDI()関数を使って，アメリカ経済のGDP（constant 2010 US$）をダウンロードしてみます．このGDPデータのindicatorコードは，上述のとおり，NY.GDP.MKTP.KDです．ダウンロードしたデータをgdp_usという名前をつけたオブジェクトに格納します．

```
gdp_us <- WDI(country = “US”, indicator = “NY.GDP.MKTP.KD”,start = 1960,end = 2019, extra
                                                                    = FALSE,cache = NULL)
```

head()やView()関数を使ってダウンロードしたデータを確認してみてください．最後に，ダウンロードしたGDPデータをggplot2を使ってグラフにしてみましょう．

```
ggplot(gdp_us)+
        geom_point(aes(year,NY.GDP.MKTP.KD))+
        labs(
                title = “USA GDP, constant 2010 US$”,
                y = “GDP”
            )
+theme_bw()
```

これを実行［Run］すると，［Plots］ウィンドウに図11が表示されます．

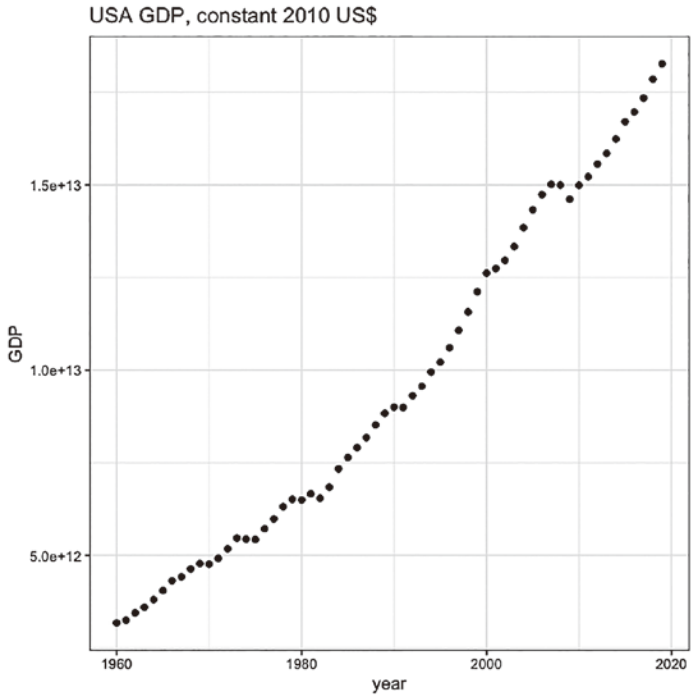


図11　アメリカ（USA）のGDPの推移，1960-2019

【参考文献一覧】

Aguirre,A and Danielsson, J. (2020) Which programming language is best for economic research: Julia, Matlab, Python or R? , voxel.org, 20 August.

Feenstra, Robert C., Robert Inklaar and Marcel P. Timmer (2015), The Next Generation of the Penn World Table, *American Economic Review*, 105(10), 3150-3182.

Perla, J., Sargent, T. J., and John Stachurski (2020) Quantitative Economcis with Julia, https://julia.quantecon.org/_downloads/pdf/quantitative_economics_with_julia.pdf

グロールマンド，G.（大橋真也・長尾高広訳）『RStudioではじめるRプログラミング入門』オライリー・ジャパン，2015年.

タディ，M.（上杉隼人・井上毅郎訳）『ビジネスデータサイエンスの教科書』すばる舎，2020年

西山慶彦，新谷元嗣，川口大司，奥井亮著『計量経済学』有斐閣，2019年.