『Rによる景気基準日付の同定と季節調整』に関する付録[1)]

地道 正行[2)], 豊原 法彦[3)]

2012年3月20日

A　Rからデータベースへ接続するための環境設定

ここでは, Rをクライアントとしてネットワークを経由してPostgreSQL[4)]がインストールされたサーバにアクセスするための環境設定に関して述べる. 設定手順は以下のようなものである.

(RD1) Linux (サーバ) 上の ODBC 環境の構築

(RD2) Windows (クライアント) 上の ODBC 環境の設定と RODBC パッケージのインストール

(RD3) データ抽出のための R のスクリプトの作成

なお, ODBC と RODBC の役割の概念図を図 1 に与えるので参照されたい .

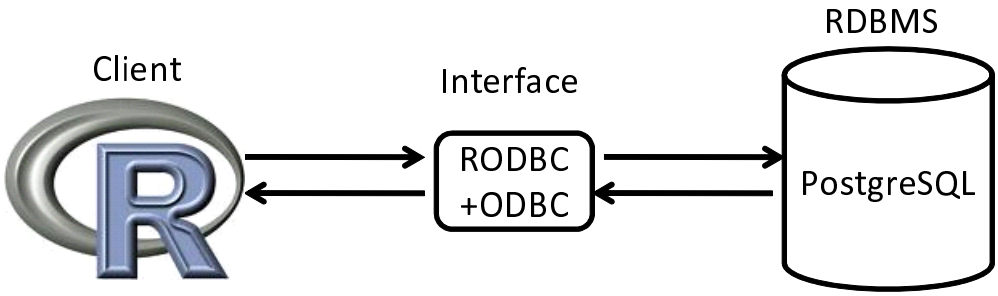


図 1: RODBC パッケージの役割

A.1　サーバ設定

データベースサーバの設定の詳細は本稿では割愛するが, ネットワークからRを用いてアクセスするための設定のポイントを以下に与えることにする. まず, Ubuntu[5)] 上にすでに PostgreSQL がインストールされていることを前提とする. ただし, `DBinstdir` は PostgreSQL がインストールされたディレクトリを表すことに注意しよう.

[1)] この文書は, 以下の書籍における第 6 章 (pp. 99–134) に関する付録である:
関西学院大学産研叢書 (35), 根岸 紳 編著, 『関西経済の構造と景気指数』, 日本評論社, 2012 年 3 月 20 日発行, ISBN 978-4-535-55682-9.

[2)] 関西学院大学商学部

[3)] 関西学院大学経済学部

[4)] オープンソースで開発されているリレーショナルデータベース管理システム (RDBMS) の一つ. PostgreSQL Global Development Group (`http://www.postgresql.org/`) により開発されており, 日本では「日本 PostgreSQL ユーザ会」(`http://www.postgresql.jp/`) を中心として普及促進を目的とした活動が行われている.

[5)] Ubuntu(ウブントゥ) とは、Ubuntu コミュニティにより開発されている Linux ベースのオペレーティングシステムである. 日本では Ubuntu Japanese Local Community Team (`http://www.ubuntulinux.jp/`) が情報提供を行っている.

(S-pg1) 設定ファイル DBinstdir/pg_hba.conf を以下のように修正する.

```
# TYPE    DATABASE    USER      CIDR-ADDRESS      METHOD
hostnossl   DB Name     all     172.20.0.0/16      trust
```

ここで, DB Name はデータベース名を表す.

(S-pg2) 設定ファイル DBinstdir/postgresql.conf を以下のように修正する.

```
listen_addresses = '*'
```

(S-pg3) 起動時に, postmaster [6] を -i オプション [7] 付きで実行する.

```
postmaster -i -D DBinstdir/ &
```

A.2 クライアント設定

ODBC 環境の設定 ここでは, Windows [8] 上での ODBC 環境の設定について述べる. 一般に, ODBC はクライアント・サーバ・システムであり, Windows と一般の UNIX サーバー上の DBMS に相互接続を可能とする. 一般的な設定手順は以下のようなものである.

ODBC 環境設定の共通手順

(手順 1) ODBC ドライバのインストール
(手順 2) DSN (Data Source Name) の設定

ここで, Windows 上では標準の ODBC 環境は用意されているけれども, 接続対象となるデータベース専用のドライバのインストールが必要となることに注意しよう. ここでは PostgreSQL に接続を前提としており, インストール手順としては PostgreSQL のサイト [9] からドライバソフト [10] をダウンロード後, インストールする必要がある. 具体的には, ダウンロードしたファイルを展開後, インストーラ (psqlodbc.msi) をダブルクリックすることによってインストールする. (図 2 のようなダイアログ (ボックス) が表示されるので, [Next] ボタンを押して順に質問に応えていけば問題はないであろう.)

[6] PostgreSQL マルチユーザベースのデータベースサーバプログラム.
[7] クライアントと TCP/IP 経由で接続を可能とするオプション.
[8] より正確には Windows7 32 bit version を想定している.
[9] http://www.postgresql.org/ftp/odbc/versions/msi/
[10] ここでは, psqlodbc_08_03_0400.zip を選択した.

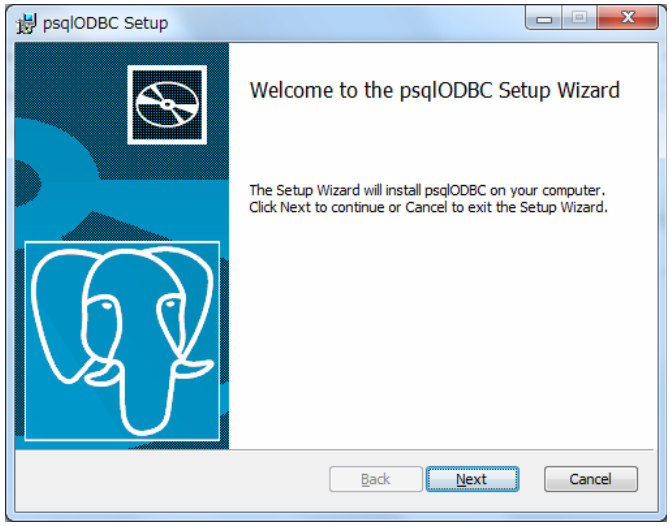


図 2: PostgreSQL 用 ODBC ドライバのインストール

次に, [コントロールパネル] の [管理ツール] から [データソース (ODBC)] アイコンをダブルクリックし, [ODBC データソースアドミニストレーター] を起動する.

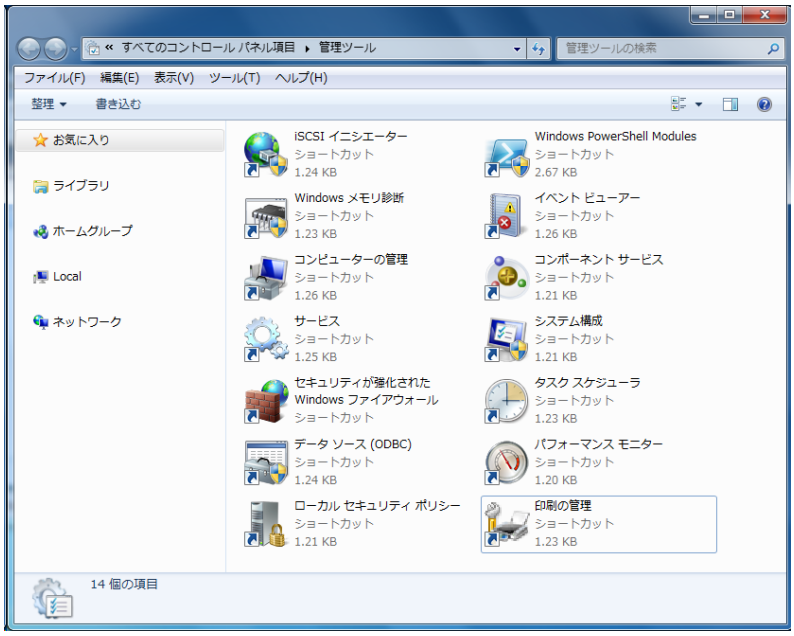


図 3: ODBC データソースアドミニストレーターの起動

[ODBC データソースアドミニストレーター] ダイアログが開くので, [ドライバ] タブで [PostgreSQL ANSI], [PostgreSQL Unicode] が表示されていれば PostgreSQL のドライバが正確にインストールされたことになる.

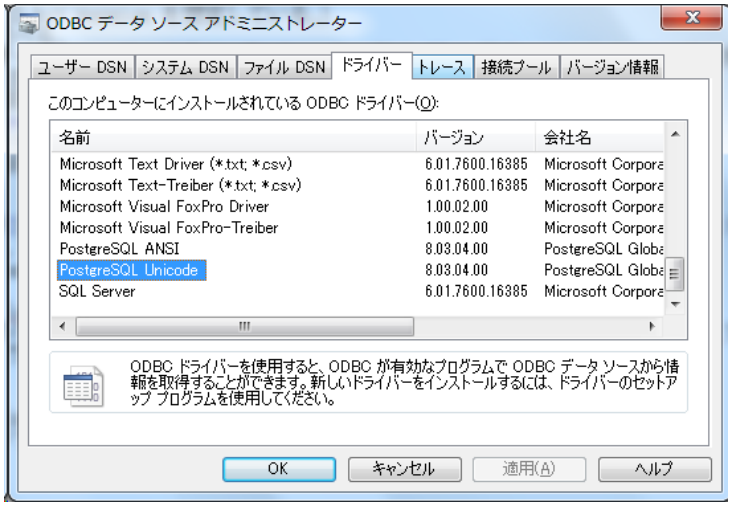


図 4: PostgreSQL 用ドライバの確認

次に, (手順 2) の DSN の設定について述べる. まず, [ODBC データソースアドミニストレーター] を起動後, [ユーザ DSN] [11] タブで [追加 (D)...] ボタンを押すことによって [データソースの新規作成] ダイアログが開くので, ドライバとして [PostgreSQL Unicode] を選択し, [OK] ボタンをクリックする.

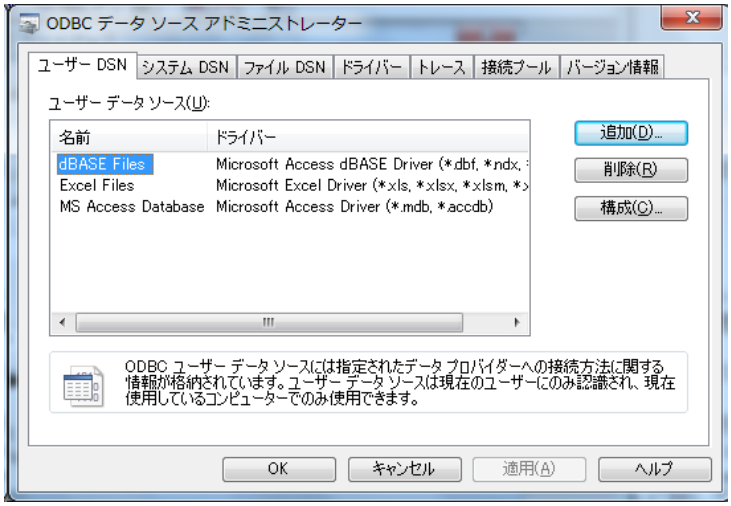


図 5: [ODBC データソースアドミニストレーター] ダイアログ

[11] 複数のユーザが利用しているコンピュータで設定する場合は [システム DSN] タブで設定することも検討する必要があろう.

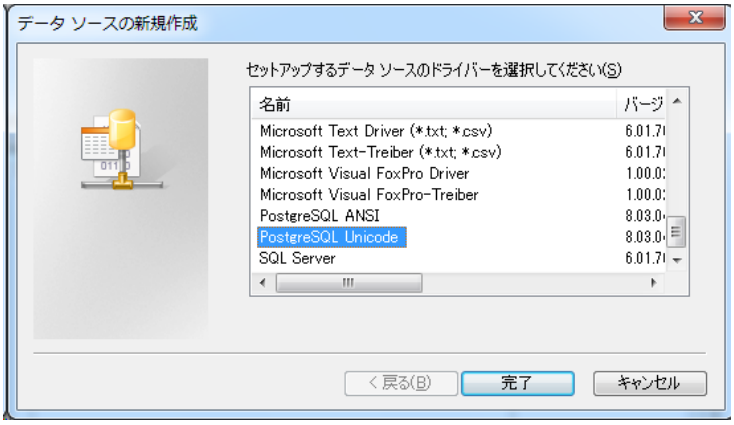


図 6: ドライバの指定

次に, [PostgreSQL Unicode ODBC セットアップ] のダイアログが開くので各項目に以下のような指定を行う.

データソース名: `PostgreSQL35W`[12)]
SSL Mode: 無効
サーバ名: `172.20.34.24`
Port: `5433`
データベース名: `ecofin`
ユーザ名: `ecofin`
パスワード: `*********`

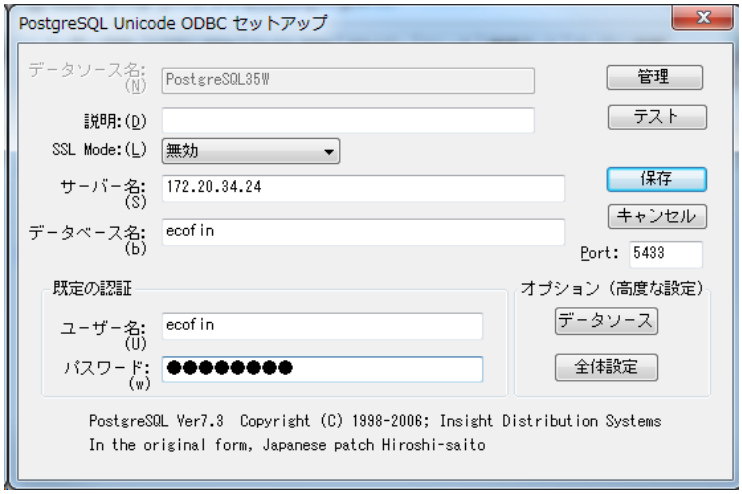


図 7: [PostgreSQL Unicode ODBC セットアップ] ダイアログ

なお, このダイアログにおいて [テスト] ボタンを押すことによって接続に関するテストを行うことができ, 以下のようなダイアログが表示されれば正しく接続できることがわかる.

[12)] 既定値として決まっていることに注意.

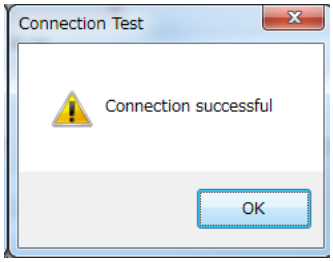


図 8: 接続が成功をしたことを表すダイアログ

接続の設定が正しく行われたならば, [PostgreSQL Unicode ODBC セットアップ] ダイアログの [保存] ボタンを押して設定を終了する. [ODBC データソースアドミニストレーター] ダイアログの [ユーザーデータソース (U):] の欄に [PostgreSQL35W] が登録されていることを確認し, [OK] ボタンを押して DSN の設定を終了する.

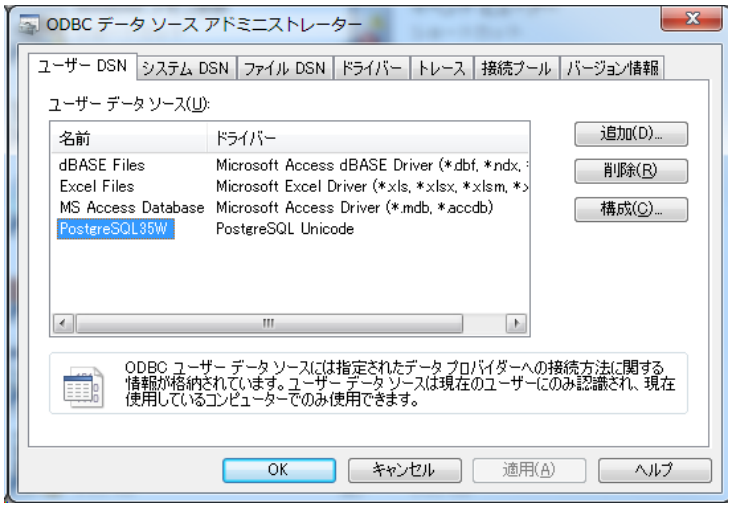


図 9: [PostgreSQL35W] の登録確認

RODBC 環境の設定　RODBC は R から ODBC サーバへネットワークを介してダイレクトにアクセスすることを可能するアドオン・パッケージであり, CRAN [13] サイトからダウンロードすればよい.

Windows 環境における設定は, 以下のような手順によって行われる. まず, [R Console] ウィンドウの [パッケージ] メニューから [CRAN ミラーサイトの設定...] を選択すると, [CRAN mirror] というダイアログが開くので, 適当なサイト (ここでは兵庫教育大学のサイト) を選択し, [OK] ボタンを押す.

[13] Comprehensive R Archive Network の略. R のソースコードや UNIX, Linux, Windows, Mac OS 用のバイナリファイル, ドキュメント, アドオンパッケージなどを提供するサイト. なお, URL は `http://cran.r-project.org/` である.

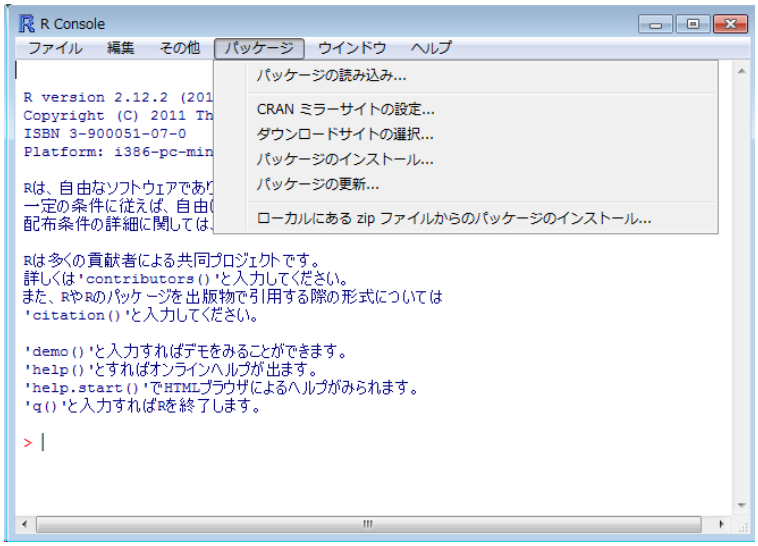


図 10: [パッケージメニュー] の選択

次に, [R Console] ウィンドウの [パッケージ] メニューから [パッケージのインストール...] を選択し [OK] ボタンを押すと, [Packages] というダイアログが開くので, RODBC を選択し [OK] ボタンを押す. すると, ミラーサイトで指定された場所からパッケージがダウンロードされインストールされる.

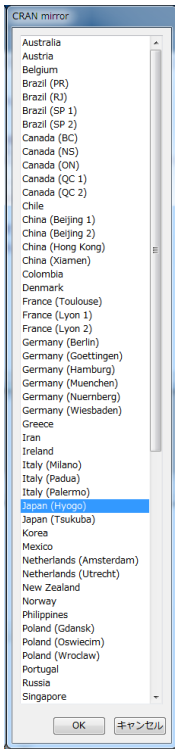


図 11: CRAN ミラーサイトの設定

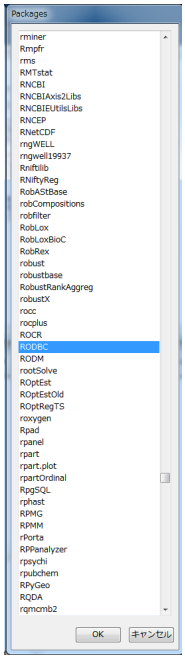


図 12: パッケージ RODBC の選択

B　R パッケージ datation について

Franck Arnaud 氏によって開発された R パッケージ datation [14] は, Bry-Boschan 法等を含む景気基準日付を決定するツールを R で利用可能とするものであり, 以下のものが提供されている.

- datation クラス (S4 仕様): 定義と要約, プロット等を行う関数の実装
- 景気基準日付に関するグラフによる比較
- Bry-Boschan 法 (関数 BB) と Harding-Pagan 法 (関数 BBQ) による転換点の景気基準日付の推定
- 2 種類の景気基準日付の提供: 米国における NBER [15] 月次景気基準日付とユーロ圏における CEPR [16] 四半期景気基準日付

これらの仕様と実装から, datation は S4 [17] に対応するなど機能面でも充実していることがわかる. なお, 詳細は Arnaud (2006) [18] を参照せよ.

datation パッケージのインストールは, 開発者のページからパッケージの zip アーカイブファイル datation_1.2.1.zip と移動平均に関するパッケージ mav の zip アーカイブファイル mav_1.0.2.zip を適当なフォルダへダウンロード後, [R Console] ウィンドウの [パッケージ] メニューから [ローカルにある zip ファイルからのパッケージのインストール...] を選択し, これら 2 つの zip ファイルを指定することによって行われる [19]. 以下のようなメッセージが [R Console] ウィンドウに出力されればパッケージが正常にインストールされたことを表している.

```
> utils:::menuInstallLocal()
 パッケージ 'datation' は無事に開封され、MD5 サムもチェックされました
 パッケージ 'mav' は無事に開封され、MD5 サムもチェックされました
```

C　日本の景気基準日付オブジェクトの作成

ESRI から発表されている景気基準日付を, パッケージ datation で定義されている datation クラスに属するオブジェクトとして作成する手順について述べる [20].

まず, datation クラスの定義を関数 getClass で以下のように確認する.

```
> getClass("datation")
Class "datation" [package "datation"]
Slots:
Name:     name  states   peaks troughs       y   param    type
Class:  vector      ts  vector  vector      ts  vector  vector
```

この結果から, datation クラスは, name, states, peaks, troughs, y, param, type という 7 つの

[14] http://arnaud.ensae.net/Rressources/Rressources.html から提供されている. なお, パッケージ名として用いられている “datation” は, 景気基準に関する「日付」を意味する単語 (仏語) であることに注意しよう.
[15] National Bureau of Economic Research の略. http://www.nber.org/
[16] Centre for Economic Policy Research の略. http://www.cepr.org/
[17] S Version 4 の意. J. Chambers 氏によって開発が進められている S 言語に関する最新バージョン. 詳しくはチェンバース (2002) を参照のこと.
[18] Arnaud, F. (2006) *Package datation@*, URL http://arnaud.ensae.net/Rressources/Rressources.html.
[19] なお, 旧バージョンに関しては, 総称関数 summary, plot の利用に関して若干の不具合があることに注意しよう.
[20] R における S4 クラス, メソッドなどに関する詳細についてはリゲス (2006) や間瀬 (2007) を参照せよ.

スロット[21] (slot) をもつことがわかり, このクラスに属するオブジェクトを作成するためには, これらのスロットに適切なクラスのオブジェクトを代入する必要がある. たとえば, スロット states は景気の状態 (state) を表し, 拡張期間は 1, 後退期間は -1 という値をもつ時系列オブジェクト (ts クラスに属するオブジェクト) を与える必要がある.

これらの情報をもとに ESRI から発表されている景気基準日付の datation クラスオブジェクトを作成しよう. まず, 以下のオブジェクトを作成する.

```
> rec.2010<-c(4,10,12,10,12,17,16,9,36,17,32,20,14)
> exp.2010<-c(27,31,42,24,57,23,22,28,28,51,43,22,69)
```

ここで, rec.2010 は 2010 年 10 月発表の資料にもとづく収縮期間のベクトルオブジェクトであり, exp.2010 は拡張期間のベクトルオブジェクトである.

次に, 景気の状態を生成する関数を以下のように作成する.

```
> create.states<-function(recession,expansion,start,start.state="peak")
+ {
+ if(start.state=="peak")
+ {
+ N<-length(recession)
+ states<-c(1)
+ for(i in 1:N) states<-c(states,c(rep(-1,recession[i]),rep(1,expansion[i])))
+ }
+ if(start.state=="trough")
+ {
+ N<-length(expansion)
+ states<-c(-1)
+ for(i in 1:(N-1)) states<-c(states,rep(1,expansion[i]),c(rep(-1,recession[i])))
+ states<-c(states,rep(1,expansion[N]))
+ }
+ states<-ts(states,start=start,frequency=12)
+ states
+ }
```

この関数を以下のように実行することによって, 景気の状態を表す時系列オブジェクトを生成することができる.

```
> create.states(recession=rec.2010,expansion=exp.2010,start=c(1951,6))
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1951                       1  -1  -1  -1  -1   1   1
1952   1   1   1   1   1   1   1   1   1   1   1   1
1953   1   1   1   1   1   1   1   1   1   1   1   1
1954   1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1   1
1955   1   1   1   1   1   1   1   1   1   1   1   1
(中略)
2006   1   1   1   1   1   1   1   1   1   1   1   1
2007   1   1   1   1   1   1   1   1   1   1
```

また, スロット peaks (「山」を表す.) , troughs (「谷」を表す.) に与えるためのオブジェクトを生成する関数を以下のように作成する.

[21] S4 オブジェクトの要素のこと.

```
> create.peaks.troughs<-function(recession,expansion,start.state="peak")
+ {
+ if(start.state=="peak")
+ {
+ N<-length(recession)
+ tmp<-NULL
+ for(i in 1:N) tmp<-c(tmp,c(recession[i],expansion[i]))
+ durations<-cumsum(tmp)+1
+ troughs<-durations[seq(1,2*N,2)]
+ peaks<-c(1,durations[seq(2,2*N,2)])
+ }
+ if(start.state=="trough")
+ {
+ N<-length(expansion)
+ tmp<-NULL
+ for(i in 1:(N-1)) tmp<-c(tmp,expansion[i],c(recession[i]))
+ tmp<-c(tmp,expansion[N])
+ durations<-cumsum(tmp)+1
+ troughs<-c(1,durations[seq(2,2*(N-1),2)])
+ peaks<-durations[seq(1,2*N,2)]
+ }
+ list(peaks=peaks,troughs=troughs)
+ }
```

この関数を以下のように実行することによって, 景気状態を表す時系列オブジェクトにおける山, 谷に対応する番号をベクトルオブジェクトとしてもつリストオブジェクトを生成することができる.

```
> create.peaks.troughs(recession=rec.2010,expansion=exp.2010)
$peaks
 [1]   1  32  73 127 161 230 270 308 345 409 477 552 594 677
$troughs
 [1]   5  42  85 137 173 247 286 317 381 426 509 572 608
```

以上の準備のもとで, ESRI から発表されている景気基準日付の datation クラスオブジェクト ESRI を関数 new を使って以下のように作成する.

```
> ESRI<-new("datation",
+ name=c("ESRI, Department of Business Statistics"),
+ states=create.states(recession=rec.2010,expansion=exp.2010,start=c(1951,6)),
+ peaks=create.peaks.troughs(recession=rec.2010,expansion=exp.2010)$peaks,
+ troughs=create.peaks.troughs(recession=rec.2010,expansion=exp.2010)$troughs,
+ y=ts(NA,c(1,1,1)),
+ param=list(),
+ type= "user-defined"
+ )
```

ここで, ESRI オブジェクトは以下のようなオブジェクトである.

```
> ESRI
Datation name : ESRI, Department of Business Statistics
      Peaks Troughs Duration
1    1951M6 1951M10        4
2    1954M1 1954M11       10
3    1957M6  1958M6       12
4   1961M12 1962M10       10
5   1964M10 1965M10       12
6    1970M7 1971M12       17
7   1973M11  1975M3       16
8    1977M1 1977M10        9
9    1980M2  1983M2       36
10   1985M6 1986M11       17
11   1991M2 1993M10       32
12   1997M5  1999M1       20
13 2000M11  2002M1       14
14 2007M10    <NA>     <NA>
```

このオブジェクトが示す景気基準日付とパッケージ datation に付属の NBER [22] を比較するためには, 以下のように入力することによってプロットを行えばよい.

```
> plot(ESRI,NBER)
```

この結果は図 13 のように与えられる .

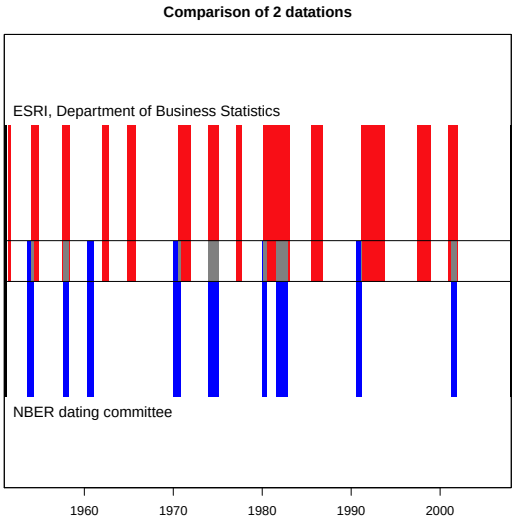


図 13: ESRI (日本) と NBER (米国) の景気基準日付の比較

同様の方法によって, 兵庫県の景気基準日付に関する datation クラスのオブジェクトを以下のように作成することができる.

[22] 米国の National Bureau of Economic Research から発表されている景気基準日付に関するオブジェクト.

```
> exp.Hyogo.2010<-c(57,22,17,27,23,52,42,14,63,10)
> rec.Hyogo.2010<-c(16,20,14,36,19,31,25,17,25)
> Hyogo<-new(
+ "datation",
+ name=c("Hyogo Pref. Data & Analysis Division"),
+ states=create.states(recession=rec.Hyogo.2010,
+ expansion=exp.Hyogo.2010,start=c(1965,12),start.state="trough"),
+ peaks=create.peaks.troughs(recession=rec.Hyogo.2010,
+ expansion=exp.Hyogo.2010,start.state="trough")$peaks,
+ troughs=create.peaks.troughs(recession=rec.Hyogo.2010,
+expansion=exp.Hyogo.2010,start.state="trough")$troughs,
+ y=ts(NA,c(1,1,1)),
+ param=list(),
+ type= "user-defined"
+ )
```

ここで, Hyogo は以下のようなオブジェクトである.

```
> Hyogo
Datation name : Hyogo Pref. Data & Analysis Division
      Peaks Troughs Duration
1      <NA> 1965M12      <NA>
2   1970M9  1972M1         16
3  1973M11  1975M7         20
4  1976M12  1978M2         14
5   1980M5  1983M5         36
6   1985M4 1986M11         19
7   1991M3 1993M10         31
8   1997M4  1999M5         25
9   2000M7 2001M12         17
10  2007M3  2009M4         25
11  2010M2    <NA>      <NA>
```

このオブジェクトが示す景気基準日付と ESRI から公表されているを日本の景気基準日付を比較するためのプロットを行う.

```
> plot(ESRI,Hyogo,start=c(1970,1),end=c(2004,12))
```

この結果は図 14 のように与えられる .

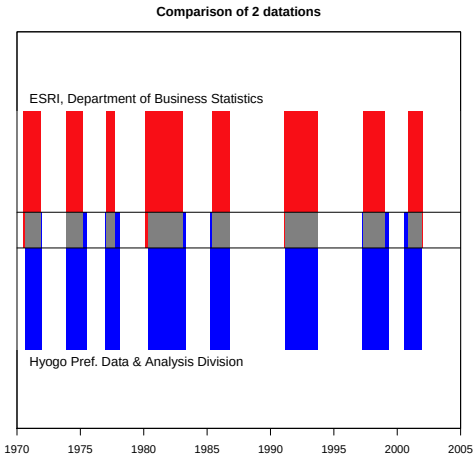


図 14: ESRI (日本) と Hyogo (兵庫県) の景気基準日付の比較

D Sweaveによる文書作成の自動化

R で得られた結果を含む文書を LaTeX [23] で作成するときに行うことは, 以下の作業を繰り返し行うことである:

1. 何らかのエディタを用いて LaTeX の文書作成
2. R を用いてデータを解析
3. 解析結果のリストや図を LaTeX の文書に挿入

この手順にあるように R を用いたデータを解析する段階で, 通常は R のスクリプトファイルを別途用意したり, EPS [24] 形式の画像ファイルなどをそろえ, LaTeX に挿入する必要がある. このような方法で作業を行う際の欠点として, 文書に挿入する R のリストや解析結果の画像ファイルは, データなどを異なるものに変更した時点で再度 LaTeX ファイルに挿入し直す必要がある.

このような作成過程の煩わしさを解消するために, 開発されたものが Sweave である [25]. Sweave は, LaTeX のソースファイルに R(または S) のコードを埋め込むことを可能にし [26], R(または S) で行った解析結果を含む LaTeX の文書作成を容易にする. また, ある種の文書自動生成機能などを持ち, 特にデータの統計的処理の結果を含むレポート作成に威力を発揮する. 以下に例を使って Sweave の活用法を説明する. なお, Sweave に関する詳細は R に付属のマニュアル [27] を参照されたい.

1. Sweave ファイルの作成 (拡張子 `Rnw`)

```
\documentclass[a4j]{jarticle}
\usepackage{C:/R/R-2.13.1/share/texmf/tex/latex/Sweave}
\begin{document}
\title{Sweave を使った文書作成}
\date{}
\maketitle
ここでは，Sweave を使った文書作成の簡単な例について書くことにする．
例として，正規乱数を発生させ，その時系列プロットを描く例を以下に与えよう:
<<>>=
n<-100
normal.data<-rnorm(n)
@
以上の入力で標準正規分布 $N(0,1)$ に従う \Sexpr{n} 個の乱数が発生できる．
次に関数 {\tt plot.ts} を使って，以下のように図を描くことができる:
\begin{center}
<<fig=TRUE,echo=TRUE,eps=TRUE>>=
plot.ts(normal.data)
@
\end{center}
\end{document}
```

これを `Sweaveexample.Rnw` というファイル名でセーブする.

2. ファイル `Sweaveexample.Rnw` がある場所を作業フォルダとして R を起動する [28].

---

[23] TeX は Donald Ervin Knuth 氏が作った組版システムであり, LaTeX は Leslie Lamport 氏が TeX の上に構築した文書処理システムである. 詳細については, たとえば,
`http://oku.edu.mie-u.ac.jp/~okumura/texwiki/` を参照されたい.

[24] Encapsulated Post Script の略. Adobe Systems 社が開発したページ記述言語 PostScript によって記述されたファイル形式の一つ. (`http://www.adobe.com/support/documentation/jp/` 参照.)

[25] Friedrich Leisch 氏による.

[26] Sweave ファイルは noweb ファイルである. noweb に関しては `http://www.cs.tufts.edu/~nr/noweb/` を参照のこと.

[27] Leisch, F. and R Core Team (2011) *Sweave User Manual*, R 2.13.1 online manual.

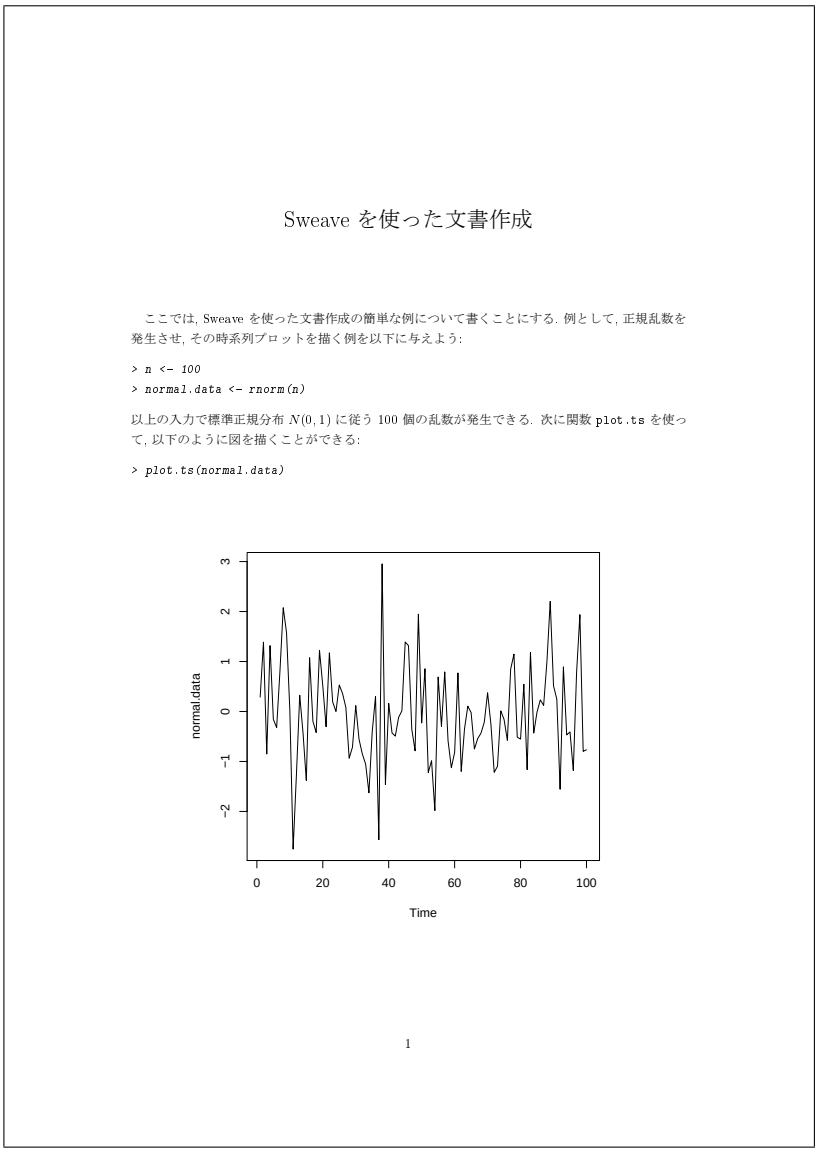
[28] R を起動してから作業フォルダをファイルが存在する場所に変更してもよい.

3. 関数 Sweave でファイル Sweaveexample.Rnw を処理[29].

```
> Sweave("Sweaveexample.Rnw",encoding="sjis")
Writing to file Sweaveexample.tex
Processing code chunks with options ...
 1 : echo term verbatim
 2 : echo term verbatim eps pdf

You can now run (pdf)latex on 'Sweaveexample.tex'
```

4. Sweaveexample.tex という名前の LaTeX のソースファイルが作成されるので, これをコンパイル & プレビューすることによって, 図 15 のような文書が作成される.



Sweave を使った文書作成

ここでは, Sweave を使った文書作成の簡単な例について書くことにする. 例として, 正規乱数を発生させ, その時系列プロットを描く例を以下に与えよう:

```
> n <- 100
> normal.data <- rnorm(n)
```

以上の入力で標準正規分布 $N(0,1)$ に従う 100 個の乱数が発生できる. 次に関数 plot.ts を使って, 以下のように図を描くことができる:

```
> plot.ts(normal.data)
```


図 15: Sweave を使って作成した文書の例

[29] R2.13.1 に付属の Sweave から, ファイルの文字コードが Shift-JIS である場合には, 引数 encoding="sjis" を指定する必要があることに注意しよう.

以上が Sweave を使った文書作成手順の簡単な例である.

では, Sweave の活用事例として, 適当に選んだ 18 本の系列に対するデータをデータベースから抽出し, 景気基準日付を同定するための表, グラフを生成し文書化するという過程を自動化するための Rnw ファイル (datationSweave.Rnw) を以下に与える.

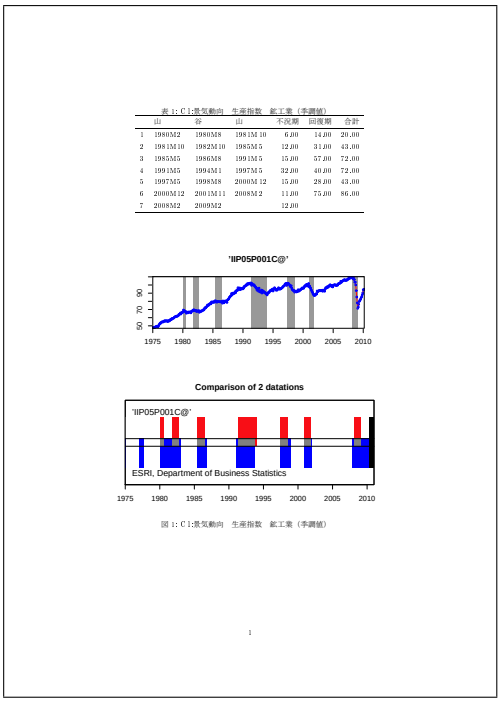
```
\documentclass[a4j]{jarticle}
\usepackage{graphicx}
\usepackage{float}
\usepackage{C:/R/R-2.13.1/share/texmf/tex/latex/Sweave}
\begin{document}
<<echo=F,results=hide>>=
# datation 読み込み
library(datation)
library(mav)
@
<<echo=F>>=
#plot.bb の定義
plot.bb<-function(bb=CIDX00C.bb,ts=CIDX00C.ts,ESRI=ESRI)
{
# Bry-Boschan 法と政府発表の景気基準日付の比較 (時系列付き, カラーバージョン)
par(mfcol=c(2,1))
plot(bb,ts,col.rec=grey(0.6))
# Bry-Boschan 法と政府発表の景気基準日付の比較 (期間限定比較, カラーバージョン)
plot(bb,ESRI)
par(mfcol=c(1,1))
}
#ESRI の定義
#2010 年 10 月発表の資料にもとづく収縮期間と拡張期間
rec.2010<-c( 4,10,12,10,12,17,16, 9,36,17,32,20,14,30,1)
exp.2010<-c(27,31,42,24,57,23,22,28,28,51,43,22,69,21,1)
# 景気状態の生成関数
create.states<-function(recession,expansion,start,start.state="peak")
{
if(start.state=="peak")
{
N<-length(recession)
states<-c(1)
for(i in 1:N) states<-c(states,c(rep(-1,recession[i]),rep(1,expansion[i])))
}
if(start.state=="trough")
{
N<-length(expansion)
states<-c(-1)
for(i in 1:(N-1)) states<-c(states,rep(1,expansion[i]),c(rep(-1,recession[i])))
states<-c(states,rep(1,expansion[N]))
}
states<-ts(states,start=start,frequency=12)
states
}
# 山谷の生成関数
create.peaks.troughs<-function(recession,expansion,start.state="peak")
{
if(start.state=="peak")
{
N<-length(recession)
tmp<-NULL
for(i in 1:N) tmp<-c(tmp,c(recession[i],expansion[i]))
durations<-cumsum(tmp)+1
troughs<-durations[seq(1,2*N,2)]
peaks<-c(1,durations[seq(2,2*N,2)])
}
if(start.state=="trough")
{
N<-length(expansion)
tmp<-NULL
for(i in 1:(N-1)) tmp<-c(tmp,expansion[i],c(recession[i]))
tmp<-c(tmp,expansion[N])
durations<-cumsum(tmp)+1
troughs<-c(1,durations[seq(2,2*(N-1),2)])
peaks<-durations[seq(1,2*N,2)]
}
list(peaks=peaks,troughs=troughs)
}## 日本版景気基準日付オブジェクトの生成関数
# クラス "datation" のオブジェクト (インスタンス) ESRI の作成
ESRI<-new(
```

```
"datation",
name=c("ESRI, Department of Business Statistics"),
states=create.states(recession=rec.2010,expansion=exp.2010),
peaks=create.peaks.troughs(recession=rec.2010,expansion=exp.2010)$peaks,
troughs=create.peaks.troughs(recession=rec.2010,expansion=exp.2010)$troughs,
y=ts(NA,c(1,1,1)),
param=list(),
type= "user-defined"
)
# RODBC 読み込み
library(RODBC,quietly=TRUE)
##サーバとの接続環境の設定
con<-odbcConnect('PostgreSQL35W;host=172.20.34.24;port=5433;dbname=ecofin1')
##edens2ts   edens から ts 形式への変換
edens2ts<-function(obj)
{
ts(obj$value,start=c(as.numeric(substring(obj$date[1],1,4)),as.numeric(substring(obj$date[1],5,6))),frequency=12)
}
#出力整序
summary.datation2<-function (object, print = TRUE)
{
    summarytab <- matsummary2(object)
    indic <- matrix(NA, 2, 2)
    colnames(indic) <- c("Amplitude", "Duration")
    rownames(indic) <- c("Exp=]T;P]", "Rec=]P;T]")
    indic[1, 1] <- mean(summarytab[summarytab[, 1] == 1, 7], na.rm = TRUE)
    indic[2, 1] <- mean(summarytab[summarytab[, 1] == -1, 7], na.rm = TRUE)
    indic[1, 2] <- mean(summarytab[summarytab[, 1] == 1, 4], na.rm = TRUE)
    indic[2, 2] <- mean(summarytab[summarytab[, 1] == -1, 4], na.rm = TRUE)
    if (isTRUE(print)) {
        df.print <- as.data.frame(summarytab)
        df.print[summarytab[, 1] == 1, 1] <- "Expansion"
        df.print[summarytab[, 1] == -1, 1] <- "Recession"
        df.print[, c(2, 3)] <- ts2char(object@states)[summarytab[, c(2, 3)]]
        df.print[, c(5, 6)] <- round(summarytab[, c(5, 6)], 0)
        df.print[, 7] <- round(summarytab[, 7], 1)
        colnames(df.print) <- c("Phase", "]Start", ";End]", "Duration", "LevStart", "LevEnd", "Amplitude")
    }
    list(df.print=df.print)
}
myformat<-function(obj)
{
N<-dim(obj)[1]
M<-(N-1)/2
tab1<-data.frame(matrix(0,M,3))
tab2<-matrix(0,M,2)
for(i in 1:M) tab1[i,]<-obj[seq(2*i-1,2*i+1,),3]
for(i in 1:M) tab2[i,]<-obj[seq(2*i,2*i+1),4]
tmp<-apply(tab2,1,sum)
table.peaks.troughs<-data.frame(tab1,tab2,tmp)
if(obj[1,1]=="Recession") col.name<-c("谷","山","谷","回復期","不況期","合計")
else col.name<-c("山","谷","山","不況期","回復期","合計")
dimnames(table.peaks.troughs)[[2]]<-col.name
table.peaks.troughs;
}
#景気動向指数のコードを直接入力
xitemcode<-c(
"'IIP05P001C@'",
"'IIP05R244L@'",
"'CEL9&PW@'",
"'IIP05O01C@'",
"'HWI95N04@'",
"'HWINMF90@'",
"'HWINMF95@'",
"'IIP05SINV@'",
"'IIP85S263@'",
"'IIP90S263@'",
"'ZCSHVB20V'",
"'ZCSHVB00V'",
"'ZBOASM@'",
"'SMSALE00@'",
"'SMSALE85@'",
"'SMSALE90@'",
"'SMSALE@'",
"'ESRAO@'")
xname<-c(
"C1:景気動向  生産指数  鉱工業（季調値）  ",
"C2:景気動向  鉱工業生産財在庫率指数（季調値）  ",
"C3:景気動向  電力使用量（九電力計）  季調値  ",
"C4:景気動向  稼働率指数  製造業（季調値）",
"C5:時間指数  労時指数  所定外  製造業（季）  ",
```

```
"C5:時間指数　労時指数　所定外　製造業（季）　　",
"C5:時間指数　労時指数　所定外　製造業（季）　　",
"C6:景気動向　投資財出荷指数（除輸送機械）",
"C6:景気動向　投資財出荷指数（除輸送機械）",
"C6:景気動向　投資財出荷指数（除輸送機械）",
"C7:商業販売額　小売業　前年同月比　",
"C8:商業販売額　卸売業　前年同月比　",
"C9:景気動向　営業利益（全産業）（季調値）",
"C10:景気動向　中小企業売上高　製造業",
"C10:中小売上高　中小企業売上高（製造業）　",
"C10:準備預金額　中小企業売上高（製造業）　",
"C10:景気動向　中小企業売上高　製造業",
"C11:一般職業　有効求人倍率（季調値）")
@
%=============================================================================================
<<results=tex,echo=F,fig=F>>=
N<-length(xitemcode)
library(xtable)
for(i in 1:N)
{
# export tables & input
sqq<-paste("SELECT itemcode,kind,date,value from edens10 where itemcode=",xitemcode[i]," order by date")
gdpmx.data<-sqlQuery(con,sqq);
gdpmx.ts=edens2ts(gdpmx.data)
gdpmx.bb<-BB(gdpmx.ts,name=xitemcode[i])
summary2.gdpmx.bb<-summary.datation2(gdpmx.bb)
file.tab=paste("table", i, ".tex", sep="")
tab.bb<-xtable(myformat(summary2.gdpmx.bb$df.print),caption=xname[i])
sink(file.tab)
print(tab.bb,table.placement="H",caption.placement="top")
sink()
cat("\\input{",file.tab,"}\n\n", sep="")
# export EPS files & input
file.eps<-paste("figure",i,".eps", sep="")
postscript(file=file.eps, paper="a4",horizontal=FALSE, width=6, height=6)
plot.bb(bb=gdpmx.bb,ts=gdpmx.ts,ESRI=ESRI)
dev.off()
cat("\\begin{figure}[H]\n\n", sep="")
cat("\\begin{center}\n\n", sep="")
cat("\\includegraphics{",file.eps,"}\n\n", sep="")
cat("\\caption{",xname[i],"}\n\n",sep="")
cat("\\end{center}\n\n", sep="")
cat("\\end{figure}\n\n", sep="")
cat("\\newpage\n\n", sep="")
}
@
%=============================================================================================

\end{document}
```

このファイルを Sweave 関数で処理し, LaTeX でコンパイルしたものは以下のようなものとなる.



. . . (省略) . . .

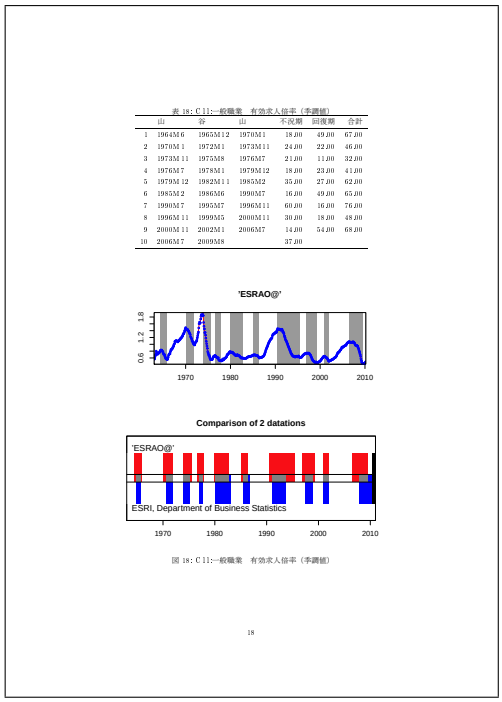


図 16: Sweave によって自動生成された景気基準日付の同定に関する文書

以上のことから, R と LaTeX を協調することによって, 単純な繰り返しであるが「手作業」では煩雑な文書作成作業を Sweave で自動化できることがわかった. ここでは 18 系列のデータに対する図表の作成のみを扱ったけれども, この系列が数百, 数千の規模に増加しても (時間やコンピュータ, ネットワークなどの環境に依存するものの) 原理的には同様の結果を得ることができることに注意しよう. このことは研究のための基礎資料を作成するときなどに強力な手段となりうることにも注意しよう.

E　Win X-12 と R パッケージ x12 のインストール

Win X-12 [30] は, 米国商務省センサス局 (U.S. Census Bureau) において公開されている季節調整ソフトウェア X-12-ARIMA [31] の Windows 向けインターフェースである. 一方, x12 は R から Win X-12 を利用して季節調整を実行するためラッパー [32] を提供するパッケージである.

以下に Win X-12 と x12 のインストール, およびそれらの設定法について述べる.

E.1　Win X-12 のインストール

Win X-12 のインストール手順は以下のようなものである.

[30] 原稿執筆時点での最新バージョンは 2.0 である.
[31] 原稿執筆時点での最新バージョンは 0.3 である.
[32] 関数がラッパー (wrapper) であるとは, その関数が単に他の関数に特定の引数を指定し, 呼び出すだけの役割を担っているものを指す. (スペクター (2008), p20 脚注参照.)

(1) Java のインストール

Win X-12 を利用する際にグラフの描画には Java^{TM}Runtime Environment が必要である．Java のサイト [33] からインストールプログラム `jxpiinstall.exe` をダウンロード後，ダブルクリックすることによってインストールする．

(2) Win X-12 のダウンロード

Win X-12 のパッケージファイル `winx12.zip` を米国商務省センサス局のダウロードサイト [34] からダウンロードする．

(3) Win X-12 のインストールと環境設定

Win X-12 のパッケージファイル `winx12.zip` を適当な場所 (ここでは，C ドライブのルート `C:\`) に展開後，そのフォルダにある Win X-12 の実行ファイル `WinX12.exe` をダブルクリックによって起動する．図 17 のようなダイアログが開くので，各設定項目に以下のように指定する．(図 18 も参照のこと.)

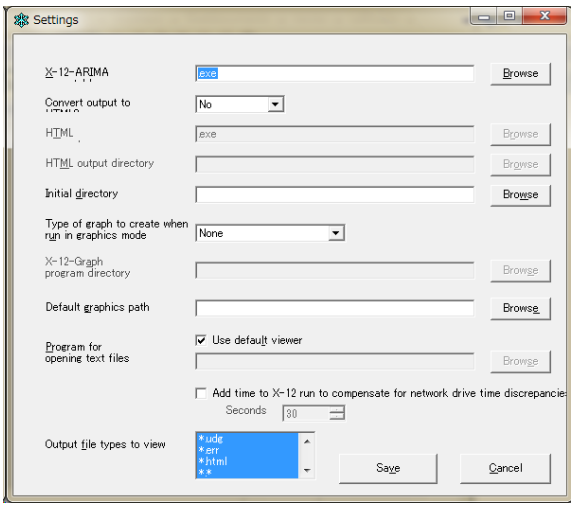


図 17: Win X-12 の設定 (1)

`X-12-ARIMA excululable` → `C:\WinX12\x12a\x12a.exe`

`Initial directory` → `C:\WinX12\data`

`Type of graph to create when run in graphics mode` → `Win X-12 JAVA graphs`

`Default graphics path` → `C:\WinX12\graphics`

[33] `http://www.java.com/ja/download/`
[34] `http://www.census.gov/srd/www/winx12/winx12_down.html`

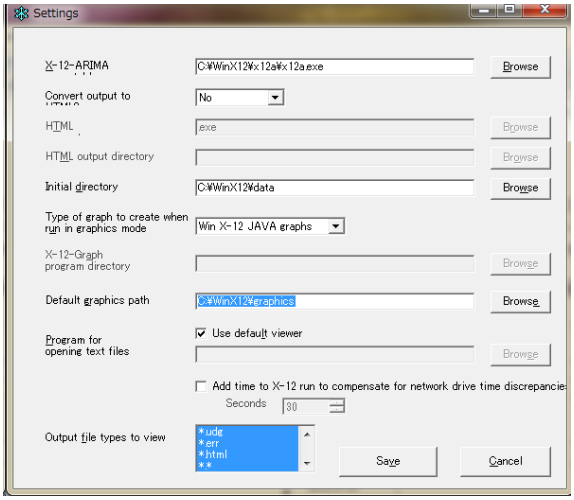


図 18: Win X-12 の設定 (2)

Save ボタンを押すことによって設定を保存後, 図 19 のようなダイアログが開けば設定が完了したことになる.

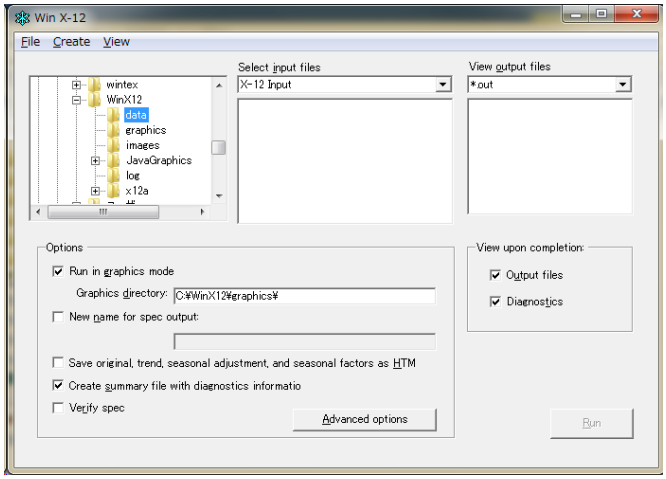


図 19: Win X-12 の起動

E.2 x12 のインストール

x12 パッケージのインストールは, 最寄りの CRAN サイトからパッケージの zip アーカイブファイル [35] を適当なフォルダへダウンロード後, [R Console] ウィンドウの [パッケージ] メニューから [ローカルにある zip ファイルからのパッケージのインストール...] を選択し, この zip ファイルを指定することによって行われる.

[35] 2011 年 8 月時点の最新バージョンは `x12_0.2-0.zip` である.

F　主要スクリプト

F.1　景気基準日付の同定に関するスクリプト

```
# ============================================
# データベースコネクションの確立
library(RODBC)
con<-odbcConnect('PostgreSQL35W;host=172.20.34.24;port=5433;dbname=ecofin')
con
sqlTables(con)
# 時系列データ
CI.coincident.data<-sqlQuery(con,"SELECT itemcode,kind,date,value
FROM edens11 WHERE itemcode='CIDX00C' ORDER BY date")
CI.coincident.data
CI.ts<-ts(CI.coincident.data$value, start=c(1980,1), frequency=12)
# 時系列オブジェクトへ自動変換する関数の作成
edens2ts<-function(obj)
{
ts(obj$value,start=c(as.numeric(substring(obj$date[1],1,4)),
as.numeric(substring(obj$date[1],5,6))),frequency=12)
}
CI.ts<-edens2ts(CI.coincident.data)
CI.ts
# datation パッケージによる Bry-Boschan (BB) 法の適用
library(datation)
CI.bb<-BB(CI.ts,name="Bry-Boschan")
summary(CI.bb)
plot(CI.bb,CI.ts)
# 日本版景気日付オブジェクト (datation class) の生成
# 2010 年 10 月発表の資料にもとづく収縮期間と拡張期間
rec.2010<-c(4,10,12,10,12,17,16,9,36,17,32,20,14)
exp.2010<-c(27,31,42,24,57,23,22,28,28,51,43,22,69)
# 景気状態の生成関数
create.states<-function(recession,expansion,start,start.state="peak")
{
if(start.state=="peak")
{
N<-length(recession)
states<-c(1)
for(i in 1:N) states<-c(states,c(rep(-1,recession[i]),rep(1,expansion[i])))
}
if(start.state=="trough")
{
N<-length(expansion)
states<-c(-1)
for(i in 1:(N-1)) states<-c(states,rep(1,expansion[i]),c(rep(-1,recession[i])))
states<-c(states,rep(1,expansion[N]))
}
states<-ts(states,start=start,frequency=12)
states
}
create.states(recession=rec.2010,expansion=exp.2010,start=c(1951,6))
# 山谷の生成関数
create.peaks.troughs<-function(recession,expansion,start.state="peak")
{
if(start.state=="peak")
{
N<-length(recession)
tmp<-NULL
for(i in 1:N) tmp<-c(tmp,c(recession[i],expansion[i]))
durations<-cumsum(tmp)+1
troughs<-durations[seq(1,2*N,2)]
peaks<-c(1,durations[seq(2,2*N,2)])
}
if(start.state=="trough")
{
N<-length(expansion)
tmp<-NULL
```

```
for(i in 1:(N-1)) tmp<-c(tmp,expansion[i],c(recession[i]))
tmp<-c(tmp,expansion[N])
durations<-cumsum(tmp)+1
troughs<-c(1,durations[seq(2,2*(N-1),2)])
peaks<-durations[seq(1,2*N,2)]
}
list(peaks=peaks,troughs=troughs)
}
create.peaks.troughs(recession=rec.2010,expansion=exp.2010)
# 日本版景気基準日付オブジェクトの生成関数
# クラス "datation" のオブジェクト (インスタンス) ESRI の作成
ESRI<-new("datation",
name=c("ESRI, Department of Business Statistics"),
states=create.states(recession=rec.2010,expansion=exp.2010,start=c(1951,6)),
peaks=create.peaks.troughs(recession=rec.2010,expansion=exp.2010)$peaks,
troughs=create.peaks.troughs(recession=rec.2010,expansion=exp.2010)$troughs,
y=ts(NA,c(1,1,1)),
param=list(),
type= "user-defined"
)
# ESRI オブジェクトの表示
ESRI
# Bry-Boschan 法と政府公表の景気基準日付の比較 (時系列付き, グレイスケールバージョン)
par(mfcol=c(2,1))
plot(CI.bb,CI.ts,col.rec=grey(0.5))
plot(ESRI,CI.ts,col.rec=grey(0.5))
par(mfcol=c(1,1))
# Bry-Boschan 法と政府公表の景気基準日付の比較 (時系列付き, カラーバージョン)
par(mfcol=c(2,1))
plot(CI.bb,CI.ts,col.rec="red")
plot(ESRI,CI.ts,col.rec="blue")
par(mfcol=c(1,1))
# 日本 ESRI と米国期間 NBER 発表の景気基準日付の比較 (期間限定比較, カラーバージョン)
plot(ESRI,NBER)
## 兵庫県の景気日付オブジェクト (datation class) の生成
#2010 年 9 月発表の資料にもとづく収縮期間と拡張期間
exp.Hyogo.2010<-c(57,22,17,27,23,52,42,14,63,10)
rec.Hyogo.2010<-c(16,20,14,36,19,31,25,17,25)
Hyogo.states<-create.states(
recession=rec.Hyogo.2010,
expansion=exp.Hyogo.2010,
start=c(1965,12),start.state="trough"
)
Hyogo.peaks.troughs<-create.peaks.troughs(
recession=rec.Hyogo.2010,
expansion=exp.Hyogo.2010,
start.state="trough"
)
# 兵庫県の景気基準日付オブジェクトの生成関数
# クラス "datation" のオブジェクト (インスタンス) Hyogo の作成
Hyogo<-new(
"datation",
name=c("Hyogo Pref. Data & Analysis Division"),
states=Hyogo.states,
peaks=Hyogo.peaks.troughs$peaks,
troughs=Hyogo.peaks.troughs$troughs,
y=ts(NA,c(1,1,1)),
param=list(),
type= "user-defined"
)
# 日本 (ESRI) と兵庫県 (Hyogo) の景気基準日付の比較
plot(ESRI,Hyogo,start=c(1970,1),end=c(2004,12))
# 兵庫県の CI 一致指数 (1981 年 1 月から 2010 年 12 月)
CI.Hyogo<-c(
  84.7, 84.2, 84.5, 84.9, 83.9, 84.7, 85.8, 87.2, 87.5, 85.8, 88.6, 88.1,
  88.1, 87.2, 88.5, 86.4, 86.0, 86.9, 85.2, 86.0, 84.2, 83.6, 83.6, 82.9,
  81.9, 81.5, 81.2, 81.7, 81.7, 82.9, 82.9, 84.4, 85.1, 85.4, 85.8, 87.4,
```

```
  87.3, 88.9, 88.4, 86.8, 87.8, 88.6, 89.9, 88.0, 88.8, 90.0, 89.1, 90.2,
  90.4, 90.5, 90.8, 93.9, 91.4, 90.7, 90.8, 91.3, 90.2, 91.3, 90.0, 87.9,
  88.4, 88.5, 86.8, 87.0, 85.1, 84.6, 83.8, 83.1, 83.2, 82.3, 81.5, 82.4,
  83.4, 85.8, 84.7, 85.0, 87.6, 88.2, 89.1, 89.8, 91.6, 94.9, 96.9, 96.9,
  98.5,101.4,100.2,102.1,101.5,103.2,103.5,105.2,107.0,106.0,108.2,108.8,
 108.5,109.9,114.2,112.8,113.6,114.6,115.0,114.9,116.3,115.6,116.6,119.2,
 117.2,118.2,120.1,119.1,122.1,119.4,120.6,121.3,117.1,118.9,118.6,118.9,
 119.0,119.6,119.4,117.7,118.0,117.0,114.0,110.7,107.9,107.5,106.4,104.3,
 102.8,101.2,100.5, 98.7, 96.1, 95.5, 93.6, 91.4, 93.3, 91.2, 89.3, 88.9,
  89.0, 87.5, 85.9, 85.7, 83.1, 83.1, 83.4, 81.1, 79.8, 80.7, 80.1, 79.8,
  81.5, 81.2, 82.7, 80.6, 81.7, 82.2, 81.5, 84.1, 83.4, 82.6, 85.9, 85.2,
  71.5, 68.6, 71.7, 77.9, 81.6, 81.5, 81.9, 85.2, 83.4, 84.7, 85.7, 87.8,
  87.8, 92.7, 92.7, 94.5, 96.3, 94.1, 96.1, 94.4, 96.8,100.0, 99.8,100.3,
 104.0,101.3,101.3,101.3,103.6,101.6,101.6,101.0,104.1, 99.8, 99.4, 98.6,
  99.1, 95.8, 94.4, 94.3, 91.2, 92.7, 89.5, 88.5, 87.6, 87.5, 86.2, 85.9,
  87.3, 86.9, 87.9, 86.3, 86.5, 85.6, 85.9, 87.6, 88.4, 87.2, 87.6, 87.4,
  87.4, 90.7, 90.1, 91.6, 90.5, 90.3, 89.4, 91.7, 92.1, 91.3, 91.3, 92.2,
  90.6, 89.6, 88.2, 88.2, 88.1, 87.5, 87.0, 83.5, 83.8, 83.0, 82.9, 81.2,
  82.2, 82.0, 83.3, 82.3, 83.3, 83.9, 84.5, 84.7, 84.3, 86.2, 87.0, 87.8,
  88.9, 88.8, 89.8, 87.9, 88.7, 88.8, 88.5, 88.6, 90.5, 93.2, 91.4, 92.3,
  93.8, 93.0, 93.2, 94.4, 94.9, 96.0, 97.4, 94.3, 95.5, 96.9, 97.2, 97.5,
  98.1, 97.2, 98.1,102.0, 98.4,101.0, 99.3,103.1, 99.9,100.1,101.0,101.7,
 103.9,105.2,106.1,108.2,107.9,109.8,108.8,110.3,109.3,108.0,108.5,107.3,
 105.0,109.4,104.3,106.9,107.0,104.1,103.9,104.6,101.5,102.3,103.2,102.9,
 102.1,102.5,101.5,101.4,102.6,101.4,101.5, 97.9, 98.5, 96.5, 91.8, 88.0,
  83.0, 77.6, 78.0, 77.4, 75.8, 78.2, 78.0, 78.4, 81.2, 82.0, 83.3, 84.3,
  85.9, 86.8, 87.4, 88.7, 91.3, 91.6, 91.6, 93.8, 92.3, 90.3, 90.5, 92.7)
# 兵庫県の CI 一致指数（1981 年 1 月から 2010 年 12 月）の時系列オブジェクト作成
CI.Hyogo.ts<-ts(CI.Hyogo,start=1981,frequency=12)
# 兵庫県の CI 一致指数に Bry-Boschan 法を適用
CI.Hyogo.bb<-BB(CI.Hyogo.ts,name="Bry-Boschan: Hyogo CI")

# Bry-Boschan 法と兵庫県公表の景気基準日付の比較（時系列付き，グレイスケールバージョン）
par(mfcol=c(2,1))
plot(CI.Hyogo.bb,CI.Hyogo.ts,col.rec=grey(0.5))
plot(Hyogo,CI.Hyogo.ts,col.rec=grey(0.5))
par(mfcol=c(1,1))
# Bry-Boschan 法と兵庫県公表の景気基準日付の比較（時系列付き，カラーバージョン）
par(mfcol=c(2,1))
plot(CI.Hyogo.bb,CI.Hyogo.ts,col.rec="red")
plot(Hyogo,CI.Hyogo.ts,col.rec="blue")
par(mfcol=c(1,1))
# 日本 ESRI と米国期間 NBER 発表の景気基準日付の比較（期間限定比較，カラーバージョン）
plot(CI.Hyogo.bb,Hyogo,end=c(2009,12))
# データベースコネクションの終了
close(con)
```

F.2 季節調整に関するスクリプト

```
#=====================================================
# データベースコネクションの確立
library(RODBC)
con <- odbcConnect('PostgreSQL35W;host=172.20.34.24;port=5433;dbname=ecofin')
# タイトル作成
xname0<-sqlQuery(con,"SELECT name,unit from edens11_header
                      where itemcode='CELL9' and dum1='N1'")
xname<-paste(xname0$name,":",xname0$unit,sep="")
xname<-gsub("  ","",xname)
xname<-gsub(" ","",xname)
# データ抽出
C03.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
                        where itemcode='CELL9' order by date")
C03.data
# データの初期を読み込み，YYYYMMDD の形式から YYYY.MM の形に変換
stx<-gsub(" ","",paste(substring(C03.data$date[1],1,4),".",substring(C03.data$date[1],5,6)))
# ファイル出力
```

```
sink(file = "./C03.spc") #出力ファイルのオープン
cat("series { title = \" ") #タイトル
writeLines(paste(xname,"\" "))
cat("data = ( \n") #処理対象データの設定
writeLines(paste(C03.data$value))
cat(")\n")
cat("start = ") #収録データの開始時点の設定
cat(stx)
cat("}\n")
cat("transform{function =auto}\n") #変換方法の設定（対数など）
cat("regression{variables = td}\n") #回帰分析の方法設定
cat("outlier{}\n") #外れ値に関する設定
cat("x11{save=d11}\n") #季調済みの値のみ出力するよう設定
sink() #出力ファイルのクローズ
# データベースコネクションの終了
close(con)
#=====================================================
# sepc ファイル自動生成関数の作成
makespecfile<-function(itemcode,idcode,spcfname)
{
# データベースコネクション
library(RODBC)
con <- odbcConnect('PostgreSQL35W;host=172.20.34.24;port=5433;dbname=ecofin')
# タイトル作成
sqq1<-paste("SELECT name,unit from edens11_header
             where itemcode='",itemcode,"' and dum1='N1'",sep="")
xname0<-sqlQuery(con,sqq1)
xname<-paste(xname0$name,":",xname0$unit,sep="")
xname<-gsub("  ","",xname)
xname<-gsub(" ","",xname)
xname<-paste(idcode,"_",xname,sep="")
# データ抽出
sqq2<-paste("SELECT itemcode,kind,date,value from edens11
             where itemcode='",itemcode,"' order by date",sep="")
edens.data<-sqlQuery(con,sqq2)
stx<-gsub(" ","",paste(substring(edens.data$date[1],1,4),".",substring(edens.data$date[1],5,6)))
# ファイル出力
sink(file=spcfname)
cat("series { title = \" ")
writeLines(paste(xname,"\" "))
cat("data = ( \n")
writeLines(paste(edens.data$value))
cat(")\n")
cat("start = ")
cat(stx)
cat("}\n")
cat("transform{function =auto}\n")
cat("regression{variables = td}\n")
cat("outlier{}\n")
cat("x11{save=d11}\n")
sink()
close(con)
}
# 各種の系列に対する spc ファイル出力
makespecfile(itemcode="IIP05P001",idcode="C01",spcfname="C01.spc")
makespecfile(itemcode="IIP05S243",idcode="C02",spcfname="C02.spc")
makespecfile(itemcode="CELL9",idcode="C03",spcfname="C03.spc")
makespecfile(itemcode="IIP05O01",idcode="C04",spcfname="C04.spc")
makespecfile(itemcode="HWI05N04",idcode="C05",spcfname="C05.spc")
makespecfile(itemcode="IIP05S204",idcode="C06.1",spcfname="C06.1.spc")
makespecfile(itemcode="IIP05S213",idcode="C06.2",spcfname="C06.2.spc")
makespecfile(itemcode="ZCSHVB20",idcode="C07",spcfname="C07.spc")
makespecfile(itemcode="ZCSHVB00",idcode="C08",spcfname="C08.spc")
makespecfile(itemcode="SIP05SS40N@",idcode="C10.1",spcfname="C10.1.spc")
makespecfile(itemcode="CGPI05S200",idcode="C10.2",spcfname="C10.2.spc")
makespecfile(itemcode="ESRAO",idcode="C11",spcfname="C11.spc")
#=====================================================
# WinX12 と R package ‘‘x12’’ の利用
# データベースコネクションの確立
library(RODBC)
```

```
con <- odbcConnect('PostgreSQL35W;host=172.20.34.24;port=5433;dbname=ecofin')
# データ抽出
C03.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='CELL9' order by date")
# 時系列オブジェクト生成
C03.ts<-edens2ts(C03.data)
# ライブラリ x12 の読み込み
library(x12)
# 季節調節オブジェクトの生成
C03.x12<-x12(C03.ts,x12path="C:/WinX12/x12a/x12a.exe")
plot(C03.x12)
# データ抽出
C01.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='IIP05P001' order by date")
C02.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='IIP05S243' order by date")
C03.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='CELL9' order by date")
C04.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='IIP05O01' order by date")
C05.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='HWI05N04' order by date")
C06.1.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='IIP05S204' order by date")
C06.2.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='IIP05S213' order by date")
C07.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='ZCSHVB20' order by date")
C08.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='ZCSHVB00' order by date")
C10.1.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='SIP05SS40N@' order by date")
C10.2.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='CGPI05S200' order by date")
C11.data<-sqlQuery(con,"SELECT itemcode,kind,date,value from edens11
where itemcode='ESRA0' order by date")
# 時系列オブジェクト生成
C01.ts<-edens2ts(C01.data)
C02.ts<-edens2ts(C02.data)
C03.ts<-edens2ts(C03.data)
C04.ts<-edens2ts(C04.data)
C05.ts<-edens2ts(C05.data)
C06.1.ts<-edens2ts(C06.1.data)
C06.2.ts<-edens2ts(C06.2.data)
C07.ts<-edens2ts(C07.data)
C08.ts<-edens2ts(C08.data)
C10.1.ts<-edens2ts(C10.1.data)
C10.2.ts<-edens2ts(C10.2.data)
C11.ts<-edens2ts(C11.data)
# x12 オブジェクト生成
C01.x12<-x12(C01.ts,x12path="C:/WinX12/x12a/x12a.exe")
C02.x12<-x12(C02.ts,x12path="C:/WinX12/x12a/x12a.exe")
C03.x12<-x12(C03.ts,x12path="C:/WinX12/x12a/x12a.exe")
C04.x12<-x12(C04.ts,x12path="C:/WinX12/x12a/x12a.exe")
C05.x12<-x12(C05.ts,x12path="C:/WinX12/x12a/x12a.exe")
C06.1.x12<-x12(C06.1.ts,x12path="C:/WinX12/x12a/x12a.exe")
C06.2.x12<-x12(C06.2.ts,x12path="C:/WinX12/x12a/x12a.exe")
C07.x12<-x12(C07.ts,x12path="C:/WinX12/x12a/x12a.exe")
C08.x12<-x12(C08.ts,x12path="C:/WinX12/x12a/x12a.exe")
C10.1.x12<-x12(C10.1.ts,x12path="C:/WinX12/x12a/x12a.exe")
C10.2.x12<-x12(C10.2.ts,x12path="C:/WinX12/x12a/x12a.exe")
C11.x12<-x12(C11.ts,x12path="C:/WinX12/x12a/x12a.exe")
# データベースコネクションの終了
close(con)
#=====================================================
# データベースコネクションの確立
library(RODBC)
con <- odbcConnect('PostgreSQL35W;host=172.20.34.24;port=5433;dbname=ecofin')
```

```
# DI 一致指数の抽出
DI.data<-sqlQuery(con,"SELECT itemcode,kind,date,value FROM edens11 WHERE itemcode='DIDXC' ORDER BY date")
# ts オブジェクトへ変換
DI.ts<-edens2ts(DI.data)
# スペクトルのプロット
spectrum(DI.ts,main ="Spectrum: DI",method='ar')
# データベースコネクションの終了
close(con)
```
