C 付録3　Stan導入

■ ■ ■

本文書では Stan バージョン 2.10 以降における Stan コード記述法の変更を受け，Stan バージョン 2.12 に準拠して付録 3 の一部内容を改訂いたしました．Stan バージョン 2.9 までの Stan コード記述法に関しては書籍版の付録をご参照ください．

付録 3 では HMC 法をソフトウェアを用いて実行し，実際に推定値を得る方法について解説します．本書では全体を通じて統計解析環境 R[*1]，HMC 法の実行ソフトウェアである Stan[*2]，および R における Stan 用インターフェース RStan[*3] を使用します [*4]．

Stan は，モデルを記述したコードを C++言語にコンパイルした後にサンプリングを行うことで，非コンパイル型のモンテカルロサンプリング用のソフトウェアに対して，サンプリング結果を得るまでが短時間で済むという長所を有しています．

まず本付録の前半では，Stan 実行方法を説明します．HMC(NUTS 法 (付録 2 参照)) を用いたサンプリングを行うために，Stan モデリング言語を用いてモデルを特定した Stan コードファイル，R 言語の記法を用いてデータを記述したデータファイル，RStan パッケージの関数を用いて記述された R コードファイルが必要となります．各ファイルは適当な名称をつけ，ディレクトリに保存されている必要があります．実際に操作するインターフェースは R コードファイルおよび R コンソールであり，これらを通じて作成した Stan ファイル，データファイルの読み込みを行い，サンプリングを実行します．

C.1　Stan コード解説 (6.2.2 項「研修効果問題」)

本書に登場した例題を解くために使用した Stan コードは巻末に一括で掲載しています．ここでは 6.2.2 項で説明された「研修効果問題」を解くための Stan コードを例にとり，Stan，RStan の具体的な実行方法について解説します．まず Stan コードを以下に示します．

```
1  data {
2          int<lower=0> N;
3          vector[2] x[N];
4  }
```

[*1] R Core Team. (2016): R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

[*2] Stan Development Team. (2016): Stan Modeling Language Users Guide and Reference Manual, Version 2.12.0. http://mc-stan.org

[*3] Stan Development Team. (2016): RStan: the R interface to Stan, Version 2.12.1. http://mc-stan.org.

[*4] 付録 3 改訂版執筆時点において，R 3.3.1，Stan 2.12.0 および RStan 2.12.1 を使用しています．


```
5  parameters {
6          vector[2] mu;
7          vector<lower=0>[2] sigma;
8          real<lower=-1,upper=1> rho;
9  }
10 transformed parameters {
11         vector<lower=0>[2] sigmasq;
12         matrix[2,2] Sigma;
13         sigmasq[1] = pow(sigma[1],2);
14         sigmasq[2] = pow(sigma[2],2);
15         Sigma[1,1] = sigmasq[1];
16         Sigma[2,2] = sigmasq[2];
17         Sigma[1,2] = sigma[1]*sigma[2]*rho;
18         Sigma[2,1] = sigma[1]*sigma[2]*rho;
19 }
20 model {
21         for(i in 1:N){
22                 x[i] ~ multi_normal(mu,Sigma);
23         }
24 }
25 generated quantities{
26         real delta;
27         real delta_over;
28         real delta_over2;
29         real rho_over;
30         real rho_over05;
31         delta = mu[2] - mu[1];
32         delta_over = step(delta);
33         delta_over2 = delta>2 ? 1 : 0;
34         rho_over = step(rho);
35         rho_over05 = rho>0.5 ? 1 : 0;
36 }
```

Stan コードは，コード内で用いる変数の定義を行う変数宣言と，プログラムが処理を実行する内容を記述した文—ステートメント (statement) の記述から成り立っています．

本例の Stan コードはプログラムブロックと呼ばれる，中括弧{}で囲まれている特定の役割を持った単位に分かれ，それぞれのブロックの内部でステートメントが記述されています．ブロックの記述順は定められています．必要のないブロックについては省略することが可能です．コメント文を除き，ステートメントは必ずブロック内にて記述する必要があります．本例に登場しないブロックについては C.4.1 項を参照してください．

コメント文記法　Stan コードにおいて特定の方法で表記された記述はコメント文として扱われ，実行されません．コメント文の記法は以下の 3 種類があります．

- 「#」以降の当該行に記された文字列はすべてコメント文として扱われる
- 「//」以降の当該行に記された文字列はすべてコメント文として扱われる
- 開始記号「/*」と終了記号「*/」で囲まれた文はすべてコメント文として扱われる

data ブロック data ブロックでは，分析に用いるデータに関係した変数定義を記述します．1 行目でブロックの開始を宣言します．2 行目からは，変数定義を行っています．まず人数を表す変数を N とすることを宣言します．N は整数値をとるため，変数の型は int を指定します．山括弧< >は変数の定義域を宣言する際に用いられ，下限は lower，上限は upper で指定します．本例の場合，N は 0 未満の値をとらないため，変数の下限を<lower=0>と宣言することとします．なお，この場合の変数定義の記述は「型 (範囲 [サイズ]) 変数名 ([サイズ]);」の順番に行う必要があります．

1 つのステートメントの記述が終わった場合には，必ず文末に「; (セミコロン)」を記し，当該行におけるコードの記述が終了したことを宣言します．

3 行目では，各人のデータ値を表す変数を x と定義しています．変数の後に x[数値] と記述した場合，当該変数は配列 (array) 型として扱われ，[] 内の数値は要素数を表します．x[N] は要素数 N の配列 x を定義したこととなります．

ただし x[N] の前に vector[2] というコードが記述されています．これは，配列 x 内の N 個の要素が，N 本のサイズ 2 のベクトルであることを表しています．具体的には，社員 20 名の研修前と後の営業成績のことを指します．vector は，それ自体が変数型として Stan 内に用意されており，実数型として扱われます．Stan において利用可能な変数型については C. 4. 2 項を参照してください．

4 行目において閉じ括弧を記述し，data ブロックが終了することを宣言します．なお，前のブロックで定義した変数は後のブロックでも用いることができますので，後続のブロックで同じ変数を再定義する必要はありません．ただし，その逆は許されません．また，model ブロックで定義した変数を generated quantities ブロックに引き継ぐことはできません．

parameters ブロック 本ブロックでは推定対象となる母数について変数宣言を行います．

6 行目の vector[2] mu は 2 変量正規分布におけるサイズ 2 の平均ベクトルとして用いるために定義しています．

7 行目，vector<lower=0>[2] sigma は，標準偏差を表す変数定義です．モデルが 2 変量正規分布であることから共分散行列の対角要素として用いるために，サイズ 2 のベクトル型としてあります．実際には共分散行列を作成する必要がありますが，それは後のブロックにて記述します．sigma はその定義から負の値をとらないため，範囲の下限を 0 に制限します．変数の定義域の記述位置に注意してください．定義域を宣言する場合には，< >を必ず変数の型宣言の直後に記述する必要があります．

8 行目は相関係数を表す変数の宣言です．その性質から実数型とし，なおかつ定義域を -1 から $+1$ としています．

transformed parameters ブロック このブロックでは，parameters ブロックにおいて宣言した変数に対し，変換を施すステートメントを記述するためのオプションブロックです．

11 行目は分散として扱う変数を宣言しています．vector<lower=0>[2] は 7 行目の標準偏差の宣言と同様の理由による記述です．

12 行目は 2 変量正規分布における 2×2 の分散共分散行列を Sigma と名付けることを宣言しています．matrix は当該変数が行列型であることを指定しており，[2,2] は行列のサイズが 2 行，2 列であることを示しています．

13，14 行目は実際に標準偏差 sigma を分散に変換し，変数 sigmasq に代入する処理を行っています．pow(引数 1, 引数 2) は累乗を行う関数であり，引数 1 に与えられた値について，(引数 2 に与えられた値) 乗を返します．ここでは標準偏差を 2 乗することを表しています．

15 から 18 行目は共分散行列 Sigma の各要素に対して値の代入操作を行っています．対角要素には分散 sigmasq の値を，非対角要素には共分散の値を与えています．

model ブロック model ブロックでは，具体的な統計モデルを記述します．また母数に関するサンプリングについても本ブロック内で記述します．本例では研修受講前と後の社員の成績が 2 変量正規分布に従っているものと仮定しました．

本例における仮説モデルのための Stan コード (サンプリングステートメント) は x[i] ~ multi_normal(mu, Sigma); となります．この文は i 番目の社員の研修前後の成績が 2 変量正規分布に従っているものとして，それぞれの母数に関してサンプリングを行うことを宣言しています．この際 i は for ループにより，1 から N まで動き，各 i ごとにサイズ 2 の平均ベクトルが取り出されます．ここでは，「~ (チルダ)」の左辺に記述された変数が，右辺に記述された分布，平均ベクトル mu，共分散行列 Sigma を持つ多変量正規分布に従っていることを表しています．

本ブロックにおいて，前ブロックまでで記述していない変数を用いる場合には，新たに変数宣言を行う必要があります．ただし後に変数のサンプリング結果を出力するためには，(transformed) parameters ブロックに記述するか，もしくは生成量として generated quantities ブロックに記述し，サンプリング実行時に stan() および print() の引数 pars に当該変数名を与えます．

generated quantities ブロック generated quantities ブロックは必要に応じて，生成量 $g(\cdot)$ について記述します．本ブロックに記述した生成量は繰り返しごとに計算され，最終的に経験分布として得られます．26 行から 30 行目では本ブロックで用いられる変数の宣言を行っています．

31 行目は研修後の平均値と，研修前の平均値の差を計算し，変数 delta に代入しています．

32 行目では仮説の確からしさを推測するための量を生成するため，関数 step() による演算を行っています．step() は delta の値が 0 以上 (> 0) であれば 1 を，それ以外には 0 を返します．

33 行目では効果量の差に関する量を生成します．条件式 a ? b : c は条件式 a について論理判定を行い，TRUE であれば b の値を，FALSE であれば c の値を返します．

なお，Stan コード内においてサンプリング対象としつつも，明示的に事前分布を指定しなかった場合には，Stan によって無情報的事前分布が仮定されます．

上記の Stan コードファイルに適当な名称をつけ，拡張子を「.stan」(たとえば「model.stan」) とし，保存します．

C. 2 data ファイル

データファイルは R 関数を用いて定義します．ファイル作成においてはデータと Stan コードの互いに定義した変数名の対応関係が保たれるように注意してください．

```
1 N <- 20
2 x <- structure(.Data=c(
3 6,11,10,13,17,10,10,7,9,1,14,7,7,11,12,12,14,12,7,13,
4 7,11,14,13,16,12,8,15,12,3,17,11,9,11,14,12,11,15,11,17),.Dim=c(20,2))
```

N には社員数を付値します．関数 structure() には引数.Data に対して関数 c() によってベクトル形式でデータを与え，引数.Dim にデータの次元数を列数，行数の順番に与えます．データベクトルが 1 次元の場合には c() のみを用いて与えてもかまいません．

上記のデータファイルに適当な名称と拡張子をつけ (たとえば「data.R」)，保存します．

C.3 RStan

RStan は R における Stan のインターフェースを提供するパッケージです．RStan を用いることで Stan ファイルの読み込み，データファイルの読み込み，結果の出力を R コンソール上で行うことが可能となります．なお R スクリプト内ではコメント文を表すコードとして「#」のみが使用可能です．

```
1  library(rstan) # パッケージの読み込み
2  scr <- "model622.stan" # Stan ファイル名
3  source("data622.R") # data ファイル名
4  data <- list(N=N,x=x) # data ファイル内の変数をリスト形式にまとめる
5  # サンプリング結果を出力する変数名
6  par<-c("mu","Sigma","rho",
7         "delta","delta_over","delta_over2","rho_over","rho_over05")
8  war <- 1000 # ウォームアップ期間
9  ite <- 11000 # 繰り返し数
10 see <- 1234 # 乱数の種
11 dig <- 3 # 有効数字
12 cha <- 1 # 連鎖構成数
13 fit <- stan(file=scr, model_name=scr, data=data, pars=par,verbose=F,
14              seed=see, chains=cha, warmup=war, iter=ite)
15 print(fit,pars=par,digits_summary=dig) # 結果の出力
16 plot(fit, pars=par)
17 traceplot(fit, inc_warmup = T, pars=par)
```

1 行目では RStan パッケージの読み込みを行います [*5)]．

2 行目から 11 行目までは関数 `stan()` によってサンプリングを行うための引数の設定を行っています．サンプリング結果の出力は 6 行目で指定した変数についてのみ行えます．

9 行目の繰り返し数はウォームアップ期間を含めた全体の更新回数を指定します．10 行目ではサンプリングの開始時における初期値を生成するための乱数の種を値で指定します．種自体は任意の値を指定できますが，結果を再現する場合には必ず同じ種を与える必要があります．

13 行目から 14 行目において，関数 `stan()` を実行することで，モデルコードのコンパイルとサンプリングが行われます．`stan()` は Stan コードを C++用コードとしてコンパイルし，Stan によるサンプリングを実行するための関数です．引数 `file` には Stan ファイル，`data` に 4 行目で作成したリストオブジェクトを指定します．`pars` には結果を出力するための変数名ベクトルを与えます．`seed` はサンプリングの初期値生成のための乱数の種を与えます．`chains` に連鎖構成数を，`warmup` にウォームアップ (バーンイン) 期間として破棄するサンプル数，`iter` にウォームアップ期間を含めた全体の連鎖更新回数を与えます．サンプリング結果はオブジェクト `fit` に付値されます．

15 行目の関数 `print()` は結果オブジェクトを与えることで，サンプリング結果の要約を出力するために用います．`pars` に対しては `stan()` に与えたものと同じオブジェクトを指定します．`digits_summary` は出力の際の有効桁数を与えます．

16，17 行目のプロットの描画については C.5 節を参照してください．

上記 RStan コードを記述したファイルに対して，適当な名称をつけ (たとえば「chap6.R」)，保存します．このとき Stan ファイル，データファイルとのディレクトリの位置関係に注意してください．

Stan ファイルおよびデータファイルを用意した上で，R コンソール上において RStan コードを実行することで，Stan によるベイズ推定を行うことができます．

[*5)] 必要なパッケージは一部の例外を除いて，R 起動時に毎回読み込む必要があります．

C.3.1 出 力 結 果

関数 `print()` を実行することで，指定した各母数のサンプリング結果についての要約指標が出力されます．たとえば例題の場合の結果については以下のように出力されます (ここでは一部のみ)．

```
          mean se_mean    sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
mu[1]   10.131   0.012 0.822  8.466  9.600 10.137 10.661 11.748  5045    1
mu[2]   11.944   0.011 0.790 10.395 11.423 11.942 12.466 13.524  4755    1
```

上記出力は左から変数名，`mean`：EAP 推定値，`se_mean`：連鎖が収束している場合の EAP 推定値の標準偏差の推定値，`sd`：事後標準偏差 (post.sd)，`2.5% - 97.5%`：95%確信区間および四分位範囲，`n_eff`：有効サンプルサイズ，`Rhat`：収束判定指標 $\hat{R}$ を表しています．

有効サンプルサイズは，連鎖更新回数に近いほどよく，$\hat{R}$ は 1.1 ないし 1.2 以下であればよいとされます．詳しい導出については付録 2 を参照してください．

C.4 Stan コード文法概略

C.4.1 ブ ロ ッ ク

1 つのファイル中の Stan コードは必要に応じて，7 つのブロックに分けて記述します．

`data` ブロックにおいてはデータとなる確率変数を指定します． `parameters` ブロックで宣言した自由母数の分布は `model` ブロックで指定します．サンプリングごとに必要となる副次的な (制約・固定・変換) 母数は `transformed parameters` ブロックで指定します．サンプリング終了後に変換する母数は `generated quantities` ブロックにて記述します．

各ブロックの役割

- `functions{}`：本ブロックでは自作関数を定義することができます．
- `data{}`：変数宣言のみを行います．Stan 外部からの入力データについての変数を本ブロックに記述します．このブロックにはステートメントは記述しません．
- `transformed data{}`：サンプリング実行中に値が変化しない変数 (データ) を扱い，当該データを変形します．もし一部でも値が変化するならば，`transformed parameters{}`：ブロックに記述します．
- `parameters{}`：母数の変数宣言を行います．母数の変換に関するステートメントの記述は次の `transformed parameters` ブロックに記述します．
- `transformed parameters{}`：本ブロックでは母数変換のステートメントが評価されます．サンプリングは行われません．サンプリングが行われるごとに本ブロックで記述したステートメントが実行されます．

- `model{}`：モデル分布を記述します．本ブロックにステートメントがあってもかまいません．
- `generated quantities{}`：本ブロックで記述されたステートメントはリープフロッグが終わった後に計算されます．新しいデータによる予測値の計算，事後期待値の計算，出力に利用するための母数の変形が行えます．`transformed parameters` ブロックに置く必然性のないステートメントについても，本ブロックに記述することができます．
- ある場所で宣言した変数のスコープは，後続するブロックすべてに渡ります．反対に，変数宣言よりも先に当該変数の操作を記述することは許されません．たとえば `transformed data` において宣言，変換した変数は `model` ブロックで使用できます．一方，`generated quantities` ブロックで宣言した変数を，それ以前のブロックで用いることはできません．

各ブロックの中括弧内に具体的なコードを記述します．`model{}`以外の不必要なブロックは省略することもできますが，順番を入れ替えることはできません．必ず上記の順に記述しなければならないことに注意してください．

C.4.2 変 数 宣 言

変数の宣言では以下に示す型の種類に従い，各変数の型を指定します．

基本データ型	コード	説明
要素	`int`	整数値
	`real`	連続値
ベクトル・行列	`vector`	列ベクトル
	`row_vector`	行ベクトル
	`matrix`	行列
配列 (array)	`x[2]`	サイズ 2 の 1 次元配列 `x`
	`matrix[2,2] x[2,3]`	サイズ 2×2 の行列を要素として含むサイズ 2×3 の 2 次元配列 `x`
制約データ型		
構造ベクトル	`ordered`	昇順に順序づけられた値を要素として持つベクトル
	`positive_ordered`	昇順に順序づけられた正の値を要素として持つベクトル
	`simplex`	総和が 1 となる非負の値を要素として持つベクトル
	`unit_vector`	要素の 2 乗和が 1 となる値を要素として持つベクトル
構造行列	`corr_matrix`	対角要素が 1 である対称正定値行列
	`cov_matrix`	対称正定値行列
	`cholesky_factor_cov`	対称正定値行列のコレスキー因子分解における因子行列であり，正の対角要素を持つ下三角行列
	`cholesky_factor_corr`	相関行列のコレスキー因子分解における因子行列であり，正の対角要素を持つ下三角行列

制約データ型についても配列型として与えることができます．また，基本データ型を制約することで制約データ型と同じ形式のデータ型を与えることも可能です．

ベクトル・行列・配列 Stan には複数の値を格納できる型としてベクトル型，行列型，配列型があります．ベクトル型，行列型内のデータは実数値として扱われます．配列型は，その内部に整数値，実数値，ベクトル型，行列型を保持させることができます．

1 つの (数学的意味での) 行列に対して，Stan では行列型や配列型，行ベクトルを縦に配列する等，複数の方法で表現することができます．ただし，行列間の数学的演算子 (行列どうしの掛け算等)，線形代数関数 (固有値や行列式の算出) の適用，多変量関数の母数等 (たとえば多変量正規分布の引数) の場合にはベクトル・行列型を使用する必要があります．

「研修効果問題」の Stan コードの場合，22 行目の `multi_normal()` の引数に与えるために，6 行目 `mu` や 12 行目 `Sigma` で行われている変数定義において，配列型を用いることはできませんので注意してください．

たとえば `multi_normal(mu, Sigma)` の引数に与える変数 `mu`，`Sigma` について
平均ベクトル `mu` に関して
```
  ベクトル型 ○  vector[2] mu;
    配列型 ×  real mu[2];
```
共分散行列 `Sigma` に関して
```
    行列型 ○  matrix[2,2] Sigma;
    配列型 ×  real Sigma[2,2]
```

変数宣言における型・サイズ・変数名の記述順序

- 要素 (`int`・`real`) 型は「型・ (範囲) ・変数名」の順に，たとえば `int x;` と記述します．
- 配列型は「型・ (範囲) ・変数名・サイズ」の順に，`int x[10];` と記述します．
- ベクトル・行列型の場合には「型・ (範囲) ・サイズ・変数名」の順に，
  `vector<lower=-3,upper=3>[n] x;` と記述します．

C.4.3 主な Stan 組込み関数

Stan コード内で用いられる関数を以下に示しましたので，適宜参照してください [*6)]．

関数のはじめについている変数型は，関数の返り値の型を表しています．また，返り値については，返り値 1, 条件 1; 返り値 2, 条件 2 の順に記してあります．なお，実際に使用する際には変数型の記述は必要ありません．関数内の条件式に対しては，値を小数形式で与える必要があることに注意してください．

返り値型 関数 (引数)	返り値 1, 条件 1; 返り値 2, 条件 2
絶対値関数	
`int int_step(int(または real) x)`	$0, x \leq 0; 1, x > 0$
境界関数	
`int min(int x, int y)`	$x, x < y; y$, それ以外
`int max(int x, int y)`	$x, x > y; y$, それ以外

[*6)] 紙数の都合上，すべての関数を掲載してはいません．割愛した関数については Stan Reference を参照してください．

数学的定数	引数をとらないことに注意
`real pi()`	円周率
`real e()`	自然対数の底
`real sqrt2()`	平方根の 2 乗
`real log2()`	2 の自然対数
`real log10()`	10 の自然対数
論理関数	
`int < (int(real)), (x<y)`	x が y 未満 $(x < y)$ ならば 1 を，それ以外には 0 を返す.
`int <=(int(real)), (x<=y)`	$1, x \leq y; 0$, それ以外
`int > (int(real)), (x>y)`	$1, x > y; 0$, それ以外
`int >=(int(real)), (x>=y)`	$1, x \geq y; 0$, それ以外
`int ==(int(real)), (x==y)`	$1, x = y; 0$, それ以外
`int !=(int(real)), (x!=y)`	$1, x \neq y; 0$, それ以外
ブーリアン演算子	
`int !(int(real) x), (!x)`	$0, x \neq 0; 1, x = 0$
`int &&(int(real) x, y),(x&&y)`	$1, x \neq 0$ かつ $y \neq 0; 0$, それ以外
`int ||(int(real) x, y), (x||y)`	$1, x \neq 0$ または $y \neq 0; 0$, それ以外
短絡版ブーリアン演算子	
`real if_else(int 条件式, real x, real y)` `if_else(a,b,c)` は `a ? b : c` へと記法が変更されました．例えば `if_else(x>2,1,0)` は `x>2 ? 1 : 0` となります.	x, 条件式 $\neq 0; y$, それ以外
`real step(real x)`	$1, x > 0; 0$, それ以外
`int is_inf(real x)`	x が $\pm\infty$ ならば 1 を，それ以外ならば 0 を返す.
`int is_nan(real x)`	x が NaN ならば 1 を，それ以外ならば 0 を返す.
絶対値関数	
`real fabs(real x)`	x の絶対値 $\|x\|$
`real fdim(real x, real y)`	$x - y, x \geq y; 0$, それ以外
境界関数	
`real fmin(real x, real y)`	$x, x \leq y; y$, それ以外
`real fmax(real x, real y)`	$x, x \geq y; y$, それ以外
算術関数	
`real fmod(real x, real y)`	x を y で割った際の余りの値を返す. $x - \lfloor (x/y) \rfloor y$, ($\lfloor \cdot \rfloor$ は床関数を表す)
丸め関数	
`real floor(real x)`	x について床につけた値を返す.
`real ceil(real x)`	x について天井につけた値を返す.
`real round(real x)`	x について四捨五入した値を返す. $\lceil x \rceil, x - \lfloor x \rfloor \geq 0.5, \lfloor x \rfloor$, それ以外
累乗および対数関数	
`real sqrt(real x)`	$\sqrt{x}$
`real cbrt(real x)`	$\sqrt[3]{x}$
`real square(real x)`	x^2
`real exp(real x)`	e^x
`real exp2(real x)`	2^x
`real log(real x)`	$\log_e(x)$
`real log2(real x)`	$\log_2(x)$

`real inv(real x)`	x^{-1}
`real inv_sqrt(real x)`	$\sqrt{x}^{-1}$
`real inv_square(real x)`	$(x^2)^{-1}$
三角関数	
`real hypot(real x, real y)`	2 辺の長さが x と y である直角三角形の斜辺の長さを返す. $\sqrt{x^2 + y^2}, (x, y) \geq 0$, NaN, それ以外
`real cos(real x)`	$\cos(x)$
`real sin(real x)`	$\sin(x)$
`real tan(real x)`	$\tan(x)$
`real acos(real x)`	$\arccos(x), -1 \leq x \leq 1$, NaN, それ以外
`real asin(real x)`	$\arcsin(x), -1 \leq x \leq 1$, NaN, それ以外
`real atan(real x)`	$\arctan(x)$
`real atan2(real x, real y)`	$\arctan(x/y)$
双曲線三角関数	
`real cosh(real x)`	$\cosh(x)$
`real sinh(real x)`	$\sinh(x)$
`real tanh(real x)`	$\tanh(x)$
`real acosh(real x)`	$\cosh^{-1}(x), x \geq 1$, NaN, それ以外
`real asinh(real x)`	$\sinh^{-1}(x)$
`real atanh(real x)`	$\tanh^{-1}(x), -1 \leq x \leq 1$, NaN, それ以外
リンク関数	
`real logit(real x)`	$\log \frac{x}{1-x}, 0 \leq x \leq 1$
`real inv_logit(real y)`	$\frac{1}{1+\exp(-y)}$
`real inv_cloglog(real y)`	$1 - \exp(-\exp(y))$
組み合わせ関数	
`real lbeta(real alpha, real beta)`	$\log \Gamma(a) + \log \Gamma(b) - \log \Gamma(a + b)$
`real tgamma(real x)`	$\Gamma(x), x \notin \{\ldots, -3, -2, -1, 0\}$
`real lgamma(real x)`	$\log \Gamma(x), x \notin \{\ldots, -3, -2, -1, 0\}$
`real digamma(real x)`	$\Gamma'(x)/\Gamma(x), x \notin \{\ldots, -3, -2, -1, 0\}$
`real trigamma(real x)`	$\sum_{n=0}^{\infty}(1/(x+n)^2)$, $x \notin \{\ldots, -3, -2, -1, 0\}$
`real lmgamma(int n, real x)`	$\pi^{n(n-1)/4} \prod_{j=1}^{n} \Gamma(x + (1-j)/2)$, $x \notin \{\ldots, -3, -2, -1, 0\}$
合成関数	
`real expm1(real x)`	$e^x - 1$
`real fma(real x, real y, real z)`	$x \times y + z$
`real multiply_log(real x, real y)`	$0, (x = y = 0); x \log y, (x, y \neq 0)$
`real log1p(real x)`	$\log(1 + x), x \geq -1$
`real log1m(real x)`	$\log(1 - x), x \leq 1$
`real log1p_exp(real x)`	$\log(1 + \exp(x))$
`real log1m_exp(real x)`	$\log(1 - \exp(x)), x < 0$
`real log_diff_exp(real x, real y)`	$\log(\exp(x) - \exp(y)), x > y$
`real log_sum_exp(real x, real y)`	$\log(\exp(x) + \exp(y))$
`real log_inv_logit(real x)`	$\log \text{logit}^{-1}(x)$
`real log1m_inv_logit(real x)`	$\log(1 - \text{logit}^{-1}(x))$

C.5 RStan の主な関数と引数

ここでは本書で用いられる RStan パッケージの主な関数と引数について概要を示します．

```
stan(file, model_name = "anon_model", model_code = "",
    fit = NA, data = list(), pars = NA, chains = 4,
    iter = 2000, warmup = floor(iter/2), thin = 1,
    init = "random", seed = sample.int(.Machine$integer.max, 1),
    algorithm = c("NUTS", "HMC", "Fixed_param"),
    control = NULL, sample_file = NULL, diagnostic_file = NULL,
    save_dso = TRUE, verbose = FALSE, include = TRUE,
    cores = getOption("mc.cores", 1L), ..., boost_lib = NULL, eigen_lib = NULL)
```

Stan モデリング言語によって定義されたモデルの当てはめを行う．

file Stan モデリング言語によって特定されたモデルコードを記述したファイル名を (ディレクトリも含めて) 指定する．モデルコードを文字列として直接与える場合には `model_name` 引数へ指定する．`fit` に対して以前の当てはめ結果を指定した場合には，本引数は無視される．

model_name モデルを名付けるための文字列．既定値は`"anon_model"`となる．ただし `model_name` が指定されないときは，`file` や `model_code` から命名される．

model_code Stan モデリング言語によって特定されたモデルの文字列，もしくはワークスペース内の文字列オブジェクト．本引数は `file` が指定されていないときのみ使用される．`fit` が特定されている場合，以前にコンパイルされたモデルが用いられるため，本引数は無視される．

fit 以前の当てはめ結果から得られたインスタンス (オブジェクト)．もし `NA` でなければ，その当てはめ結果に関連してコンパイルされたモデルが再利用される．本引数を利用することで C++コードを再コンパイルする時間が節約可能となる．

data データとして用意されたリストオブジェクト．あるいはワークスペース内にあるデータとして用いるオブジェクト名の文字列ベクトル．

pars 関心のある母数を特定した文字列ベクトル．既定値は `NA` であり，モデル内の全母数を指定したものとして扱われる．本引数に与えた母数に関してのみ，結果のオブジェクトにサンプリング結果が保存される．

chains 連鎖構成数を指定する正の整数．既定値は 4.

iter 各チェーンで繰り返しを何回行うかを指定する正の整数 (ウォームアップ期間を含んだ数である)．既定値は 2000 回．

warmup ウォームアップ期間の数を指定する正の整数．ウォームアップ期間のサンプルは推測に用いられない．`iter` に指定した数よりも大きな数は指定できない．既定値は `iter/2`.

thin サンプリングを保存する間隔を指定する正の整数．既定値は 1.

init 初期値を指定する．`"0"`: 初期値はすべて 0 とする．`"random"`: Stan によってランダムに生成される．その際，乱数の種は引数 `seed` に与えた値によって指定可能である．`seed` を固定した場合は初期値は同じ値が用いられる．`"list"`: 初期値を与えたリスト．

seed Stan の乱数を生成するために，正の整数を与える．既定では，1 から R でサポートされている最大整数値までが生成される．そのため，R の乱数生成器の種の固定により，本質的に Stan の種を固定することができる．複数の連鎖を構成する場合でも 1 つの種を与えればよく，他の連鎖の種は，乱数間の依存性を防ぐために，最初の連鎖の種から生成される．数値を与えた場合，`as.integer` が適用される．`as.integer` の適用結果が `NA` を返す場合，種はランダムに生成される．種の数字を文字列として与えることもでき，その場合は整数に変換される．

algorithm Stan 内で用いられるアルゴリズム．NUTS 法，もしくは静的 HMC 法か `Fixed_param` を指定する．

sample_file 母数のサンプルを出力するためのファイルの名前を与える文字列．引数に文字列を与えない場合，ファイルは作成されない．指定されたフォルダーが書き込み可能でない場合，`tempdir()` が使用される．複数連鎖がある場合，アンダースコアとチェーンの番号がファイル名に追加される．

```
print(x, pars = x@sim$pars_oi,
    probs = c(0.025, 0.25, 0.5, 0.75, 0.975),
    digits_summary = 2, ...)
```

当てはめられたモデルと関心があるサンプリングされた母数についての要約と基本情報を出力する．

```
plot(x, pars, display_parallel = FALSE,
    ask = TRUE, npars_per_page = 6)
```

当てはめられたモデルに関する母数の点推定値と確信区間を描画する．

```
traceplot(object, pars, inc_warmup = TRUE, ask = FALSE,
    nrow = 4, ncol = 2, window = NULL, ...)
```

トレースプロットを描画する．

C.6 分 布 一 覧

主要な分布を一覧表として掲載しましたので，必要に応じて参照してください [*7]．分布名・サンプリングのためのステートメント・変数型の定義を記しています [*8]．

[*7] 分布関数の表記に関しては原則として Stan reference 2.12.0 に従っています．そのため第 8 章までに登場した分布関数の表記と一部異なっている場合がありますので注意してください．

[*8] $\mathbb{N}$: 整数，$\mathbb{R}$: 実数，$\mathbb{R}^+$: 正の実数．

ベルヌイ分布 `y ~ bernoulli(theta);`

$$Bern(y|\theta) = \begin{cases} \theta & y = 1 \\ 1-\theta & y = 0 \end{cases}, \theta \in [0,1], y \in \{0,1\}$$

ロジット変換ベルヌイ分布 `y ~ bernoulli_logit(alpha);`

$$BernLogit(c|\alpha) = Bern(c|logit^{-1}(\alpha)) = \begin{cases} logit^{-1}(\alpha) & y = 1 \\ 1 - logit^{-1}(\alpha) & y = 0 \end{cases},$$

$\alpha \in \mathbb{R}$, $c \in \{0,1\}$

2 項分布 `n ~ binomial(N,theta);`

$Binom(n|N,\theta) = \binom{N}{n}\theta^n(1-\theta)^{N-n}$, $N \in \mathbb{N}$, $\theta \in [0,1]$, $n \in \{0,\ldots,N\}$

ロジット変換 2 項分布 `n ~ binomial_logit(N,alpha);`

$BinomLogit(n|N,\alpha) = Binom(n|N, logit^{-1}(\alpha))$

$= \binom{N}{n}(logit^{-1}(\alpha))^n(1-logit^{-1}(\alpha))^{N-n}$, $N \in \mathbb{N}$, $\alpha \in \mathbb{R}$, $n \in \{0,\ldots,N\}$

ベータ–2 項分布 `n ~ beta_binomial(N,alpha,beta);`

$BetaBinom(n|N,\alpha,\beta) = \binom{N}{n}\frac{B(n+\alpha,N-n+\beta)}{B(\alpha,\beta)}$, $N \in \mathbb{N}$, $\alpha \in \mathbb{R}^+$, $\beta \in \mathbb{R}^+$,

$n \in \{0,\ldots,N\}$

超幾何分布 `n ~ hypergeometric(N,a,b);`

$HG(n|N,a,b) = \frac{\binom{a}{n}\binom{b}{N-n}}{\binom{a+b}{N}}$, $a \in \mathbb{N}$, $b \in \mathbb{N}$, $N \in \{0,\ldots,a+b\}$,

$n \in \{\max(0,N-b),\ldots,\min(a,N)\}$

カテゴリカル分布 `categorical(theta);`

$Cat(y|\theta) = \theta_y$, $N \in \mathbb{N}$, $N > 0$, $\theta \in \mathbb{R}^N$が N 次シンプレックス, $y \in \{1,\ldots,N\}$

順序ロジスティック分布 `k ~ ordered_logistic(eta,c);`

$$OrdLogis(k|\eta,c) = \begin{cases} 1 - logit^{-1}(\eta - c_1) & k = 1 \\ logit^{-1}(\eta - c_{k-1}) - logit^{-1}(\eta - c_k) & 1 < k < K \\ logit^{-1}(\eta - c_{K-1}) - 0 & k = K \end{cases}$$

$K \in \mathbb{N}$, $K > 2$, $c \in \mathbb{R}^{K-1}$, $c_k < c_{k+1}$ for $k \in \{1,\ldots,K-2\}$, $\eta \in \mathbb{R}$, $k \in \{1,\ldots,K\}$, $k = 1$, $k = K$ の場合, $logit^{-1}(-\infty) = 0$, $logit^{-1}(\infty) = 1$ であるような $c_0 = -\infty$, $c_K = +\infty$ という設定によって一般的定義に含ませることができる.

負の 2 項分布 (再母数化) `y ~ neg_binomial_2(mu,phi);`

$NegBinom2(y|\mu,\phi) = \binom{y+\phi-1}{y}\left(\frac{\mu}{\mu+\phi}\right)^y\left(\frac{\phi}{\mu+\phi}\right)^\phi$,

$\eta \in \mathbb{R}$, $\mu \in \mathbb{R}^+$, $\phi \in \mathbb{R}^+$, $y \in \mathbb{N}$, $E[Y] = \mu = e^\eta$, $V[Y] = \mu + \frac{\mu^2}{\phi}$

ポアソン分布 `n ~ poisson(lambda);`

$Poi(n|\lambda) = \frac{1}{n!}\lambda^n\exp(-\lambda)$, $\lambda \in \mathbb{R}^+$, $n \in \mathbb{N}$

多項分布 `y ~ multinomial(theta);`

$MultiNom(y|\theta) = \binom{N}{y_1,\ldots,y_K}\prod_{k=1}^K\theta_k^{y_k}$, $\binom{N}{y_1,\ldots,y_K} = \frac{N!}{\prod_{k=1}^K y_k!}$,

$K \in \mathbb{N}$, $N \in \mathbb{N}$, θ K 次シンプレックス, $y \in \mathbb{N}^K$, $\sum_{k=1}^K y_k = N$

正規分布 `y ~ normal(mu,sigma);`

$N(y|\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right)$, $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$, $y \in \mathbb{R}$

非対称正規分布 `y ~ skew_normal(mu,sigma,alpha);`

$SkewN(y|\mu,\sigma,\alpha) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right)\left(1+\mathrm{erf}\left(\alpha\left(\frac{y-\mu}{\sigma\sqrt{2}}\right)\right)\right)$, $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$,

$\alpha \in \mathbb{R}$, $y \in \mathbb{R}$

t 分布 `y ~ student_t(nu,mu,sigma);`

$t(y|\nu,\mu,\sigma) = \frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)}\frac{1}{\sqrt{\nu\pi}\sigma}\left(1+\frac{1}{\nu}\left(\frac{y-\mu}{\sigma}\right)^2\right)^{-(\nu+1)/2}$, $\nu \in \mathbb{R}^+$, $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$,

$y \in \mathbb{R}$

コーシー分布 `y ~ cauchy(mu,sigma);`

$Cauchy(y|\mu,\sigma) = \frac{1}{\pi\sigma}\frac{1}{1+((y-\mu)/\sigma)^2}$, $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$, $y \in \mathbb{R}$

2 重指数分布 `y ~ double_exponential(mu,sigma);`

$DExp(y|\mu,\sigma) = \frac{1}{2\sigma}\exp\left(-\frac{|y-\mu|}{\sigma}\right)$, $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$, $y \in \mathbb{R}$

ロジスティック分布 `y ~ logistic(mu,sigma);`

$Logis(y|\mu,\sigma) = \frac{1}{\sigma}\exp\left(-\frac{y-\mu}{\sigma}\right)\left(1+\exp\left(-\frac{y-\mu}{\sigma}\right)\right)^{-2}$, $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$, $y \in \mathbb{R}$

ガンベル分布 `y ~ gumbel(mu,beta);`

$Gumbel(y|\mu,\beta) = \frac{1}{\beta}\exp\left(-\frac{y-\mu}{\beta} - \exp\left(-\frac{y-\mu}{\beta}\right)\right)$, $\mu \in \mathbb{R}$, $\beta \in \mathbb{R}^+$, $y \in \mathbb{R}$

対数正規分布 `y ~ lognormal(mu,sigma);`

$LogN(y|\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\frac{1}{y}\exp\left(-\frac{1}{2}\left(\frac{\log y-\mu}{\sigma}\right)^2\right)$, $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$, $y \in \mathbb{R}^+$

カイ 2 乗分布 `y ~ chi_square(nu);`

$\chi^2(y|\nu) = \frac{2^{-\nu/2}}{\Gamma(\nu/2)}y^{\nu/2-1}\exp\left(-\frac{1}{2}y\right)$, $\nu \in \mathbb{R}^+$, $y \in \mathbb{R}^+$

逆カイ 2 乗分布 `y ~ inv_chi_square(nu);`

$Inv\chi^2(y|\nu) = \frac{2^{-\nu/2}}{\Gamma(\nu/2)}y^{-(\nu/2-1)}\exp\left(-\frac{1}{2}\frac{1}{y}\right)$, $\nu \in \mathbb{R}^+$, $y \in \mathbb{R}^+$

指数分布 `y ~ exponential(beta);`

$Exp(y|\beta) = \beta\exp(-\beta y)$, $\beta \in \mathbb{R}^+$, $y \in \mathbb{R}^+$

ガンマ分布 `y ~ gamma(alpha,beta);`

$G(y|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}y^{\alpha-1}\exp(-\beta y)$, $\alpha \in \mathbb{R}^+$, $\beta \in \mathbb{R}^+$, $y \in \mathbb{R}^+$

逆ガンマ分布 `y ~ inv_gamma(alpha,betae);`

$InvG(y|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}y^{-(\alpha+1)}\exp\left(-\beta\frac{1}{y}\right)$, $\alpha \in \mathbb{R}^+$, $\beta \in \mathbb{R}^+$, $y \in \mathbb{R}$

ワイブル分布 `y ~ weibull(alpha,sigma);` $Weib(y\|\alpha,\sigma) = \frac{\alpha}{\sigma}\left(\frac{y}{\sigma}\right)^{\alpha-1}\exp\left(-\left(\frac{y}{\sigma}\right)^{\alpha}\right)$, $\alpha\in\mathbb{R}^+$, $\sigma\in\mathbb{R}^+$, $y\in[0,\infty)$
パレート分布 `y ~ pareto(y_min,alpha);` $Pareto(y\|y_{\min},\alpha) = \frac{\alpha y_{\min}^{\alpha}}{y^{\alpha+1}}$, $y_{\min}\in\mathbb{R}^+$, $\alpha\in\mathbb{R}^+$, $y\in\mathbb{R}^+$, $y\geq y_{\min}$
ベータ分布 `theta ~ beta(alpha,beta);` $Beta(\theta\|\alpha,\beta) = \frac{1}{B(\alpha,\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1}$, $\alpha\in\mathbb{R}^+$, $\beta\in\mathbb{R}^+$, $\theta\in(0,1)$, 注意 :$\theta=1$ もしくは $\theta=1$ の場合，確率は 0 となり対数確率は $-\infty$ となる．よって正の母数 $\alpha,\beta>0$ をとらなければならない．
一様分布 `y ~ uniform(alpha, beta);` $Unif(y\|\alpha,\beta) = \frac{1}{\beta-\alpha}$, $\alpha\in\mathbb{R}$, $\beta\in(\alpha,\infty)$, $y\in[\alpha,\beta]$
多変量正規分布 `y ~ multi_normal(mu,Sigma);` $MN(y\|\mu,\Sigma) = \frac{1}{(2\pi)^{K/2}}\frac{1}{\sqrt{\|\Sigma\|}}\exp\left(-\frac{1}{2}(y-\mu)'\Sigma^{-1}(y-\mu)\right)$, $K\in\mathbb{N}$, $\mu\in\mathbb{R}^K$, 対称正定値行列 $\Sigma\in\mathbb{R}^{K\times K}$, $y\in\mathbb{R}^K$
多変量 t 分布 `y ~ multi_student_t(nu,mu,Sigma)` $Mt(y\|\nu,\mu,\Sigma) = \frac{1}{\pi^{K/2}}\frac{1}{\nu^{K/2}}\frac{\Gamma\chi((\nu+K)/2)}{\Gamma(\nu/2)}\frac{1}{\sqrt{\|\Sigma\|}}\left(1+\frac{1}{\nu}(y-\mu)'\Sigma^{-1}(y-\mu)\right)^{-(\nu+K)/2}$, $K\in\mathbb{N}$, $\nu\in\mathbb{R}^+$, $\mu\in\mathbb{R}^K$, 対称正定値行列 $\Sigma\in\mathbb{R}^{K\times K}$, $y\in\mathbb{R}^K$
ディリクレ分布 `theta ~ dirichlet(alpha);` $Dirchlet(\theta\|\alpha) = \frac{\Gamma(\sum_{k=1}^K\alpha_k)}{\prod_{k=1}^K\Gamma(\alpha_k)}\prod_{k=1}^K\theta_k^{\alpha_{k-1}}$, $K\in\mathbb{N}$, $\alpha\in(\mathbb{R}^+)^K$, $\theta\in K$ 次シンプレックス 注意 : θ の任意の成分が $\theta_i=0$ もしくは $\theta_i=1$ を満たす場合，確率は 0 となり，対数確率は $-\infty$ となる．同様に各 i について $\alpha_i>0$ であるような正の母数が必要となる．
ウィッシャート分布 `W ~ wishart(nu,Sigma);` $Wish(W\|\nu,S) = \frac{1}{2^{\nu K/2}}\frac{1}{\Gamma_K\left(\frac{1}{2}\right)}\|S\|^{-\nu/2}\|W\|^{(\nu-K-1)/2}\exp\left(-\frac{1}{2}\mathrm{tr}(S^{-1}W)\right)$, $K\in\mathbb{N}$, $\nu\in(K-1,\infty)$, 対称正定値行列 $S\in\mathbb{R}^{K\times K}$, 対称正定値行列 $W\in\mathbb{R}^{K\times K}$ であり，$\Gamma_K()$ は多変量ガンマ関数 $\Gamma_K(\chi) = \frac{1}{\pi^{K(K-1)/4}}\prod_{k=1}^K\Gamma\left(\chi+\frac{1-k}{2}\right)$ である．
逆ウィッシャート分布 `W ~ inv_wishart(nu,Sigma);` $InvWish(W\|\nu,S) = \frac{1}{2^{\nu K/2}}\frac{1}{\Gamma_K\left(\frac{1}{2}\right)}\|S\|^{\nu/2}\|W\|^{-(\nu-K-1)/2}\exp\left(-\frac{1}{2}\mathrm{tr}(SW^{-1})\right)$, $K\in\mathbb{N}$, $\nu\in(K-1,\infty)$, 対称正定値行列 $S\in\mathbb{R}^{K\times K}$, 対称正定値行列 $W\in\mathbb{R}^{K\times K}$

C.7 練 習 問 題

1）★ 以下に示す 20 個の対数変換された値 $y=\ln x$ の真数 x に対して正規分布モデルを適用し，平均，標準偏差，分散について，Stan (Stan コード・データファイル・RStan スクリプトを作成) を用いて推定し，値を報告してください．なお，データと母数の変換も Stan

内で行ってください．事前分布は明示的に設定しなくて結構です．連鎖更新回数は 10000 回，ウォームアップ期間 1000 回，乱数の種は 12345，連鎖構成数は 1 とします．補足：データの変換は `transformed data` 内で関数 `exp()` を使用してください．標準偏差の変換は `transformed parameters` 内で `pow()` を使用します．サンプリングは `normal(平均,SD)` を用いて行います．データファイルは次の内容を用いてください．データファイル:`N<-20;x_log<-c(3.844,4.035,3.986,4.030,3.932,4.040,3.788,3.896,3.929,4.072,3.477,3.882,3.895,3.908,3.945,3.980,4.028,4.137,3.930,4.016);`

2）★ 問題 C.7.1 について，標準偏差の変換を `generated quantities` ブロックで行ってください．

3）★ 問題 C.7.1 について，連鎖更新回数 50 回，ウォームアップ期間を 10 回とした場合の推定値を報告してください．また，連鎖更新回数を 20000 回 (ウォームアップ期間 1000 回) に増やした場合の推定値を報告してください．これらの更新期間は十分といえるか検討してください．

練習問題 解答例

1）Stan コード例

```
data{
int<lower=0> N;
vector<lower=0>[N] log_x;
}
transformed data{
vector[N] x;
for(n in 1:N){
x[n] = exp(log_x[n]);
}
}
parameters{
real mu;
real<lower=0> sigma;
}
transformed parameters{
real<lower=0> sig2;
sig2 = pow(sigma,2);
}
model{
for(n in 1:N){
x[n] ~ normal(mu,sigma);
}
}
```

2）Stan コード例

```
data{
int<lower=0> N;
vector<lower=0>[N] log_x;
}
transformed data{
vector[N] x;
for(n in 1:N){
x[n] = exp(log_x[n]);
}
}
parameters{
real mu;
real<lower=0> sigma;
}
model{
for(n in 1:N){
x[n] ~ normal(mu,sigma);
}
}
generated quantities{
real<lower=0> sig2;
sig2 = pow(sigma,2);
}
```