**Online materials**

WinBUGSを使ったBayesianメタアナリシス

「診療ガイドラインのためのGRADEシステム（第2版）」（相原、凸版メディア、2015年出版）のオンライン追加資料です。<http://www.grade-jpn.com/online_supplementals/online_supplemental_materials.html>

* 詳細は書籍の**4.4.3章**を参照ください。
* 本資料のファイル名：Bayesian\_meta\_heparin.docx（相原守夫、2014年7月作成）
* 以下の内容は、更新される可能性がありますので、上記URLに表示する年月日を参考にしてください。

\*BUGSは Bayesian inference using Gibbs Samplingの略で、ベイズ統計解析を行うためのソフトウエアである。現在は、OpenBUGSへの移行が進行している。

|  |  |
| --- | --- |
| Bayesianメタアナリシスの手順 |  |
| ＊ WinBUGSのインストール | <http://www.mrc-bsu.cam.ac.uk/bugs/winbugs/contents.shtml>より、WinBUGS1.4.3インストールする。WinBUGS14.exe,　free keyファイル、Patch for 1.4.3（バージョン1.43へのアップグレードのための）をダウンロードする。 |
| 1．データの準備-方法WinBUGSのファイル名には拡張子として、odcが付けられるが、メタアナリシスのための基本データは、テキストエディタで作成可能である。ここでは、以下のデータを利用する。RR\_scale\_with\_prediction\_Beysian-heparin.txtFile>Open>RR\_scale\_with\_prediction\_Beysian-heparin.txt | ＊K個のRCTで、対照群の症例数ncとその内のアウトカム陽性の症例数ｒc、介入群の症例数nTとその内のアウトカム陽性の症例数rTが表のように分かっている場合のランダム効果モデルによるベイジアンメタアナリシスをWinBUGSで行う。

|  |  |  |  |
| --- | --- | --- | --- |
|  | Outcome (+) | Outcome (-) | Sum |
| Treated:Intervention (+) | ri T | niT - riT | ni T |
| Control:Intervention (-) | riC | niC-riC | niC |

(iは研究番号）　　　　8個の研究で、研究の番号順に並べたデータを解析する。*list(k=8,rc=c(155,30,109,107,110,32,41,10),nc=c(381,42,184,154,139,87,69,10),rt=c(333,22,103,88,96,45,40,8),nt=c(769,42,190,148,138,99,68,10)* |
| DATA #k is the number of studies.list(k=8,rc=c(155,30,109,107,110,32,41,10),nc=c(381,42,184,154,139,87,69,10),rt=c(333,22,103,88,96,45,40,8),nt=c(769,42,190,148,138,99,68,10) |  |
| ２．モデルの構文をチェックする[Modelメニュー]⇒Specification tool → check model | Modelメニューから、Specificationを選択し、Specification toolを表示して、次に、コードのmodelと書いてある部分をダブルクリックして反転して選択した状態にしてから、“check model”ボタンをクリックする。モデルに問題がなければ、ウインドウの左下に model is syntactically correctと表示される。問題があれば、カーソルがその位置に移動し、エラーの内容に応じたメッセージが表示される。 |
| 3. データをロードする[Modelメニュー]⇒Specification tool → load data | データのロード（読み込み）には、まずデータの頭の部分のlistと記述している部分をダブルクリックして反転させ選択した状態にする。その上で、load dataボタンをクリックする。データがうまくロードされるとやはりウインドウの左下枠に data loadedと表示される。 |
| ４．モデルのコンパイル[Modelメニュー]⇒Specification tool → compile | ＊コンパイルというのは、実際に実行されるコンピュータプログラムをモデルとデータに基づいて作成する作業である。コンパイルが成功すると、同様に左下にmodel compileが表示される。 |
| ５．初期値を設定する[Modelメニュー]⇒Specification tool → load initsまたは/およびgen inits | 初期値として、それぞれの分布からランダムに引き出した値を設定するようなコードの場合には、list(～～～～）以下のデータ部分を反転させて、**gen inits**ボタンをクリックする。初期値の設定が成功すると、initial values generated, model initializedというメッセージが表示される。また，「Specification Tool」の「gen inits」が使用不可能になる。*注：このコードでは、delt, tauは事前分布が設定され、そこからランダムに値を得るように記述されており、gen initsでそれらの分布からランダムに得られた値が初期値として使われる。deltはdnorm(0,01)でほぼフラットな分布、tauはdunif(0,2)で0から2までのフラットな分布である。これらは対数スケール上での値で、いずれもnon-informativeなほぼフラットな分布である。もし、以前の研究で事前分布が想定できる場合は、それに合わせて値を変えることもできる。* |
| *ここまでで、MCMCを走らせる準備が完成したことになる。* |
| ６．MCMC実行の準備\*[Modelメニュー] ⇒Update... → Update Tool\*[Inferenceメニュー]　⇒Samples　→Sample Monitoring Tool | ＊「Model」メニューから「Update...」を選択し，「Update Tool」ダイアログを表示させておく．＊また，「Inference...」メニューから「Samples...」を選択し，「Sample Monitoring Tool」ダイアログを表示させておく． |
| ７．モニターするnode（変数）の設定[Inferenceメニュー]⇒Samples → Sample Monitor tool → node → set*Nodeとして設定できるのは、delt, tau.squared, tau, RR, delt.new, RR.new, pdelt0, pdelt.new0*。 | 事後分布を見たい変数を設定する。Sample Monitor Toolを開いたら、nodeを設定する。すなわち、MCMCの過程でモニターし、記録する変数を設定する。nodeの名称は、コード中で使用された名称である。この例では、muと入力してsetボタンをクリックする（次に、delt, RR, RR.newなどと追加入力する）。 |
|  | \*「node」にアスタリスク「\*」を入力すると、「clear」「trace」「history」ボタンが使用できるようになる。 |
| ８．MCMCの実行[Modelメニュー]⇒Updates... → （Update回数設定）→ update●burn-in sampling | 「Sample Monitor Tool」の「trace」ボタンをクリックし，「Dynamic trace」ウインドウを表示させる。MCMCを実行するには、ModelメニューからUpdate..を選択し、MCMCを何回繰り返すかをUpdatesに設定する。＊Updates回数はデフォルトでは1000回になっているが、通常は**1万回**で十分である。Updatesの回数は追加される回数なので、最初の実行時には、その回数だけMCMCが実行され、iterationの数値がカウンターとして表示され、設定される回数で停止する。ここでは、thinは１、refreshは100のままで実行する。thinを例えば、整数Kに設定すると、updates回数ｘK回実行し、最後のものだけを記録する。「Update Tool」の「update」ボタンをクリックすると，サンプリングがおこなわれ，「Dynamic trace」にその状況が表示される．また，「Sample Monitor Tool」の「density」「stats」「coda」「qualities」「bgr diag」「auto cor」ボタンが使用できるようになる。\*「Dynamic trace」の状況を見て，burn-inが不十分である場合には，サンプリングを適宜追加する． |
| ●パラメータ推定 | 上記によって、nodeが設定されたので、「Update Tool」の、サンプリング回数を5万回にするため，「updates」に「50000」と入力する．「update」ボタンを押すとサンプリングが始まり，「Dynamic trace」ウインドウに経過が表示される。5万回として実行すると、iterationのカウンターは60000に変わる。モデルの複雑さにもよるが、正式の解析には5万回程度で十分である。＊このように、updateは少なくとも2回実行することになる。1回目のupdateはburn inと見なされるので、それだけではmu（とsigma）の事後分布を見ることはできないので注意が必要である。 |
| ９．結果を見る[Inferenceメニュー] ⇒Samples → Sample Monitor Tool　→ history, density, statsなど | ＊burn-inを除いた10001～60000回目のサンプルをもとにパラメータ推定をするため，「Sample Monitor Tool」の「beg」と「end」に「10001」と「60000」を入力する．\*「Sample Monitor Tool」のnodeを選択し、「history」ボタンをクリックすると，サンプリングの安定性を確認できる。＊「density」ボタンをクリックすると，パラメータ値のカーネル密度推定のチャートが表示される。\* stats」ボタンをクリックすると，各パラメータ推定値の平均や信用区間などが表示される．見たいnode（各パラメータ推定値）を選択して（例、RR）、「stats」ボタンをクリックすると、平均値、標準偏差、中央値、信頼区間（precentilesで設定できるがデフォルトでは95%信頼区間）、採用したMCMCの開始回数などが表示される。 |
| １０．結果とその解釈 | *●結果として見るべき変数はRRである。RRのdensityとstatで得られる中央値と95％信頼区間がいわゆる統合値とその信頼区間に相当する。**●Pdelt.new0のdensityとstatを見ることによって、将来の研究でRR＞1となる確率を知ることができる。この値は、deltとtau.squaredで規定される正規分布からのランダムサンプリングで得られるものである。* |
| ◆RR | RRは、ベイジアンメタアナリシスによる統合値。図10-1, RR＝0.923 (95%CI: 0.818～1.035）  |
| ◆pdelt0  | pdelt0は、RRが1.0を超える確率。図10-2, 確率： 0.069 |
| ◆相対リスクの予測値：RR.new | RR.newは、ランダム効果モデルメタアナリシスの統合値、分散（tau.squared）に基づいたリスク比の予測値が格納された変数。図10-3, RR=　0.923（95%信頼区間0.691～1.223) |
| ◆リスク比が1.0を超える確率 | Pdelt.new0は、将来の研究でRR＞1となる確率を示す。図10-4, 確率：　0.198注：将来の研究でリスク比が1.0を超える可能性は約20%、この確率は、図10-１に示した統合値の値と信頼区間から受ける印象とはかなり異なることがわかる。すなわち、信頼区間がより広く、不確実性が高い。このように、ランダム効果モデルは“予測”が一番堅牢な結果の提示法となる。 |