

The Impact of Covariance Priors on Arm-based Bayesian Network Meta-Analyses with Binary Outcomes

Supplementary Materials

Zhenxun Wang* Lifeng Lin[†] James S. Hodges* Haitao Chu*

A Definition of restricted Wishart distribution and its properties

In this appendix, we want to show the equivalence of restricted Wishart distribution and LKJ distribution Lewandowski et al. (2009) in some sense. We start with the definition of Wishart distribution. As we know, if $T \times 1$ vectors $\boldsymbol{\nu}_1, \boldsymbol{\nu}_2, \dots, \boldsymbol{\nu}_K$ are independent random variables that are multivariate normally distributed with mean $\mathbf{0}$ and variance $\boldsymbol{\Sigma}$, and we have a $T \times K$ matrix $\boldsymbol{\nu} = (\boldsymbol{\nu}_1, \boldsymbol{\nu}_2, \dots, \boldsymbol{\nu}_K)$, then the T -dimensional positive definite matrix $\mathbf{S} = \boldsymbol{\nu}\boldsymbol{\nu}'$ ($\boldsymbol{\nu}'$ is the transpose of $\boldsymbol{\nu}$) follows the Wishart distribution with degree of freedom $K > T - 1$ and positive definite scale matrix $\boldsymbol{\Sigma}$:

$$\mathbf{S} \sim W_T(K, \boldsymbol{\Sigma}) \equiv \{2^{\frac{1}{2}KT} \Gamma_T(\frac{1}{2}K)\}^{-1} |\boldsymbol{\Sigma}|^{-\frac{1}{2}K} |\mathbf{S}|^{\frac{1}{2}(K-T-1)} \exp(-\frac{1}{2}tr(\boldsymbol{\Sigma}^{-1}\mathbf{S})) \quad (1)$$

*Division of Biostatistics, School of Public Health, University of Minnesota, Minneapolis, MN 55455, USA

[†]Department of Statistics, Florida State University, Tallahassee, FL 32304, USA

where $|\bullet|$ is the determinant, tr is the trace, and Γ_T is the multivariate (T-variate here) gamma function. After that, we can formally define the restricted Wishart distribution based on the Wishart distribution:

Definition 1. A positive definite correlation matrix \mathbf{R} follows the restricted Wishart distribution $RW_T(m)$, if $\mathbf{R} = \mathbf{\Delta}\mathbf{\Sigma}\mathbf{\Delta}$, where $\mathbf{\Sigma} \sim W_T(m, \mathbf{\Psi})$ with $\mathbf{\Psi}$ being diagonal matrix with diagonal entries $\psi_{11}, \psi_{22}, \dots, \psi_{TT}$, $\mathbf{\Delta}$ is a diagonal matrix with the i^{th} diagonal element $\sigma_{ii}^{-1/2}$ and σ_{ii} is the i^{th} diagonal element of $\mathbf{\Sigma}$.

Theorem 1. If R follows the restricted Wishart distribution $RW_T(m)$, then

(a) the joint distribution of \mathbf{R} is:

$$f(\mathbf{R}) = b_T(m) |\mathbf{R}|^{\frac{m-T-1}{2}} \quad (2)$$

where

$$b_T(m) = \frac{\Gamma^T(\frac{m}{2})}{\Gamma_T(\frac{m}{2})} \quad (3)$$

is the normalizing constant. When $m = T + 1$, R becomes a jointly uniform distribution on a compact subspace of the $T(T - 1)/2$ dimensional hypercube $[-1, 1]^{T(T-1)/2}$.

(b) this normalizing constant $b_T(m)$ matches the following alternative formula given by Lewandowski et al. (2009):

$$c_T = 2^{\sum_{k=1}^{T-1} (m-k-1)(k-T)} \times \prod_{k=1}^{T-1} [B(\frac{m-k}{2}, \frac{m-k}{2})]^{k-T} \quad (4)$$

where $B(\bullet, \bullet)$ is the beta function.

(c) the marginal distribution of each element ρ_{ij} ($i \neq j$) in matrix \mathbf{R} is

$$f(\rho_{ij}) = \frac{\Gamma(\frac{m}{2})}{\Gamma(\frac{m-1}{2})\Gamma(\frac{1}{2})}(1 - \rho_{ij}^2)^{\frac{m-3}{2}}, \quad -1 \leq \rho_{ij} \leq 1 \quad (5)$$

which is exactly the $Beta(\frac{m-1}{2}, \frac{m-1}{2})$ distribution on $[-1, 1]$, and will be $Beta(\frac{T}{2}, \frac{T}{2})$ when $f(\mathbf{R}) = b_T(T + 1)$.

Proof of Theorem 1. (a). We first calculate the Jacobian matrix of the transformation $\Sigma \rightarrow (\sigma_{11}, \dots, \sigma_{TT}, \mathbf{R})$

$$\mathbf{J}_1 = \mathbf{J}(\sigma_{11} = \sigma_{11}, \dots, \sigma_{TT} = \sigma_{TT}, \rho_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}) = \left[\begin{array}{c|c} \mathbf{I}_T & 0 \\ \hline * & \mathbf{C} \end{array} \right] \quad (6)$$

where σ_{ij} is the entry of matrix Σ , \mathbf{I}_T is the T -dimensional identity matrix, \mathbf{C} is the $T(T-1)/2$ dimensional diagonal matrix with entries $1/\sqrt{\sigma_{ii}\sigma_{jj}}$ ($i \neq j$). Since J_1 is a lower triangular matrix, its determinant $|\mathbf{J}_1|$ equals $\prod_{i=1}^T \sigma_{ii}^{-(T-1)/2}$ and sub-matrix in (6) given as * need not be derived. Then, given $\Sigma \sim W_T(m, \Psi)$, we can derive the joint distribution of \mathbf{R} and $(\sigma_{11}, \dots, \sigma_{TT})$ is

$$\begin{aligned} f(\mathbf{R}, \sigma_{11}, \dots, \sigma_{TT}) &= f(\Sigma)|\mathbf{J}_1|^{-1} \\ &= \frac{\Gamma^T(\frac{m}{2})|\mathbf{R}|^{\frac{m-T-1}{2}}}{\Gamma_T(\frac{m}{2})} \prod_{i=1}^T \left\{ \frac{\sigma_{ii}^{\frac{m}{2}-1} \exp(-\frac{\sigma_{ii}}{\psi_{ii}})}{2^{\frac{m}{2}} \psi_{ii}^{\frac{m}{2}} \Gamma(\frac{m}{2})} \right\} \end{aligned} \quad (7)$$

Clearly, $\frac{\sigma_{ii}}{\psi_{ii}}, i = 1, \dots, T$ are independently distributed as chi-square with m degrees of freedom. Also, the density of \mathbf{P} is

$$f(\mathbf{R}) = \frac{\Gamma^T(\frac{m}{2})}{\Gamma_T(\frac{m}{2})} |\mathbf{R}|^{\frac{m-T-1}{2}} \quad (8)$$

(b). We just need to prove function $f(T, m) = b_T(m)/c_T$ equals 1 for any integer $T (\geq 2)$

and $m > T - 1$. Given $\Gamma_T(\frac{m}{2}) = \pi^{T(T-1)/4} \prod_{k=1}^T \Gamma(\frac{m}{2} + \frac{1-k}{2})$ (JamesJames (1964), p. 483), we can simplify $f(T, m)$ as

$$\begin{aligned}
f(T, m) &= \frac{\Gamma^T(\frac{m}{2})}{\Gamma_T(\frac{m}{2})} \prod_{k=1}^{T-1} [B(\frac{m-k}{2}, \frac{m-k}{2})]^{T-k} \times 2^{\sum_{k=1}^{T-1} (m-k-1)(T-k)} \\
&= \frac{\Gamma^T(\frac{m}{2})}{\pi^{T(T-1)/4} \Gamma(\frac{m}{2}) \prod_{k=1}^{T-1} \Gamma(\frac{m-k}{2})} \prod_{k=1}^{T-1} [\frac{\Gamma^2(\frac{m-k}{2})}{\Gamma(m-k)}]^{T-k} \times 2^{\sum_{k=1}^{T-1} (m-k-1)(T-k)} \quad (9) \\
&= \prod_{k=1}^{T-1} [2^{(m-k-1)(T-k)} \times \frac{\Gamma(\frac{m}{2})}{\pi^{T/4}} \frac{\Gamma^{2T-2k-1}(\frac{m-k}{2})}{\Gamma^{T-k}(m-k)}]
\end{aligned}$$

We prove that (9) equals 1 using mathematical induction. Start with $T = 2$, then for any $m > 1$, $f(2, m)$ reduces to

$$2^{m-2} \pi^{-\frac{1}{2}} \frac{\Gamma(\frac{m}{2}) \Gamma(\frac{m-1}{2})}{\Gamma(m-1)} \equiv 1 \quad (10)$$

which is known as a special case ($k = 1$) of the identity $\Gamma(m-k) \equiv 2^{m-k-1} \pi^{-1/2} \Gamma(\frac{m-k}{2}) \Gamma(\frac{m-k+1}{2})$ (Abramowitz and StegunAbramowitz and Stegun (1965), p. 483) . Assume $f(T, m) = 1$ holds for $T = t$. It must then be shown that $f(t+1, m) = 1, \forall m > T - 1$, where

$$\begin{aligned}
f(t+1, m) &= \prod_{k=1}^t [2^{(m-k-1)(t+1-k)} \times \frac{\Gamma(\frac{m}{2})}{\pi^{(t+1)/4}} \frac{\Gamma^{2t-2k+1}(\frac{m-k}{2})}{\Gamma^{t+1-k}(m-k)}] \\
&= [2^{m-t-1} \frac{\Gamma(\frac{m}{2})}{\pi^{(t+1)/4}} \frac{\Gamma(\frac{m-t}{2})}{\Gamma(m-t)}] \times \prod_{k=1}^{t-1} [2^{(m-k-1)(t+1-k)} \times \frac{\Gamma(\frac{m}{2})}{\pi^{(t+1)/4}} \frac{\Gamma^{2t-2k+1}(\frac{m-k}{2})}{\Gamma^{t+1-k}(m-k)}] \\
&= [2^{m-t-1} \frac{\Gamma(\frac{m}{2})}{\pi^{(t+1)/4}} \frac{\Gamma(\frac{m-t}{2})}{\Gamma(m-t)}] \times f(t, m) \times \prod_{k=1}^{t-1} [2^{m-k-1} \frac{1}{\pi^{1/4}} \frac{\Gamma^2(\frac{m-k}{2})}{\Gamma(m-k)}] \quad (11)
\end{aligned}$$

Using the same identity $\Gamma(m-k) \equiv 2^{m-k-1} \pi^{-1/2} \Gamma(\frac{m-k}{2}) \Gamma(\frac{m-k+1}{2})$ multiple times, $f(t+1, m)$

is

$$\begin{aligned}
f(t+1, m) &= \left[\frac{1}{\pi^{(t-1)/4}} \frac{\Gamma(\frac{m}{2})}{\Gamma(\frac{m-t+1}{2})} \right] \times 1 \times \prod_{k=1}^{t-1} \left[\frac{1}{\pi^{-1/4}} \frac{\Gamma(\frac{m-k}{2})}{\Gamma(\frac{m-k+1}{2})} \right] \\
&= \prod_{k=1}^t \left[\frac{\Gamma(\frac{m-k+1}{2})}{\Gamma(\frac{m-k+1}{2})} \right] \\
&= 1
\end{aligned} \tag{12}$$

(c). According to EatonEaton (1983) (p. 256), if $\Sigma \sim W_T(m, \Psi)$, then any $T_1 \times T_1$ principal sub-matrix Σ_1 of Σ has following distribution

$$\Sigma_1 \sim W_{T_1}(m, \Psi_1) \tag{13}$$

with Ψ_1 being the $T_1 \times T_1$ principal sub-matrix of Ψ . By applying it to the (2), any $T_1 \times T_1$ principal sub-matrix \mathbf{R}_1 of \mathbf{R} has density function $b_{T_1}(m) |\mathbf{R}_1|^{\frac{m-T_1-1}{2}}$. Simply let $T_1=2$, we could obtain the marginal distribution of ρ_{ij}

$$f(\rho_{ij}) = \frac{\Gamma^2(\frac{m}{2})}{\Gamma_2(\frac{m}{2})} (1 - \rho_{ij}^2)^{\frac{m-3}{2}}, \quad -1 \leq \rho_{ij} \leq 1 \tag{14}$$

□

B Stan code for AB-NMA models

B.1 AB-IW model

```

1 data{
2   int<lower=0> len; //total number of data size
3   int<lower=0> ntrt; //total number of treatments
4   int<lower=0> nstudy; //total number of studies
5   int<lower=0> t[len]; //indices of treatment

```

```

6  int<lower=0> s[len]; //indices of study
7  int<lower=0> r[len]; //number of events
8  int<lower=0> totaln[len]; //number of participants
9  int<lower=0,upper=1> higher_better; //to indicate whether higher
    ↪ absolute risk is better or not
10 vector<lower=0>[ntrt] zeros; //mean effects in MVN
11 matrix[ntrt,ntrt] Lambda; //parameter lambda in Inverse-Wishart,
    ↪ generally indentity matrix
12 real<lower=1> nu; //parameter nu in Inverse-Wishart, generally ntrt+1
13 }
14 parameters {
15   matrix[nstudy,ntrt] vi; //random effects
16   vector[ntrt] mu; //fixed effect for treatment
17   cov_matrix[ntrt] Sigma; //covariance matrix
18 }
19 transformed parameters {
20   vector<lower=0,upper=1>[len] p; //Study_specified Absolute Risk
21   for(i in 1:len){
22     p[i]=inv_logit(mu[t[i]]+vi[s[i],t[i]]);
23   }
24 }
25 model{
26   for(i in 1:len){
27     r[i]~binomial(totaln[i],p[i]);
28   }
29   for(j in 1:nstudy){
30     vi[j,]~multi_normal(zeros,Sigma);
31   }
32   for(j in 1:ntrt){
33     mu[j]~normal(0,100);
34   }

```

```

35   Sigma~inv_wishart(nu,Lambda);
36 }
37 generated quantities {
38   real AR[ntrt]; //Populaiton averaged absolute risk
39   real MU[ntrt];
40   real SD[ntrt]; //standard deviation of covariance matrix
41   real OR[ntrt,ntrt]; //Odds Ratio
42   real LOR[ntrt,ntrt]; //Log odds Ratio
43   real cLOR[ntrt,ntrt]; //cLog odds Ratio
44   real CORR[ntrt,ntrt]; //correlation matrix
45   int rk[ntrt]; // rank of AR
46   int rank_prob[ntrt,ntrt]; //rank probability
47   real DEV=0; // Total Residual Deviance
48   for(j in 1:ntrt){
49     AR[j]=inv_logit(mu[j]/sqrt(1+Sigma[j,j]*256/75/pi()/pi()));
50     SD[j]=sqrt(Sigma[j,j]);
51     MU[j]=mu[j];
52   }
53   for(j in 1:ntrt){
54     for(k in 1:ntrt){
55       OR[j,k]=AR[j]/(1-AR[j])/AR[k]*(1-AR[k]);
56       LOR[j,k]=log(OR[j,k]);
57       cLOR[j,k]=MU[j]-MU[k];
58       CORR[j,k]=Sigma[j,k]/SD[j]/SD[k];
59     }
60   }
61   for(j in 1:ntrt){
62     rk[j]=higher_better?ntrt-rank(AR,j):rank(AR,j)+1;
63   }
64   for(j in 1:ntrt){
65     for(k in 1:ntrt){

```

```

66     rank_prob[k,j]=(rk[k]==j);
67   }
68 }
69 for(i in 1:len){
70   DEV=DEV+2*(r[i]*(-log(p[i]*totaln[i]))+(totaln[i]-r[i])*(-log(totaln[i]
↪ ]-p[i]*totaln[i]))));
71 }
72 }

```

B.2 AB-RW model

```

1 data{
2   int<lower=0> len; //total number of data size
3   int<lower=0> ntrt; //total number of treatments
4   int<lower=0> nstudy; //total number of studies
5   int<lower=0> t[len]; //indices of treatment
6   int<lower=0> s[len]; //indices of study
7   int<lower=0> r[len]; //number of events
8   int<lower=0> totaln[len]; //number of participants
9   real<lower=0> uni_upper; //uniform distribution upper bound
10  int<lower=0,upper=1> higher_better; //to indicate whether higher
↪ absolute risk is better or not
11  vector<lower=0>[ntrt] zeros; //mean effects in MVN
12  real<lower=0.1> eta; //parameter eta in LKJ, generally 1
13 }
14 parameters {
15   matrix[nstudy,ntrt] vi; //random effects
16   vector[ntrt] mu; //fixed effect for treatment
17   cholesky_factor_corr[ntrt] L0omega; //LKJ prior
18   vector[ntrt] ssd; //seperate standard deviation
19 }

```



```

20 transformed parameters {
21   vector<lower=0,upper=1>[len] p; //Study_specified Absolute Risk
22   corr_matrix[ntrt] CORR; //correlation matrix
23   for(i in 1:len){
24     p[i]=inv_logit(mu[t[i]]+vi[s[i],t[i]]);
25   }
26   CORR <- multiply_lower_tri_self_transpose(LOmega);
27 }
28 model{
29   for(i in 1:len){
30     r[i]~binomial(totaln[i],p[i]);
31   }
32   for(j in 1:nstudy){
33     vi[j,]~multi_normal(zeros, quad_form_diag(CORR,ssd));
34   }
35   for(j in 1:ntrt){
36     mu[j]~normal(0,100);
37     ssd[j]~uniform(0,uni_upper);
38   }
39   LOmega~lkj_corr_cholesky(eta);
40 }
41 generated quantities {
42   real AR[ntrt]; //Populaiton averaged absolute risk
43   real MU[ntrt];
44   real SD[ntrt]; //standard deviation
45   real OR[ntrt,ntrt]; //Odds Ratio
46   real LOR[ntrt,ntrt]; //Log odds Ratio
47   real cLOR[ntrt,ntrt]; //cLog odds Ratio
48   int rk[ntrt]; // rank of AR
49   int rank_prob[ntrt,ntrt]; //rank probability
50   real DEV=0; // Total Residual Deviance

```

```

51  for(j in 1:ntrt){
52    AR[j]=inv_logit(mu[j]/sqrt(1+ssd[j]*ssd[j]*256/75/pi()/pi()));
53    SD[j]=ssd[j];
54    MU[j]=mu[j];
55  }
56  for(j in 1:ntrt){
57    for(k in 1:ntrt){
58      OR[j,k]=AR[j]/(1-AR[j])/AR[k]*(1-AR[k]);
59      LOR[j,k]=log(OR[j,k]);
60      cLOR[j,k]=MU[j]-MU[k];
61    }
62  }
63  for(j in 1:ntrt){
64    rk[j]=higher_better?ntrt-rank(AR,j):rank(AR,j)+1;
65  }
66  for(j in 1:ntrt){
67    for(k in 1:ntrt){
68      rank_prob[k,j]=(rk[k]==j);
69    }
70  }
71  for(i in 1:len){
72    DEV=DEV+2*(r[i]*(-log(p[i]*totaln[i]))+(totaln[i]-r[i])*(-log(totaln[i]
↪ ]-p[i]*totaln[i]))));
73  }
74 }

```

B.3 AB-RIW model

```

1 data{
2   int<lower=0> len; //total number of data size
3   int<lower=0> ntrt; //total number of treatments

```

```

4  int<lower=0> nstudy; //total number of studies
5  int<lower=0> t[len]; //indices of treatment
6  int<lower=0> s[len]; //indices of study
7  int<lower=0> r[len]; //number of events
8  int<lower=0> totaln[len]; //number of participants
9  real<lower=0> uni_upper; //uniform distribution upper bound
10 int<lower=0,upper=1> higher_better; //to indicate whether higher
    ↪ absolute risk is better or not
11 vector<lower=0>[ntrt] zeros; //mean effects in MVN
12 matrix[ntrt,ntrt] Lambda; //parameter lambda in Inverse-Wishart,
    ↪ generally indentity matrix
13 real<lower=1> nu; //parameter nu in Inverse-Wishart, generally ntrt+1
14 }
15 parameters {
16  matrix[nstudy,ntrt] vi; //random effects
17  vector[ntrt] mu; //fixed effect for treatment
18  cov_matrix[ntrt] Sigma; //covariance matrix
19  vector[ntrt] ssd; //seperate standard deviation
20 }
21 transformed parameters {
22  vector<lower=0,upper=1>[len] p; //Study_specified Absolute Risk
23  corr_matrix[ntrt] CORR; //correlation matrix
24  for(i in 1:len){
25    p[i]=inv_logit(mu[t[i]]+vi[s[i],t[i]]);
26  }
27  for(j in 1:ntrt){
28    for(k in 1:ntrt){
29      CORR[j,k]=Sigma[j,k]/sqrt(Sigma[j,j])/sqrt(Sigma[k,k]);
30    }
31  }
32 }

```

```

33 model{
34   for(i in 1:len){
35     r[i]~binomial(totaln[i],p[i]);
36   }
37   for(j in 1:nstudy){
38     vi[j,]~multi_normal(zeros, quad_form_diag(CORR,ssd));
39   }
40   for(j in 1:ntrt){
41     mu[j]~normal(0,100);
42     ssd[j]~uniform(0,uni_upper);
43   }
44   Sigma~inv_wishart(nu,Lambda);
45 }
46 generated quantities {
47   real AR[ntrt]; //Populaiton averaged absolute risk
48   real MU[ntrt];
49   real SD[ntrt]; //standard deviation
50   real OR[ntrt,ntrt]; //Odds Ratio
51   real LOR[ntrt,ntrt]; //Log odds Ratio
52   real cLOR[ntrt,ntrt]; //cLog odds Ratio
53   int rk[ntrt]; // rank of AR
54   int rank_prob[ntrt,ntrt]; //rank probability
55   real DEV=0; // Total Residual Deviance
56   for(j in 1:ntrt){
57     AR[j]=inv_logit(mu[j]/sqrt(1+ssd[j]*ssd[j]*256/75/pi()/pi()));
58     SD[j]=ssd[j];
59     MU[j]=mu[j];
60   }
61   for(j in 1:ntrt){
62     for(k in 1:ntrt){
63       OR[j,k]=AR[j]/(1-AR[j])/AR[k]*(1-AR[k]);

```

```

64     cLOR[j,k]=MU[j]-MU[k];
65     LOR[j,k]=log(OR[j,k]);
66 }
67 }
68 for(j in 1:ntrt){
69     rk[j]=higher_better?ntrt-rank(AR,j):rank(AR,j)+1;
70 }
71 for(j in 1:ntrt){
72     for(k in 1:ntrt){
73         rank_prob[k,j]=(rk[k]==j);
74     }
75 }
76 for(i in 1:len){
77     DEV=DEV+2*(r[i]*(-log(p[i]*totaln[i]))+(totaln[i]-r[i])*(-log(totaln[i]
↪ ]-p[i]*totaln[i]))));
78 }
79 }

```

B.4 AB-EQ model

```

1 data{
2     int<lower=0> len; //total number of data size
3     int<lower=0> ntrt; //total number of treatments
4     int<lower=0> nstudy; //total number of studies
5     int<lower=0> t[len]; //indices of treatment
6     int<lower=0> s[len]; //indices of study
7     int<lower=0> r[len]; //number of events
8     int<lower=0> totaln[len]; //number of participants
9     real<lower=0> uni_upper; //uniform distribution upper bound
10    int<lower=0,upper=1> higher_better; //to indicate whether higher
↪ absolute risk is better or not

```

```

11  vector<lower=0>[ntrt] zeros; //mean effects in MVN
12 }
13 parameters {
14  matrix[nstudy,ntrt] vi; //random effects
15  vector[ntrt] mu; //fixed effect for treatment
16  vector[ntrt] ssd; //seperate standard deviation
17  real CORR; //correlation
18 }
19 transformed parameters {
20  vector<lower=0,upper=1>[len] p; //Study_specified Absolute Risk
21  matrix[ntrt,ntrt] Omega; //precision matrix
22  real diag;
23  real offdiag;
24  diag=(1+(ntrt-2)*CORR)/(1+(ntrt-2)*CORR-(ntrt-1)*CORR^2);
25  offdiag=(-CORR/(1+(ntrt-2)*CORR-(ntrt-1)*CORR^2));
26  for(i in 1:len){
27    p[i]=inv_logit(mu[t[i]]+vi[s[i],t[i]]);
28  }
29  for(j in 1:ntrt){
30    for(k in 1:ntrt){
31      Omega[j,k]=(j==k?diag:offdiag)/ssd[j]/ssd[k];
32    }
33  }
34 }
35 model{
36  for(i in 1:len){
37    r[i]~binomial(totaln[i],p[i]);
38  }
39  for(j in 1:nstudy){
40    vi[j,]~multi_normal_prec(zeros, Omega);
41  }

```

```

42  for(j in 1:ntrt){
43    mu[j]~normal(0,100);
44    ssd[j]~uniform(0,uni_upper);
45  }
46  CORR~uniform((-1.000)/(ntrt-1)+0.0001,0.9999);
47 }
48 generated quantities {
49   real AR[ntrt]; //Populaiton averaged absolute risk
50   real MU[ntrt];
51   real SD[ntrt]; //standard deviation
52   real OR[ntrt,ntrt]; //Odds Ratio
53   real LOR[ntrt,ntrt]; //Log odds Ratio
54   real cLOR[ntrt,ntrt]; //cLog odds Ratio
55   int rk[ntrt]; // rank of AR
56   int rank_prob[ntrt,ntrt]; //rank probability
57   real DEV=0; // Total Residual Deviance
58   for(j in 1:ntrt){
59     AR[j]=inv_logit(mu[j]/sqrt(1+ssd[j]*ssd[j]*256/75/pi()/pi()));
60     SD[j]=ssd[j];
61     MU[j]=mu[j];
62   }
63   for(j in 1:ntrt){
64     for(k in 1:ntrt){
65       OR[j,k]=AR[j]/(1-AR[j])/AR[k]*(1-AR[k]);
66       cLOR[j,k]=MU[j]-MU[k];
67       LOR[j,k]=log(OR[j,k]);
68     }
69   }
70   for(j in 1:ntrt){
71     rk[j]=higher_better?ntrt-rank(AR,j):rank(AR,j)+1;
72   }

```

```

73  for(j in 1:ntrt){
74    for(k in 1:ntrt){
75      rank_prob[k,j]=(rk[k]==j);
76    }
77  }
78  for(i in 1:len){
79    DEV=DEV+2*(r[i]*(-log(p[i]*totaln[i]))+(totaln[i]-r[i])*(-log(totaln[i]
↪ ]-p[i]*totaln[i])));
80  }
81 }

```

B.5 AB-EQ-EV model

```

1  data{
2    int<lower=0> len; //total number of data size
3    int<lower=0> ntrt; //total number of treatments
4    int<lower=0> nstudy; //total number of studies
5    int<lower=0> t[len]; //indices of treatment
6    int<lower=0> s[len]; //indices of study
7    int<lower=0> r[len]; //number of events
8    int<lower=0> totaln[len]; //number of participants
9    real<lower=0> uni_upper; //uniform distribution upper bound
10   int<lower=0,upper=1> higher_better; //to indicate whether higher
↪ absolute risk is better or not
11   vector<lower=0>[ntrt] zeros; //mean effects in MVN
12 }
13 parameters {
14   matrix[nstudy,ntrt] vi; //random effects
15   vector[ntrt] mu; //fixed effect for treatment
16   real ssdd; //seperate standard deviation
17   real CORR; //correlation

```



```

18 }
19 transformed parameters {
20   vector<lower=0,upper=1>[len] p; //Study_specified Absolute Risk
21   matrix[ntrt,ntrt] Omega; //precision matrix
22   real diag;
23   real offdiag;
24   diag=(1+(ntrt-2)*CORR)/(1+(ntrt-2)*CORR-(ntrt-1)*CORR^2);
25   offdiag=(-CORR/(1+(ntrt-2)*CORR-(ntrt-1)*CORR^2));
26   for(i in 1:len){
27     p[i]=inv_logit(mu[t[i]]+vi[s[i],t[i]]);
28   }
29   for(j in 1:ntrt){
30     for(k in 1:ntrt){
31       Omega[j,k]=(j==k?diag:offdiag)/ssdd/ssdd;
32     }
33   }
34 }
35 model{
36   for(i in 1:len){
37     r[i]~binomial(totaln[i],p[i]);
38   }
39   for(j in 1:nstudy){
40     vi[j,]~multi_normal_prec(zeros, Omega);
41   }
42   for(j in 1:ntrt){
43     mu[j]~normal(0,100);
44   }
45   ssdd~uniform(0,uni_upper);
46   CORR~uniform((-1.000)/(ntrt-1)+0.0001,0.9999);
47 }
48 generated quantities {

```

```

49  real AR[ntrt]; //Populaiton averaged absolute risk
50  real MU[ntrt];
51  real SD; //standard deviation
52  real OR[ntrt,ntrt]; //Odds Ratio
53  real LOR[ntrt,ntrt]; //Log odds Ratio
54  real cLOR[ntrt,ntrt]; //cLog odds Ratio
55  int rk[ntrt]; // rank of AR
56  int rank_prob[ntrt,ntrt]; //rank probability
57  real DEV=0; // Total Residual Deviance
58  for(j in 1:ntrt){
59      AR[j]=inv_logit(mu[j]/sqrt(1+ssdd*ssdd*256/75/pi()/pi()));
60      MU[j]=mu[j];
61  }
62  SD=ssdd;
63  for(j in 1:ntrt){
64      for(k in 1:ntrt){
65          OR[j,k]=AR[j]/(1-AR[j])/AR[k]*(1-AR[k]);
66          LOR[j,k]=log(OR[j,k]);
67          cLOR[j,k]=MU[j]-MU[k];
68      }
69  }
70  for(j in 1:ntrt){
71      rk[j]=higher_better?ntrt-rank(AR,j):rank(AR,j)+1;
72  }
73  for(j in 1:ntrt){
74      for(k in 1:ntrt){
75          rank_prob[k,j]=(rk[k]==j);
76      }
77  }
78  for(i in 1:len){
79      DEV=DEV+2*(r[i]*(-log(p[i]*totaln[i])))+(totaln[i]-r[i])*(-log(totaln[i

```

```

      ↪ ]-p[i]*totaln[i])));
80 }
81 }

```

C R Simulation codes for Table 1

```

1 logit <- function(x){
2   return(log(x/(1-x)))
3 }
4 #####
5 library(rstan)
6 library(bayesplot)
7 library(coda)
8 library(mvtnorm)
9 library(randomizr)
10 #####
11 job = 1
12 #####
13 d = 3
14 nstudy = 18
15 pop_size <- 500
16 set.seed(seed)
17 sd = c(0.7,0.4,0.1)
18 cor <- matrix(c(1,2/3,4/9,2/3,1,2/3,4/9,2/3,1),nrow=3,ncol=3)
19 sigma <- matrix(NA,nrow=3,ncol=3)
20 for(i in 1:d){
21   for(j in 1:d){
22     sigma[i,j] <- cor[i,j]*sd[i]*sd[j]
23   }
24 }
25 v <- rmvnorm(nstudy, rep(0, length = 3), sigma)

```

```

26 rep.row<-function(x,n){
27   matrix(rep(x,each=n),nrow=n)
28 }
29 mu <- rep.row(c(-1,-1.5,-2),nstudy)
30 pi1 <- exp(mu+v)/(1+exp(mu+v))
31 event <- matrix(NA,nrow=nstudy,ncol=3)
32 for(i in 1:d){
33   event[,i] <- rbinom(nstudy,pop_size,pi1[,i])
34 }
35
36 missing_pattern_determine <- function(ind=1, event){
37   d = 3
38   nstudy = 18
39
40   # 1. No missing
41   if(ind==1){
42     missing_ind <- rep(0,nstudy)
43   }
44
45   # 2. MAR
46   if(ind==2){
47     missing_ind <- rep(0,nstudy)
48     for(i in 1:nstudy){
49       if(event[i,2]>=median(event[,2])){missing_ind[i] <- 3}
50       else{missing_ind[i] <- 1}
51     }
52   }
53
54   # 3. MNAR
55   beta_mar_0 <- 4.5
56   beta_mar_1 <- 1

```

```

57 if(ind==3){
58   missing_ind <- rep(0,nstudy)
59   eventm <- event
60   for(i in 1:nstudy){
61     if(event[i,2]==0){eventm[i,2]=0.5}
62     if(event[i,2]==500){eventm[i,2]=499.5}
63     if(event[i,1]==0){eventm[i,1]=0.5}
64     if(event[i,1]==500){eventm[i,1]=499.5}
65     if(event[i,3]==0){eventm[i,3]=0.5}
66     if(event[i,3]==500){eventm[i,3]=499.5}
67     tempmisind <- exp(beta_mar_0+beta_mar_1*logit(eventm[i,2]/500)+beta_
↪ mar_1*logit(eventm[i,1]/500)+beta_mar_1*logit(eventm[i,3]/500))/(1+
↪ exp(beta_mar_0+beta_mar_1*logit(eventm[i,2]/500)+beta_mar_1*logit(
↪ eventm[i,1]/500)+beta_mar_1*logit(eventm[i,3]/500)))
68     if(rbinom(1,1,tempmisind)==1){missing_ind[i] <- 1}
69     else{missing_ind[i] <- 3}
70   }
71 }
72
73 ### Construct the non-missing pattern
74 non_missing_pattern <- matrix(1,nrow=nstudy,ncol=d)
75 for(i in 1:nstudy){
76   if(missing_ind[i]!=0){
77     non_missing_pattern[i,missing_ind[i]] <- 0
78   }
79 }
80 return(non_missing_pattern)
81 }
82
83 ##### statistical analyze part
84 for(missing_ii in 1:3){

```

```

85 #missing_ii <- 3
86 set.seed(seed*missing_ii)
87 # missing pattern
88 non_missing_pattern <- missing_pattern_determine(ind=missing_ii,event)
89 # set up network
90 len <- sum(sum(non_missing_pattern))
91 s.id <- rep(0,len)
92 t.id <- rep(0,len)
93 event.n <- rep(0,len)
94 total.n <- rep(pop_size,len)
95 ind <- 1
96 for(i in 1:nstudy){
97   for(j in 1:d){
98     if(non_missing_pattern[i,j]!=0){
99       s.id[ind] <- i
100      t.id[ind] <- j
101      event.n[ind] <- event[i,j]
102      ind <- ind+1
103    }
104  }
105 }
106 ntrt <- d
107 r <- event.n
108 n <- total.n
109
110
111 ##### AB-IW
112 data.stan <- list(len=len,ntrt=ntrt,nstudy=nstudy,t=t.id,s=s.id,r=event.
  ↪ n,totaln=total.n, zeros=rep(0,ntrt),Lambda=diag(ntrt),nu=ntrt+1,
  ↪ higher_better=1,uni_upper=5)
113 init.stan <- function() { list(mu=rep(0, ntrt),vi=matrix(0,nstudy,ntrt),

```

```

    ↪ Sigma=diag(ntrt))}
114 fit <- sampling(IW,pars=c("MU","LOR","cLOR","AR","SD","CORR","rank_prob"
    ↪ ,"DEV","p"),data=data.stan,init=init.stan,chains=4,iter=2000,warmup
    ↪ =1000,verbose=TRUE,seed=(1234+job),thin=1,control = list(max_
    ↪ treedepth = 15))
115 sum_fit <- summary(fit)$summary
116 saveRDS(sum_fit,paste("IW_",missing_ii,"_",job,sep=""))
117
118 ##### AB-RW
119 data.stan <- list(len=len,ntrt=ntrt,nstudy=nstudy,t=t.id,s=s.id,r=event.
    ↪ n,totaln=total.n, zeros=rep(0,ntrt),eta=1,higher_better=1,uni_upper
    ↪ =5)
120 init.stan<- function() { list(mu=rep(0, ntrt),vi=matrix(0,nstudy,ntrt),
    ↪ LOmega=diag(ntrt),ssd=runif(ntrt,0,1)) }
121 fit <- sampling(RW,pars=c("MU","LOR","cLOR","AR","SD","CORR","rank_prob"
    ↪ ,"DEV","p"),data=data.stan,init=init.stan, chains=4,iter=2000,warmup
    ↪ =1000,verbose=TRUE,seed=(1234+job),thin=1,control = list(adapt_delta
    ↪ = 0.99,max_treedepth = 15))
122 sum_fit <- summary(fit)$summary
123 saveRDS(sum_fit,paste("RW_",missing_ii,"_",job,sep=""))
124
125
126 ##### AB-RIW
127 data.stan <- list(len=len,ntrt=ntrt,nstudy=nstudy,t=t.id,s=s.id,r=event.
    ↪ n,totaln=total.n, zeros=rep(0,ntrt),Lambda=diag(ntrt),nu=ntrt+1,
    ↪ higher_better=1,uni_upper=5)
128 init.stan<- function() { list(mu=rep(0, ntrt),vi=matrix(0,nstudy,ntrt),
    ↪ Sigma=diag(ntrt),ssd=runif(ntrt,0,1)) }
129 fit <- sampling(RIW,pars=c("MU","LOR","cLOR","AR","SD","CORR","rank_prob"
    ↪ ,"DEV","p"),data=data.stan,init=init.stan, chains=4,iter=2000,
    ↪ warmup=1000,verbose=TRUE,seed=(1234+job),thin=1,control = list(adapt

```

```

    ↪ _delta = 0.99,max_treedepth = 15))
130 sum_fit <- summary(fit)$summary
131 saveRDS(sum_fit,paste("RIW_",missing_ii,"_",job,sep=""))
132
133
134 ##### AB-EQ
135 data.stan <- list(len=len,ntrt=ntrt,nstudy=nstudy,t=t.id,s=s.id,r=event.
    ↪ n,totaln=total.n, zeros=rep(0,ntrt),higher_better=1,uni_upper=5)
136 init.stan<- function() { list(mu=rep(0, ntrt),vi=matrix(0,nstudy,ntrt),
    ↪ CORR=0,ssd=runif(ntrt,0,1)) }
137 fit <- sampling(EQ,pars=c("MU","LOR","cLOR","AR","SD","CORR","rank_prob"
    ↪ ,"DEV","p"),data=data.stan,init=init.stan, chains=4,iter=2000,warmup
    ↪ =1000,verbose=TRUE,seed=(1234+job),thin=1,control = list(adapt_delta
    ↪ = 0.99,max_treedepth = 15))
138 sum_fit <- summary(fit)$summary
139 saveRDS(sum_fit,paste("EQ_",missing_ii,"_",job,sep=""))
140 }

```

References

- Abramowitz, M. and I. A. Stegun (Eds.) (1965). *Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables (Dover Books on Mathematics)*. Dover Publications.
- Eaton, M. L. (1983). *Multivariate Statistics: A Vector Space Approach (Wiley Series in Probability and Statistics)*. John Wiley & Sons Inc.
- James, A. T. (1964). Distributions of matrix variates and latent roots derived from normal samples. *The Annals of Mathematical Statistics* 35(2), 475–501.
- Lewandowski, D., D. Kurowicka, and H. Joe (2009). Generating random correlation matrices

based on vines and extended onion method. *Journal of Multivariate Analysis* 100(9), 1989–2001.