SUPPLEMENTARY MATERIALS

The supplementary methods provide a detailed description of the analytic strategy under DataSHIELD for the study-level meta-analysis (SLMA) of peripheral systolic blood pressure (SBP) data (scenario 1, Figure 4), and for the individual-level meta-analysis (ILMA) of acute myocardial infarction data (scenario 2, Figure 5).

Contents:

S1. SCENARIO 1. Using DataSHIELD to enhance the flexibility and efficiency of study level meta-analysis (SLMA)

Simulation

Six hypothetical studies were simulated. They were set up to investigate the influence of age (AGE) and a single nucleotide polymorphism (SNP) on peripheral systolic blood pressure (SBP). The six studies consisted of 1,000, 2,000, 3,000, 4,000, 2,500 and 2,500 participants, respectively. Recruits were aged between 50 and 70 years. For the j^{th} individual (j = 1,…,4,000) in the i^{th} study (i = 1,…,6), AGE_{ij} was generated from a uniform distribution with bounding parameters 50 and 70, and centralized by subtracting the mean (60 years). SNP_{ij} was generated as the sum of two calls from a Bernoulli distribution with p = 0.2, corresponding to a minor-allele frequency of 0.2. The three genotypes were coded 0 (= no copies of the minor-allele), 1 (= one copy of the minor-allele) or 2 (= two copies of the minor-allele). Given the coding of the SNP variable, the data simulated reflect an additive genetic model.

Having generated the simulated values for AGE_{ij} and SNP_{ij}, the linear predictor for each individual, LP_{ij}, was generated as:

where , , and . SBP_{ij} was then generated from a normal distribution with *mean* = LP_{ij}, and *sd* = 11.

R code for simulating the data (cut and paste to use directly in R):

```
###############
#  SIMULATION  #
###############

#set up data structure
set.seed(18984)
numsubs.study<-c(1000,2000,3000,4000,2500,2500)
numsubs<-sum(numsubs.study)
numstudies<-length(numsubs.study)
study.id<-rep(1:numstudies,numsubs.study)

#set up model structure and parameters
numpara<-3
beta0<-125
betaAGE<-0.25
betaSNP<-0.5
MAF<-0.2

#simulate data
AGE<-runif(numsubs,50,70) - 60
SNP.1<-rbinom(numsubs,1,MAF)
SNP.2<-rbinom(numsubs,1,MAF)
SNP<-SNP.1+SNP.2
lp<-beta0+betaAGE*AGE+betaSNP*SNP
SBP<-rnorm(numsubs,lp,11)
```

Analysis

Analysis was based on the multiple linear regression model:

R code for overall regression analysis (all individual level data combined into one data set)

```
######################
#   OVERALL ANALYSIS    #
######################

#All studies together
model.overall<-lm(SBP~AGE+SNP)
cat("\n\nOVERALL ANALYSIS","\n")
summary(model.overall)
```

Abbreviated output from overall regression analysis

```
OVERALL ANALYSIS
Call:
lm(formula = SBP ~ AGE + SNP)

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept) 125.15770    0.10943 1143.724   <2e-16
AGE           0.25937    0.01549   16.745   <2e-16
SNP           0.44796    0.15806    2.834   0.0046
---
```

R code for study-specific analyses

```
##########################
# STUDY SPECIFIC ANALYSES #
##########################

#create empty results matrices
beta.s<-matrix(NA,nrow=numpara,ncol=numstudies)
se.s<-matrix(NA,nrow=numpara,ncol=numstudies)

#work with each study one at a time
for(k in 1:numstudies)
{
SBP.s<-SBP[study.id==k]
AGE.s<-AGE[study.id==k]
SNP.s<-SNP[study.id==k]

model.study.specific<-lm(SBP.s~AGE.s+SNP.s)
cat("\n\nSTUDY",k,"\n")
print(summary(model.study.specific))
beta.s[,k]<-summary(model.study.specific)$coefficients[,1]
se.s[,k]<-summary(model.study.specific)$coefficients[,2]
}
```

Abbreviated output from study-specific analyses

STUDY 1
```
Call:
lm(formula = SBP.s ~ AGE.s + SNP.s)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 124.95070    0.43910 284.560  < 2e-16
AGE.s         0.31155    0.06315   4.933 9.47e-07
SNP.s         1.66853    0.67835   2.460   0.0141
---
```

STUDY 2
```
Call:
lm(formula = SBP.s ~ AGE.s + SNP.s)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 125.29081    0.29894 419.119  < 2e-16
AGE.s         0.22046    0.04265   5.169 2.59e-07
SNP.s        -0.28092    0.42458  -0.662    0.508
---
```

STUDY 3
```
Call:
lm(formula = SBP.s ~ AGE.s + SNP.s)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 125.1460     0.2467 507.308   <2e-16
AGE.s         0.2990     0.0349   8.566   <2e-16
SNP.s         0.8101     0.3483   2.326   0.0201
---
```

STUDY 4
```
Call:
lm(formula = SBP.s ~ AGE.s + SNP.s)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 125.10805    0.21473 582.633   <2e-16
AGE.s         0.27995    0.03003   9.321   <2e-16
SNP.s         0.44297    0.30450   1.455    0.146
---
```

STUDY 5
```
Call:
lm(formula = SBP.s ~ AGE.s + SNP.s)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 125.23344    0.26195 478.086  < 2e-16
AGE.s         0.24625    0.03726   6.609  4.7e-11
SNP.s         0.37170    0.38534   0.965    0.335
---
```

```
STUDY 6***
Call:
lm(formula = SBP.s ~ AGE.s + SNP.s)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 125.12993    0.26289 475.974  < 2e-16
AGE.s         0.20305    0.03727   5.448 5.59e-08
SNP.s         0.25418    0.39068   0.651    0.515
---
```

Meta-analysis

Meta-analysis involved calculating the weighted mean of each regression coefficient. Weighting was based on the number of subjects in each study.

Standard errors were converted into precisions (inverse variances) by squaring and inverting the standard errors for each coefficient in each study. The precisions for each coefficient were then summed across the six studies to obtain the overall precision for each coefficient, and the overall precisions were converted back to standard errors by inverting and taking the square root.

```
#################
#  META-ANALYSIS  #
#################
#set up analysis weights
analysis.wt<-numsubs.study/numsubs

#set up a vector of 1s to use in summing precisions
simple.sum<-rep(1,numstudies)

#calculate mean of regression coefficients weighted for study sample sizes
# "%*%" denotes vector multiplication
beta.overall<-beta.s%*%analysis.wt

#convert standard errors into precisions
precision.s <-1/(se.s)^2

#sum precisions across studies
precision.overall<-precision.s%*%simple.sum

#convert precisions back to standard errors
se.overall <-1/(precision.overall)^0.5

#round outputs
beta.overall<-round(beta.overall,digits=4)
se.overall<-round(se.overall,digits=4)
```

#create output results matrix
meta.analysis.results<-cbind(beta.overall,se.overall)
dimnames(meta.analysis.results)<-list(c("Intercept","AGE","SNP"),c("Coefficients","SE"))

#print output
print(analysis.wt)
print(meta.analysis.results)

Output from meta-analysis

```
print(analysis.wt)
[1] 0.06667 0.13333 0.20000 0.26667 0.16667 0.16667
```

```
print(meta.analysis.results)
> print(meta.analysis.results)
          Coefficients     SE
Intercept     125.1541 0.1094
AGE             0.2595 0.0155
SNP             0.4582 0.1580
```

Comment

Although it is not really germane to the validity of the DataSHIELD, the similarity of the estimates and standard errors from the overall regression analysis (above) and the results of this meta-analysis confirm that in this particular simulated scenario, the chosen approach to meta-analysis (*i.e.* overall parameter estimates obtained as the mean of study specific coefficients weighted by study size) was appropriate.

The matrix of summary statistics in Figure 4 in the main text, corresponds to the study-specific results of study 6 (see *** above).

S2. The IRLS Algorithm

The Iteratively Reweighted Least Squares (IRLS) algorithm is an iterative method of maximum likelihood estimation for *generalised linear models*(**1**) (GLMs). It is closely related to the Newton-Raphson procedure, but uses the *expected information matrix* to iteratively update the regression parameters at each iteration while the usual Newton-Raphson procedure uses the *observed information matrix.* The general form of the IRLS algorithm for the *r*th iteration is:

Equation 1

where is the vector of estimated regression coefficients at the start of the current iteration, is the estimated *expected information* matrix, is the *score* vector and is the updated vector of regression coefficients at the end of iteration r that provides the coefficient values to be used in the *next* iteration (r+1). The inverse of the expected information matrix is the *variance-covariance* matrix of parameter estimates.

In the practical example described in Figure 5 in the main text, the data are consistent with fitting a conventional (unconditional) logistic regression model (Bernoulli error, logistic link)(**2**). For a GLM of this particular type, each component of the IRLS algorithm is derived as follows:

In the first iteration, and for the *i*th subject, the linear predictor LP_i is derived by multiplying out the model equation:

where X is the design matrix, *i.e.* the matrix of covariates (see Figure 2 main text), and is the vector of initial ('guessed') regression coefficients.

Fitted probabilities are then obtained using the inverse logistic transformation, sometimes known as the expit transformation:

.

The expected information matrix is now estimated:

where W_r (here W_1) is a weight matrix (X is again the design matrix), and the superscript 'T' indicates matrix transposition. The weight matrix is a diagonal matrix with diagonal elements w_i, each equal to

where $g'(\mu_i)$ indicates the first differential of the link function - here the logistic function - and V_i is the variance function for the *i*th subject. For a logistic regression model: ; ; and . At the rth iteration W_r is derived from the particular parameter values that pertain at that iteration, and W_1 is therefore based on the parameter values that apply in the first iteration.

Finally, the score function is derived:

,

where *X* and *W* are as before, and ***u₁*** is a vector of subject-specific terms (u_i), where : if subject i is a case; if subject i is a control.

At the end of the first iteration, is derived from using and .

The next iteration, *r* = 2, is then performed taking as the vector of regression coefficients, generating and and using these via Equation 1 to update to obtain . This whole process is repeated successively until convergence is achieved.

Critically, in the context of DataSHIELD when data are partitioned into N different studies, and at iteration r can each be obtained by summing their study-specific components across all studies. Thus if is the component of the information matrix calculated as above, but using solely the data from study k, the overall information matrix at iteration r can simply be obtained as , and **,** the overall score function, as . This means that the update in the regression coefficients from to can be obtained knowing *only* the information matrix and score function from each study at each iteration – these are *sufficient statistics* - and there is therefore no requirement for the analysis centre to have access to *any* of the individual level data from the studies: the derivation of the study specific information matrices and score functions can be carried out entirely on the local computers at each study.

In determining whether convergence has been achieved, R, the *glm* function has a convergence criterion that is a function of the residual deviance for the current model, D_r, and the residual deviance for the previous model, D_{r-1}. For instance, the default convergence criterion for *glm* in R satisfies the following condition:

where ε = 1e-8

The residual deviance is calculated at each iteration:

,

where log L_F is the log-likelihood for the full ('saturated') model and log L_C is the log-likelihood for the current model. For a logistic regression model with binary (1,0) outcomes, as used in the current example, the log-likelihood for the current model is

where *C* is a constant, , and all other parameters are as before.

S3. SCENARIO 2. Using DataSHIELD to undertake a true individual level meta-analysis (ILMA) using a generalized linear model (GLM)

Simulation

Data were simulated for six hypothetical case-control studies set up to investigate the relationship between the risk of acute myocardial infarction, body mass index (BMI), and a single nucleotide polymorphism (SNP). For each individual, *j*, in each study, *i*, BMI was generated from a normal distribution with mean 23 kg/m^2 and standard deviation 4 kg/m^2. BMI was then centralised by subtracting the mean, 23 kg/m^2, from each measurement. A genotype for the SNP of interest was generated for each individual in a manner equivalent to the sum of two calls to a Bernoulli distribution with p = 0.3(**2**). The minor-allele frequency was thus 0.3, and each genotype was either 0 (= no copies of the minor-allele), 1 (= one copy of the minor-allele) or 2 (= two copies of the minor-allele). Given the coding of the SNP variable, the data simulated reflect an additive genetic model.

In addition to the regression coefficients for the intercept ($b_{\text{intercept}}$) and two simulated covariates, and , the model also incorporated an interaction term, , to allow for between-study heterogeneity in the magnitude of the effect of the BMI covariate on the log-odds of MI. The interaction covariate took the value zero for individuals in studies 1, 2, and 3, and the BMI value for individuals in studies 4, 5, and 6, while the interaction coefficient implied that a one unit change in BMI in a subject in studies 4, 5 or 6 increased the log-odds of MI by an amount higher than an equivalent change in a subject in studies 1, 2 or 3.

The following model was thus used to generate the linear predictor:

where , , , and .

Probabilities for developing acute myocardial infarction, p_{ij}, were derived by taking the inverse logistic (expit) transformation of the linear-predictor:

Case-control status, y_{ij}, was then generated for each individual by taking a random draw from a Bernoulli distribution with $p = p_{ij}$:

If the sampled value of y_{ij} was 1, the subject was designated to be a case, if y_{ij} was 0 the subject was designated a control.

The case-control composition of the six simulated studies are summarised in the table below:

Study	Cases	Controls	Total
1	962	1038	2000
2	1486	1514	3000
3	761	739	1500
4	142	158	300
5	1036	964	2000
6	360	340	700

R code for simulating the data:

In order to use the following R code, cut at the top and bottom of the block, paste into a new "script file" in R and then run the script file

```
#>>>>>>>>>>>>>>>>>>>>>>>>START OF FIRST BLOCK OF R CODE>>>>>>>>>>>>>>>>>>>>>>>>
#In order to provide a clear illustration of the use of a pooled GLM analysis
#based on the partitioned IRLS algorithm, this first block of code simulates data for six studies,
#writes the data out to six separate data files in different folders on the local computer that is
#being used to simulate them. These six folders are called:
#C:\DataSHIELD.Example\DC1
#C:\DataSHIELD.Example\DC2
#C:\DataSHIELD.Example\DC3
#C:\DataSHIELD.Example\DC4
#C:\DataSHIELD.Example\DC5
#C:\DataSHIELD.Example\DC6

#These folders correspond to what would be folders on the local data computers
#that would hold the real data files in a real example pooling data from multiple sites
#Readers wishing to run this R code in order to fully explore this example will need to
# create a top level directory (C:\DataSHIELD.Example) with the six study specific directories
# beneath it. Alternative folder and file names can of course be used but the relevant names and
# locations will then have to be changed in the R code.

#For convenience, a copy of the full data set (all six studies combined) is also sent to the root level
#folder C:\DataSHIELD.Example\

#An additional folder (C:\DataSHIELD.Example\AC) is also required in order to represent the folder
#where output from the individual data computers is sent to the analysis centre (AC) at the
#end of each iteration and where the data computers can obtain the current beta vector.

#We have deliberately written this code in a way that to experienced R users will look
#very inefficient. For example, rather than writing a loop that will apply the same code
#with appropriate modifications to each study sequentially, we have written the code for
#each study in full and have stacked this code one on top of another. We believe that this
#inefficiency makes the code far easier to follow for novice R users.

#In addition, we have deliberately chosen to save objects on hard disk and then recall them into
#R, rather than just leaving them in the active memory of R. This again makes things clearer. Users
#can stop after one iteration and see what model components have been constructed. It is
#also realistic, because when DataSHIELD is used in a real analysis
```

```
#the objects created from the R analysis at a remote data computer will
#not magically appear in the active memory of the local R program running at the analysis centre.

##############
#  SIMULATION  #
##############

#Start R code preparation
#First maximise memory allocation to ensure that you do not run out of space in R
memory.limit(4095)

#For convenience, start by setting up file names ahead of time
#Note that by convention slashes in path statements in R are forward not backward
DC1.data.file<-"C:/DataSHIELD.Example/DC1/Study.1.csv"
DC2.data.file<-"C:/DataSHIELD.Example/DC2/Study.2.csv"
DC3.data.file<-"C:/DataSHIELD.Example/DC3/Study.3.csv"
DC4.data.file<-"C:/DataSHIELD.Example/DC4/Study.4.csv"
DC5.data.file<-"C:/DataSHIELD.Example/DC5/Study.5.csv"
DC6.data.file<-"C:/DataSHIELD.Example/DC6/Study.6.csv"
AC.beta.vector<-"C:/DataSHIELD.Example/AC/beta.vector.csv"
ALL.data.file<-"C:/DataSHIELD.Example/Study.ALL.csv"

#SET UP DATA STRUCTURE
#Random number seed so others can precisely repeat analysis on their own implementation of R
set.seed(1028)
#Specify study sizes and generate IDs for studies and individuals
numsubs.study<-c(2000,3000,1500,300,2000,700)
numsubs<-sum(numsubs.study)
numstudies<-length(numsubs.study)
study.id<-rep(1:numstudies,numsubs.study)
id<-c(1:numsubs.study[1], 1:numsubs.study[2], 1:numsubs.study[3],
        1:numsubs.study[4], 1:numsubs.study[5], 1:numsubs.study[6])

#SET UP MODEL STRUCTURE AND PARAMETERS
#Number of and values of regression coefficients
numpara<-4
beta0<--0.3
beta.bmi<-0.02
beta.bmi456<-0.04
beta.snp<-0.5
#Minor allele frequency
MAF<-0.3

#SIMULATE DATA
#Generate covariates
bmi<- rnorm(numsubs,mean=23,sd=4)-23
bmi456<-c(rep(0,6500),bmi[6501:9500])
snp<-rbinom(numsubs,2,MAF)
```

```
#Generate linear predictor and equivalent probabilities of response
lp<-beta0 + beta.bmi*bmi  +beta.bmi456*bmi456 + beta.snp*snp
probresp<-exp(lp)/(1+exp(lp))
#Randomly sample case control status
CC<-rbinom(numsubs,1,probresp)

#ASSEMBLE AND WRITE OUT COMPLETE DATA SET
all.data<-data.frame(study.id,id,CC,bmi,snp,bmi456)
write.csv(all.data,file=ALL.data.file,row.names=FALSE)

#PREPARE AND WRITE OUT DATA FILES FOR EACH STUDY INDIVIDUALLY
Study<-list()
Study[[1]]<-all.data[study.id==1,]
write.csv(Study[[1]],file=DC1.data.file,row.names=FALSE)
Study[[2]]<-all.data[study.id==2,]
write.csv(Study[[2]],file=DC2.data.file,row.names=FALSE)
Study[[3]]<-all.data[study.id==3,]
write.csv(Study[[3]],file=DC3.data.file,row.names=FALSE)
Study[[4]]<-all.data[study.id==4,]
write.csv(Study[[4]],file=DC4.data.file,row.names=FALSE)
Study[[5]]<-all.data[study.id==5,]
write.csv(Study[[5]],file=DC5.data.file,row.names=FALSE)
Study[[6]]<-all.data[study.id==6,]
write.csv(Study[[6]],file=DC6.data.file,row.names=FALSE)
#>>>>>>>>>>>>>>>>>>>>>>>>END OF FIRST BLOCK OF R CODE>>>>>>>>>>>>>>>>>>>>>>>>
```

For the convenience of readers who do not have access to R, the six data files and the complete data data file can be obtained directly from our web site: http://www2.le.ac.uk/departments/health-sciences/extranet/BGE/genetic-epidemiology/softw-progs

S4. Full output from the pooled logistic regression model fitted using DataSHIELD in Figure 5 in the main text

1st Iteration:

$$\hat{b}_{r=1} = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$$

All data computers use this coefficient vector for iteration 1

Data Computer 1:

$$\begin{bmatrix} 500 & -11.61089 & 0 & 294.75 \\ -11.61089 & 7972.37088 & 0 & -25.55092 \\ 0 & 0 & 0 & 0 \\ 294.75 & -25.55092 & 0 & 387.75 \end{bmatrix}$$

$$\begin{bmatrix} -38 & 203.1316 & 0 & 87.5 \end{bmatrix}$$

Data Computer 2:

$$\begin{bmatrix} 750 & -8.417491 & 0 & 443 \\ -8.417491 & 12492.689094 & 0 & -11.777043 \\ 0 & 0 & 0 & 0 \\ 443 & -11.777043 & 0 & 578.5 \end{bmatrix}$$

$$\begin{bmatrix} -14 & 370.8722 & 0 & 162 \end{bmatrix}$$

Data Computer 3:

$$\begin{bmatrix} 375 & 34.88511 & 0 & 226.75 \\ 34.88511 & 6407.52995 & 0 & -26.82820 \\ 0 & 0 & 0 & 0 \\ 226.75 & -26.82820 & 0 & 293.75 \end{bmatrix}$$

$$\begin{bmatrix} 11 & -14.2244 & 0 & 70.5 \end{bmatrix}$$

Data Computer 4:

$$\begin{bmatrix} 75 & 16.13902 & 16.13902 & 47 \\ 16.13902 & 1265.49746 & 1265.49746 & -12.16424 \\ 16.13902 & 1265.49746 & 1265.49746 & -12.16424 \\ 47 & -12.16424 & -12.16424 & 61.5 \end{bmatrix}$$

$$\begin{bmatrix} -8 & 68.06208 & 68.06208 & 12 \end{bmatrix}$$

____

Data Computer 5:

$$\begin{bmatrix} 500 & 70.56657 & 70.56657 & 297 \\ 70.56657 & 7646.29164 & 7646.29164 & 65.39412 \\ 70.56657 & 7646.29164 & 7646.29164 & 65.39412 \\ 297 & 65.39412 & 65.39412 & 382 \end{bmatrix}$$

$$\begin{bmatrix} 36 & 487.2951 & 487.2951 & 149 \end{bmatrix}$$

____

Data Computer 6:

$$\begin{bmatrix} 175 & 11.5221 & 11.5221 & 102.25 \\ 11.5221 & 2864.847 & 2864.847 & -28.50817 \\ 11.5221 & 2864.847 & 2864.847 & -28.50817 \\ 102.25 & -28.50817 & -28.50817 & 132.25 \end{bmatrix}$$

$$\begin{bmatrix} 10 & 149.3701 & 149.3701 & 47.5 \end{bmatrix}$$

____

Information matrices and score vectors generated by each study are transmitted to AC.

Central Summation at AC:

$$\begin{bmatrix} 2375 & 113.08442 & 98.22769 & 1410.75 \\ 113.08442 & 38649.22602 & 11776.63610 & 39.43446 \\ 98.22769 & 11776.63610 & 11776.63610 & 24.72170 \\ 1410.75 & -39.43446 & 24.72170 & 1835.75 \end{bmatrix}$$

$$\begin{bmatrix} -3 & 1264.5067 & 704.7273 & 528.5 \end{bmatrix}$$

Convergence criterion tested: **Not met.**

Derive update vector:

$$\begin{bmatrix} -0.32183281 & 0.02228647 & 0.03911561 & 0.53516954 \end{bmatrix}$$

Add update vector to original coefficient vector to produce coefficient vector for second iteration:

$$= \begin{bmatrix} -0.32183281 & 0.02228647 & 0.03911561 & 0.53516954 \end{bmatrix}$$

<u>2nd Iteration:</u>

$$\begin{bmatrix} -0.32183281 & 0.02228647 & 0.03911561 & 0.53516954 \end{bmatrix}$$

Procedure used in iteration 1 repeated, all data computers use this coefficient vector for iteration 2

For clarity, the information matrix, score vector, and deviance contributions from the individual data computers are omitted from the presentation of this iteration.

Information matrices and score vectors are generated by each study and are transmitted to AC

Central Summation at AC:

$$\begin{bmatrix} 2295.0536 & 115.0395 & 92.74410 & 1338.4 \\ 115.0395 & 37006.6888 & 11006.04639 & -160.8173 \\ 92.7441 & 11006.0464 & 11006.04639 & -48.61056 \\ 1338.4 & -160.8173 & -48.61056 & 1707.81381 \end{bmatrix}$$

$$\begin{bmatrix} 4.679958 & 46.098158 & 29.761157 & 17.657043 \end{bmatrix}$$

Convergence Criterion: **Not met.**

Derive update vector:

$$\begin{bmatrix} -0.0077096093 & 0.0007061835 & 0.0021357681 & 0.0165082231 \end{bmatrix}$$

Add update vector to original coefficient vector to produce coefficient vector for third iteration:

$$= \begin{bmatrix} -0.32954242 & 0.02299265 & 0.04125137 & 0.55167776 \end{bmatrix}$$

<u>*3rd Iteration:*</u>

$$\begin{bmatrix} -0.32954242 & 0.02299265 & 0.04125137 & 0.55167776 \end{bmatrix}$$

Procedure used in iteration 1 repeated, all data computers use this coefficient vector for iteration 3

For clarity, the information matrix, score vector, and deviance contributions from the individual data computers are omitted from the presentation of this iteration.

Information matrices and score vectors are generated by each study and are transmitted to AC

Central Summation at AC:

$$\begin{bmatrix} 2290.07166 & 114.04663 & 91.77508 & 1333.57918 \\ 114.04663 & 36898.8956 & 10949.7004 & -169.08819 \\ 91.77508 & 10949.7004 & 10949.7004 & -53.88901 \\ 1333.57918 & -169.0882 & -53.88901 & 1699.55867 \end{bmatrix}$$

$$\begin{bmatrix} 0.02188718 & 0.16208605 & 0.11946433 & 0.05791435 \end{bmatrix}$$

Convergence Criterion: **Not met.**

Derive update vector:

$$\begin{bmatrix} -2.089082e\text{-}10 & 1.660537e\text{-}11 & 1.390907e\text{-}10 & 5.496857e\text{-}10 \end{bmatrix}$$

Add update vector to original coefficient vector to produce coefficient vector for third iteration:

$$= \begin{bmatrix} -0.32956275 & 0.02299454 & 0.04126082 & 0.55172828 \end{bmatrix}$$

4th Iteration:

$$\begin{bmatrix} -0.32956275 & 0.02299454 & 0.04126082 & 0.55172828 \end{bmatrix}$$

Procedure used in iteration 1 repeated, all data computers use this coefficient vector for iteration 3

For clarity, the information matrix, score vector, and deviance contributions from the individual data computers are omitted from the presentation of this iteration.

Information matrices and score vectors are generated by each study and are transmitted to AC

Central Summation at AC:

$$\begin{bmatrix} 2290.05551 & 114.04125 & 91.77065 & 1333.56304 \\ 114.04125 & 36898.5281 & 10949.48836 & -169.11693 \\ 91.77065 & 10949.48836 & 10949.48836 & -53.90879 \\ 1333.56304 & -169.11693 & -53.90879 & 1699.53140 \end{bmatrix}$$

$$\begin{bmatrix} 2.692875e\text{-}07 & 2.018901e\text{-}06 & 1.655988e\text{-}06 & 6.453095e\text{-}07 \end{bmatrix}$$

Convergence Criterion: **Met.**

Variance-covariance matrix now obtained by taking the inverse of the summed information matrix

$$\begin{bmatrix} 8.054620e\text{-}04 & -3.499749e\text{-}06 & -6.365439e\text{-}06 & -6.325681e\text{-}04 \\ -3.499749e\text{-}06 & 3.856388e\text{-}05 & -3.850815e\text{-}05 & 5.362074e\text{-}06 \\ -6.365439e\text{-}06 & -3.850815e\text{-}05 & 1.299160e\text{-}04 & 5.283779e\text{-}06 \\ -6.325681e\text{-}04 & 5.362074e\text{-}06 & 5.283779e\text{-}06 & 1.085453e\text{-}03 \end{bmatrix}$$

and standard errors are obtained by taking the square-root of the diagonal elements of this matrix.

Final Results:

Coefficient	**Estimate**	**Std Error**
Intercept	-0.32960	0.02838
BMI	0.02300	0.00621
BMI.456	0.04126	0.01140
SNP	0.55170	0.03295

Residual deviance: 12824.7 on 9496 degrees of freedom

S5. R Code used to undertake the partitioned IRLS analysis (*i.e.* the R code that generates the output in S4)

In order to use the following R code, cut at the top and bottom of the block, paste into a new “script file” in R and then run the script file

```
#>>>>>>>>>>>>>>>>>>>>>>>>START OF SECOND BLOCK OF R CODE>>>>>>>>>>>>>>>>>>>>>>>>
###########################
#  R CODE TO SET UP ANALYSIS  #
############################


#Specify folder for storing objects on the coordinating computer at the AC
AC.Directory<-"C:/DataSHIELD.Example/AC/"


#Create initial vector of regression coefficients for first iteration
beta.vect.next<-c(0,0,0,0)

#Save to the folder where data computers can find it
save(beta.vect.next,file=paste(AC.Directory,"beta.vect.next.RData",sep=""))

#Iterations need to be counted. Start off with the count at 0
#and increment by 1 at each new iteration
iteration.count<-0

#Provide arbitrary starting value for deviance to enable subsequent calculation of the
#change in deviance between iterations
dev.old<-9.99e+99

#Convergence state needs to be monitored. Start by allocating
#a “convergence not met” status
converge.state<-"NOT MET"

#Define a convergence criterion. This value of epsilon corresponds to that used
#by default for GLMs in R (see section S3 for details)
epsilon<-1.0e-08

#>>>>>>>>>>>>>>>>>>>>>>>>END OF SECOND BLOCK OF R CODE>>>>>>>>>>>>>>>>>>>>>>>>
```

In order to use the following R code, cut at the top and bottom of the block, paste into a new “script file” in R and then run the script file. This block needs to be run in full repeatedly. Each time it is run, R carries out a single extra iteration of the partitioned IRLS algorithm

```
#>>>>>>>>>>>>>>>>>>>>>>>>START OF THIRD BLOCK OF R CODE>>>>>>>>>>>>>>>>>>>>>>>>
##################################################################
#  R CODE TO CARRY OUT A PARTITIONED IRLS FIT ONE ITERATION AT A TIME #
#  RUN THIS WHOLE BLOCK OF CODE ONE ITERATION AT A TIME UNTIL THE     #
# MODEL OUTPUT INDICATES THAT CONVERGENCE HAS BEEN ACHIEVED
##################################################################

#Increment count of iterations
iteration.count<-iteration.count+1


#R CODE THAT WOULD RUN LOCALLY ON EACH OF THE REMOTE DATA COMPUTERS

#############
#START STUDY 1
#Load full local data from the specified local data file (in THIS particular simulated
#example, the file names and locations are specified in the simulation code
#in subsection S2)

#Read in full data
data.DC<-read.table(file=DC1.data.file, sep=",",header=T)

#Strip out header row
data.DC<-data.DC[,-1]

#Calculate number of subjects available in the current study
#(by enumerating length of ID column)
nsubs<-length(data.DC$id)

#Define design matrix (matrix of covariates) to contain BMI, SNP and the
#interaction covariate and add a column of 1s at the start for the regression constant
X.mat<-cbind(rep(1,nsubs),data.DC$bmi,data.DC$bmi456,data.DC$snp)


#Load the current value of the beta vector (vector of regression coefficients) from its
#location on the AC computer (stored during activation of block 2 of R code)
load(file=paste(AC.Directory,"beta.vect.next.RData",sep=""))


# Use this current value of the beta vector to calculate elements from the current study
beta.vect<-beta.vect.next


# Calculate linear predictors from observed covariate values and elements of
# current beta vector
lp.current<-beta.vect[1]+beta.vect[2]*X.mat[,2]+beta.vect[3]*X.mat[,3]+ beta.vect[4]*X.mat[,4]

# Apply inverse logistic transformation
mu.current<-exp(lp.current)/(1+exp(lp.current))
```

```
# Derive variance function and diagonal elements for weight matrix (using squared
# first differential of link function)
var.i<-(mu.current*(1-mu.current))
g2.i<-(1/(mu.current*(1-mu.current)))^2
W.mat<-diag(1/(var.i*g2.i))

#Calculate information matrix
info.matrix<-t(X.mat)%*%W.mat%*%X.mat

#Derive u terms for score vector
u.i<- (data.DC$CC-mu.current)* (1/(mu.current*(1-mu.current)))

#Calculate score vector
score.vect<-t(X.mat)%*%W.mat%*%u.i
#Calculate log likelihood and deviance contribution for current study

#For convenience, ignore the element of deviance that relates to the full saturated
# model, because that will cancel out in calculating the change in deviance from one
# iteration to the next (Dev.total – Dev.old [see below]) because the element relating
# to the saturated model will be the same at every iteration).
log.L<-sum(data.DC$CC*log(mu.current) + (1-data.DC$CC)*log(1-mu.current))
dev<- -2*log.L

#Create study specific versions of all key model components
info.matrix.1<-info.matrix
score.vect.1<-score.vect
dev.1<-dev
nsubs.1<-nsubs

#Send all of the key model components from the current study to the AC
save(info.matrix.1,file=paste(AC.Directory,"info.matrix.1.RData",sep=""))
save(score.vect.1,file=paste(AC.Directory,"score.vect.1.RData",sep=""))
save(dev.1,file=paste(AC.Directory,"dev.1.RData",sep=""))
save(nsubs.1,file=paste(AC.Directory,"nsubs.1.RData",sep=""))

#END STUDY 1
###########



#############
#START STUDY 2
#Load full local data from the specified local data file (in THIS particular simulated
#example, the file names and locations are specified in the simulation code
#in subsection S2)

#Read in full data
data.DC<-read.table(file=DC2.data.file, sep=",",header=T)

#Strip out header row
data.DC<-data.DC[,-1]
```

```
#Calculate number of subjects available in the current study
#(by enumerating length of ID column)
nsubs<-length(data.DC$id)

#Define design matrix (matrix of covariates) to contain BMI, SNP and the
#interaction covariate and add a column of 1s at the start for the regression constant
X.mat<-cbind(rep(1,nsubs),data.DC$bmi,data.DC$bmi456,data.DC$snp)


#Load the current value of the beta vector (vector of regression coefficients) from its
#location on the AC computer (stored during activation of block 2 of R code)
load(file=paste(AC.Directory,"beta.vect.next.RData",sep=""))


# Use this current value of the beta vector to calculate elements from the current study
beta.vect<-beta.vect.next


# Calculate linear predictors from observed covariate values and elements of
# current beta vector
lp.current<-beta.vect[1]+beta.vect[2]*X.mat[,2]+beta.vect[3]*X.mat[,3]+ beta.vect[4]*X.mat[,4]

# Apply inverse logistic transformation
mu.current<-exp(lp.current)/(1+exp(lp.current))

# Derive variance function and diagonal elements for weight matrix (using squared
# first differential of link function)
var.i<-(mu.current*(1-mu.current))
g2.i<-(1/(mu.current*(1-mu.current)))^2
W.mat<-diag(1/(var.i*g2.i))

#Calculate information matrix
info.matrix<-t(X.mat)%*%W.mat%*%X.mat

#Derive u terms for score vector
u.i<- (data.DC$CC-mu.current)* (1/(mu.current*(1-mu.current)))

#Calculate score vector
score.vect<-t(X.mat)%*%W.mat%*%u.i

#Calculate log likelihood and deviance contribution for current study

#For convenience, ignore the element of deviance that relates to the full saturated
# model, because that will cancel out in calculating the change in deviance from one
# iteration to the next (Dev.total – Dev.old [see below]) because the element relating
# to the saturated model will be the same at every iteration).
log.L<-sum(data.DC$CC*log(mu.current) + (1-data.DC$CC)*log(1-mu.current))
dev<- -2*log.L

#Create study specific versions of all key model components
info.matrix.2<-info.matrix
score.vect.2<-score.vect
dev.2<-dev
nsubs.2<-nsubs
```

```
#Send all of the key model components from the current study to the AC
save(info.matrix.2,file=paste(AC.Directory,"info.matrix.2.RData",sep=""))
save(score.vect.2,file=paste(AC.Directory,"score.vect.2.RData",sep=""))
save(dev.2,file=paste(AC.Directory,"dev.2.RData",sep=""))
save(nsubs.2,file=paste(AC.Directory,"nsubs.2.RData",sep=""))

#END STUDY 2
###########


#############
#START STUDY 3
#Load full local data from the specified local data file (in THIS particular simulated
#example, the file names and locations are specified in the simulation code
#in subsection S2)

#Read in full data
data.DC<-read.table(file=DC3.data.file, sep=",",header=T)

#Strip out header row
data.DC<-data.DC[,-1]

#Calculate number of subjects available in the current study
#(by enumerating length of ID column)
nsubs<-length(data.DC$id)

#Define design matrix (matrix of covariates) to contain BMI, SNP and the
#interaction covariate and add a column of 1s at the start for the regression constant
X.mat<-cbind(rep(1,nsubs),data.DC$bmi,data.DC$bmi456,data.DC$snp)

#Load the current value of the beta vector (vector of regression coefficients) from its
#location on the AC computer (stored during activation of block 2 of R code)
load(file=paste(AC.Directory,"beta.vect.next.RData",sep=""))

# Use this current value of the beta vector to calculate elements from the current study
beta.vect<-beta.vect.next

# Calculate linear predictors from observed covariate values and elements of
# current beta vector
lp.current<-beta.vect[1]+beta.vect[2]*X.mat[,2]+beta.vect[3]*X.mat[,3]+ beta.vect[4]*X.mat[,4]

# Apply inverse logistic transformation
mu.current<-exp(lp.current)/(1+exp(lp.current))

# Derive variance function and diagonal elements for weight matrix (using squared
# first differential of link function)
var.i<-(mu.current*(1-mu.current))
g2.i<-(1/(mu.current*(1-mu.current)))^2
W.mat<-diag(1/(var.i*g2.i))

#Calculate information matrix
info.matrix<-t(X.mat)%*%W.mat%*%X.mat
```

```
#Derive u terms for score vector
u.i<- (data.DC$CC-mu.current)* (1/(mu.current*(1-mu.current)))

#Calculate score vector
score.vect<-t(X.mat)%*%W.mat%*%u.i

#Calculate log likelihood and deviance contribution for current study

#For convenience, ignore the element of deviance that relates to the full saturated
# model, because that will cancel out in calculating the change in deviance from one
# iteration to the next (Dev.total – Dev.old [see below]) because the element relating
# to the saturated model will be the same at every iteration).
log.L<-sum(data.DC$CC*log(mu.current) + (1-data.DC$CC)*log(1-mu.current))
dev<- -2*log.L

#Create study specific versions of all key model components
info.matrix.3<-info.matrix
score.vect.3<-score.vect
dev.3<-dev
nsubs.3<-nsubs

#Send all of the key model components from the current study to the AC
save(info.matrix.3,file=paste(AC.Directory,"info.matrix.3.RData",sep=""))
save(score.vect.3,file=paste(AC.Directory,"score.vect.3.RData",sep=""))
save(dev.3,file=paste(AC.Directory,"dev.3.RData",sep=""))
save(nsubs.3,file=paste(AC.Directory,"nsubs.3.RData",sep=""))

#END STUDY 3
###########


#############
#START STUDY 4
#Load full local data from the specified local data file (in THIS particular simulated
#example, the file names and locations are specified in the simulation code
#in subsection S2)

#Read in full data
data.DC<-read.table(file=DC4.data.file, sep=",",header=T)

#Strip out header row
data.DC<-data.DC[,-1]

#Calculate number of subjects available in the current study
#(by enumerating length of ID column)
nsubs<-length(data.DC$id)

#Define design matrix (matrix of covariates) to contain BMI, SNP and the
#interaction covariate and add a column of 1s at the start for the regression constant
X.mat<-cbind(rep(1,nsubs),data.DC$bmi,data.DC$bmi456,data.DC$snp)

#Load the current value of the beta vector (vector of regression coefficients) from its
#location on the AC computer (stored during activation of block 2 of R code)
load(file=paste(AC.Directory,"beta.vect.next.RData",sep=""))
```

```
# Use this current value of the beta vector to calculate elements from the current study
beta.vect<-beta.vect.next

# Calculate linear predictors from observed covariate values and elements of
# current beta vector
lp.current<-beta.vect[1]+beta.vect[2]*X.mat[,2]+beta.vect[3]*X.mat[,3]+ beta.vect[4]*X.mat[,4]

# Apply inverse logistic transformation
mu.current<-exp(lp.current)/(1+exp(lp.current))

# Derive variance function and diagonal elements for weight matrix (using squared
# first differential of link function)
var.i<-(mu.current*(1-mu.current))
g2.i<-(1/(mu.current*(1-mu.current)))^2
W.mat<-diag(1/(var.i*g2.i))

#Calculate information matrix
info.matrix<-t(X.mat)%*%W.mat%*%X.mat

#Derive u terms for score vector
u.i<- (data.DC$CC-mu.current)* (1/(mu.current*(1-mu.current)))

#Calculate score vector
score.vect<-t(X.mat)%*%W.mat%*%u.i

#Calculate log likelihood and deviance contribution for current study

#For convenience, ignore the element of deviance that relates to the full saturated
# model, because that will cancel out in calculating the change in deviance from one
# iteration to the next (Dev.total – Dev.old [see below]) because the element relating
# to the saturated model will be the same at every iteration).
log.L<-sum(data.DC$CC*log(mu.current) + (1-data.DC$CC)*log(1-mu.current))
dev<- -2*log.L

#Create study specific versions of all key model components
info.matrix.4<-info.matrix
score.vect.4<-score.vect
dev.4<-dev
nsubs.4<-nsubs

#Send all of the key model components from the current study to the AC
save(info.matrix.4,file=paste(AC.Directory,"info.matrix.4.RData",sep=""))
save(score.vect.4,file=paste(AC.Directory,"score.vect.4.RData",sep=""))
save(dev.4,file=paste(AC.Directory,"dev.4.RData",sep=""))
save(nsubs.4,file=paste(AC.Directory,"nsubs.4.RData",sep=""))

#END STUDY 4
###########
```

```
#############
#START STUDY 5
#Load full local data from the specified local data file (in THIS particular simulated
#example, the file names and locations are specified in the simulation code
#in subsection S2)

#Read in full data
data.DC<-read.table(file=DC5.data.file, sep=",",header=T)

#Strip out header row
data.DC<-data.DC[,-1]

#Calculate number of subjects available in the current study
#(by enumerating length of ID column)
nsubs<-length(data.DC$id)

#Define design matrix (matrix of covariates) to contain BMI, SNP and the
#interaction covariate and add a column of 1s at the start for the regression constant
X.mat<-cbind(rep(1,nsubs),data.DC$bmi,data.DC$bmi456,data.DC$snp)

#Load the current value of the beta vector (vector of regression coefficients) from its
#location on the AC computer (stored during activation of block 2 of R code)
load(file=paste(AC.Directory,"beta.vect.next.RData",sep=""))

# Use this current value of the beta vector to calculate elements from the current study
beta.vect<-beta.vect.next

# Calculate linear predictors from observed covariate values and elements of
# current beta vector
lp.current<-beta.vect[1]+beta.vect[2]*X.mat[,2]+beta.vect[3]*X.mat[,3]+ beta.vect[4]*X.mat[,4]

# Apply inverse logistic transformation
mu.current<-exp(lp.current)/(1+exp(lp.current))

# Derive variance function and diagonal elements for weight matrix (using squared
# first differential of link function)
var.i<-(mu.current*(1-mu.current))
g2.i<-(1/(mu.current*(1-mu.current)))^2
W.mat<-diag(1/(var.i*g2.i))

#Calculate information matrix
info.matrix<-t(X.mat)%*%W.mat%*%X.mat

#Derive u terms for score vector
u.i<- (data.DC$CC-mu.current)* (1/(mu.current*(1-mu.current)))

#Calculate score vector
score.vect<-t(X.mat)%*%W.mat%*%u.i

#Calculate log likelihood and deviance contribution for current study
```

```
#For convenience, ignore the element of deviance that relates to the full saturated
# model, because that will cancel out in calculating the change in deviance from one
# iteration to the next (Dev.total – Dev.old [see below]) because the element relating
# to the saturated model will be the same at every iteration).
log.L<-sum(data.DC$CC*log(mu.current) + (1-data.DC$CC)*log(1-mu.current))
dev<- -2*log.L

#Create study specific versions of all key model components
info.matrix.5<-info.matrix
score.vect.5<-score.vect
dev.5<-dev
nsubs.5<-nsubs

#Send all of the key model components from the current study to the AC
save(info.matrix.5,file=paste(AC.Directory,"info.matrix.5.RData",sep=""))
save(score.vect.5,file=paste(AC.Directory,"score.vect.5.RData",sep=""))
save(dev.5,file=paste(AC.Directory,"dev.5.RData",sep=""))
save(nsubs.5,file=paste(AC.Directory,"nsubs.5.RData",sep=""))

#END STUDY 5
###########



#############
#START STUDY 6
#Load full local data from the specified local data file (in THIS particular simulated
#example, the file names and locations are specified in the simulation code
#in subsection S2)

#Read in full data
data.DC<-read.table(file=DC6.data.file, sep=",",header=T)

#Strip out header row
data.DC<-data.DC[,-1]

#Calculate number of subjects available in the current study
#(by enumerating length of ID column)
nsubs<-length(data.DC$id)

#Define design matrix (matrix of covariates) to contain BMI, SNP and the
#interaction covariate and add a column of 1s at the start for the regression constant
X.mat<-cbind(rep(1,nsubs),data.DC$bmi,data.DC$bmi456,data.DC$snp)

#Load the current value of the beta vector (vector of regression coefficients) from its
#location on the AC computer (stored during activation of block 2 of R code)
load(file=paste(AC.Directory,"beta.vect.next.RData",sep=""))

# Use this current value of the beta vector to calculate elements from the current study
beta.vect<-beta.vect.next

# Calculate linear predictors from observed covariate values and elements of
# current beta vector
lp.current<-beta.vect[1]+beta.vect[2]*X.mat[,2]+beta.vect[3]*X.mat[,3]+ beta.vect[4]*X.mat[,4]
```

```
# Apply inverse logistic transformation
mu.current<-exp(lp.current)/(1+exp(lp.current))

# Derive variance function and diagonal elements for weight matrix (using squared
# first differential of link function)
var.i<-(mu.current*(1-mu.current))
g2.i<-(1/(mu.current*(1-mu.current)))^2
W.mat<-diag(1/(var.i*g2.i))

#Calculate information matrix
info.matrix<-t(X.mat)%*%W.mat%*%X.mat

#Derive u terms for score vector
u.i<- (data.DC$CC-mu.current)* (1/(mu.current*(1-mu.current)))

#Calculate score vector
score.vect<-t(X.mat)%*%W.mat%*%u.i

#Calculate log likelihood and deviance contribution for current study

#For convenience, ignore the element of deviance that relates to the full saturated
# model, because that will cancel out in calculating the change in deviance from one
# iteration to the next (Dev.total – Dev.old [see below]) because the element relating
# to the saturated model will be the same at every iteration).
log.L<-sum(data.DC$CC*log(mu.current) + (1-data.DC$CC)*log(1-mu.current))
dev<- -2*log.L

#Create study specific versions of all key model components
info.matrix.6<-info.matrix
score.vect.6<-score.vect
dev.6<-dev
nsubs.6<-nsubs

#Send all of the key model components from the current study to the AC
save(info.matrix.6,file=paste(AC.Directory,"info.matrix.6.RData",sep=""))
save(score.vect.6,file=paste(AC.Directory,"score.vect.6.RData",sep=""))
save(dev.6,file=paste(AC.Directory,"dev.6.RData",sep=""))
save(nsubs.6,file=paste(AC.Directory,"nsubs.6.RData",sep=""))

#END STUDY 6
###########



##########
#ITERATION ON ALL LOCAL COMPUTERS NOW COMPLETED
# KEY MODEL ELEMENTS HAVE BEEN TRANSMITTED TO AC

#AC WILL NOW USE THESE ELEMENTS TO GENERATE UPDATE TERMS
```

```
#Read back into R, the key elements generated by the local data computers and
#sent to the AC
load(file=paste(AC.Directory,"info.matrix.1.RData",sep=""))
load(file=paste(AC.Directory,"info.matrix.2.RData",sep=""))
load(file=paste(AC.Directory,"info.matrix.3.RData",sep=""))
load(file=paste(AC.Directory,"info.matrix.4.RData",sep=""))
load(file=paste(AC.Directory,"info.matrix.5.RData",sep=""))
load(file=paste(AC.Directory,"info.matrix.6.RData",sep=""))

load(file=paste(AC.Directory,"score.vect.1.RData",sep=""))
load(file=paste(AC.Directory,"score.vect.2.RData",sep=""))
load(file=paste(AC.Directory,"score.vect.3.RData",sep=""))
load(file=paste(AC.Directory,"score.vect.4.RData",sep=""))
load(file=paste(AC.Directory,"score.vect.5.RData",sep=""))
load(file=paste(AC.Directory,"score.vect.6.RData",sep=""))

load(file=paste(AC.Directory,"dev.1.RData",sep=""))
load(file=paste(AC.Directory,"dev.2.RData",sep=""))
load(file=paste(AC.Directory,"dev.3.RData",sep=""))
load(file=paste(AC.Directory,"dev.4.RData",sep=""))
load(file=paste(AC.Directory,"dev.5.RData",sep=""))
load(file=paste(AC.Directory,"dev.6.RData",sep=""))

load(file=paste(AC.Directory,"nsubs.1.RData",sep=""))
load(file=paste(AC.Directory,"nsubs.2.RData",sep=""))
load(file=paste(AC.Directory,"nsubs.3.RData",sep=""))
load(file=paste(AC.Directory,"nsubs.4.RData",sep=""))
load(file=paste(AC.Directory,"nsubs.5.RData",sep=""))
load(file=paste(AC.Directory,"nsubs.6.RData",sep=""))

#Read in the current beta vector
load(file=paste(AC.Directory,"beta.vect.next.RData",sep=""))


#Sum the key elements across all studies
info.matrix.total<-info.matrix.1+info.matrix.2+info.matrix.3+
                   info.matrix.4+info.matrix.5+info.matrix.6

score.vect.total<-score.vect.1+score.vect.2+score.vect.3+
                  score.vect.4+score.vect.5+score.vect.6

dev.total<-dev.1+dev.2+dev.3+dev.4+dev.5+dev.6

nsubs.total<-nsubs.1+nsubs.2+nsubs.3+nsubs.4+nsubs.5+nsubs.6

#Create variance covariance matrix as inverse of information matrix
#(solve() denotes matrix inversion in R )
variance.covariance.matrix.total<-solve(info.matrix.total)

#Create beta vector update terms (see subsection S2)
beta.update.vect<-variance.covariance.matrix.total %*% score.vect.total

#Add update terms to current beta vector to obtain new beta vector for next iteration
beta.vect.next<-beta.vect.next+beta.update.vect
```

```
#Calculate value of convergence statistic and test whether meets convergence criterion
#(see subsection S2)
converge.value<-abs(dev.total-dev.old)/(abs(dev.total)+0.1)
if(converge.value<=epsilon)converge.state<-"MET"
if(converge.value>epsilon)dev.old<-dev.total


#If this is first iteration print out information matrix, score vector, deviance
#and number of subjects for each individual study (to reflect
#content of supplementary materials in subsection S4)

if(iteration.count==1)
{
print("Components from individual studies at first iteration")
cat("\n\n\nSTUDY 1\n")
cat("\nInformation matrix\n")
print(info.matrix.1)
cat("\nScore vector\n")
print(score.vect.1)
cat("\nDeviance\n")
print(dev.1)
cat("\nSample size\n")
print(nsubs.1)

cat("\n\n\nSTUDY 2\n")
cat("\nInformation matrix\n")
print(info.matrix.2)
cat("\nScore vector\n")
print(score.vect.2)
cat("\nDeviance\n")
print(dev.2)
cat("\nSample size\n")
print(nsubs.2)

cat("\n\n\nSTUDY 3\n")
cat("\nInformation matrix\n")
print(info.matrix.3)
cat("\nScore vector\n")
print(score.vect.3)
cat("\nDeviance\n")
print(dev.3)
cat("\nSample size\n")
print(nsubs.3)

cat("\n\n\nSTUDY 4\n")
cat("\nInformation matrix\n")
print(info.matrix.4)
cat("\nScore vector\n")
print(score.vect.4)
cat("\nDeviance\n")
print(dev.4)
cat("\nSample size\n")
print(nsubs.4)
```

```
cat("\n\n\nSTUDY 5\n")
cat("\nInformation matrix\n")
print(info.matrix.5)
cat("\nScore vector\n")
print(score.vect.5)
cat("\nDeviance\n")
print(dev.5)
cat("\nSample size\n")
print(nsubs.5)

cat("\n\n\nSTUDY 6\n")
cat("\nInformation matrix\n")
print(info.matrix.6)
cat("\nScore vector\n")
print(score.vect.6)
cat("\nDeviance\n")
print(dev.6)
cat("\nSample size\n")
print(nsubs.6)
}

#For ALL iterations summarise model state after current iteration

cat("\nSUMMARY OF MODEL STATE after iteration No",iteration.count,
   "\n\nCurrent deviance",dev.total,"on",
   (nsubs.total-length(beta.vect.next)), "degrees of freedom",
   "\nConvergence criterion    ",converge.state,"\n\n")

cat("Information matrix overall\n")
print(info.matrix.total)

cat("Score vector overall\n")
print(score.vect.total)

#If convergence has been obtained, declare final (maximum likelihood) beta vector,
#and calculate the corresponding standard errors, z scores and p values
#(the latter two to be consistent with the output of a standard GLM analysis)
#Then print out final model summary

if(converge.value<=epsilon)
{
beta.vect.final<-beta.vect.next
se.vect.final<-sqrt(diag(variance.covariance.matrix.total))
z.vect.final<-beta.vect.final/se.vect.final
pval.vect.final<-2*pnorm(-abs(z.vect.final))

model.parameters<-cbind(beta.vect.final,se.vect.final,z.vect.final,pval.vect.final)
dimnames(model.parameters)<-
list(c("Intercept","BMI","SNP","BMI.456"),c("Coefficient","SE","z-value","p-value"))

model.parameters<-signif(model.parameters,digits=4)
```

```
#If converged print out final model summary
cat("\n\nFINAL MODEL\n")

print(model.parameters)

cat("\nCurrent deviance",dev.total,"on",(nsubs.total-length(beta.vect.next)), "degrees of
freedom","\nAfter iteration No",iteration.count,"\n")
}


#Repeat summary of final model state
cat("\nSUMMARY OF MODEL STATE after iteration No",iteration.count,
   "\n\nCurrent deviance",dev.total,"on",
   (nsubs.total-length(beta.vect.next)), "degrees of freedom",
   "\nConvergence criterion    ",converge.state,"\n\n")


#Update the stored value of the beta vector to reflect the current estimate – to set
#up the next iteration
save(beta.vect.next,file=paste(AC.Directory,"beta.vect.next.RData",sep=""))


#>>>>>>>>>>>>>>>>>>>>>>>>>END OF THIRD BLOCK OF R CODE>>>>>>>>>>>>>>>>>>>>>>>>>
```

S6. R code and output for a conventional unconditional logistic regression model (glm in R) fitted on a pooled individual level data file generated from all six studies combined

```
#Fit model on all data sets combined
ALL.data.file<-"C:/DataSHIELD.Example/Study.ALL.csv"
ALL.data<-read.table(file=ALL.data.file, sep=",",header=T)

summary(glm(CC~bmi+ bmi456 + snp,family=binomial(logit),data=ALL.data))
```

Output:

Coefficients:

	Estimate	Std Error	Z value	Pr(>|z)
(Intercept)	-0.32956	0.02838	-11.612	<2e-16 ***
BMI	0.02300	0.00621	3.703	0.000213 ***
BMI.456	0.04126	0.01140	3.620	0.000295 ***
SNP	0.55173	0.03295	16.746	< 2e-16 ***

Residual deviance: 12825 on 9496 degrees of freedom

Output from DataSHIELD analysis (above) for comparison

Final Results:

Coefficient	Estimate	Std Error
Intercept	-0.32960	0.02838
BMI	0.02300	0.00621
BMI.456	0.04126	0.01140
SNP	0.55170	0.03295

Residual deviance: 12824.7 on 9496 degrees of freedom

Comment

The estimates, standard errors and deviance obtained from the partitioned analysis preformed using DataSHIELD are identical (aside from rounding errors) from those obtained using a conventional GLM analysis on a single data set containing the individual level data from all six studies combined.

REFERENCES

1. Aitkin M, Anderson D, Francis B, Hinde J. *Statistical Modelling in GLIM*. Oxford: Clarendon Press; 1989.

2. McCullagh P, Nelder J. *Generalized linear models*. London: Chapman and Hall; 1989.