DIGITAL ACCESS TO SCHOLARSHIP AT HARVARD
DASH.HARVARD.EDU

HARVARD LIBRARY
Office for Scholarly Communication

# El: A Program for Ecological Inference

**Citation**
King, Gary. 2004. El: A program for ecological inference. Journal of Statistical Software 11(7): 1-41.

**Published Version**
http://www.jstatsoft.org/

**Permanent link**
http://nrs.harvard.edu/urn-3:HUL.InstRepos:3965179

**Terms of Use**
This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA

## Share Your Story

The Harvard community has made this article openly available.
Please share how this access benefits you. Submit a story .

Accessibility

Journal of Statistical Software

*September 2004, Volume 11, Issue 7.* http://www.jstatsoft.org/

# $\mathfrak{E}I$: A Program for Ecological Inference

**Gary King**
Harvard University

**Abstract**

The program $\mathfrak{E}I$ provides a method of inferring individual behavior from aggregate data. It implements the statistical procedures, diagnostics, and graphics from the book *A Solution to the Ecological Inference Problem: Reconstructing Individual Behavior from Aggregate Data* (King 1997).

*Ecological inference*, as traditionally defined, is the process of using aggregate (i.e., "ecological") data to infer discrete individual-level relationships of interest when individual-level data are not available. Ecological inferences are required in political science research when individual-level surveys are unavailable (e.g., local or comparative electoral politics), unreliable (racial politics), insufficient (political geography), or infeasible (political history). They are also required in numerous areas of major significance in public policy (e.g., for applying the Voting Rights Act) and other academic disciplines ranging from epidemiology and marketing to sociology and quantitative history.

*Keywords*: ecological inference, aggregation, random effects, conditional inference.

## 1. Introduction

$\mathfrak{E}I$ provides a method of inferring individual behavior from aggregate data that works in practice. It implements the statistical procedures, diagnostics, and graphics from the book *A Solution to the Ecological Inference Problem: Reconstructing Individual Behavior from Aggregate Data* (King 1997). Please read the book prior to trying this program (a sample chapter and other related information is available at my web site `http://GKing.Harvard.Edu/preprints.shtml#sei`. Except where indicated, all references to page, section, chapter, table, and figure numbers in this document refer to the book.

*Ecological inference*, as traditionally defined, is the process of using aggregate (i.e., "ecological") data to infer discrete individual-level relationships of interest when individual-level data are not available. As existing methods usually lead to inaccurate conclusions about the empirical world, the ecological inference *problem* had been to develop a method that gives accurate answers. Ecological inferences are required in political science research when individual-level

surveys are unavailable (e.g., local or comparative electoral politics), unreliable (racial politics), insufficient (political geography), or infeasible (political history). They are also required in numerous areas of major significance in public policy (e.g., for applying the Voting Rights Act) and other academic disciplines ranging from epidemiology and marketing to sociology and quantitative history. Most researchers using aggregate data have encountered some form of the ecological inference problem.

Because the ecological inference problem is caused by the lack of individual-level information, no method of ecological inference, including that introduced in this book and estimated by this program, will produce precisely accurate results in every instance. However, potential difficulties are minimized here by models that include more available information, diagnostics to evaluate when assumptions need to be modified, easy methods of modifying the assumptions, and uncertainty estimates for all quantities of interest. I recommend reviewing Chapter 16 while using this program for actual research.

## 2. Hardware and software requirements

$\mathfrak{E}I$ is written in GAUSS, and will run on any computer hardware and operating system that runs GAUSS. The GAUSS module CML (constrained maximum likelihood, by Ronald J. Schoenberg) is also required. No other special hardware or software is required except the program that accompanies this documentation. A menu-oriented, stand-alone version of this program, by Kenneth Benoit and me, is available at my homepage, http://GKing.Harvard.Edu/stats.shtml. This version, $\mathfrak{E}zI$: *A(n easy) Program for Ecological Inference*, does not require GAUSS, but it is somewhat less flexible.

GAUSS and CML are available for DOS/Windows, Windows, UNIX, and other operating systems from Aptech Systems, Inc.; 23804 S.E. Kent-Kangley Road; Maple Valley, Washington 98038; (206) 432-7855; FAX: (206) 432-7832; E-mail: sales@aptech.com. If you do not have a lot of RAM, but you do have a lot of observations, consider turning on GAUSS' virtual memory feature.

## 3. Installation

If you received the program in a zip file, unzip it (with the public domain program unzip).

**On a DOS or Windows-based system:** Copy `ei.lcg` to your `lib` subdirectory, such as `\GAUSS\LIB`, and copy the remaining files (`*.src`, `ei.dec`, `ei.ext`, and `sample.asc`) to somewhere on the GAUSSPATH or src_path, such as the default `\GAUSS\SRC`.

**On a UNIX-based system:** If you have supervisor privleges, follow analogous steps as for DOS systems. If you do not have supervisor privileges, or do not wish the program to be installed for all users, unzip ei.zip into a directory and then specify ei in the library command with a path, e.g.: `library /files/gking/ecinf/ei;`.

**Information on Program Updates** See the file `whatsnew` for a list of the major recent and planned changes to $\mathfrak{E}I$, and the web page http://GKing.Harvard.Edu/netmind.shtml

| **Race of Voting Age Person** | **Voting Decision** Vote | No vote | |
|---|---|---|---|
| black | $\beta_i^b$ | $1-\beta_i^b$ | $X_i$ |
| white | $\beta_i^w$ | $1-\beta_i^w$ | $1-X_i$ |
| | $T_i$ | $1-T_i$ | |

Table 1: *Notation for Precinct $i$. The goal is to infer the quantities of interest, $\beta_i^b$ (the fraction of blacks who vote) and $\beta_i^w$ (the fraction of whites who vote), from the aggregate variables $X_i$ (the fraction of voting age people who are black) and $T_i$ (the fraction of people who vote), along with $N_i$ (the number of voting age people). This is Table 2.3 (page 31) in the book.*

to be automatically notified (via email) of future program updates.

## 4. Overview

As only four commands are required to use $\mathfrak{E}I$ the program can be easily run interactively, or in batch mode as a regular GAUSS program (other GAUSS commands can be used at any point if desired). Each command may also be used with many optional globals and subcommands. An example of these commands are as follows:

```
library ei;                  @ initialize EI @
dbuf = ei(t,x,n,1,1);        @ run main prog., save results in dbuf @
call eigraph(dbuf,"tomog"); @ draw tomography graph @
v = eiread(dbuf,"betab");    @ extract precinct estimates from dbuf @
```

In most applications, `eigraph` and `eiread` would likely be run multiple times with different options chosen, and other commands would be included with these four to read in the data (`t`, `x`, and `n`).[1]

In this section, I describe these four commands through a very simple use of $\mathfrak{E}I$. (Refer to the reference section below for further details and more sophisticated uses. That section also includes sample data you can run with simple examples, and minor differences required when running under Unix.) For this purpose, and without loss of generality, I use the running example from the book portrayed in Table 2.3 (page 31) and reproduced below in Table 1. This example uses the fraction of the voting age population who are black ($X_i$), the fraction turning out to vote ($T_i$), and the number of voting age people ($N_i$) in each precinct ($i = 1, \ldots, p$) to infer the fraction of blacks who vote ($\beta_i^b$) and the fraction of whites who vote ($\beta_i^w$), also in each precinct.

Of course, this is only an example; $\mathfrak{E}I$ can be used for any analogous ecological inference. It can be used for *larger tables*, as described in Sections 8.4 and 15.1 of the book (see procedure `ei2` in the reference section). $\mathfrak{E}I$ can also be used without modification if $T_i$ is an average of individual-level *dichotomous variables* (as in the example), *interval-coded discrete variables*,

[1]For example, to read in the data from an ascii text file, you can include these two commands after the `library` command: `clear t,x,n;` and `loadvars sample.asc t x n`. A detailed example appears on page 5.

or fully *continuous variables*, so long as the variables are scaled to the [0,1] interval (see Section 14.3).

1. At the start of every program, use `library ei;` to initialize $\mathfrak{E}I$. (To reinitialize all globals between programs, use `eiset;` if desired.)

2. Run the main procedure, `dbuf = ei(t,x,n,1,1);`, which takes three $p \times 1$ vectors as *inputs*: `t` (e.g., the fraction of the voting age population turning out to vote); `x` (e.g., the fraction of the voting age population who are black); and `n` (e.g., the total number of people in the voting age population). (The remaining two inputs are for optional covariates; for the basic model, set them each to 1 for no covariates.) The *output* of this procedure is `dbuf`, a gauss data buffer. A data buffer is a single entity that can be saved from gauss in a single file on disk, but which can include many different strings, vectors, matrices, or other elements (see the GAUSS command `vput` in the GAUSS manual for more information). After running `ei`, it is a good idea to save `dbuf` on disk (with a GAUSS command such as `save dbuf;`). While `ei` is running, various intermediate results appear on the screen, such as iteration numbers, but no final results of substantive interest are printed. As this step is by far the longest (about 2–5 minutes for 250 observations on an HP 715/80 or Pentium 90), it is usually most convenient to run this separately from the remaining steps.

3. The output data buffer from `ei` includes a large variety of different results useful for understanding the results of the analysis. A minimal set of nonrepetitive information is stored in this data buffer, and a large variety of other information can be easily computed from it. Fortunately, you do not need to know whether the information you request is stored or computed as both are treated the same. [2] To extract information from the data buffer, two procedures are available:

   (a) For *graphics*, use `eigraph(dbuf,"name");`, where `dbuf` is the data buffer that is the output of `ei`, and `name` can be any of a long list of ready-made graphs. For example, use `eigraph(dbuf,"fit");` to assess the fit of the model, `eigraph(dbuf,"tomog");` to print a tomography graph, or `eigraph(dbuf,"xgraph");` to display a scatter-cross graph.

   (b) For *numerical information*, use `v = eiread(dbuf,"name")`, where `v` is the item extracted, `dbuf` is the data buffer that is the output of `ei`, and `name` can be any of a long list of output possibilities. For example, use `betab` for a vector of point estimates of $\beta_i^b$, `ci80w` for 80% confidence intervals for $\beta_i^w$, or `sum` to print a summary of district-level estimates and information.

[2] Also stored in the output data buffer are the data input to this procedure, and all global options chosen during the analysis. As such, this data structure is very convenient if you are in the habit of following *the replication standard* by making publically available all the data and information necessary to replicate your published analyses. In fact, the output data buffer is an automatically created and self-documented "replication data set." To follow the replication standard, you only need to provide this $\mathfrak{E}I$ data buffer to the ICPSR or some other public archive; others will then have access to the data and information necessary to replicate your results (of course, they won't have any other information unless you decide to provide it). To replicate an $\mathfrak{E}I$ analysis, you only need to load in the data buffer, `loadm dbuf;`, and call this procedure: `dbufNew=eirepl(dbuf);`. See King (1995).

# 5. Advanced topics: Bayesian model averaging

$\mathfrak{E}I$ includes facilities to define and run multiple $\mathfrak{E}I$ models all at once, and to combine them using formal Bayesian model averaging procedures. The procedure, `dbufdef = eimodels_def( dbufdef,num,t,x,n,zb,zw);`, enables you to define a model specificiation (including all input variables `t,x,n,zb,zw`, and all globals input to $\mathfrak{E}I$), to assign it a scalar integer model number, `num`, and to store it in a "meta data buffer," `dbufdef`. The meta data buffer contains a set of individual data buffers, each containing everything needed to define one specification. You can run `eimodels_def` multiple times to store as many models in `dbufdef` as needed (the first time you run the procedure, set `dbufdef="";`).

Once all model specifications have each been defined and stored, use `dbufrun = eimodels_run( dbufdef);` to run `ei` on each of the stored specifications and save the results in another meta data buffer, `dbufrun`. The results of each `ei` run can be read using `v = eiread(dbufrun);` and `eigraph(dbufrun);` while specifying the model number with the global variable, `_EIMetaR` (default of this variable is 1 which reads the results of `ei` run for Model 1).

Finally, run `dbufavg = eimodels_avg(dbufrun);` to use Bayesian Model Averaging to combine the results from multiple models with weights (the procedure calculates) indicating how much the data supports each model. The results of this model averaging can be read using `v = eiread(dbufavg);` and `eigraph(dbufavg);` where `dbufavg` is the output data buffer. To use this method, the prior distribution for covariates should be specified for each model using the global variables, `_EalphaB` and `_EalphaW`, so that the resulting posterior distribution is known to be proper.

# 6. Reference

## 6.1. EI

**Format:** `dbuf = ei(t,x,n,zb,zw);`, where `t`, `x`, and `n` are $p \times 1$ vectors, `zb` is a scalar 1 for no covariates or a $p \times k^b$ matrix of covariates, and `zw` is the scalar 1 for no covariates or a $p \times k^w$ matrix of covariates (do not include a constant term). `dbuf` is an output data buffer.

**Purpose:** `ei` gives observation-level estimates (and various related statistics) of $\beta_i^b$ and $\beta_i^w$ given variables $T_i$ and $X_i$ $(i = 1, \ldots, n)$ in this accounting identity: $T_i = \beta_i^b X_i + \beta_i^w (1 - X_i)$. Results are stored in `dbuf`, a data buffer that should be read with `eiread` or graphed with `eigraph`. See the model in Chapter 6, and extensions in Chapter 9.

**Example:** The following example uses the sample file `sample.asc` that comes with $\mathfrak{E}I$. This file includes hypothetical data from 75 precincts in rows and three variables, `t`, `x`, and `n`, in columns. You may use any other variable names or ASCII files and may include recodes or other GAUSS commands in this file.

```
new;                          @ clear workspace @
library ei;                   @ initialize libraries @
clear t,x,n;                  @ clear all variables in dataset @
```

```
loadvars sample.asc t x n;  @ load variables from disk file @
dbuf = ei(t,x,n,1,1);       @ run ei @
save racevote=dbuf;         @ save dbuf in file racevote.fmt @
```

The command `loadvars` is a keyword included with this program that makes it easy to read variables in from an ASCII data set (see page 30 below). If your data are in a GAUSS data file, use `subdatv` instead (see page 30 below). Because running `ei` takes several minutes, it is usually easiest to write one program such as this, and to examine the results interactively (or with a separate program) using `eiread` and `eigraph`.

**Globals:**

**_EalphaB** (cols(Zb)×2) matrix of means (in the first column) and standard deviations (in the second) of an independent normal prior distribution on elements of $\alpha^b$. If you specify `Zb`, you should probably specify a prior, at least with mean zero and some variance (default = {.}; which indicates no prior). (See Equation 9.2, page 170, to interpret $\alpha^b$). (If you are using $\mathfrak{E}zI$ and have trouble setting something to 0, try 0.000001 or some such; this gets around an error in a GAUSS proc and should give essentially the same answer empirically.)

**_EalphaW** (cols(Zw)×2) matrix of means (in the first column) and standard deviations (in the second) of an independent normal prior distribution on elements of $\alpha^w$. If you specify `Zw`, you should probably specify a prior, at least with mean zero and some variance (default = {.}; which indicates no prior). (See Equation 9.2, page 170, to interpret $\alpha^w$). (If you are using $\mathfrak{E}zI$ and have trouble setting something to 0, try 0.000001 or some such; this gets around an error in a GAUSS proc and should give essentially the same answer empirically.)

**_Ebeta** Standard deviation of the "flat normal" prior on $\breve{\mathfrak{B}}^b$ and $\breve{\mathfrak{B}}^w$. The flat normal prior is uniform within the unit square and dropping outside the square according to the normal distribution. Set to zero for no prior (default). Setting to positive values probabilistically keeps the estimated mode within the unit square. 0.25 is a reasonable value to experiment with at first.

**_Ebounds** 1 if set CML bounds on parameters automatically unless z's are included; 0 if don't use bounds; $k \times 2$ (where $k$ is the number of starting values) or $1 \times 2$ matrix to indicate upper∼lower bounds. (Do not confuse the bounds referred to here with the bounds on the quantities of interest.) Default = 1.

**_Ecdfbvn** Determines which procedure to use for computing the area of the bivariate normal distribution above the unit square: 1 based on the GAUSS function `CDFBVN`; 2 Martin van der Ende's method (based on D.R. Divgi, "Calculation of the univariate and bivariate Normal integral," *Annals of Statistics*, 1979, 903-910, with additional options available for this method in the proc `cdfbvn_div`); 3 Integration of log of the unit square; 4 Direct integration on unit square; 5, fairly accurate and fast, based on direct integration on the unit square from a new GAUSS internal procedure (DEFAULT); 6, most accurate but slow, based on a cdfbvn procedure by Alan Genz (using results from Drezner, Z. and G.O. Wesolowsky, 1989. "On the computation of the bivariate normal integral," *Journal of Statist. Comput. Simul.* 35: 101–107). See Appendix F.

Option 5 (the default) appears to be the best tradeoff between speed and accuracy currently available (and so this global should not be changed to anything other than 6, which is more accurate but much slower, unless you have a good reason to do so). However, fundamental progress remains to be made on methods of integrating the bivariate normal, as all currently available methods are innacurate and jump discontinuously and for very small values. Because of this, small values are truncated at the global `_EcdfTol`, which you may wish to adjust.

**_EcdfTol** Tolerance for the lncdfbvn function (when `_Ecdfbvn=5`, its default), with smaller calculated values truncated at the value of this global (DEFAULT=2.220446e-11). This can be any positive number, although lncdfbvn gets imprecise for small values. Only set to smaller values if you think you need the precision, such as if most of your values of $T_i$ or $X_i$ are very small.

**_Echeck** 1 check inputs and globals and give nice error messages if problems (default); 0 don't check, which saves some time. There is little reason to choose 0 unless you are running a large number of estimations and you are certain all the inputs are correctly specified. (Inessential global: not stored in dbuf.)

**_EdirTol** direction tolerance for CML convergence. Default = 0.0001. Set to smaller values if most of your values of $T_i$ or $X_i$ are very small.

**_EdoML** 1 do maximum likelihood (default); 0 don't do maximum likelihood, using instead the values of $\phi$ stored in `_EdoML_phi` and vcphi in `_EdoML_vcphi`.

**_EdoML_phi** if `_EdoML` = 1, this should include a vector of values of $\phi$ and will be used instead of the output of the likelihood maximization. (This global is ignored unless `_EdoML=1`.)

**_EdoML_vcphi** if `_EdoML` = 1, this should include a matrix of values of estimated variance matrix $V(\phi)$ and will be used instead of the output of the likelihood maximization procedure. (This global is ignored unless `_EdoML=1`.)

**_EdoSim** 1 do simulations (default); 0 don't do simulations; −1 don't do simulations or compute the maxlik variance (use this option for computing conditional log-likelihood of eta's).

**_Eeta** Automatically includes $X_i$ in the inputs `Zb` and/or `Zw`. The actual inputs `Zb` and `Zw` must be set to 1 if the default is changed. Using this global is better than explicitly including $X_i$ in the inputs, because eiread and eigraph will be "aware" of the contents of `Zb` and `Zw`. If you set this global, it is generally best to also set the priors `_EalphaB` and `_EalphaW`. See Chapter 9, and the parameterization in Equation 9.2 (page 170). Options include:

- `_Eeta=0` excludes $X_i$, which is equivalent to setting $\alpha^b = \alpha^w = 0$ (default).
- `_Eeta=1` sets `zb` = `x`, `zw` = 1, which estimates $\alpha^b$ and fixes $\alpha^w = 0$
- `_Eeta=2` sets `zb` = 1, `zw` = `x`, which estimates $\alpha^w$ and fixes $\alpha^b = 0$
- `_Eeta=3` sets `zb` = `zw` = `x`, which estimates $\alpha^b$ and $\alpha^w$.

- Set to a $4 \times 1$ vector with elements `_Eeta` $= \alpha^b|\alpha^w|\text{se}(\alpha^b)|\text{se}(\alpha^w)$ to fix $\alpha^b$ and $\alpha^w$, and their standard deviations, during estimation.
- Finally, set to a $3 \times 1$ vector $4|a|b$ to set `zb` = `x` and `zw` = 1, to estimate $\alpha^b$, and fix $\alpha^w$ =a and its standard error to b. Set to a $3 \times 1$ vector $5|a|b$ to set `zb` = 1 and `zw` = `x`, to estimate $\alpha^w$, and fix $\alpha^b$ =a and its standard error to b.

**_EI_vc** $M \times 2$ matrix ($M \geq 1$), each row of which represents instructions for one attempt to compute an estimated positive definite variance matrix of $\phi$. The procedure exits after the first positive definite hessian is found. Options to include in various rows are: {1 0} the usual numerical hessian computation (using GAUSS's hessp.src proc); {1 $d$} use usual hessian procedure and then adjust eigenvalues together so they are greater than $d$; {2 $f$} use wide step lengths at $f$ fraction falloff in the likelihood function; {3 $f$} use quadradic approximation with falloff in likelihood function set at $f$; {4 0} use a generalized inverse (to deal with singularity) and a generalized cholesky (to deal with non-positive definiteness) based on work in progress by Jeff Gill and Gary King; {5 0} use wide step lengths but check that the gradients for each are correct (and if necessary search for better ones); {-1 0} avoid the computation of the variance covariance matrix in case of non-positive definiteness and use the singular value decomposition for the multinomial normal sampling (i.e. _EisT has to be set to 0). In order to use this option, also make sure to define relatively narrow upper and lower bounds of the parameters by using _Ebounds. DEFAULT=`{1 0, 4 0, 2 0.1, 2 0.05, 3 0.1, 1 0.1, 1 0.2}`. The variance computation only very rarely gets beyond the second try.

When the likelihood surface is normal (i.e., quadratic), which is true asymptotically, all options produce identical results. In practice, this procedure is useful for ensuring that a positive definite variance matrix can be found due to numerical, rather than theoretical or empirical, difficulties, as can happen when the mode of the truncated normal is far from the unit square due to imprecision in the function that computes the bivariate normal CDF. (Another, sometimes better, way to fix these numerical problems is to reduce the variances of the priors in `_Erho` and `_Esigma`.) Because importance sampling is used after this procedure, different values of the variance matrix can produce identical estimates of the quantities of interest. Be sure to verify that the simulations are being appropriately drawn from the estimated contours (see compare the right two figures in eigraph's `tomogS`).

**_EIgraph_bvsmth** smoothing parameter for nonparametric estimation; used only if `_Enonpar` = 1. Default = 0.08. (The same parameter controls the nonparametric bivariate density estimation for diagnostic purposes in `eigraph`.) See Section 9.3.2.

**_EisChk** 0 to do nothing (default); 1 change `lnir` from the scalar mean importance ratio to a (`_Esims`*`_Eisn`)×(rows($\phi$)+1) matrix containing the log of the importance ratio as the first column and normal simulations of $\tilde{\phi}$ as the remaining columns. Also changes `PhiSims` from the mean and standard deviation of the posterior phi's to a `_Esims`×rows($\phi$) matrix of normal simulations of phi.

**_EiLlikS** 1 if save (_Esims×1) the log-likelihoods evaluated for each simulation; 0 saves only the means of these likelihoods (default). These can be used for computing the marginal likelihood.

**_EisFac** factor to multiply by estimated variance matrix in the normal approximation for use in importance sampling, or set to $-1$ to use normal approximation only or $-2$ to condition on the maximum posterior estimates. Adjust this, `_Eisn`, or `_Eist` if eiread's `resamp` larger than 15 or 20. If this is set too low, estimation variability will not be sufficient and your confidence intervals may be too narrow; it must be greater than zero and should probably be at least one. See Section 7.5. (Default = 4).

**_Eisn** factor to multiply by `_Esims` to compute the number of normals to draw before resampling. This is used to to try to get `_Esims` samples from exact posterior. Increase this or change `_EisFac` or `_EisT` if `resamp` is larger than 15 or 20. Default = 10. See Section 7.5.

**_EisT** 0 (default) to use multivariate normal density to draw random numbers for initial approximation for importance sampling; or if greater than 2, use the multivariate Student $t$ density, with degrees of freedom `_Eist`. Use this, `_EisFac`, or `_Eisn` if `resamp` is larger than 15 or 20. See Section 7.5.

**_EmaxIter** Maximum number of iterations for CML. Default = 500.

**_EnonEval** Number of nonparametric density evaluations for each tomography line (default = 11). Only used if `_EnonPar` = 1.

**_EnonNumInt** Number of points to evaluate for numerical integration in computing the denominator for the bivariate kernel density (default = 50). Only used if `_EnonPar` = 1.

**_EnonPar** 0 do not run nonparametric model (default); 1 run nonparametric model. (When choosing nonparametric estimation, only relevant options will be available under eigraph and eiread.) See Section 9.3.2.

**_EnumTol** Numerical tolerance. A homogeneous precinct is one for which $X_i <$`_EnumTol` or $X_i >$(1-`_EnumTol`). Default is 0.0001. Set to smaller values if most of your values of $T_i$ or $X_i$ are very small.

**_Eprt** 0 print nothing; 1 print only final output from each stage; 2 also prints friendly iteration numbers etc (default); 3 also prints all sorts of checks along the way. Use eiread and eigraph instead of this global to see output. (Inessential global: not stored in dbuf.)

**_Eres** If items are `vput` into `_Eres` before running `ei`, they are passed through into dbuf. For example, identifiers for each aggregate unit would be useful in interpreting the results, or using them in subsequent analyses (try: `_Eres=vput(_Eres,caseid,"caseid")` before calling EI. If a title is vput and given the name `titl`, the title is printed in convenient places. See `eiread` for further information. Do not use the name of any globals to this procedure or options listed under eiread(), or your variable will be lost.

**_Erho** The first element is the standard deviation of normal prior on $\phi_5$ for the correlation; set to 0 to fix $\phi_5$ to a second element, `_Erho[2]`; set to $-1$ to estimate without a prior. Default = 0.5. `_Erho` should be a scalar unless the first element is 0, in which case it should be a $2 \times 1$ vector, where the second element is the value at which the $\phi_5$ is fixed (and not estimated). See Section 7.4.

**_Eselect** Controls which observations are included in the estimation stage, including both likelihood maximization and importance sampling. All observations are included in the simulation stage unless you delete them from the data set before starting $\mathfrak{E}I$. This allows users to base the truncated bivariate normal contours on a subset of observations that might be more representative (such as those for which $T_i$ is not 0 or 1). Set to $p \times 1$ vector to of 1's to include and 0's to exclude individual observations.

**_EselRnd** Set to scalar 1 to include all observations not already deleted by `_Eselect` (default), or a scalar greater than 0 and less than 1 to randomly select this fraction of observations in the estimation stage. This global is especially useful for speeding up estimation in very large datasets, since thousands of observations are not always needed for estimating $\phi$. Since all observations will still be included in the simulation stage, precinct-level estimates of all quantities of interest will still be available. (If used with EI2, each iteration of $\mathfrak{E}I$ includes a different randomly selected set of observations.)

**_Esigma** Standard deviation of an underlying normal distribution, from which a half normal is constructed as a prior for both $\breve{\sigma}_b$ and $\breve{\sigma}_w$. Note: the expected value under this prior is `_Esigma`$\sqrt{2/\pi} \approx$`_Esigma`0.8. Set to zero or negative for no prior. Default = 0.5. See Section 7.4.

**_Esims** Number of simulations. Default is 100.

**_Estval** *For gradient methods:* Scalar 1, use best guess starting values (default); or set to $k \times 1$ vector of starting values. If `_Eeta[1]` $= 0$ (its default), $k = 5$ with elements guesses of $\phi$, that is on the scale of estimation. If you have starting values on the untruncated normal scale, $\breve{\psi} = \{\mathfrak{B}^b, \mathfrak{B}^w, \sigma_b, \sigma_w, \rho\}$, you can reparameterize as in this example: `_Estval=eireparinv(.5|.5|.2|.2|-.1)`. If `_Eeta[1]` $= 1, 2, 4, or 5$, $k = 6$; if `_Eeta` $= 3$, $k = 7$; and if covariates are used and rows(`_Eeta`) $= 4$, then $k$ is 5 plus the number of covariates included, with Zb coming before Zw.

*For a grid search:* Set `_Estval` to scalar 0 (with 5 divisions per zoom), or to a scalar integer greater than or equal to 3 for a grid search with this number of divisions per zoom. (That is, the grid search procedure divides the parameter space into a number of divisions, evaluates the likelihood for every combination of values on all the parameters, chooses the region of highest likelihood, zooms in and repeats the procedure on the narrower parameter region. This continues until differences in the parameters differ by the global `_Edirtol`.

**_EvTol** Numerical tolerance for the conditional variance calculation. Must be greater than 0; Default is $1e-322$.

**Output Global:** `_Eres`, a data buffer containing the time of completion, if procedure finishes. If the procedure is aborted, `_Eres` is a data buffer containing the data and all items computed up to that point.

### 6.2. EI2

**Format:** `dbuf2 = ei2(V,dbuf,zb,zw);`, where `v` is a $p \times 1$ vector, `dbuf` is a data buffer output from `ei()` or an earlier run of this procedure, `zb` is a scalar 1 for no covariates or a $p \times k^b$ matrix of covariates, and `zw` is the scalar 1 for no covariates or a $p \times k^w$ matrix of covariates. (Do not include a constant term.) `dbuf2` is the output data buffer.

**Purpose:** Gives observation-level estimates of parameters from $2 \times C$ tables with $C > 2$. First decompose the $C$ outcome categories into a series of dichotomous choices. For example, in the $2 \times 3$ running example in the book (see Table 2.2, page 31), Democrat/Republican/No Vote can be decomposed into Vote/No vote (which is analyzed as usual with `ei`) and Democrat/Republican, among those who vote (which `ei2` can analyze given $V_i$, the Democratic fraction of the vote, and the output from `ei`). If desired, covariates in either `ei` or `ei2` can be used to avoid possibly incorrect independence assumptions.

In this running example, this procedure will estimate $\lambda_i^b$ and $\lambda_i^w$ (and various related statistics) given $V_i$ (Democratic fraction of the vote) and given an *estimate* of $x_i$ (black fraction of *voters*) in this accounting identity: $V_i = \lambda_i^b x_i + \lambda_i^w (1 - x_i)$. A first-stage estimate of $\beta_i^b$ from `ei()` is used to estimate $x_i$, since $x_i = \beta_i^b X_i / T_i$ (these first stage results should be studied with eiread and eigraph before proceeding). Multiple imputation is used to incorporate the extra uncertainty due to $x_i$ being estimated. The details of this procedure are discussed in Section 8.4.

The separately imputed data buffers are stored in the output global `_ei2_mta`. The simulations from each data buffer are combined in a single data buffer, `dbuf2`. Both can be read with `eiread` or graphed with `eigraph`. Because these programs work for both `ei()` and `ei2()`, reading estimates requires using the names "betaB" and "betaW" for the parameters (even though they might be more appropriately called "lambdaB" and "lambdaW" for this particular example). In `dbuf2`, `x` are the mean posterior estimates of $x_i$, and `x2` is a ($p \times$ `_Esims`) matrix of simulations of $x_i$ (see eiread and eigraph for further information). Diagnostic outputs in this data buffer, such as `resamp`, are those from the final imputed data set.

**Example:** The following example references a file `sample2.asc`, included with $\mathfrak{E}I$, that includes data from 100 precincts in rows along with four variables, `v`, `t`, `x`, and `n`, in columns.

```
new;                              @ clear workspace @
library ei;                       @ initialize libraries @
clear v,t,x,n;                    @ clear all variables @
loadvars sample2.asc v t x n;     @ load variables from disk file @
dbuf = ei(t,x,n,1,1);             @ estimate betaB and betaW @
save racevote=dbuf;               @ save dbuf in file racevote.fmt @
dbuf2 = ei2(v,dbuf,1,1)           @ estimate lambdaB and lambdaW @
save raceDem=dbuf2;               @ save dbuf2 in file raceDem.fmt @
```

For example, given the $2 \times 3$ running example in the book (Table 2.2, Page 30), `eiread(dbuf, "betab")` are point estimates of $\beta_i^b$, the fraction of blacks who vote; `eiread(dbuf,"betaw")` are point estimates of $\beta_i^w$, the fraction of whites who vote; `eiread(dbuf2,"betab")` are

point estimates of $\lambda_i^b$, the fraction of blacks who vote Democratic, among those who vote; and `eiread(dbuf2,"betaw")` are point estimates of $\lambda_i^w$, the fraction of whites who vote Democratic, among those who vote.

**Globals:**

**`ei` globals** All the globals from `ei` apply here too.

**_EI2_m** Number of data sets to multiply impute (must be less than or equal to `_Esims`). If it is set to -1, `ei2` will use the posterior mean of $\beta^b$ to impute one data set. This option should be used only when `ei2` has the computational difficulty since the standard error is likely to be underestimated. The point estimate will not be biased. Default = 4.

**Output Global:** `_ei2_mta`, a "meta-data buffer" having as elements separate data buffers named dbuf1, dbuf2, dbuf3,. . . , one for each of `_ei2_m` imputations. Each is the output from a separate imputation run. If `ei2` fails, `_ei2_mta` contains results from each of the imputation runs already completed. `eiread` and `eigraph` understand when you give it a meta-data buffer as an argument; by default it uses `dbuf1` (this default can be changed by setting the `eiread` global `_EIMetaR` to the number of the imputation data buffer you wish to analyze.)

For example, to save the meta-data buffer to a file for safe keeping, and then examine the results from the first data buffer, add these lines to the example above:

```
save MraceDem=_ei2_mta;          @ save meta-data buffer @
eiread(_ei2_mta,"sum");          @ show summary statistics @
graphon;                         @ open graphics window @
eigraph(_ei2_mta,"estsims");     @ plot simulations @
graphoff;                        @ close graphics window @
```

## 6.3. EIGRAPH

**Format:** `eigraph(dbuf,"name");`, where `dbuf` is a data buffer that is the output of `ei` and `name` is a string chosen from the options listed below.

**Purpose:** Graphs results taken from the `ei` output data buffer (i.e., for $2 \times 2$ tables) and for `ei2` data buffers ($2 \times C$ tables). See `eiread` for extracting numerical information.

**Example:** This example assumes you have run `ei` and saved a data buffer (called `racevote.fmt` in the example below). `eigraph` can be run interactively or via a batch file. Here is a typical set of commands that you might run interatively on a DOS-based machine. (The calls to `graphclr;` prevent one graphic image being overlaid on top of the next.) GAUSS for Unix requires you to open graphics windows explicitly; see the reference item GRAPH on page 29 below.)

```
new;
library ei;              @ initialize ei @
loadm dbuf = racevote;   @ load in data buffer @
graphon;                 @ open graphics window @
eigraph(dbuf,"xgraph"); @ check info from bounds @
graphclr;                @ clear graphics screen @
eigraph(dbuf,"fit");     @ evaluate model fit @
graphclr;                @ clear graphics screen @
eigraph(dbuf,"tomog");   @ tomography plot @
graphclr;                @ clear graphics screen @
eigraph(dbuf,"post");    @ draw district-level posterior @
graphoff;                @ close graphics window @
```

**Options:** Choose any of the items below in place of `name` to see the corresponding graph. (Items that use true values from individual-level data will only work if you have `vput` this information into `_Eres` prior to running `ei` under the name `truth`. These are useful for verifying the method if individual-level data are available, but they are not useful for most ecological inference applications, where this information is not available. The options are included here only because I needed them while writing the book.)

**beta** betaB and betaW

**betaB** density estimate (i.e., a smooth version of a histogram) of point estimates of $\beta_i^b$'s, with whiskers.

**betaW** density estimate (i.e., a smooth version of a histogram) of point estimates of $\beta_i^w$'s, with whiskers.

**betabw** lines,Tlines,bivar,Tbivar. See Figure 10.8, page 214.

**bias** combination of biasB, biasW, TbiasB, TbiasW. See the points in Figure 13.2, page 238.

**biasB** $X_i$ by *estimated* $\beta_i^b$.

**biasW** $X_i$ by *estimated* $\beta_i^w$.

**bivar** *estimated* $\beta_i^b$ by $\beta_i^w$. See the right graph in Figure 13.6, page 243.

**boundX** combination of boundXB, boundXW, boundXB overlaid on TbiasB, boundXW overlaid on TbiasW. See Figure 13.2, page 238.

**boundXB** $X_i$ by the bounds on $\beta_i^b$ (each precinct appears as one vertical line), see the lines in the left graph in Figure 13.2, page 238.

**boundXW** $X_i$ by the bounds on $\beta_i^w$ (each precinct appears as one vertical line), see the lines in the right graph in Figure 13.2, page 238.

**estsims** All the simulated $\beta_i^b$'s by all the simulated $\beta_i^w$'s. The simulations should take roughly the shape of the mean posterior contours, except for those sampled from outlier tomography lines. Homogeneous precincts are excluded.

**fit** Combination of xtfit and tomogP

**fitT** Combination of xtfit and tomogT

**goodman** $X_i$ by $T_i$ plot with Goodman's regression line plotted. See Figure 4.1, page 60.

**lines** $X_i$ by $T_i$ plot with one *estimated* line per precinct. Homogenous precincts do not appear. See Figure 10.8, page 214.

**movie** For *each* observation ($i = 1, \ldots, p$), one page of graphics appears with the posterior distribution of $\beta_i^b$ and $\beta_i^w$ (with whiskers drawn at each simulated value), a plot of simulated values of $\beta_i^b$ by $\beta_i^w$ from the tomography line) and the numerical values of $T_i$, $X_i$, $N_i$, and $i$. After the graph for each observation appears, the user can choose to view the next observation (hit return), jump to a specific observation number (type in the number and hit return), or stop (hit "s" and return). See Figure 8.1, page 148.

**movieD** For *each* observation ($i = 1, \ldots, p$), one tomography plot appears (like tomogD) with the line for observation $i$ darkened. After the graph for each observation appears, the user can choose to view the next observation (hit return), jump to a specific observation number (type in the number and hit return), or stop (hit "s" and return). See Figure 5.1, page 81.

**nonpar** combination of tomogD and a nonparametric density estimation with contour plot and surface plot representations. See Figure 9.9, page 195.

**post** combination of postB and postW. See Figure 10.4, page 208.

**postB** density estimate of (weighted) district quantity of interest, the posterior distribution for $B^b$. See Figure 10.4, page 208.

**postW** density estimate of (weighted) district quantity of interest, the posterior distribution for $B^w$. See Figure 10.4, page 208.

**prectB** plot of estimated $\beta_i^b$ by true $\beta_i^b$. See Figure 10.5, page 210.

**prectW** plot of estimated $\beta_i^w$ by true $\beta_i^w$. See Figure 10.5, page 210

**profile** Profile plots — that is plots of the posterior of each element of $\phi$, holding constant all other values at their maxima. The maximum posterior point, as found by CML is displayed as a vertical line from the max down to the bottom of the graph. After seeing these plots, you can zoom in by changing `_eigraph_pro` (as is useful for seeing how curved the likelihood is near the maximum). (Kinks in the profile are due to imprecision in the cdfbvn function necessitating truncation by `_EcdfTol`.) This option is very useful for verifying whether the maximization procedure did indeed find the maximum.

**profileR** Profile plots of the cdfbvn function, (Equation 6.15, page 104). These are plots of R by each element of $\phi$, holding constant all other values at their maxima. The maximum posterior point, as found by CML is displayed as a vertical line from the max down to the bottom of the graph. After seeing these plots, you can zoom in by changing `_eigraph_pro`. Kinks in the profile are due to imprecision in the cdfbvn

function necessitating truncation by `_EcdfTol`. This option is very useful for verifying whether the posterior is numerical stable near the maximum.

**ptile** combination of ptileB and ptileW. See Figure 10.7, page 213.

**ptileB** true percentile at which $\beta_i^b$ falls by estimated $\beta_i^b$. See Figure 10.7, page 213.

**ptileW** true percentile at which $\beta_i^w$ falls by estimated $\beta_i^w$. See Figure 10.7, page 213.

**results** combination of postB, postW, betaB, betaW.

**sims** combination of simsB and simsW. See Figure 10.6, page 212.

**simsB** simulations of $\beta_i^b$ by true $\beta_i^b$. See Figure 10.6, page 212.

**simsW** simulations of $\beta_i^w$ by true $\beta_i^w$. See Figure 10.6, page 212.

**TbiasB** $X_i$ by *true* $\beta_i^b$. See the points in the left graph in Figure 13.2, page 238.

**TbiasW** $X_i$ by *true* $\beta_i^w$. See the lines in the right graph in Figure 13.2, page 238.

**Tbivar** *true* $\beta_i^b$ by $\beta_i^w$. See Figure 13.2, page 238.

**Tlines** $X_i$ by $T_i$ plot with one *true* line per precinct. Homogenous precincts do not appear. See Figure 3.1, page 41.

**tomog** tomography plot with ML contours. See Figure 10.2, page 204. Dashed lines are added for observations `_Eselect`'d out of the estimation stage, and a note appears at the bottom left and top right corners if any observations are included for which $T_i = 0$ or $T_i = 1$ (since as tomography lines they are represented as barely visible points). The global `_EselRnd` is ignored.

**tomogCI** tomography plot with 80% confidence intervals. Confidence intervals appear on the screen in red with the remainder of the tomography line in yellow. On printed output, which is often easier to see than the screen display, the confidence interval portion is printed thicker than the rest of the line. See Figure 9.5, page 179.

**tomogCI95** tomography plot with 95% confidence intervals. Confidence intervals appear on the screen in red with the remainder of the tomography line in yellow. On printed output, which is often easier to see than the screen display, the confidence interval portion is printed thicker than the rest of the line. See Figure 9.5, page 179.

**tomogD** tomography plot with data only. See Figure 5.1, page 81. Dashed lines are added for observations `_Eselect`'d out of the estimation stage, and a note appears at the bottom left and top right corners if any observations are included for which $T_i = 0$ or $T_i = 1$ (since as tomography lines they are represented as barely visible points). The global `_EselRnd` is ignored.

**tomogE** tomography plot with estimated mean posterior $\beta_i^b$ and $\beta_i^w$ points. Dashed lines are added for observations `_Eselect`'d out of the estimation stage, and a note appears at the bottom left and top right corners if any observations are included for which $T_i = 0$ or $T_i = 1$ (since as tomography lines they are represented as barely visible points). The global `_EselRnd` is ignored.

**tomogP** tomography plot with mean posterior contours. Dashed lines are added for observations `_Eselect`'d out of the estimation stage, and a note appears at the bottom left and top right corners if any observations are included for which $T_i = 0$ or $T_i = 1$ (since as tomography lines they are represented as barely visible points). The global `_EselRnd` is ignored.

**tomogS** combination of tomog, tomogp, tomogCI, and Tbivar (or estsims if truth isn't available). See Figures 10.2 (page 204), 9.5 (page 179), 7.1 (page 126). Dashed lines are added for observations `_Eselect`'d out of the estimation stage, and a note appears at the bottom left and top right corners if any observations are included for which $T_i = 0$ or $T_i = 1$ (since as tomography lines they are represented as barely visible points). The global `_EselRnd` is ignored.

**tomogT** tomography plot with true $\beta_i^b$ and $\beta_i^w$ points. See Figure 7.1 (page 126). Dashed lines are added for observations `_Eselect`'d out of the estimation stage, and a note appears at the bottom left and top right corners if any observations are included for which $T_i = 0$ or $T_i = 1$ (since as tomography lines they are represented as barely visible points). The global `_EselRnd` is ignored.

**truth** combination of post, precB, precW (compare truth to estimates at district and precinct-level). See Figures 10.4 (page 208) and 10.5 (page 210).

**xgraph** a scattercross with data plotted. See Figure 12.1, page 227.

**XgraphC** a scattercross with data plotted, with size proportional to $N_i$.

**xt** basic $X_i$ by $T_i$ scatterplot

**xtC** basic $X_i$ by $T_i$ scatterplot with circles sized proportional to $N_i$ or some other variable defined by the global `_eigraph_circ`.

**xtfit** $X_i$ by $T_i$ plot with estimated $\mathsf{E}(T_i|X_i)$ and conditional 80% confidence intervals. See Figure 10.3, page 206.

**xtfitg** xtfit with Goodman's regression line superimposed

**Globals:** These globals need not be changed for the most common uses.

**_eigraph_bb** string: xlabel for $\beta_i^b$ by $\beta_i^w$ plots. Default = `"betaB"`

**_eigraph_bbhi** high end for betaB plots. Default = 1.

**_eigraph_bblo** low end for betaB plots. Default = 0.

**_eigraph_bvsmth** smoothing parameter for nonparametric density estimation; used only for option `nonpar`. Default = 0.08. (The same parameter is used in `ei` for nonparametric estimation.

**_eigraph_bw** string: ylabel for $\beta_i^b$ by $\beta_i^w$ plots. Default = `"betaW"`

**_eigraph_bwhi** high end for betaW plots. Default = 1.

**_eigraph_bwlo** low end for betaW plots. Default = 0.

**_eigraph_dbuf** if 1, create data buffer called `_eigraph_dbuf` with px, py, pz inputs to the contour plot when option `nonpar` is chosen.

**_eigraph_eval** number of equally spaced evaluations on each axis of the nonparametric density estimation. Default = 31.

**_eigraph_loess** 1 if show simulated and fitted loess for xtfit; 0 if show fit only (default).

**_eigraph_pro** rows($\phi$)×2. The first column is the lower limit and the second is the upper limit of the range in which eigraph's `profile` command uses for plotting. The default, when set to missing, is to use eiread's `ebounds` option.

**_eigraph_smpl** Set this in the range (0,1] to randomly select this fraction of observations to use in tomography plots. This is useful if $p$ is so large that it is difficult to see patterns because of the large number of lines. Default = 1.

**_eigraph_T** string: ylabel for xt plots. Default = `"T"`

**_eigraph_Thi** High end of T graphs. Default = 1.

**_eigraph_Tlo** How end of T graphs. Default = 0.

**_eigraph_thick** value to add to line thickness parameter (default = 1).

**_eigraph_X** string: xlabel for xt plots. Default = `"X"`.

**_eigraph_Xhi** High end of X graphs. Default = 1.

**_eigraph_Xlo** How end of X graphs. Default = 0.

**_eigraphC** scalar: Multiply by circle size to change sizes. Default = 1, which means do not change. set to (0,1) to reduce circle size and to (1,∞] to increase circle size. Must be greater than zero.

**_EIMetaR** If `dbuf` is a meta-data buffer (i.e., from the `_ei2_mta` output global of ei2), this global denotes which of the imputed data buffers stored in dbuf should be accessed when running `eigraph`. Default = 1.

**_tomogClr** Colors for each contour drawn on tomography plots, default = { 12, 9, 10, 11, 13, 5 }.

**_tomogPct** Vector of percentage values at which to draw contours. One contour is drawn for each element with this fraction of this distribution's volume falling within the drawn ellipse. Default = {.5, .95}; rows(_tomogClr) must be ≥ rows(_tomogPct).

## 6.4. EIMODELS

**Format:** `dbufdef = eimodels_def(dbufdef,num,t,x,n,zb,zw);`, where `dbufdef` is an input meta data buffer which contains all the model specifications previously defined as separate $\mathfrak{E}I$ data buffers (or `""` for the first run), and to which the current model definition will be added, `num` is the scalar integer model number to be assigned to the current model definition, and the rest are defined as in `ei`: `t`, `x`, and `n` are $p \times 1$ vectors, `zb` is a scalar 1 for no covariates or a $p \times k^b$ matrix of covariates, and `zw` is the scalar 1 for no covariates or a $p \times k^w$ matrix of covariates (do not include a constant term). The output, `dbufdef`, is an updated meta data buffer which contains the current model definition along with the previously stored ones.

`dbufrun = eimodels_run(dbufdef);`, where an input, `dbufdef`, is the output meta data buffer from `eimodels_def` which contains multiple model definitions. The output, `dbufrun`, is another meta data buffer, containing one data buffer for each data buffer of model specifications in `dbufdef`.

`dbufavg = eimodels_avg(dbufrun);`, where `dbufrun` is the output meta data buffer from `eimodels_run` containing the results of `ei` run for multiple models. `dbufavg` is a regular output data buffer.

**Purpose:** `eimodels_def` stores multiple model definitions in a meta data buffer by adding one model at a time. Each model can use different covariates and/or different values of various global variables used for `ei` estimation. `eimodels_def` only stores model specifications; it does not run any models. Results are stored in `ndbuf2`, a meta data buffer which should be read with `eiread` with the model number specified as the value of the global variable, `_EIMetaR`.

`eimodels_run` takes the output meta data buffer from `eimodels_def` as an input and runs all the models stored in that meta data buffer. The output is another meta data buffer which contains the results of `ei` runs for each model as a single component data buffer. This output data buffer should be read with `eiread` and `eigraph` with the model number specified using the global variable, `_EIMetaR` (the default for which is 1, for the first model).

`eimodels_def` and `eimodels_run` are useful when you have different numbers of $\mathfrak{E}I$ models to run but want to store the results in one meta data buffer rather than saving them as many separate data buffers. The two procedures can also be used for the method called *Bayesian Model Averaging* as explained below.

`eimodels_avg` takes the output meta data buffer from `eimodels_run` as an input and implements Bayesian Model Averaging over all the models stored in that data buffer. To use this procedure, three global variables are required for each of the component $\mathfrak{E}I$ runs. First, when a model includes covariates, the prior distributions for $\alpha^a$ and $\alpha^b$ need to be specified for Bayesian Model Averaging in order to ensure that the posterior distribution is proper. The two global variables, `_Ealpha_B` and `_Ealpha_W`, perform this function. Second, $\mathfrak{E}I$ uses the Laplace approximation (default) or the harmonic mean estimator, for computing the marginal likelihoods. It is recommended that the number of simulations, `_Esims`, should be set to a relatively large number in order to improve the precision of this estimation. If you use the harmonic mean estimator, `_EiLikS` should be set to 1 for each model so that the output data buffer from `eimodels_run` stores the values of the log-likelihood at each simulation. These values are necessary to compute the marginal likelihood for this method. Finally, the output data buffer should be read with `eiread` and `eigraph`.

**Example:** The following example uses the sample file `sample.asc` that comes with $\mathfrak{E}I$. Consult section 6.1 for the description of data and explanation of the commands to load the data set. The code first stores the three different model definitions in a meta data buffer using `eimodels_def`, and then it runs `ei` for each of the three models. Finally, it implements the Bayesian Model Averaging to combine the results from the three models using `eimodels_avg`.

```
new;                                        @ clear workspace @
library ei;                                 @ initialize libraries @
clear t,x,n;                                @ clear all variables in dataset @
loadvars sample.asc t x n;                  @ load variables from disk file @

eiset;                                      @ clear for Model 1 @
_Eres = vput(_Eres, "Model 1", "titl");     @ print out model number @
_Eeta = 1;                                  @ zb=x, zw=1 @
_Ealpha_B = {0 2};                          @ prior for betaB @
_Esims = 1000;                              @ number of simulation @
dbufdef = eimodels_def("",1,t,x,n,1,1);     @ save Model 1, the first model @

eiset;                                      @ clear for Model 2 @
_Eres = vput(_Eres, "Model 2", "titl");
_Eeta = 2;                                  @ zb=1, zw=x @
_Ealpha_W = {0 2};                          @ prior for betaW @
_Esims = 1000;
dbufdef = eimodels_def(dbufdef,2,t,x,n,1,1); @ save Model 2 @

eiset;                                      @ clear for Model 3 @
_Eres = vput(_Eres, "Model 3", "titl");
_Eeta = 3;                                  @ zb=x, zw=x @
_Ealpha_B = {0 2};                          @ prior for betaB @
_Ealpha_W = {0 2};                          @ prior for betaW @
_Esims = 1000;
dbufdef = eimodels_def(dbufdef,3,t,x,n,1,1); @ save Model 3 @

save rvdef = dbufdef;                       @ save ndbuf in file rvdef.fmt @
dbufrun = eimodels_run(dbufdef);            @ run ei on all the models @
save rvrun = dbufrun;                       @ save ndbres in file rvrun.fmt @

_EIMetaR = 1;
call eiread(dbufrun, "sum");                @ summary of ei run for Model 1 @

_EIMetaR = 3;
graphon;
call eigraph(dbufrun, "tomog");             @ tomography plot for Model 3 @
graphoff;

_EI_bma_prior = {0.2,0.4,0.4};              @ set prior model probabilities @
dbufavg = eimodels_avg(dbufrun);            @ Bayesian Model Averaging @
```

```
save rvavg = dbufavg;                    @ save dbres in file rvavg.fmt @

graphon;
call eiread(dbufavg,"sum");              @ summary for dbres @
call eigraph(dbufavg,"post");            @ draw district level posterior @
graphoff;                                @ close graphics window @
```

**Globals:**

**_EImodels_save** set to "filename" if you want to save the results at each iteration in a data buffer on the disk (default = " "; for no save).

**_EImodels_est** scalar, if 1 the Laplace approximation will be used to estimate the marginal likelihood (default). If 2, the harmonic mean estimator (an importance sampling scheme) will be used.

**_EI_bma_prior** ((# of models)×1) The discrete prior probability of each model in the ascending order of the model number (default = 0 which assigns the uniform prior). The elements of this vector must sum up to 1.

## 6.5. EIREAD

**Format:** `v = eiread(dbuf,"name");`, where `dbuf` is the output from procedure `ei` or `ei2`, `name` is the item you desire to be extracted or computed from `dbuf` (see the list below), and `v` is the output vector, matrix, or string.

**Purpose:** To extract or calculate numerical information from the output data buffer of `ei` (for $2 \times 2$ tables) or `ei2` (for $2 \times C$ tables), from the ecological inference model.

**Example:** This example assumes that you have first run `ei` and saved a data buffer (called `racevote.fmt` in the example below). `eiread` can be run interactively or via a batch file. Here is a typical set of commands that you might run, although many other options to replace the items in quotes are available below.

```
new;
library ei;                        @ initialize ei @
loadm dbuf = racevote;             @ load in data buffer @
format/rd 7,4;                     @ nicely format output; 4 decimal places @
eiread(dbuf,"sum");                @ summary info @
betaw = eiread(dbuf,"betaw");      @ extract point estimates for betaW @
betabCI = eiread(dbuf,"ci80b");    @ extract betaB confidence intervals @
call eiread(dbuf,"goodman");       @ run Goodman's regression @
```

The last example illustrates the GAUSS "call" statement, which discards the output of eiread, leaving only the results that are explicitly printed to the screen by some eiread options. To save output from these procedures in an ascii text file, include commands like these:

```
output file=filename.txt reset;  @ start saving to filename.txt @
  betaw;                         @ print previously saved output @
  eiread(dbuf,"betab");          @ print betaB results @
output off;                      @ close output file @
```

**Globals:**

**_Eprt** if $\geq 1$, in addition to outputting results in `v`, print a display of the item to the screen (available for options whose description begins with an asterisk, below) (default); 0 do not print.

**_EIMetaR** If `dbuf` is a meta-data buffer (i.e., from the `_ei2_mta` output global of ei2; the output buffer from `eimodels_def` and `eimodels_run`), this global denotes which of the imputed data buffers (in case of `_ei2_mta`) or which of the stored model (in case of `eimodels_def` and `eimodels_run`) in dbuf should be accessed when running `eiread` and `eigraph`. Default = 1.

**Options:** Choose from the following items for `name`. The possibilities include items stored as is in `dbuf` (and so could also be retrieved with the GAUSS command `vread`), calculated from `dbuf`, special optional items that can be added ahead of time, and additional items that can be computed but require the added items. >From the perspective of the user, items in each of these categories can be treated identically, as all calculations are automatic. Items are not case sensitive. (Items that use true values from individual-level data will only work if you have `vput` this information into `_Eres` prior to running `ei` under the name `truth`. These are useful for verifying the method if individual-level data are available, but they are not useful for most ecological inference applications, where this information is not available. The options are included here only because I needed them while writing the book.)

When used with an EI2 output data buffer (i.e., for $2 \times C$ tables), `eiread` uses use the mean posterior estimate for $X_i$ for some items; the correct multiply imputed values of $x_i$ (`x2`) are used only where specifically noted.

**_EalphaB** value of this global (priors on $\alpha^b$)

**_EalphaW** value of this global (priors on $\alpha^w$)

**_Ebeta** value of this global (priors on $\breve{\mathfrak{B}}$)

**_Ebounds** value of this global (bounds for CML, not quantities of interest).

**_Ecdfbvn** value of this global (method of calculating CDF of the bivariate normal)

**_EdirTol** value of this global (CML convergence tolerance)

**_EdoML** value of this global (do maxlik)

**_EdoML_phi** value of this global (input $\phi$'s). Only relevant if `_EdoML` = 0.

**_EdoML_vcphi** value of this global (input variance-covariance of $\phi$'s). Only relevant if `_EdoML` = 0.

**_Eeta** value of this global (expanded model specifications; see Chapter 9).

**_EI_bma_prior** value of this global (prior model probabilities for Bayesian Model Averaging).

**_EI_vc** value of this global (variance matrix computation)

**_EImodels_save** value of this global (file name for `eimodels_run`).

**_EIgraph_bvsmth** value of this global (smoothing parameter for nonparametric density estimation)

**_EisChk** value of this global (check importance sampling)

**_EisFac** value of this global (number to multiply by variance matrix by in importance sampling or $-1$ for normal approximation)

**_EisN** value of this global (first stage importance sampling factor)

**_EisT** value of this global (multivariate $t$ or normal for importance sampling)

**_EmaxIter** value of this global (maximum iterations for CML)

**_EnonEval** value of this global (number of nonparametric density evaluations for each tomography line). Only relevant if `_Enonpar` $= 1$.

**_EnonNumInt** value of this global (number of points to evaluate for numerical integration in computing the denominator for the bivariate kernel density). Only relevant if `_Enonpar` $= 1$.

**_EnonPar** value of this global (0, parametric or 1, nonparametric, estimation).

**_EnumTol** value of this global (numerical tolerance for homogeneous and unanimous precincts)

**_Erho** value of this global (prior on $\rho$). See Section 7.4.

**_Eselect** value of this global (vector of 0/1 to delete/select observations during likelihood stage, or 1 to select all).

**_EselRnd** value of this global (fraction of observations to select randomly).

**_Esigma** value of this global (priors on $\breve{\sigma}_b$ and $\breve{\sigma}_w$). See Section 7.4.

**_Esims** value of this global (number of simulations). See Appendix F.

**_Estval** value of this global when `ei` was run (starting values).

**_n** $p \times 1$: the original variable $N_i$ from the first stage $\mathfrak{E}I$ analysis. This works only for EI2.

**_t** $p \times 1$: the original variable $T_i$ from the first stage $\mathfrak{E}I$ analysis. This works only for EI2.

**_EvTol** value of this global (numerical tolerance for the conditional variance).

**_x** $p \times 1$: the original variable $X_i$ from the first stage $\mathfrak{E}I$ analysis. This works only for EI2.

**_Ez** $2 \times 1$: number of covariates (see Chapter 9), including implied constant term for Zb|Zw; also clears and sets this in global memory.

**ABounds** $2 \times 2$: aggregate bounds rows:lower,upper; columns:betab,betaw. See Chapter 5.

**ABounds2** $2 \times 2$: aggregate bounds for the $\lambda$'s from EI2, rows:lower,upper; columns:lambdaB, lambdaW. See Chapter 5.

**AggBias** $4 \times 2$: regressions of true $\beta_i^b$ and $\beta_i^w$ on a constant and $X_i$. Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Table 11.1, page 219.

**aggs** `_Esims`$\times 2$: simulations of district-level quantities of interest, $\hat{B}^b \sim \hat{B}^w$. See Section 8.3. (Uses `x2` simulations when used with `ei2`).

**AggTruth** $2 \times 1$: true district-level $B^b$ and $B^w$. Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Chapter 2.

**beta** $p \times 2$ point estimates of $\beta_i^b$ (in the first column) and $\beta_i^w$ (in the second) based on their mean posterior. See Section 8.2. (Uses `x2` simulations when used with `ei2`.)

**betaB** $p \times 1$ point estimates of $\beta_i^b$ based on its mean posterior. See Section 8.2. (Uses `x2` simulations when used with `ei2`.)

**betaBs** $p \times$ `_Esims`: simulations of $\beta_i^b$. See Chapter 8. (Uses `x2` simulations when used with `ei2`).

**betaW** $p \times 1$ point estimate of $\beta_i^w$ based on its mean posterior. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**betaWs** $p \times$_Esims: simulations of $\beta_i^w$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**bounds** $p \times 4$: bounds on $\beta_i^b$ and $\beta_i^w$, lowerB$\sim$upperB$\sim$lowerW$\sim$upperW. See Chapter 5.

**bounds2** $p \times 4$: bounds on $\lambda_i^b$ and $\lambda_i^w$, lowerB$\sim$upperB$\sim$lowerW$\sim$upperW. See Chapter 5, eqns 5.4.

**checkR** rows($\phi$)$\times 2$ matrix with rows corresponding to $\phi$, columns corresponding to slightly less$\sim$more (by amount `_EdirTol`) than the MLEs, and each element indicating that the CDFBVN function is sufficiently precise (when 1) and insufficiently precise (when 0). This function calculates the $R$ portion of the likelihood function to make the comparison (Equation 6.15, p.104). See option R.

**CI50b** $p \times 2$: lower$\sim$upper 50% confidence intervals for $\beta_i^b$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**CI50w** $p \times 2$: lower$\sim$upper 50% confidence intervals for $\beta_i^w$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**CI80b** $p \times 2$: lower$\sim$upper 80% confidence intervals for $\beta_i^b$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**CI80bw** $p \times 4$: lowerB$\sim$upperB$\sim$lowerW$\sim$upperW 80% confidence intervals for $\beta_i^b$ and $\beta_i^w$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**CI80w** $p \times 2$: lower∼upper 80% confidence intervals for $\beta_i^w$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**CI95b** $p \times 2$: lower∼upper 95% confidence intervals for $\beta_i^b$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**CI95bw** $p \times 4$: lowerB∼upperB∼lowerW∼upperW 95% confidence intervals for $\beta_i^b$ and $\beta_i^w$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**CI95w** $p \times 2$: lower∼upper 95% confidence intervals for $\beta_i^w$. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**coverage** $2 \times 4$: confidence interval coverage; percent of true values within the 50% and 80% confidence intervals: 50b∼80b∼50w∼80w (1st row = means, 2nd = weighted means). Requires `truth` to have been vput into `_Eres` prior to running `ei`. (Uses `x2` simulations when used with `ei2`).

**CsbetaB** $p \times 1$ confidence interval-based standard error of $\beta_i^b$. (Uses `x2` simulations when used with `ei2`).

**CsbetaW** $p \times 1$ confidence interval-based standard error of $\beta_i^w$. (Uses `x2` simulations when used with `ei2`).

**DataSet** Zb∼Zw∼x∼t, used for input to `eiloglik`. If `_Eselect` is a scalar less than 1 (for random selection of cases). The global `_EselRnd` is ignored.

**date** a string containing the date and time at which execution completed, as well as the version number and date of the program $\mathfrak{E}I$ that created the input data buffer.

**double** $2 \times 1$ coefficients from a double regression. Requires an EI2 data buffer as input. See Section 4.3.

**EaggBias** $4 \times 2$: regressions of estimated $\beta_i^b$ and $\beta_i^w$ on a constant term and $X_i$. First row: coefficients, second row: standard errors. See Section 9.2.4.

**etaC** $2 \times 1$ coefficients implied by global _Eeta

**etaS** $2 \times 1$ standard errors implied by global _Eeta

**ExpVarCI** $100 \times 4$: 80% confidence intervals for $T_i$ given $X_i$. X∼20%CI∼mean∼80%CI of sims from $P(T_i|X)$, where $X$ in this context is 100 numbers equally spaced between 0 and 1. See Section 8.5. (Uses `x2` simulations when used with `ei2`).

**ExpVarCI0** $p \times 4$: 80% confidence intervals for $T_i$ given the observed values of $X_i$ and/or `Zb` and `Zw` when applicable. T∼20%CI∼mean∼80%CI of sims from $P(T_i|X, Zb, Zw)$ (Uses `x2` simulations when used with `ei2`).

**ExpVarCIs** $100 \times 4$: same as expvarci, but smoothed with LOESS. Used for eigraph's xtfit. See Section 8.5. (Uses `x2` simulations when used with `ei2`).

**GEbw** $p \times 3$: $\beta_i^b \sim \beta_i^w \sim$Nsims. Point estimates of $\beta_i^b$ and $\beta_i^w$, based on the mean posterior under the prior that $\beta_i^b \geq \beta_i^w$. Use this option (instead of `beta`) if you are reasonably certain that $\beta_i^b \geq \beta_i^w$. This procedure is based on simulations for which the inequality holds, and thus also reports `Nsims`, the number of simulations on which each estimate is based (If `Nsims` is not close to `_Esims`, you may wish to question your assumption or increase `_Esims`).

**GEbwa** $p \times 2$: $B^b \sim B^w$. Aggregate level mean posterior estimates, based on simulations where $\beta_i^b \geq \beta_i^w$. See `GEbw` for more information.

**GEwb** $p \times 3$: $\beta_i^b \sim \beta_i^w \sim$Nsims. Point estimates of $\beta_i^b$ and $\beta_i^w$, based on the mean posterior under the prior that $\beta_i^b \leq \beta_i^w$. Use this option (instead of `beta`) if you are reasonably certain that $\beta_i^b \leq \beta_i^w$. This procedure is based on simulations for which the inequality holds, and thus also reports `Nsims`, the number of simulations on which each estimate is based (If `Nsims` is not close to `_Esims`, you may wish to question your assumption or increase `_Esims`).

**GEwba** $p \times 2$: $B^b \sim B^w$. Aggregate level mean posterior estimates, based on simulations where $\beta_i^b \leq \beta_i^w$. See `GEwb` for more information.

**GhActual** value of the output global `_GhActual`, the row of `_ei_vc` for which a positive definite variance matrix was found.

**Goodman** $2 \times 2$: row 1: Goodman's Regression coefficients, row 2: standard errors. See Section 3.1.

**lnIR** If `_EisChk` = 1, this is a `_Esims`*`_Eisn`×rows($\phi$)+1 matrix , containing the log of the importance ratio as the first column and normal simulations of $\tilde{\phi}'$ as the remaining columns. If `_EisChk` = 0, this is the scalar mean importance ratio (equivalent to `meanIR` below).

**LogLik** value of log-likelihood at the maximum (unnormalized)

**LogLikS** value of log-likelihood at the maximum (unnormalized) for each observation $i$.

**LLikSims** If `_EiLlikS` = 1, this is `_Esims`×1 vector of log-likelihood values for each simulation; otherwise it is a scalar mean of these.

**Maggs** $2 \times 1$: point estimate of 2 district-level parameters, $\hat{B}^b$ and $\hat{B}^w$: meanc(aggs). See Section 8.3. (Uses `x2` simulations when used with `ei2`).

**MeanIR** scalar log of the mean importance ratio

**mpPsiu** Mean Posterior of $\breve{\psi}$ (rather than MLEs).

**N** $p \times 1$: number of individual elements in precinct $i$, $N_i$, an input to `ei`.

**Nb** $p \times 1$: denominator of `x` and `t`, equal to `x.*n`, $N_i^b$ (e.g., number of blacks in the voting age population).

**Nb2** for `ei2` data buffers only, $p \times$`_Esims`: denominator of `x2` and `V`, equal to `x2.*n`, $N_i^b$ (e.g., number of blacks in the voting age population).

**Nt** $p \times 1$: numerator of `t`, equal to `t.*n`, $N_i^t$ (e.g., number of people who turnout).

**NbN** $p \times 1$: $N_i^{bN}$ (e.g., number of blacks who don't vote). Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Chapter 2.

**NbT** $p \times 1$: $N_i^{bT}$ (e.g., number of blacks who vote). Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Chapter 2.

**Neighbor** Freedman et al.'s neighborhood model point estimates (i.e., assuming $\beta_i^b = \beta_i^w$, with implied standard errors of zero).

**nobs** scalar: number of observations, $p$.

**Nw** $p \times 1$: equal to `(1-x).*n`, $N_i^w$ (e.g., number of whites in the voting age population)

**Nw2** for `ei2` data buffers only, $p \times$`_Esims`: equal to `(1-x2).*n`, $N_i^w$ (e.g., number of whites in the voting age population)

**NwN** $p \times 1$ $N_i^{wN}$: (e.g., number of whites who don't vote). Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Chapter 2.

**NwT** $p \times 1$: $N_i^{wT}$ (e.g., number of whites who Turnout). Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Chapter 2.

**Paggs** $2 \times 2$: row 1: $\hat{B}^b$ and $\hat{B}^w$; row 2: standard errors. See Section 8.3. (Uses `x2` simulations when used with `ei2`).

**Palmquist** scalar: Palmquist's Inflation Factor. See Equation 3.14, page 52.

**ParNames** character vector of names for $\phi$ (`_cml_parnames`).

**phi** maximum posterior estimates from CML.

**PhiSims** If `_EisChk`==1, `PhiSims` is a `_Esims`$\times$rows($\phi$) matrix of random simulations of $\phi$; otherwise, it is a rows($\phi$)$\times$2 matrix of the means (in the first column) and standard deviations (in the second column) of the simulations (which are the mean and standard deviations of the posterior distribution of $\phi$). See Section 8.2.

**Pphi** $2 \times 5$ maximum posterior estimates. row 1: $\phi$, row 2: standard errors. See Chapter 7.

**psi** reparameterized $\phi$ into ultimate truncated scale. See Section 6.2.2.

**PsiTruth** $5 \times 1$: true values of $\psi$ (i.e., on truncated scale). Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Table 10.3, page 207.

**psiu** $\breve{\psi}$, which was reparameterized from $\phi$ into untruncated scale. See Equation 7.4, page 136.

**R** The sum of the log of the volume above the unit square under the bivariate normal, $R(\breve{\mathfrak{B}}, \breve{\Sigma})$. This is the last piece of the likelihood function. See checkR.

**Ri** The log of the volume above the unit square under the bivariate normal, $R(\breve{\mathfrak{B}}, \breve{\Sigma})$ for each $i$. This is the last piece of the likelihood function, and will differ over $i$ only if covariates are included. See checkR.

**resamp** number of resampling tries. This number will range between 1 and `_Esims` and is better if small. If it is greater than 15–20, you can try adjusting `_Eisn` or `_EisFac` and rerunning `ei`.

**RetCode** CML return code. zero means everything is ok.

**RNbetaBs** $p\times$`_Esims`: randomly horizontally permuted simulations of $\beta_i^b$. This is essentially equivalent to `betaBs` except that it randomly permutes estimation variation also. (Uses `x2` simulations when used with `ei2`).

**RNbetaWs** $p\times$`_Esims`: randomly horizontally permuted simulations of $\beta_i^w$. This is essentially equivalent to `betaWs` except that it randomly permutes estimation variation also. (Uses `x2` simulations when used with `ei2`).

**sbetaB** $p \times 1$ standard error for the estimate of $\beta_i^b$, based on the standard deviation of its posterior. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**sbetaW** $p \times 1$ standard error for the estimate of $\beta_i^w$, based on the standard deviation of its posterior. See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**STbetaBs** $p\times$_Esims: SORTED simulations of $\beta_i^b$ (e.g., the 80% confidence interval lower bound is `STbetaBs[int(0.1*_Esims)]`). See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**STbetaWs** $p\times$_Esims: SORTED simulations of $\beta_i^w$ (e.g., the 80% confidence interval upper bound is `STbetaWs[int(0.9*_Esims)]`). See Section 8.2. (Uses `x2` simulations when used with `ei2`).

**sum** prints a summary of district-level information

**t** $p \times 1$: outcome variable proportion, $T_i$ (e.g., turnout); input to `ei`. If dbuf is the output from ei2, then this option gives $V_i$ (e.g., Democratic fraction of the major party vote), an input to `ei2`.

**Thomsen** $2 \times 1$: Estimates of $B^b|B^w$ from Thomsen's Ecological Logit Model.

**titl** string: a title with descriptive information. Must have been vput into `_Eres` prior to running `ei`.

**truPtile** p x 2: percentile of sorted simulates at which the true value falls for $\beta_i^b$ and $\beta_i^w$. Requires `truth` to have been vput into `_Eres` prior to running `ei`. See Figure 10.7, page 213. (Uses `x2` simulations when used with `ei2`).

**truth** $p \times 2$: true values of the precinct-level quantities of interest $\beta_i^b \sim \beta_i^w$. Must have been vput into `_Eres` prior to running `ei`.

**truthB** `truth[.,1]`, the true $\beta_i^b$. Requires `truth` to have been vput into `_Eres` prior to running `ei`.

**truthW** `truth[.,2]`, the true $\beta_i^w$. Requires `truth` to have been vput into `_Eres` prior to running `ei`.

**tsims** 100×`_Esims`+1: simulations of $T_i$ given $X_i$. rows correspond to 100 values of $X_i$ equally spaced between zero and one, columns are $X_i$ and sorted simulations of $T_i|X_i$. See Section 8.5. (Uses `x2` simulations when used with `ei2`).

**tsims0** $p$×`_Esims`+1: simulations of $T_i$ given observed values of $X_i$ and/or `Zb` and `Zw`. The first column is the actual values of $T_i$ (Uses `x2` simulations when used with `ei2`).

**VCaggs** $2 \times 2$: variance matrix of 2 district-level parameters, $\hat{B}^b$ and $\hat{B}^w$. See Section 8.3. (Uses `x2` simulations when used with `ei2`).

**VCphi** global variance matrix of coefficients phi from gvc(). If `_EI_vc` is set to {-1 0 }, then it returns the inverse of the variance matrix.

**x** $p \times 1$: explanatory variable proportion, $X_i$ (e.g., black voting age population); input to `ei`. (for `ei2`, this is the mean posterior estimate of $x_i$, as, e.g., the black fraction of those voting).

**x2** for `ei2` data buffers only, $p \times$ `_Esims`: simulations of the explanatory variable proportion, $x_i$ (e.g., black fraction of voters).

**x2rn** for `ei2` data buffers only, $p \times$ `_Esims`: horizontally randomly permuted simulations of the explanatory variable proportion, $x_i$ (e.g., black fraction of voters). (This is useful because `x2` has only `_EI2_m` unique columns.)

**Zb** matrix of covariates for $\beta_i^b$ or 1 for none; as affected by _Eeta; input to `ei`. See Section 9.2.1.

**Zw** matrix of covariates for $\beta_i^w$ or 1 for none; as affected by _Eeta; input to `ei`. See Section 9.2.1.

## 6.6. EIREPL

**Format:** `dbuf=eirepl(dbufIN);` where `dbufIN` is a GAUSS data buffer output from `ei()`, containing t, x, n, Zb, Zw, and all globals in standard `ei` format, and where `dbuf` is an output data buffer.

**Purpose:** The purpose of this proc is to make it very easy to replicate results from an existing data buffer. It extracts and sets all globals from `dbufIN`, extracts the variables `t`, `x`, `n`, `Zb`, and `Zw`, and then runs `ei()`. The original results stored in `dbufIN` are not used. This procedure has no globals. Note that `ei` uses simulation to approximate various quantities and so you should not expect to replicate anything to the last decimal point. If you wish replication to more decimal points, increase `_Esims` before running `ei()` the first time or set GAUSS's `rndseed`.

**Example:** This example assumes that you have first run `ei` and saved a data buffer (called `racevote.fmt` in the example below), or you received `racevote.fmt` from someone else's analysis.

```
new;
library ei;              @ initialize ei @
loadm dbuf = racevote;   @ load in data buffer @
dbufNew = eirepl(dbuf);  @ replicate @
```

## 6.7. EISET

**Format:** `eiset;`

**Purpose:** To reset all $\mathfrak{E}I$ globals to their defaults.

## 6.8. GRAPHON, GRAPHONS, GRAPHCLR, GRAPHNO, GRAPHWAIT, GRAPHOFF

**Format:** `graphon;` to open a graphics window, `graphons;` to open a small graphics window, `graphclr;` to clear a graphics window already opened, `graphno;` to leave the graphics window open and switch output to the text window, `graphwait;` to pause between graphs, and `graphoff;` to wait for a key to be pressed in the text window and then to close the graphics window. (Use `wincloseall;` to close the graphics window without a pause.)

**Purpose:** Use these commands for easy interactive (or batch) window handling for windowing systems only (such as X-windows under Unix or win95). `graphclr;` should be used between graphs on a dos system so that one graph is not overlaid on top of the previous one.

**Example:** After running `ei` and saving the data buffer on disk in filename `racevote.fmt`, a typical interactive session might include the following:

```
new;
library ei;
loadm dbuf = racevote;          @ load in output from ei @
eiread(dbuf,"sum");             @ print summary info @
graphon;                        @ open graphics window @
eigraph(dbuf,"fit");            @ print graph @
graphclr;                       @ clear graphics window @
eigraph(dbuf,"bias");           @ print graph @
graphno;                        @ switch to text window @
eiread(dbuf,"Maggs");           @ print textual info @
graphclr;                       @ clear & switch to graphics @
eigraph(dbuf,"betabw");         @ print graph @
wincloseall;                    @ close graphics window @
```

## 6.9. LOADVARS

**Format:** `loadvars file var1 var2...`, where `file` is a string with a file name and is followed by a list of valid GAUSS variable names, one for each column of `file`.

**Purpose:** An ASCII data file is loaded from disk into the variable names listed. The data file must be delimited by spaces, tabs, or commas. It is often convenient to have one row per observation and one variable per column, but observations may wrap around to the next row if desired. If used in a batch file, be sure to clear the variable names listed prior to this procedure. This procedure is included for the convenience of users; if you are already familiar with other ways of loading data into GAUSS, they can also be used.

**Example:** After including these lines at the top of your program, the variables will be in memory and available for further use. `sample.asc` is a sample data file that comes with $\mathfrak{E}I$.

```
new;
library ei;
clear t,x,n;                @ clear variables in dataset @
loadvars sample.asc t x n;  @ load in vars into memory @
```

## 6.10. SUBDATV

**Format:** `call subdatv("dataset",vrs);` where `dataset` is a string with the filename of your GAUSS data set, and `vrs` is a character vector of variable names to read in or zero to read them all in.

**Purpose:** Reads in a GAUSS data set into variable names in memory by the same name. If used in a batch program, be sure to `clear` all variable names prior to issuing this command. This procedure is included for the convenience of users; if you are already familiar with other ways of loading data into GAUSS, they can also be used.

**Example:** After including these lines at the top of your program, the variables will be in memory and available for further use.

```
new;
library ei;
clear give,dog,a,bone;               @ clear variables in dataset @
call subdatv("C:\data\racedogs",0);  @ load all variables  @
```

# 7. Frequently asked questions

**How does $\mathfrak{E}I$ relate to Goodman's regression and the method of bounds?** The only commonly used methods before $\mathfrak{E}I$ were Goodman's regression and the method of bounds. Goodman's regression worked when the assumptions held but, as Leo Goodman made clear, it did not work when the assumptions were wrong. Within the Goodman framework, the data alone provided no information about whether the assumptions were right or wrong. The method of bounds always gave correct ranges into which the quantities of interest fell, but the ranges were often wider than was desirable (only in part because the wrong method of computing them was frequently used).

$\mathfrak{E}I$ combines the two methods (hence resolving most controversies between adherents of these two popular approaches) and adds some additional features. Instead of there being two situations, as under Goodman's approach (i.e., the assumptions applied and the method worked or they don't and it doesn't), we now have five, only the last one of which is a problem for EI:

1. Under EI, if the assumptions are correct, you get the right answer. For an example, see Chapter 10 or the Monte Carlos in Chapter 9.

2. If the assumptions are wrong, $\mathfrak{E}I$ still does "well" (in the sense of small MSE or squared bias) when the bounds (and other information in the tomography lines) are sufficiently informative. An important point is that the degree to which the bounds are informative can easily be assessed from the aggregate data, and so the risks of making ecological inferences are largely known. As one example, see Chapter 11.

3. If the assumptions are wrong and the bounds are not sufficiently informative, but the diagnostics are sufficiently informative, then the assumptions can easily be changed, and $\mathfrak{E}I$ will do well. The analyses reported in Figure 9.5 (p. 179) and the left graph in Figure 13.2 (p. 238) for aggregation bias and Figures 9.7 and 9.9 (Pp. 187, 195) for distributional violations are examples. The third assumption, no spatial autocorrelation, seems to have minor effects.

4. If the assumptions are wrong and the bounds and the diagnostics are not sufficiently informative, but the researcher has additional qualitative knowledge of the problem, then appropriate assumptions can be chosen. In this case, either $\mathfrak{E}I$ will do well, or the formal measures of uncertainty produced by $\mathfrak{E}I$ (standard errors and confidence intervals, etc., which are based only upon the data and model) can be supplemented and expanded accordingly. Since the ecological inference problem is about information that has been aggregated away, only by adding some information is it possible to make reliable inferences in general. Qualitative information is of course subject to more interpretation and hence more uncertainty, but reliable inferences permit no other option other than to add assumptions or other information. The book discusses a lot of ways to bring in qualitative information (see also Gary King, Robert Keohane, and Sidney Verba. 1994. *Designing Social Inquiry: Scientific Inference in Qualitative Data.* Princeton University Press).

5. If the assumptions are wrong and the bounds and the diagnostics are not sufficiently informative, and the researcher has no time or resources to collect additional qualitative

information, then EI will perform poorly. An example of data like this appear in Figure 9.2 (p. 163). Even in this worst case scenario, and the others, EI will be more robust than Goodman's. By this I mean that the maximum amount of bias from $\mathfrak{E}I$ is capped at a fixed and knowable level, in contrast to Goodman's approach. The dotted line (corresponding to $\tau = 0$ for the default model) in Figure 9.6 (p. 180) shows that bias in $\mathfrak{E}I$ estimates increases with the degree of aggregation bias for small levels of aggregation bias; at some point, however, the maximum bias maxes out and increases no further. The point at which the error maxes out depends on the data. Under Goodman's approach, the error linearly increases without limit as aggregation bias increases.

The likelihood of the first four cases coming up relative to the fifth (as compared to the likelihood of the assumptions applying vs not applying under Goodman's) summarizes the advantage of EI. Basically what $\mathfrak{E}I$ does is to chip off pieces of Goodman's worst case (the assumptions not applying). The benefits of $\mathfrak{E}I$ will therefore quite obviously depend on the area and application and how much effort is put into collecting qualitative information.

**In what precise statistical sense is $\mathfrak{E}I$ "Robust"?** As aggregation bias increases, Goodman's regression becomes biased without limit. The bias in EI, in contrast, has a maximum value that is a function of the bounds. This is easy to see in Figure 9.6, p.180; note how the mean absolute error for the dotted line at the top, for example, is linear in the low aggregation bias region (where $\alpha < 0.5$), but for higher levels of aggregation bias it levels off; in contrast, the mean absolute error for Goodman's and any method that does not incorporate the bounds will increase linearly without limit until the correlation of $X$ and $\beta_i^b, \beta_i^w$ is 1.

**How does $\mathfrak{E}I$ relate to the neighborhood model?** The neighborhood model was designed for the purpose of critiquing the Goodman model, and is not thought of as a plausible method of producing ecological inferences in its own right (see pp. 43–44). The neighborhood model's advantage is that it is in the class of models that are consistent with the method of bounds for each precinct and thus cannot be rejected solely from information in the aggregate data (this class of models is described on p. 191). As such, it can be seen as a special case of EI. Its disadvantage is its assumption that $\beta_i^b = \beta_i^w$, which is of course appropriate in some cases and far off in others, but whichever it is it assumes the answer to the question being asked. A consequence of the neighborhood model assumption having no error term and supposedly holding exactly is that its standard errors are always zero, which is unreasonable. If, somewhat more reasonably, the neighborhood model assumption is *approximately* plausible for a particular application, then it is best to run $\mathfrak{E}I$ with priors suitably adjusted to reflect this information. If priors are strong enough and not substantially contradicted by the likelihood, $\mathfrak{E}I$ will give similar point estimates to the neighborhood model, but it will give more reasonable (nonzero) standard errors and confidence intervals for the precinct-level quantities.

**Can $\mathfrak{E}I$ give the misleading answers?** Yes. Nothing in $\mathfrak{E}I$, this manual, or *A Solution to the Ecological Inference Problem* promises to give you the correct answer every time without thought. "The method" proposed in the book is *not* what comes spinning out of $\mathfrak{E}I$ with all globals set at their defaults. Appropriate inferences, according to the argument put forward in the book, require full use of the diagnostics, to evaluate the amount of information lost in

aggregation (such as how wide the bounds are for different groups of observations) and how well the model fits and its assumptions apply. Since assumptions about joint distributions for $\beta_i^b$ and $\beta_i^w$ cannot be rejected if they merely have positive mass over *any* curve that connects the bottom left and top right points of a tomography plot (p.191), there is no way to make certain inferences about individual level behavior from aggregate data alone. The only solution to this fundamental lack of information is to bring in some of the vast array of qualitative information available to most social scientists about the problems we study — including ethnographies, particiant observations, partial survey data, journalistic accounts, historical studies, prior quantitative research, and the like — the full range of data collection schemes used in modern social science. Interpreting qualitative information in the context of statistical inference is of course open to more interpretation and ambiguity than formal statistical tests, but stopping at quantitative data, especially for this problem, is insufficient.

Considerable thought, analysis, and qualitative information may be necessary to settle on the right version of the model to run. $\mathfrak{E}I$ includes dozens of global variables that govern the main parts of the model; combinations of these globals can produce estimates from millions of possible specifications, even given identical input variables. The choice among these models requires the same degree of reasoned analysis and reanalysis, checking assumptions, and rerunning that the appropriate use of any method does. The actual method of ecological inference proposed in the book requires careful attention to each item in the checklist provided in the concluding chapter (Chapter 16); since several of the items require the user to consult qualitative evidence and other substantive knowledge about the problem, this program alone implements only part of the proposed method. Moreover, even with considerable thought, some misinformation or lack of information can sometimes lead to incorrect estimates; Chapter 9 provides extensive examples of precisely what can go wrong and under what conditions. If you have an example where you suspect that $\mathfrak{E}I$ does not recover the truth, then one of the problems discussed in that chapter is likely at fault, and so you might consider some of the alternative approaches and model extensions also given there.

Finally, if you are comparing $\mathfrak{E}I$ results to an external source of information to judge the "truth", consider whether the external source may be biased. For example, an estimate from survey data is just an estimate and not necessarily be better than an ecological inference. One of the best academic surveys, the National Election Studies, overestimates turnout by 8-10% and vote for incumbent House candidates by about 8%. (Even the NES's "voter validation studies," which check each respondent's turnout from public records, contain errors.) Other surveys, especially about controversial issues, or politically or personally sensitive topics, often generate larger biases. The point is that every source of information, ecological and individual, comes with some potential biases or errors.

See also the questions below on the advantages of EI, computational problems, statistical fit, and standard error interpretation.

**How do I understand $\mathfrak{E}I$ standard errors?** $\mathfrak{E}I$ standard errors (and other uncertainty estimates, such as confidence intervals, etc.) are logically very similar to, and can be interpreted analogously to, standard errors for least squares regression (LS) coefficients:

1. In LS, the variance of predicted values as estimates of $y_i$, $V(\hat{y}_i + \epsilon) = V(x_i b) + \sigma^2$ do not go to zero as $n$ gets larger. In $\mathfrak{E}I$, the variances of the precinct-level parameters ($V(\beta_i^b)$ and $V(\beta_i^w)$) do not go to zero as $n$ gets larger.

2. In LS, $V(\hat{y})$, i.e. the "sample" variance of $\hat{y}_i$ over $i$, does not go to zero as $n$ gets larger. In $\mathfrak{E}I$, $V(\beta^b)$ and $V(\beta^w)$, i.e. the "sample" variance of $\beta_i^b$ and $\beta_i^w$ over $i$, does not go to zero as $n$ gets larger.

3. In LS, $V(b)$ goes to zero as $n$ gets larger (where $b$ is essentially any scalar function of $\hat{y}_i$, for all $i$, such as a LS coefficient). In $\mathfrak{E}I$, $V(B)$ goes to zero as $n$ gets larger (where B is the weighted average of the $\beta_i$'s over all precincts).

4. In LS, the number of explanatory variables we include tends in common practice to be an increasing function of the number of observations we have, and so we don't see very small standard errors unless there's a mistake. In $\mathfrak{E}I$, covariates are only sometimes used to correct for some types of aggregation bias and the number included is, in practice, independent of the number of observations and the number of quantities that may be of interest.

5. In LS, the variance of a prediction is a function of estimation uncertainty (sampling error, $V(b)$, where $b$ is the regression coefficient) and fundamental uncertainty ($V(\epsilon_i)$). In $\mathfrak{E}I$, $V(\beta_i^b)$ and $V(\beta_i^w)$ are functions of estimation uncertainty ($V(\psi)$, where psi are the 5 parameters of the truncated bivariate normal), and fundamental uncertainty ($\Sigma$, which is composed of three elements of $\psi$).

6. In LS, it is standard practice is to include in the formal computation of the standard error only estimation and fundamental variability and to exclude uncertainty due to the possibility of omitted variable bias, endogeneity, measurement error, selection bias, etc. It is possible to include these other possible problems in the computation of the standard error, but few computer programs allow it. In $\mathfrak{E}I$, the standard errors include only estimation and fundamental uncertainty, and exclude possible violations of the 3 model assumptions that the model in an application may not be robust to (i.e., serious violations of no aggregation bias, truncated bivariate normality, and spatial independence). Only by using the diagnostics and bringing in additional qualitative information, not represented in the aggregate data, can one add assessments of uncertainty to the formal measures given by the program. Of course, many of the standard problems of regression do not affect ecological inferences.

**How do I get a quick list of options and globals?** Use the GAUSS help facility. As long as the library command (`library ei;`) has been issued, you can ask for help on any of the $\mathfrak{E}I$ commands. For example, in DOS, you can type Alt-H,H,eiread to get all of eiread's options.

**What computational details should I check?** Make sure that the maximization procedure worked properly. For example, make sure that eiread's `retcode` is zero. Is the variance matrix from the normal approximation reasonable? Most importantly, check to make sure the simulations are coming from the mean posterior contours by comparing eigraph's `tomogp` to `estsims` (these are both in `tomogS`); verify that eigraph's `post` gives unimodal distributions. Was the importance sampling successful? Check to make sure that eiread's `resamp` is not much larger than 20 or so (it is not fatal if it is larger, but its worth checking).

**What statistical issues should I check?** It is essential to verify that the model fits the data. First, look at eigraph's `fit`: for the X by T graph on the left, verify that the $E(T_i|X_i)$ line passes through the middle of the points, and the 80% confidence intervals capture around 80% of the points, vertically for each value of $X$ on the horizontal axis. For the tomography plot of the right of the `fit` graph, verify that the contours capture the place from where the "emissions" come from (roughly speaking, where the lines are crossing). Then check eigraph's `tomogS` and verify that the maximum likelihood contours (on the top left, the same as the tomography plot in the `fit` graph) and mean posterior contours (on the top right) both fit the data in roughly the same way. If there are problems here, see the next question in this FAQ.

Other issues to check are whether there outliers or multiple modes. Is there aggregation bias? Check the results at the aggregate level (eigraph's `post`; eiread's `Paggs`) and the precinct level (e.g., eigraph's `beta` or eiread's `betaB` and `betaW`); are these consistent with your qualitative knowledge? Verify that the relationship between $\beta_i^b$ and $X_i$ and between $\beta_i^w$ and $X_i$ are consistent with your understanding of your substantive problem (see eigraph's `boundX`). Is there survey, qualitative, or other external information you could have used but didn't? Chapter 16 provides a complete checklist that should be used for every serious application.

**What global values should I set? Which $\mathfrak{EI}$ model should I run?** $\mathfrak{EI}$ will enable you to run any of a large set of related models. No one model from this set or any other will work all the time. The program defaults are chosen for speed and simplicity, not for what will be most reasonable empirically. For the latter, I would read Section 9.2.3 (especially Figure 9.6) and consider globals like these:

```
_Eeta=3;
_EalphaB=0~0.25;
_EalphaW=0~0.25;
```

**What do I do if my model doesn't fit the data?** There are several approaches, depending on the problem. Don't miss the first item.

1. The most important technical problem with lack of fit occurs because of numerical issues related to imprecision in the lnCDFbvn function used in the likelihood function (the volume above the unit square under the bivariate normal, Equation 7.2, p.134). This imprecision can induce artificial local maxima in the likelihood function, leading to convergence at the wrong parameter values. It can also create artificial maxima higher than the correct global maximum. These problems occur most often when the maximization routine is looking far from optimal value. A good way to fix both problems is to give the program starting values in the region of the right answer (set `_Estval`), and constrain the search to a region that includes these values (set `_Ebounds`). One way to find better starting values, it is to pick the parameter values by looking at a tomography plot, as if we were on the truncated scale. That is, identify the region from where the "emissions" are coming (roughly, where the tomography lines are crossing) and record the coordinates for betaB and betaW, the width of the emissions at the central point, and then a likely correlation (or 0). Then transform these onto the scale of estimation with Equations 7.4 (p.136), or use `eireparinv` to do this automatically.

Then set `_Ebounds` to regions around those starting values — not too narrow because you determine the answer (a concern if the parameters turn out to be maximized at boundary values), and not too wide because you may run into numerical problems. If this does not work, it will be helpful to narrow `_Ebounds`. The new grid search procedure is especially helpful here (set `_Estval` to 0 or $> 5$). If it is an especially difficult problem, you may need to change the tolerance of the lncdfbvn function, with `_EcdfTol`.

2. If you have very small values of $T$, see the FAQ question below.

3. One common problem is coding errors, or small precincts, for which $T_i$ is very close to 0 or 1 (look at the corners of eigraph's `tomog` for a count of these); if these values are outliers, they can have a disproportionate effect on the likelihood results, despite the fact that in many applications $T$ only gets to the corners when there are data errors. To delete them from the estimation stage but include them in the simulation stage, you could set

   ```
   _Eselect=(t.>0.001).and(t.<0.999);
   ```

   or perhaps an even narrower range would be wise. You could also delete them from the data set to skip both stages.

4. Do you suspect extensive aggregation bias? Perhaps you should try the globals `_Eeta=3;` `_EalphaB=0~0.1;` and `_EalphaW=0~0.1;` to start (see Chapter 9).

5. Does eigraph's `tomog` suggest multiple modes? Consider specifying `Zb` or `Zw` coded to pick up the modes.

6. Is eiread's `resamp` much larger than 20? If so, you might try using a $t$ distribution as the first approxiation for importance sampling by setting `_EisT` to 3 or higher, or adjusting `_EisFac` (usually downwards or set to $-1$, especially if `tomog` fits but `tomogP` does not) or `_Eisn` (always upwards) (e.g., you might try `_EisFac=1` or `_EisFac=`$-1$).

7. If eigraph's `estsims` does not look approximately like `tomogp`, or if the graphs in `post` are bimodal, you need to do something. You may try a different method of computing the variance matrix. You could also narrow the variance of the priors on $\sigma_b$, $\sigma_w$, and $\rho$ by setting `_Esigma` and `_Erho`. Or more simply, you could use the maximum likelihood solution and set `_EisFac=-2;`.

8. If the relationship between $X_i$ and $\beta_i^b$ or $\beta_i^w$ does not correspond to your substantive knowledge of the problem, consider setting `_Eeta=3` and adding a prior on $\alpha^b$ and $\alpha^w$ (with `_EalphaB` and `_EalphaW`).

9. If you have additional information in the form of survey or qualitative evidence, you could change the priors, add covariates in `Zb` or `Zw`, or divide the data set.

See Chapters 9 and 16 for more detailed suggestions.

**What do I do if the Hessian is not positive definite?** If you have covariates, make sure you didn't include a constant term or variables that are very highly collinear. Whether you included covariates or not, what often helps is to narrow the variance of the prior on $\sigma_b$, $\sigma_w$, and $\rho$ by setting `_Esigma` and `_Erho` to smaller numbers than their defaults. Following this action is also sometimes helpful in getting the model to fit appropriately, if $\mathfrak{E}I$ took several tries to compute a valid variance matrix. You should be careful, of course, that a non-positive definite Hessian doesn't indicate an impossible estimation problem. Also be careful whenever `eiread`'s `ghactual` is other than {0 1} (if it is, see the global `_EI_vc`).

**What do I need to know about studying voter transition rates?** The model applies as usual, but note how the quantities of interest are parameterized. In the running example, it is the fraction of blacks who vote, not the fraction of people who are both black and who vote. If the latter parameterization seems more natural for voter transition applications (e.g., the fraction of people who vote Democratic at time 1 and time 2), you need to translate the output of $\mathfrak{E}I$.

**Why do I get slightly different results every time?** $\mathfrak{E}I$ uses random simulation to approximate various statistics. This entails no compromise, and it makes computations easy that would otherwise be impossible. The only issue is ensuring that you have sufficient random draws for the level of precision (number of significant digits) you desire. For example, if you need 2 digits to the right of the decimal point for a table you plan to present, you can check that the number of simulations is set high enough by running the analysis twice and verifying that the first two digits do not change. If they do change, increase `_Esims` and try again until the two runs give the same answer to two digits. (If you really want exactly the same numbers for each replication, set the GAUSS random number seed with the GAUSS command `rndseed` before running $\mathfrak{E}I$, although this would only be useful for testing.) See Chapter 8.

**How do I get this program to run faster?** A variety of $\mathfrak{E}I$ options can significantly increase the speed of the program. In the order in which we suggest you try them, these are,

1. If you have many observations, you can use a random subset in the first stage (by setting `_EselRnd` to the fraction of observations to include). Setting this global still results in all observations still being included in the second stage, the result being that estimates will still be available for all observations. If a large $n$ is the issue, then this is the best way to speed estimation without much cost.

2. You can use the asymptotic normal approximation, and eliminate the importance sampling refinement, by setting `_EisFac`$= -1$ or to go even faster use `_EisFac`$= -2$ to exclude estimation uncertainty.

3. Choose good starting values (by setting `_Estval`). For example, if you have many similar analyses to run, do a typical analysis and use those numbers as starting values for the others. If you have a guess as to the values of $\phi$ on the untruncated scale, you can use the following command (documented only in `eirepar.src`) to set the starting values: e.g., `_Estval=eireparinv(0|0|-1|.4|.1)`.

4. Buying enough RAM so you do not need to take advantage of the virtual memory feature of GAUSS is helpful. Or, just buy a really fast computer!

5. If you are willing to live with lower precision, you can draw fewer simulations by setting `_Esims` to a number smaller than the default (100).

6. You can set `_EdirTol` to a larger number. The default is 0.0001; you can try 0.001 or larger. You could also reduce the maximum number of iterations with `_EmaxIter`.

7. If you are running many analyses, run one analysis to make sure that the globals are properly defined and then set `_Echeck=0` to turn off global checking. This will only save a small bit of time and so is only helpful if you are running many analyses. (If you turn off checking, and you define a global incorrectly, $\mathfrak{E}I$ will not give a pretty error message.)

8. For EI2, set `_ei2_m`$= -1$.

**Why is the program slower when using covariates?** $\mathfrak{E}I$ needs to evaluate the cumulative bivariate normal distribution, which is computationally intensive. If no covariates are used, only one set of computations need to be done for the entire data set. If any covariates are used, a new computation must be computed for every observation.

**What are the parameter names printed during maximization?** Without covariates, `Zb0` and `Zw0` are estimates of $\phi_1$ and $\phi_2$ respectively. With covariates, `Zb1`, `Zb2`,..., refer to estimates of the elements of the vector $\alpha^b$ and `Zw1`, `Zw2`,..., are estimates of the elements of $\alpha^w$. The symbols `sigB`, `sigW`, and `rho` refer to estimates of $\phi_3$, $\phi_4$, and $\phi_5$. Finally, the global `_Eeta` allows $\alpha^b$ and $\alpha^w$ to be set to fixed values, which are `etaB` and `etaW` (the default for is zero for each).

**What should I look for during the iterations?** The iterations will converge when direction vector (which, roughly speaking, is the number of digits of precision in locating the maximum) gets close to all zeros. (If you prefer knowing when it will end, choose the grid search procedure by setting `_Estval=0`).

**What about data with very small or large values of $T$?** Data with many small or large $T_i$ cause no problems with the method in theory. Moreover, data like these produce estimates with very narrow bounds (since the tomography lines cut off the top right or bottom left corner of the plot). This is good news for estimation, although if the difference between, say, $\beta_i^b = 0.01$ and $\beta_i^b = 0.005$ is substantively large (as for mortality rates), what might otherwise be considered "narrow" bounds (such as [0,0.05]) could in some cases still be too wide for particular substantive purposes, so be careful about interpretation. To read the graphs, it is helpful to zoom in on the relevant regions, by setting globals such as `_eigraph_Xlo`, `_eigraph_BBhi`, etc. Be sure to check and probably reduce the global `_EnumTol`, since the default value (0.0001) would define some data sets as unanimous ($T_i = 0, 1$) for too many observations. Beware also of other numerical problems with data like these (by verifying the fit of the contours to the data with eigraph's `tomogS` and `fit`), since the contours must be fit to a very small area of the tomography plot and so calculating the cdf of the truncated

bivariate normal can be imprecise. It is often necessary to use better starting values (see `_Estval`) and search constraints (`_Ebounds`) than the defaults. The grid search procedure can also be helpful (`_Estval`). Since odds are you want to distinguish between very small differences in the quantities of interest, you will probably also wish to increase the number of simulations substantially (`_Esims`). Finally, posterior densities from $\mathfrak{C}I$ are rarely normal or symmetric in the presence of rare events, and in this situation mean posterior point estimates are not good summaries of EI's inference in these situations. Better would be to use a density estimate of the simulations, such as eigraph's `post` option.

**How do I make inferences in larger tables?** For $2 \times C$ tables, see the `ei2` in the reference section of this manual (see Section 8.4). For general $R \times C$ tables, you can partition the data into a series of $2 \times 2$ (or $2 \times C$) tables, each of which this program can analyze. The results can then be combined to produce estimates of your larger table. See Section 15.1 for details and the rest of Chapter 15 for more general procedures. (We're working on an easier approach for $R \times C$ tables.)

**What do I do if I have computational problems with EI2?** Try `_Ei2_m=-1`. This may make the standard errors slightly too small, but it will make the procedure numerically more reliable and about four times faster. This is the procedure described at the start of Section 8.4.1 (pp 151–2).

**Why does eiread report some precinct-level estimates as missing values?** When $X_i = 0$ (in the running example, this indicates no blacks in the precinct), $\beta_i^b$ (the fraction of blacks who vote) is undefined, and its estimate is set at missing. Similarly, if $X_i = 1$, $\beta_i^w$ is set to missing. In `ei2`, if $T_i = 0$ from the previous stage then $V_i$ should be undefined and both $\beta_i^b$ and $\beta_i^w$ should be missing (although it is best to remove observations for which $T_i = 0$ from the `ei2` run.)

**How do I test whether $B^b - B^w = 0$?** You can get simulations of this difference (racially polarized voter turnout in the running example) by doing: `aggs=eiread(dbuf,"aggs");` and `diff=aggs[.,1]-aggs[.,2];`. Then you can use `diff` as you might normally use simulations of $\beta_i^b$. For example, a point estimate of the aggregate difference is `meanc(diff)`, a standard error is `stdc(diff)`. You can also see the full posterior distribution by plotting the histogram with `call hist(diff,7)` or a kernel density estimate with `call dens(diff)`.

**How do I make inferences about aggregated continuous variables?** If the individual-level variables corresponding to $X_i$ and/or $T_i$ are continuous but restricted to the the unit interval (such as the fraction correct on a test) or rescaled to this interval, then this program can be used without any changes. Only the interpretation of the estimates change. See Section 14.3.

**How many observations do I need?** The answer depends on the problem. For example, one observation with $X_1 = 0$ will give you a correct answer for $\beta_1^w$ and a standard error of zero. On the other hand, two million observations with $X_i = 0$, won't be enough to estimate $B^b$. Another way to think about this question is that the basic model has five parameters and (like

a regression model with five parameters) you would probably want at least 30–50 observations or so. Problems for which the parameters are more highly correlated (like regression problems with high collinearity among the explanatory variables) will require additional observations to achieve the same level of confidence. (That is, check the tomography plot to ascertain what kind of problem you are working with.) You will also want additional observations if your substantive problem demands answers with more precision. Of course, if you can selectively add observations that are especially informative, then there may be great power to be had from collecting just a bit more data.

**Has this method been used in Court?** A federal district court judge declared that *A Solution to the Ecological Inference Problem* "is the best method currently available to measure racial bloc voting" [i.e., whether blacks and whites vote differently] (*Mallory, et al., v. State of Ohio, et al.*, USDC Southern District Court of Ohio, case no. C-2-95-381.) This decision was upheld on appeal before the three judges of the Circuit Court, and similar language was used to describe the method (in *Mallory, et al. v. State of Ohio, et al.*, United States Court of Appeals, Sixth Circuit, No. 97-4425, 4/13/99). It was also accepted by the court in *Bill Jones vs. Ca Democratic Party* and in a variety of others. In each case, $\mathfrak{E}I$ was unchallenged by experts on the opposing side. Since $\mathfrak{E}I$ starts with Goodman's regression, which had previously been declared as the appropriate method by the Supreme Court, and adds information known to be true (i.e., the bounds), EI has been relatively uncontroversial in these situations.

**What if I find a bug?** First try to replicate the problem with the current version of the program (available at `http://GKing.Harvard.Edu`). If the problem still exists, copy down *exactly* what you see on the screen when the program bombs, and email it along with any changes in globals to `kbenoit@tcd.ie` if you think the error was in the $\mathfrak{E}zI$ front end or `King@Harvard.Edu` if the error appears to be in $\mathfrak{E}I$. Comments and suggestions for changes are welcome.

## Acknowledgements

$\mathfrak{E}I$ is © Copyright 1995–2001, All rights reserved. (Center for Basic Research in the Social Sciences, 34 Kirkland Street, Harvard University, Cambridge MA 02138, USA. E-mail: `King@Harvard.Edu`, phone: (617) 495-2027.)

This document can be read on-line in hypertext format from `http://GKing.Harvard.Edu`.

$\mathfrak{E}I$ was written in the GAUSS Programming Language, ©Copyright Aptech Systems, Inc., and uses GAUSS' Constrained Maximum Likelihood Module written by Ronald J. Schoenberg. For making available public domain GAUSS code, I am grateful to David Baird (for an inverse cumulative normal procedure), Martin van der Ende (for an accurate cumulative bivariate normal procedure), and Simon Jackman (for a loess procedure). For research assistance, my thanks goes to Kosuke Imai and Eric Dickson, and for research support, my thanks goes to the National Science Foundation (IIS-9874747), the National Institutes of Aging (P01 AG17625-01), and the World Health Organization. $\mathfrak{E}I$ is copyrighted, but you may copy and distribute this program provided that no charge is made and the copy is identical to the original. To request an exception, please contact me.

## References

King G (1995). "Replication, Replication." *PS: Political Science and Politics*, **XXVIII**(3), 443–499. URL http://gking.harvard.edu/replrepl/replrepl.html.

King G (1997). *A Solution to the Ecological Inference Problem: Reconstructing Individual Behavior from Aggregate Data*. Princeton University Press, Princeton. URL http://gking.harvard.edu/preprints.shtml#sei.

### Affiliation:

Gary King
Center for Basic Research in the Social Sciences
Harvard University
34 Kirkland Street
Cambridge, Massachusetts 02138, USA
E-mail: King@Harvard.edu
URL: http://GKing.Harvard.Edu/

*Journal of Statistical Software*
September 2004, Volume 11, Issue 7.
http://www.jstatsoft.org/

*Submitted:* 2003-09-29
*Accepted:* 2004-09-28