Writing a *qi* function

Matthew Owen

October 4, 2010

1 Introduction

For any Zelig module, the *qi* function is ultimately the most important piece of code that must be written; it describes the actual process which simulates the *quantities of interest*. Because of the nature of this process - and the gamut of statistical packages and their underlying statistical model - it is rare that the simulation process can be generalized for arbitrary fitted models. Despite this, it is possible to break down the simulation process into smaller steps.

2 Notable Features of *qi* Function

The typical *qi* function has several basic procedures:

1. *Call the param function*: This is entirely optional but sometimes important for the clarity of your algorithm. This step typically consists of taking random draws from the fitted model's underlying probability distribution.
2. *Compute the Quantity of Interest*: Depending on your model, there are several ways to compute necessary quantities of interest. Typical methods for computing quantities of interest include:
   (a) Using the 'predict' method of your given linear model.
   (b) Using the sample provided by 'param' to generate simulations of the *Quantities of Interest*.
   (c) Using a Maximum-likelihood estimate on the fitted model.
3. *Create a list of titles for your Quantities of Interest*:
4. *Generate the Quantity of Interest Object*: Finally, with the computed Quantities of Interest, you must

3 Basic Layout of a *qi* Function

Now with the general outline of a *qi* function defined, it is important to discuss the expected procedures and specifics of implementation.

3.1 The Function's Signature

The *qi* function's signature accepts 4 parameters:

@z: An object of type "*zelig*". This wraps the fitted model in the slot "result".

@x: An object of type "*setx*". This object is used to compute important coefficients, parameters, and features of the data.frame passed to the function call.

@x1: Also an object of type "*setx*". This object is used in a similar fashion, however its presence allows a variety of *quantities of interest* to be computed. Notably, this is a necessary parameter to compute first-differences.

@num: The number of simulations to compute

3.2 Code Example: *qi* Function Signature

```
qi.your_model_name <- function(z, x=NULL, x1=NULL, num=1000) {
# start typing your code here
# ...
# ...
```

Note: In the above example, the function name "qi.your_model_name" is merely a placeholder. In order to register a *qi* function with zelig, the developer must follow the naming convention qi.*your mode name*, where *your_model_name* is the name of the developer's module. For example, if a developer titled his or her zelig module "logit", then the corresponding *qi* function is titled "*qi.logit*".

3.3 Call to the *param* Function

This step is common in many zelig models, however, its existence - though highly recommended - is purely optional. Typically, during this step, samples are taken from the distribution governing the statistical model. This is then used to simulate values for the *quantities of interest*.

3.4 The Function Body

The function body of *qi* function varies largely from model to model. As a result, it is impossible to create general guidelines to simulate *quantities of interest* - or even determine what the *quantity of interest* is. Typical methods for computing *quantities of interest* include:

- Implementing sampling algorithms based on the underlying fitted model, or
- "Predicting" a large number of values from the fitted model

3.5 The Return Value

In order for Zelig to process the simulations, they must be returned in one of several formats:

- ```
  list(
        "TITLE OF QI 1" = val1,
        "TITLE OF QI 2" = val2,
        # any number of title-val pairs
        # ...
        "TITLE OF QI N" = val.n
        )
  ```

- ```
  make.qi(
           titles = list(title1, title2),
           stats  = list(val1, val2)
           )
  ```

In the above example,*val1, val2*are data.frames, matrices, or lists representing the simulations of the *quantities of interests*, and *title1, title2* - and any number of titles - are character-strings that will act as human-readable descriptions of the *quantities of interest*. Once results are returned in this format, Zelig will convert the results into a machine-readable format and summarize the simulations into a comprehensible format.

NOTE: Because of its readability, it is suggested that the first method is used when returning *quantities of interest*.

4 Example *qi* function (qi.logit.R)

```
qi.ls <- function(z, x=NULL, x1=NULL, num=1000) {
  # error-catching
  if (missing(x))
    stop("x cannot be missing while computing the 'ls' model")


  # get 'parameters'
  # In this example, this amounts to sampling
  # a multivariate normal distribution
  coefs <- param(z, num=num)


  # compute expected values using X
  ev <- coefs %*% t(x$matrix)
  ev1 <- NA
  fd <- NA

  # if x1 exists:
  #   compute expected values using X1
  #   compute finite differences
  if (!is.null(x1)) {
    ev1 <- coefs %*% t(x1$matrix)
    fd <- ev1 - ev
  }

  # return
  list("Expected Value: E(Y|X)" = ev,
       "Expected Value (of X1): E(Y|X1)" = ev1,
       "First Difference in Expected Values: E(Y|X1) - E(Y|X)" = fd
       )
}
```

5 The *qi* API

In Development