

Bayesian Analysis with SAS®

Rodney A. Sparapani, MS <mailto:rsparapa@mcw.edu>
Center for Patient Care and Outcomes Research

October 25, 2007

Example 1: Some SAS macros for BUGS data

This dataset is called the “Surgical Unit Data”. 54 patients underwent a liver operation in the surgical unit. Y is the time to event (there was no censoring) and $\log X_1$ is the \log (blood-clotting score). The analysis was performed by OpenBUGS with the help of some SAS macros.

$$\begin{aligned}y_i &\sim N(\mu_i, 1/\tau) \\ \mu_i &= b_0 + b_1 * \log X_{1i} \\ b_0 &\sim N(0, 10^4) \\ b_1 &\sim N(0, 10^4) \\ \tau &\sim \text{Gamma}(0.001, 0.001)\end{aligned}$$

1. Open the file `Z:\examples\example1.sas` with SAS
2. Take a few moments to look it over: these investigations will generate a lot of output so we are going to restrict our attention to the slope, `b1`

`data surg...` this DATASTEP creates the temporary SAS dataset `SURG`; ordinarily we would want to specify a permanent SAS dataset, e.g. `LIB.SURG`, where `LIB` is a permanent SAS library specified with a `LIBNAME` statement

`%_lexport...` this SAS macro creates the OpenBUGS input file `example1.txt`

`/*...*/` the first comment block describes how I ran the example in OpenBUGS

`%_decoda...` this SAS macro reads the OpenBUGS output file and summarizes the data

`/*...*/` the second comment block describes how you might use the `debugs` macro

3. Submit the SAS program by pressing F3

In the SAS Explorer window, the Results tab will be populated with a hierarchical list of Results. Clicking on the \boxplus nodes will turn them into \boxminus , a hyperlist of Tables and/or Plots.

4. Navigate to \boxminus Univariate... \boxminus obs=0... \boxminus b1–Location Counts: μ_0 defaults to 0 which you can over-ride with the `MU0=` option to `decoda`. Estimate $P[b_1 > 0]$

5. Navigate to `Print...Data Set...`: Estimate L and U such that $P[L < b_1 < U] = 0.95$
6. Navigate to `Arima...obs=0...Identification 1-Autocorrelations`: Do consecutive samples of b_1 appear to be independent? See the discussion in the second comment block on the `debugs` macro mentioned above
7. Go on to explore other results of interest as time permits

Notice that the Results window will quickly get very crowded and confusing. If this was a problem that you were working on, you might save the SAS program and the SAS Output (also known as the SAS Listing). Then, you would clean up before going on to the next task. Here's how:

8. Select the Results tab
9. On the Display Manager command line above it, type: `clear; clear log; clear output`
10. And press enter

This will clear the current window (the Results tab in this case), the SAS Log window and the SAS Output window. We will do this at the end of each example. Just typing the first few letters on the Display Manager command line will bring up the command and you can press enter to run it again.

1 Example 2: PROC BGENMOD and the Surgical Unit Data

PROC BGENMOD Statement/option	Consequence
MODEL Y=... / DIST=NORMAL	$y_i \sim N(\mu_i, 1/Scale)$
MODEL ...=logX1	$\mu_i = b_{Intercept} + b_{logX1} * logX1_i$
default, BAYES PPRIOR=GAMMA	$Scale \sim \text{Gamma}(0.001, 0.001)$
default, BAYES COEFFPRIOR=UNIFORM	$b_{Intercept} \sim U(-\infty, \infty)$
	$b_{logX1} \sim U(-\infty, \infty)$

1. Open the file `Z:\examples\example2.sas` with SAS
2. Take a few moments to look it over. This is the same dataset as in Example 1. Once again, we are going to restrict our attention to the slope parameter. In SAS, the coefficient is represented by the name of the covariate: `logX1` in this case.
3. Notice that the first two statements are `ods html;` and `ods graphics on;`. These statements will create HTML with graphics output for our Tables and Plots. Press F3 to submit the SAS program. If you find Java error messages in the SAS Log window, then you will not be

able to produce HTML with graphics output until a Java conflict is resolved. Your best bet is to check out SAS Technical Support at <http://support.sas.com/> for this or any other seemingly unsolvable problem with SAS.

4. We requested a linear regression by specifying a `MODEL` statement with the `DIST=NORMAL` option which refers to the distribution of `Y`.
5. We requested a Bayesian analysis with the `BAYES` statement. The `SEED=` option allows you to set a seed that you can re-use later and get exactly the same results. The `COEFFPRIOR=` option allows us to specify a prior for the regression coefficients; the default prior is `UNIFORM`: the noninformative and improper prior of a constant. The `DIAGNOSTICS=` option allows us to request convergence diagnostics and `THIN=20` will thin the posterior (we'll come back to both of these later). The default is `PLOTS=ALL` which means produce trace, density and auto-correlations plots. Finally, the `ODS OUTPUT PosteriorSample=` statement allows us to save the posterior.
6. Navigate to `☐Bgenmod...☐Posterior Sample Convergence Diagnostics☐Geweke Test`: which reveals a list of two Geweke Test icons
 - (a) one associated with SAS (text only)
 - (b) one associated with Internet Explorer (for HTML with graphics)
7. double-click on the Geweke Test icon associated with Internet Explorer: the Geweke Test results will appear in an Internet Explorer window which appears within the SAS application
8. the Geweke Test is described on pp. 24-25 of the manual. However, an understanding of convergence testing relies on the underlying theory of Markov chains which you can investigate later. For the moment, it is important to note that Rejecting the Null Hypothesis implies that posterior sampling has not reached convergence. Has the `logX1` coefficient reached convergence?
9. if you scroll down, next you should come to the diagnostics plots: trace, density and auto-correlation. Do consecutive “thinned” samples of the `logX1` coefficient appear to be independent?
10. Estimate $P \left[b_{\log X1} > 0 \right]$
11. Go on to explore other results of interest

Now, clear the Results tab, etc., as we did in Example 1.

Example 3: Liver Cancer Clinical Trial Data

First PROC BGENMOD Statement/option	Consequence
<pre>MODEL Y=... / DIST=POISSON ... MODEL ...=X1-X6 / LINK=LOG ... BAYES COEFFPRIOR=NORMAL</pre>	$y_i \sim \text{Poisson}(\mu_i)$ $\log \mu_i = b_{\text{Intercept}} + b_{X1} * X1_i + \dots + b_{X6} * X6_i$ $b_{\text{Intercept}} \sim N(0, 10^6)$ $b_{X1} \sim N(0, 10^6)$ $b_{X2} \sim N(0, 10^6)$ <p>...</p> $b_{X6} \sim N(0, 10^6)$
Second PROC BGENMOD Statement/option	Consequence
<pre>MODEL Y=... / DIST=POISSON ... MODEL ...=X1-X6 / LINK=LOG ... BAYES COEFFPRIOR=NORMAL(INPUT=...)</pre>	$y_i \sim \text{Poisson}(\mu_i)$ $\log \mu_i = b_{\text{Intercept}} + b_{X1} * X1_i + \dots + b_{X6} * X6_i$ $b_{\text{Intercept}} \sim N(0, 10^6)$ $b_{X1} \sim N(0.0435, 0.0005)$ $b_{X2} \sim N(0, 10^6)$ <p>...</p> $b_{X6} \sim N(0, 10^6)$

1. Open the file `Z:\examples\example3.sas` with SAS
2. Take a few moments to look it over. The `DATA LIVER` block of code creates the dataset: `Y` is the number of cancerous nodes found in their liver when they have entered a clinical trial and `X1-X6` are six baseline covariates which may or may not be associated with `Y`.
3. Now, take a look at the two statements `%_LEXPOR`T and `%_SEXPOR`T. As we saw in `example1.sas`, `%_LEXPOR`T is used to create an OpenBUGS input file, specifically, an input file of scalars. Here, it serves the same purpose, however, we have added the option `CLOSE=0`. This leaves the file that we are creating, `example3.txt`, open for further additions. These further additions are appended by `%_SEXPOR`T when we give it the `APPEND=` option (had we used the `FILE=` option, the file would be re-created from the beginning and we would have lost the scalars). `%_SEXPOR`T is for vector data. In this case, `VAR=X1-X6` will create an array of vectors, covariates, which we would access in OpenBUGS as `x[i, j]`; contrast that with the outcome, a scalar, `y[i]`.
4. Submit the SAS program by pressing F3.
5. We requested a Poisson regression with a `MODEL` statement followed by the `DIST=POISSON` and `LINK=LOG` options.
6. We requested a Bayesian analysis with the `BAYES` statement and we requested independent, noninformative Normal priors, $N(0, 10^6)$, for the coefficients with `COEFFPRIOR=NORMAL` in the

first PROC BGENMOD. In the second PROC BGENMOD, we requested independent, noninformative Normal priors, $N(0, 10^6)$, for all coefficients except the coefficient for X1 which had an informative prior, $N(0.0435, 0.0005)$. That was accomplished by the option: `COEFFPRIOR=NORMAL(INPUT=NORMALPRIOR)`. The authors speculate that the informative prior on the coefficient of X1 will produce a posterior where the coefficient of X1 is more likely to be positive than for the noninformative prior. Navigate to the two `Univariate...x1-Location Counts` summaries and compare their results. Does the analysis support their contention that the informative prior leads to an X1 coefficient that is more likely to be positive than a noninformative prior?

- Finally, let's take a look at the results from the two calls to the BAYESINTERVALS macro. The BAYESINTERVALS macro creates multiple credible intervals, one for each parameter, which each would be wider than a single credible interval performed on each parameter. For example, you could compare the results of `BAYESINTERVALS(DATA=POST, VARS=X1-X6)` with `BAYESINTERVALS(DATA=POST, VARS=X1)`. Now, getting back to our concern: navigate to `Print...Data Set...` and compare the results for X1 from the informative and noninformative priors. Are there obvious differences? If so, how would you describe them?

Now, clear the Results tab, etc.

Example 4: Rat Cancer Data

First PROC BPHREG Statement/option	Consequence
<code>MODEL Days*Status(0)=Group ...</code> default, <code>BAYES COEFFPRIOR=UNIFORM</code>	$\lambda(t) = \lambda_0(t)e^{Group b_{Group}}$ $b_{Group} \sim U(-\infty, \infty)$
First PROC BPHREG Statement/option	Consequence
<code>MODEL Days*Status(0)=Group ...</code> <code>BAYES PIECEWISE=HAZARD ...</code> default, <code>BAYES COEFFPRIOR=UNIFORM</code>	$\lambda(t) = \left(\prod_{j=1}^8 \lambda_j^{I(a_{j-1} \leq t < a_j)} \right) e^{Group b_{Group}}$ $b_{Group} \sim U(-\infty, \infty)$

- Open the file `Z:\examples\example4.sas` with SAS
- Take a few moments to look it over. The `DATA RATS` block of code creates the dataset: 40 female rats received one of two treatments, `GROUP`, prior to an exposure by a carcinogen; `DAYS` is the days to death by vaginal cancer; censoring was represented by `STATUS=0`. Here we represent the time to event data, with censoring, by the SAS syntax: `Days*Status(0)=Group`. However, suppose that we would also like to create an OpenBUGS input file for, let's say, a

Weibull regression. The way to represent censored data in OpenBUGS is exemplified by `SURV` for the time to event and, `CENS`, a censoring time. `SURV` would be missing when censored and `CENS` would be > 0 . `SURV` would be > 0 for an event and `CENS` would be 0. The OpenBUGS code would be something like: `surv[i] ~ dweib(r, lambda[i]) I(cens[i],);`. The line to create that OpenBUGS input file is `%_LEXPORT...`

3. Submit the SAS program by pressing F3.
4. Instead of the Geweke Test, we have requested the Gelman-Rubin Test with the option `BAYES ... DIAGNOSTICS=(GELMAN)` which is explained on pp. 22-24. The Gelman-Rubin test is often preferable to the Geweke Test, but, in the interest of time, we have used the Geweke Test. The Gelman-Rubin test is more computationally intensive and, therefore, more time-consuming. The Gelman-Rubin Test requires parallel chains. By default, 3 chains are generated. The option `BAYES ... NMC=2000` requests that each chain has 2000 iterations; the default is 10000. Since we need 3 parallel chains, $3 \times 2000 = 6000$ iterations will be needed. If we had left it at the default, $3 \times 10000 = 30000$ iterations would be necessary. But, don't confuse this as a good idea. If this were a real problem, we would most likely generate the 30000 iterations and check for convergence. It's only since we are pressed for time, and it seems to work for this example, that this shortcut is permissible. Gelman-Rubin statistics "near" 1 suggest convergence. By "near" 1, we mean 1 to several decimal places. The SAS output also provides us with a 97.5% upper bound. Check both of these models for convergence by navigating to `▢Bphreg...▢Convergence...▢Gelman...-Gelman and Rubin`. OpenBUGS also provides the Gelman-Rubin Test when multiple chains are requested, but presents it in a graphical format rather than as table of statistics.
5. In the first `PROC BPHREG`, we have requested a Cox model which is the default. In the second `PROC BPHREG`, we have requested a piecewise exponential model with the option `BAYES PIECEWISE=HAZARD`. The piecewise exponential model defaults to 8 intervals each having a roughly equal number of events and a different parameter λ_j . The `HAZARDRATIO 'Group 1 vs. Group 0' GROUP` command generates a summary of $e^{b_{Group}}$ rather than b_{Group} . The quantity $e^{b_{Group}}$ is called the hazard ratio by SAS, but is more commonly known as the risk ratio or relative risk. A risk ratio > 1 (< 1) means that, statistically speaking, an individual in `Group 1` is more (less) likely to experience an event at any given time than an individual in `Group 0`. Navigate to `▢Bphreg...▢Group-Group` and compare the credible intervals for the two models. Does the treatment `GROUP` have any impact on the outcome event? If so, then what is it?
6. Explore other items of interest as time permits

This demonstration should have given you an idea of what SAS has to offer for Bayesian analysis. And, I hope that you enjoyed it. Please close the SAS application, but do NOT log off. Thanks for coming.