# SAS/ETS® 13.2 User's Guide: High-Performance Procedures

# The HPCOUNTREG Procedure

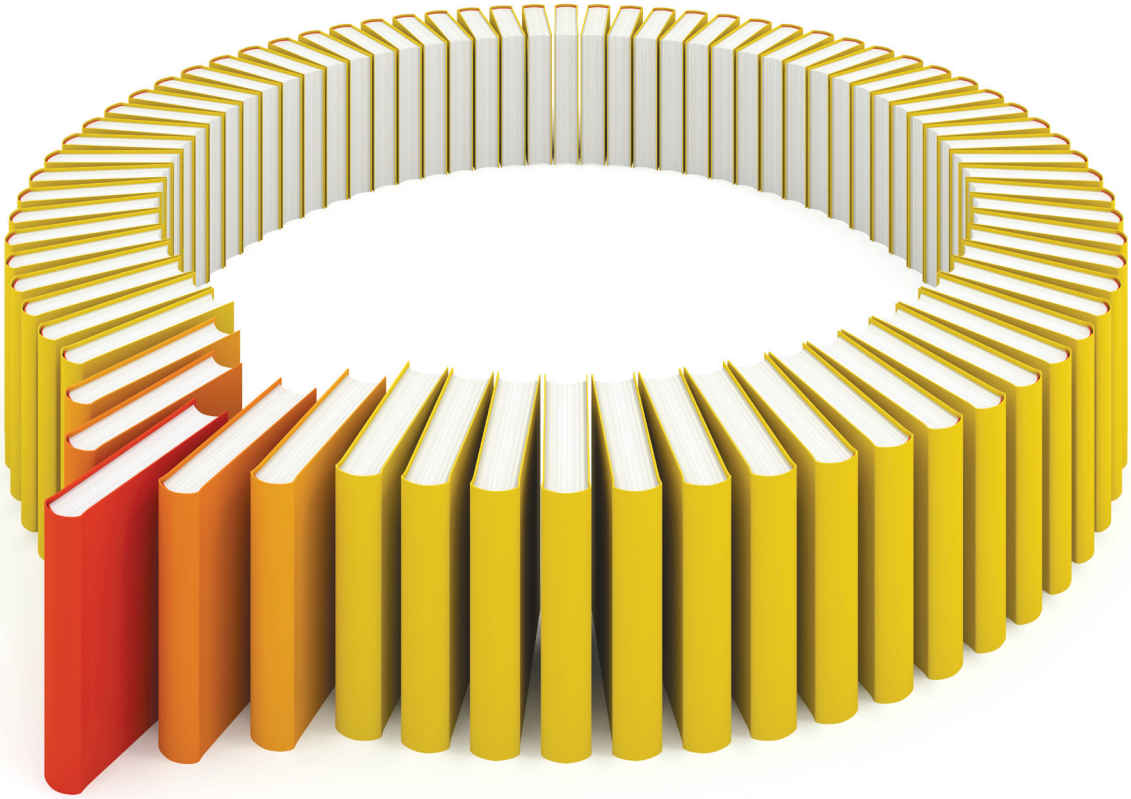
The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2014. *SAS/ETS® 13.2 User's Guide: High-Performance Procedures*. Cary, NC: SAS Institute Inc.

**SAS/ETS® 13.2 User's Guide: High-Performance Procedures**



SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

August 2014



# Chapter 6
# The HPCOUNTREG Procedure

## Contents

# Overview: HPCOUNTREG Procedure

The HPCOUNTREG procedure is a high-performance version of the COUNTREG procedure in SAS/ETS software. Like the COUNTREG procedure, the HPCOUNTREG procedure fits regression models in which the dependent variable takes on nonnegative integer or count values. Unlike the COUNTREG procedure, which can be run only on an individual workstation, the HPCOUNTREG procedure takes advantage of a computing environment that enables it to distribute the optimization task among one or more nodes. In addition, each node can use one or more threads to carry out the optimization on its subset of the data. When several nodes are employed, with each node using several threads to carry out its part of the work, the result is a highly parallel computation that provides a dramatic gain in performance.

The HPCOUNTREG procedure enables you to read and write data in distributed form and perform analyses in distributed mode and single-machine mode. For information about how to affect the execution mode of SAS high-performance analytical procedures, see the section "Processing Modes" on page 10 in Chapter 3, "Shared Concepts and Topics."

The HPCOUNTREG procedure is specifically designed to operate in the high-performance distributed environment. By default, PROC HPCOUNTREG performs computations in multiple threads.

## PROC HPCOUNTREG Features

The HPCOUNTREG procedure estimates the parameters of a count regression model by maximum likelihood techniques. The following list summarizes some basic features of the HPCOUNTREG procedure:

- can perform analysis on a massively parallel high-performance appliance
- reads input data in parallel and writes output data in parallel when the data source is the appliance database
- is highly multithreaded during all phases of analytic execution
- performs maximum likelihood estimation
- supports multiple link functions
- uses the WEIGHT statement for weighted analysis
- uses the FREQ statement for grouped analysis
- uses the OUTPUT statement to produce a data set that contains predicted probabilities and other observationwise statistics

# Getting Started: HPCOUNTREG Procedure

Except for its ability to operate in the high-performance distributed environment, the HPCOUNTREG procedure is similar in use to other regression model procedures in the SAS System. For example, the following statements are used to estimate a Poisson regression model:

```
proc hpcountreg data=one ;
   model y = x / dist=poisson ;
run;
```

The response variable $y$ is numeric and has nonnegative integer values.

This section illustrates two simple examples that use PROC HPCOUNTREG. The data are taken from Long (1997). This study examines how factors such as gender (fem), marital status (mar), number of young children (kid5), prestige of the graduate program (phd), and number of articles published by a scientist's mentor (ment) affect the number of articles (art) published by the scientist.

The first 10 observations are shown in Figure 6.1.

**Figure 6.1** Article Count Data

| Obs | art | fem | mar | kid5 | phd | ment |
|---|---|---|---|---|---|---|
| **1** | 3 | 0 | 1 | 2 | 1.38000 | 8.0000 |
| **2** | 0 | 0 | 0 | 0 | 4.29000 | 7.0000 |
| **3** | 4 | 0 | 0 | 0 | 3.85000 | 47.0000 |
| **4** | 1 | 0 | 1 | 1 | 3.59000 | 19.0000 |
| **5** | 1 | 0 | 1 | 0 | 1.81000 | 0.0000 |
| **6** | 1 | 0 | 1 | 1 | 3.59000 | 6.0000 |
| **7** | 0 | 0 | 1 | 1 | 2.12000 | 10.0000 |
| **8** | 0 | 0 | 1 | 0 | 4.29000 | 2.0000 |
| **9** | 3 | 0 | 1 | 2 | 2.58000 | 2.0000 |
| **10** | 3 | 0 | 1 | 1 | 1.80000 | 4.0000 |

The following SAS statements estimate the Poisson regression model. The model is executed in the distributed computing environment with two threads and four nodes.

```
/*-- Poisson Regression --*/
proc hpcountreg data=long97data;
   model art = fem mar kid5 phd ment / dist=poisson method=quanew;
   performance nthreads=2 nodes=4 details;
run;
```

The "Model Fit Summary" table that is shown in Figure 6.2 lists several details about the model. By default, the HPCOUNTREG procedure uses the Newton-Raphson optimization technique. The maximum log-likelihood value is shown, in addition to two information measures—Akaike's information criterion (AIC) and Schwarz's Bayesian information criterion (SBC)—which can be used to compare competing Poisson models. Smaller values of these criteria indicate better models.

**Figure 6.2** Estimation Summary Table for a Poisson Regression

**The HPCOUNTREG Procedure**

| Model Fit Summary | |
|---|---|
| **Dependent Variable** | art |
| **Number of Observations** | 915 |
| **Data Set** | WORK.LONG97DATA |
| **Model** | Poisson |
| **Log Likelihood** | -1651 |
| **Maximum Absolute Gradient** | 0.0002080 |
| **Number of Iterations** | 13 |
| **Optimization Method** | Quasi-Newton |
| **AIC** | 3314 |
| **SBC** | 3343 |

Figure 6.3 shows the parameter estimates of the model and their standard errors. All covariates are significant predictors of the number of articles, except for the prestige of the program (phd), which has a $p$-value of 0.6271.

**Figure 6.3** Parameter Estimates of Poisson Regression

| Parameter Estimates | | | | | |
|---|---|---|---|---|---|
| **Parameter** | **DF** | **Estimate** | **Standard Error** | **t Value** | **Pr > \|t\|** |
| **Intercept** | 1 | 0.3046 | 0.1030 | 2.96 | 0.0031 |
| **fem** | 1 | -0.2246 | 0.05461 | -4.11 | <.0001 |
| **mar** | 1 | 0.1552 | 0.06137 | 2.53 | 0.0114 |
| **kid5** | 1 | -0.1849 | 0.04013 | -4.61 | <.0001 |
| **phd** | 1 | 0.01282 | 0.02640 | 0.49 | 0.6271 |
| **ment** | 1 | 0.02554 | 0.002006 | 12.73 | <.0001 |

To allow for variance greater than the mean, you can fit the negative binomial model instead of the Poisson model by specifying the DIST=NEGBIN option, as shown in the following statements. Whereas the Poisson model requires that the conditional mean and conditional variance be equal, the negative binomial model allows for overdispersion, in which the conditional variance can exceed the conditional mean.

```
/*-- Negative Binomial Regression --*/
proc hpcountreg data=long97data;
   model art = fem mar kid5 phd ment / dist=negbin(p=2) method=quanew;
   performance nthreads=2 nodes=4 details;
run;
```

Figure 6.4 shows the fit summary and Figure 6.5 shows the parameter estimates.

**Figure 6.4** Estimation Summary Table for a Negative Binomial Regression

**The HPCOUNTREG Procedure**

| Model Fit Summary | |
|---|---|
| **Dependent Variable** | art |
| **Number of Observations** | 915 |
| **Data Set** | WORK.LONG97DATA |
| **Model** | NegBin |
| **Log Likelihood** | -1561 |
| **Maximum Absolute Gradient** | 0.0000666 |
| **Number of Iterations** | 16 |
| **Optimization Method** | Quasi-Newton |
| **AIC** | 3136 |
| **SBC** | 3170 |

**Figure 6.5** Parameter Estimates of Negative Binomial Regression

| Parameter Estimates | | | | | |
|---|---|---|---|---|---|
| **Parameter** | **DF** | **Estimate** | **Standard Error** | **t Value** | **Pr > \|t\|** |
| **Intercept** | 1 | 0.2561 | 0.1386 | 1.85 | 0.0645 |
| **fem** | 1 | -0.2164 | 0.07267 | -2.98 | 0.0029 |
| **mar** | 1 | 0.1505 | 0.08211 | 1.83 | 0.0668 |
| **kid5** | 1 | -0.1764 | 0.05306 | -3.32 | 0.0009 |
| **phd** | 1 | 0.01527 | 0.03604 | 0.42 | 0.6718 |
| **ment** | 1 | 0.02908 | 0.003470 | 8.38 | <.0001 |
| **_Alpha** | 1 | 0.4416 | 0.05297 | 8.34 | <.0001 |

The parameter estimate for _Alpha of *0.4416* is an estimate of the dispersion parameter in the negative binomial distribution. A $t$ test for the hypothesis $H_0 : \alpha = 0$ is provided. It is highly significant, indicating overdispersion ($p < 0.0001$).

The null hypothesis $H_0 : \alpha = 0$ can be also tested against the alternative $\alpha > 0$ by using the likelihood ratio test, as described by Cameron and Trivedi (1998, pp. 45, 77–78). The likelihood ratio test statistic is equal to $-2(\mathcal{L}_P - \mathcal{L}_{NB}) = -2(-1651 + 1561) = 180$, which is highly significant, providing strong evidence of overdispersion.

# Syntax: HPCOUNTREG Procedure

The following statements are available in the HPCOUNTREG procedure. Items within angle brackets (< >) or square brackets ([ ]) are optional.

**PROC HPCOUNTREG** *<options>* **;**
**BOUNDS** *bound1* [ *, bound2 . . .* ] **;**
**BY** *variables* **;**
**FREQ** *freq-variable* **;**
**INIT** *initialization1 < , initialization2 . . . >* **;**
**MODEL** *dependent-variable* **=** *regressors </ options>* **;**
**OUTPUT** *<output-options>* **;**
**PERFORMANCE** *performance-options* **;**
**RESTRICT** *restriction1* [*, restriction2 . . .* ] **;**
**WEIGHT** *variable </ option>* **;**
**ZEROMODEL** *dependent-variable* ∼ *zero-inflated-regressors </ options>* **;**

There can be only one MODEL statement. The ZEROMODEL statement, if used, must appear after the MODEL statement. If a FREQ or WEIGHT statement is specified more than once, the variable specified in the first instance is used.

## Functional Summary

Table 6.1 summarizes the statements and options used with the HPCOUNTREG procedure.

**Table 6.1** PROC HPCOUNTREG Functional Summary

| Description | Statement | Option |
|---|---|---|
| **Data Set Options** | | |
| Specifies the input data set | HPCOUNTREG | DATA= |
| Specifies the identification variable for panel data analysis | HPCOUNTREG | GROUPID= |
| Writes parameter estimates to an output data set | HPCOUNTREG | OUTEST= |
| Writes estimates to an output data set | OUTPUT | OUT= |
| Specifies BY-group processing | BY | |
| Specifies an optional frequency variable | FREQ | |
| Specifies an optional weight variable | WEIGHT | |
| **Printing Control Options** | | |
| Prints the correlation matrix of the estimates | HPCOUNTREG | CORRB |
| Prints the covariance matrix of the estimates | HPCOUNTREG | COVB |
| Suppresses the normal printed output | HPCOUNTREG | NOPRINT |
| Requests all printing options | HPCOUNTREG | PRINTALL |
| **Options to Control the Optimization Process** | | |
| Specifies maximum number of iterations allowed | HPCOUNTREG | MAXITER= |
| Selects the iterative minimization method to use | HPCOUNTREG | METHOD= |
| Specifies maximum number of iterations allowed | HPCOUNTREG | MAXITER= |
| Specifies maximum number of function calls | HPCOUNTREG | MAXFUNC= |
| Specifies the upper limit of CPU time in seconds | HPCOUNTREG | MAXTIME= |
| Specifies absolute function convergence criterion | HPCOUNTREG | ABSCONV= |

| Description | Statement | Option |
|---|---|---|
| Specifies absolute function convergence criterion | HPCOUNTREG | ABSFCONV= |
| Specifies absolute gradient convergence criterion | HPCOUNTREG | ABSGCONV= |
| Specifies relative function convergence criterion | HPCOUNTREG | FCONV= |
| Specifies relative gradient convergence criterion | HPCOUNTREG | GCONV= |
| Specifies absolute parameter convergence criterion | HPCOUNTREG | ABSXCONV= |
| Specifies matrix singularity criterion | HPCOUNTREG | SINGULAR= |
| Sets boundary restrictions on parameters | BOUNDS | |
| Sets initial values for parameters | INIT | |
| Sets linear restrictions on parameters | RESTRICT | |
| **Model Estimation Options** | | |
| Specifies the type of model | HPCOUNTREG | DIST= |
| Specifies the type of covariance matrix | HPCOUNTREG | COVEST= |
| Specifies the type of error components model for panel data | MODEL | ERRORCOMP= |
| Suppresses the intercept parameter | MODEL | NOINT |
| Specifies the offset variable | MODEL | OFFSET= |
| Specifies the zero-inflated offset variable | ZEROMODEL | OFFSET= |
| Specifies the zero-inflated link function | ZEROMODEL | LINK= |
| **Output Control Options** | | |
| Includes covariances in the OUTEST= data set | HPCOUNTREG | COVOUT |
| Includes correlations in the OUTEST= data set | HPCOUNTREG | CORROUT |
| Outputs SAS variables to the output data set | OUTPUT | COPYVAR= |
| Outputs probability of the actual value | OUTPUT | PROB= |
| Outputs expected value of response variable | OUTPUT | PRED= |
| Outputs estimates of XBeta $= \mathbf{x}_i'\boldsymbol{\beta}$ | OUTPUT | XBETA= |
| Outputs estimates of ZGamma $= \mathbf{z}_i'\boldsymbol{\gamma}$ | OUTPUT | ZGAMMA= |
| Outputs probability of a zero value as a result of the zero-generating process | OUTPUT | PROBZERO= |
| **Performance Options** | | |
| Requests a table that shows a timing breakdown | PERFORMANCE | DETAILS |
| Specifies the number of threads to use | PERFORMANCE | NTHREADS= |
| Specifies the number of nodes to use on the SAS appliance | PERFORMANCE | NODES= |

## PROC HPCOUNTREG Statement

**PROC HPCOUNTREG** <*options*> **;**

The following *options* can be used in the PROC HPCOUNTREG statement.

## Input Data Set Options

**DATA=***SAS-data-set*
specifies the input SAS data set. If the DATA= option is not specified, PROC HPCOUNTREG uses the most recently created SAS data set.

**GROUPID=***variable*
specifies an identification variable when a panel data model is estimated. The identification variable is used as a cross-sectional ID variable.

## Output Data Set Options

**OUTEST=***SAS-data-set*
writes the parameter estimates to the specified output data set.

**CORROUT**
writes the correlation matrix for the parameter estimates to the OUTEST= data set. This option is valid only if the OUTEST= option is specified.

**COVOUT**
writes the covariance matrix for the parameter estimates to the OUTEST= data set. This option is valid only if the OUTEST= option is specified.

## Printing Options

You can specify the following options in either the PROC HPCOUNTREG statement or the MODEL statement:

**CORRB**
prints the correlation matrix of the parameter estimates.

**COVB**
prints the covariance matrix of the parameter estimates.

**NOPRINT**
suppresses all printed output.

**PRINTALL**
requests all printing options.

## Estimation Control Options

You can specify the following options in either the PROC HPCOUNTREG statement or the MODEL statement:

**COVEST=HESSIAN | OP | QML**
specifies the type of covariance matrix for the parameter estimates.

The default is COVEST=HESSIAN. You can specify the following values:

| | |
|---|---|
| **HESSIAN** | specifies the covariance from the Hessian matrix. |
| **OP** | specifies the covariance from the outer product matrix. |
| **QML** | specifies the covariance from the outer product and Hessian matrices. |

## Optimization Control Options

PROC HPCOUNTREG uses the nonlinear optimization (NLO) subsystem to perform nonlinear optimization tasks. You can specify the following *options* in either the PROC HPCOUNTREG statement or the MODEL statement.

**ABSCONV=***r*

**ABSTOL=***r*

specifies an absolute function value convergence criterion by which minimization stops when $f(\theta^{(k)}) \leq r$. The default value of $r$ is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

**ABSFCONV=***r*

**ABSFTOL=***r*

specifies an absolute function difference convergence criterion by which minimization stops when the function value has a small change in successive iterations:

$$|f(\theta^{(k-1)}) - f(\theta^{(k)})| \leq r$$

The default is 0.

**ABSGCONV=***r*

**ABSGTOL=***r*

specifies an absolute gradient convergence criterion. Optimization stops when the maximum absolute gradient element is small:

$$\max_j |g_j(\theta^{(k)})| \leq r$$

The default is 1E–5.

**ABSXCONV=***r*

**ABSXTOL=***r*

specifies an absolute parameter convergence criterion. Optimization stops when the Euclidean distance between successive parameter vectors is small:

$$\| \theta^{(k)} - \theta^{(k-1)} \|_2 \leq r$$

The default is 0.

**FCONV=***r*

**FTOL=***r*

specifies a relative function convergence criterion. Optimization stops when a relative change of the function value in successive iterations is small:

$$\frac{|f(\theta^{(k)}) - f(\theta^{(k-1)})|}{|f(\theta^{(k-1)})|} \leq r$$

The default value is $2\epsilon$, where $\epsilon$ denotes the machine precision constant, which is the smallest double-precision floating-point number such that $1 + \epsilon > 1$.

**GCONV=***r*

**GTOL=***r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA, optimization stops when the normalized predicted function reduction is small:

$$\frac{g(\theta^{(k)})^T[H^{(k)}]^{-1}g(\theta^{(k)})}{|f(\theta^{(k)})|} \leq r$$

For the CONGRA technique (where a reliable Hessian estimate $H$ is not available), the following criterion is used:

$$\frac{\| g(\theta^{(k)}) \|_2^2 \quad \| s(\theta^{(k)}) \|_2}{\| g(\theta^{(k)}) - g(\theta^{(k-1)}) \|_2 \, |f(\theta^{(k)})|} \leq r$$

The default is 1E–8.

**MAXFUNC=***i*

**MAXFU=***i*

specifies the maximum number of function calls in the optimization process. The default is 1,000.

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number of calls that are specified by this option.

**MAXITER=***i*

**MAXIT=***i*

specifies the maximum number of iterations in the optimization process. The default is 200.

**MAXTIME=***r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The default value is the largest floating-point double representation of your computer. The time that is specified by this option is checked only once at the end of each iteration. Therefore, the actual run time can be much longer than *r*. The actual run time includes the remaining time needed to finish the iteration and the time needed to generate the output of the results.

**METHOD=***value*

specifies the iterative minimization method to use. The default is METHOD=NEWRAP. You can specify the following *values*:

**CONGRA** specifies the conjugate-gradient method.

**DBLDOG** specifies the double-dogleg method.

**NEWRAP** specifies the Newton-Raphson method (this is the default).

**NONE** specifies that no optimization be performed beyond using the ordinary least squares method to compute the parameter estimates.

**NRRIDG** specifies the Newton-Raphson Ridge method.

**QUANEW** specifies the quasi-Newton method.

**TRUREG** specifies the trust region method.

**SINGULAR=***r*

specifies the general singularity criterion that is applied by the HPCOUNTREG procedure in sweeps and inversions. The default is 1E–8.

## BOUNDS Statement

**BOUNDS** *bound1* [*, bound2 . . .*] **;**

The BOUNDS statement imposes simple boundary constraints on the parameter estimates. You can specify any number of BOUNDS statements.

Each *bound* is composed of parameter names, constants, and inequality operators as follows:

*item operator item* [ *operator item* [ *operator item . . .* ] ]

Each *item* is a constant, a parameter name, or a list of parameter names. Each *operator* is <, >, <=, or >=. Parameter names are as shown in the Effect column of the "Parameter Estimates" table.

You can use both the BOUNDS statement and the RESTRICT statement to impose boundary constraints. However, the BOUNDS statement provides a simpler syntax for specifying these kinds of constraints. For more information, see the section "RESTRICT Statement" on page 137.

The following BOUNDS statement illustrates the use of parameter lists to specify boundary constraints. It constrains the estimates of the parameter for z to be negative, the parameters for x1 through x10 to be between 0 and 1, and the parameter for x1 in the zero-inflation model to be less than 1.

```
bounds z < 0,
       0 < x1-x10 < 1,
       Inf_x1 < 1;
```

## BY Statement

**BY** *variables* **;**

A BY statement can be used with PROC HPCOUNTREG to obtain separate analyses on observations in groups defined by the BY variables. When a BY statement appears, the input data set should be sorted in order of the BY variables.

BY statement processing is not supported when the HPCOUNTREG procedure runs alongside the database or alongside the Hadoop Distributed File System (HDFS). These modes are used if the input data are stored in a database or HDFS and the grid host is the appliance that houses the data.

## FREQ Statement

**FREQ** *freq-variable* **;**

The FREQ statement identifies a variable (*freq-variable*) that contains the frequency of occurrence of each observation. PROC HPCOUNTREG treats each observation as if it appears *n* times, where *n* is the value

of *freq-variable* for the observation. If the value for the observation is not an integer, it is truncated to an integer. If the value is less than 1 or missing, the observation is not used in the model fitting. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

## INIT Statement

**INIT** *initialization1* < , *initialization2* . . . > **;**

The INIT statement sets initial values for parameters in the optimization.

Each *initialization* is written as a parameter or parameter list, followed by an optional equal sign (=), followed by a number:

*parameter* <=> *number*

Parameter names are as shown in the Effect column of the "Parameter Estimates" table.

## MODEL Statement

**MODEL** *dependent-variable* **=** *regressors* </ *options*> **;**

The MODEL statement specifies the dependent variable and independent regressor variables for the regression model. The dependent count variable should take only nonnegative integer values from the input data set. PROC HPCOUNTREG rounds any positive noninteger count value to the nearest integer. PROC HPCOUNTREG discards any observation that has a negative count.

Only one MODEL statement can be specified. You can specify the following *options* in the MODEL statement after a slash (/).

**DIST=***value*
specifies a type of model to be analyzed. You can specify the following *values*:

**POISSON | P** specifies the Poisson regression model.

**NEGBIN(P=1)** specifies the negative binomial regression model that uses a linear variance function.

**NEGBIN(P=2) | NEGBIN** specifies the negative binomial regression model that uses a quadratic variance function.

**ZIPOISSON | ZIP** specifies zero-inflated Poisson regression.

**ZINEGBIN | ZINB** specifies zero-inflated negative binomial regression.

You can also specify the DIST option in the HPCOUNTREG statement.

**ERRORCOMP=FIXED | RANDOM**
specifies a type of conditional panel model to be analyzed. You can specify the following model types:

**FIXED** specifies a fixed-effect error component regression model.

**RANDOM** specifies a random-effect error component regression model.

**NOINT**
suppresses the intercept parameter.

**OFFSET=***offset-variable*
specifies a variable in the input data set to be used as an offset variable. The *offset-variable* is used to allow the observational units to vary across observations. For example, when the number of shipping accidents could be measured across different time periods or the number of students who participate in an activity could be reported across different class sizes, the observational units need to be adjusted to a common denominator by using the offset variable. The offset variable appears as a covariate in the model with its parameter restricted to 1. The offset variable cannot be the response variable, the zero-inflation offset variable (if any), or any of the explanatory variables. The "Model Fit Summary" table gives the name of the data set variable that is used as the offset variable; it is labeled "Offset."

### Printing Options

You can specify the following options in either the PROC HPCOUNTREG statement or the MODEL statement:

**CORRB**
prints the correlation matrix of the parameter estimates.

**COVB**
prints the covariance matrix of the parameter estimates.

**NOPRINT**
suppresses all printed output.

**PRINTALL**
requests all printing options.

## OUTPUT Statement

**OUTPUT** < *output-options* > **;**

The OUTPUT statement creates a new SAS data set that includes variables created by the *output-options*. These variables include the estimates of $\mathbf{x}_i'\boldsymbol{\beta}$, the expected value of the response variable, and the probability of the response variable taking on the current value. Furthermore, if a zero-inflated model was fit, you can request that the output data set contain the estimates of $\mathbf{z}_i'\boldsymbol{\gamma}$ and the probability that the response is zero as a result of the zero-generating process. These statistics can be computed for all observations in which the regressors are not missing, even if the response is missing. By adding observations that have missing response values to the input data set, you can compute these statistics for new observations or for settings of the regressors that are not present in the data without affecting the model fit.

You can specify only one OUTPUT statement. You can specify the following *output-options*:

**OUT=***SAS-data-set*
names the output data set

**COPYVAR=***SAS-variable-names*
**COPYVARS=***SAS-variable-names*
adds SAS variables to the output data set.

**PRED=***name*
names the variable to contain the predicted value of the response variable.

**PROB=***name*
names the variable to contain the probability that the response variable will take the actual value, $\Pr(Y = y_i)$.

**PROBCOUNT(***value1* **<** *value2* **. . . >)**
outputs the probability that the response variable will take particular values. Each value should be a nonnegative integer. Nonintegers are rounded to the nearest integer. For *value*, you can also specify a list of the form X TO Y BY Z. For example, PROBCOUNT(0 1 2 TO 10 BY 2 15) requests predicted probabilities for the counts 0, 1, 2, 4, 5, 6, 8, 10, and 15. This option is not available for the fixed- and random-effects panel models.

**PROBZERO=***name*
names the variable to contain the value of $\varphi_i$, which is the probability that the response variable will take the value of 0 as a result of the zero-generating process. This variable is written to the output file only if the model is zero-inflated.

**XBETA=***name*
names the variable to contain estimates of $\mathbf{x}_i'\boldsymbol{\beta}$.

**ZGAMMA=***name*
names the variable to contain estimates of $\mathbf{z}_i'\boldsymbol{\gamma}$.

## PERFORMANCE Statement

**PERFORMANCE** < *performance-options* > **;**

The PERFORMANCE statement specifies options to control the multithreaded and distributed computing environment and requests detailed results about the performance characteristics of the HPCOUNTREG procedure. You can also use the PERFORMANCE statement to control whether the HPCOUNTREG procedure executes in single-machine or distributed mode. The most commonly used *performance-options* in the PERFORMANCE statement are as follows:

**DETAILS**
requests a table that shows a timing breakdown of the procedure steps.

**NODES=***n*
specifies the number of nodes in the distributed computing environment, provided that the data are not processed alongside the database.

**NTHREADS=***n*
specifies the number of threads for analytic computations and overrides the SAS system option THREADS | NOTHREADS. If you do not specify the NTHREADS= option, PROC HPCOUNTREG creates one thread per CPU for the analytic computations.

For more information about the PERFORMANCE statement for high-performance analytical procedures, see the section "PERFORMANCE Statement" on page 36 of Chapter 3, "Shared Concepts and Topics."

## RESTRICT Statement

**RESTRICT** *restriction1* [, *restriction2* . . . ] **;**

The RESTRICT statement imposes linear restrictions on the parameter estimates. You can specify any number of RESTRICT statements.

Each *restriction* is written as an expression, followed by an equality operator (=) or an inequality operator (<, >, <=, >=) and then by a second expression, as follows:

*expression operator expression*

The *operator* can be =, <, >, <=, or >=.

Restriction expressions can be composed of parameter names, constants, and the following operators: times ($*$), plus ($+$), and minus ($-$). Parameter names are as shown in the Effect column of the "Parameter Estimates" table. The restriction expressions must be a linear function of the variables.

Lagrange multipliers are reported in the "Parameter Estimates" table for all the active linear constraints. They are identified by the names Restrict1, Restrict2, and so on. The probabilities of these Lagrange multipliers are computed using a beta distribution (LaMotte 1994). Nonactive (nonbinding) restrictions have no effect on the estimation results and are not noted in the output.

The following RESTRICT statement constrains the negative binomial dispersion parameter $\alpha$ to 1, which restricts the conditional variance to be $\mu + \mu^2$:

```
restrict _Alpha = 1;
```

## WEIGHT Statement

**WEIGHT** *variable* < / *option* > **;**

The WEIGHT statement specifies a variable to supply weighting values to use for each observation in estimating parameters. The log likelihood for each observation is multiplied by the corresponding weight variable value.

If the weight of an observation is nonpositive, that observation is not used in the estimation.

The following *option* can be added to the WEIGHT statement after a slash (/).

**NONORMALIZE**

does not normalize the weights. (By default, the weights are normalized so that they add up to the actual sample size. The weights $w_i$ are normalized by multiplying them by $\frac{n}{\sum_{i=1}^{n} w_i}$, where $n$ is the sample size.) If the weights are required to be used as they are, then specify the NONORMALIZE option.

## ZEROMODEL Statement

**ZEROMODEL** *dependent-variable* ∼ *zero-inflated-regressors* < / *options* > **;**

The ZEROMODEL statement is required if either ZIP or ZINB is specified in the DIST= option in the MODEL statement. If ZIP or ZINB is specified, then the ZEROMODEL statement must follow the MODEL statement. The dependent variable in the ZEROMODEL statement must be the same as the dependent variable in the MODEL statement.

The zero-inflated (ZI) regressors appear in the equation that determines the probability ($\varphi_i$) of a zero count. Each of these $q$ variables has a parameter to be estimated in the regression. For example, let $\mathbf{z}_i'$ be the $i$th observation's $1 \times (q+1)$ vector of values of the $q$ ZI explanatory variables ($w_0$ is set to 1 for the intercept term). Then $\varphi_i$ is a function of $\mathbf{z}_i'\boldsymbol{\gamma}$, where $\boldsymbol{\gamma}$ is the $(q+1) \times 1$ vector of parameters to be estimated. (The zero-inflated intercept is $\gamma_0$; the coefficients for the $q$ zero-inflated covariates are $\gamma_1, \ldots, \gamma_q$.) If $q$ is equal to 0 (no ZI explanatory variables are provided), then only the intercept term $\gamma_0$ is estimated. The "Parameter Estimates" table in the displayed output shows the estimates for the ZI intercept and ZI explanatory variables; they are labeled with the prefix "Inf_". For example, the ZI intercept is labeled "Inf_intercept". If you specify Age (a variable in your data set) as a ZI explanatory variable, then the "Parameter Estimates" table labels the corresponding parameter estimate "Inf_Age".

You can specify the following *options* in the ZEROMODEL statement after a slash (/):

**LINK=LOGISTIC | NORMAL**
specifies the distribution function used to compute probability of zeros. The supported distribution functions are as follows:

**LOGISTIC** specifies logistic distribution.

**NORMAL** specifies standard normal distribution.

If this option is omitted, then the default ZI link function is logistic.

**OFFSET=***zero-inflated-offset-variable*
specifies a variable in the input data set to be used as a zero-inflated (ZI) offset variable. The ZI offset variable *zero-inflated-offset-variable* is included as a term, with coefficient restricted to 1, in the equation that determines the probability ($\varphi_i$) of a zero count and represents an adjustment to a common observational unit. The ZI offset variable cannot be the response variable, the offset variable (if any), or any of the explanatory variables. The name of the data set variable that is used as the ZI offset variable is displayed in the "Model Fit Summary" table, where it is labeled as "Inf_offset".

# Details: HPCOUNTREG Procedure

## Missing Values

Any observations in the input data set that have a missing value for one or more of the regressors are ignored by PROC HPCOUNTREG and not used in the model fit. PROC HPCOUNTREG rounds any positive noninteger count values to the nearest integer and ignores any observations that have a negative count.

If the input data set contains any observations that have missing response values but nonmissing regressors, PROC HPCOUNTREG can compute several statistics and store them in an output data set by using the OUTPUT statement. For example, you can request that the output data set contain the estimates of $\mathbf{x}_i'\boldsymbol{\beta}$, the expected value of the response variable, and the probability that the response variable will take the current value. Furthermore, if a zero-inflated model was fit, you can request that the output data set contain the estimates of $\mathbf{z}_i'\boldsymbol{\gamma}$, and the probability that the response is 0 as a result of the zero-generating process. Note that the presence of such observations (that have missing response values) does not affect the model fit.

## Poisson Regression

The most widely used model for count data analysis is Poisson regression. Poisson regression assumes that $y_i$, given the vector of covariates $\mathbf{x}_i$, is independently Poisson distributed with

$$P(Y_i = y_i|\mathbf{x}_i) = \frac{e^{-\mu_i}\mu_i^{y_i}}{y_i!}, \quad y_i = 0, 1, 2, \ldots$$

and the mean parameter—that is, the mean number of events per period—is given by

$$\mu_i = \exp(\mathbf{x}_i'\boldsymbol{\beta})$$

where $\boldsymbol{\beta}$ is a $(k+1) \times 1$ parameter vector. (The intercept is $\beta_0$; the coefficients for the $k$ regressors are $\beta_1, \ldots, \beta_k$.) Taking the exponential of $\mathbf{x}_i'\boldsymbol{\beta}$ ensures that the mean parameter $\mu_i$ is nonnegative. It can be shown that the conditional mean is given by

$$E(y_i|\mathbf{x}_i) = \mu_i = \exp(\mathbf{x}_i'\boldsymbol{\beta})$$

Note that the conditional variance of the count random variable is equal to the conditional mean in the Poisson regression model:

$$V(y_i|\mathbf{x}_i) = E(y_i|\mathbf{x}_i) = \mu_i$$

The equality of the conditional mean and variance of $y_i$ is known as *equidispersion*.

The standard estimator for the Poisson model is the maximum likelihood estimator (MLE). Because the observations are independent, the log-likelihood function is written as

$$\mathcal{L} = \sum_{i=1}^{N}(-\mu_i + y_i \ln \mu_i - \ln y_i!) = \sum_{i=1}^{N}(-e^{\mathbf{x}_i'\boldsymbol{\beta}} + y_i\mathbf{x}_i'\boldsymbol{\beta} - \ln y_i!)$$

For more information about the Poisson regression model, see the section "Poisson Regression" (Chapter 11, *SAS/ETS User's Guide*).

The Poisson model has been criticized for its restrictive property that the conditional variance equals the conditional mean. Real-life data are often characterized by *overdispersion*—that is, the variance exceeds the mean. Allowing for overdispersion can improve model predictions because the Poisson restriction of equal mean and variance results in the underprediction of zeros when overdispersion exists. The most commonly used model that accounts for overdispersion is the negative binomial model.

## Negative Binomial Regression

The Poisson regression model can be generalized by introducing an unobserved heterogeneity term for observation $i$. Thus, the individuals are assumed to differ randomly in a manner that is not fully accounted for by the observed covariates. This is formulated as

$$E(y_i|\mathbf{x}_i, \tau_i) = \mu_i \tau_i = e^{\mathbf{x}_i'\boldsymbol{\beta}+\epsilon_i}$$

where the unobserved heterogeneity term $\tau_i = e^{\epsilon_i}$ is independent of the vector of regressors $\mathbf{x}_i$. Then the distribution of $y_i$ conditional on $\mathbf{x}_i$ and $\tau_i$ is Poisson with conditional mean and conditional variance $\mu_i \tau_i$:

$$f(y_i|\mathbf{x}_i, \tau_i) = \frac{\exp(-\mu_i \tau_i)(\mu_i \tau_i)^{y_i}}{y_i!}$$

Let $g(\tau_i)$ be the probability density function of $\tau_i$. Then, the distribution $f(y_i|\mathbf{x}_i)$ (no longer conditional on $\tau_i$) is obtained by integrating $f(y_i|\mathbf{x}_i, \tau_i)$ with respect to $\tau_i$:

$$f(y_i|\mathbf{x}_i) = \int_0^\infty f(y_i|\mathbf{x}_i, \tau_i) g(\tau_i) d\tau_i$$

An analytical solution to this integral exists when $\tau_i$ is assumed to follow a gamma distribution. This solution is the negative binomial distribution. If the model contains a constant term, then in order to identify the mean of the distribution, it is necessary to assume that $E(e^{\epsilon_i}) = E(\tau_i) = 1$. Thus, it is assumed that $\tau_i$ follows a gamma$(\theta, \theta)$ distribution with $E(\tau_i) = 1$ and $V(\tau_i) = 1/\theta$,

$$g(\tau_i) = \frac{\theta^\theta}{\Gamma(\theta)} \tau_i^{\theta-1} \exp(-\theta \tau_i)$$

where $\Gamma(x) = \int_0^\infty z^{x-1} \exp(-z) dz$ is the gamma function and $\theta$ is a positive parameter. Then, the density of $y_i$ given $\mathbf{x}_i$ is derived as

$$\begin{aligned}
f(y_i|\mathbf{x}_i) &= \int_0^\infty f(y_i|\mathbf{x}_i, \tau_i) g(\tau_i) d\tau_i \\
&= \frac{\theta^\theta \mu_i^{y_i}}{y_i! \Gamma(\theta)} \int_0^\infty e^{-(\mu_i+\theta)\tau_i} \tau_i^{\theta+y_i-1} d\tau_i \\
&= \frac{\theta^\theta \mu_i^{y_i} \Gamma(y_i+\theta)}{y_i! \Gamma(\theta)(\theta+\mu_i)^{\theta+y_i}} \\
&= \frac{\Gamma(y_i+\theta)}{y_i! \Gamma(\theta)} \left(\frac{\theta}{\theta+\mu_i}\right)^\theta \left(\frac{\mu_i}{\theta+\mu_i}\right)^{y_i}
\end{aligned}$$

If you make the substitution $\alpha = \frac{1}{\theta}$ $(\alpha > 0)$, the negative binomial distribution can then be rewritten as

$$f(y_i|\mathbf{x}_i) = \frac{\Gamma(y_i+\alpha^{-1})}{y_i! \Gamma(\alpha^{-1})} \left(\frac{\alpha^{-1}}{\alpha^{-1}+\mu_i}\right)^{\alpha^{-1}} \left(\frac{\mu_i}{\alpha^{-1}+\mu_i}\right)^{y_i}, \quad y_i = 0, 1, 2, \ldots$$

Thus, the negative binomial distribution is derived as a gamma mixture of Poisson random variables. It has the conditional mean

$$E(y_i|\mathbf{x}_i) = \mu_i = e^{\mathbf{x}_i'\boldsymbol{\beta}}$$

and the conditional variance

$$V(y_i|\mathbf{x}_i) = \mu_i[1 + \frac{1}{\theta}\mu_i] = \mu_i[1 + \alpha\mu_i] > E(y_i|\mathbf{x}_i)$$

The conditional variance of the negative binomial distribution exceeds the conditional mean. Overdispersion results from neglected unobserved heterogeneity. The negative binomial model with variance function $V(y_i|\mathbf{x}_i) = \mu_i + \alpha\mu_i^2$, which is quadratic in the mean, is referred to as the NEGBIN2 model (Cameron and Trivedi 1986). To estimate this model, specify DIST=NEGBIN(P=2) in the MODEL statement. The Poisson distribution is a special case of the negative binomial distribution where $\alpha = 0$. A test of the Poisson distribution can be carried out by testing the hypothesis that $\alpha = \frac{1}{\theta_i} = 0$. A Wald test of this hypothesis is provided (it is the reported $t$ statistic for the estimated $\alpha$ in the negative binomial model).

The log-likelihood function of the negative binomial regression model (NEGBIN2) is given by

$$\begin{aligned}\mathcal{L} \quad = \quad & \sum_{i=1}^{N}\left\{\sum_{j=0}^{y_i-1}\ln(j+\alpha^{-1}) - \ln(y_i!)\right. \\ & \left. -(y_i+\alpha^{-1})\ln(1+\alpha\exp(\mathbf{x}_i'\boldsymbol{\beta})) + y_i\ln(\alpha) + y_i\mathbf{x}_i'\boldsymbol{\beta}\right\}\end{aligned}$$

where use of the following fact is made if $y$ is an integer:

$$\Gamma(y+a)/\Gamma(a) = \prod_{j=0}^{y-1}(j+a)$$

Cameron and Trivedi (1986) consider a general class of negative binomial models that have mean $\mu_i$ and variance function $\mu_i + \alpha\mu_i^p$. The NEGBIN2 model, with $p = 2$, is the standard formulation of the negative binomial model. Models that have other values of $p$, $-\infty < p < \infty$, have the same density $f(y_i|\mathbf{x}_i)$, except that $\alpha^{-1}$ is replaced everywhere by $\alpha^{-1}\mu^{2-p}$. The negative binomial model NEGBIN1, which sets $p = 1$, has the variance function $V(y_i|\mathbf{x}_i) = \mu_i + \alpha\mu_i$, which is linear in the mean. To estimate this model, specify DIST=NEGBIN(P=1) in the MODEL statement.

The log-likelihood function of the NEGBIN1 regression model is given by

$$\begin{aligned}\mathcal{L} \quad = \quad & \sum_{i=1}^{N}\left\{\sum_{j=0}^{y_i-1}\ln\left(j+\alpha^{-1}\exp(\mathbf{x}_i'\boldsymbol{\beta})\right)\right. \\ & \left. -\ln(y_i!) - \left(y_i+\alpha^{-1}\exp(\mathbf{x}_i'\boldsymbol{\beta})\right)\ln(1+\alpha) + y_i\ln(\alpha)\right\}\end{aligned}$$

For more information about the negative binomial regression model, see the section "Negative Binomial Regression" (Chapter 11, *SAS/ETS User's Guide*).

## Zero-Inflated Count Regression Overview

The main motivation for using zero-inflated count models is that real-life data frequently display overdispersion and excess zeros. Zero-inflated count models provide a way to both model the excess zeros and allow for overdispersion. In particular, there are two possible data generation processes for each observation. The result of a Bernoulli trial is used to determine which of the two processes to use. For observation $i$, Process 1 is chosen with probability $\varphi_i$ and Process 2 with probability $1 - \varphi_i$. Process 1 generates only zero counts. Process 2 generates counts from either a Poisson or a negative binomial model. In general,

$$y_i \sim \begin{cases} 0 & \text{with probability} \quad \varphi_i \\ g(y_i) & \text{with probability} \quad 1 - \varphi_i \end{cases}$$

Therefore, the probability of $\{Y_i = y_i\}$ can be described as

$$\begin{aligned} P(y_i = 0|\mathbf{x}_i) &= \varphi_i + (1 - \varphi_i)g(0) \\ P(y_i|\mathbf{x}_i) &= (1 - \varphi_i)g(y_i), \quad y_i > 0 \end{aligned}$$

where $g(y_i)$ follows either the Poisson or the negative binomial distribution.

If the probability $\varphi_i$ depends on the characteristics of observation $i$, then $\varphi_i$ is written as a function of $\mathbf{z}_i'\boldsymbol{\gamma}$, where $\mathbf{z}_i'$ is the $1 \times (q + 1)$ vector of zero-inflated covariates and $\boldsymbol{\gamma}$ is the $(q + 1) \times 1$ vector of zero-inflated coefficients to be estimated. (The zero-inflated intercept is $\gamma_0$; the coefficients for the $q$ zero-inflated covariates are $\gamma_1, \ldots, \gamma_q$.) The function $F$ that relates the product $\mathbf{z}_i'\boldsymbol{\gamma}$ (which is a scalar) to the probability $\varphi_i$ is called the zero-inflated link function,

$$\varphi_i = F_i = F(\mathbf{z}_i'\boldsymbol{\gamma})$$

In the HPCOUNTREG procedure, the zero-inflated covariates are indicated in the ZEROMODEL statement. Furthermore, the zero-inflated link function $F$ can be specified as either the logistic function,

$$F(\mathbf{z}_i'\boldsymbol{\gamma}) = \Lambda(\mathbf{z}_i'\boldsymbol{\gamma}) = \frac{\exp(\mathbf{z}_i'\boldsymbol{\gamma})}{1 + \exp(\mathbf{z}_i'\boldsymbol{\gamma})}$$

or the standard normal cumulative distribution function (also called the probit function),

$$F(\mathbf{z}_i'\boldsymbol{\gamma}) = \Phi(\mathbf{z}_i'\boldsymbol{\gamma}) = \int_0^{\mathbf{z}_i'\boldsymbol{\gamma}} \frac{1}{\sqrt{2\pi}} \exp(-u^2/2)du$$

The zero-inflated link function is indicated by using the LINK= option in the ZEROMODEL statement. The default ZI link function is the logistic function.

## Zero-Inflated Poisson Regression

In the zero-inflated Poisson (ZIP) regression model, the data generation process that is referred to earlier as Process 2 is

$$g(y_i) = \frac{\exp(-\mu_i)\mu_i^{y_i}}{y_i!}$$

where $\mu_i = e^{\mathbf{x}_i'\boldsymbol{\beta}}$. Thus the ZIP model is defined as

$$\begin{aligned} P(y_i = 0|\mathbf{x}_i, \mathbf{z}_i) &= F_i + (1 - F_i)\exp(-\mu_i) \\ P(y_i|\mathbf{x}_i, \mathbf{z}_i) &= (1 - F_i)\frac{\exp(-\mu_i)\mu_i^{y_i}}{y_i!}, \quad y_i > 0 \end{aligned}$$

The conditional expectation and conditional variance of $y_i$ are given by

$$E(y_i|\mathbf{x}_i, \mathbf{z}_i) = \mu_i(1 - F_i)$$

$$V(y_i|\mathbf{x}_i, \mathbf{z}_i) = E(y_i|\mathbf{x}_i, \mathbf{z}_i)(1 + \mu_i F_i)$$

Note that the ZIP model (in addition to the ZINB model) exhibits overdispersion because $V(y_i|\mathbf{x}_i, \mathbf{z}_i) > E(y_i|\mathbf{x}_i, \mathbf{z}_i)$.

In general, the log-likelihood function of the ZIP model is

$$\mathcal{L} = \sum_{i=1}^{N} \ln\left[P(y_i|\mathbf{x}_i, \mathbf{z}_i)\right]$$

After a specific link function (either logistic or standard normal) for the probability $\varphi_i$ is chosen, it is possible to write the exact expressions for the log-likelihood function and the gradient.

## ZIP Model with Logistic Link Function

First, consider the ZIP model in which the probability $\varphi_i$ is expressed by a logistic link function, namely

$$\varphi_i = \frac{\exp(\mathbf{z}_i'\boldsymbol{\gamma})}{1 + \exp(\mathbf{z}_i'\boldsymbol{\gamma})}$$

The log-likelihood function is

$$\begin{aligned} \mathcal{L} = & \sum_{\{i:y_i=0\}} \ln\left[\exp(\mathbf{z}_i'\boldsymbol{\gamma}) + \exp(-\exp(\mathbf{x}_i'\boldsymbol{\beta}))\right] \\ & + \sum_{\{i:y_i>0\}} \left[y_i\mathbf{x}_i'\boldsymbol{\beta} - \exp(\mathbf{x}_i'\boldsymbol{\beta}) - \sum_{k=2}^{y_i} \ln(k)\right] \\ & - \sum_{i=1}^{N} \ln\left[1 + \exp(\mathbf{z}_i'\boldsymbol{\gamma})\right] \end{aligned}$$

### ZIP Model with Standard Normal Link Function

Next, consider the ZIP model in which the probability $\varphi_i$ is expressed by a standard normal link function: $\varphi_i = \Phi(\mathbf{z}_i'\boldsymbol{\gamma})$. The log-likelihood function is

$$\begin{aligned}\mathcal{L} &= \sum_{\{i:y_i=0\}} \ln\left\{\Phi(\mathbf{z}_i'\boldsymbol{\gamma}) + \left[1-\Phi(\mathbf{z}_i'\boldsymbol{\gamma})\right]\exp(-\exp(\mathbf{x}_i'\boldsymbol{\beta}))\right\} \\ &+ \sum_{\{i:y_i>0\}} \left\{\ln\left[\left(1-\Phi(\mathbf{z}_i'\boldsymbol{\gamma})\right)\right] - \exp(\mathbf{x}_i'\boldsymbol{\beta}) + y_i\mathbf{x}_i'\boldsymbol{\beta} - \sum_{k=2}^{y_i}\ln(k)\right\}\end{aligned}$$

For more information about the zero-inflated Poisson regression model, see the section "Zero-Inflated Poisson Regression" (Chapter 11, *SAS/ETS User's Guide*).

## Zero-Inflated Negative Binomial Regression

The zero-inflated negative binomial (ZINB) model in PROC HPCOUNTREG is based on the negative binomial model that has a quadratic variance function (when DIST=NEGBIN in the MODEL or PROC HPCOUNTREG statement). The ZINB model is obtained by specifying a negative binomial distribution for the data generation process referred to earlier as Process 2:

$$g(y_i) = \frac{\Gamma(y_i+\alpha^{-1})}{y_i!\Gamma(\alpha^{-1})}\left(\frac{\alpha^{-1}}{\alpha^{-1}+\mu_i}\right)^{\alpha^{-1}}\left(\frac{\mu_i}{\alpha^{-1}+\mu_i}\right)^{y_i}$$

Thus the ZINB model is defined to be

$$\begin{aligned}P(y_i=0|\mathbf{x}_i,\mathbf{z}_i) &= F_i + (1-F_i)\,(1+\alpha\mu_i)^{-\alpha^{-1}} \\ P(y_i|\mathbf{x}_i,\mathbf{z}_i) &= (1-F_i)\,\frac{\Gamma(y_i+\alpha^{-1})}{y_i!\Gamma(\alpha^{-1})}\left(\frac{\alpha^{-1}}{\alpha^{-1}+\mu_i}\right)^{\alpha^{-1}} \\ &\times \left(\frac{\mu_i}{\alpha^{-1}+\mu_i}\right)^{y_i}, \quad y_i>0\end{aligned}$$

In this case, the conditional expectation ($E$) and conditional variance ($V$) of $y_i$ are

$$E(y_i|\mathbf{x}_i,\mathbf{z}_i) = \mu_i(1-F_i)$$

$$V(y_i|\mathbf{x}_i,\mathbf{z}_i) = E(y_i|\mathbf{x}_i,\mathbf{z}_i)\,[1+\mu_i(F_i+\alpha)]$$

Like the ZIP model, the ZINB model exhibits overdispersion because the conditional variance exceeds the conditional mean.

### ZINB Model with Logistic Link Function

In this model, the probability $\varphi_i$ is given by the logistic function, namely

$$\varphi_i = \frac{\exp(\mathbf{z}_i'\boldsymbol{\gamma})}{1+\exp(\mathbf{z}_i'\boldsymbol{\gamma})}$$

The log-likelihood function is

$$\begin{aligned}
\mathcal{L} &= \sum_{\{i:y_i=0\}} \ln\left[\exp(\mathbf{z}_i'\boldsymbol{\gamma}) + (1+\alpha\exp(\mathbf{x}_i'\boldsymbol{\beta}))^{-\alpha^{-1}}\right] \\
&+ \sum_{\{i:y_i>0\}} \sum_{j=0}^{y_i-1} \ln(j+\alpha^{-1}) \\
&+ \sum_{\{i:y_i>0\}} \left\{-\ln(y_i!) - (y_i+\alpha^{-1})\ln(1+\alpha\exp(\mathbf{x}_i'\boldsymbol{\beta})) + y_i\ln(\alpha) + y_i\mathbf{x}_i'\boldsymbol{\beta}\right\} \\
&- \sum_{i=1}^{N} \ln\left[1+\exp(\mathbf{z}_i'\boldsymbol{\gamma})\right]
\end{aligned}$$

### ZINB Model with Standard Normal Link Function

For this model, the probability $\varphi_i$ is expressed by the standard normal distribution function (probit function): $\varphi_i = \Phi(\mathbf{z}_i'\boldsymbol{\gamma})$. The log-likelihood function is

$$\begin{aligned}
\mathcal{L} &= \sum_{\{i:y_i=0\}} \ln\left\{\Phi(\mathbf{z}_i'\boldsymbol{\gamma}) + \left[1-\Phi(\mathbf{z}_i'\boldsymbol{\gamma})\right](1+\alpha\exp(\mathbf{x}_i'\boldsymbol{\beta}))^{-\alpha^{-1}}\right\} \\
&+ \sum_{\{i:y_i>0\}} \ln\left[1-\Phi(\mathbf{z}_i'\boldsymbol{\gamma})\right] \\
&+ \sum_{\{i:y_i>0\}} \sum_{j=0}^{y_i-1} \left\{\ln(j+\alpha^{-1})\right\} \\
&- \sum_{\{i:y_i>0\}} \ln(y_i!) \\
&- \sum_{\{i:y_i>0\}} (y_i+\alpha^{-1})\ln(1+\alpha\exp(\mathbf{x}_i'\boldsymbol{\beta})) \\
&+ \sum_{\{i:y_i>0\}} y_i\ln(\alpha) \\
&+ \sum_{\{i:y_i>0\}} y_i\mathbf{x}_i'\boldsymbol{\beta}
\end{aligned}$$

For more information about the zero-inflated negative binomial regression model, see the section "Zero-Inflated Negative Binomial Regression" (Chapter 11, *SAS/ETS User's Guide*).

## Computational Resources

The time and memory that PROC HPCOUNTREG requires are proportional to the number of parameters in the model and the number of observations in the data set being analyzed. Less time and memory are required for smaller models and fewer observations. When PROC HPCOUNTREG is run in the high-performance distributed environment, the amount of time required is also affected by the number of nodes and the number of threads per node as specified in the PERFORMANCE statement.

The method that is chosen to calculate the variance-covariance matrix and the optimization method also affect the time and memory resources. All optimization methods available through the METHOD= option have similar memory use requirements. The processing time might differ for each method, depending on the number of iterations and functional calls needed. The data set is read into memory to save processing time. If not enough memory is available to hold the data, the HPCOUNTREG procedure stores the data in a utility file on disk and rereads the data as needed from this file, substantially increasing the execution time of the procedure. The gradient and the variance-covariance matrix must be held in memory. If the model has $p$ parameters including the intercept, then at least $8 * (p + p * (p + 1)/2)$ bytes of memory are needed. The processing time is also a function of the number of iterations needed to converge to a solution for the model parameters. The number of iterations that are needed cannot be known in advance. You can use the MAXITER= option to limit the number of iterations that PROC HPCOUNTREG executes. You can alter the convergence criteria by using the nonlinear optimization options available in the PROC HPCOUNTREG statement. For a list of all the nonlinear optimization options, see "Optimization Control Options" on page 131.

## Covariance Matrix Types

The COVEST= option in the PROC HPCOUNTREG statement enables you to specify the estimation method for the covariance matrix. COVEST=HESSIAN estimates the covariance matrix that is based on the inverse of the Hessian matrix; COVEST=OP uses the outer product of gradients; and COVEST=QML produces the covariance matrix that is based on both the Hessian and outer product matrices. Although all three methods produce asymptotically equivalent results, they differ in computational intensity and produce results that might differ in finite samples. The COVEST=OP option provides the covariance matrix that is typically the easiest to compute. In some cases, the OP approximation is considered more efficient than the Hessian or QML approximation because it contains fewer random elements. The QML approximation is computationally the most complex because it requires both the outer product of gradients and the Hessian matrix. In most cases, the OP or Hessian approximation is preferred to QML. The need for QML approximation arises in cases where the model is misspecified and the information matrix equality does not hold. The default is COVEST=HESSIAN.

## Displayed Output

PROC HPCOUNTREG produces the following displayed output.

## Model Fit Summary

The "Model Fit Summary" table contains the following information:

- dependent (count) variable name
- number of observations used
- number of missing values in data set, if any
- data set name
- type of model that was fit
- offset variable name, if any
- zero-inflated link function, if any
- zero-inflated offset variable name, if any
- log-likelihood value at solution
- maximum absolute gradient at solution
- number of iterations
- AIC value at solution (smaller value indicates better fit)
- SBC value at solution (smaller value indicates better fit)

A line in the "Model Fit Summary" table indicates whether the algorithm successfully converged.

## Parameter Estimates

The "Parameter Estimates" table in the displayed output gives the estimates for the ZI intercept and ZI explanatory variables; they are labeled with the prefix "Inf_". For example, the ZI intercept is labeled "Inf_intercept". If you specify "Age" (a variable in your data set) as a ZI explanatory variable, then the "Parameter Estimates" table labels the corresponding parameter estimate "Inf_Age". If you do not list any ZI explanatory variables (for the ZI option VAR=), then only the intercept term is estimated.

"_Alpha" is the negative binomial dispersion parameter. The $t$ statistic that is given for "_Alpha" is a test of overdispersion.

## Covariance of Parameter Estimates

If you specify the COVB option in the PROC HPCOUNTREG or MODEL statement, the HPCOUNTREG procedure displays the estimated covariance matrix, which is defined as the inverse of the information matrix at the final iteration.

### Correlation of Parameter Estimates

If you specify the CORRB option in the PROC HPCOUNTREG or MODEL statement, the HPCOUNTREG procedure displays the estimated correlation matrix, which is based on the Hessian matrix used at the final iteration.

## OUTPUT OUT= Data Set

The OUTPUT statement creates a new SAS data set that contains various estimates that you specify. You can request that the output data set contain the estimates of $\mathbf{x}_i'\boldsymbol{\beta}$, the expected value of the response variable, and the probability that the response variable will take the current value. Furthermore, if a zero-inflated model is fit, you can request that the output data set contain the estimates of $\mathbf{z}_i'\boldsymbol{\gamma}$ and the probability that the response is 0 as a result of the zero-generating process. These statistics can be computed for all observations in which the regressors are not missing, even if the response is missing. By adding observations with missing response values to the input data set, you can compute these statistics for new observations or for settings of the regressors that are not present in the data without affecting the model fit. Because of potential space limitations on the client workstation, the data set that is created by the OUTPUT statement does not contain the variables in the input data set.

## OUTEST= Data Set

The OUTEST= data set is made up of at least two rows: the first row (with _TYPE_='PARM') contains each of the parameter estimates in the model, and the second row (with _TYPE_='STD') contains the standard errors for the parameter estimates in the model.

If you use the COVOUT option in the PROC HPCOUNTREG statement, the OUTEST= data set also contains the covariance matrix for the parameter estimates. The covariance matrix appears in the observations with _TYPE_='COV', and the _NAME_ variable labels the rows with the parameter names.

## ODS Table Names

PROC HPCOUNTREG assigns a name to each table that it creates. You can use these names to denote the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These table names are listed in Table 6.2.

**Table 6.2** ODS Tables Produced in PROC HPCOUNTREG

| ODS Table Name | Description | Option |
|---|---|---|
| **ODS Tables Created by the MODEL Statement** | | |
| FitSummary | Summary of nonlinear estimation | Default |
| ConvergenceStatus | Convergence status | Default |
| ParameterEstimates | Parameter estimates | Default |
| CovB | Covariance of parameter estimates | COVB |
| CorrB | Correlation of parameter estimates | CORRB |

# Examples: The HPCOUNTREG Procedure

## Example 6.1: High-Performance Zero-Inflated Poisson Model

This example shows the use of the HPCOUNTREG procedure with an emphasis on large data set processing and the performance improvements that are achieved by executing in the high-performance distributed environment.

The following DATA step generates one million replicates from the zero-inflated Poisson (ZIP) model. The model contains seven variables and three variables that correspond to the zero-inflated process.

```
data simulate;
   call streaminit(12345);
   array vars x1-x7;
   array zero_vars z1-z3;

   array parms{7}  (.3 .4 .2 .4 -.3 -.5 -.3);
   array zero_parms{3} (-.6 .3 .2);

   intercept=2;
   z_intercept=-1;
   theta=0.5;

   do i=1 to 1000000;
      sum_xb=0;
      sum_gz=0;
      do j=1 to 7;
         vars[j]=rand('NORMAL',0,1);
         sum_xb=sum_xb+parms[j]*vars[j];
      end;
      mu=exp(intercept+sum_xb);
      y_p=rand('POISSON', mu);

      do j=1 to 3;
         zero_vars[j]=rand('NORMAL',0,1);
         sum_gz = sum_gz+zero_parms[j]*zero_vars[j];
      end;
      z_gamma = z_intercept+sum_gz;
      pzero = cdf('LOGISTIC',z_gamma);
      cut=rand('UNIFORM');
      if cut<pzero then y_p=0;
      output;
   end;
keep y_p x1-x7 z1-z3;
run;
```

The following statements estimate a zero-inflated Poisson model.

```
option set=GRIDHOST="&GRIDHOST";
option set=GRIDINSTALLLOC="&GRIDINSTALLLOC";

proc hpcountreg data=simulate dist=zip;
   performance nthreads=2 nodes=1 details
       host="&GRIDHOST" install="&GRIDINSTALLLOC";
   model y_p=x1-x7;
   zeromodel y_p ~ z1-z3;
run;
```

The model is executed in the distributed computing environment on two threads and only one node. These settings are used to obtain a hypothetical environment that might resemble running the HPCOUNTREG procedure on a desktop workstation with a dual-core CPU. To run these statements successfully, you need to set the macro variables GRIDHOST and GRIDINSTALLLOC to resolve to appropriate values, or you can replace the references to the macro variables in the example with the appropriate values. Output 6.1.1 shows the "Performance Information" table for this hypothetical scenario.

**Output 6.1.1** Performance Information with One Node and One Thread

| Performance Information | |
|---|---|
| **Host Node** | << your grid host >> |
| **Install Location** | /opt/v940m2/laxno/TKGrid |
| **Execution Mode** | Distributed |
| **Number of Compute Nodes** | 1 |
| **Number of Threads per Node** | 2 |

Output 6.1.2 shows the results for the zero-inflated Poisson model. The "Model Fit Summary" table shows detailed information about the model and indicates that all one million observations were used to fit the model. All parameter estimates in the "Parameter Estimates" table are highly significant and correspond to their theoretical values set during the data generating process. The optimization of the model that contains one million observations took 42.57 seconds.

**Output 6.1.2** Zero-Inflated Poisson Model Execution on One Node and Two Threads

| Model Fit Summary | |
|---|---|
| **Dependent Variable** | y_p |
| **Number of Observations** | 1000000 |
| **Data Set** | WORK.SIMULATE |
| **Model** | ZIP |
| **ZI Link Function** | Logistic |
| **Log Likelihood** | -2215238 |
| **Maximum Absolute Gradient** | 2.0586E-8 |
| **Number of Iterations** | 7 |
| **Optimization Method** | Newton-Raphson |
| **AIC** | 4430500 |
| **SBC** | 4430642 |

Convergence criterion (FCONV=2.220446E-16) satisfied.

**Output 6.1.2** *continued*

| Parameter Estimates | | | | | |
|---|---|---|---|---|---|
| Parameter | DF | Estimate | Standard Error | t Value | Pr > \|t\| |
| Intercept | 1 | 2.0005 | 0.000492 | 4069.80 | <.0001 |
| x1 | 1 | 0.2995 | 0.000352 | 850.17 | <.0001 |
| x2 | 1 | 0.3998 | 0.000353 | 1132.23 | <.0001 |
| x3 | 1 | 0.2008 | 0.000352 | 570.27 | <.0001 |
| x4 | 1 | 0.3994 | 0.000353 | 1132.85 | <.0001 |
| x5 | 1 | -0.2995 | 0.000353 | -848.95 | <.0001 |
| x6 | 1 | -0.5000 | 0.000353 | -1414.9 | <.0001 |
| x7 | 1 | -0.3002 | 0.000352 | -852.14 | <.0001 |
| Inf_Intercept | 1 | -0.9993 | 0.002521 | -396.45 | <.0001 |
| Inf_z1 | 1 | -0.6024 | 0.002585 | -233.02 | <.0001 |
| Inf_z2 | 1 | 0.2976 | 0.002454 | 121.25 | <.0001 |
| Inf_z3 | 1 | 0.1974 | 0.002430 | 81.20 | <.0001 |

| Procedure Task Timing | | |
|---|---|---|
| Task | Seconds | Percent |
| Reading and Levelizing Data | 0.70 | 1.60% |
| Communication to Client | 0.08 | 0.19% |
| Optimization | 42.57 | 97.90% |
| Post-Optimization | 0.14 | 0.31% |

In the following statements, the PERFORMANCE statement is modified to use a grid with 10 nodes, with each node capable of spawning eight threads:

```
proc hpcountreg data=simulate dist=zip;
   performance nthreads=8 nodes=10 details
               host="&GRIDHOST" install="&GRIDINSTALLLOC";
   model y_p=x1-x7;
   zeromodel y_p ~ z1-z3;
run;
```

Because the two models being estimated are identical, it is reasonable to expect that Output 6.1.2 and Output 6.1.3 would show the same results. However, you can see a significant difference in performance between the two models. The second model, which was run on a grid that used 10 nodes with eight threads each, took only 1.69 seconds instead of 42.57 seconds to optimize.

In certain circumstances, you might observe slight numerical differences in the results, depending on the number of nodes and threads involved. This happens because the order in which partial results are accumulated can make a difference in the final result, owing to the limits of numerical precision and the propagation of error in numerical computations.

**Output 6.1.3** Zero-Inflated Poisson Model Execution on 10 Nodes with Eight Threads Each

**The HPCOUNTREG Procedure**

| Model Fit Summary | |
|---|---|
| **Dependent Variable** | y_p |
| **Number of Observations** | 1000000 |
| **Data Set** | WORK.SIMULATE |
| **Model** | ZIP |
| **ZI Link Function** | Logistic |
| **Log Likelihood** | -2215238 |
| **Maximum Absolute Gradient** | 2.0608E-8 |
| **Number of Iterations** | 7 |
| **Optimization Method** | Newton-Raphson |
| **AIC** | 4430500 |
| **SBC** | 4430642 |

Convergence criterion (FCONV=2.220446E-16) satisfied.

**Parameter Estimates**

| Parameter | DF | Estimate | Standard Error | t Value | Pr > \|t\| |
|---|---|---|---|---|---|
| **Intercept** | 1 | 2.0005 | 0.000492 | 4069.80 | <.0001 |
| **x1** | 1 | 0.2995 | 0.000352 | 850.17 | <.0001 |
| **x2** | 1 | 0.3998 | 0.000353 | 1132.23 | <.0001 |
| **x3** | 1 | 0.2008 | 0.000352 | 570.27 | <.0001 |
| **x4** | 1 | 0.3994 | 0.000353 | 1132.85 | <.0001 |
| **x5** | 1 | -0.2995 | 0.000353 | -848.95 | <.0001 |
| **x6** | 1 | -0.5000 | 0.000353 | -1414.9 | <.0001 |
| **x7** | 1 | -0.3002 | 0.000352 | -852.14 | <.0001 |
| **Inf_Intercept** | 1 | -0.9993 | 0.002521 | -396.45 | <.0001 |
| **Inf_z1** | 1 | -0.6024 | 0.002585 | -233.02 | <.0001 |
| **Inf_z2** | 1 | 0.2976 | 0.002454 | 121.25 | <.0001 |
| **Inf_z3** | 1 | 0.1974 | 0.002430 | 81.20 | <.0001 |

**Procedure Task Timing**

| Task | Seconds | Percent |
|---|---|---|
| **Reading and Levelizing Data** | 0.02 | 1.18% |
| **Communication to Client** | 0.06 | 2.95% |
| **Optimization** | 1.69 | 86.82% |
| **Post-Optimization** | 0.18 | 9.05% |

As this example suggests, increasing the number of nodes and the number of threads per node improves performance significantly. When you use the parallelism afforded by a high-performance distributed environment, you can see an even more dramatic reduction in the time required for the optimization as the number of observations in the data set increases. When the data set is extremely large, the computations might not even be possible in some cases, given the typical memory resources and computational constraints of a desktop computer. Under such circumstances the high-performance distributed environment becomes a necessity.

## References

Cameron, A. C. and Trivedi, P. K. (1986), "Econometric Models Based on Count Data: Comparisons and Applications of Some Estimators and Some Tests," *Journal of Applied Econometrics*, 1, 29–53.

Cameron, A. C. and Trivedi, P. K. (1998), *Regression Analysis of Count Data*, Cambridge: Cambridge University Press.

LaMotte, L. R. (1994), "A Note on the Role of Independence in *t* Statistics Constructed from Linear Statistics in Regression Models," *The American Statistician*, 48, 238–240.

Long, J. S. (1997), *Regression Models for Categorical and Limited Dependent Variables*, Thousand Oaks, CA: Sage Publications.

# Subject Index

# Syntax Index