SLAC-PUB-5687
October 1991
(A/I)

History Data Facility in the SLC Control System

Ralph G. Johnson   Gregory R. White

Stanford Linear Accelerator Center, Stanford University, Stanford CA 94305 *

Abstract

Two major enhancements to the SLC History Data Facility [1] are described separately. First the internal design and procedures used for saving and using long term history data. Second the user interface, facilities and application of the History Data Comparisons sub-system, which is used for analyzing and correlating two or more accelerator device histories.

Overview and Rationale

The history data facility of the Stanford Linear Collider (SLC) records device and feedback loop parameter values. We use the term 'device' to mean any machine unit, such as a monitoring instrument, magnet, power supply, etc. A 'parameter' of a device then, is the current value of one of its properties. Most devices report more than one parameter. A typical example would be a Toroid device for which an important parameter is its detected beam current.

Extensive software has been developed for displaying recorded history data. Much can be learned from looking at the locus of any one machine parameter over time, such as effects predicted by diurnal variation, or the correspondence of a device with a particular accelerator configuration. The comparison of two or more parameters can be used for failure diagnosis, machine tuning, experimentation and the like.

The parameters of any device which is defined in the control system database, may be saved. Currently we are recording over 23,000 different device parameters at a typical interval of 6 minutes. New developments to support historical correlations and long term analyses is presented. Our objective is to promote the use of historical data in analyzing machine characteristics for increasing beam luminosity.

Long Term Histories

As outlined in a preceeding paper [1], a data collection process saves data in a ring buffers - the 'Daily' data files. Each time that process is executed it reads and saves the values for a set of device parameters. File activity is minimized by sequentially saving data in a single buffer. Data in these files covers a period of one day.

We now also have history files which contain data for one week. These 'Weekly' files are organized by device. That is, there is a separate ring buffer for each device parameter. All ring buffers are of the same length. There is also a single ring buffer containing the time-stamps of all the recorded data. For any given time-stamp, the value of each parameter at that time is located at the same offset within each data ring buffer. This format permits easy data management and efficient data retrieval.

Shortly after midnight each day, the data in each Daily file is translated to the Weekly format. Thus we have one process saving data every 2 to 6 minutes and another moving that data to week long ring buffers once each day.

Archiving

Since the Weekly data files are ring buffered, data older than a week is lost each midnight. To provide long term data saving there is another set of files named 'Archive' (also commonly called 'Yearly' files). These are of the same internal format as the Weekly files. Once each week, or whenever needed, a process makes a snapshot copy of each of the current Weekly files. Each of these is given a name which includes the date. To conserve disk space this version is not a complete copy; rather we preserve only every nth data point, in order to achieve a desired file size. Although each history file could be compressed by a different ratio, all are currently compressed by 10:1. The primary consideration has been disk space.

As a consequence of our normal disk backup to tape, these 'Archive' files are preserved in secure storage. At some point we could delete older files from the disk as we would still have that data on tape. We have not yet implemented a procedure or software for autonomously retrieving any data from the backup. However, one could use the standard system file restore procedures for any file, whereupon it would then be available automatically.

Consequently, we always have the past week's history in "fine structure" form (at most every 6 minutes) - used typically for diagnosing current problems; and older data in "coarse structure" (each hour) for looking at trends and past setpoint values.

Data Retrieval

We have attempted to minimize the total disk accesses required to retrieve all the parameter data for a given time span. First, any data available in the Weekly file is gath-

*Work supported by the Department of Energy, contract DE-AC03-76SF00515

Contributed to the IEEE Particle Accelerator Conference, San Francisco, CA, May 6-9, 1991

-ered. Then, if more recent data is also needed, it is obtained from the Daily file by a request to the history process. If older data is needed, the list of available Archive files is obtained. Since the names of these files includes the date of the last data point in each we can then sequentially retrieve data only from those files pertinent to the requested time span. Also, by first using all available Weekly data we insure the presentation of data with the greatest number of values within the requested time period. As each request for new data is processed, any data already available in memory from prior requests is utilized, again minimizing file accesses.

Correlating Parameters over Time

This section describes the History Data Comparisons subsystem of the History Data facility, used to plot and analyze the recorded parameter values of two or more devices. The analyses fall roughly into two classes; *qualitative* - by-eye examinations of the loci of some devices, and *quantitative* - correlation charts, statistics and expressions.

This distinction is interesting when the user community of each class and their respective applications is considered. First we describe the interface and displays available to the users, and then consider the applications to which various users put them.

User Input Interface

The interface to the History Data facility is via the SLC Control Program (SCP) - a large multi-image software suite used to control most of the system wide functions necessary to run the SLC accelerator. Users navigate through the sub-systems of this program via a hierarchical arrangement of push-button 'panels' [5][these proceedings]. There are thirty or so panels in the SCP dealing with History Plots, each oriented toward one facility in the accelerator. From all of these, one can enter the History Buffer Comparisons panel.

With 23,000 or so devices available for inspection, some help is given to the user in selecting the device and parameter they are interested in. This help makes use of the fact that all devices and their parameters are identified by their database entry name, which is moderately formal. A device and each parameter under it, is uniquely identified by a tuple composed of four, or sometimes five, domains. Reading the tuple from left-to-right, each successive domain identifies the device more closely. The help system attempts to satisfy a partially instantiated device name and displays all the possibilities for the most significant missing domain. The user may assign up to 3 devices to *variable labels* A, B and C. All plots are referred to as plots of these variables, either against each other, or vs. time.

Default values for each domain are also maintained, chosen from consideration of the previous assignment of the variable, the assignment of the other variables, and any partial tuple already entered.

Users may also elect to assign to a variable label, a device they had examined when using one of the other History Data facilities.

Device names, however assigned, are reflected on the panel screen. The time range for which History Data Comparisons is to base its plots, is input from any of the panels dealing with the History Data facility as a whole. Once set, all plots for all devices are generated for that range, though the range can be changed at any time and the plot will be re-drawn.

All plots can be scaled manually or automatically. In general, when auto-scaling is selected, the entire range of data values gathered for the device will be charted and used in computing statistics. The function of manual-scaling is slightly different depending on whether a qualitative or quantitative plot is requested; see below.

Qualitative Plots

To compare devices over a common time range, their loci can be plotted either one-above-another (a 'Stripchart') or superimposed (an 'Overlay'). Manual-scaling for these plots refers simply to the upper and lower bounds of their ordinate axes. Major applications of qualitative plots have been in diagnosing errors and investigating unexpected machine behaviour, very much as an *ad hoc* problem solving tool.

Quantitative Plots

Quantitative charts explicitly treat the parameter values of the devices over time as a data vector. Before the vectors are used in expressions and, more significantly, before statistics are calculated on them, we make them congruent. That is, we make sure they all have the same number of data points, that they range over exactly the same time and that all corresponding data points refer to the same delta time from the stamp of the first point. This is achieved by linear interpolation, which is always performed assuming device variable A is the *independent*, and therefore B is "aligned" to it.

There are two groups of quantitative plots. Those plotting the locus of an arithmetic expression in the data vectors, vs time, named 'expression' plots; and one displaying the scatter-plot of the data vectors, named 'A vs B'.

At the time of writing only two expressions are available, being those requested by users to fulfil specific applications, discussed below. They are:

$$L[i] = A[i] + (r * B'[i]) \tag{1}$$

$$L[i] = A[i]/(r + B'[i]) \tag{2}$$

where L is the resultant vector, r is a real value entered from one of the buttons, and i ranges from 1 to the number of original data points taken for A (before, if applicable, A was 'bound'). For both expression plots and scatter plots, the user may elect to bind, or 'cut' the data value vectors

-before they're used. This they do by by manually scaling the device they wish to limit. In this case, original data vector elements outside the range will be replaced by linear interpolation between the previous and next elements whose values are inside the manual-scaling range. This is slightly different to the manual-scaling function when used for the Stripchart and Overlay plots.

In addition, the plot of the resultant locus of an expression plot can be manually scaled.

A significant portion of the systems analysis time was spent looking at anomalies arising from interpolations and their effect on statistics in the data. For instance, there may be periods when little or no data is recorded for one of the devices while the other is recorded normally. The exact effect of this on the plot and on the correlation statistics is different depending on whether it was the independent, or dependent variable for which data was 'missing '.We concluded that it was sufficient to draw attention to these effects in the help and documentation rather than employ more sophisticated data smoothing methods.

Correlation Statistics

The linear correlation coefficient (by Pearson) between A and B is displayed on the A vs B plot.

Pearson's r measures only the extent of correspondence between variables known to have a significant association. That is, it should not be used to decide whether two variables have an association in the first place. It is not yet clear whether users of History Data Comparisons are using the facility informally, to 'find' correlations.

Given some assumptions regarding r, for instance that many calculations of r, over a very long time frame, for the same variables, would yield a symmetric distribution curve, then we can for any one evaluation of r decide roughly how 'reliable' it is. This is quantified by a corrected Complimentary Error Function. That figure is also given with every chart.

Linear interpolation is used to bound the input data vectors on manual-scaling and "align" the vectors in preparation for a plot (this changes the data values most when significantly more data points are taken for one device than the other). All these will make the correlation coefficient figure artificially high.

Quantitative plots are used in machine tuning and accelerator physics experiments. For instance, the positron yield is a simple plot of the ratio of electrons onto the target and resultant positrons, and can be plotted for any one of the current monitoring devices down beam of the target.

Current Development

The simple expressions currently available were those requested by control room operators with specific applications in mind. We are now extending this function to plot 'arbitrarily' complex expressions on two or more device vectors. To do this we are using the Free Software Foundation's (GNU) implementations of two popular Unix[1] program development tools, Lex and Yacc. The GNU versions are named 'Flex' and 'Bison' [3] respectively. Lex is a lexical analyzer, which we use to pass the tokenized expression to a Yacc generated function which parses and evaluates the data. This is a recognized, virtually textbook application [2].

We are also studying the utility of non-linear methods in the normalization of data for the quantitative plots and giving more sophisticated statistics. Operators wishing to analyze the data in depth have the option to down-load the history data and look at it with their own tools. There has therefore been the argument that the on-line control system should provide only a minimum, intuitive, package. We are collecting comments from users before implementing, for instance, polynomial interpolation.

The History Data Facility in general is a major software component of the Control System for the SLC. It is continually upgraded and extended to accommodate tracking and analysis for new sub-systems in the SLC.

References

[1] Johnston, R. Stanford Linear Collider History Data Facility. *Proceedings of the 1989 IEEE Particle Accelerator Conference* 1989. 1716-1718

[2] Kernighan B. W. and Pike P. The Unix Programming Environment. Prentice-Hall. 1984. Chapter 8.

[3] Corbett R. and Stallman R. BISON The YACC-compatible Parser Generator. Free Software Foundation. 1988,1989,1990.

[4] Press W.H., Flannery B.P, Teukolsky S.A., Vetterling W.T. Numerical Recipes, The Art of Scientific Computing. Cambridge University Press. 1989.

[5] Flores M., Hendrickson L., Phinney N., Sanchez-Chopitea L. Correlation Plot Facility in the SLC Control System *Proceedings of the 1991 IEEE Particle Accelerator Conference* 1991.

[1] Unix is a trade mark of Bell Laboratories, ok