The United Nations University

GEOTHERMAL TRAINING PROGRAMME
Orkustofnun, Grensásvegur 9,
IS-108 Reykjavík, Iceland

Reports 2003
Number 4

# DATABASE SYSTEM AND APPLICATIONS DEVELOPED FOR RESERVOIR MODELLING AND MONITORING OF GEOTHERMAL FIELDS IN THE PHILIPPINES

**Jaime Jemuel C. Austria, Jr.**
Reservoir and Resource Management Department
PNOC – Energy Development Corporation
PNPC Complex, Merritt Road, Ft. Bonifacio
Makati City, Metro Manila
PHILIPPINES
*rrmd_reservoir@energy.com.ph*

**ABSTRACT**

The present database system of PNOC – Energy Development Corporation for reservoir engineering is improved to support rapid correlation of diverse sets of reservoir data for model conceptualization and reservoir modelling. Simple application programs for data visualization were created to show how database functionality can be enhanced from mere data storage into a useful tool for reservoir simulation and monitoring.

Logical data blocks are drawn from an Oracle database on the fly within a UNIX shell script to create dynamic line graphs and contour planes of temperature and pressure providing basic graphic representation of the reservoir. Developer forms are created for graphical retrieval and handling of data.

## 1. INTRODUCTION

Management of a geothermal reservoir relies on adequate information on the geothermal system (Stefánsson and Steingrímsson, 1980). The prediction of the behaviour of a reservoir during the production stage depends on the conceptual model of that reservoir, the governing physical processes, and the quality of data used in the interpretation. Good quality of data is not an assurance of a perfect conceptual model but is important in realizing that goal (Grant et al., 1982).

The study deals with the optimization of the design of, and inclusion of, new application tools to the present database system of the Reservoir and Resource Management Department (RRMD) of PNOC–Energy Development Corporation (PNOC-EDC). The RRMD database has been redesigned to capture more accurate information on geothermal systems, and to support the data requirements of reservoir modelling and field monitoring.

At present, the database contains information from close to 350 wells that have been drilled into the geothermal fields of Bacon-Manito, Palinpinon, Tongonan, and Mindanao; and into exploration areas of

Northern Negros, Mt. Labo, Central Leyte, and Southern Leyte. The data in all geothermal fields are contained in spreadsheets with a file-based structure. In 1997, a decision was made to create a data management system for geothermal data for all project locations to improve the quality of data used for managing the reservoir. Web-based database set-up using thin-client architecture was chosen to be ideal for the collection of data from remote locations (Zapanta et al., 1999). As a result, a geothermal database management system broadcasted on the web was adopted for RRMD.

Geothermal data from different areas are standardized and collected in a central database in Fort Bonifacio, Makati. Collection of data includes contextual information such as wellhead and well track coordinates, and boundaries of pads, sectors and projects. Numerical data from temperature, pressure, spinner, calliper, and sonic logs can be downloaded and used to create graphs. Maps of well locations of different geothermal projects are available in static hypertext mark-up language (HTML) format. Well test interpretations provide information regarding feed zone locations, injectivity indices, permeability thickness and skin values.

Access to the database is comprehensive as it allows personnel, with an approved internet protocol (IP) address and valid user ID, to view and update information as needed by using only their web browsers.

The study includes a review of the current reservoir engineering database setup for PNOC-EDC against a data management model for reservoir simulation.

The database has application forms to perform basic calculations like data interpolation for temperature and pressure. The database management system also has basic computational tools, but its architecture has to be refined to provide better collection of data for reservoir conceptualization and modelling. Moreover, database application programs need to be developed to provide a basic graphical representation of the reservoir data.

The database and applications were developed using the software installed on the UNIX server, called Strokkur, at Orkustofnun. These UNIX-based programs were accessed from a personal computer operating on Windows XP 2000 by installing an X-windows terminal on the PC to emulate the UNIX environment.

## 2. BACKGROUND OF STUDY

### 2.1 Current web-based database system of PNOC-EDC for reservoir engineering

In 1997, the Reservoir and Resource Management Department of PNOC - EDC, developed a web-based geothermal database management system using thin-client architecture to handle large volumes of data coming from remote and geographically diffuse project locations. Data from these sources come in heterogeneous types and formats.

Database management systems using thin-client architecture operate like a corporate intranet with database connectivity. Transmission control protocol / internet protocol (TCP/IP) is used as a communications protocol by the intranet to provide hypertext transfer protocol (HTTP) services containing information from the database to users.

TCP/IP is a two-layer program. The upper layer, TCP, manages the assembly of a message into smaller packets that are transmitted over the Internet and received by a TCP layer that reassembles the packets into the original message. The lower layer, IP, handles the address part of each packet to make sure it gets to the right destination. Each gateway computer on the network checks this address to see where to forward the message.

Thin-client architecture is a multi-level architecture with distributed computing power. Thin-client architecture has a web-browser client on the front-end, an application server in the middle, and a database server on the back-end. The client portion, as shown in Figure 1, is an entry screen for making requests and receiving services, such as a web page, from a computer functioning as an application server in the network.

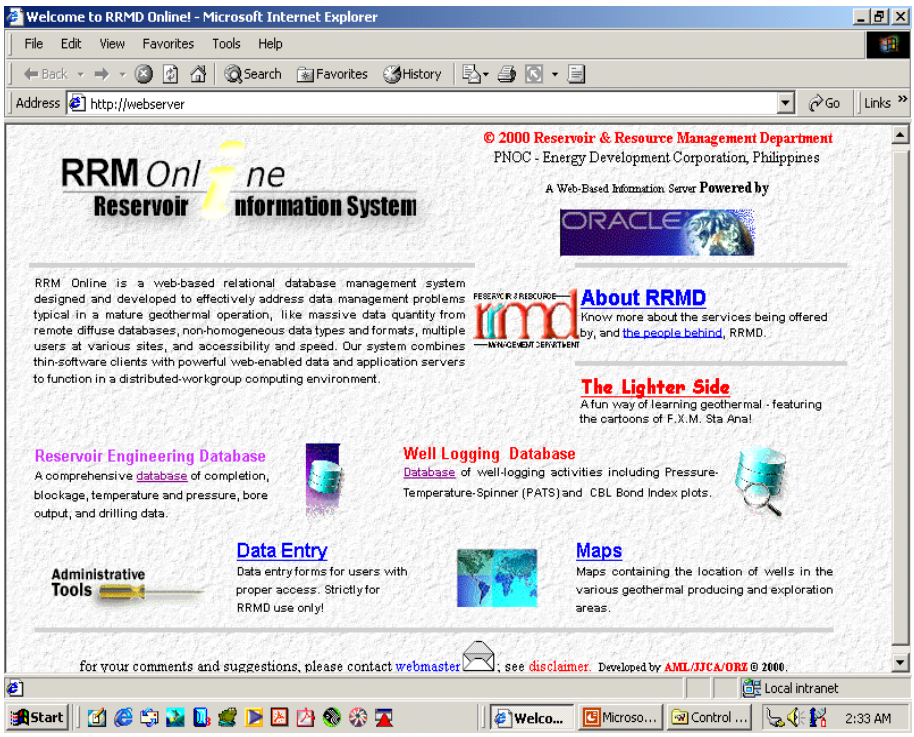



FIGURE 1: User interface for RRMD web-based database

## 2.2 Web-based database architecture and its advantages

The first tier in a multi-tier web-based database system is the user interface. In the RRMD configuration, a web browser that comes bundled with preinstalled application programs in desktops and mobile computers is used as a user interface. Computer users from Fort Bonifacio, Makati central office, and from satellite offices in Tongonan, Palinpinon, Bacon-Manito, and Mindanao geothermal projects can access data using the web browsers on their computers as long as they are connected to the LAN and WAN.

The second tier is an application server containing application programs and forms. An application server may be any server supporting web services and web scripting languages like HTML and JavaScript. In the RRMD configuration, the second tier is a Microsoft NT computer running Internet Information Server (IIS). The data entry and retrieval forms are designed using Microsoft FrontPage and JavaScript.

The third tier is a web server which is an HTTP server that has object database connectivity (ODBC). ODBC allows creation of HTML documents using data from a relational database. In the RRMD set-up, Oracle Webserver's Listener triggers the Oracle web agent when a database connection is requested, to make a connection to the Oracle database. Stored procedures written in procedural language/structured query language (PL/SQL), which provide program logic, are compiled in an Oracle 8.05 server to create dynamic HTML pages, and storage for dynamic data in relational tables.

The fourth tier that completes the architecture is the geothermal data which is stored in an Oracle relational database management system (RDBMS). The Oracle RDBMS runs on SUN's Solaris platform.

The main advantage of a web-based database system is the database and analysis modules are stored in a centralized system. Updates to programming applications only have to be made on the server. No additional programs are needed at remote client desktop computers. Changes, whether in the data, forms, or application programs, are dynamically made available to all users at their web browsers without having any need to install additional programs.

## 3. DEVELOPMENT OF GEOTHERMAL DATABASE

### 3.1 Objective of the study

This UNU geothermal project deals primarily with the improvement of the present database system to extend its function as storage of archival geothermal data into an effective tool for reservoir conceptualization and modelling. The database was restructured to be more fitting for database application programs that were developed to facilitate the rapid correlation of different sets of reservoir data and provide visualization of different reservoir parameters.

### 3.2 Effective geothermal database design

A systematic collection of data is important in understanding a geothermal resource in its natural state and in monitoring changes of the resource during exploitation. The database provides a structure for systematic gathering of information. By placing data in a database, standards are imposed on data that improve the quality of data and make data easily accessible in the future. Data have to be interpreted and analyzed before a conceptual model can be developed. A conceptual model represents understanding of the resource at a given time based on a particular set of data. In order to have better understanding of a resource, all available data sets are used to create many conceptual models. Different models of the resource are integrated to create an integral model of the resource. The data management system must therefore provide a flexible retrieval and comparison of data from different sources for data analysis. Furthermore, the data management system must allow data to be shared with all experts who are involved with the analysis of data (Anderson, 1995).

Management decisions regarding the resource are made based on this integral model. An effective data management system should indicate assumptions made during development of the conceptual model so that models can be updated easily; and decisions will be based on known assumptions. Moreover, a sound data management system must ensure that data can be improved upon, and improvements are noted while original data remains the same (Anderson, 1995).

### 3.3 Data review based on role of data management in numerical simulation

Data modelling is important to define areas where the present data management system is adequate and areas where it needs improvement. The reservoir simulation experience of PNOC-EDC is evaluated against the data flow diagram showing how data management and related software is used in reservoir simulation (Figure 2).

Batches of raw data verified to be correct are sent from field offices to the Makati head office by electronic-mail for uploading to the database. The architecture of the database allows for interactive updating from remote locations using the web browser. However, due to bandwidth limitation from the central office to remote locations, this method is slow and is not practical. Data from the field offices, in file-based spreadsheet format and in flat file databases, are recorded on recordable compact discs for archival purposes. Meanwhile, electronic logging data are stored on tape cartridges.

Data viewing is done using the RRMD web page interface. After data are uploaded to the database, users with a proper IP and authorized user ID can view a web report from computers that are part of the LAN and WAN. The reports can be saved directly as HTML, or can be copied and pasted into a text editor.

The database, ideally, is linked seamlessly to various stand-alone reservoir engineering programs used to determine stable formation temperature, interpret well tests, and simulate wellbore conditions. However, the current reservoir database is not linked seamlessly to any stand-alone reservoir application software. Data is searched from the database and downloaded before it can be used as input for a well test analysis, wellbore modelling or other software.

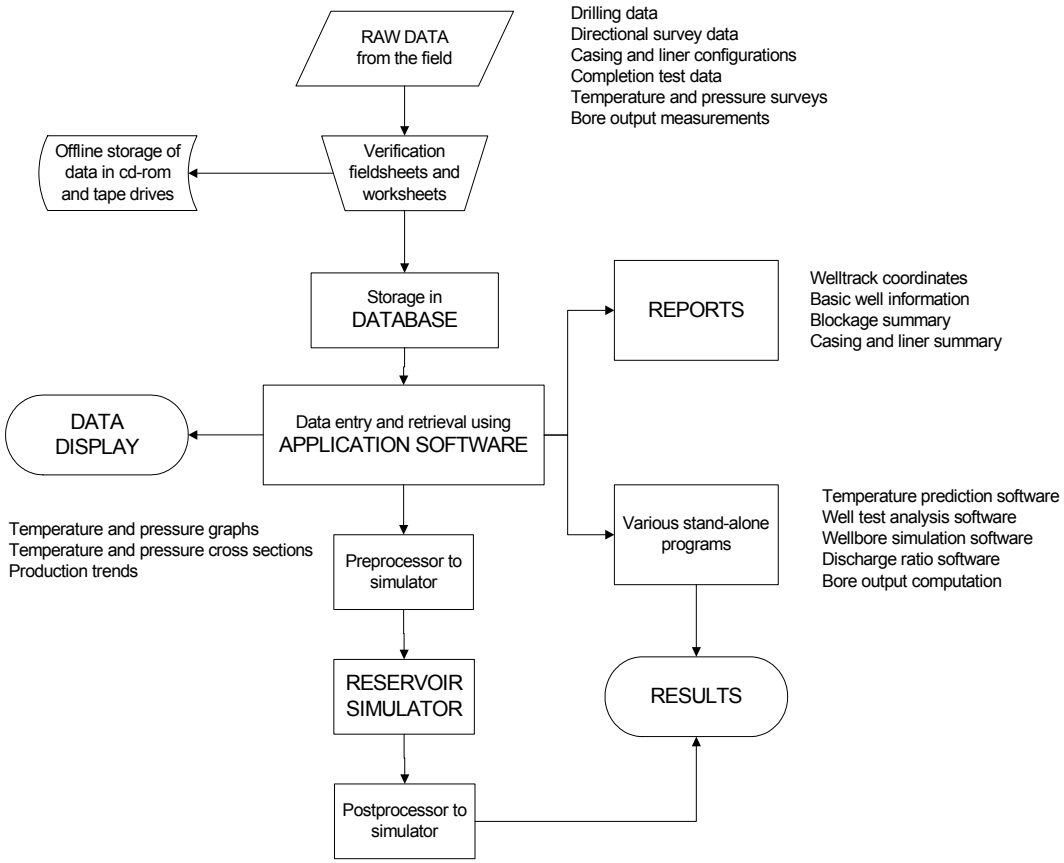



FIGURE 2: Role of data management and related software in reservoir simulation patterned after GeothermEx Inc. (1995)

The database, ideally, is linked seamlessly to a reservoir simulator pre-processor via the application software. In this manner, the database is able to supply information required for a numerical simulation model, which is nearly the entire database collected in the course of exploration, drilling, well logging, well testing and production/injection operations over the life of the project. In this aspect, much work needs to be done on the RRMD database before it can generate sufficient data for reservoir model conceptualization, or perhaps create a preliminary input deck for the simulator.

The reservoir simulator should be able to pass simulated data to a postprocessor for model calibration. TETRAD, the reservoir simulator currently being used, has a postprocessor that allow results of modelling to be viewed graphically and calibrated against measured data.

The database should be able to display graphics on the fly using data from the database. Currently, temperature and pressure contours and cross-sections are prepared manually using third-party graphing and contouring applications. The current reservoir database does not have any capability to display data dynamically.

### 3.4 Tools used for database and applications development

Assortments of programming languages and graphing tools are used to develop the database and the applications. These include SQL+, Oracle SQL* Loader, UNIX shell scripting, AWK, Oracle Developer, GNUPLOT, and GMT.

The database engine used is Oracle8i Enterprise Edition Release 8.1.7.3.0. Being a relational database, information is stored into Oracle's tables with the ability to link data between tables at the record level.

SQL+ is used to create the database. SQL+, or Structured Query Language Plus, is the language used for relational database development. The language consists of statements to insert, update, delete, query and protect data. The language is non-procedural which means users only have to specify which data they want and how this data must be found. Oracle SQL* Loader is used to load data in batches. SQL*Loader is an Oracle-supplied utility that allows the user to load data from a flat file into one or more database tables (Moran and Dimmick, 1987).

SQL+ statements are embedded in simple programs written using UNIX shell scripts. The shell scripts run embedded SQL+ statements to query the database and spool selected data into a file. The shell script automates the plotting program to use the input data and to display the data in graphical form. Data files from the Oracle database are formatted into the desired data input structure using AWK. AWK, named after its developers Aho, Weinberger, and Kernighan, is a UNIX utility that uses regular expression for pattern matching. An AWK program searches a set of patterns and actions that tell what to look for in the input data and what to do when it is found. The action language looks like C but there are no declarations, and strings and numbers are the built-in type (Aho et al., 1988; Dougherty, 1992).

Forms and reports interface applications are created using Oracle Developer. Oracle Forms can be used to view, insert, and edit data while Oracle Reports allows the user to access data and to publish it (Lakshman, 2000).

For the X-Y plots, GNUPLOT is used to create the graphs using input data from the database. GNUPLOT is a command-line driven interactive plotting utility that runs on UNIX, MSDOS, and VMS platforms. The software is copyrighted but is freely distributed. GNUPLOT can handle both curves (2 dimensions) and surfaces (3 dimensions).

Generic Mapping Tools (GMT) is used to generate the contours. GMT is a free software package, with General Public License, that can be used to manipulate columns of tabular data, time-series, and gridded data sets, and display these data in a variety of forms ranging from simple X-Y plots to maps and colour, perspective and shaded-relief illustrations (Wessel and Smith, 1998).

All of the software used are installed on the Strokkur server of Orkustofnun. To run these UNIX-based programs on a personal computer operating on Windows XP 2000, an X-windows terminal is installed first on the PC to emulate the UNIX environment.

### 3.5 Database design and creation

In designing the database, key information required for successful resource management as described by Stefánson and Steingrímsson (1980) is considered. These include:

- Knowledge on the volume, geometry, and boundary conditions of a reservoir;
- Knowledge on the properties of the reservoir rock, i.e. permeability, porosity, density, heat capacity and heat conductivity; and
- Knowledge on the physical conditions in a reservoir, which are determined by temperature and pressure distribution.

There is a conscious effort to separate original data from derived data, to include the source of interpretations, and indicate the accuracy of measurements. The improved design also uses strictly ID numbers as the primary key instead of well-names to describe relationships between different data sets.

#### 3.5.1 Well location module

The Location module includes tables for pad, sector, project, and cellar that describe the geographical

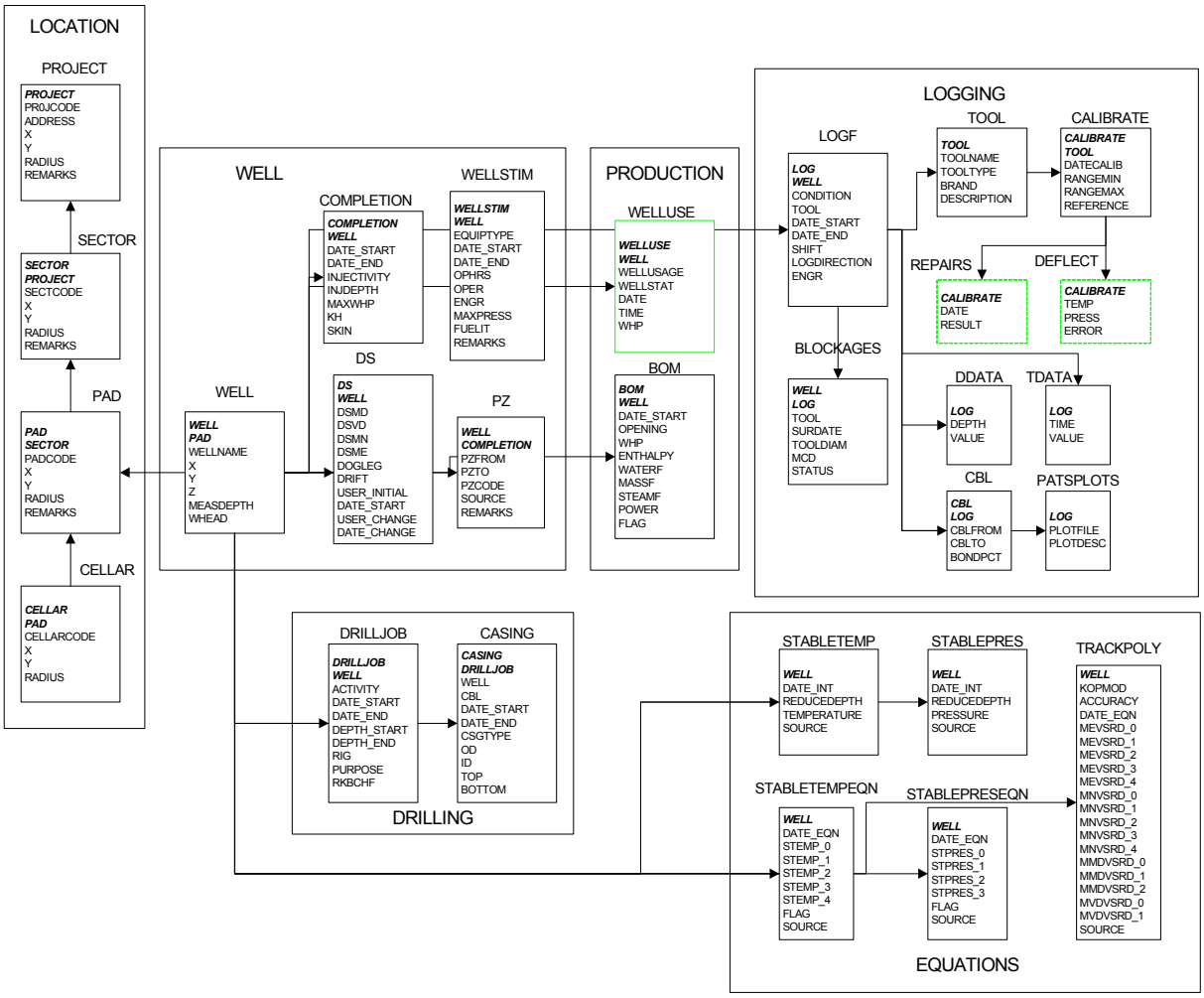



FIGURE 3: Entity-relationship diagram for reservoir engineering database

location of a well by area (see Figure 3). The tables in the Location module describe the location on a well by giving its rough centre and radius.

- **PAD** table contains ID of pad and sector where pad belongs, name of pad, easting (x) and northing (y) coordinates of pad, radius of circular area and a remarks column describing the pad location.

- **SECTOR** table contains ID of sector and project where sector belongs, name of sector, easting (x) and northing (y) coordinates of centre of sector area, radius of sector area, and a remarks column for the official name of the sector.

- **PROJECT** table contains ID of project, official name of project, mailing address of project, easting (x) and northing (y) coordinates of centre of project area, estimated radius of project area, and a remarks column for the official name of the project.

- **CELLAR** table contains ID of cellar and pad where cellar belongs, name or code of the cellar, easting (x) and northing (y) coordinates of centre of cellar, and estimated radius of cellar area.

### 3.5.2 Drilling module

The Drilling module contains information about the drilling activity and the casing configuration of a well (see Figure 3).

- **DRILLJOB** table contains drilling ID, well ID, type of drilling activity, date of start and finish of drilling activity, initial and final depth of drilling, name of rig used for the activity, reason for activity, and distance from rotary table to wellhead.

- **CASING** table contains casing ID, ID of drill job, well ID, sonic log ID, date of start and finish of setting of casing, type of casing, outer diameter of casing or slotted liner, inner diameter of casing or slotted liner, top of casing, and bottom of casing.

### 3.5.3 Well module

The Well module has information about the wellhead coordinates, well track coordinates, results of well tests and stimulation activities done on a well (see Figure 3).

- **WELL** table contains ID of well and pad record where well belongs, nickname of well, easting (x), northing (y), and elevation (z) wellhead coordinates. Well coordinates are based on National Mapping and Resource Information Authority (NAMRIA) maps with a scale of 1:50,000, and use the Clarke 1866 spheroid and Transverse Mercator projection as coordinates (Los Banos, pers. comm., 2003). The table also contains the total measured depth of well and ID of well head. WELL is a central table to which all other tables are linked. Several views can be created to link WELL with tables describing location to speed up searches for a well by pad, sector, or project.

- **COMPLETION** table contains the interpreted results of a well completion test. A well completion includes several tests like waterloss, injectivity, and pressure transients. COMPLETION stores the ID of the completion test and well, date of start and finish of well tests, injectivity index of well computed from injectivity tests, depth of setting of the pressure tool during injection test, maximum wellhead pressure during injection test, computed permeability thickness and skin.

- **DS** table contains results of directional surveys. DS is widely used in mapping applications when projections of well track of deviated wells are made on different surfaces and cross-sections. Entities in this table include the ID of the directional survey, the ID of the well where the survey was done, measured length along the casing and vertical depth where a directional survey point was taken, easting (x) and northing (y) coordinates of the point where the survey is taken, dogleg severity, drift angle, names of encoder or personnel who made changes to the record, and dates when record was inputted or changed.

- **PZ** table includes permeable zones of a well interpreted from temperature, and in some cases, spinner logs. PZ contains the ID of the well where the log was done, completion ID when the permeable zone is identified from a log during well completion, interval of permeable zones, rough classification of the permeable zone as major or minor, data source such as logs performed to identify a permeable zone or a wellbore simulation, and author of interpretation.

- **WELLSTIM** table covers parameters measured when a well is initiated to discharge by air compression. WELLSTIM contains the ID of well stimulation activity, the ID of the well being discharged, the type of equipment used to discharge the well, date of start and finish of well activity, total hours of operation, maximum pressure reached during compression, volume of fuel consumed by equipment, names of engineers and operators who carried out the project, and comments on how to improve the next discharge stimulation activity.

### 3.5.4 Production module

The Production module contains tables that describe the production history of wells, utilization of production and re-injection wells, and measurements of well output (see Figure 3).

- **WELLUSE** table contains the ID record of well utilization, ID of the well, present use of the well (e.g. used for production, re-injection, or monitoring), status of the well (e.g. shut, in use for power plant, flowing, or bleed off), date and time of start of utilization, and wellhead pressure while being used.

- **BOM** table or bore output measurement table contains ID of bore output records and well, date of start of output measurement test, wellhead condition, computed enthalpy, water flow, mass flow, steam flow, and power output, and a flag for stable outputs that can be used for generating bore output curves.

### 3.5.5 Logging module

Logs give information on well performance and design. This information is necessary during drilling to avoid anticipated drilling difficulties, and to estimate success. Logs give information on structure, physical properties and performance penetrated by wells (Stefansson and Steingrímsson, 1980).

The Logs module contains information about geophysical logs such as temperature, pressure, spinner, sonic, calliper, blockage detection, and flow. The Logs module also contains a description of the tools used for logging, results of calibration performed on a tool to keep it accurate, and records of repairs done on tools. The modules for logging are designed to allow tracking the number of times a tool has been used and the wells where it has been used. This feature is very important in tracing all the wells logged by a tool that is giving wrong readings due to an off-calibration (see Figure 3).

- **LOGF** table contains the ID of the log and well, the condition of the well while it is being logged (e.g. shut, flowing, on-bleed), tool used for logging well, date of start and finish of logging activity, shift value to correct for errors in zeroing to the reference point during logging, direction of log (e.g. log is made going up, going down or stationary), and names of loggers.

- **DDATA** table contains logging data against depth. Data from moving logs of temperature, pressure, and spinner against depth are stored in this table.

- **TDATA** table includes logging data against time or time series data. Data from stationary logs of pressure like those performed during multi-rate injection and pressure transient tests are stored in this table.

- **TOOL** table contains the ID of the tool, name of the tool, kind of tool (e.g. temperature, pressure, spinner, flow meter, neutron, gamma, resistivity, X-Y caliper, casing inspection or multifinger caliper), brand or manufacturer of the tool, and general description of the tool.

- **BLOCKAGES** table contains records of all blockages detected during a Go-devil, lead impression block, and sinker bar survey. BLOCKAGES contains the ID of the log and well, date of the log, outside diameter of the tool used, and depth where the blockage is detected. This table can be a part of the LOGF table.

- **CBL** table contains data on cement bond logs. CBL includes the ID of the log, the casing interval where CBL is performed, and the quality of cement bond or percent bond index. This table can also be a part of the LOG table.

- **PATSPLOTS** table contains previous plots of logs performed on wells in HTML format.

- **CALIBRATE, REPAIRS** and **DEFLECT** tables contain information about calibration of tools, equations used in the calibration of tools, and details of repairs done on tools.

### 3.5.6 Equations module

The Equations module contains interpreted stable formation temperature and pressure from downhole logs and polynomial equations describing the stable profiles. There is also a table describing the well tracks using polynomial equations. The equations module is mostly used for finding temperature and pressure of wells at different depths in the reservoir (see Figure 3).

- **STABLETEMP** and **STABLEPRES** tables contain stable formation temperature and pressure. The tables have interpreted stable formation temperatures and pressures, depth where measurements are taken, and the name of the person who made the interpretation.

- **STABLETEMPEQN** and **STABLEPRESEQN** tables contain equations of the polynomial fit line to the interpreted stable formation temperature and pressure plots.

- **TRACKPOLY** table contains polynomial equations describing the welltrack. The polynomial equations relate easting, northing, measured depth, and vertical depth to absolute depth. The table also stores the name of the person who created the polynomial fit.

The details of these tables are contained in the data dictionary included as Appendix I.

## 3.6 Techniques in data migration and population

Data relevant to the project were imported into the Strokkur server of Orkustofnun using Oracle's import utility. Data from previous tables are migrated using SQL statements from the command line when the columns in the table have the same data types and field lengths.

Volumes of data in ASCII format are batch-loaded using SQL*Loader. AWK is used to format ASCII data into a structure readable by SQL*Loader. A feature of SQL*Loader, the control file, is written to describe data for loading and which tables and columns will accept the data. After writing the control file, the load process is started by invoking SQL*Loader executable and pointing it to the control file. ASCII data is read by SQL*Loader and loaded into the database.

The design of a database is not a single and discrete phase of a data management project, therefore, control scripts are kept in an archive folder in case the tables need to be dropped and rebuilt in the life cycle of the project.

# 4. DEVELOPMENT OF SIMPLE DATABASE APPLICATIONS

## 4.1 Development of temperature and pressure program

Graphing modules **TPLOT** and **PPLOT** are developed to give reservoir analysts a quick view of the plot of all available temperature and pressure data for all wells in the database. These modules display dynamic two-dimensional plots of temperature and pressure for any given well within a given time frame. Updated temperature and pressure data are readily seen on the graphs when the graphs are generated again after the changes are made.

The **TPLOT** program is capable of extracting temperature data from the database and generating a plot of temperature against depth. The **PPLOT** program, on the other hand, can produce a plot of pressure against depth (see Figures 4 and 5).

The basic algorithm calls for a UNIX kornshell script to run an embedded SQL+ statement to retrieve temperature values against measured depth and spool it into an initial input file (Bolsky and Korn, 1995). The same kshell script contains an AWK string used to filter the initial input file and create a file in a format readable by plotting software. The kshell script automates the plotting program, which is in this case GNUPLOT, to generate a graph with the desired features.

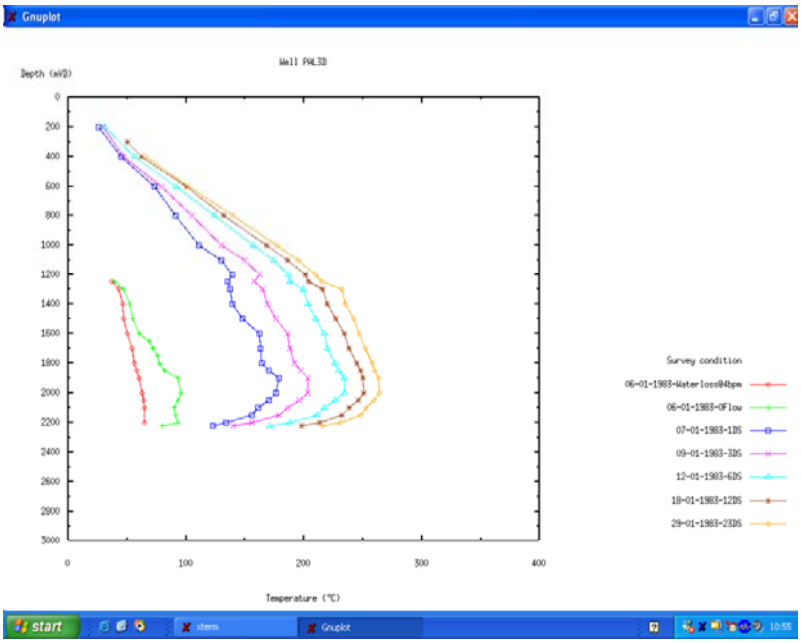



FIGURE 4: TPLOT showing temperature profile during heat-up of well PAL3D in the Bacon-Manito geothermal production field

**TPLOT** and **PPLOT** programs can display temperature and pressure logs taken at specified inclusive ranges of dates. The temperature logs taken at different dates are displayed in various line colours and symbols for distinction. The condition at the time when the log is taken is also displayed along side the data as a guide for analysis.

Both programs run under a UNIX environment. They can also be used on a Windows platform provided there is an X-Window that emulates a UNIX environment.

The user is prompted to enter a well ID, and the range of the dates of the survey to generate the desired plots. By changing **TPLOT** to **PPLOT** and leaving the rest of the input the same, corresponding pressure surveys are obtained using the same given dates.

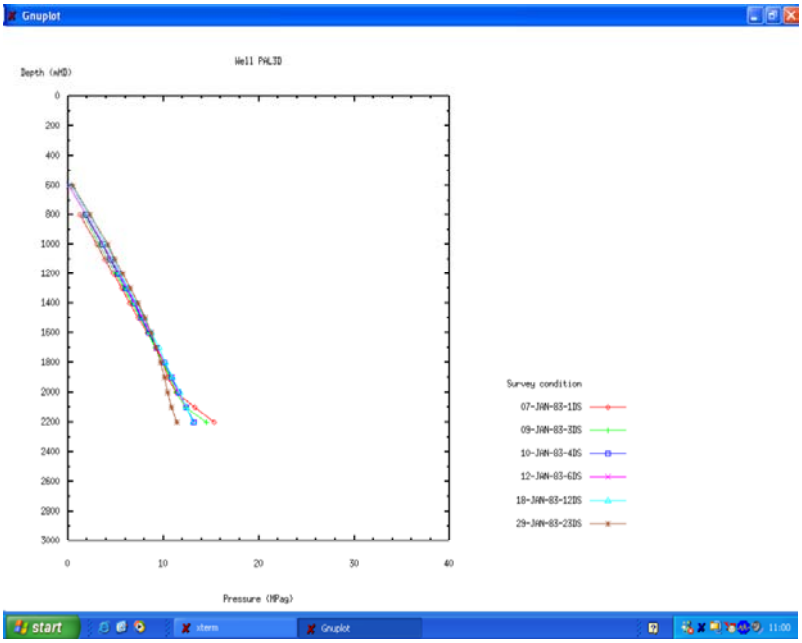



FIGURE 5: PPLOT showing pressure profiles of well PAL3D in the Bacon-Manito geothermal production field

The syntax to run **TPLOT** and **PPLOT** programs is shown below. The script for this simple X-Y plot can be seen in Appendices II and III.

```
Program T P L O T to get Temperature log from Oracle
Usage:  tplot wellname fromdate(DD-MM-YYYY) todate(DD-MM-YYYY)
Program P P L O T to get Pressure log from Oracle
Usage:  pplot wellname fromdate(DD-MM-YYYY) todate(DD-MM-YYYY)
```

## 4.2 Practical applications of TPLOT and PPLOT

The simple programs TPLOT and PPLOT can be very useful tools for analyzing downhole logs. Behaviour of wells as inferred from downhole logs used as a basis to explain physical processes happening down in a geothermal reservoir. Graphs of temperature and pressure generated from TPLOT show how the program can be useful in visualizing well behaviour from the logs.

The permeable zone can be located from downhole logs, such as successful water loss tests.

A profile of well PAL4D in the Bacon-Manito geothermal production field (BGPF) shows a well warming by conduction as water flows down the casing and casing section (Figure 6). Below the region of slow warming, a depth is reached where the temperature rises sharply. Water is not moving at this section of the well below this water level and is lost into formation.

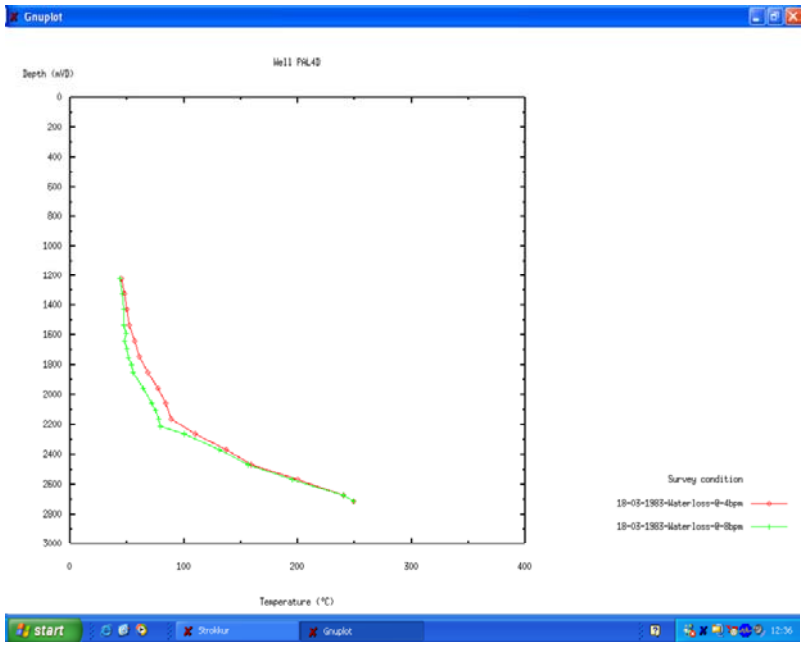



FIGURE 6: Water loss profile of well PAL4D generated by TPLOT

A profile of CN2RD in the BGPF shows partial water loss in the upper zone (Figure 7). Reduction in flow as water flows down the well leads to greater heating by conduction. The upper zone loss is indicated by steep change in the thermal gradient. In this case, the upper zone is most likely the major zone as most of the water is lost in this part causing significant heating.

A profile of CN3D in the BGPF shows a step change in temperature indicating inflow of hot water mixing with injected cold water (Figure 8). The interval of log measurements has to be closely spaced to be able to identify the step change.

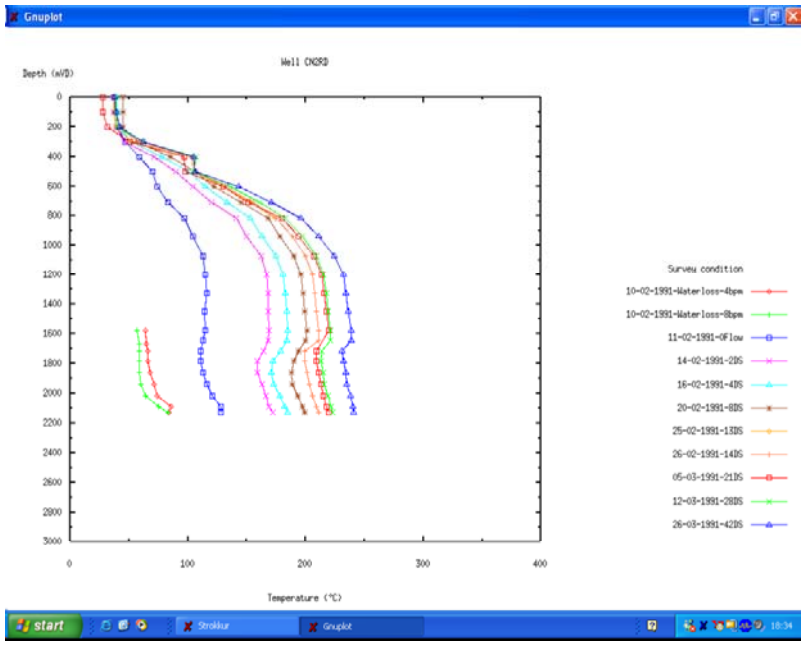



FIGURE 7: Water loss and heat-up profile of well CN2RD generated from TPLOT

Formation temperature and pressure can be inferred from downhole logs if the well is shut long enough to adjust to stable temperature and pressure. However, temperature and pressure measurements from logs do not necessarily correspond to those of the reservoir. Correspondence depends strongly on the reservoir pressure gradient and distribution of feed zones.

It is possible to determine the probability for a well to self-discharge by analyzing the well's thermal recovery or heat-up profile. The heat-up process is not uniform as the fastest rate of heating occurs early. Temperature surveys are scheduled in intervals of 2, 4, 8, 12, 18, 24, 30, and 45-days until the well is discharged. By using the extrapolated stable formation temperature and the boiling point with depth curve, the probability of a well to self-discharge can be computed from the ratio of the area of condensation to the area of flashing (Sta. Ana, 1985).

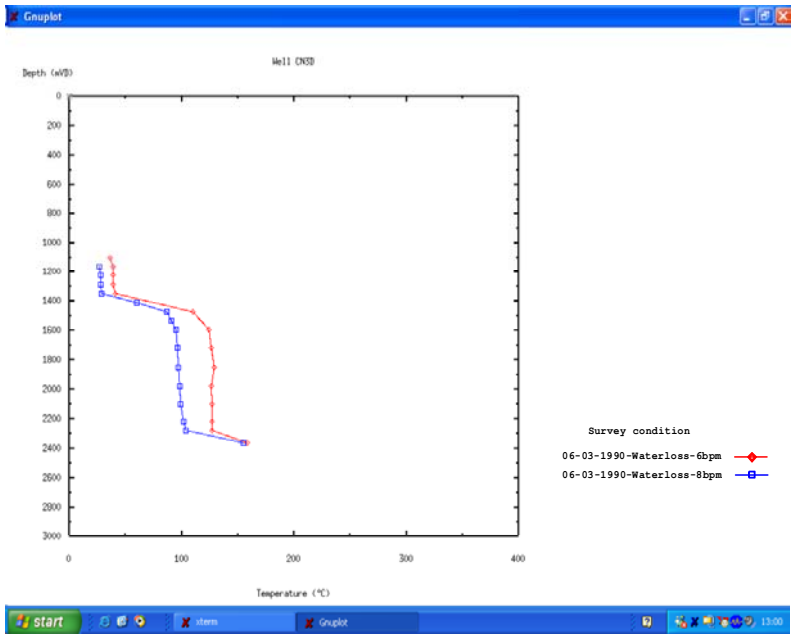



FIGURE 8: Water loss and heat-up profile of well CN3D generated from TPLOT

### 4.3 Forms development

Oracle Forms are used to create a graphic interface for retrieval, review and modification of data from the database. With these forms, the user does not need to learn how to use the SQL+ language to retrieve, update, and save changes to the database. One or more tables can be used to create Oracle Forms as the design and contents of Oracle Forms can be customized depending on the data requirements of the user (Lakshman, 2000).

Oracle Forms Designer is run from the Strokkur server through an X-Windows terminal emulating the UNIX environment. The command **`f60desm`**, which stands for Forms 6.0 design mode, is issued to invoke Oracle Forms Designer. Several tools are available for forms development under Forms Designer, such as the layout editor and property palette.

A form can be designed to view or update any table or view in the database. The example shown in Figure 9 is a form that can be used to search for all the logs done on a well. A query string can be entered in any of the input boxes. After a query is executed, the tables WELL, LOGF, and DATA in the database are searched for information matching the query string. In the example below, the name of the well PAL3D is used as a query

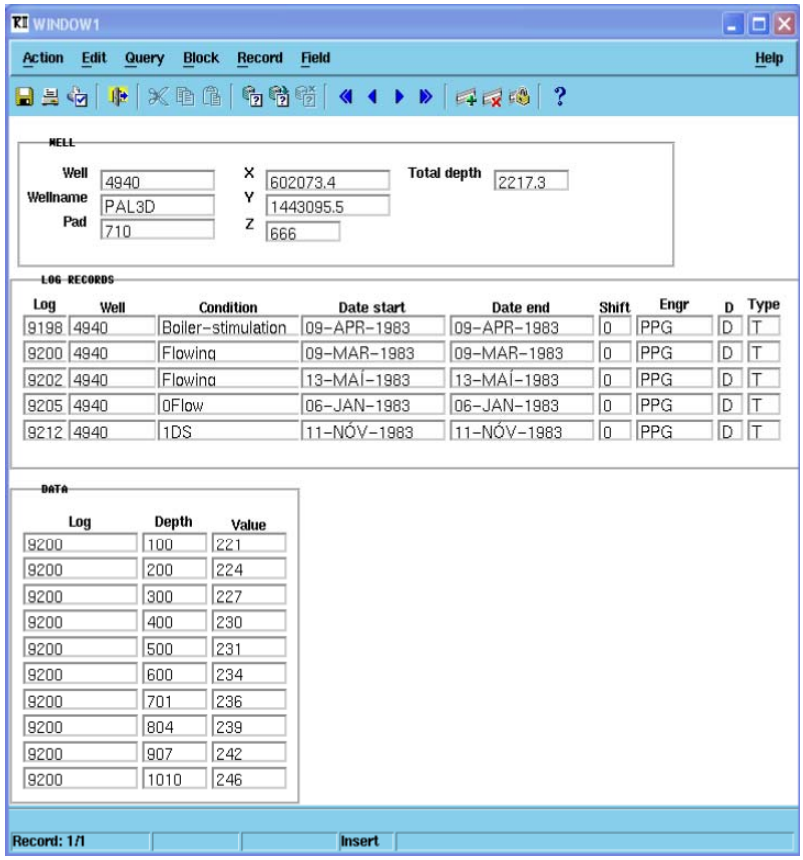



FIGURE 9: Forms for data entry and review; this form searches for logs performed on a well and data obtained from these logs

string. The query, after execution, searches the database for the wellhead coordinates and logs performed on well PAL3D, and returns these key values on screen. These values can be changed by a user who has the proper permission. The cursor can be clicked on other parts of the forms, for example in the log records section to search for logging data.

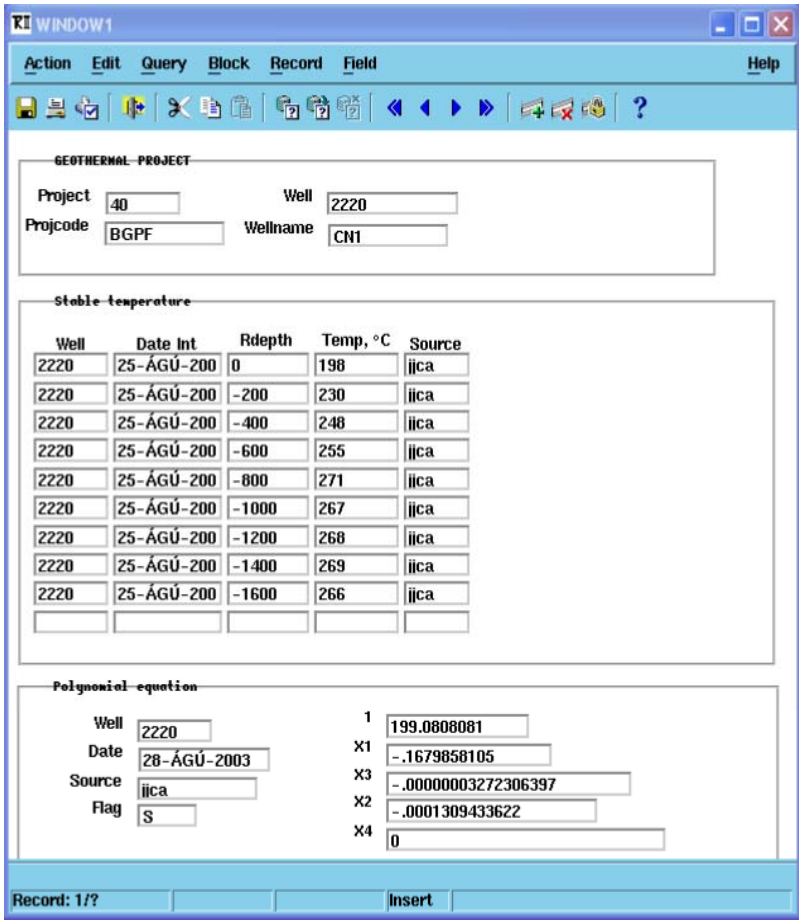



FIGURE 10: Another example of a form for data entry and review; this form displays interpreted stable well temperature and polynomial equations describing stable profile

A variation of the form is shown in Figure 10. The form can be used to search for interpreted stable temperature for any well in the database and the polynomial equations describing these temperature and pressure profiles.

The polynomial fits for these stable temperature and pressure profiles are shown in Appendix IV. The coefficients for these polynomial equations can be changed from this form when there are new interpretations. The author or source of the new interpretation will be recorded. A flag can be used to group these polynomial equations according to the model being developed by the engineer.

To run these forms, the command **`f60runm <name of file>`**, is issued to invoke the forms run mode. At the moment, these forms are run under the UNIX environment, but they can be viewed like a web page under the web-enabled database version with the corresponding http server.

### 4.4 Development of time series plots

A module for graphing data in time series is developed to give the reservoir analyst a quick view of the bore output measurements with time. This program called **TSERIES** has the option of displaying wellhead pressure, massflow, and enthalpy trends of all production wells in the database that have bore output measurements. To create the TSERIES module, UNIX shell scripting and GNUPLOT are used.

The program provides a menu from which the user can select the well output parameter the user wants to view. Depending on the parameter chosen by the user, the UNIX script extracts wellhead pressure, massflow, or enthalpy data from a BOM table from the database and generates an input file called bore_output.lst. GNUPLOT is then automated to plot the bore_output.lst file using a time format axis. Bore output data is plotted on the y-axis while the date of measurement is plotted on the x-axis. The code for this program is shown in Appendix V.

Time series plots are very useful for monitoring individual well output. Monitoring of individual well extraction through periodic output measurement and well utilization monitoring provides useful data on well performance.

Close monitoring of well performance is necessary to ensure a steady supply of steam for the power plant. Corrective measures, like drillout of blockage and/or acidizing, can be undertaken on wells with declining output to clean the feed zones by removing mineral deposition. Well output monitoring is used also to identify wells suffering from drawdown or to indicate reservoir changes like cooling due to pressure changes or due to inflow of natural recharge or injection returns (Sarmiento, 2000).

Shown in Figures 11 to 13 are the wellhead pressure, mass flow, and enthalpy trends, respectively with time of a well in Bacon-Manito geothermal production field. The output trends show a decline because of a mineral blockage detected in the well.

The program is launched by calling the shell script. By varying the input parameter, the user can choose which discharge parameter will be plotted. The program usage is described as:

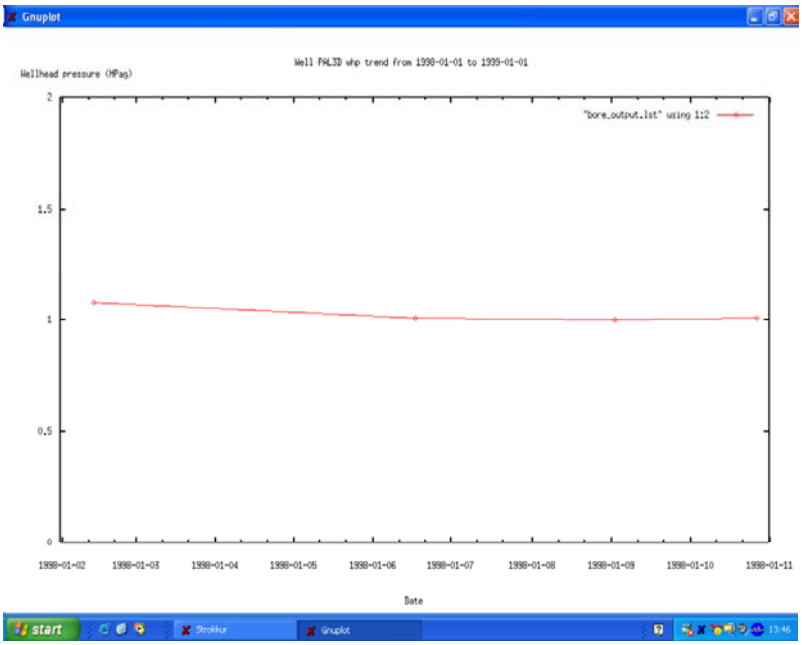



FIGURE 11: Wellhead pressure trend of well PAL3D in the Bacon-Manito geothermal production field

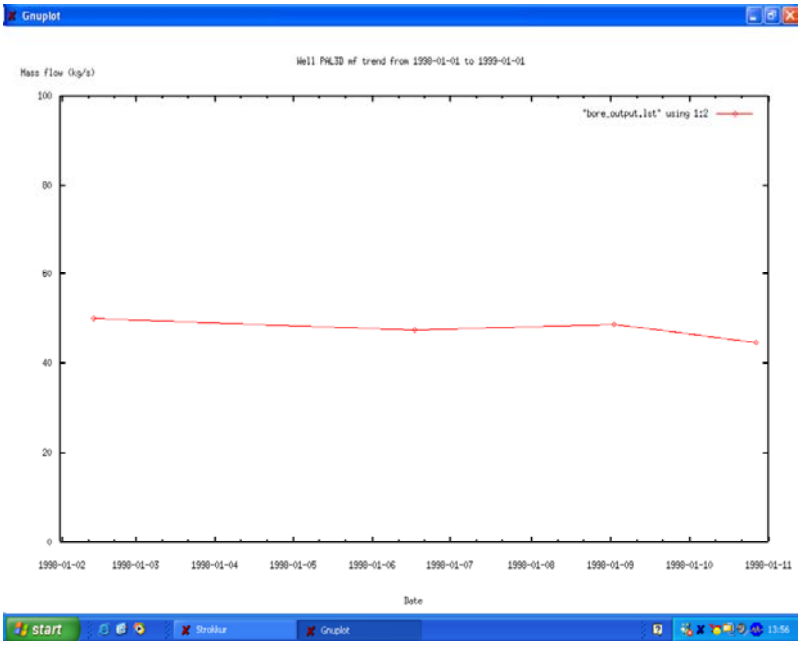



FIGURE 12: Mass flow trend of well PAL3D in the Bacon-Manito geothermal production field

```
Usage:

tseries -w wellname -p param -s start(YY-DD-MM) -e end(YY-DD-MM)

for example: tseries -w PAL3D -p whp -s 1990-01-30 -e 2002-01-30

        PARAMETER -p:
        whp   : Wellhead pressure
        mf    : Massflow
        h     : Enthalpy

        OPTIONS
        -w    : wellname
        -s    : starting date
        -e    : ending date
```

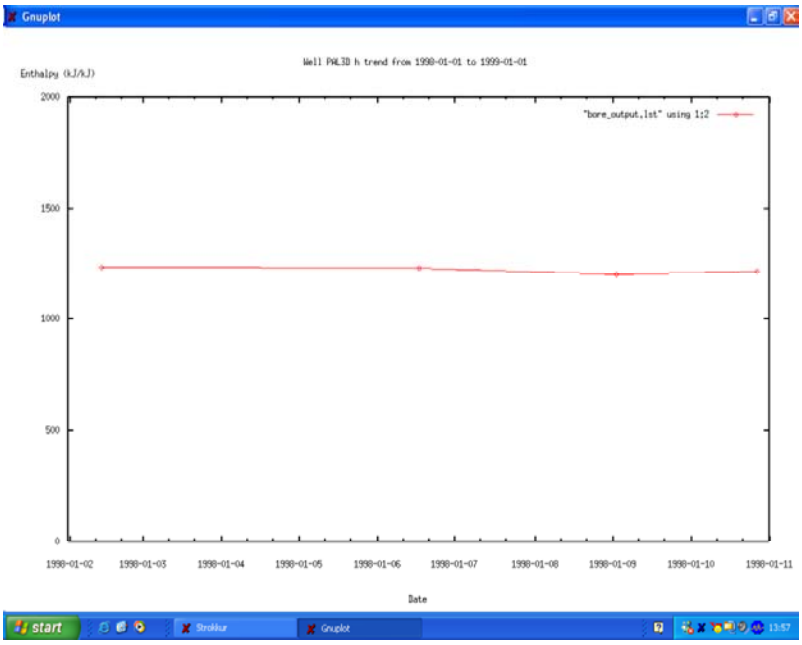



FIGURE 13: Enthalpy trend of well PAL3D in the Bacon-Manito geothermal production field

## 4.5 Development of contouring application

A contour map is a two-dimensional representation of three-dimensional data. The first two dimensions are the XY coordinates, and the third dimension (Z) is represented by lines of equal value. The relative spacing of contour lines indicates the relative slope of the surface. The area between two contour lines contains only grid nodes having Z values within the limits defined by the two enclosing contours. The difference between two contour lines is defined as the contour interval.

Temperature and pressure contours provide an important picture of the temperature and pressure distribution at different levels of the reservoir. From contours, a general pattern of flow can be inferred and used in assigning the heat source or the outflow crop in the reservoir. To facilitate the creation of contours at different depths of interest, a simple contouring program was written to access stable temperature, pressure, and well tracks data from the database to make contour maps (Figures 14-17).

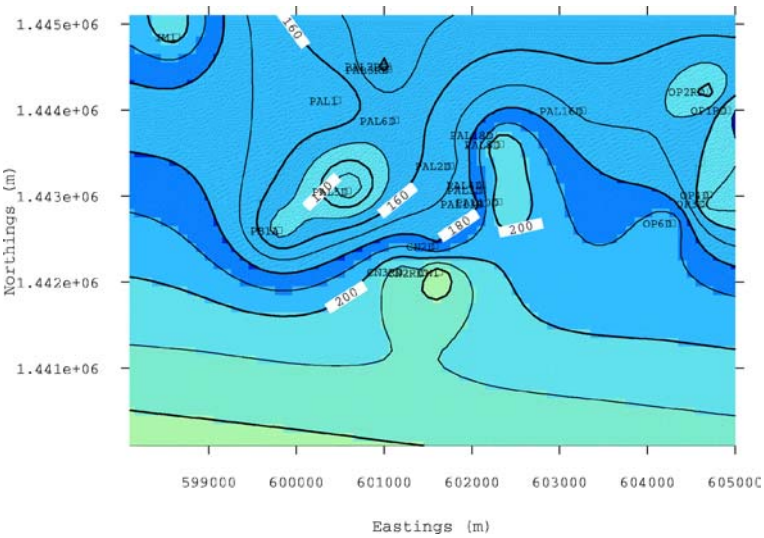



FIGURE 14: Temperature contour of the Bacon-Manito geothermal production field at 200 m b.s.l.

Plots of temperature and pressure are analyzed to estimate the stable formation temperature and pressure. Relevant portions of temperature and pressure surveys are chosen to give a best approximation of formation temperature into an estimated temperature profile which is plotted against elevation. Temperature and pressure points, selected by reservoir engineers to establish a stable well profile, are inputted into the Oracle database.

The polynomial equations describing these stable temperature and pressure profiles are also loaded into the Oracle database. Polynomial equations are used to allow interpolation between points of the log where there is no data. Such is the case when a mechanical gauge is used for logging, resulting in gaps in the logged depths. Polynomial equations for stable temperature and pressure, in conjunction with polynomial equations describing the track of vertical and deviated wells, are used to obtain welltrack coordinates and temperature and pressure values at any given depth. In turn, these estimated stable temperature and pressure values are used to generate contours.

A simple contouring program called **CONTOUR** was developed to retrieve data from the database and

generate dynamic contour plots. The program basically has two major parts: data retrieval and data display. The first part is an embedded SQL+ script in UNIX which is used to select welltrack coordinates, temperature, and pressure values for any project location at any depth of interest from the table of polynomial equations and generate an input file named xyztemp.

The second part is a GMT manipulation kshell script that is used to generate contours. The kshell script looks for the input file xyztemp, grids the input file, and contours the gridded file. The script also sets the color scheme, puts on the name of the wells, and the values of prominent contour lines. The program **CONTOUR** can create temperature and pressure distribution for any project location at any given depth. The script for the program is shown in Appendix V.

Like TPLOT and PPLOT X-Y plotting programs, the **CONTOUR** program can display dynamic data. Any changes made on the data using the forms editor or SQL+ commands are automatically reflected in the contour plots. The program is very scalable, in the sense that, any reservoir parameter can be contoured by updating the *Select statement* in the program code to select other reservoir parameters.

The program **CONTOUR** is used to create temperature contours at different elevations for the Bacon-Manito geothermal production field. These contours can be improved further by masking areas where there are no wells and therefore no temperature data, and by including the complete tracks of wells instead of only the point in the track of the well at the given elevation (see Figures 14 to 17).

To run the program, the user is prompted to enter the ID of the

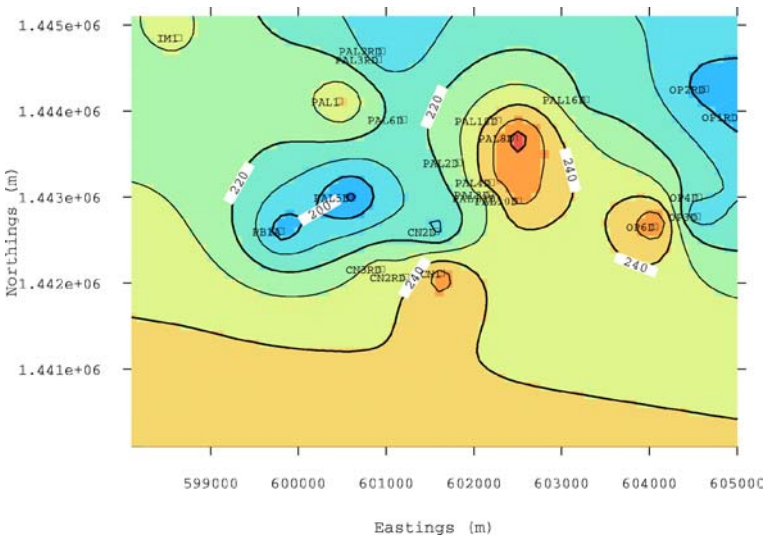



FIGURE 15: Temperature contour of the Bacon-Manito geothermal production field at 500 m b.s.l.

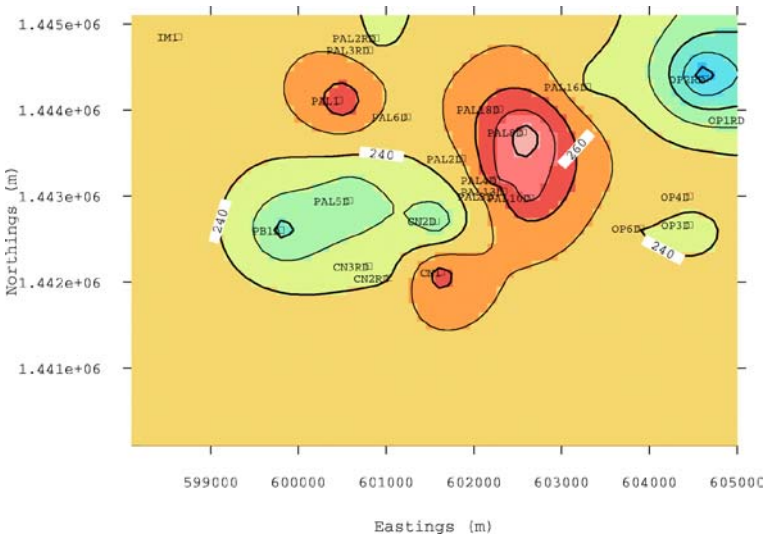



FIGURE 16: Temperature contour of the Bacon-Manito geothermal production field at 700 m b.s.l.

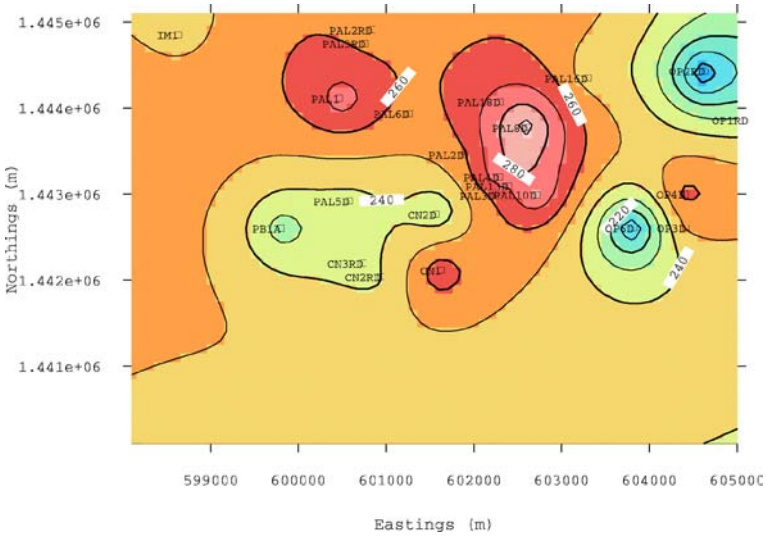



FIGURE 17: Temperature contour of the Bacon-Manito geothermal production field at 800 m b.s.l.

geothermal project and the depth where the contour is to be generated. The program usage is described in the lines below:

```
Program T – C O N T O U R to get Temperature planes from Oracle
Usage:  tcontour project_ID depth_of_interest
```

The code for this program is listed in Appendix VI.

### 4.6 Scripts to generate data file for vertical slice

After creating surface contours, temperature of the wells at different elevations can be used to create vertical sections. Temperature and pressure can be projected at a given cross-sectional cut to generate a vertical section. A SQL+ script is written to select the data points to plot well tracks and contours in a vertical slice. The same SQL+ script used in program CONTOUR is used to get easting, northing, and temperature data. Then a simple triangulation procedure is used to project these data points to a cutting plane from a reference point (see Figures 18a and 18b).

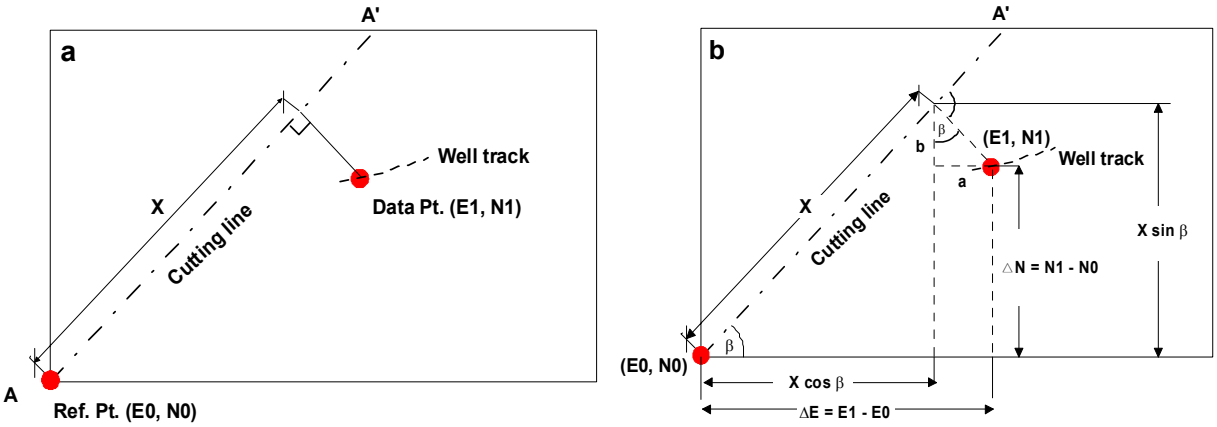



FIGURE 18: a) Surface map; b) Solution by similar triangles

The procedure below describes the calculation procedure in computing the projected distance, **X**, to the cutting plane.

$$\tan\beta = \frac{\Delta E - X\cos\beta}{X\sin\beta - \Delta N} \tag{1}$$

Transposing Equation 1 gives:

$$X\tan\beta\sin\beta - \tan\beta\Delta N = \Delta E - X\cos\beta \tag{2}$$

Equation 2 can also be expressed as:

$$\left(\frac{X\sin^2\beta}{\cos\beta}\right) - \left(\frac{\sin\beta}{\cos\beta}\right)\Delta N = \Delta E - X\cos\beta \tag{3}$$

Multiplying both sides of Equation 3 by $\cos\beta$:

$$X\sin^2\beta - \Delta N\sin\beta = \Delta E\cos\beta - X\cos^2\beta \tag{4}$$

When similar terms are grouped, Equation 4 becomes:

$$X(\sin^2\beta + \cos^2\beta) = \Delta E\cos\beta + \Delta N\sin\beta \quad (5)$$

However, $\sin^2\beta + \cos^2\beta = 1$. Hence Equation 5 becomes:

$$X = \Delta E\cos\beta + \Delta N\sin\beta \quad (6)$$

where $\Delta E = E_1 - E_o$ and $\Delta N = N_1 - N_o$.

When these definitions are used in Equation 6, then it becomes:

$$X = (E_1 - E_0)\cos\beta + (N_1 - N_0)\sin\beta \quad (7)$$

Equation 7 can be embedded in a SQL+ script to project the welltracks and temperature of all the wells in a given geothermal field to a given cross-sectional plane, and create a vertical section. Vertical cross-sections of temperature and pressure are very useful in analyzing the fluid movement in the reservoir and in making a conceptual model of the reservoir. Figure 19 shows how a vertical section is used to visualize the direction of fluid flow in the reservoir.

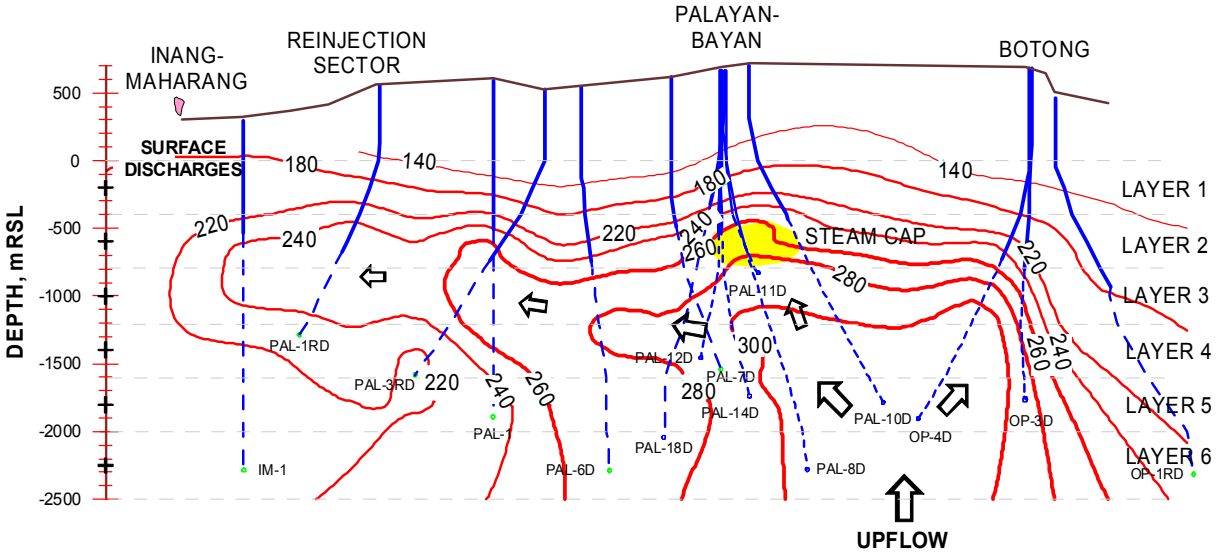



FIGURE 19: Bacon-Manito geothermal production field vertical section showing temperature contours, welltracks, upflow and outflow areas, grid layers, and centres of layers for numerical simulation

### 4.7 Mesh creation for numerical simulation

Modelling constitutes the most powerful tool available to the reservoir engineer, and is applied to various management purposes. Modelling needs lots of data and in this regard, mathematical models are developed on the basis of the historical data stored in the Oracle database, and data visualization tools. Through modelling, the nature and properties of the system can be estimated and its behaviour understood.

The mesh grid shown in Figure 20 is produced using Grapher (and a simple macro written in Excel) to draw the grid lines (see Appendix VII). The mesh grid shows the model layer from 1.2 km. to 1.6 km. below sea level. It contains welltracks, coordinates of the well track at this elevation, and the grid blocks. Block properties for the input deck are stored in tables in the Oracle database. These include properties of the reservoir rocks and model volume, geometry, and boundary conditions which were interpreted using data collected over time from surface exploration, exploration drilling, logging, and well testing.

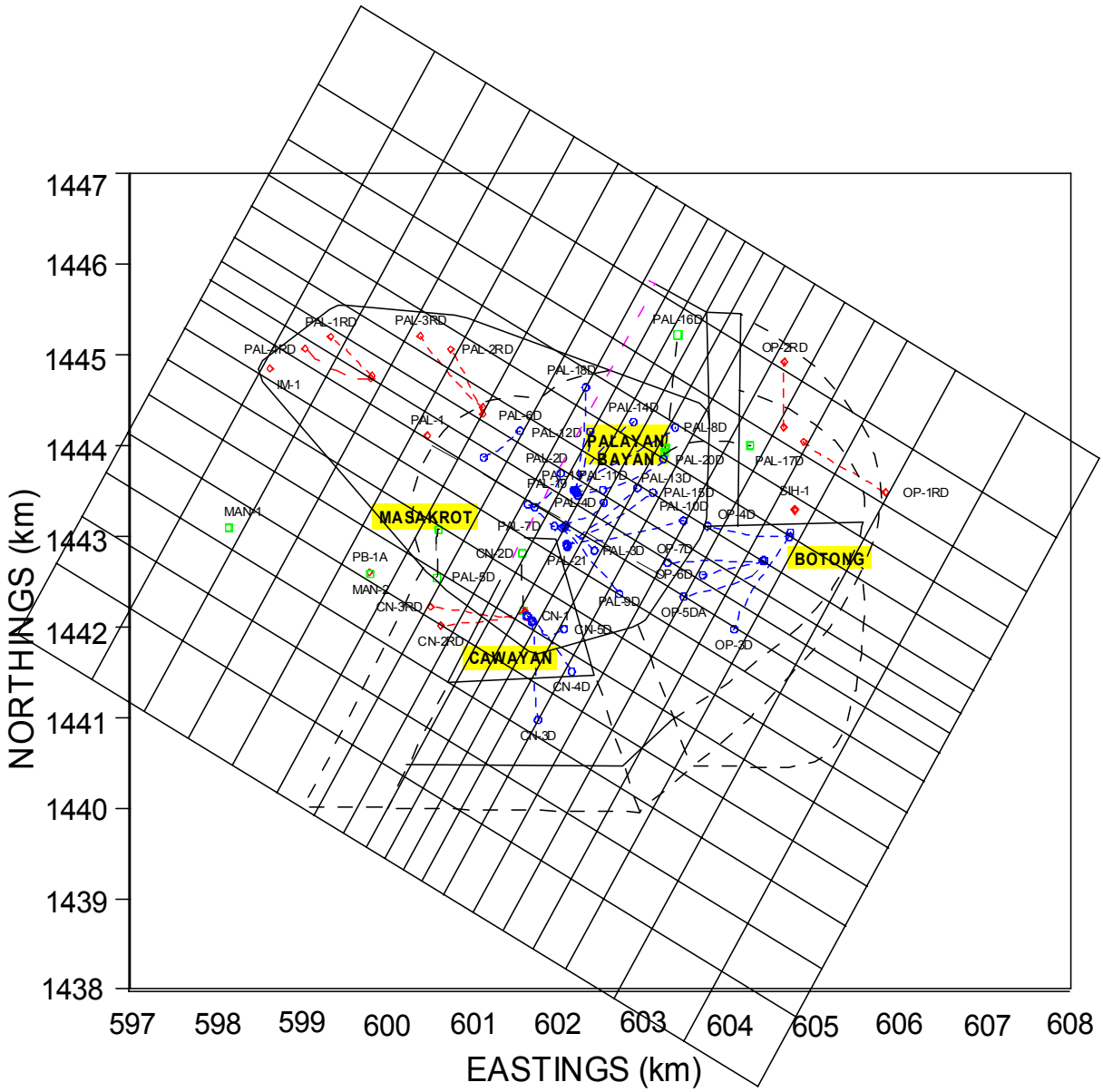



FIGURE 20: Bacon-Manito geothermal production field plan map showing welltracks and grid blocks for numerical simulation

## 5. CONCLUSIONS

Database tables were normalized during the development process with the objective of separating original data from derived data, and including author of interpretations and accuracy of measurements. The improved design used strictly ID numbers as primary keys to relate attributes from different tables.

Batch-loading of volumes of records using SQL*Loader was used to populate the tables. A control file, a feature of SQL*Loader, is written to describe ASCII data for loading, and columns and tables will accept the data. A number of simple database applications for data visualization using data dynamically from the database were developed.

These database applications include:

- **TPLOT** and **PPLOT** programs for generating dynamic X-Y plots of temperature and pressure against depth;

- **CONTOUR** program for generating dynamic temperature and pressure contours at different elevations;

- **TSERIES** program for monitoring wellhead pressure, massflow, and enthalpy trends of individual wells;

- Algorithms for projecting temperature, pressure, and well track to a given cross-sectional cut which is used to generate vertical sections;

- Simple Excel macro program to create a mesh for numerical modelling.

Oracle Developer forms for viewing welltest, temperature, and pressure data were created for graphical retrieval, review, and modification of data from the database. With these forms, the user does not need to learn how to use SQL+ programming language to retrieve, update, and save changes to the database.

The data visualization programs that were developed are very scalable. The embedded SQL+ statement in these programs can be easily modified for visualization of other reservoir data of interest. Generic mapping packages like GNUPLOT and GMT were used in the program to minimize the amount of code development.

The database system and applications developed to retrieve data blocks from the Oracle database to create dynamic line graphs and contour planes, Developer forms, and projection algorithms from within a UNIX shell script, are very useful tools in providing a basic graphic representation of a reservoir. With the database system and visualization tools, the interpretation time during data analysis can be increased by reducing wasted time spent in looking for, loading, and editing data.

This project focussed on the development of the database and database applications for data visualizations. The next phase will focus on the integration of data visualization modules developed in this project to the existing reservoir engineering database of PNOC-EDC; the linkage of reservoir engineering database with other geothermal databases to obtain a more comprehensive collection of data for model conceptualization; and the development of more graphical visualization tools possibly using the Windows platform.

## DEFINITION OF TERMS

| | |
|---|---|
| AWK | Aho, Kernighan, Weinberger programming language |
| GMT | Generic Mapping Tools |
| HTML | Hypertext Mark-up Language |
| HTTP | Hypertext Transfer Protocol |
| IIS | Internet Information Server |
| LAN | Local Area Network |
| PL/SQL | Procedural Language / Structured Query Language |
| RDBMS | Relational Database Management System |
| NAMRIA | National Mapping and Resource Information Authority |
| SQL | Structured Query Language |
| TCP/IP | Transmission Control Protocol / Internet Protocol |
| WAN | Wide Area Network |

## ACKNOWLEDGEMENTS

I would like to express my gratitude to the government of Iceland, the United Nations Geothermal Training Programme, and the management of PNOC-Energy Development Corporation for allowing me to participate in the 25th United Nations University Geothermal Training Programme. My heartfelt

gratitude to Dr. Ingvar Birgir Fridleifsson, director of the UNU Geothermal Training Programme, Mr. Lúdvík S. Georgsson, deputy director of the UNU Geothermal Training Programme, and Ms. Gudrún Bjarnadóttir for their steady guidance throughout our six months of training.

My sincerest thanks to the people who helped me with my project- especially Sigvaldi Thordason and Dr. Hjálmar Eysteinsson for helping me with GMT, and Ms. Maria-Victoria Gunnarson for meticulously proofreading my work.

Certainly this work would never have been realized without the help of my very diligent and dedicated supervisor, Hilmar Sigvaldasson. My special thanks to Hilmar. Many thanks to my co-Fellows, especially Ariel, Francisco, Martha, Li, and Zhang for making my project time a worthwhile endeavour.

This work is for my wife Cynthia, my son Joshua, and my newborn daughter Jamie Carylle who was born on June 10, 2003 while I was on training in Iceland.

## REFERENCES

Aho, A., Kernighan, B., and Weinberger, P., 1988: *The AWK programming language*. Addison-Wensley Publishing Company, USA, 210 pp.

Anderson, E.B., 1995: Geothermal data management case studies: Resource assessment. In: Anderson, E.B. (convenor), *Course on data management and related software in geothermal applications*. World Geothermal Congress 1995, IGA pre-congress course, Pisa, Italy, 13-28.

Bolsky, M., and Korn, D., 1995: *The new Korn shell. Command and programming language*. Prentice Hall PTR, USA, 400 pp.

Dougherty, D., 1992: *Sed and Awk*. O'Reilly & Associates, Inc., USA, 397 pp.

Grant, M.A., Donaldson, I.G., and Bixley, P.F., 1982: *Geothermal reservoir engineering*. Academic Press, New York, 369 pp.

Lakshman, B., 2000: *Oracle developer forms techniques*. SAMS, USA, 248 pp.

Moran, R., and Dimmick S., 1987: *SQL*Loader user guide*. Oracle Corporation, USA.

Sarmiento, Z.F., 2000: Physical monitoring II: high enthalpy geothermal systems. In: *Long-term monitoring of high- and low-enthalpy fields under exploitation.* World Geothermal Congress 2000, pre-congress short courses, Kokonoe, Japan, May, 23-56.

Sta.Ana, F.X.M., 1985: *A study on stimulation by air compression on some of the Philippine geothermal wells*. Geothermal Institute, University of Auckland, New Zealand, Report GEOTHERM.85.21.

Stefánsson, V., and Steingrímsson, B.S., 1980: *Geothermal logging I, an introduction to techniques and interpretation.* Orkustofnun, Reykjavík, report OS-80017/JHD-09, 117 pp.

Wessel, P., and Smith, H., 1998: *The Generic Mapping Tools (GMT), version 3. Technical Reference and Cookbook*. School of Ocean and Earth Science Technology/National Ocean and Atmospheric Administration, 77 pp.

Zapanta, O.R., Yglopaz, D.M., Lacanilao, A.M., Molina, P.O., and Sarmiento, Z.F., 1999: Developing a web-based geothermal database management system using thin-client architecture. *Proceedings of the 24th Workshop on Geothermal Reservoir Engineering, Stanford University, California,* 325-328.

# APPENDIX I: Data dictionary for PNOC-EDC reservoir engineering database

**Table PROJECT - defines a well location by geographical coordinates of a project**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Project*** | project ID | N 5 | PK | |
| Projcode | acronym of project name | VC 10 | Not null | BGPF, CLGP, LGPP, MCGP, MGPF, MLGP, MNGP, MPGP, NNGP, SLGP, SNGP |
| Address | mailing address of project | VC 50 | | Sorsogon, Sorsogon |
| X | X coordinate | N 10,1 | 999999999.9 | easting, (m) |
| Y | Y coordinate | N 10,1 | 999999999.9 | northing, (m) |
| Radius | radius of a circular area | N 5,1 | 9999.9 | (m) |
| Remarks | official name of a geothermal field | VC 50 | | Bacon-Manito Geothermal Production Field, Central Leyte Geothermal Project, Leyte Geothermal Production Field, Mt. Cagua Geothermal Project, Mt. Apo Geothermal Production Field, Mt. Labo Geothermal Project, Mt. Natib Geothermal Project, Mt. Pinatubo Geothermal Project, Northern Negros Geothermal Project, Southern Leyte Geothermal Prod. Field, Southern Negros Geothermal Production Field |

**Table SECTOR - defines a well location by geographical coordinates of a sector**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Sector*** | sector ID | N 5 | PK | Validate using Project |
| ***Project*** | project ID | N 5 | FK | |
| Sectcode | sector name | VC 10 | | Alto Peak (AP), Upper Mahiao (UM), Lower Mahiao (LM), etc. |
| X | X coordinate, easting | N 10,1 | 999999999.9 | easting, (m) |
| Y | Y coordinate, northing | N 10,1 | 999999999.9 | northing, (m) |
| Radius | radius of a circular area | N 5,1 | 9999.9 | (m) |
| Remarks | general comments about sector area | VC 50 | | Tongonan, Pataan, Palayang Bayan |

**Table PAD - defines well location by geographical coordinates of a pad area**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Pad*** | pad ID | N 5 | PK | Validate using Sector |
| ***Sector*** | sector ID | N 5 | FK | |
| Padcode | pad name | VC 10 | | 212, MGRD1B, OK8, PALRC, PALRA, 300B, etc. |
| X | X coordinate, easting | N 10,1 | 999999999.9 | easting, (m) |
| Y | Y coordinate, northing | N 10,1 | 999999999.9 | northing, (m) |
| Radius | radius of a circular area | N 5,1 | 9999.9 | (m) |
| Remarks | general comments about pad area | VC 50 | | |

**Table CELLAR - defines a well location by geographical coordinates of a cellar**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Cellar*** | cellar ID | N 5 | PK | Validate using Pad |
| ***Pad*** | pad ID | N 5 | FK | |
| Cellarcode | cellar name | VC 10 | | Deep cellar (D), standard cellar (S) |
| X | X coordinate, easting | N 10,1 | 999999999.9 | easting, (m) |
| Y | Y coordinate, northing | N 10,1 | 999999999.9 | northing, (m) |
| Radius | radius of a circular area | N 5,1 | 9999.9 | (m) |

**Table WELL - defines a well location by geographical coordinates of a well**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Well*** | well ID | N 10 | PK | |
| ***Pad*** | pad ID | N 5 | FK | Validate using Sector |
| Wellname | well name | VC 10 | | 101, MN1, MG23D, 4R4D, OP5DA |
| X | X coordinate of wellhead, easting | N 10,1 | 999999999.9 | easting, (m) |
| Y | Y coordinate of wellhead, northing | N 10,1 | 999999999.9 | northing, (m) |
| Z | elevation of wellhead | N 5,1 | 9999.9 | elevation, from sea level (m) |
| Measdepth | total measured depth of the well | N 5,1 | 9999.9 | (m) |
| Whead | wellhead ID | N 10, 1 | 999999999.9 | indicate wells drilled on same wellhead but with different tracks |

**Table DRILLJOB - contains details of a drilling activity done on a well**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Drilljob*** | drilling ID | N 5 | PK | |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| Activity | type of drilling activity | VC 5 | | New well drilling (NW), sidetrack (ST), tophole (TH), workover (WO), workover and acidizing (WOA), casing relining (RL), redrill to make new well (RD), reentry (RE), perforation and acidizing (PFA), perforation (PF), hydraulic fracturing (HF), air lifting (AIR), plug and abandon (PA). |
| Date_start | date of start of drilling | D | | Format : mm/dd/yyyy |
| Date_end | date of end of drilling | D | | Format : mm/dd/yyyy |
| Depth_start | starting depth of drilling activity | N 4,1 | 9999.9 | |
| Depth_end | final depth of drilling activity | N 4,1 | 9999.9 | |
| Rig | name of rig used | VC 10 | | rig 9 |
| Purpose | reason for doing activity | VC 150 | | to increase reinjection capacity for Mahanagdong |
| Rkbchf | distance from rotary table to wellhead | N 4,2 | 99.1 | |

**Table DS - contains results of deviation surveys conducted on a well**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Ds*** | directional survey ID | N 8 | PK | |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| Dsmd | directional survey, measured depth | N 5,1 | 9999.9 | measured depth, (m) |
| Dsvd | directional survey, vertical depth | N 5,1 | 9999.1 | vertical depth, (m) |
| Dsmn | directional survey, meters northing | N 10,1 | 999999999.9 | northing, (m) |
| Dsme | directional survey, meters easting | N 10,1 | 999999999.9 | easting, (m) |
| Dogleg | dogleg severity | N 5,1 | 999999999.9 | |
| Drift | drift angle | N 4,2 | 99.99 | |
| User_initial | id of user who inputted record | VC 8 | | I5108 |
| Date_start | date of inputting | D | | Format : MM-DD-YYYY |
| User_change | id of user who changed record | VC 8 | | I5108 |
| Date_change | date of change | D | | Format : MM-DD-YYYY |

**Table COMPLETION - contains completion test results for a well**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Completion*** | completion test ID | N 5 | PK | |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| Date_start | date of start of drilling | D | | Format : mm/dd/yyyy |
| Date_end | date of end of drilling | D | | Format : mm/dd/yyyy |
| Injectivity | injectivity index | N 4,1 | 999.9 | (li/s-Mpa) |
| Injdepth | tool setting depth during injection test | N 5,1 | 9999.9 | measured depth, (m) |
| Maxwhp | maximum whp during injectivity test | N 5,2 | 999.9 | (MPag) |
| kh | permeability thickness | N 8,3 | 999.99999 | (Darcy-meter) |
| Skin | skin, positive value shows damage while negative value indicates enhanced permeabilty | N 4,1 | 999.9 | 5, -10 |

**Table CASING - contains casing configuration of a well**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Casing*** | casing ID | N 5 | PK | |
| ***Drilljob*** | drilling ID | N 5 | FK | Validate using Drilljob |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| ***CBL*** | sonic log ID | N 5 | FK | |
| Date_start | date of start of setting of casing | D | | F : mm/dd/yyyy |
| Date_end | date of end of setting of casing | D | | F : mm/dd/yyyy |
| Csgtype | type of casing | C 2 | | anchor casing (AC), blank liner (BL), intermediate casing (IC), prod. casing (PC), surface casing (SC), slotted liner (SL) |
| OD | outer diameter | N 4,1 | 999.9 | (mm) |
| ID | inner diameter | N 4,1 | 999.9 | (mm) |
| Top | top of casing | N 5,1 | 9999.9 | vertical depth, (m) |
| Bottom | bottom of casing | N 5,1 | 9999.9 | vertical depth, (m) |

**Table PZ - details on permeable zones of wells delineated from different tests**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Well*** | well ID | N 5 | FK | Validate using Project |
| ***Completion*** | completion test ID | N 5 | FK | Validate using Completion |
| Pzfrom | payzone interval from | N 5,1 | 9999.9 | measured depth, (m) |
| Pzto | payzone interval to | N 5,1 | 9999.9 | measured depth, (m) |
| Pzcode | classification of permeable zone | VC 2 | | major (MJ), minor (MN) |
| Source | description of data source | VC 3 | | completion test (CT), electronic-line survey (TP) |
| Remarks | description of feedzone | VC 30 | | coincides with Kinabkaban fault |

**Table WELLSTIM - details on stimulation jobs performed on wells**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Wellstim*** | well stimulation ID | N 10 | PK | |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| Equiptype | type of equipment used | VC 12 | | Bauer, Sullair |
| Date_start | date of start of stimulation activity | D | | F : mm/dd/yyyy |
| Date_end | date of end of stimulation activity | D | | F : mm/dd/yyyy |
| Ophrs | operating hours | N 5, 2 | 1000.0 | time, (hr) |
| Oper | id of operator who performed the job | VC 25 | | |
| Engr | id of engineer who supervised the job | VC 25 | | I5108 |
| Maxpress | maximum pressure | N 4, 1 | 9999.9 | (Mpag) |
| Fuelit | fule consumed | N 5 | 99999.0 | liters, (l) |
| Remarks | general remarks about stimulation job | VC 100 | | well succesfully discharged |

**Table WELLUSE - details about how a well is utilized**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Welluse*** | well use ID | N 5 | PK | |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| Welluse | current use of the well | C 2 | | production (P), reinjection (RI), monitoring (M), temperature gradient (TG), exploratory (EX) |
| Wellstat | current status of the well | C 2 | | shut (S), flowing (F), on bleed (OB), medium term discharge (MTD), vertical discharge (VD), workover (WO), plugged and abandoned (PA) heat-up (H) |
| Date | date of well utilization record | D | | Format : mm/dd/yyyy |
| Time | time of well utilization record | Time | 12:30:30 | Format : hh:mm:ss |
| Whp | wellhead pressure | N 4,1 | 999.9 | (MPag) |

**Table BOM - processed bore output measurement data**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Bom*** | bore output measurements id | N 10 | PK | |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| Date_start | data of start of bore output test | D | | Format : mm/dd/yyyy |
| Opening | wellhead condition, back pressure plate series | VC 3 | | fullbore (FB), throtled (THR), BPP series : (A1), (B2), etc. |
| WHP | whp during bore output test | N 5, 2 | 999.9 | (MPag) |
| Enthalpy | computed enthalpy | N 4 | 9999.0 | (kJ/kg) |
| Waterf | computed waterflow | N 4, 1 | 99.9 | (kg/s) |
| Massf | computed massflow | N 4, 1 | 99.9 | (kg/s) |
| Steamf | computed steamflow | N 4, 1 | 99.9 | (kg/s) |
| Power | computed power | N 4, 1 | 99.9 | (MWe) |
| Flag | flags measurements used for curves | N 2 | 99 | stable (S) |

**Table LOGF** - general information about a logging activity

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Log*** | log ID | N 10 | PK | Validate using Log |
| ***Well*** | well ID | N 5 | FK | Validate using Well |
| ***Tool*** | tool id | N 5 | FK | Validate using Tool |
| Condition | well condition while survey is being done | VC 30 | | 1 day shut; after post acidizing completion test; waterloss |
| Date_start | date of start of logging activity | D | | Format : mm/dd/yyyy |
| Date_end | date of end of logging activity | D | | Format : mm/dd/yyyy |
| Shift | shift in depth to correct reference depth | N 4,1 | 999.9 | m |
| Logdirection | direction of log | VC 1 | | upward (U), downward (D), stationary (S) |
| Engr | id of engineer in charge of log | VC 25 | | I5108 |

**Table DDATA - logging data with respect to depth**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Log*** | log ID | N 10 | FK | Validate using Log |
| Depth | depth of log | N 5 | 9999.9 | reduced depth (mRSL) |
| Value | value recorded in log | N 5, 2 | 999.99 | temperature, (°C); pressure, (MPag) |

**Table TDATA - logging data with respect to time**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Log*** | log ID | N 10 | FK | Validate using Log |
| Time | time of measurement | Time | 12:30:30 | Format: hh:mm:ss |
| Value | value recorded in log | N 5, 2 | 999.99 | temperature, (°C); pressure, (MPag) |

**BLOCKAGE - Table of blockages detected inside the well during surveys and tests**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Well*** | well ID | N 5 | PK | Validate using Well |
| ***Log*** | log ID | N10 | FK | Validate using Log |
| ***Tool*** | tool id | C 3 | FK | Validate using Tool<br>dummy (DM), Go-devil (GD), Kylen sampler (KA), Kuster temperature (KT), Kuster pressure (KP), lead impression block (LIB), sinker bar (SB), scraper (SC) |
| Surdate | date of survey | D | | Format : mm/dd/yyyy |
| Tooldiam | outer diameter of tool used in survey | N 3,1 | 99.9 | (mm) |
| Mcd | maximum clear depth tagged during survey | N 5,1 | 9999.9 | measured depth, (m) |
| Status | short description of nature of blockage | VC 50 | | calcite, anhydrite, etc. |

**Table TOOL - information on tools used during logging**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Tool*** | tool ID | N 8 | PK | |
| Toolname | name of the tool | VC 8 | NOT NULL | |
| Tooltype | type of instrument | VC 3 | NOT NULL | temperature (T), pressure (P), spinner (S), flowmeter (FM), sonic (VDL, CBL or CET), neutron (N), gamma (G), resistivity (R), caliper (XY or CIC), casing collar locator (CCL), freepoint (FP) |
| Brand | brand name of the tool | VC 20 | | |
| Description | specifications of tool used | VC 40 | | |

**Table CALIBRATE - results of calibration of instrument used in logging**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Calibrate*** | calibration_ID | N 8 | PK | |
| ***Tool*** | tool_ID | N 8 | FK | Validate using Tool |
| Datecalib | date of calibration | D | | Format : mm/dd/yyyy |
| Rangemin | minimum range of tool after calibration | N 4,1 | 999.9 | |
| Rangemax | maximum range of tool after calibration | N 4,1 | 999.9 | |
| Reference | standard used in calibration | VC 20 | | |

**Table DEFLECT- equations from calibration of instrument used in logging**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Calibrate*** | calibration_ID | N 5 | FK | |
| Temp | temperature | N 5 | FK | |
| Press | pressure | D | | |
| Error | error | | | |

**Table REPAIRS - details of repair of instruments used in logging**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Calibrate*** | calibration ID | N 5 | FK | Validate using Calibrate |
| Date | date of repair | D | | Format : mm/dd/yyyy |
| Result | results of repair | VC 50 | | |

**Table CBL - processed cement bond, variable density, and cement evaluation log information**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***CBL*** | sonic log ID | N 5 | PK | |
| ***Log*** | log ID | N 5 | FK | Validate using Logf |
| CBLfrom | start of cbl interval (top) | N 5,1 | 9999.9 | measured depth, (m) |
| CBLto | end of cbl interval (bottom) | N 5,1 | 9999.9 | measured depth, (m) |
| Bondpct | percent of bond index | N 3,1 | 99.9 | (%) |

**Table TRACKPOLY - contains polynomial equations describing the welltrack of well against reduced depth**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| ***Well*** | well ID | N 5 | 99999.0 | Validate using Well |
| Kopmod | kick off point, modified to simplify polynomials | N 6, 2 | 10000.0 | |
| Accuracy | accuracy of polynomial equations | N 2 | 99.0 | accuracy of 10 means accurate within 10 meters, (m) |
| Date_eqn | date of creation of polynomial equations | D | | Format : mm/dd/yyyy |
| Mevsrd_0 | coef of deg 0 polynomial for easting vs. RD | N 10, 2 | 99999999.99 | |
| Mevsrd_1 | coef of deg 1 polynomial for easting vs. RD | N 12, 10 | 99.9999999999 | |
| Mevsrd_2 | coef of deg 2 polynomial for easting vs. RD | N 17, 15 | 99.9999999999999 | |
| Mevsrd_3 | coef of deg 3 polynomial for easting vs. RD | N 19, 17 | 99.9999999999999 | |
| Mevsrd_4 | coef of deg 4 polynomial for easting vs. RD | N 22, 20 | 99.9999999999999 | |
| Mnvsrd_0 | coef of deg 0 polynomial for northing vs. RD | N 10, 2 | 99999999.99 | |
| Mnvsrd_1 | coef of deg 1 polynomial for northing vs. RD | N 11, 9 | 99.999999999 | |
| Mnvsrd_2 | coef of deg 2 polynomial for northing vs. RD | N 16, 14 | 99.9999999999999 | |
| Mnvsrd_3 | coef of deg 3 polynomial for northing vs. RD | N 19, 17 | 99.9999999999999 | |
| Mnvsrd_4 | coef of deg 4 polynomial for northing vs. RD | N 22, 20 | 99.9999999999999 | |
| Mmdvsrd_0 | coef of deg 0 polynomial for MD vs. RD | N 12, 8 | 9999.99999999 | |
| Mmdvsrd_1 | coef of deg 1 polynomial for MD vs. RD | N 11, 9 | 99.999999999 | |
| Mmdvsrd_2 | coef of deg 2 polynomial for MD vs. RD | N 15, 13 | 99.9999999999999 | |
| Mvdvsrd_0 | coef of deg 0 polynomial for VD vs. RD | N 6, 2 | 9999.99 | |
| Mvdvsrd_1 | coef of deg 1 polynomial for VD vs. RD | N 3, 2 | 9.99 | |
| Source | ID of author of equations | VC 10 | | |

***Note : vertical depth (VD), measured depth (MD), reduced depth (RD)***

**Table STABLETEMP - contains interpreted stable temperatures for wells**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| Well | well ID | N 5 | 99999.0 | Validate using Well |
| Date_int | date interpretation was made and inputted to dbase | D | | Format : mm/dd/yyyy |
| Reducedepth | reduced depth where temperature is given | N 5 | 99999.0 | |
| Temperature | interpreted stable temperature | N 4, 1 | 9999.9 | |
| Source | ID of author of interpreted temperatures | VC 15 | | |

**Table STABLEPRES - contains interpreted stable pressures for wells**

| Column name | Description | Data type | Constraint | Remarks |
|---|---|---|---|---|
| Well | well ID | N 5 | 99999.0 | Validate using Well |
| Date_int | date of interpretation and inputting | D | | Format : mm/dd/yyyy |
| Reducedepth | reduced depth where pressure is given | N 5 | 99999.0 | |
| Pressure | interpreted stable pressure | N 4, 1 | 9999.9 | |
| Source | ID of author of interpreted pressures | VC 15 | | |

## APPENDIX II: UNIX shell script for temperature plotting program TPLOT

```
#! /bin/ksh
#
#    T  P L O T
#    Year created 2003
#    Jaime and Hilmar
#    Script using S Q L + and G N U P L O T © to get Temperature logs
#    from Oracle tables WELL, DDATA, and LOGF
#
echo "Program T P L O T to get Temperature logs from Oracle"

if [[$# -lt 3 || $1 = "-h”]]
then
echo "Usage:  tplot wellname fromdate(DD-MM-YYYY) todate(DD-MM-YYYY)" 1>&2
  exit 1
fi
echo "
set echo off
set feedback off
set flush off
set term off
set verify off
set newpage 0
set lines 130
set pages 0

break on log on notes skip 2

spool  t
select distinct a.value, a.depth, b.date_start, b.log, to_char (b.date_start,'DD-MM-YYYY')||'-
'||b.condition Notes
from ddata a, logf b, well c
where a.log = b.log
and b.well = c.well
and b.condition = ‘T’
and c.wellname = '$1'
and b.date_start >=to_date('$2', 'DD-MM-YYYY')
and b.date_start<=to_date('$3', 'DD-MM-YYYY')
order by b.date_start, b.log, a.depth;
spool off

quit" > t.sql
```

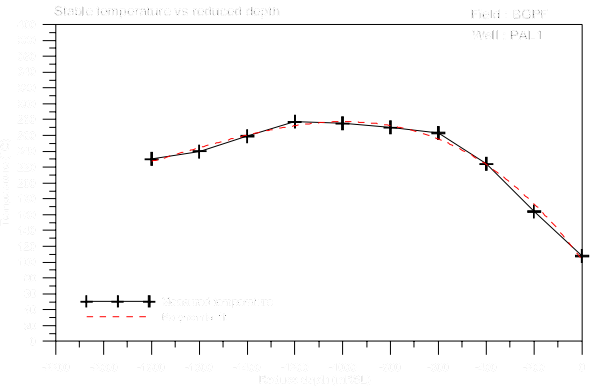
```
sqlplus / @t.sql > /dev/null

#     uses Sed to remove blank lines
sed 's/^  *$//' t.lst > tt.lst
#     saves changes to temporary file bb.lst and renames bb.lst to t.lst
awk 'NR >= 1 {print $1, $2, $3, $4, $5}' tt.lst > ttt.lst
mv ttt.lst t.lst

#   Create GNUPLOT commands file
echo "
set xdata
set ydata
set title 'Well $1'
set yrange [0:3000] reverse
set xrange [0:400]
set xtics 100
set mxtics 2
set ytics 200
set mytics 2
set xlabel 'Temperature (˚C)'
set ylabel 'Depth (mVD)'
set key right bottom outside
set keytitle 'Survey condition'
set size ratio 0 1, 1
show size" > sk.tp

awk 'BEGIN{printf("plot ");j=0}
NF>4 {a[j]=$5;j++}
END{for (i=0;i<j-1;i++) {
printf("\"t.lst\" index  %d:%d title '\"%s'\" with linespoints,",i,i,a[i])}
printf("\"t.lst\" index  %d:%d title '\"%s'\" with linespoints \npause -1",i,i,a[i])}' t.lst >> sk.tp
gnuplot sk.tp
```

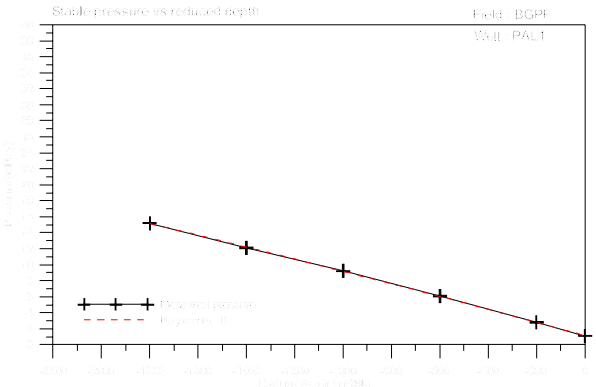
## APPENDIX III: UNIX shell script for pressure plotting program PPLOT

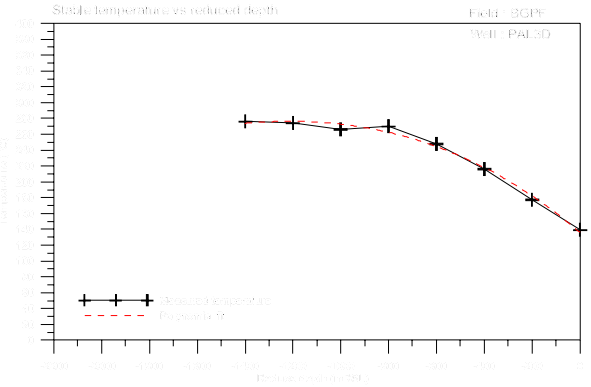
```
#! /bin/ksh
#
#     P  P L O T
#     Year created 2003
#     Jaime and Hilmar
#     Script using S Q L + and G N U P L O T (C) to get Pressure logs from Oracle
#     from Oracle tables WELL, DDATA, and LOGF
#
echo "Program P P L O T  to get Pressure logs from Oracle"

if [[$# -lt 3 || $1 = "-h”]]
then
echo "Usage:  tplot wellname fromdate(DD-MM-YYYY) todate(DD-MM-YYYY)" 1>&2
  exit 1
fi

echo "
set echo off
```

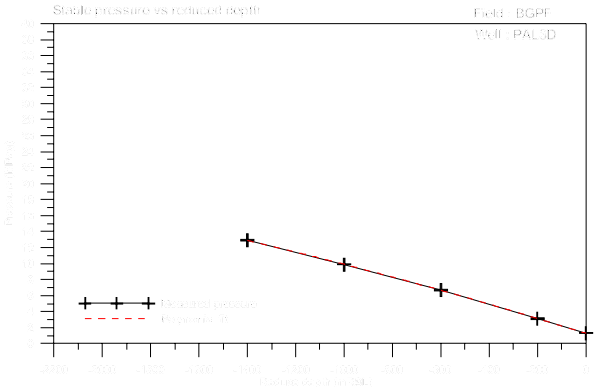
```
set feedback off
set flush off
set term off
set verify off
set newpage 0
set lines 130
set pages 0

break on log on notes skip 2

spool p
select distinct a.value, a.depth, b.date_start, b.log, to_char (b.date_start,'DD-MM-YYYY')||'-
'||b.condition Notes
from ddata a, logf b, well c
where a.log = b.log
and b.well = c.well
and b.condition = 'P'
and c.wellname = '$1'
and b.date_start >=to_date('$2', 'DD-MM-YYYY')
and b.date_start<=to_date('$3', 'DD-MM-YYYY')
order by b.date_start, b.log, a.depth;
spool off

quit" > p.sql
sqlplus / @p.sql > /dev/null

#     uses Sed to remove blank lines
sed 's/^  *$//' p.lst > pp.lst
#     saves changes to temporary file bb.lst and renames bb.lst to t.lst
awk 'NR >= 1 {print $1, $2, $3, $4, $5}' pp.lst > ppp.lst
mv ttt.lst t.lst

# Create GNUPLOT commands file
echo "
set xdata
set ydata
set title 'Well $1'
set yrange [0:3000] reverse
set xrange [0:400]
set xtics 100
set mxtics 2
set ytics 200
set mytics 2
set xlabel 'Pressure (MPag)'
set ylabel 'Depth (mVD)'
set key right bottom outside
set keytitle 'Survey condition'
set size ratio 0 1, 1
show size" > sk.tp

awk 'BEGIN{printf("plot ");j=0}
NF>4 {a[j]=$5;j++}
END{for (i=0;i<j-1;i++) {
printf("\"p.lst\" index  %d:%d title '\"%s'\" with linespoints,",i,i,a[i])}
printf("\"p.lst\" index  %d:%d title '\"%s'\" with linespoints \npause -1",i,i, a[i])}' p.lst >> sk.tp
gnuplot sk.tp
```

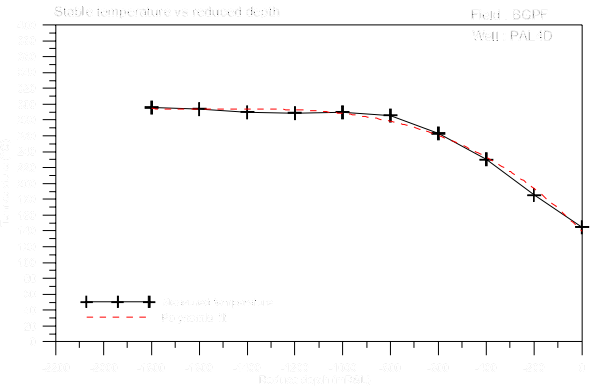
**APPENDIX IV: Interpreted stable temperature and pressure and corresponding polynomial fits of wells in Bacon-Manito geothermal production field**

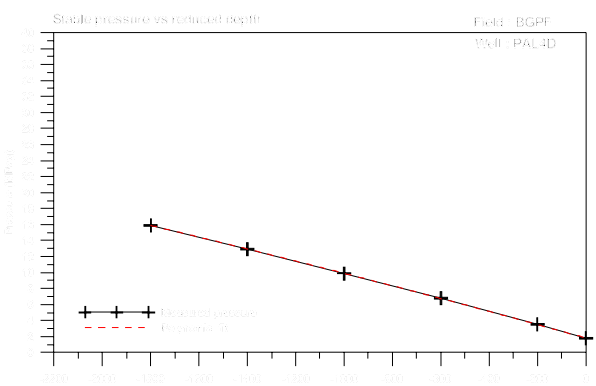


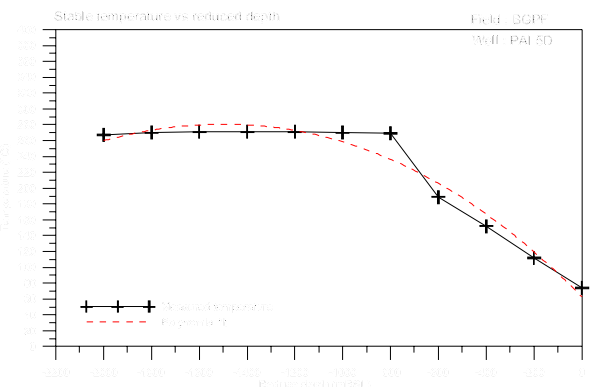



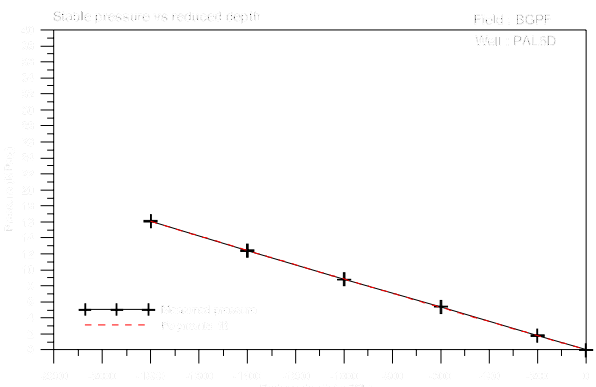



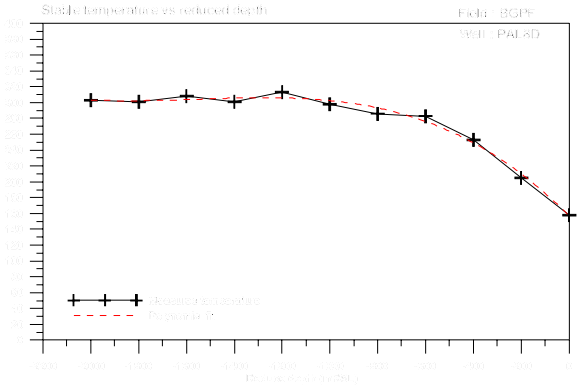



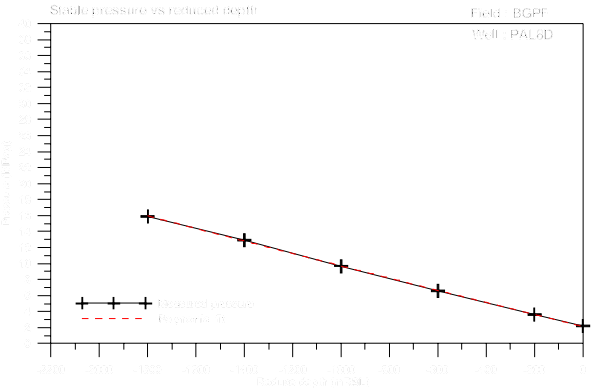



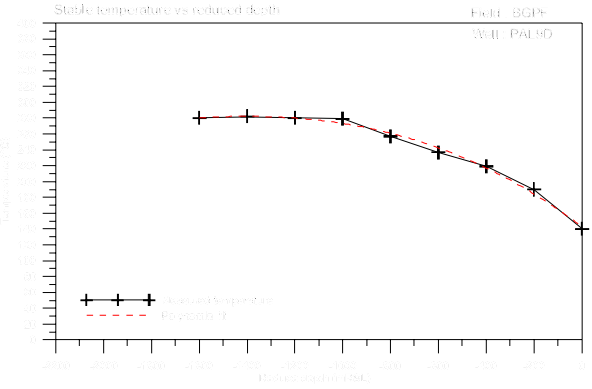



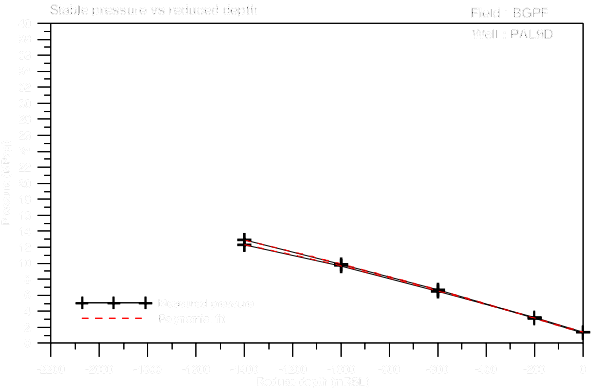



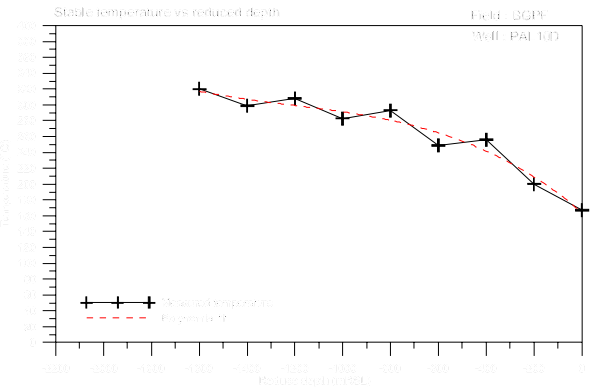



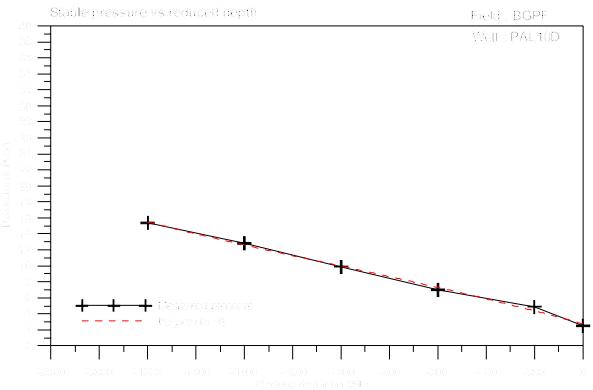

Stable temperature vs reduced depth
Field : BGPF
Well : PAL6D

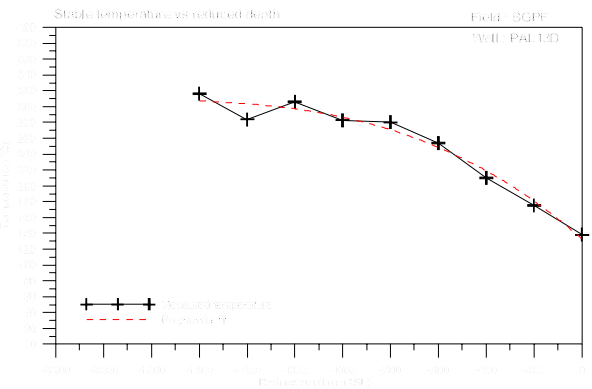

Stable pressure vs reduced depth
Field : BGPF
Well : PAL6D

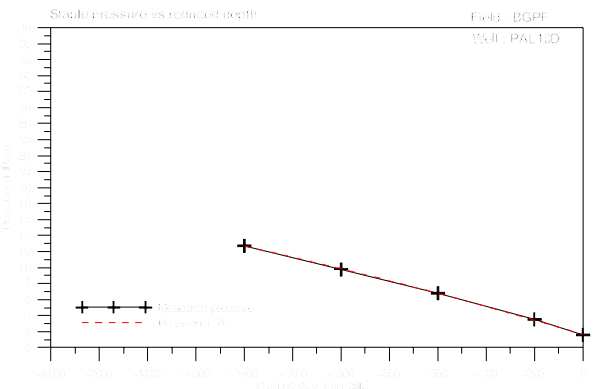

Stable temperature vs reduced depth
Field : BGPF
Well : PAL9D

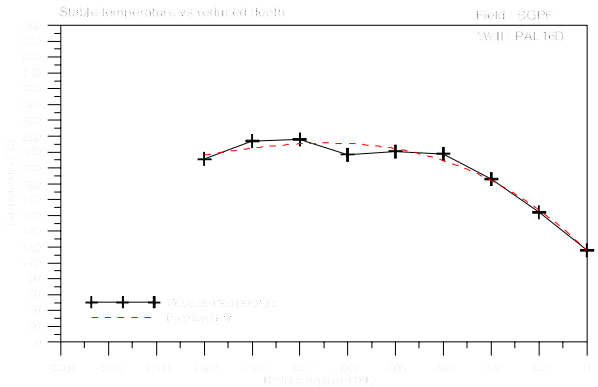

Stable pressure vs reduced depth
Field : BGPF
Well : PAL9D

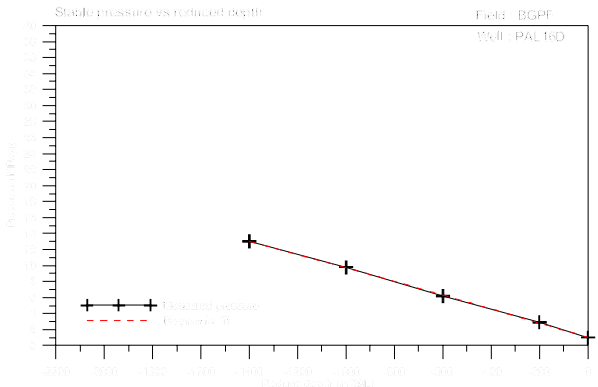

Stable temperature vs reduced depth
Field : BGPF
Well : PAL10D

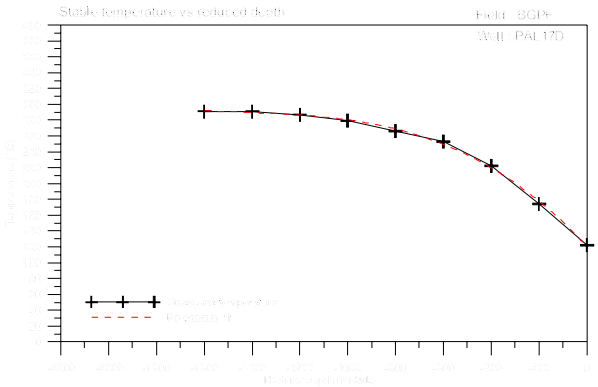

Stable pressure vs reduced depth
Field : BGPF
Well : PAL10D

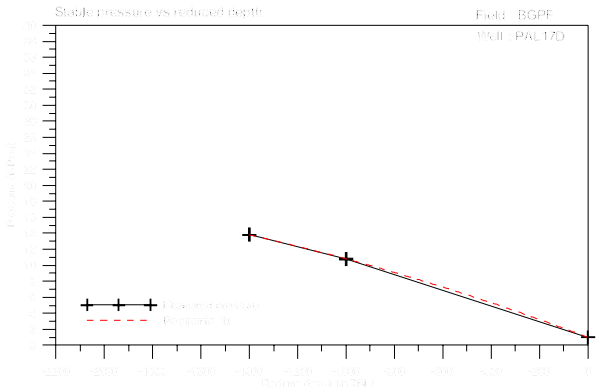

Stable temperature vs reduced depth
Field : BGPF
Well : PAL13D

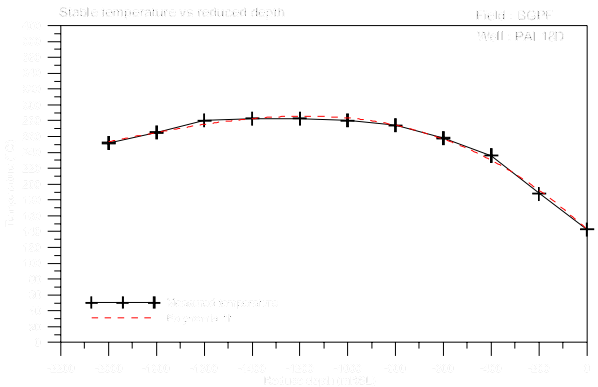

Stable pressure vs reduced depth
Field : BGPF
Well : PAL13D

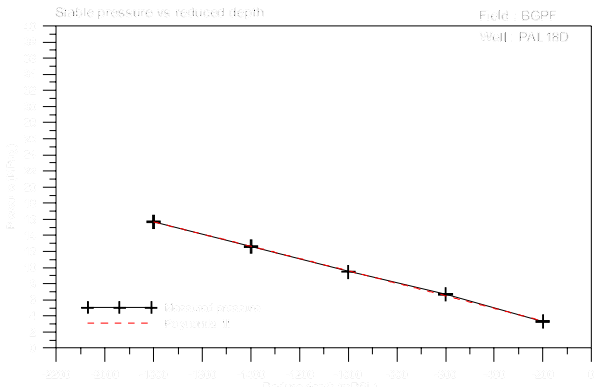

Stable temperature vs reduced depth
Field : BGPF
Well : PAL16D

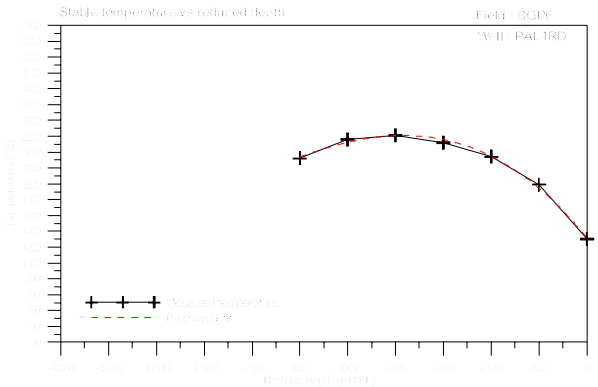

Stable pressure vs reduced depth
Field : BGPF
Well : PAL16D

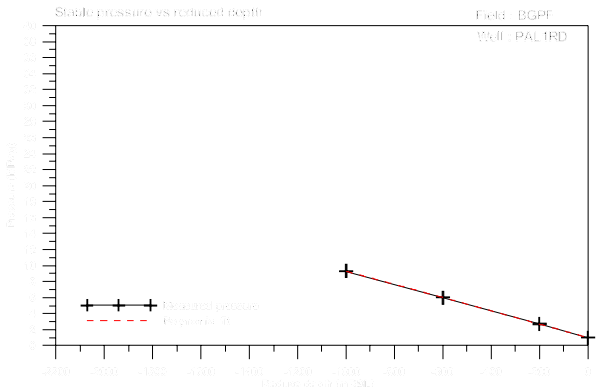

Stable temperature vs reduced depth
Field : BGPF
Well : PAL17D

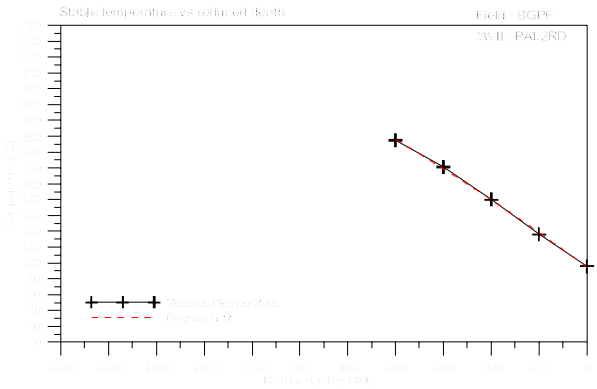

Stable pressure vs reduced depth
Field : BGPF
Well : PAL17D

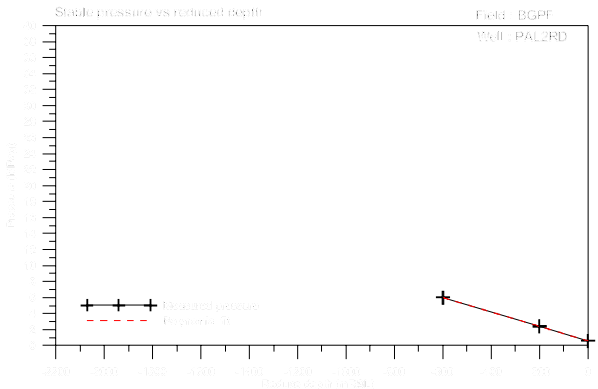

Stable temperature vs reduced depth
Field : BGPF
Well : PAL18D

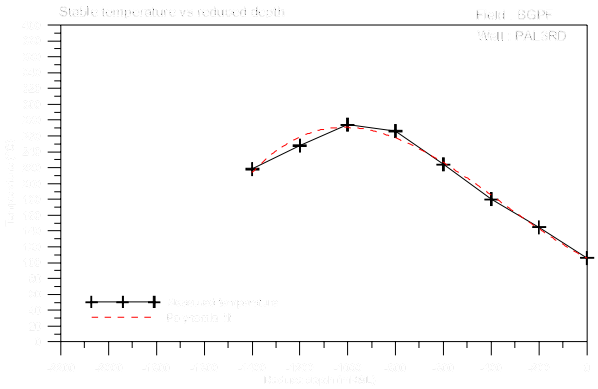

Stable pressure vs reduced depth
Field : BGPF
Well : PAL18D

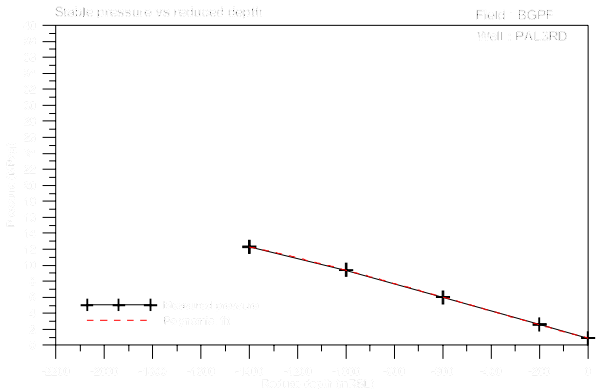

Stable temperature vs reduced depth
Field : BGPF
Well : PAL18D

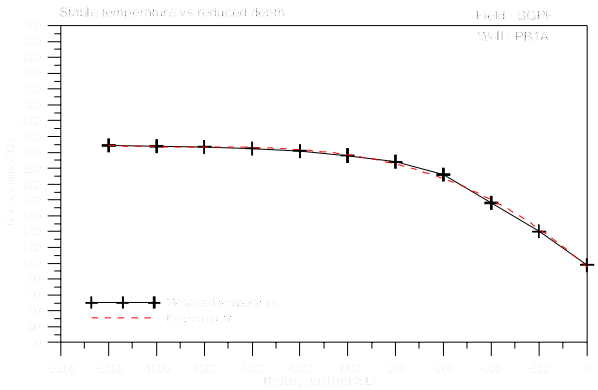

Stable pressure vs reduced depth
Field : BGPF
Well : PAL18D

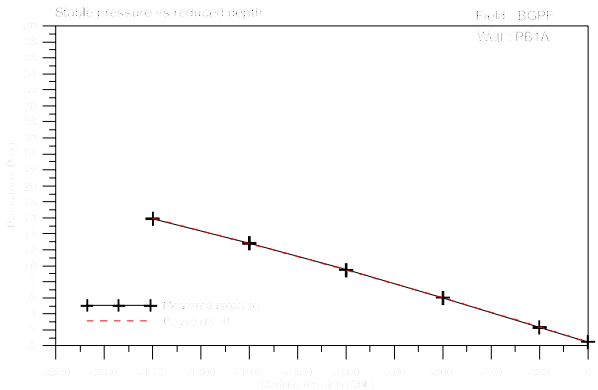

Stable temperature vs reduced depth
Field : BGPF
Well : PAL2RD

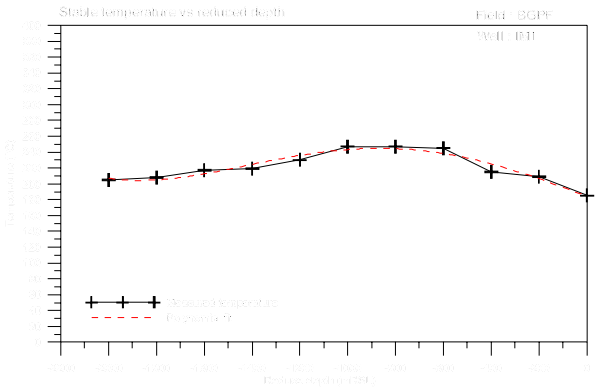

Stable pressure vs reduced depth
Field : BGPF
Well : PAL2RD

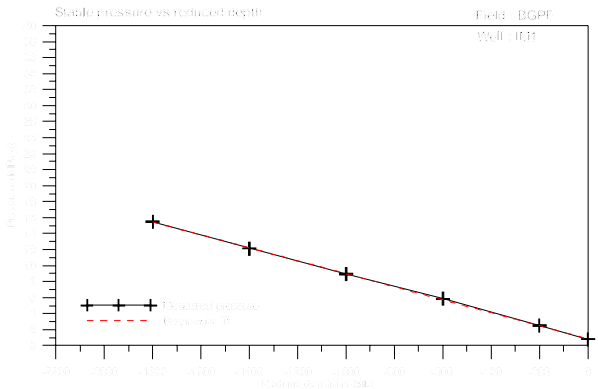

Stable temperature vs reduced depth
Field : BGPF
Well : PAL3RD

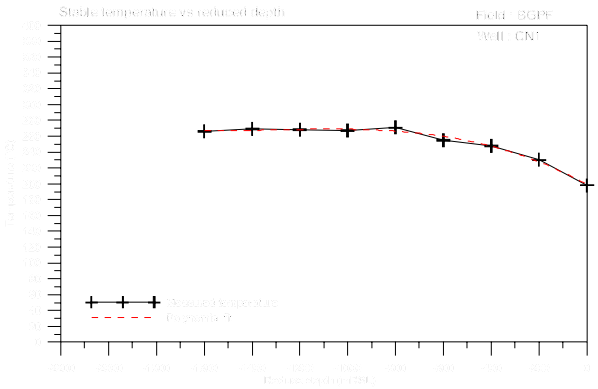

Stable pressure vs reduced depth
Field : BGPF
Well : PAL3RD

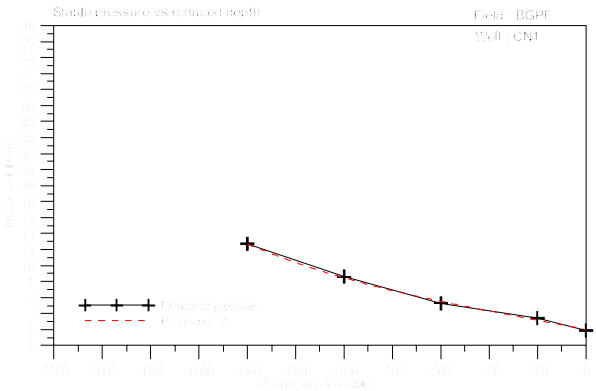

Stable temperature vs reduced depth
Field : BGPF
Well : PB1A

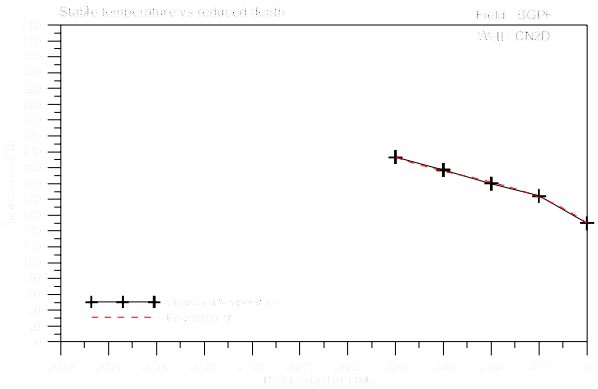

Stable pressure vs reduced depth
Field : BGPF
Well : PB1A

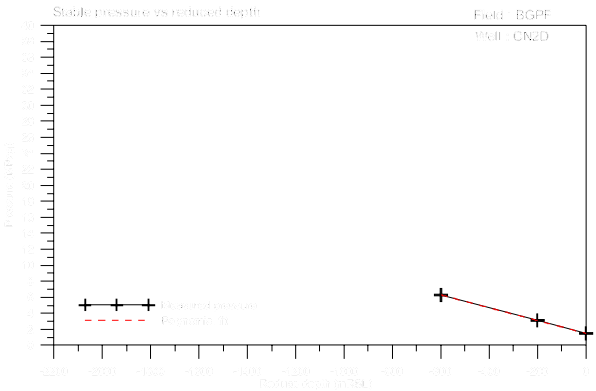

Stable temperature vs reduced depth
Field : BGPF
Well : PM1

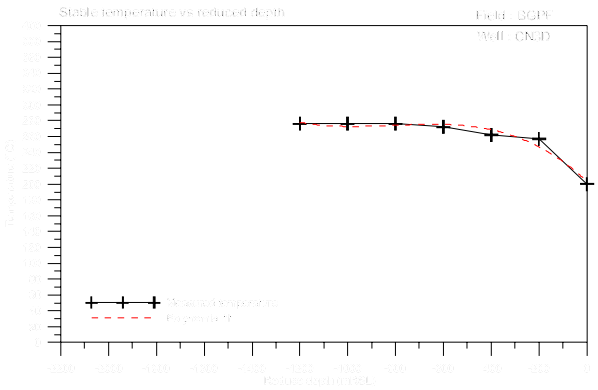

Stable pressure vs reduced depth
Field : BGPF
Well : PM1

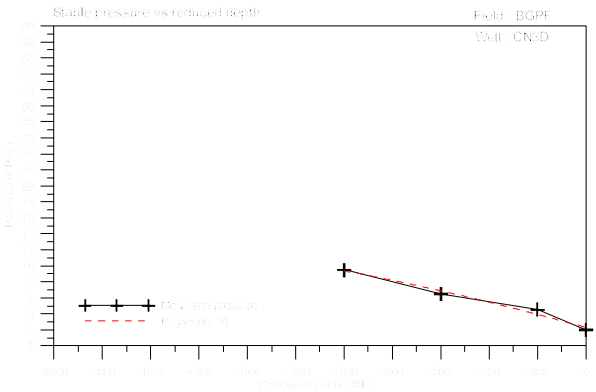

Stable temperature vs reduced depth
Field : BGPF
Well : CN1

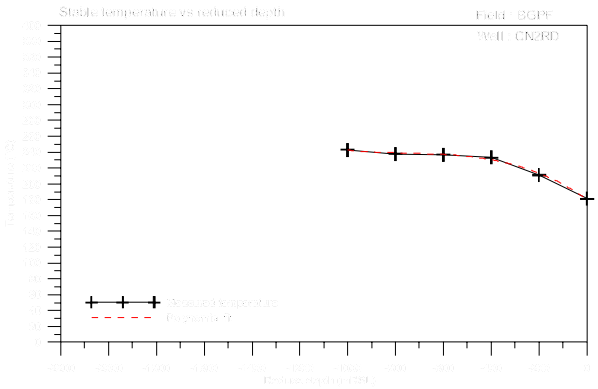

Stable pressure vs reduced depth
Field : BGPF
Well : CN1

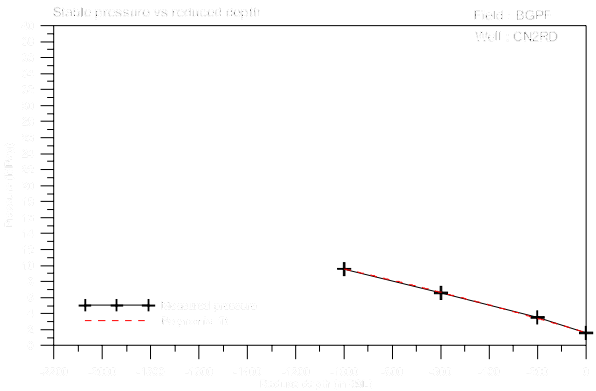

Stable temperature vs reduced depth
Field : BGPF
Well : CN2D

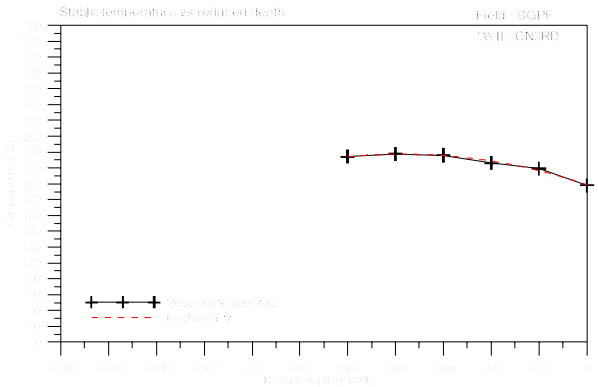

Stable pressure vs reduced depth
Field : BGPF
Well : CN2D

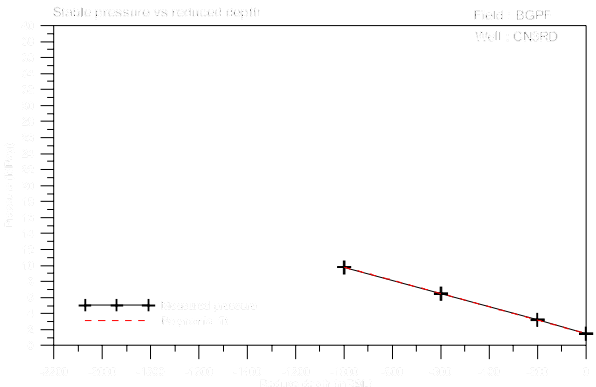

Stable temperature vs reduced depth
Field : BGPF
Well : CN3D

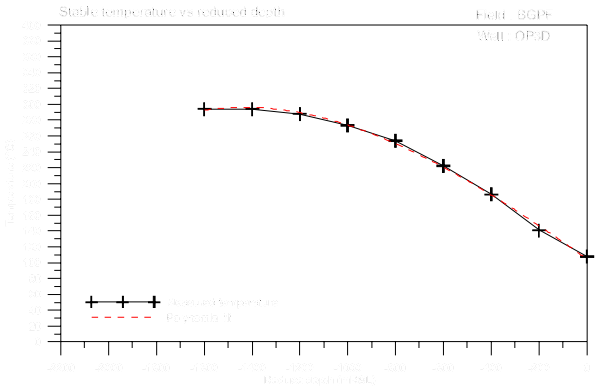

Stable pressure vs reduced depth
Field : BGPF
Well : CN3D

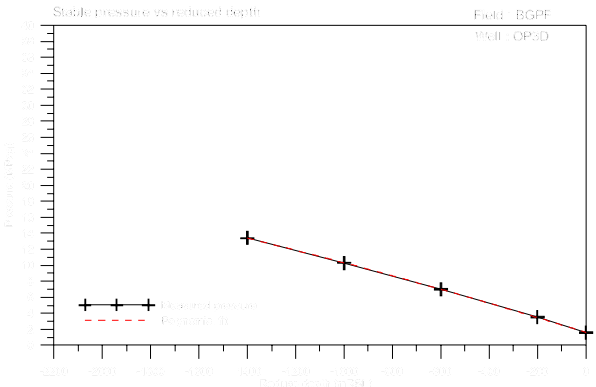

Stable temperature vs reduced depth
Field : BGPF
Well : CN2RD

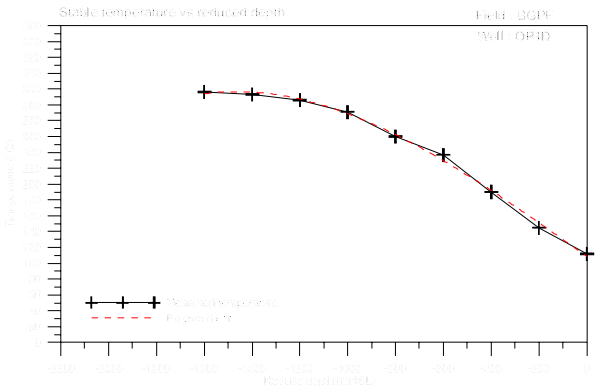

Stable pressure vs reduced depth
Field : BGPF
Well : CN2RD

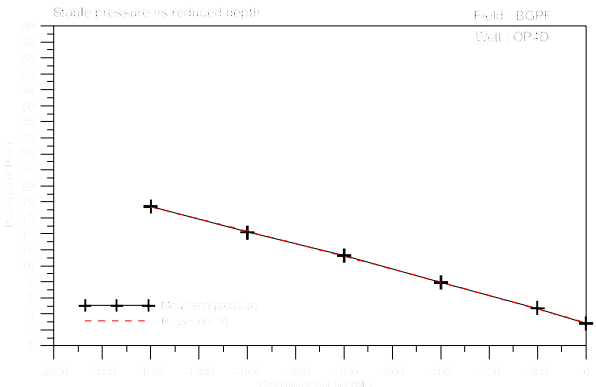

Stable temperature vs reduced depth
Field: BGPF
Well: CN3RD

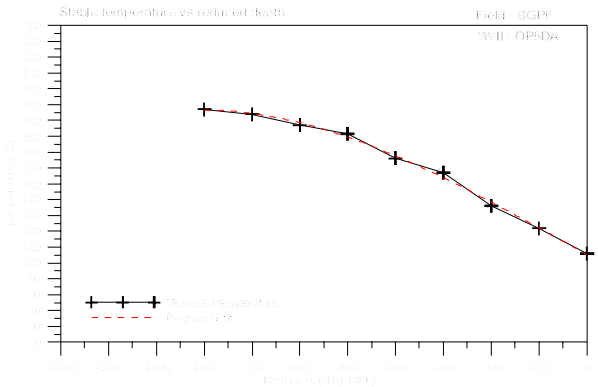

Stable pressure vs reduced depth
Field: BGPF
Well: CN3RD

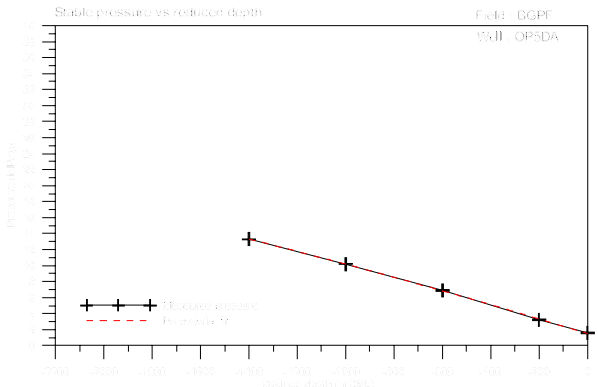

Stable temperature vs reduced depth
Field: BGPF
Well: OP3D

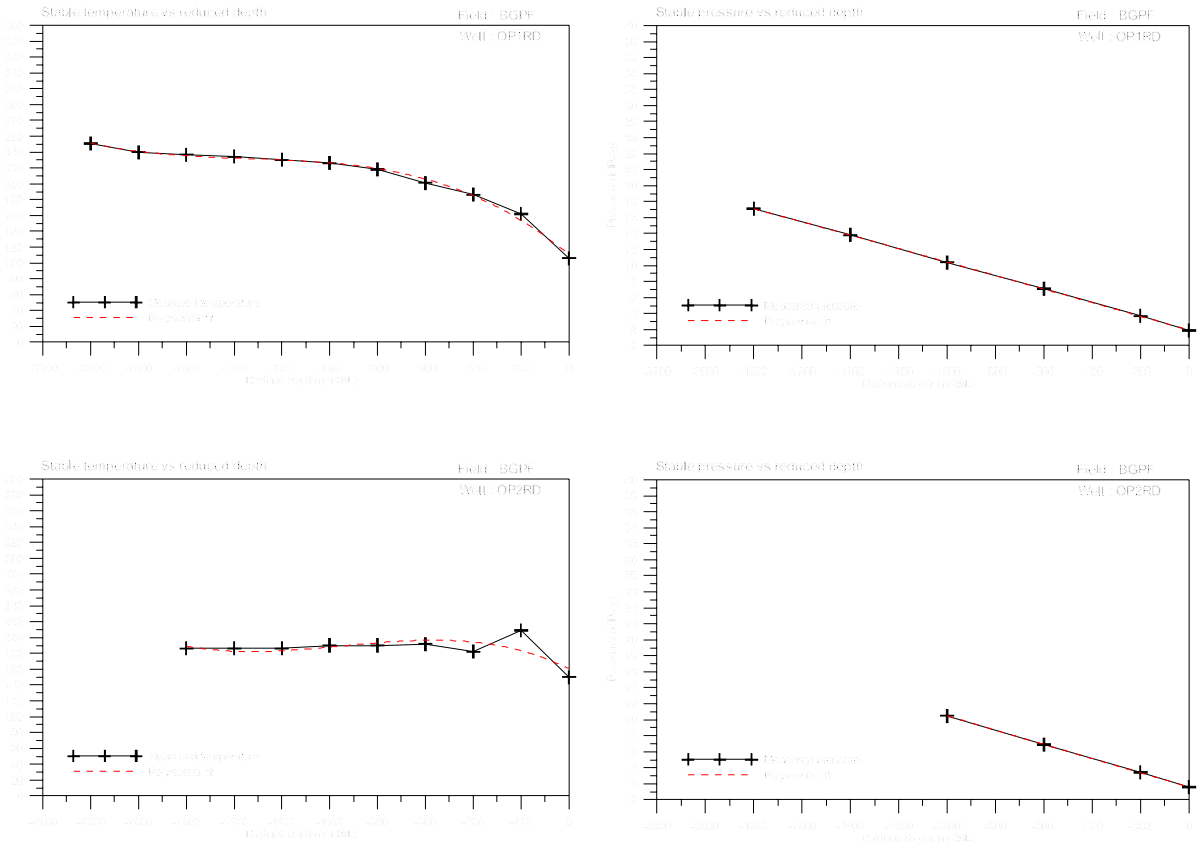

Stable pressure vs reduced depth
Field: BGPF
Well: OP3D


Stable temperature vs reduced depth
Field: BGPF
Well: OP3D


Stable pressure vs reduced depth
Field: BGPF
Well: OP3D


Stable temperature vs reduced depth
Field: BGPF
Well: OP5DA


Stable pressure vs reduced depth
Field: BGPF
Well: OP5DA



**APPENDIX V: UNIX shell script for bore output trend plotting program TSERIES**

```
#!/bin/ksh
#
#    T  S E R I E S
#    Year created September 2003
#    Jaime and Hilmar
#
#    Script using S Q L + and G N U P L O T (c)
#    to create time series plots of bore output trends
#    using data from Oracle database
#
#    Flags are 1) Name or ID of well;
#                2) Parameter to be plotted (WHP, MF or H vs. time);
#                3) Starting date; and
#                4) Ending date.
#
if [[ $# -lt 8 || $1 = "-h" ]]
then
echo "
      Display wellhead pressure, mass flow, enthalpy of a well over a time period
      Usage:  tseries  -w wellname  -p param  -s start(DD-MM-YYYY)  -e end(DD-MM-YYYY)
              for example: tseries -w PAL3D -p whp -s 01-01-1990 -e 12-31-1990
```

```
                PARAMETER -p:
                whp  :    Wellhead pressure
                mf   :    Massflow
                h    :    Enthalpy

OPTIONS
       -w   : wellname
       -s   : starting date
       -e   : ending date
" 1>&2
  exit 1
fi

# Options
set -- $(getopt "w:p:s:e:" "$@")
for arg in $@
do
case $arg in
 -w) wellname=$2 shift 2;;
 -p) parameter=$2  shift 2;;
 -s) startdate=$2     shift 2;;
 -e) enddate=$2     shift 2;;
 --) shift;       break;;
esac
done

echo Parameters $wellname $parameter $startdate $enddate

# Submenu: wellhead pressure
if test $parameter = "whp"
then
echo "
set echo off
set feedback off
set flush off
set term off
set verify off
set newpage 0
set lines 80
set pages 0

spool bore_output
select a.date_start, a.whp
from bom a, well b
where a.well = b.well
and b.wellname = '$wellname'
and a.date_start >= '$startdate'
and a.date_start <= '$enddate'
order by a.date_start;
spool off
quit" > run.sql
```

```
# Run the SQL+ commands - and data goes to file bore_output.lst
sqlplus / @run.sql > /dev/null

# Create GNUPLOT commands file (run.gnu)

echo "set title 'Well $wellname $parameter trend from $startdate to $enddate '
set xlabel 'Date'
set ylabel 'Wellhead pressure (MPag)'
set yrange [0:2]
set xdata time
set timefmt \"%Y-%m-%d\"
set format x \"%m-%d\"
plot \"bore_output.lst\" using 1:2 with linespoints
pause -1" > run.gnu

# Run the gnuplot-file
gnuplot run.gnu
fi

# Submenu: massflow
if test $parameter = "mf"
then
echo "
set echo off
set feedback off
set flush off
set term off
set verify off
set newpage 0
set lines 130
set pages 0

spool  bore_output
select a.date_start, a.massf
from bom a, well b
where a.well = b.well
and b.wellname = '$wellname'
and a.date_start >='$startdate'
and a.date_start<='$enddate'
order by a.date_start;
spool off
quit" > run.sql

# Run the sql-commands - and data goes to file bore_output.lst
sqlplus / @run.sql > /dev/null

# Create GNUPLOT commands file (run.gnu)
echo "set title 'Well $wellname $parameter trend from $startdate to $enddate '
set xlabel 'Date'
set ylabel 'Mass flow (kg/s)'
set yrange [0:100]
set xdata time
set timefmt \"%Y-%m-%d\"
```

```
set format x \"%m-%d\"
plot \"bore_output.lst\" using 1:2 with linespoints
pause -1" > run.gnu

# Run the gnuplot-file
gnuplot run.gnu
fi

# Submenu: enthalpy
if test $parameter = "h"
then
echo "
set echo off
set feedback off
set flush off
set term off
set verify off
set newpage 0
set lines 130
set pages 0

spool bore_output
select a.date_start, a.enthalpy
from bom a, well b
where a.well = b.well
and b.wellname = '$wellname'
and a.date_start >='$startdate'
and a.date_start<='$enddate'
order by a.date_start;
spool off
quit" > run.sql

# Run the sql-commands - and data goes to file bore_output.lst
sqlplus / @run.sql > /dev/null

# Create gnuplot-commands file (run.gnu)
echo "set title 'Well $wellname $parameter trend from $startdate to $enddate '
set xlabel 'Date'
set ylabel 'Enthalpy (kJ/kJ)'
set yrange [0:2000]
set xdata time
set timefmt \"%Y-%m-%d\"
set format x \"%m-%d\"
plot \"bore_output.lst\" using 1:2 with linespoints
pause -1" > run.gnu

# Run the gnuplot-file
gnuplot run.gnu
fi
```

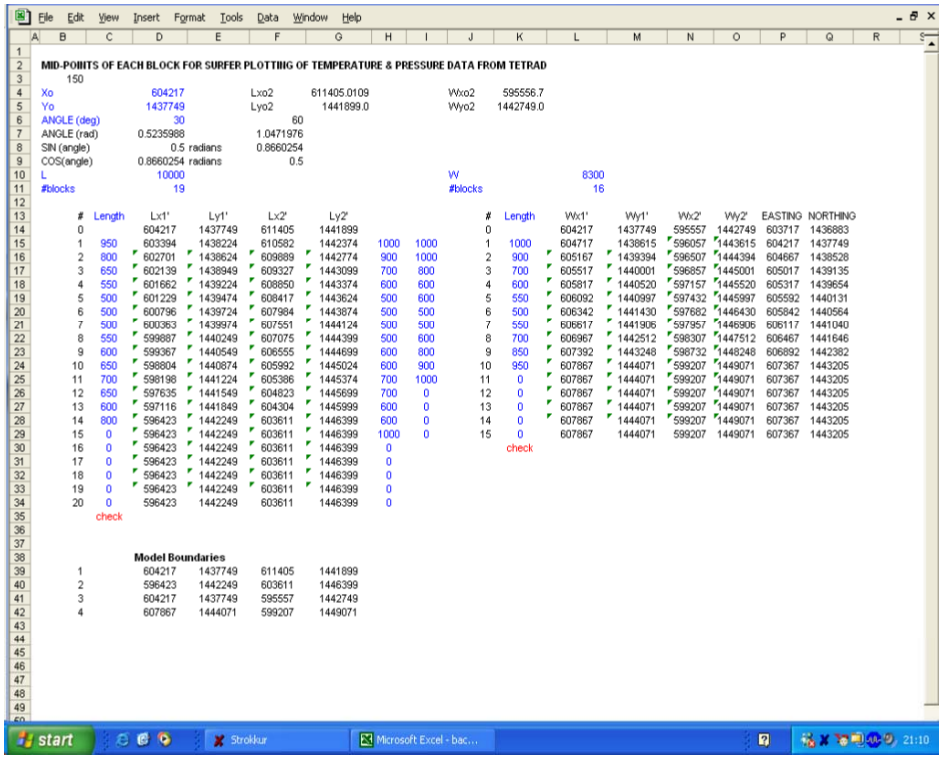
**APPENDIX VI: UNIX shell script for temperature contouring program CONTOUR**

```
#!/bin/ksh
#
#  CONTOUR
#  Year created 2003
#  Hilmar and Jaime
#  Script using S Q L + and G M T ©   to generate input for contouring application
#
echo "Function XYZ to generate input deck for contouring application"

if [[ $# -lt 2 || $1 = "-h" ]]
then
echo "Usage:  contour  Project_ID  Depth (mrsl)
    " 1>&2

  exit 1
fi

# Gets kick off point (kop)
echo "
set echo off
set feedback off
set flush off
set term off
set verify off
set newpage 0
set lines 80
set pages 0
column kop format 999.9

spool kop
select distinct a.well, a.kopmod
from ops\$jaja.trackpoly a, ops\$jaja.wellbyproj b
where a.well = b.well
and b.project = '$1';
spool off

quit" > j1.sql
sqlplus / @j1.sql > /dev/null

# Gets (wellname, easting, northing, vertical depth, temperature) for all vertical wells
echo "
set echo off
set feedback off
set flush off
set term off
set verify off
set newpage 0
set lines 80
set pages 0
column easting format 9999999
column northing format 99999999
column vertical format 9999.9
```

```
column measured format 9999.9
column temperature format 9999
spool xyztemp" > j2.sql

awk 'BEGIN{printf("select a.wellname, a.x easting, a.y northing, a.z - %d vertical, \n",'$2');
printf("( b.stemp_0 + b.stemp_1 * power((%5.1f),1) + b.stemp_2 * power((%5.1f),2) + b.stemp_3 *
power((%5.1f),3) + b.stemp_4 * power((%5.1f),4) ) temperature \n", '$2','$2','$2','$2');
printf("from ops$jaja.well a, ops$jaja.stabletempeqn b\nwhere a.well = b.well\nand a.well  in (\n")}
$2 <= '$2' {well[j] = $1;j++}
END {for (i = 1;i<j-1;i++) {printf("%d,\n",well[i])}
printf("%d)\n;\n",well[j-1])}'  kop.lst >> j2.sql

# Gets (wellname, easting, northing, vertical depth, temperature, measured depth) for deviated wells

awk 'BEGIN {printf("select b.wellname, (a.mevsrd_0 +a.mevsrd_1 * power ((%5.1f), 1) + a.mevsrd_2
* power ((%5.1f), 2) + a.mevsrd_3 * power ((%5.1f), 3) + a.mevsrd_4 * power ((%5.1f), 4))
easting,\n",'$2','$2','$2','$2');
printf("(a.mnvsrd_0 * power ((%5.1f), 0) + a.mnvsrd_1 * power ((%5.1f), 1) + a.mnvsrd_2 * power
((%5.1f), 2) + a.mnvsrd_3 * power ((%5.1f), 3) + a.mnvsrd_4 * power ((%5.1f), 4)) northing,
\n",'$2','$2','$2','$2','$2');
printf("(a.mvdvsrd_0 * power ((%5.1f), 0) +a.mvdvsrd_1 * power ((%5.1f), 1)) vertical, \n", '$2', '$2');
printf("( c.stemp_0 + c.stemp_1 * power((%5.1f),1) + c.stemp_2 * power((%5.1f),2) + c.stemp_3 *
power((%5.1f),3) + c.stemp_4 * power((%5.1f),4) ) temperature, \n", '$2','$2','$2','$2');
printf("(a.mmdvsrd_0 * power ((%5.1f), 0) + a.mmdvsrd_1 * power ((%5.1f), 1) + a.mmdvsrd_2 *
power ((%5.1f), 2) ) measured \n",'$2','$2','$2');
printf("from ops$jaja.trackpoly a, ops$jaja.WELL b, ops$jaja.stabletempeqn c\nwhere a.well =
c.well\nand a.well = b.well\nand c.well in (\n")}
$2 > '$2' {well[j] = $1;j++}
END {for (i=1; i< j-1;i++) {printf("%d,\n",well[i])}
printf("%d)\n;\n",well[j-1])}' kop.lst >> j2.sql
echo "spool off

quit" >> j2.sql
sqlplus / @j2.sql > /dev/null

# Deletes data rows with negative temperature and vertical depth
# Prints (easting, northing, temperature, wellname)
awk '$4 > 0 {print $2,$3, $5, $1}' xyztemp.lst > xyztemp

# GMT part

GMT_HOME=/usr/local/apps/GMT3.4/bin
$GMT_HOME/gmtset LABEL_FONT_SIZE 11 ANOT_FONT_SIZE 10 UNIX_TIME_POS -1/-1.3

# Set limits of x and y for better appearance; instead GMT program minmax gives min/max values:
x1=598100 x2=605000 y1=1440100 y2=1445100
# Sets data file source
datafile=xyztemp
# Sets output file
psout=test.ps

rm .gmtcommands
# command for initiating projection
$GMT_HOME/psbasemap  -JX14/10c -R$x1/$x2/$y1/$y2 -Ba1000f1000:"Eastings
(m)":/a1000f1000:"Northings (m)"::.$head:SWne  -X5 -Y5 -P -K > $psout
```

```
# color ramp for the temperature, always the same
# Tin   R     G       B    Tfin    R     G       B
#
echo "-40       40       0       150     110     40      0       150
110     0       10      200     160     0       10      200
160     0       40      224     180     0       40      224
180     13      129     248     190     13      129     248
190     50      190     255     200     50      190     255
200     97      225     240     210     97      225     240
210     124     235     200     220     124     235     200
220     172     245     168     230     172     245     168
230     223     245     141     240     223     245     141
240     247     215     104     250     247     215     104
250     255     160     69      260     255     160     69
260     238     80      78      270     238     80      78
270     255     124     124     280     255     124     124
280     245     179     174     290     245     179     174
290     255     215     215     300     255     215     215
300     255     225     225     310     255     225     225
310     255     235     235     320     255     235     235
320     255     245     245     330     255     245     245
330     255     255     255     360     255     255     255" > color.palette

awk '{print $1,$2,$3}' $datafile | $GMT_HOME/blockmean -I0.5/0.1 -R | surface -Gtmp.grd -I100 -R
-T0.3

$GMT_HOME/grdimage tmp.grd -JX -R -Ccolor.palette -O -K -Y >> $psout
$GMT_HOME/grdcontour tmp.grd -JX -R -W4 -C10 -Wc2 -A20 -Wa4 -O -K >> $psout

# Plot locations with symbols and text
awk '{print $1,$2}' $datafile | $GMT_HOME/psxy  -JX -R -Ss0.2 -O -K >> $psout
# Add well names to plot
awk '{print $1,$2,8,0,1,7,$4}' $datafile | $GMT_HOME/pstext -JX -R -N -O -K >> $psout
# adding parameters to plot
awk '{print $1,$2,8,0,2,7,$3}' $datafile | $GMT_HOME/pstext -JX -R -N -O >> $psout
# display surface
gv $psout &
# clean up
 rm -r kop.lst j1.sql j2.sql xyztemp xyztemp.lst tmp.grd color.palette
# Comments: In the PATH variable this directory -> /usr/local/apps/gmt/bin
# was used to run the GMT-commands which gave us version 3.0 I think
# By declaring GMT_HOME=/usr/local/apps/GMT3.4/bin we are using the version 3.4.3
# This makes a big difference and now we can use the on-line help on the web
```

## APPENDIX VII: The Excel macro spreadsheet used to create the grid files



**MID-POINTS OF EACH BLOCK FOR SURFER PLOTTING OF TEMPERATURE & PRESSURE DATA FROM TETRAD**

150

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Xo | 604217 | Lxo2 | 611405.0109 | Wxo2 | 595556.7 | | |
| Yo | 1437749 | Lyo2 | 1441899.0 | Wyo2 | 1442749.0 | | |
| ANGLE (deg) | 30 | | 60 | | | | |
| ANGLE (rad) | 0.5235988 | | 1.0471976 | | | | |
| SIN (angle) | 0.5 | radians | 0.8660254 | | | | |
| COS(angle) | 0.8660254 | radians | 0.5 | | | | |
| L | 10000 | | | W | 8300 | | |
| #blocks | 19 | | | #blocks | 16 | | |

| # | Length | Lx1' | Ly1' | Lx2' | Ly2' | | |
|---|---|---|---|---|---|---|---|
| 0 | | 604217 | 1437749 | 611405 | 1441899 | | |
| 1 | 950 | 603394 | 1438224 | 610582 | 1442374 | 1000 | 1000 |
| 2 | 800 | 602701 | 1438624 | 609889 | 1442774 | 900 | 1000 |
| 3 | 650 | 602139 | 1438949 | 609327 | 1443099 | 700 | 800 |
| 4 | 550 | 601662 | 1439224 | 608850 | 1443374 | 600 | 800 |
| 5 | 500 | 601229 | 1439474 | 608417 | 1443624 | 500 | 600 |
| 6 | 500 | 600796 | 1439724 | 607984 | 1443874 | 500 | 500 |
| 7 | 500 | 600363 | 1439974 | 607551 | 1444124 | 500 | 500 |
| 8 | 550 | 599887 | 1440249 | 607075 | 1444399 | 500 | 600 |
| 9 | 600 | 599367 | 1440549 | 606555 | 1444699 | 600 | 800 |
| 10 | 650 | 598804 | 1440874 | 605992 | 1445024 | 600 | 900 |
| 11 | 700 | 598198 | 1441224 | 605386 | 1445374 | 700 | 1000 |
| 12 | 650 | 597635 | 1441549 | 604823 | 1445699 | 700 | 0 |
| 13 | 600 | 597116 | 1441849 | 604304 | 1445999 | 600 | 0 |
| 14 | 800 | 596423 | 1442249 | 603611 | 1446399 | 600 | 0 |
| 15 | 0 | 596423 | 1442249 | 603611 | 1446399 | 1000 | 0 |
| 16 | 0 | 596423 | 1442249 | 603611 | 1446399 | 0 | |
| 17 | 0 | 596423 | 1442249 | 603611 | 1446399 | 0 | |
| 18 | 0 | 596423 | 1442249 | 603611 | 1446399 | 0 | |
| 19 | 0 | 596423 | 1442249 | 603611 | 1446399 | 0 | |
| 20 | 0 | 596423 | 1442249 | 603611 | 1446399 | 0 | |

check

| # | Length | Wx1' | Wy1' | Wx2' | Wy2' | EASTING | NORTHING |
|---|---|---|---|---|---|---|---|
| 0 | | 604217 | 1437749 | 595557 | 1442749 | 603717 | 1436883 |
| 1 | 1000 | 604717 | 1438615 | 596057 | 1443615 | 604217 | 1437749 |
| 2 | 900 | 605167 | 1439394 | 596507 | 1444394 | 604667 | 1438528 |
| 3 | 700 | 605517 | 1440001 | 596857 | 1445001 | 605017 | 1439135 |
| 4 | 600 | 605817 | 1440520 | 597157 | 1445520 | 605317 | 1439654 |
| 5 | 550 | 606092 | 1440997 | 597432 | 1445997 | 605592 | 1440131 |
| 6 | 500 | 606342 | 1441430 | 597682 | 1446430 | 605842 | 1440564 |
| 7 | 550 | 606617 | 1441906 | 597957 | 1446906 | 606117 | 1441040 |
| 8 | 700 | 606967 | 1442512 | 598307 | 1447512 | 606467 | 1441646 |
| 9 | 850 | 607392 | 1443248 | 598732 | 1448248 | 606892 | 1442382 |
| 10 | 950 | 607867 | 1444071 | 599207 | 1449071 | 607367 | 1443205 |
| 11 | 0 | 607867 | 1444071 | 599207 | 1449071 | 607367 | 1443205 |
| 12 | 0 | 607867 | 1444071 | 599207 | 1449071 | 607367 | 1443205 |
| 13 | 0 | 607867 | 1444071 | 599207 | 1449071 | 607367 | 1443205 |
| 14 | 0 | 607867 | 1444071 | 599207 | 1449071 | 607367 | 1443205 |
| 15 | 0 | 607867 | 1444071 | 599207 | 1449071 | 607367 | 1443205 |

check

**Model Boundaries**

| | | | | |
|---|---|---|---|---|
| 1 | 604217 | 1437749 | 611405 | 1441899 |
| 2 | 596423 | 1442249 | 603611 | 1446399 |
| 3 | 604217 | 1437749 | 595557 | 1442749 |
| 4 | 607867 | 1444071 | 599207 | 1449071 |