554054

Analysis Report AP-111 Revision 1

Culebra Water Level Monitoring Network Design

(AP-111: Analysis Plan for Optimization and Minimization of the Culebra Monitoring Network for the WIPP)

Task Number 1.4.2.3
Report Date: August 4, 2010

Author: [illegible] Date: 8/4/10
Kristopher L. Kuhlman
Repository Performance Department (6712)

Technical Review: Mario Chavez for Date: 8/6/10
Sean A. McKenna
National Security Applications Department (6311)

QA Review: Mario Chavez Date: 8/6/10
Mario J. Chavez
Carlsbad Programs Group (6710)

Management Review [illegible] Date: 8/4/2010
Christi D. Leigh
Manager, Repository Performance Department (6712)

WIPP:1.4.2.3:TD:QA-L:RECERT:540476

Acknowledgements

Thanks to David Hart and Sean McKenna for their previous work on the calibration of the transmissivity fields, utilized as inputs in Section 4 of this analysis. Thanks to Rick Beauheim for his comments and guidance on early drafts of this report.

Table of Contents

List of Figures

List of Tables

Executive Summary

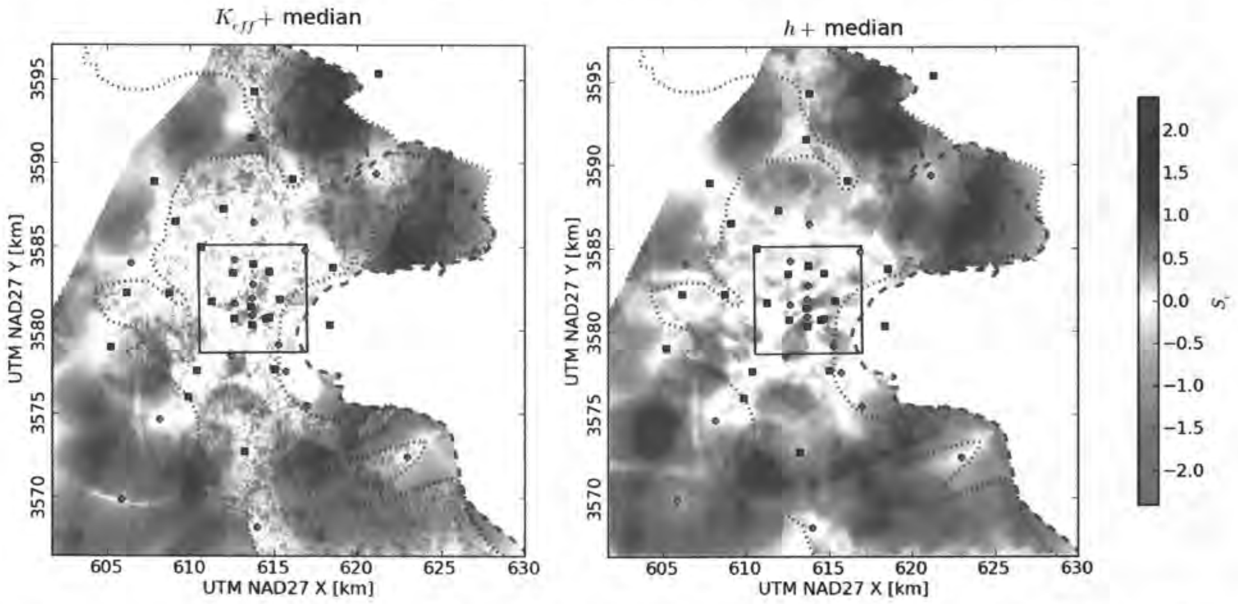
This analysis report presents the methods, data, and results of calculations done in support of Culebra head and hydraulic gradient monitoring network design and optimization. The three metrics used include:

1. freshwater head kriging variance reduction,
2. triangle geometry shape quality maximization, and
3. identification of areas where there is a high statistical correlation between model-predicted travel times and either
    a. model input effective hydraulic conductivity (K_{eff}) or
    b. heads (h)

These three different and largely independent approaches to monitoring network design are discussed individually in detail (Sections 2, 3 and 4) and are combined (Section 5) for two different types of results,

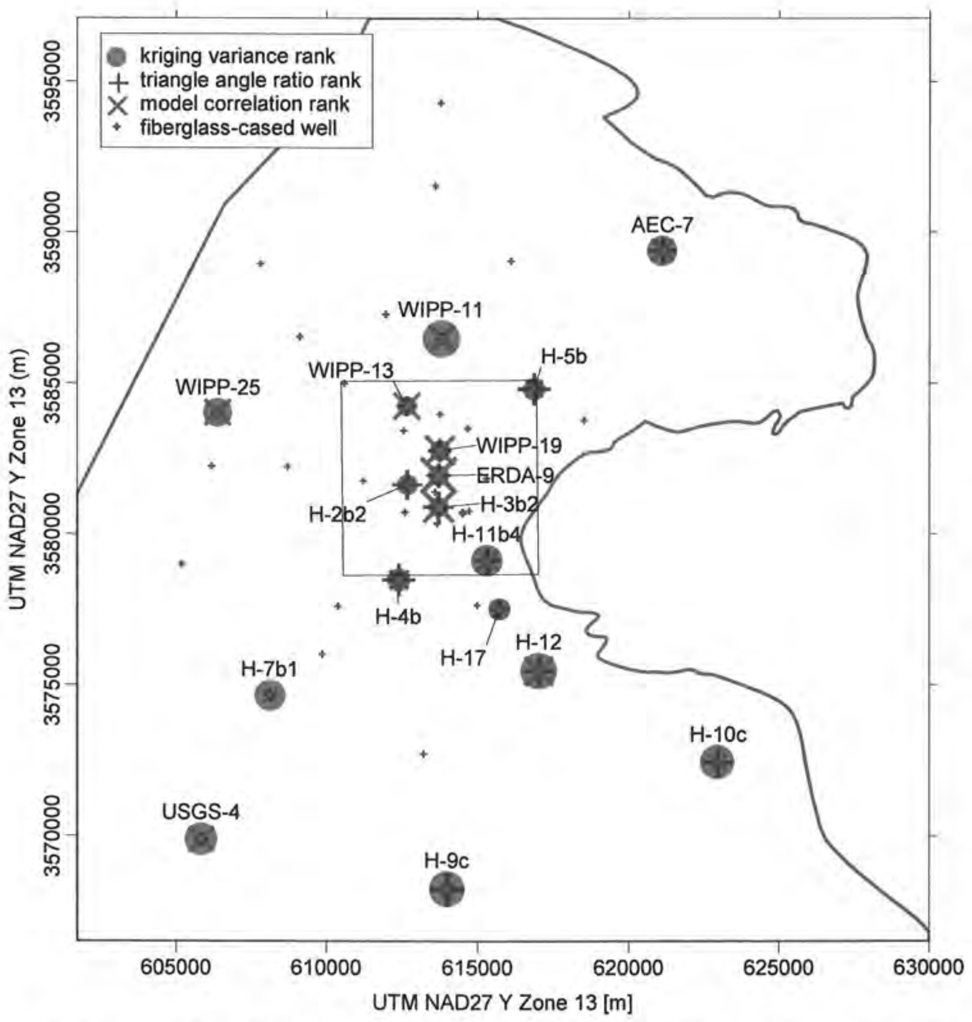
1. ranking of possible locations for new wells, and
2. ranking the importance of maintaining existing locations.

The combinations of the three metrics for the suitability of a location for a new monitoring location are shown in the following two figures (discussed more fully in the Section 5).



In these two figures, red and orange areas are poor locations for a new well, while dark blue and purple areas are good locations for a new well. The left figure includes metrics 1, 2, and 3a (from the top bulleted list), while the right fiure includes metrics 1, 2, and 3b. While the two figures are different in some details, they both show that the areas between monitoring locations that are distant from the WIPP LWB (interior black square) rank highly overall (dark blue). Areas roughly consisting of 'spokes" radiating away from the WIPP LWB – between closely spaced monitoring wells – rank poorly overall (yellow and orange).

Using the same three metrics for ranking the existing steel-cased wells (assuming fiberglass-cased wells will have a long life), the results are combined in the following figure (discussed more fully in Section 5).



In this figure, the size of the different symbols is related to the relative importance of each of the steel-cased wells, ranked via the three metrics. Many wells are important to one or two metrics and unimportant to another (e.g., closely-spaced wells inside the WIPP LWB perform poorly in the kriging variance reduction, but might be in important areas for the model input/output correlation). Overall, wells H-12, H-10c, and AEC-7 have relatively high ranks in all three metrics. These wells are somewhat isolated and therefore are individually important in their contributions to the success of the overall monitoring network.

1.0 Introduction

This analysis report presents the methods, data, and results of calculations done in support of Culebra head and hydraulic gradient monitoring network design and optimization. Three different and largely independent approaches to monitoring network design are examined. These approaches include optimal locations for additional monitoring wells and identification of wells in the current monitoring network that could be removed with minimal effect on meeting the monitoring objectives. The three different sets of results are then combined into a final set of maps indicating potential areas for the installation of new monitoring wells. Additionally, several wells in the existing network could be removed with minimal effect on the ability of the monitoring network to predict heads at unmonitored locations and to detect changes in the hydraulic gradient. The three approaches used here are similar to approaches used in the 2004 ground water monitoring network design calculations, and this allows for direct comparison of some results with those obtained five years ago.

1.1. Background

The Waste Isolation Pilot Plant (WIPP) is located in southeastern New Mexico and has been developed by the U.S. Department of Energy (DOE) for the geologic (deep underground) disposal of transuranic (TRU) waste. Containment of TRU waste at the WIPP is regulated by the U.S. Environmental Protection Agency (EPA) according to the regulations set forth at Title 40 of the Code of Federal Regulations, Parts 191 and 194. The DOE demonstrates compliance with the containment requirements in the regulations by means of a performance assessment (PA), which estimates releases from the repository for the regulatory period of 10,000 years after closure.

Groundwater monitoring and modeling activities at the WIPP are an integral part of the DOE's broader requirements to demonstrate that WIPP operations are performed in a manner that ensures protection of the environment, the health and safety of workers and the public, proper characterization of the disposal system, and compliance of the WIPP with applicable regulations. Continued compliance with regulations must be demonstrated every five years during the operational phase of the WIPP. The monitoring requirements apply not only for the current operational phase (~35 years), but extend through the post-closure phase of the facility to meet applicable regulations. Because of these long-term requirements, DOE's Carlsbad Field Office (CBFO) has developed the WIPP Groundwater Protection Program Plan (DOE, 2009) that describes: relevant regulatory (EPA and New Mexico Environment Department) drivers; the current groundwater-monitoring network and how it has evolved over time; current groundwater program elements; strategies for maintaining compliance; methods for implementing the strategies; and roles and responsibilities of monitoring program participants.

This analysis report is a revision of McKenna (2004), which identified wells that could be removed from the existing network as well as looked at potential locations to expand the monitoring network. Since 2004, the number of monitoring wells available for analysis have increased by 40%, from 30 to 42. Now after the SNL-series fiberglass-cased wells have been constructed, this report is re-evaluating the well network based on the new information obtained from these new wells and the updated Culebra PA flow model, completed for the compliance recertification application (CRA) 2009 performance assessment baseline calculation (PABC).

1.2. Purpose

The purpose of these calculations is primarily to determine which of the remaining steel-cased wells can be plugged and abandoned (P&Aed) without degrading the monitoring network. A secondary goal is to identify optimal locations for any new Culebra monitoring wells. The calculations herein will be focused on meeting the goals of:

1. The monitoring network must allow the determination of the direction and rate of groundwater flow across the WIPP site. This is both an NMED and an EPA requirement (NMAC, 2000 incorporating 40 CFR Part 194 §264.98(e) (U.S. EPA, 1996));
2. The monitoring network must provide data needed to infer causes of changes in water levels that might be observed. This is an EPA requirement, 40 CFR Part 194, Subpart C §194.42 (U.S. EPA, 1996); and
3. The monitoring network must provide spatially distributed head data adequate to allow both defensible boundary conditions to be inferred for Culebra flow models and defensible calibration of those models (PA requirements).

The degree to which these objectives can be reduced to quantitative measures is evaluated as part of the work reported in this analysis report.

The minimized and optimized monitoring network will be created using available information including existing wells and up to date understanding of the hydrology of the Culebra. The optimization and minimization process takes the following factors into consideration:

1. Existing locations of fiberglass-cased wells
2. Existing well locations that are not needed
3. Culebra hydraulic property variations and geologic boundaries

1.3. Outline

This report documents the data, methods, and summary results of the work completed under Analysis Plan 111 (Kuhlman, 2008). The analysis has four main components, which look at the network optimization from the perspective of:

1. **kriging**: considers the spatial clustering of observation points and the geostatistical structure of the data via the variogram (see Section 2.0);
2. **local gradient estimators**: Delaunay triangles that consider the geometric quality of the well network and the observed gradient across the well network (see Section 3.0);
3. **flow model correlation**: uses the structure embodied in the calibrated flow model regarding formation heterogeneity and geologic processes (see Section 4.0);
4. combining the results of the three above methods into one result (see Section 5.0)

1.4. Calculation domain

The spatial domain used for the different calculations in support of monitoring network design is the same as the model domain used in the two-dimensional (2D) Culebra groundwater flow model (Hart et al., 2008; 2009). This model domain is aligned with the Universal Transverse Mercator (UTM) coordinate system and is 30.7 km long by 28.4 km wide (872 km^2 total, 587 km^2 active). The corners of the Culebra numerical groundwater model domain are listed in Table 1-1. Relative to the CRA 2004 calculations, the eastern extent of the model domain has moved from 624000 m to 630000 m UTM 1927 North American datum (NAD27) meters, as explained in (Hart et al., (2008), §2.1). These coordinates define the center of 100 m × 100 m model cells at the four corners of the model domain. All monitoring calculations that produce results on a spatial grid employ the same grid as used for the 2D Culebra flow model (see e.g., Kuhlman, 2010b), unless otherwise noted.

Table 1-1. Culebra flow model domain UTM NAD27 Zone 13 coordinates

Model domain corner	X [m]	Y [m]
Northeast	630000	3597100
Northwest	601700	3597100
Southeast	630000	3566500
Southwest	601700	3566500

The WIPP land-withdrawal boundary (LWB) encloses 16 township and range sections (approximately 41 km^2) near the center of the MODFLOW model domain. The boundary of the WIPP site is defined by the corners of the 16 sections, which have the UTM coordinates given in Table 1-2. For the calculations described in this report, the coordinates given in Table 1-1 and Table 1-2 are used to delineate areas, across which we average different measures of effectiveness for the monitoring network.

Table 1-2. The WIPP LWB UTM NAD27 Zone 13 coordinates

WIPP boundary corner	X [m]	Y [m]
Northeast	616941	3585109
Northwest	610495	3585068
Southeast	617015	3578681
Southwest	610567	3578623

1.5. *Observed Data*

The approaches developed in this report can be applied to any set of nearly-simultaneous undisturbed head measurements (i.e., a "snapshot" in time of the hydraulic head in the Culebra). The wells used here are shown in Figure 1-1 and the data observed at these wells are listed in Table 1-3 (freshwater head data from (Johnson, 2009)). The majority of the calculated freshwater head values correspond to those used in the calibration of the CRA-2009 PABC transmissivity fields (Hart et al., 2009) with four exceptions and one note:

1. A representative 2004 value from AEC-7 was used (this well was left out of the flow model calibration due to known configuration problems in 2007). Freshwater heads at AEC-7 have been very stable historically (1988 through 2004), and are now representative of previous trends after well reconfiguration. Over 15 years (12/1988 through 3/2004) there were 172 head measurements with a standard deviation of only 0.56 m;

2. Freshwater heads from March 2007 were used at the H-19 wellpad, to include the six redundant wells (H-19b{2,3,4,5,6,7}), which are only monitored quarterly. These wells are only included in the variogram modeling, to better constrain head variation at short distance scales (see discussion about optimal well networks for estimating variograms in Warrick & Myers (1984) or Conwell, et al. (1997)). The central H-19b0 well is used as

the sole H-19b well in the rest of the analyses discussed in this report. The coordinates of the H-19b wells reflect their computed UTM *x*, *y* locations at the Culebra (229 m below ground surface (bgs)), accounting for observed deviations from vertical completion (Meigs et al., 2000). H-19b0 freshwater heads are within 2 cm between the March and May 2007 observation times.

3. SNL-6 and SNL-15 have not recovered since being drilled in 2005, and will likely take hundreds of years to recover to "static" conditions. These wells use land-surface elevations in place of water levels in the model calibration (>1000 m above mean sea level (AMSL)); they are not used in situations where a representative head value is needed (e.g., variogram modeling and gradient estimation), but their locations are included otherwise (e.g., kriging and network geometry optimization).
4. WIPP-30 is not included in the network optimization, since this well was plugged and abandoned in May 2007.
5. WIPP-25 is used both here and in the CRA-2009 PABC Culebra flow modeling exercise, although this well was P&Aed in 2009.

In addition to the calculated May 2007 freshwater heads, calibration results from the most recent iteration of the Culebra PA T-fields (Hart et al., 2009) are also used. These results include the simulated head values, calibrated transmissivity and anisotropy values, and particle travel times from C-2737 to the WIPP LWB for each of the 100 model realizations. These results are used in the third sensitivity-based approach.

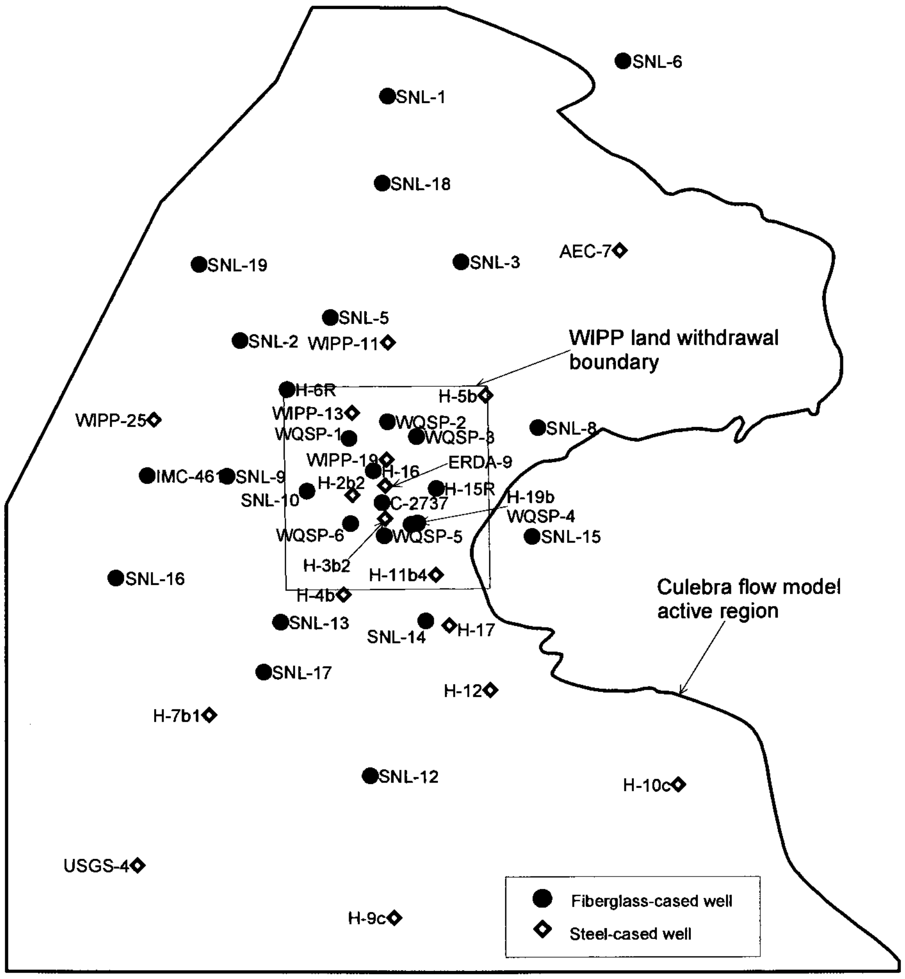


Figure 1-1. Locations of monitoring wells used in this study

Table 1-3. Freshwater Heads from May 2007 used in analysis (Johnson, 2009)

	Well	UTM NAD27 *x* Zone 13[m]	UTM NAD27 *y* Zone 13 [m]	Freshwater Head [m AMSL]
1	AEC-7^(1)	621126	3589381	933.03
2	C-2737	613598.0	3581400.9	921.23
3	ERDA-9	613696.1	3581944.3	924.88
4	H-2b2	612662.5	3581639.7	929.62
5	H-3b2	613693.6	3580899.6	918.68
6	H-4b	612376.0	3578478.5	916.34
7	H-5b	616866.0	3584807.0	939.12
8	H-6b	610598.6	3584986.9	936.44
9	H-7b1	608122.8	3574646.4	914.58
10	H-9c	613971.1	3568237.2	912.80
11	H-10c	622976.3	3572444.3	922.02
12	H-11b4	615297.3	3579123.5	917.09
13	H-12	617022.0	3575460.5	916.53
14	H-15	615310.0	3581855.2	920.32
15	H-17	615717.0	3577507.8	916.24
16	H-19b0^(2)	614515.2	3580718.9	918.82
17	H-19b2^(2)	614516.2	3580693.8	918.64
18	H-19b3^(2)	614526.1	3580719.6	918.57
19	H-19b4^(2)	614494.6	3580727.6	918.77
20	H-19b5^(2)	614502.3	3580713.6	918.60
21	H-19b6^(2)	614518.0	3580738.5	918.58
22	H-19b7^(2)	614516.0	3580706.7	918.54
23	IMC-461	606182.6	3582246.4	928.95
24	SNL-1	613781.4	3594299.0	941.86
25	SNL-2	609113.1	3586529.1	937.65
26	SNL-3	616103.0	3589046.9	939.81
27	SNL-5	611970.2	3587284.7	938.59
28	SNL-6^(3)	621244.6	3595390.0	856.00
29	SNL-8	618522.8	3583783.3	929.94
30	SNL-9	608704.8	3582237.7	932.05
31	SNL-10	611229.3	3581764.8	931.54
32	SNL-12	613223.4	3572727.4	915.24
33	SNL-13	610394.3	3577599.8	918.19
34	SNL-14	614989.7	3577652.0	916.33
35	SNL-15^(3)	618353.2	3580336.4	865.65
36	SNL-16	605191.8	3578999.7	918.68
37	SNL-17	609863.2	3576016.1	916.78
38	SNL-18	613605.8	3591528.6	939.87
39	SNL-19	607813.5	3588947.4	937.58
49	USGS-4	605841.0	3569887.0	911.11
41	WIPP-11	613788.2	3586474.0	940.65
42	WIPP-13	612645.0	3584241.7	939.78
43	WIPP-19	613738.8	3582773.5	933.66
44	WIPP-25	606385.7	3584022.8	937.57
45	WQSP-1	612559.4	3583430.3	938.28
46	WQSP-2	613770.4	3583972.2	939.87
47	WQSP-3	614685.5	3583506.8	936.43
48	WQSP-4	614724.5	3580762.8	919.50
49	WQSP-5	613666.5	3580353.6	918.18
50	WQSP-6	612602.3	3580737.9	921.96

1. representative water level from 2004
2. H-19 wells from March 2007
3. not used in variogram estimation

1.6. Run Control

Nearly all the calculations done for this analysis report were completed on a Dell Precision 690 workstation, equipped with two quad-core 2.66-GHz Intel Xeon chips (X5355). The work was done on this system running the Microsoft Windows XP (service pack 2) operating system. Two of the scripts were run on the PA Pentium 4 Linux cluster (`alice.sandia.gov`); these scripts checked the Culebra MODFLOW model results out of CVS (only accessible from Linux) and performed the binary-to-ASCII conversion on the model-produced heads before creating a zip archive of the files for transfer to Windows. The input files, scripts, and outputs are contained within the `analysis` directory on the CD-ROM associated with this analysis report; the contents of the CD are listed in Section 8.1.

Each section has a run control subsection describing the software and scripts that were used to perform the analysis in that section. All scripts created for this analysis report are listed in Section 8.0 with syntax highlighting and line numbers. Table 1-4 lists the software used throughout this report, all software is either commercial off the shelf (COTS), or it is qualified for use with WIPP PA.

Table 1-4. Summary of software used

Software	Version	Type	Use
Golden Software Surfer	9.9	COTS	Map plotting / Variograms
Microsoft Office Excel	2007 (SP1)	COTS	Plotting / Regression
R	2.10	COTS	Statistical Script Interpreter
Enthought Python (EPD)	6.1	COTS	Script Interpreter / Plotting
GSLIB program KT3D	2.0 (1996)	Qualified	Kriging
The Mathworks MATLAB	R2009b	COTS	Script Interpreter / Plotting
Gnu Bash	3.00.15	COTS	Script Interpreter (Linux)
Windows XP `cmd.exe`	5.1.2600	COTS	Script Interpreter (Windows)

Scripts for Python, R, Bash, and MATLAB are ASCII and are listed in Section 8.0, while Surfer and Excel input files are binary and therefore are included on the CD (see listing of contents of CD in Section 8.1).

1.7. Notation

Throughout this analysis report the following conventions are used:

1. file names and directory paths are listed in the `Courier New` monospaced font;
2. source code excerpts are listed in the `Lucida Console` monospaced font;
3. program functions and classes are listed as code excerpts with trailing parentheses for clarity;
4. units are given in metric, specified in square brackets (unless used as an adjective);
5. scalar variables are in italic font; and
6. vector variables are in bold font.

2.0 Geostatistical Variance Reduction

Geostatistics is the modeling and prediction of spatially-correlated information and it has been used extensively over the past 30 years in areas including ore reserve estimation, contaminant mapping in soils and groundwater, and modeling spatial variability of physical properties of aquifers and petroleum reservoirs. Kriging is the geostatistical algorithm used for spatial estimation; compared to other spatial interpolation algorithms (e.g., inverse distance or linear interpolation), kriging uniquely estimates both a value and its variance at unsampled locations.

Previous studies (Rouhani, 1985) have used kriging variance as a measure of the ability of a groundwater monitoring network to predict hydraulic heads at locations where no wells exist. Groundwater monitoring network design can be optimized to either minimize average kriging variance across the domain or to minimize the maximum predicted kriging variance. The estimation variance can also be used as a metric to justify removing wells from an existing network such that the overall kriging variance has a minimal increase. As an example, (Tuckfield et al., 2001) used the kriging variance of contaminants in a plume to determine the redundancy of groundwater contaminant monitoring wells and targeted those wells with the highest redundancy for removal from the network.

Kriging variance is a direct function of the spatial distribution of observations and the variogram (which is fitted to observed data). Kriging variance is only indirectly a function of the observed values; this is a major advantage of using kriging in monitoring network optimization. Therefore, changes in the kriging variance from the addition or removal of a well can be estimated prior to adding or removing that well with a standard kriging calculation.

The geostatistical analysis presented here utilizes ordinary kriging of the residual freshwater heads, after removing a linear trend. The freshwater heads in the Culebra across the model domain have a clear trend (i.e., the regional north-south gradient, see Figure 2-1). Although it is possible to krige values while simultaneously estimating a trend (i.e., universal kriging), this approach is not used here. Universal kriging does obviate the need to first estimate the linear trend for the kriging, but model-fitting to the observed variogram is made more complex, requiring a non-linear optimization or iterative refinement between variogram fitting and kriging with a trend (Armstrong, 1984; Goovaerts, 1998).

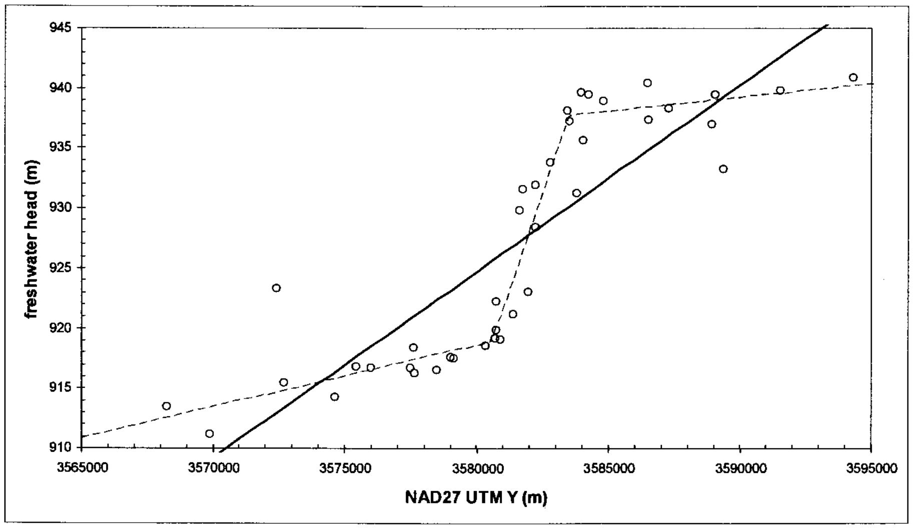


Figure 2-1.Projection of freshwater heads (circles), piecewise linear trend (blue dashed line), and best-fit linear trend (red line) onto *y*-head plane at *x*-midpoint of MODFLOW model domain.

2.1. *Trend Fitting and Residual Calculations*

The residuals associated with the head observations from May 2007 are used for the geostatistical variance reduction analysis. A best-fit linear surface through these heads was calculated using the COTS statistical software R. The equation for the best-fit plane through May 2007 freshwater heads is

$$h(x,y) = Ax + By + C \, . \qquad (1)$$

The results of fitting this equation to the data in Table 1-3 are $A = -9.0\times10^{-5}$, $B = 1.5\times10^{-3}$ and $C = -4.6\times10^{3}$ m (see Table 2-1 for more significant digits and fit statistics). The *y* component (*B*) of the gradient is approximately an order of magnitude larger than the *x* component (*A*). Both *B* and *C* have *t* statistic values indicating significance ($|t|>2$), but *A* does not (see Table 2-1); the east-west component of the regional gradient cannot be estimated accurately from the given data. This same linear fit resulted in coefficients of $A = 1.98\times10^{-4}$, $B = 1.62\times10^{-3}$ and $C = -5007.74$ in the 2004 version of this analysis. The *y*-component of the gradient (*B*) has not changed much, but the *x*-component (*A*) and the additive constant (*C*) have changed. Overall, the resulting gradient vectors are quite similar (see Figure 2-2), considering the large number of wells that have changed between the two studies.

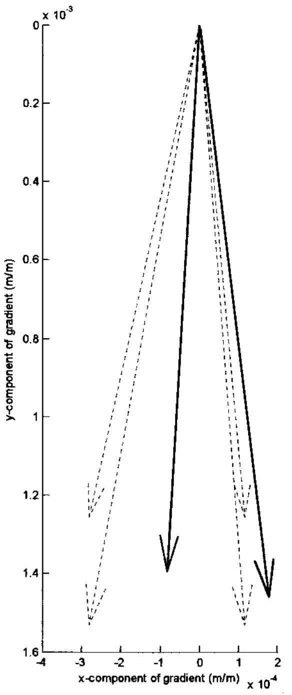


Figure 2-2. Comparison of gradient vectors corresponding to best-fit planes through 2003 (black) and 2007 (red) data. Red dotted lines correspond to estimated gradient ± gradient standard error.

Figure 2-3 illustrates several plots related to the fit of Equation (1) to freshwater heads. The upper-left plot shows the residual (measured – trend) as a function of the trend. The outliers from the moving-average residual trend are H-10c, WQSP-2 and WIPP-13 (see red line and labeled points in the upper left plot in Figure 2-3). The upper-right normal quantile plot (Q-Q) shows that aside from the extreme values, the residuals are ordered approximately normally (plotting quantiles, rather than values makes this plot non-parametric). The lower-left scale-location plot shows magnitude of residuals against the trend value, illustrating that the steep gradient across the WIPP site (920-935 m elevation) is where residuals are largest on average. The lower-right leverage plot shows the relative effects that removing a well has on the predicted surface, plotted against residuals. The wells with the most leverage and the largest residuals are wells at the extremities of the domain, including H-10c, H-9c and AEC-7.

Table 2-1. Fit statistics for linear surface (Equation 1) through freshwater head data (Table 1-3)

```
(Intercept)             x             y
-4.563833e+03 -9.023153e-05  1.548563e-03

Residuals:
   Min     1Q Median     3Q    Max
-7.361 -4.833 -0.459  3.901  9.911

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.564e+03  5.594e+02  -8.158 5.83e-10 ***
x           -9.023e-05  2.231e-04  -0.405    0.688
y            1.549e-03  1.537e-04  10.077 2.06e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.392 on 39 degrees of freedom
Multiple R-squared: 0.7226,     Adjusted R-squared: 0.7083
F-statistic: 50.79 on 2 and 39 DF,  p-value: 1.386e-11
```

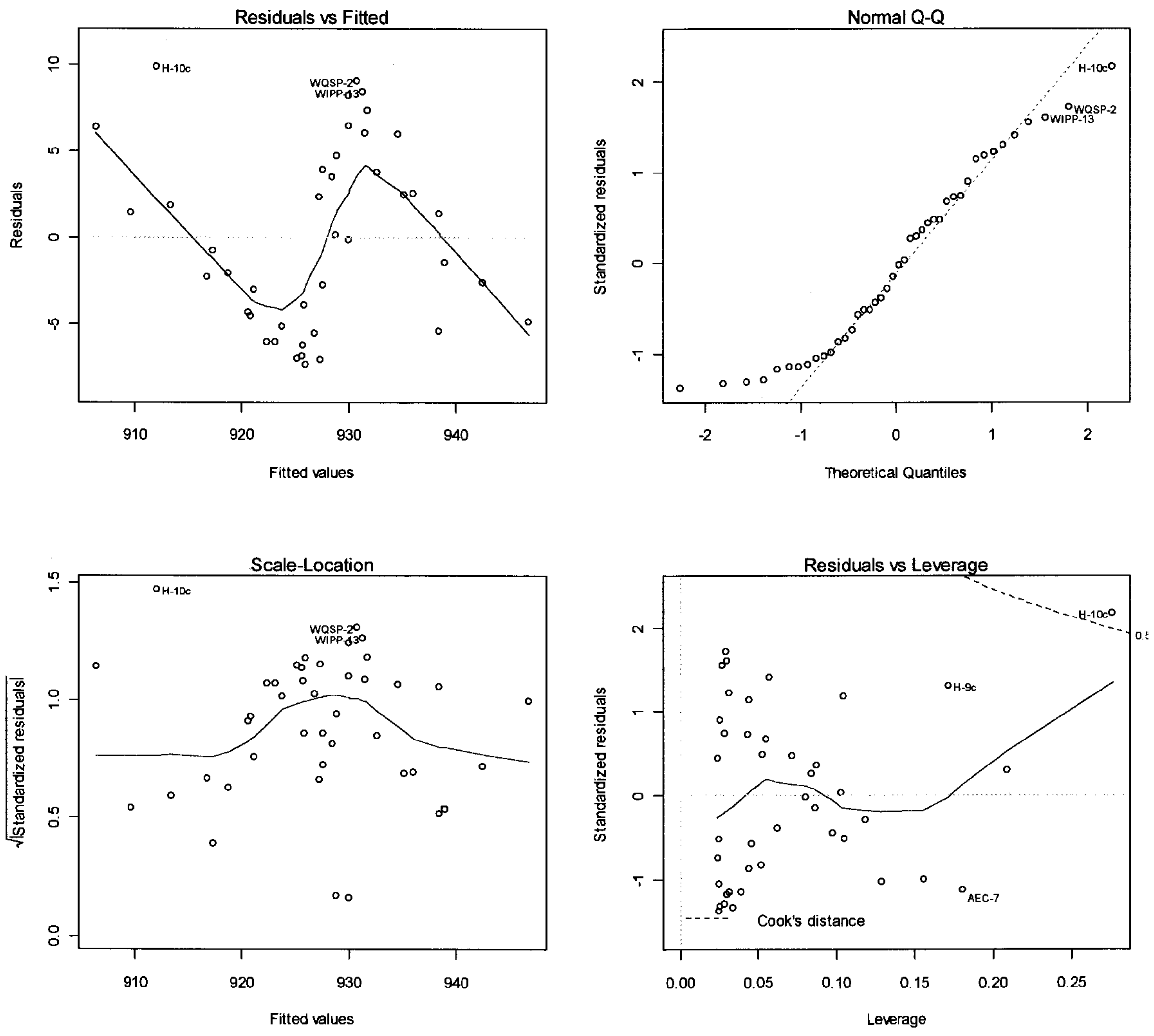


Figure 2-3. Statistics of linear surface (Equation 1) fit to May 2007 freshwater heads

With these parameter values, Equation 1 fits the May 2007 heads with $R^2=0.7083$ (see penultimate row of Table 2-1). This best-fit plane has a hydraulic gradient of 1.55×10^{-3} and a flow direction 3 degrees east of south, but the exact angle is poorly defined. The residuals between the estimated and measured heads are used as the input data for the geostatistical analysis. In the 2004 version of this analysis, R^2=0.6, the gradient magnitude and angle were computed to be 1.64×10^{-3} and 7 degrees east of south (see Figure 2-2 for comparison).

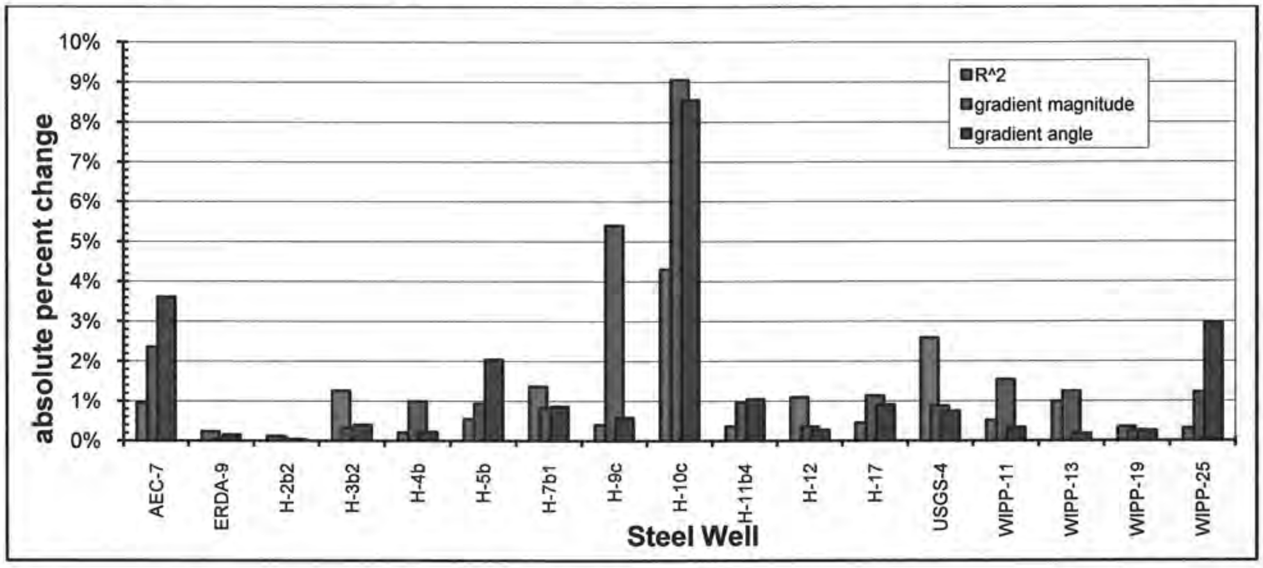


Figure 2-4. Effects of removing a steel-cased well on parameters related to the estimated linear trend.

Adding or removing a single data point from the dataset (Table 2-2) has two potential effects on the results of kriging. First, the best-fit trend surface can change (see "residuals vs. leverage" plot in Figure 2-3 and Figure 2-4), especially if the point being added or removed is at the extremities of the domain (e.g., wells AEC-7, H-9c, and H-10c). Second, the experimental variogram computed from the residuals can change (see next section).

Figure 2-4 shows the results of removing each steel-cased well individually, and fitting Equation (1) using least-squares to the resulting smaller dataset. As was shown in the "residuals vs. leverage" plot in Figure 2-3, H-10c, H-9c, and AEC-7 have a large effect on the R^2 measure of the fit quality. The blue and green bars in Figure 2-4 illustrate the change on the magnitude and angle of the gradient due to removing a single steel-cased well. H-9c has a large effect on the magnitude but not the angle of the gradient; it is located in the south-central portion of the domain. In contrast, WIPP-25 and AEC-7 have larger effects on the angle than the magnitude of the gradient; these wells are on the east and west extremities of the MODFLOW domain, respectively.

Table 2-2. May 2007 freshwater head (FWH) data and residual. The residuals in the right column are calculated as measured – modeled head, sorted by residual magnitude.

	Well	Observed FWH [m]	Residual [m]
1	H-3b2	918.68	-7.36
2	H-19b6	918.58	-7.14
3	H-19b7	918.54	-7.13
4	H-19b3	918.57	-7.12
5	H-19b5	918.6	-7.08
6	H-15	920.32	-7.05
7	WQSP-5	918.18	-7.02
8	H-19b2	918.64	-7.01
9	H-19b4	918.77	-6.93
10	H-19b0	918.82	-6.87
11	WQSP-4	919.5	-6.24
12	H-4b	916.34	-6.07
13	H-11b4	917.09	-6.06
14	C-2737	921.23	-5.60
15	AEC-7	933.03	-5.47
16	SNL-16	918.68	-5.19
17	SNL-1	941.86	-4.92
18	SNL-14	916.33	-4.56
19	H-17	916.24	-4.37
20	WQSP-6	921.96	-3.93
21	SNL-13	918.19	-3.04
22	ERDA-9	924.88	-2.78
23	SNL-18	939.87	-2.64
24	H-7b1	914.58	-2.28
25	SNL-17	916.78	-2.04
26	SNL-19	937.58	-1.45
27	H-12	916.53	-0.79
28	SNL-8	929.94	-0.13
29	IMC-461	928.95	0.15
30	SNL-3	939.81	1.37
31	USGS-4	911.11	1.41
32	SNL-12	915.24	1.81
33	H-2b2	929.62	2.34
34	SNL-2	937.65	2.48
35	SNL-5	938.59	2.51
36	SNL-9	932.05	3.49
37	H-6b	936.44	3.79
38	SNL-10	931.54	3.94
39	WIPP-19	933.66	4.72
40	WIPP-11	940.65	5.99
41	WIPP-25	937.57	6.03
42	H-9c	912.8	6.39
43	WQSP-3	936.43	6.44
44	H-5b	939.12	7.31
45	WQSP-1	938.28	8.22
46	WIPP-13	939.78	8.47
47	WQSP-2	939.87	9.08
48	H-10c	922.02	9.91

2.2. *Variogram Estimation and Modeling*

The experimental variogram is calculated and modeled using Surfer. Introductions to variogram modeling and geostatistics in general are found in many places in the geostatistics literature; e.g., (Isaaks and Srivastava, 1989; American Society of Civil Engineers, 1990; Kitanidis, 1997). The experimental variogram is calculated in Surfer as

$$\hat{\gamma}(\boldsymbol{h}) = \frac{1}{2N(\boldsymbol{h})} \sum_{i=1}^{N(\boldsymbol{h})} [z(\mathbf{u}_i) - z(\mathbf{u}_i + \boldsymbol{h})]^2 , \tag{2}$$

where ***h*** is the lag spacing vector [m] (most generally ***h*** is a vector, but later it will assumed to be a scalar distance), $z(\mathbf{u}_i)$ are the residual freshwater head values at $\mathbf{u}_i$, $\mathbf{u}_i$ is a vector of spatial coordinates (x,y) for the sample locations of each residual value, and $N(\boldsymbol{h})$ is the number of pairs of data points separated by ***h*** (within a given tolerance of ***h***). The values of the experimental variogram $\hat{\gamma}$, are plotted as a function of $|\boldsymbol{h}|$ and a variogram model (a mathematical function) is fit to these data. Valid variogram models ensure a positive-definite covariance matrix in the kriging equations.

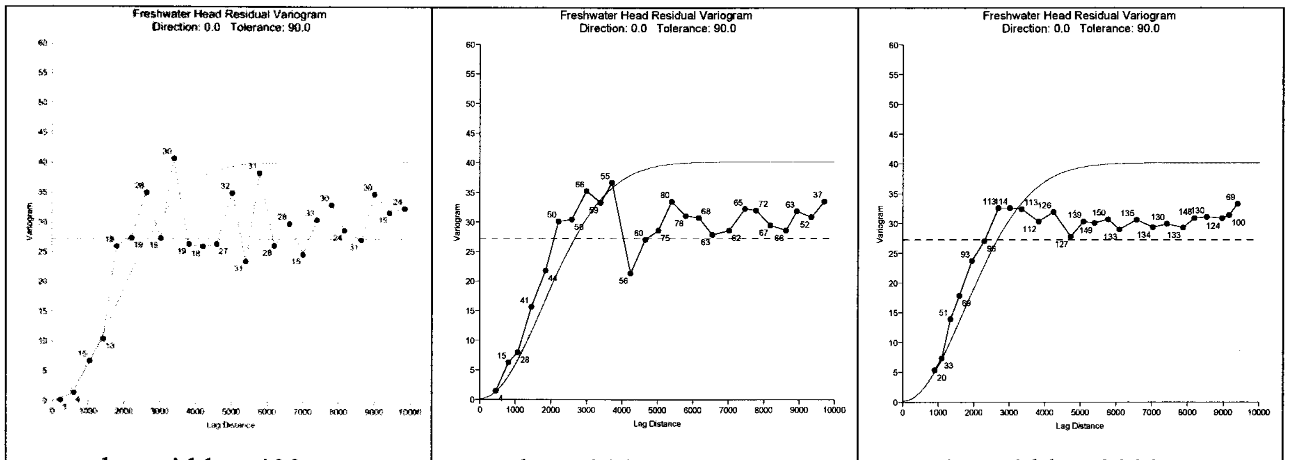
In the current analysis, the infinitely differentiable Gaussian variogram model is chosen to fit the experimental variogram. Since freshwater hydraulic head and residuals computed from it are assumed to be smoothly varying properties (with well-defined first and second spatial derivatives), a variogram that is at least second-order smooth is appropriate. The Gaussian variogram model, as implemented in the kriging program KT3D in GSLIB (Deutsch and Journel, 1998) is

$$\gamma(\boldsymbol{h}) = C\left\{1 - \exp\left[-\left(\frac{3\boldsymbol{h}}{a}\right)^2\right]\right\} \tag{3}$$

where C is the sill [m^2] and a is the range [m]. The variogram modeling is performed using Surfer, which models the Gaussian variogram model without the factor 3 in the exponential. The Gaussian model fit to the experimental variogram, computed from the residual heads, is shown in Figure 2-5. This model has a nugget value of 0.1 m^2, a sill of 40 m^2 and an effective range of 7500 m (the 2004 report had a nugget of 13.0 m^2, a sill of 45.2 m^2, and an effective range of 9000 m). The numbers of data pairs used in the calculation of each point in the experimental variogram are also shown. The calculation of the experimental variogram was done by considering combinations of pairs of data points in all directions. By not considering direction, only distance, the variogram is an omnidirectional calculation using h, where h is the length of the vector ***h***. An omnidirectional variogram was also used in the 2004 version of this analysis.

Although a small number of pairs (<30) exist for many of the shorter lag spacing in Figure 2-5, it is felt this variogram is still valid and representative. The only short-lag observation pairs are the redundant wells on the H-19 wellpad. These wells were included in the variogram analysis to get some approximation of the short-lag behavior of freshwater head residual, coupled with the knowledge that the freshwater head residual is a smooth function (i.e., the reason the differentiable Gaussian variogram was used in the first place). The model variogram used in 2004 had a much larger nugget value than the current model does (13 m^2 – compared to 0.1 m^2), but the 2004 model did not use the redundant H-19 wells, which solely contribute to the short-lag experimental variogram.

Although it is possible to calculate directionally dependent variograms, this was not done. The steep north-south hydraulic head gradient observed in the Culebra across the WIPP site is coincident with the densest clustering of observation wells (see steep segment in the center of Figure 2-1). This produces a greater east-west correlation between data compared to correlation in the north-south direction (across the steep gradient) for the entire domain. Since most of the domain where kriging is being used to estimate values is outside the LWB, an anisotropic model would be misrepresentative of this apparent anisotropy, although it may fit the observed data. Although kriging effectively handles clustered data during the estimation process, the effects which data clustering can have on the variogram modeling process must be considered by the analyst.



lag width = 400 m | lag width = 1000 m | lag width = 2000 m

Figure 2-5. Experimental variograms (points) and best-fit Gaussian model variogram (lines) for three different lag widths. NB: there is a factor-of-three difference in definition of variograms between Surfer and GSLIB (multiply lag by 3.0).

It is possible to fit different models or models with different parameters to the same data, but it is felt that the choice of variogram model and parameters given here sufficiently represents the data and corroborates with the presumed knowledge of the system. If a different type of surface were fit to the data, the residuals would have a different structure and therefore a different variogram as well.

Following up on results of the trend surface sensitivity to removing a steel-cased well (see Section 2.1), the experimental variogram is re-computed for each well removed and shown in Figure 2-6. The exact values of the experimental variograms are not important, just the qualitative observation that the relative variability between the different experimental variograms is small. Both the change in the best-fit surface, and the subsequent changes in the variogram of head residuals, due to removing (or adding) a single observation will diminish as the dataset becomes larger. For the current dataset of over 40 monitoring points, the variogram is virtually unchanged upon removal of a steel well, re-calculation of the trend surface (see Figure 2-4) and residuals and model variogram calculation (see Figure 2-6).

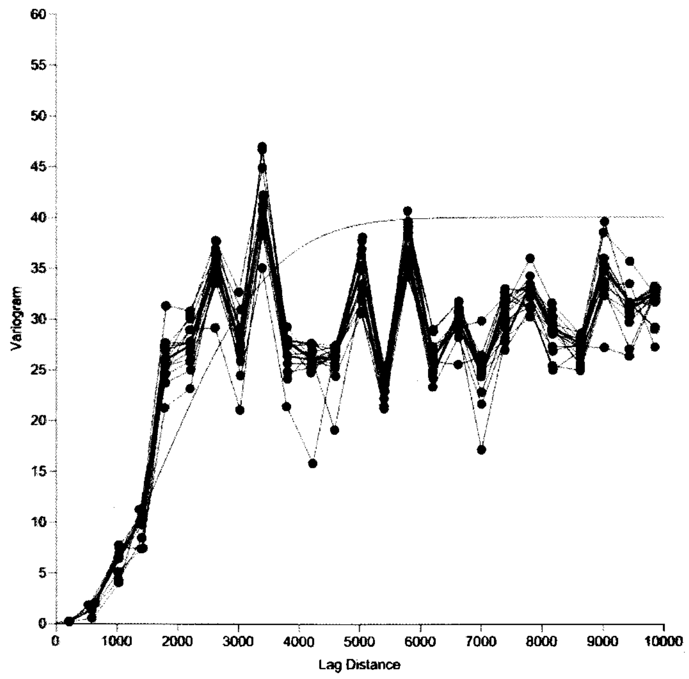


Figure 2-6. Experimental variograms after removing a steel-cased well (variogram for all wells shown in red). Residuals are re-computed based on the best-fit linear trend for each new set of wells and the variogram is re-computed for each new set of residuals.

2.3. *Ordinary Kriging*

Kriging is a geostatistical algorithm for estimating a property at unsampled locations. The kriging equations are formulated to provide an unbiased, minimum variance estimate of the property from a linear combination of the surrounding measured data. Kriging additionally provides a measure of the uncertainty associated with each estimate. The uncertainty measure is known as the kriging variance or the estimation variance. Details on the many variants of the kriging algorithm and its application can be found in the literature, e.g., (Deutsch and Journel, 1998; Goovaerts, 1998). For this work, we use ordinary kriging (OK) and the details of the OK algorithm are presented briefly.

Consider the problem of estimating the value of a continuous attribute, *z*, (e.g. head residual) at an unsampled location **u**. The information available consists of measurements of *z* at *n* locations $\mathbf{u}_\alpha$, $z(\mathbf{u}_\alpha)$, $\alpha = 1,2, ..., n$. Kriging is a form of generalized least-squares regression and therefore all univariate kriging estimates are variants of the general linear regression estimate $z^*(\mathbf{u})$ defined as

$$z^*(\mathbf{u}) - m(\mathbf{u}) = \sum_{\alpha=1}^{n(\mathbf{u})} \lambda_\alpha(\mathbf{u})\left[z(\mathbf{u}_\alpha) - m(\mathbf{u}_\alpha)\right] \tag{4}$$

where $\lambda_\alpha(\mathbf{u})$ is the dimensionless weight indicating the contribution of $z(\mathbf{u}_\alpha)$ - $m(\mathbf{u}_\alpha)$ to the estimate of z^* (*z* at unsampled locations), and $m(\mathbf{u})$ is the trend or mean component of the spatially varying attribute [m].

The most common kriging estimator is OK, which estimates the unsampled value $z^*(\mathbf{u})$ as a linear combination of neighboring observations

$$z^*_{OK}(\mathbf{u}) = \sum_{\alpha=1}^{n(\mathbf{u})} \lambda_\alpha(\mathbf{u}) z(\mathbf{u}_\alpha) \tag{5}$$

OK weights λ_α are determined so as to minimize the error or estimation variance $\sigma^2(\mathbf{u}) = \mathrm{Var}\{z^*(\mathbf{u}) - z(\mathbf{u})\}$ under the constraint of unbiasedness of the estimate. These weights are obtained by solving a system of linear equations, which is known as the ordinary kriging system of equations. Solution of the kriging system requires that covariance, Cov($\boldsymbol{u}_\alpha$,$\boldsymbol{u}_\beta$), between any two locations be calculated. Covariance is derived from the variogram model under an assumption of second-order stationarity. The unbiasedness of the OK estimator is ensured by constraining the weights to sum to one, which requires the definition of the Lagrange parameter $\mu(\mathbf{u})$ within the system of equations (Bazaraa et al., 1993),

$$\begin{cases} \sum_{\beta=1}^{n(\mathbf{u})} \lambda_\beta(\mathbf{u})\gamma(\mathbf{u}_\alpha - \mathbf{u}_\beta) - \mu(\mathbf{u}) = \gamma(\mathbf{u}_\alpha - \mathbf{u}) \qquad \alpha = 1,\ldots,n(\mathbf{u}) \\ \sum_{\beta=1}^{n(\mathbf{u})} \lambda_\beta(\mathbf{u}) = 1. \end{cases} \tag{6}$$

The kriging variance is also derived from the set of weights and the Lagrange parameter determined through solution of (6) and it is given as:

$$\sigma^2_{OK}(\mathbf{u}) = \mathrm{Cov}(\mathbf{u},\mathbf{u}) - \sum_{\alpha=1}^{N} \lambda_\alpha \mathrm{Cov}(\mathbf{u},\mathbf{u}_\alpha) - \mu \tag{7}$$

The covariance [m^2] used to calculate the ordinary kriging variance is derived from the model variogram. The covariance between two points separated by zero lag, Cov(**u**,**u**) = Cov(0) is equal to the variance of the data set. It is important to note that the OK variance is not a direct function of the specific data values, other than how those data values define the experimental variogram of the residuals (see discussion associated with Figure 2-6), to which the model variogram is fit.

2.4. Estimation Variance Calculations

The program KT3D (Deutsch and Journel, 1998) is used with the model variogram determined above (estimated and plotted using Surfer) to calculate both the estimated residuals and variance at all locations. The full calculation domain is 87188 100-m × 100-m cells, with 36213 of those cells (41 percent) inactive, lying either beyond the no-flow boundary on the west or the composite H2/M2 – H3/M3 Rustler halite margins on the east. Those inactive cells are not included in the calculations of estimation variance. For the calculations done herein, the average estimation variance both within the flow domain and within the WIPP site are calculated for different monitoring well configurations.

The map of estimation variance for the May 2007 monitoring network defined in Table 2-2 is shown in Figure 2-7. The effect of the monitoring network configuration on the resulting estimates of variance is obvious. The lowest estimation variance values (blue) occur at the well locations and the highest values (red) occur at locations that are beyond the distance of the variogram range (7500 m) away from existing wells. The minimum possible value of the kriging variance is the value of the nugget in the variogram model (0.1 m^2). The maximum kriging variance in these calculations is approximately 46.4 m^2. In the following analysis, the actual values of the kriging variance are not significant, it is only the relative changes in the kriging variance due to the addition, or subtraction, of wells to or from the monitoring network that are of interest.

The full monitoring network of 48 wells and the model variogram calculated from the head residuals at those wells produce an average estimation variance within the flow domain of 29.1 m^2 and an average estimation variance within the land withdrawal boundary of 7.0 m^2. From Figure 2-7 it is obvious that there are many locations outside of the WIPP site where the addition of a well would have large impact on the estimation variance. Within the WIPP site, the estimation variance is already relatively low at nearly all locations. In fact, given the small distances between some wells relative to the range of the variogram, it is possible to remove some of the existing wells within the WIPP site boundary with only minimal increase in the estimation variance.

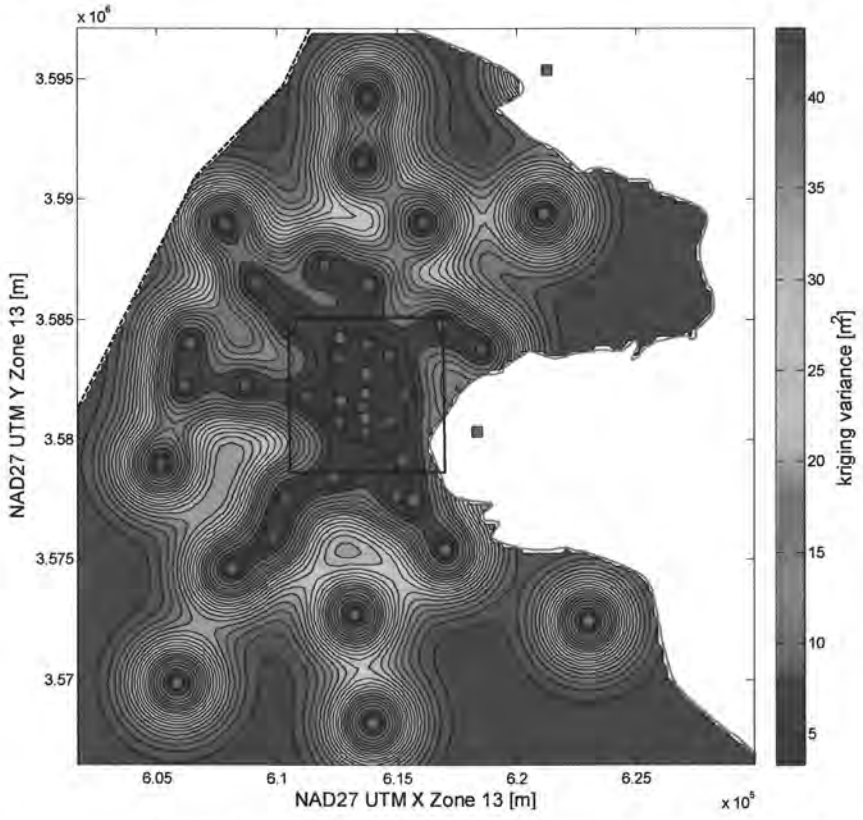


Figure 2-7. Kriging estimation variance for freshwater head residuals. Steel-cased wells are red circles, fiberglass wells are green squares, WIPP LWB is solid black line.

2.4.1. Add one new well

Any proposed new well locations can be added to the current well network and the estimation variance can be recalculated including the additional point. This approach takes advantage of the fact that the estimation variance does not depend on the data values, only on their spatial configuration. This approach does require the assumption that the model variogram does not change significantly with the addition of new locations (analogous to the qualitative sensitivity study illustrated in Figure 2-4 and Figure 2-6 for the case of removing one well).

Figure 2-8 and Figure 2-9 show the relative effects an additional observation point has on the model-domain-wide mean and median of the kriging variance, computed as

$$\Delta\sigma_{+1}^2 = -\frac{\langle\sigma_{+1}^2\rangle - \langle\sigma_{base}^2\rangle}{\langle\sigma_{base}^2\rangle} \tag{8}$$

where σ_{+1}^2 is the kriging variance for the case with one additional well, σ_{base}^2 is the variance for the base case with the 2007 well network and $\langle x\rangle$ is the averaging operator (in this case averaged over the model domain). Steel-cased wells are red circles, fiberglass-cased wells are green squares. The contours in these figures illustrate the decrease in the domain-wide average variance, not the distribution of the variance due to any one distribution of wells. While the mean and median largely show the same trends, the median values are larger and their distribution is less sensitive to a few extreme high or low values, which can skew the mean. Areas with a small average decrease (i.e., inside and near the LWB) indicate an additional observation at these locations would not significantly improve the estimation variance, averaged over the entire domain. The highest values (i.e., the red 'bulls-eye' areas in Figure 2-8 and Figure 2-9) are located midway between wells along the periphery of the monitoring well network.

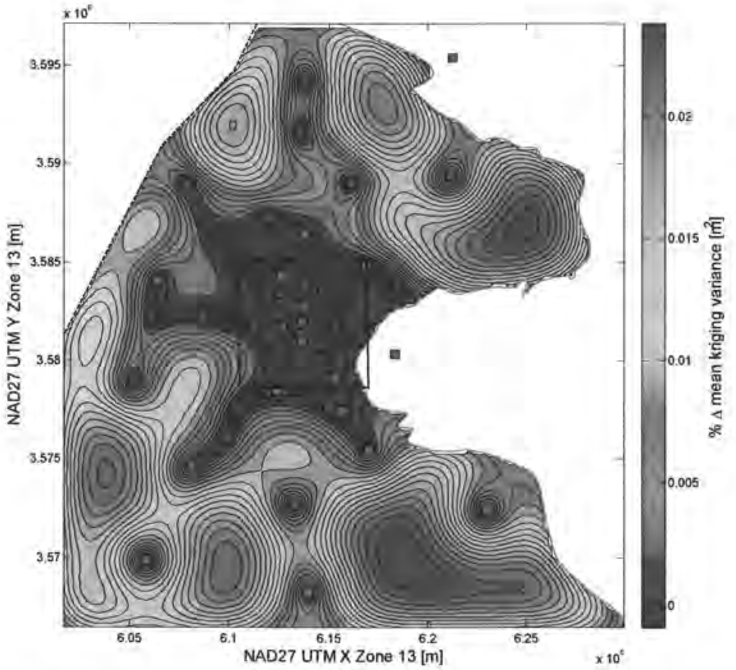


Figure 2-8. Percent decrease in mean kriging variance over model domain due to one additional well

Figure 2-8 and Figure 2-9 account for edge effects near the boundaries of the domain. Areas of high kriging variance (i.e., the red regions in Figure 2-7) mostly correspond to the areas where the largest mean change in variance occurs upon the addition of a new observation point. In the corners of the model domain (especially the southeast corner), the increase is not so large, indicating that much of the effects of a new observation point at this location would be "wasted" outside the model domain.

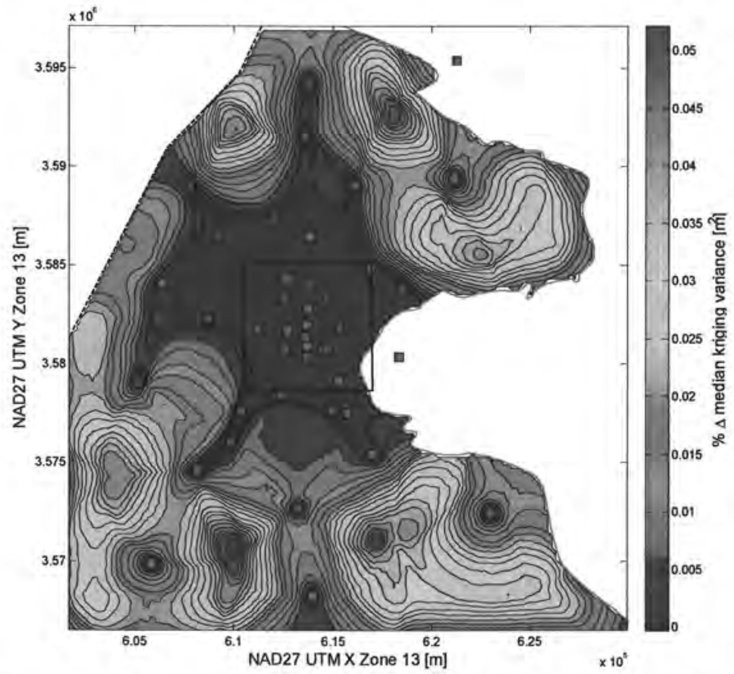


Figure 2-9. Percent decrease in median kriging variance over model domain due to one additional well

Figure 2-10 shows the fractional change in the standard deviation of the kriging variance due to the addition of one more observation location. The standard deviation of the values in the kriging variance field is computed across the entire matrix of values corresponding to adding one additional observation location, without regard to spatial distribution of values. A negative value (blue) indicates the additional location will "smooth out" the kriging variance field (lowering its standard deviation), while a positive number (red) indicates the kriging variance field becomes more variable; a heavy black line indicates the zero-change contour. The positive (red) regions are located in areas distant to the WIPP LWB, and indicate where an additional well would "extend" the current network. The negative regions (blue) are located closer to the WIPP LWB, and indicate where an additional well would "fill in a gap" in the current network.

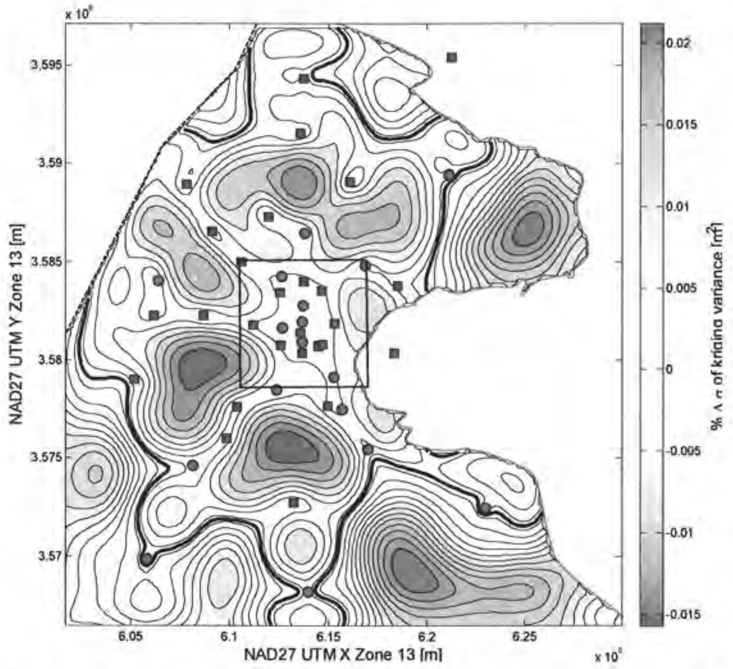


Figure 2-10. Percent change in standard deviation of kriging variance over model domain due to one additional well; heavy black line is zero contour.

Similar calculations were also done with the WIPP LWB as the area of interest, excluding a small area in the southeast corner of the LWB that is constant head in the Culebra MODFLOW model. The region that affects the results within the WIPP LWB is confined to the center of the model domain. The edge effects are clearly evident for the case of averaging over the WIPP LWB, only the LWB and a 3.5-km region surrounding the LWB are shown in Figure 2-11 through Figure 2-13.

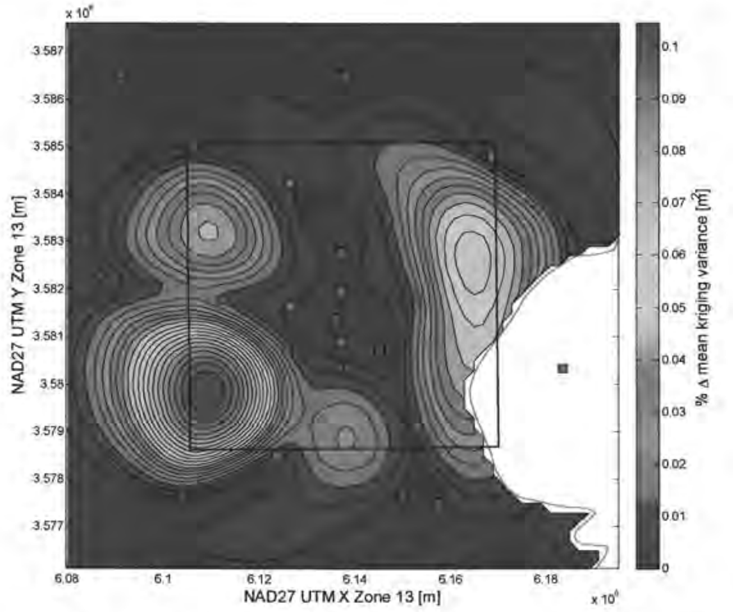


Figure 2-11. Change in mean kriging variance over WIPP LWB due to one additional well

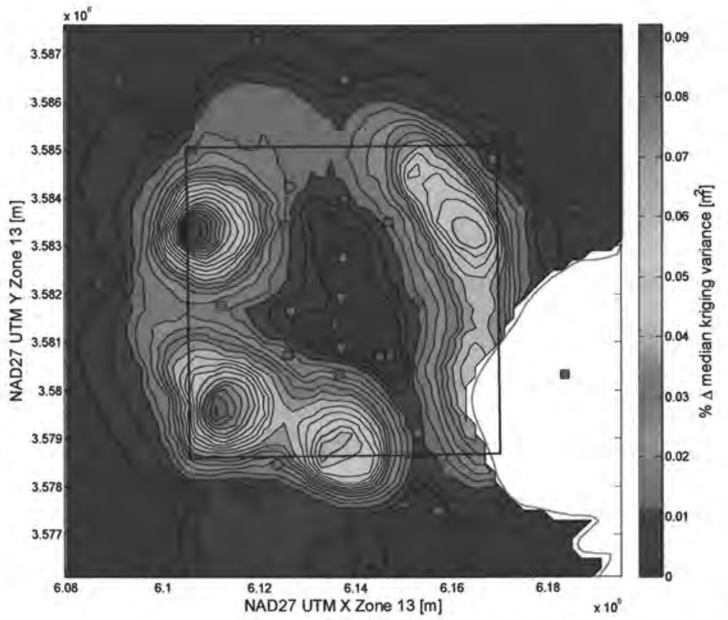


Figure 2-12. Change in median kriging variance over WIPP LWB due to one additional well

Figure 2-11, Figure 2-12, and Figure 2-13 indicate the southwest corner of the LWB is the location that would maximally benefit from an additional monitoring location. The northeast corner is next in relative importance for a new location. Figure 2-12 indicates that the northwest corner would have the largest effect on the median kriging variance across the WIPP LWB. All three figures indicate that the areas along the periphery of the LWB, where there are no existing wells would be good locations for reducing uncertainty through an additional observation point, because a large number of wells already exist in the center of the WIPP LWB.

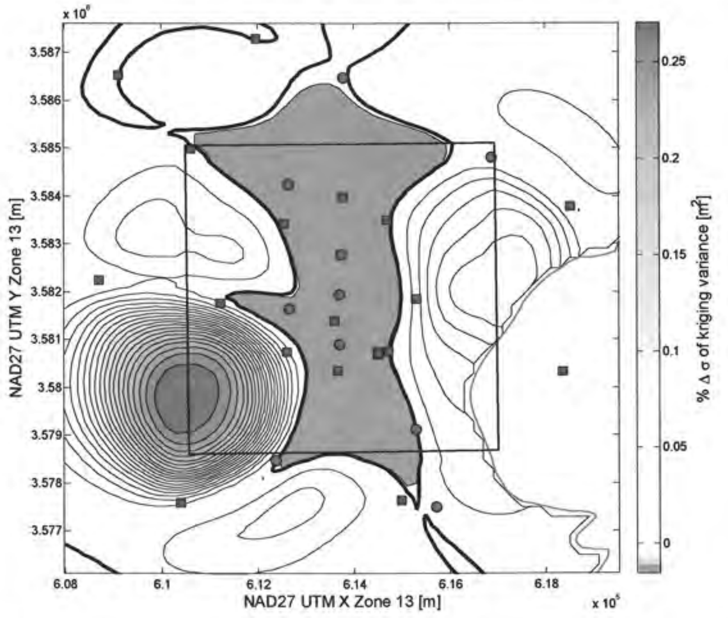


Figure 2-13. Change in standard deviation of kriging variance over WIPP LWB due to one additional well

2.4.2. Remove one steel well

The same approach for determining the variance reduction due to the addition of a new monitoring well can also be used to compute the potential increase in the estimation variance from the removal of an existing well. In this case, it is possible to recalculate the variogram model from the remaining wells after any number of wells are removed; however, to make the process more efficient, the same variogram is used for all calculations done herein. This approach assumes that the variogram does not change significantly with the loss of any one of the wells (see discussion associated with Figure 2-6).

Each existing steel-cased well is removed and the average estimation variances across the flow domain and the WIPP site are recalculated. Those wells that cause the smallest increase in average estimation variance are the ones that could be removed with a minimal impact on the ability of the monitoring network to provide accurate predictions of heads at locations without monitoring wells. The results of these calculations are shown in Table 2-3.

Table 2-3 shows the change in the average estimation variance within the flow domain as well as within the WIPP site area as calculated for the less-by-one networks associated with removing steel-cased wells. Removal of fiberglass-cased wells is not considered, since they are expected to have a long useful life. Table 2-4 shows the same results only averaged over the WIPP LWB when steel-cased wells are removed from the network. Removal of wells that result in the largest increases in the estimation variance are the wells that are most important with respect to the ability of the network to predict heads. Therefore, if the goal is to predict heads across the entire domain, the wells that create the largest increases in estimation variance when removed are generally those located distant from other wells: H-10c, USGS-4, H-9c, AEC-7, H-11b4, and WIPP-11. Small decreases in the estimation variance can also occur with the removal of a well (e.g., ERDA-9, H-3b2, H-2b2, and WIPP-19). These decreases are due to the configuration of the current wells creating negative kriging weights in the solution of kriging equations (see positive values in mean and median columns of Table 2-3 and Table 2-4). These decreases are always less than two-tenths of one percent of the original variance and are considered as insignificant near-zero changes in this work.

Table 2-3. Results of estimation variance changes over the entire model domain for the removal of one steel-cased well from the network. A large integer rank indicates an important well, while a small rank is associated with wells with little impact on the entire model domain.

	|Δ standard deviation|		Δ mean		Δ median		avg rank
ERDA-9	0.48%	2	0.19%	1	0.13%	2	1.67
H-3b2	0.49%	3	0.19%	2	0.13%	1	2.00
H-2b2	0.52%	4	0.18%	3	0.13%	2	3.00
WIPP-19	0.75%	5	0.09%	4	0.13%	2	3.67
H-17	1.36%	9	-0.24%	5	0.02%	6	6.67
H-12	0.46%	1	-1.78%	10	-3.13%	11	7.33
H-4b	1.46%	10	-0.36%	6	-0.13%	7	7.67
WIPP-25	1.04%	7	-1.10%	8	-1.49%	9	8.00
WIPP-13	2.00%	13	-0.70%	7	0.07%	5	8.33
H-7b1	1.22%	8	-2.13%	11	-2.54%	10	9.67
H-5b	2.51%	15	-1.26%	9	-1.09%	8	10.67
WIPP-11	0.80%	6	-3.26%	14	-3.52%	13	11.00
H-11b4	1.67%	11	-2.62%	13	-3.47%	12	12.00
AEC-7	2.15%	14	-2.52%	12	-3.86%	14	13.33
H-9c	1.88%	12	-3.97%	15	-7.20%	16	14.33
USGS-4	2.59%	16	-4.39%	16	-6.98%	15	15.67
H-10c	3.90%	17	-4.88%	17	-7.80%	17	17.00

The five wells that could be removed from the network and create the smallest increase in the mean or median estimation variance are those wells in close proximity to other existing wells. These include: ERDA-9, H-2b2, H-3b2, H-17, and WIPP-19 (see Table 2-3 and Figure 2-14). The change in the standard deviation of the kriging variance is shown in Figure 2-14.

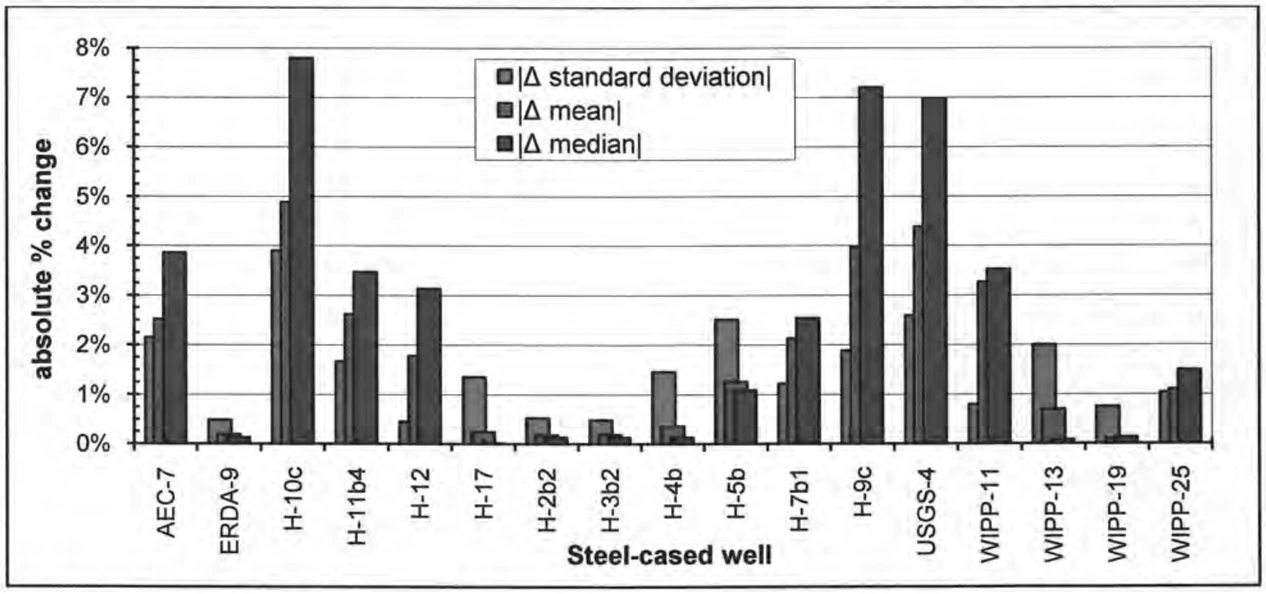


Figure 2-14. Change in estimation variance averaged across entire model domain (data from Table 2-3)

The wells on the other end of the spectrum, which would have the largest effect on the mean or median estimation variance, include wells on the periphery of the domain (i.e., large bars in Figure 2-14).

Table 2-4. Results of estimation variance changes over WIPP Land Withdrawal Boundary for removal of one well from the network. A high rank indicates importance while a low rank indicates little impact on the area within the LWB.

	|Δ standard deviation|		Δ mean		Δ median		avg rank
H-10c	0.0001%	1	0.00001%	1	0%	1	1.00
AEC-7	0.0001%	2	-0.0001%	2	0%	1	1.67
USGS-4	0.01%	3	-0.02%	3	0%	1	2.33
H-9c	0.48%	7	-0.09%	4	-0.02%	5	5.33
WIPP-11	0.23%	4	-0.72%	5	-0.65%	8	5.67
WIPP-25	0.84%	8	-0.89%	6	-0.01%	4	6.00
H-3b2	0.33%	6	-1.32%	7	-0.51%	7	6.67
ERDA-9	0.87%	9	-1.37%	8	-0.42%	6	7.67
H-2b2	1.07%	10	-1.88%	9	-0.90%	9	9.33
WIPP-13	0.27%	5	-2.83%	10	-5.67%	16	10.33
WIPP-19	1.65%	11	-2.85%	11	-4.31%	11	11.00
H-7b1	24.04%	14	-7.51%	13	-1.57%	10	12.33
H-11b4	11.87%	12	-7.35%	12	-5.64%	14	12.67
H-12	12.88%	13	-7.70%	14	-5.66%	15	14.00
H-4b	28.12%	15	-11.73%	16	-4.65%	12	14.33
H-17	29.34%	16	-11.57%	15	-4.75%	13	14.67
H-5b	47.64%	17	-19.15%	17	-6.72%	17	17.00

The removal of wells far from the WIPP site creates the largest increases in the estimation variance averaged over the flow domain, but the removal of many of these steel-cased wells has little or no effect on the estimation variance averaged across the WIPP site. These wells, AEC-7, H-9c, H-10c, USGS-4, and WIPP-25 are too far away from the WIPP site to directly impact the mean or median estimation variance inside the LWB. The most important monitoring wells, those that create the largest variance mean or median increase upon removal, for predicting heads within the WIPP site are: H-5b, H-4b, H-11b4, H-12, and H-17. All these wells are located near the LWB (H-12 being the furthest away from the LWB). Some of the wells have very large effects on the standard deviation of the kriging variance within the site, but the trends also follow those for the mean and median kriging variance.

Figure 2-15 summarizes the results in Table 2-3 and Table 2-4 graphically, indicating where the wells with high or low rank are located geographically.

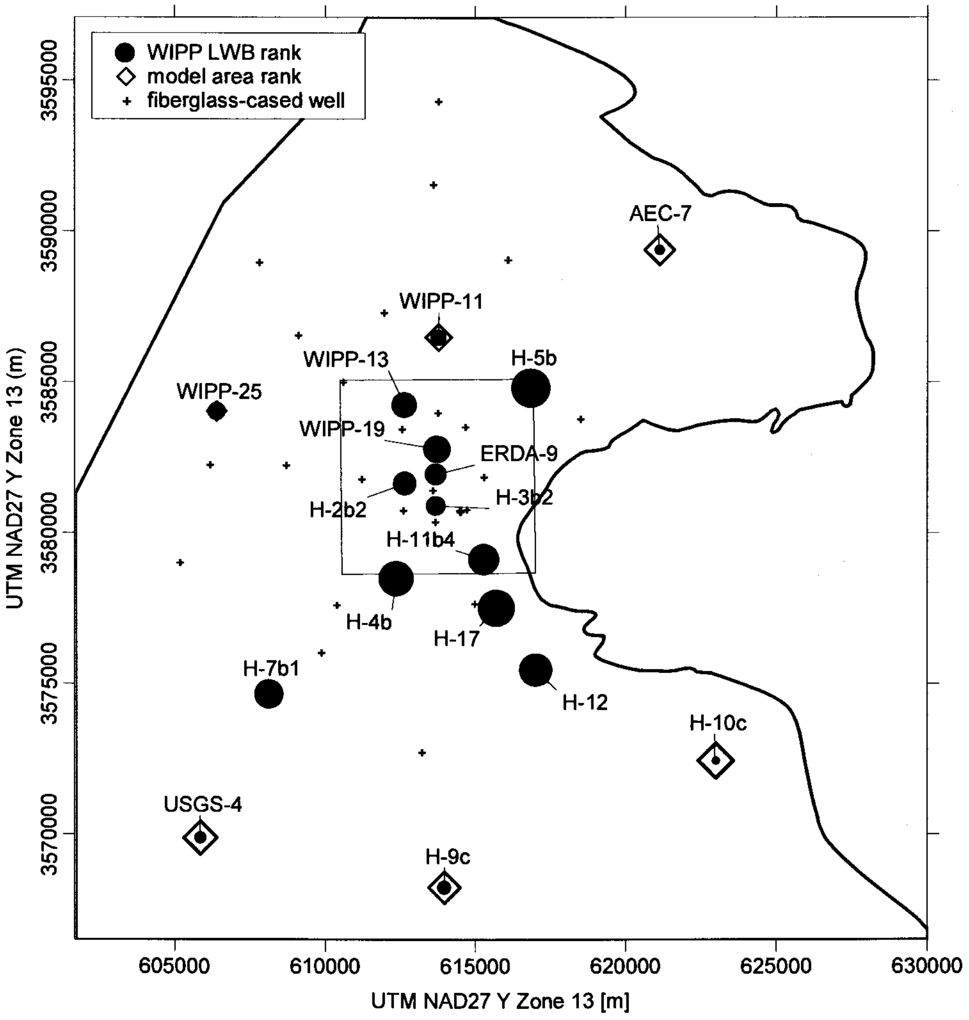


Figure 2-15. Steel-cased wells ranked by effect of removal on kriging variance; symbol sizes are proportional to overall rank (and therefore importance), data in Table 2-3 and Table 2-4. MODFLOW active model domain delineated in green, WIPP LWB is black square.

The wells that create the smallest increases in estimation variance upon removal for both the WIPP site and the flow domain are: ERDA-9, H-2b2, and H-3b2. Any one of these three wells could be removed with minimal effect on the ability of the network to predict heads across both the domain and the WIPP site. These calculations are for removal of a single well.

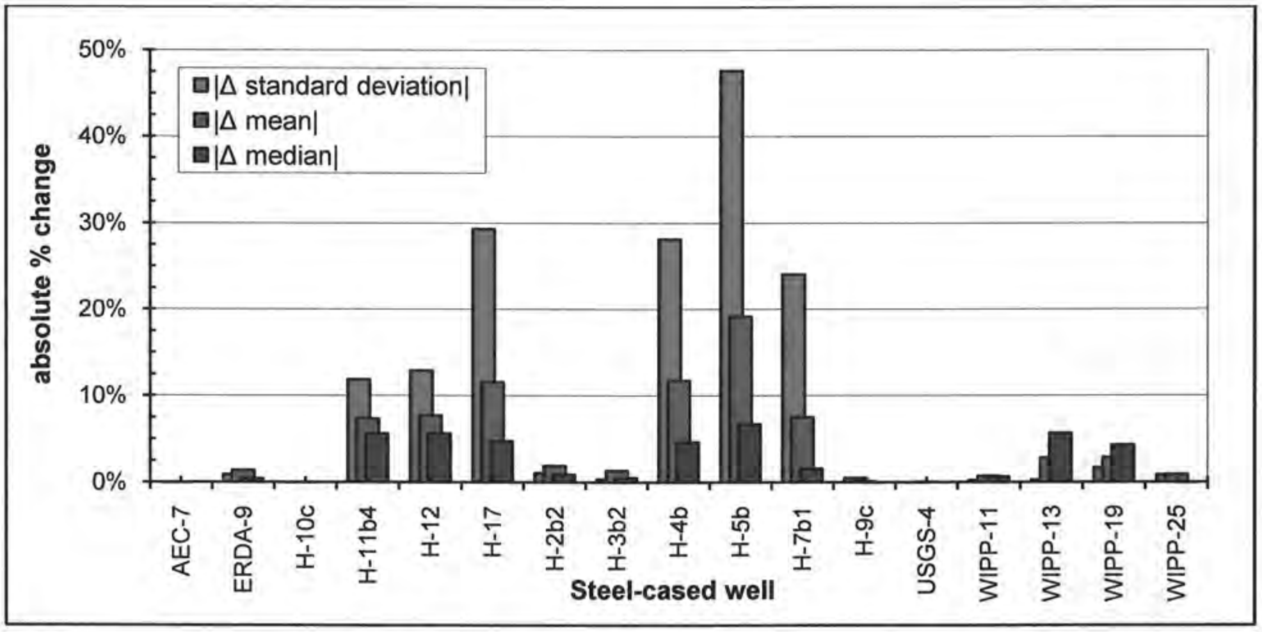


Figure 2-16. Change in estimation variance averaged across WIPP LWB, data in Table 2-4

2.4.3. Remove Two Steel Wells

Based on expectations that the following steel wells will soon need to be replaced or P&Aed: WIPP-25, WIPP-13, H-12 and H-7b1; the analysis of the previous section is carried out removing each of these wells, then additionally removing each of the remaining steel-cased wells one at a time.

Table 2-5 shows the percent change in the kriging variance averaged across the entire model domain computed as (changed - base)/base for combinations of steel-cased wells being removed. The 4×17 image shows the same results in the table. This shows that removing AEC-7 and H-9c, along with any of the three wells represented as columns, makes a large relative change across the entire model domain. This is to be expected, as both of these wells are far away from other wells; removing two wells a large distance from each other affects the largest potential area. Conversely, the column corresponding to WIPP-13 leads to the smallest relative changes (some even being slightly positive), indicating the kriging area of influence for this well has a large amount of overlap with those from other wells, since this well is located in the north-central portion of the WIPP LWB.

Table 2-5. Change in mean kriging variance across model domain upon removal of two steel-cased wells; integer values indicate rank within each column

		H-12		H-7b1		WIPP-13		WIPP-25	
1	AEC-7	-4.15%	15	-4.17%	15	-2.49%	14	-3.70%	14
2	ERDA-9	-1.52%	3	-1.53%	3	0.11%	2	-1.09%	2
3	H-10c	-4.18%	16	-1.73%	8	-2.51%	15	-3.72%	15
4	H-11b4	-1.78%	8	-3.27%	14	-0.09%	5	-1.29%	6
5	H-12			-1.76%	9	-1.61%	11	-2.81%	11
6	H-17	-1.51%	1	-1.52%	1	-0.10%	6	-1.30%	7
7	H-2b2	-1.53%	5	-1.55%	5	0.09%	3	-1.10%	4
8	H-3b2	-1.51%	2	-1.53%	2	0.12%	1	-1.08%	1
9	H-4b	-2.09%	9	-2.10%	10	-0.46%	7	-1.65%	8
10	H-5b	-2.41%	11	-2.43%	12	-0.78%	9	-1.98%	10
11	H-7b1	-3.27%	13			-1.62%	12	-2.84%	12
12	H-9c	-3.95%	14	-4.19%	16	-2.29%	13	-3.51%	13
13	USGS-4	-2.31%	10	-2.33%	11	-2.56%	16	-3.78%	16
14	WIPP-11	-1.61%	7	-1.62%	7	-0.70%	8	-1.88%	9
15	WIPP-13	-1.53%	4	-1.55%	4			-1.18%	5
16	WIPP-19	-2.81%	12	-2.84%	13	-1.18%	10	-1.10%	3
17	WIPP-25	-1.58%	6	-1.60%	6	-0.05%	4		

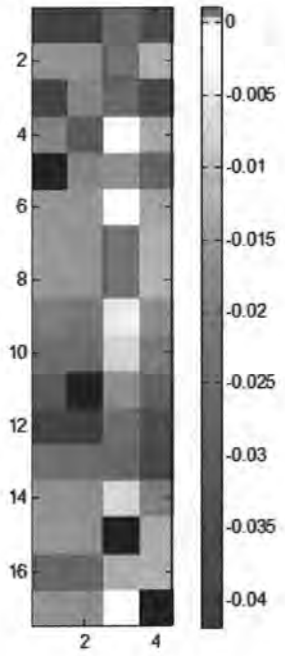
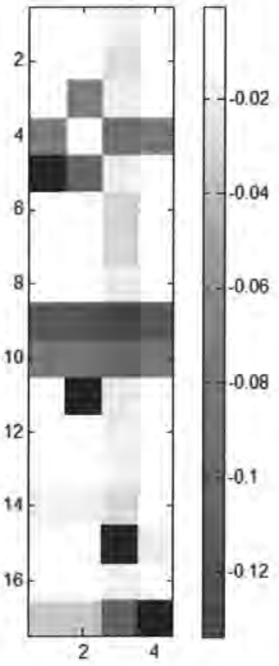


Table 2-6 gives the same statistics as in Table 2-5, but the results are averaged over the area within the WIPP LWB only (rather than the entire model domain). The conclusions from this analysis are different, since WIPP-13 is now the column which on average is darkest blue, (rather than lightest in Table 2-5). Wells H-4b and H-5b are the two "row" wells which have the largest negative change across the four columns. These wells are located on or near the WIPP LWB, similar to how AEC-7 and H-9c from the analysis in the previous paragraph, are located along the periphery of the model domain.

Table 2-6. Change in mean kriging variance across WIPP LWB upon removal of two steel-cased wells; integer values indicate rank within each column

		H-12		H-7b1		WIPP-13		WIPP-25	
1	AEC-7	-0.03%	3	-0.09%	2	-1.95%	2	0.00%	2
2	ERDA-9	-0.69%	8	-0.76%	7	-2.65%	9	-0.67%	8
3	H-10c	-0.03%	1	-7.45%	13	-1.95%	1	0.00%	1
4	H-11b4	-7.70%	14	-0.12%	4	-9.32%	13	-7.36%	14
5	H-12			-10.50%	15	-1.97%	6	-0.03%	5
6	H-17	-0.08%	5	-0.14%	5	-3.07%	12	-1.13%	12
7	H-2b2	-1.05%	11	-1.12%	10	-2.99%	11	-1.03%	11
8	H-3b2	-0.30%	7	-0.37%	6	-2.22%	8	-0.28%	7
9	H-4b	-11.49%	16	-11.64%	16	-13.40%	16	-11.46%	16
10	H-5b	-7.71%	15	-7.78%	14	-9.66%	14	-7.69%	15
11	H-7b1	-0.12%	6			-2.04%	7	-0.10%	6
12	H-9c	-0.03%	2	-0.09%	1	-1.95%	4	0.00%	4
13	USGS-4	-0.75%	9	-0.81%	8	-1.95%	3	0.00%	3
14	WIPP-11	-1.97%	12	-2.04%	11	-2.83%	10	-0.72%	9
15	WIPP-13	-0.92%	10	-0.98%	9			-1.95%	13
16	WIPP-19	-0.03%	4	-0.10%	3	-1.95%	5	-0.89%	10
17	WIPP-25	-3.35%	13	-3.41%	12	-10.32%	15		



In Figure 2-17 and Figure 2-18 the overall rank (between all 64 combinations of two steel-cased wells, rather than 16 steel-cased wells individually) is plotted against the distance between the two wells comprising the pair as a measure of the impact of removing a pair of wells. The overall rank is computed as an average of the results of three metrics (mean, median and standard deviation of the change in the kriging variance).

Figure 2-17 shows a weak positive correlation ($R^2=0.35$ fit through all points) between the relative impact of removing a pair of steel wells and the distance between those wells, when the impact is averaged over the entire model domain. A high rank indicates a higher impact to the kriging variance averaged over the model domain; a low rank indicates low impact. Wells which are separated by large distances tend to have the largest cumulative effect. The different symbols in the figure show that individual wells (these four symbols correspond to the wells represented as columns in Table 2-5) tend to also follow this trend

Figure 2-18 shows the same results, only averaging the impact over the area encompassed by the WIPP LWB. Here the correlation is even weaker, and also negative. Wells that are closer together have a larger relative impact when they are removed. WIPP-13 (black triangles) is only associated with higher rank pairs (>32), but in general all the individual wells follow the weak overall trend.

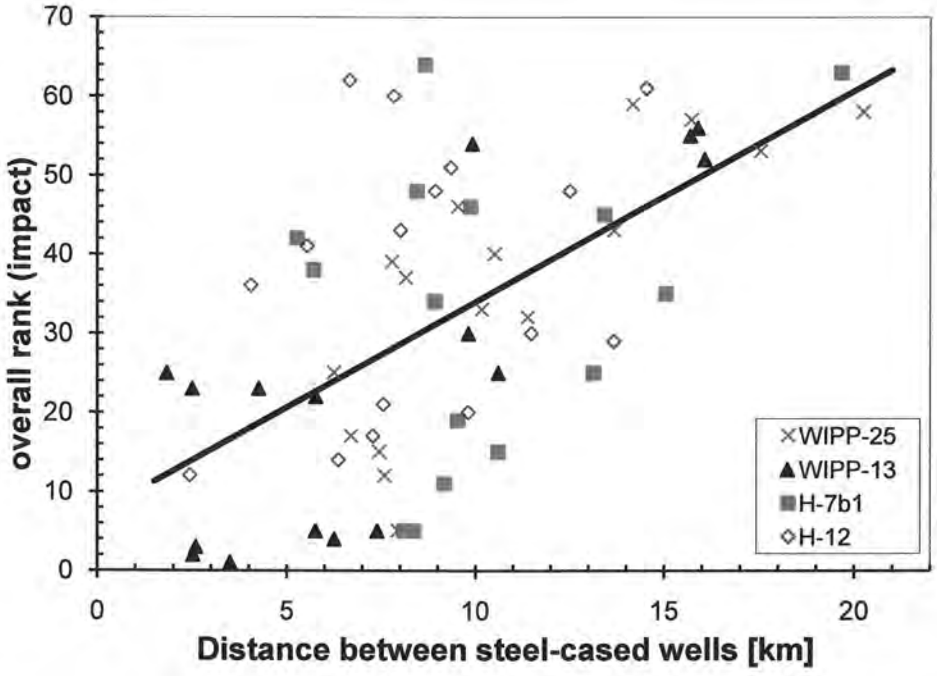


Figure 2-17. Rank and distance between wells for removal of pairs of steel-cased wells, averaging effects across the entire model domain (R^2=0.35 for linear fit). High numerical rank indicates a large impact on the model-wide kriging variance.

WIPP-25, H-12, and H-7b1 are all located outside the WIPP LWB. WIPP-13 is inside the WIPP LWB, and all pairs of wells including WIPP-13 have high rank in Figure 2-18. The smallest linear dimension across the WIPP LWB is 6.4 km (east-west or north-south), while the largest linear dimension of the WIPP LWB would be a diagonal across the site 9.09 km. Pairs of wells with distances larger than these values must include at least one well outside the LWB, possibly both. The pairs of wells with very large separations are pairs of wells where both wells are distant from the WIPP site. Removing these wells would obviously have little direct impact on the region inside the WIPP LWB.

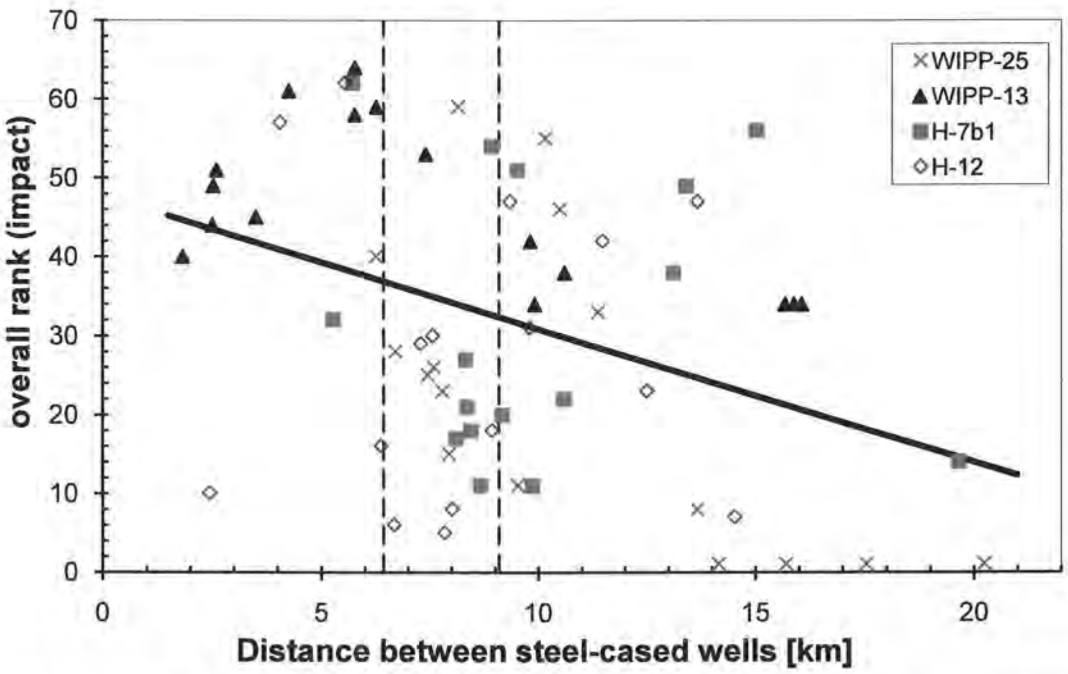


Figure 2-18. Rank and distance between pairs of steel-cased wells, considering WIPP LWB (R^2=0.14 for linear fit). Vertical dashed lines represent the min (6.4 km) and max (9.1 km) distances across the LWB. High numerical rank indicates a large impact on the kriging variance inside the WIPP LWB.

This analysis supports the idea that removing two distant steel-cased wells from the network has the largest impact on the kriging variance, averaged over the entire model domain. The relationship that inter-well distance plays in the removing of two wells, when only considering the area within the LWB is more complex and less conclusive. Apparently, there is a negative correlation between distance and importance, but likely because wells with large inter-well separations are likely distant to the WIPP LWB as well.

All the well-removal scenarios in this section have the assumption that the variogram does not change upon removal of the selected wells. In situations where we are only removing one well, this is a very good assumption (see discussion associated with Figure 2-6); removing two or more wells still should not violate the assumption that their removal would not change the variogram.

2.5. Kriging Variance Reduction Summary

It is relatively simple to calculate the decrease or increase in the kriging estimation variance over a specified area from the addition or removal of one or more monitoring wells. The maximum reduction in estimation variance, or increase in the ability to predict heads, can be achieved by placing a new monitoring well in any location of the flow domain that is far away from an existing well or the model boundary. There are a large number of locations in the domain where a new well could be placed to meet this condition. At this point in the analysis, a maximal reduction in variance from a new well can be considered as a necessary input, but not sufficient condition for locating a new well.

Removal of wells from the existing monitoring network was also examined using the kriging estimation variance. The impact of well removal was evaluated by calculating the increase in estimation variance for both the entire flow domain and the area within the WIPP LWB. These calculations were done for the removal of a single well from a base case of 42 wells and the results are only valid for the removal of the one specified well. These results also safely assume that the variogram is constant across all monitoring network configurations (see Figure 2-6). These calculations were completed again for removal of combinations of multiple wells when those combinations of interest are defined. Wells that are most important to the existing monitoring network that should not be removed are listed above and are, generally, those wells most distant from any existing wells. Wells that have the smallest influence on the ability of the current network to predict heads at unmeasured locations across the entire flow domain as well as within the WIPP site are also listed above. If more than one well is to be removed, the combinations of wells should be selected from this list.

2.6. Kriging Variance Reduction Run Control Summary

The kriging variance reduction analysis performed in this section is described here in terms of files, programs, and scripts used. The required files are located on the CD and are described in sufficient detail to allow recreation of the results given in the text.

2.6.1. Linear trend fit and variogram calculations

The linear trend fitting to the computed freshwater head values (see Section 2.1, Table 2-1, and Figure 2-3) was computed in the COTS statistical software R. The script `plot_linear_fit_summary.R` (Section 8.2.1) uses the built-in linear model function `lm()` to produce a linear fit, then standard statistics are produced by summarizing this fit (see see Table 2-1 produced by `summary()` in line 15 of script) and standard diagnostic plots (see Figure 2-3) are created by plotting this fit (see lines 16 and 17 of script).

The sensitivity of the experimental variogram to removal of a single steel-cased well was investigated (see Figure 2-4 and Figure 2-6) using the Python script `remove_one_variogram_effects.py` (Section 8.2.2). This script loaded the well data (lines 4 through 35) and performed a least-squares fit of a linear surface (see Equation 1) through the data (e.g., see Menke (1984), Chapter 3), looping through the data to remove one steel-cased well at a time, re-computing the fit (lines 37 to 58). An ASCII text file was output (see line 24 for the filename) with summary statistics relating to each network-minus-one fit corresponding to the lines of the output file. This csv file was imported into MS-Excel, resulting in the plot of relative percent change in the slope and direction of the best-fit linear surface due to removing each steel-cased well, as shown in Figure 2-4 (see file `trend_surface_remove_one_results.xls` on the CD in the `report/figures/02_kriging` directory).

This same Python script also wrote a set of data files corresponding to the main dataset less a single steel-cased well for variogram analysis. These data files were imported into Surfer for experimental variogram plotting to create Figure 2-6; see the CD in directory `report/figures/02_kriging` for the data files and resulting Surfer file `perturbation_spread_of_variograms.srf` used to plot this figure. In the same directory on the CD, the Surfer file used to generate the final experimental variogram plot in Figure 2-5 can be found (`may2007_variogram_modela.srf`).

2.6.2. Kriging variance reduction calculation

The kriging variance minimization (see Section 2.4) consisted of a main Python script, `krig_plus_one.py` (Section 8.2.3), which drives the kriging process for different inputs and summarizes the outputs. This main script uses two subsidiary scripts `shared_data.py` (Section 8.2.4) and `kt3d_driver.bat` (Section 8.2.5) to perform its duties.

2.6.2.1. Kriging add one

The `krig_plus_one.py` script is explained here. Most of the first half of the file (lines 18 to 130) is the definition of the function `krig()`, which is called with arguments related to where to put an additional data point. Lines 28 through 53 are the input file for KT3D saved as a string having key parameters in the input file substituted with variables passed to the function (e.g., see pattern `%(varname)d` on lines 38 and 39). The data file used as input to KT3D is written on lines 56 to 61, potentially with an additional point appended to the end of the file. The DOS batch file `kt3d_driver.bat` is called to run `kt3d.exe` on lines 65 to 73. The kriging variance output created by running KT3D is read on lines 75 to 78, while certain subsets of the variance arrays are selected on lines 87 and 88. Lines 91 through 130 are located inside a threading `with` lock block, since they are writing summary results to a global variable (there are potentially more than one thread running at a time and the lock is to prevent two threads attempting to write at the same time and corrupting the data). Lines 95 through 115 are related to model-domain wide statistics, while lines 117 through 130 are related to the same statistics but only computed over the WIPP LWB.

The actual program flow begins at line 139, with the reading and preparation of various data from disk (lines 139 to 207). The last portion of the script (inside the `if __name__ == "__main__"` conditional on line 214) is only executed if the script is called from the command line. This portion is not executed if the script is imported (as is done in the `krig_remove_one_steel.py` and `krig_remove_two_steel.py` scripts). This final section sets up the arrays for saving the results (lines 216 to 218), calls `krig()` with the original unmodified dataset for comparison (line 220), and loops over all locations on a grid, adding one point at a time to the analysis (lines 229 through 242). Finally, the results of the entire analysis are saved to disk (lines 251 through 260 – these results are on the CD in the `analysis/kriging/kriging_add_well/output` directory); these matrices of results are used to plot the color figures in Section 2.4.1 using the MATLAB script `krig_add_one_plotting.m` one the CD in the `report/figure/02_kriging/` directory (since this MATLAB script was not used for analysis (only creation of color contour maps from ASCII data files), it is not listed in Section 8.2).

At lines 171 through 175, ASCII matrices are imported that indicate whether a given model cell is inside the active portion of the MODFLOW model domain. These matrices are written by the MATLAB script `generate_model_cell_masks.m` (Section 8.2.6), which uses the built-in function `inpolygon()` to determine which cells are inside the irregular polygon defining the MODFLOW active model area.

The `krig_plus_one.py` script is threaded because each call to KT3D takes roughly 10 to 15 seconds and there are tens of thousands of locations in the model domain where a point can be added; since the results of adding each point are done individually, the problem lends itself well to parallelization (i.e., a speedup of over four times using eight processors).

2.6.2.2. Kriging remove wells

As already stated, the scripts that run KT3D for the analysis of removing a well from the network import the majority of their functionality from the `krig_plus_one.py` script described in the previous section. The scripts for removing one (Section 8.2.7) and two wells (Section 8.2.8) are quite similar, and are described here. Each script reads in the relevant well and model domain data, the in the remove-one well case each steel well is individually removed from the network and the `krig()` function defined in the import `krig_plus_one.py` script are called to do run KT3D and summarize the results (lines 34 to 49). The results are written to ASCII files (see lines 44 through 49) for summarizing into Table 2-3, Table 2-4, Figure 2-14, and Figure 2-16. The source of the tables and bar-chart figures is saved in the spreadsheet `remove_one_well_results2_2010.xls` on the CD in the `analysis/kriging/kriging_remove_steel/` directory.

The map in Figure 2-15, showing the relative importance of removing steel-cased wells from the network with respect to the entire domain and the WIPP LWB, was created in Surfer from the tabular results. The Surfer file (`remove_one_steel_well.srf`) is included on the CD in the `report/figure/02_kriging/` directory.

In the remove-two-wells case, a list of four "most likely to be removed" steel-cased wells are used as the first well, then the remaining steel-cased wells are each additionally removed, similarly calling the imported `krig()` function to run kt3d and summarize the results (lines 38 to 68). Similarly, these results are written to files for summary in Table 2-5 and Table 2-6.

The source of the scatter plots in Figure 2-17 and Figure 2-18 and the tables of data is the spreadsheet `remove_two_well_results3.xls`, located on the CD in the `report/figure/02_kriging/` directory.

3.0 Local Gradient Estimation with Triangulation

The methodology used for local gradient estimation in the previous revision of this analysis report (McKenna, 2004) and in the associated follow-up paper (McKenna and Wahi, 2006) involved the use of "three-point estimators" to assess the ability to estimate head gradients in a 2D aquifer. The analysis presented here is instead in terms of a simpler approach using non-overlapping Delaunay triangles (a small subset of the triangles included in the three-point estimators).

Although three-point estimators have been used several places in the literature to estimate a regional gradient value from observed data; see e.g., (Cole and Silliman, 1996; Conwell et al., 1997; Silliman and Frost, 1998; Silliman and Mantz, 2000; McKenna and Wahi, 2006), few practicing hydrologists take this approach to estimating the gradients when presented with 2D head data. It is a more common approach to contour observed heads (i.e., potentials), estimating gradients from equipotential contours. While there are numerous techniques for creating contour maps from point measurements (e.g., kriging, inverse distance, splines), linear interpolation could be considered the most basic and easily understood approach. Often a geologist will sketch in the results of linear interpolation between data as a first step to hand contouring depth or thickness data, and then they will modify these results with their own professional judgment. In two dimensions, three points define both a triangle and a piecewise-constant estimate of the gradient across that triangle. A group of more than three points defines a network of triangles (bounded by their convex hull) and a piecewise-constant estimate of the gradient across the area inside the convex hull.

Linear interpolation is used for the local gradient-based estimation, since linear interpolation is a straightforward method that is easy to visualize and understand, and triangulation is readily implemented using available tools in the COTS software MATLAB (i.e., the built-in functions `delaunay()` and `voroni()`).

3.1. Delaunay Triangulation

In the three-point estimator approach of (McKenna, 2004), all possible combinations of three points were constructed into triangles to assess the quality of the network (with a fraction of the triangles discarded based on selection criteria). Many thousand overlapping triangles made visualization of results difficult (see Figure 3-1). For 30 wells, there are 4060 possible three-well combinations and for 40 wells there are 9880 possible combinations. In the current approach, the much smaller subset of non-overlapping triangles produced by Delaunay triangulation is used.

Since the triangles will not overlap, the gradients estimated with this technique are as local as possible with the given set of points. When using overlapping triangles, the selection of one gradient estimate over another (when two triangles cover the same area) may become complex, or some sort of averaging must be done to produce a useful result.

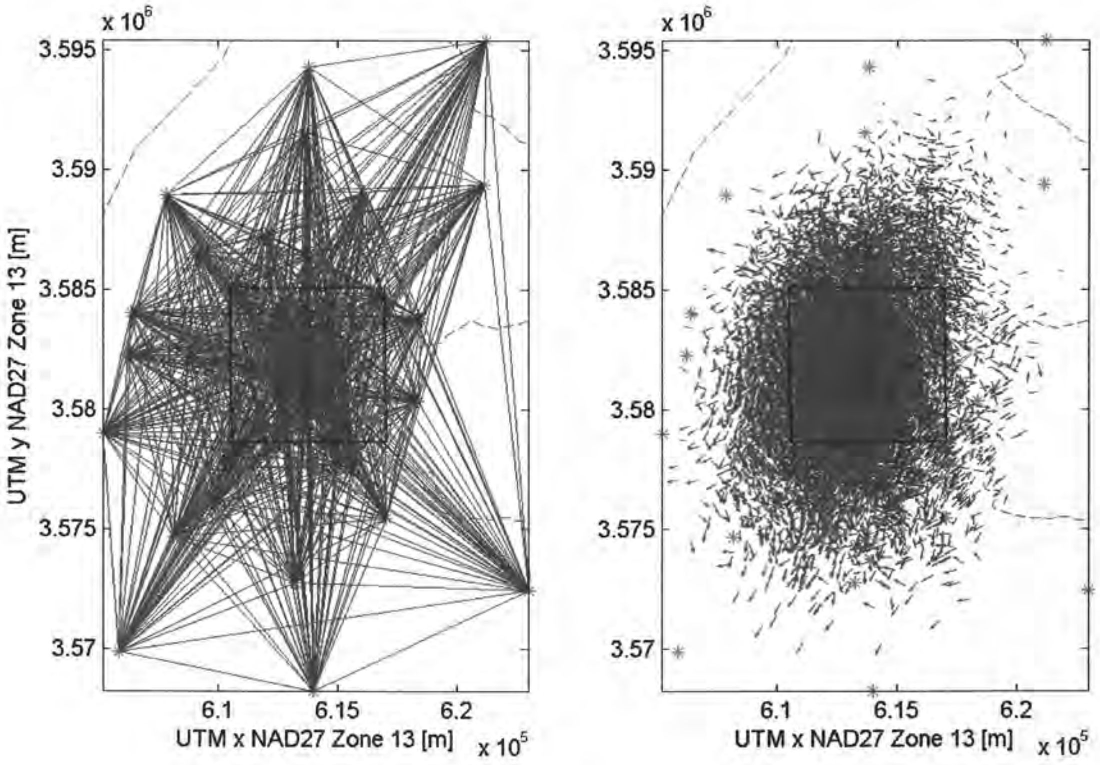


Figure 3-1. All possible triangles (left) and corresponding gradient vectors (right) for May 2007 Culebra monitoring network (as was used in the three-point estimator approach from first revision of this report). Vectors are $\log_{10}$ length scaled; tails of vectors are anchored at the center of their triangle.

Given a 2D set of data points, Delaunay triangulation produces a set of triangles, where each triangle bounds a point and its natural neighbors (see Figure 3-2a). Delaunay triangles are directly related to Voronoi polygons, which are the unique polygons circumscribing the area closer to a given observation well than any other well (see Figure 3-2b).

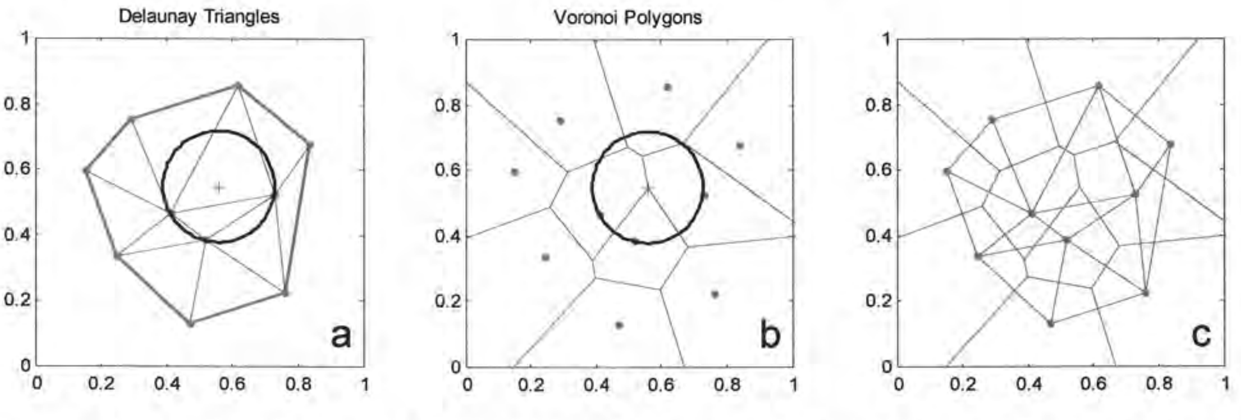


Figure 3-2. Delaunay triangles and Voronoi polygons for 10 randomly located points (red symbols). Black circle with green center used to illustrate the relationship between triangles (a) and polygons (b). Red lines indicate the convex hull, (c) shows both sets of polygons together.

Some properties of these unique triangles and polygons are:

- Delaunay triangles uniquely tessellate the area within a convex hull enclosing the data (except in certain symmetric cases);
- Voronoi polygons fill the entire plane; the polygons corresponding to the data on the convex hull have infinite area;
- Vertices of Voronoi polygons correspond to the centers of circles that uniquely go through the three neighboring points (see Figure 3-2b);
- In a square grid of points, Delaunay triangles become right isosceles triangles (two equal angles and sides) and Voronoi polygons become squares (see Figure 3-3).
- In a triangular grid of points, Delaunay triangles become equilateral and Voronoi polygons become regular hexagons (see Figure 3-4).

Three points are the minimum required to estimate direction and magnitude of a gradient from 2D point observations; Delaunay triangles therefore define piecewise-constant gradient over the area enclosed by the convex hull surrounding all points. Delaunay triangles, when assigned z values at the vertices (i.e., heads), become a triangular irregular network (TIN); these are often used in engineering to approximate irregular surfaces.

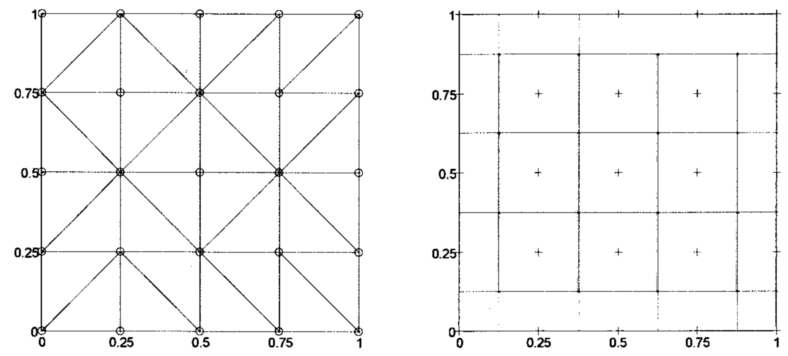


Figure 3-3. Delaunay triangles and Voronoi polygons for symmetric square grid; note ambiguity in triangles

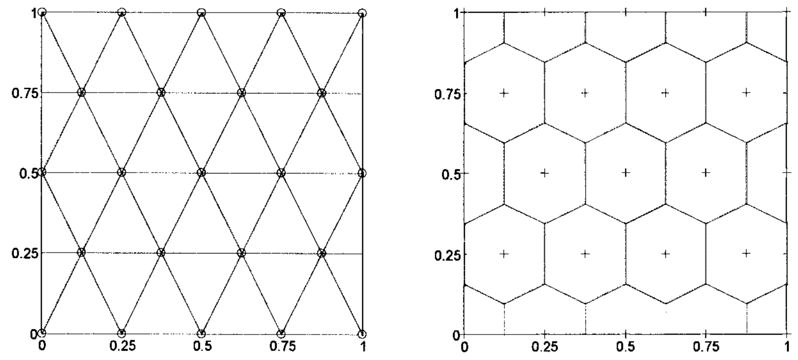


Figure 3-4. Delaunay triangles and Voronoi polygons for symmetric triangular mesh

The regular grids of points in Figure 3-3 and Figure 3-4 illustrate the shapes of triangles that arise under these ideal conditions (compared to the random arrangement of points in Figure 3-2, and seen in the following Culebra monitoring network analysis).

Voronoi polygons are not used in this analysis, but they are included in this introductory discussion because it is clear that they are unique for a given set of points and there is a unique mapping from Voronoi polygons to Delaunay triangles, therefore it is illustrated how the Delaunay triangles are also unique. The non-unique case corresponds to the extreme symmetry shown in Figure 3-3; squares can equivalently be cut into triangles along either diagonal. This will not affect the results of this analysis, since the Culebra monitoring wells are not located on a symmetric rectangular grid.

3.2. Triangle Shape Metric

To rank the quality of the shape of triangles, the ratio of the minimum and maximum of the interior angles is assessed; this value is believed to capture the quality of a triangle for the purposes of gradient estimation from three data points. The lengths of the sides of the triangles can be related to the size of the angles through the law of sines,

$$\frac{a}{b}=\frac{\sin(A)}{\sin(B)}, \tag{9}$$

where a and b are the shortest and longest sides of the triangle (i.e., the min/max length ratio), and A and B are the corresponding largest and smallest angles of the triangle – angle A opens up to side a (i.e., the ratio of the sines of the min/max angles).

In the case illustrated in Figure 3-3, the triangles have angle ratios of 0.5 (one 90° and two 45° degree angles). Figure 3-4 illustrates triangles with an angle ratio of 1 (three 60° angles); this is the maximum ratio. Using the angle ratio as the metric, therefore the "best" triangle is an isosceles one. Likewise, triangles with one dimension or angle much smaller than the others will have a very small angle ratio, approaching zero in the limit as the three points become collinear. Triangles with large aspect ratios (proportional to the inverse of the angle ratio) tend to produce worse estimates of the gradient, based on an assumed unbiased normal distribution of errors associated with observing heads in a well (McKenna and Wahi, 2006).

Figure 3-5 shows the distribution of triangle size, interior angle ratio and the magnitude of the gradient computed from observed May 2007 freshwater heads. In Figure 3-5a, the logarithm of area is used to color-code the triangles that make up the 2007 Culebra monitoring network. Some very elongate triangles have small areas, considering how distant the wells are that make up their corners (e.g., the blue triangle along the west-central edge of the area, comprising wells WIPP-25, IMC-461, and SNL-16). Figure 3-5b shows the distribution of the angle ratio, for the 2007 Culebra network; the dark red triangles are nearly isosceles, while the dark blue triangles have one large obtuse angle. Figure 3-5c shows the logarithm of the head gradient magnitude, computed from the three corner wells. Aside from the two anomalously high gradient areas associated with SNL-6 and SNL-15 (east-central and north-east areas), there is an east-west yellow band across the middle of the model area, with blues north and south of it, representing the observed higher freshwater head gradient across the center of the LWB (e.g., see Figure 2-1).

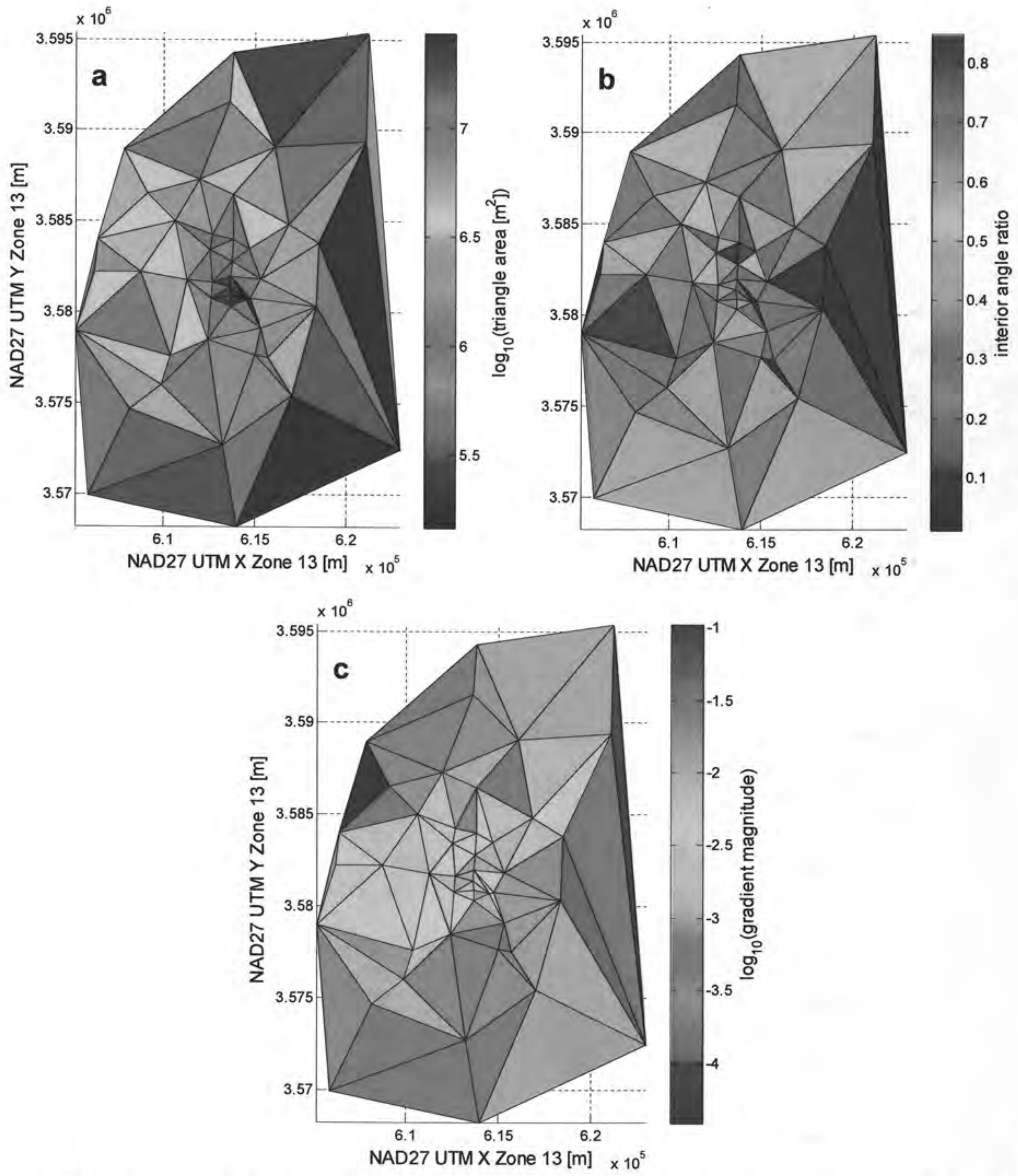


Figure 3-5. Distribution of geometry metrics for 2007 Culebra well network: (a) area of triangle, (b) angle ratio, and (c) May 2007 freshwater head gradient magnitude.

Figure 3-6 shows scatter plots of the quantities represented spatially in Figure 3-5 (each dot represents a triangle); these illustrate that there is essentially no correlation (positive or negative) between the triangle angle ratio and area (a), or the angle ratio and the magnitude of the gradient (b). This is because angle ratio represents the triangle shape, while shape and size are two unrelated quantities (in this case). Additionally, the gradient is a function of the head observed at the wells, while the angle ratio is not affected by observed head. Figure 3-6c indicates a possible, but very weak, negative correlation between the size of triangles and the gradient magnitude. There is a greater density of wells within the WIPP LWB, where steeper gradients are observed; this trend is corrupted by the anomalous steep gradients associated with large triangles containing SNL-6 and SNL-15.

Based on this heuristic analysis, the angle ratio is thought to be an adequate primary metric for triangle quality. Triangle size is considered to be independent information, but it is not directly correlated with a desired monitoring network objective. While smaller triangles resolve more detail than larger ones, a dense network is much more expensive and large triangles are allowable in the portions of the domain further from the WIPP land withdrawal boundary. This metric obviously only considers the network geometry; there may be important hydrologic or geologic information to be gained from locating a well at locations which may be sub-optimal solely from a geometric point of view.

Freshwater head gradient direction and magnitude are illustrated in Figure 3-7 using vectors scaled to the gradient magnitude. Figure 3-7a shows the network for the 2007 Culebra monitoring network, while Figure 3-7b shows the remaining network after leaving out SNL-6 and SNL-15, which are non-representative of heads west of the composite H2/M2 – H3/M3 Rustler halite margins (Johnson, 2009). Leaving out these two wells removes the spurious large gradients around these wells, but also changes the overall shape of the network on the eastern third of the domain (see Figure 3-7).

Figure 3-8 and Figure 3-9 show the same quantities in Figure 3-5 and Figure 3-6 for the Delaunay triangles that correspond to the existing network without SNL-6 and SNL-15. Most of the triangles in the domain are unaffected by leaving these wells out, since only 10 triangles include either of these points in the existing network. Apparent changes elsewhere in the domain are due to rescaling of the color gradient in the figures, because the minimum or maximum values are linked to triangles changed by leaving out these two wells. The steeper gradient across the WIPP LWB is more evident in Figure 3-8c (due to color scaling). The negative correlation between area and gradient is also clearer in Figure 3-9c, as most of the large triangles with steep gradients were connected to the low values in either SNL-6 or SNL-15.

This section introduces the triangle interior angle ratio as a continuous metric that identifies isosceles-like triangles and is not spuriously correlated to triangle size or observed gradient between head observations at wells. Based on the comparison with and without SNL-6 and SNL-15, these wells are left out of any analysis that requires head values (i.e., the remove-one-well analysis)

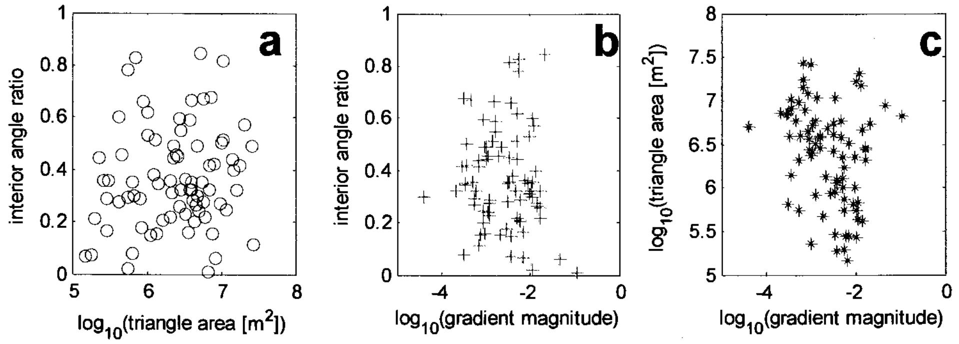


Figure 3-6. Scatter plots of relationships between different triangle metrics for 2007 Culebra well network.

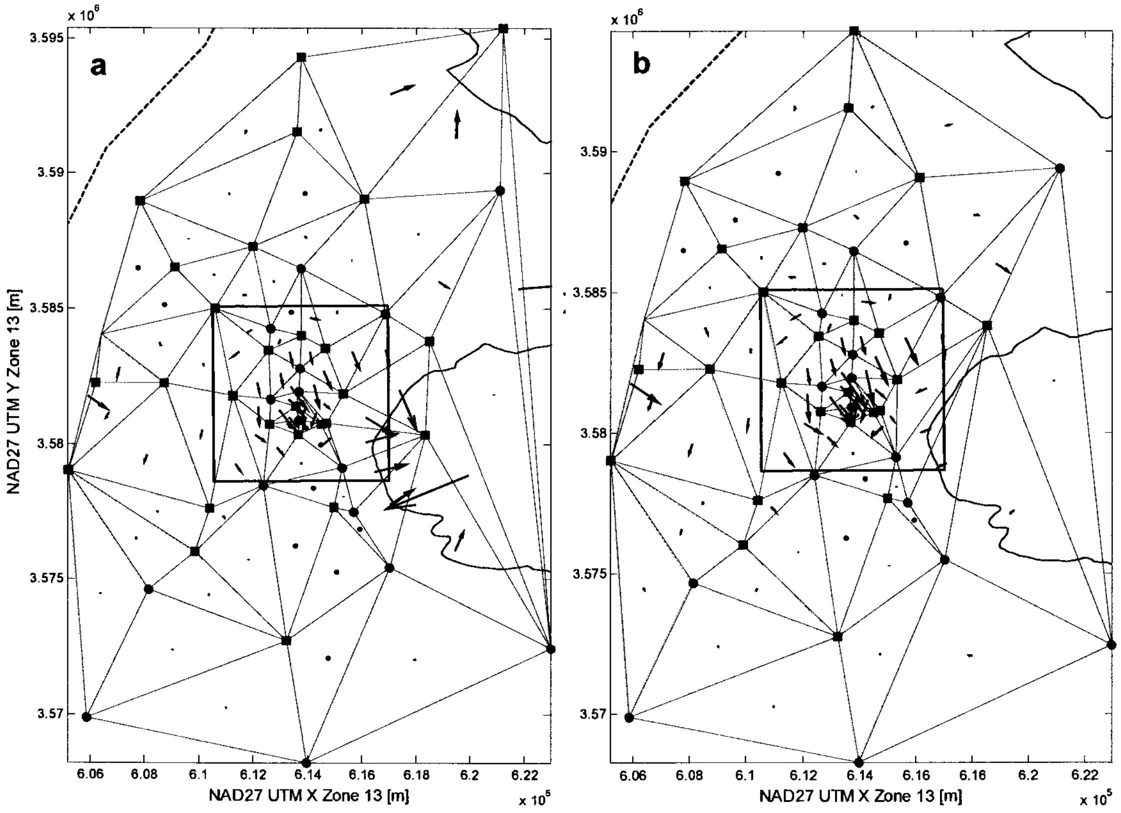
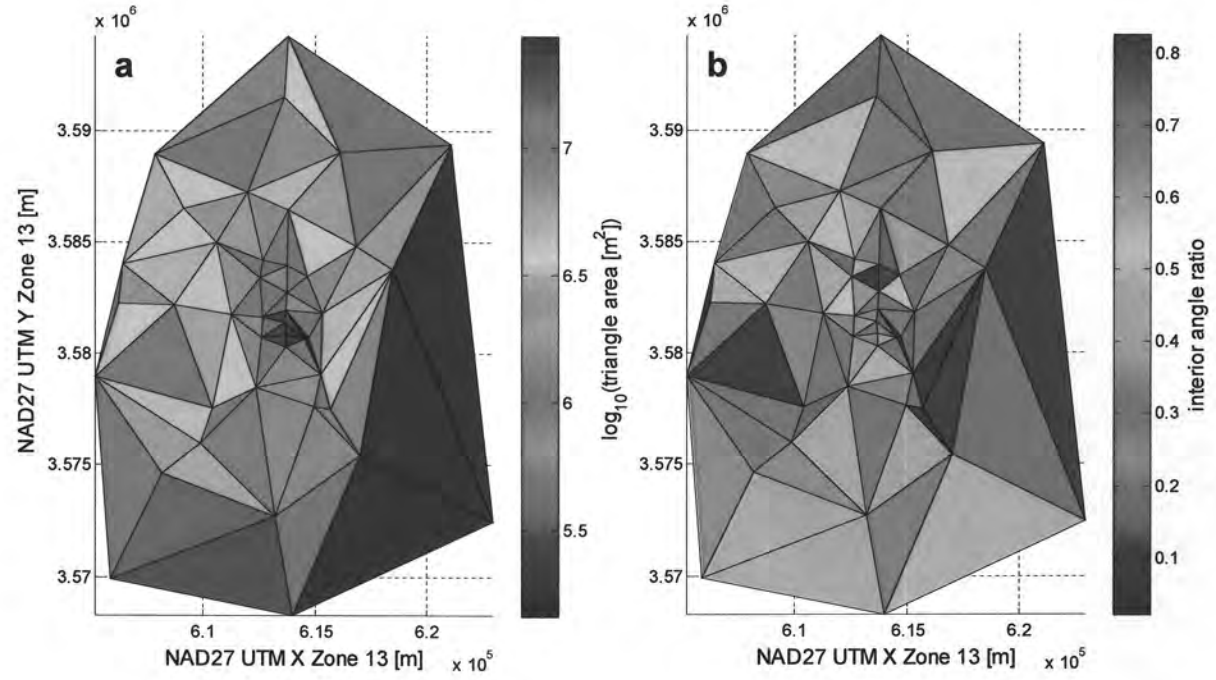


Figure 3-7. Delaunay triangles for 2007 Culebra network (a) with and (b) without SNL-6 and SNL-15. Gradient plotted as arrows (tails starting at center of triangle); length of arrow is proportional to magnitude gradient, arrow orientation indicates groundwater flow direction. WIPP LWB (black solid), M2/H2 - M3/H3 composite Rustler halite margins (magenta), and no-flow (black dashed) boundaries shown.



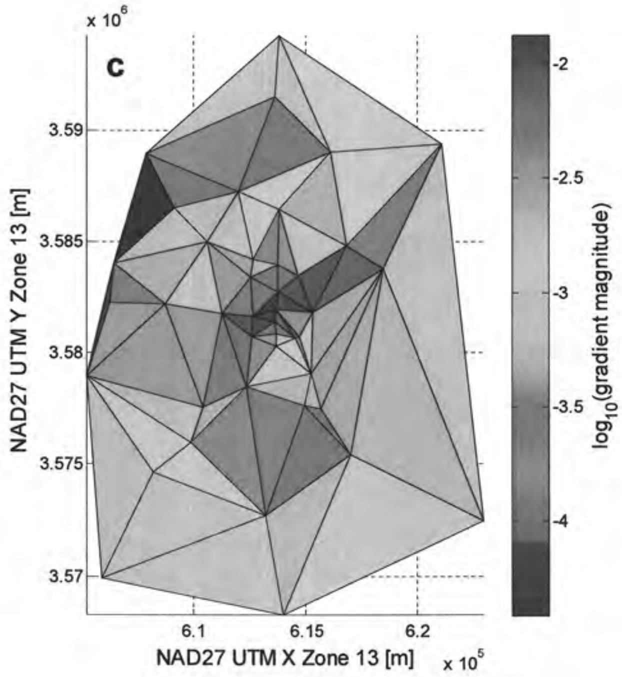


Figure 3-8. Distribution of triangle geometry metrics: (a)triangle size, (b) interior angle ratio, and (c) May 2007 freshwater head gradient magnitude for 2007 network without SNL-6 and SNL-15

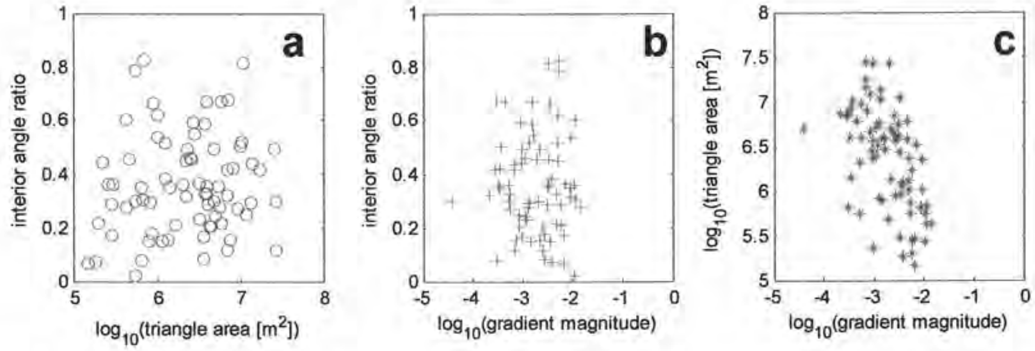


Figure 3-9. Scatter plots of relationships between different triangle metrics for 2007 network without SNL-6 and SNL-15.

3.3. *Add One New Well*

Similar to Section 2.4.1, we explore the effects of adding one more monitoring well to the network, but using the angle ratio metric discussed in the previous section. For each model cell in the Culebra MODFLOW model grid, a monitoring point is added, and the triangulation process is repeated. Statistics regarding the resulting triangular network are summarized in the following plots.

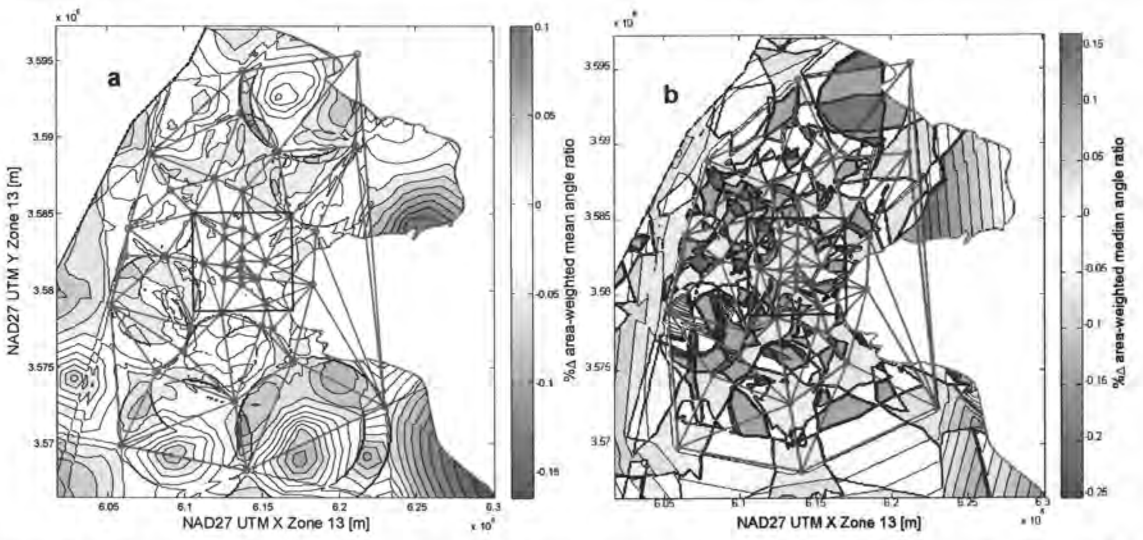

a
b
NAD27 UTM Y Zone 13 [m]
NAD27 UTM X Zone 13 [m]
%Δ area-weighted mean angle ratio
%Δ area-weighted median angle ratio

Figure 3-10. Increase (red) or decrease (blue) in area-weighted (a) median and (b) mean angle ratio for triangle network, due to one additional well.

The addition of an observation point can only be judged using geometry metrics, because the head that would be observed at the new location is yet unknown. Figure 3-10 shows the 2007 Culebra monitoring network (red circles are well locations, green lines are the Delaunay triangles for the 2007 Culebra well network). Contour colors indicate whether adding a well at that location and re-triangulating the network (not shown) would increase or decrease the interior angle ratio, averaged over the model domain.

The mean and median angle ratios shown in Figure 3-10 are weighted by triangle area. Each triangle's angle ratio is multiplied by its area, and then the mean or median of these products (for all the triangles in the network) is divided by the total area covered by the network. The total area inside the convex hull surrounding the whole network may increase if the proposed monitoring point is located outside the convex hull of the current network.

Areas that are blue in Figure 3-10 indicate locations where a new well would create additional large elongate triangles in the Delaunay network. Triangles with low angle ratios make poor estimators of the local gradient. Areas between wells in the southern and eastern parts of the domain show the largest relative increase (red) in both the mean and median triangle shape metric averaged across the model domain. The north-central portion of the domain shows a large positive increase in the median triangle shape metric, but not in the mean (the median is less sensitive to large changes in a single triangle, e.g., along the southern edge of the network). An additional monitoring point at a red location in Figure 3-10a or b would be the best in terms of the relative geometry of the resulting network. These locations change large elongate triangles into smaller triangles with three similar angles; smaller, more symmetric triangles are better for estimating local gradients, given the same relation between the observed gradient and measurement error.

McKenna and Wahi (2006) (and likewise the 2004 version of this analysis) performed statistical analyses of three-point estimators to evaluate their ability to estimate the gradient from three point measurements, as a function of the relative head measurement error (RHME), the orientation of the principle groundwater flow direction, and triangle shape. This analysis only takes the triangle shape or size into account. Triangles that are small and symmetric, but which cover an area of very low gradient magnitude may be bad estimators as well, given the current distribution of heads. This analysis only considers the geometry of the network.

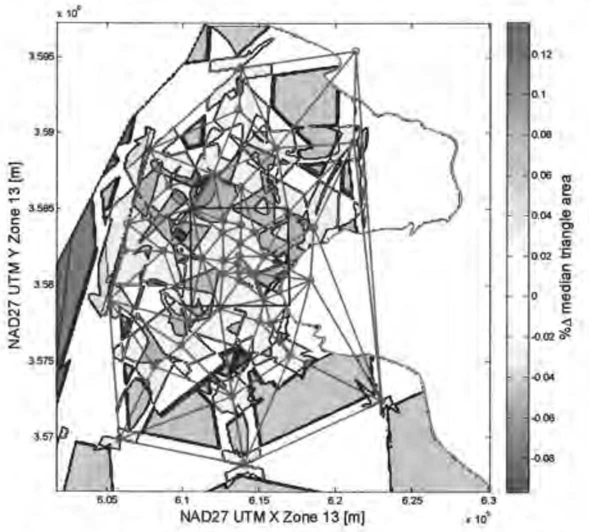


Figure 3-11. Increase (red) or decrease (blue) in median triangle area, due to including one additional well

In Figure 3-11, the relative change in the median triangle size is shown for the same scenario of adding one additional well to the network. Although triangle size is not the main metric, it shows different information from the angle ratio plot. Since adding a monitoring point to the network will always create more triangles, but the total network area will only change if the additional well is outside the original convex hull; Figure 3-11 shows whether the additional location will make nearby triangles smaller or larger. Most of the regions within and near the WIPP LWB are blue, indicating the triangles in the proposed network will on average become smaller, while new wells located at the extremities of the existing network will increase the median triangle size, especially the area along the southern end of the no-flow boundary (see bright red area in west-central portion of Figure 3-11).

Following common sense, Figure 3-11 shows that adding monitoring locations near the edges of the domain will add more large triangles to the existing network. Adding a point near the middle of the domain will instead add more small triangles to the existing network, since the density of wells inside the WIPP LWB is already high. This does not consider the fact that expanding the overall size of the monitoring network would likely add useful information, regardless of the network shape.

3.4. Remove One Steel Well

Compared to the addition of a new well, more can be said about the removal of a well from the network, since heads have been observed at the location proposed for removal. In this section, the effect of removing a well is computed by first copying the results of triangulation (which assigns a piecewise-constant gradient value to every point inside the convex hull) onto points corresponding to the centers of the finite-difference cells of the MODFLOW model grid. Only points in the MODFLOW grid which fall within the convex hull are compared.

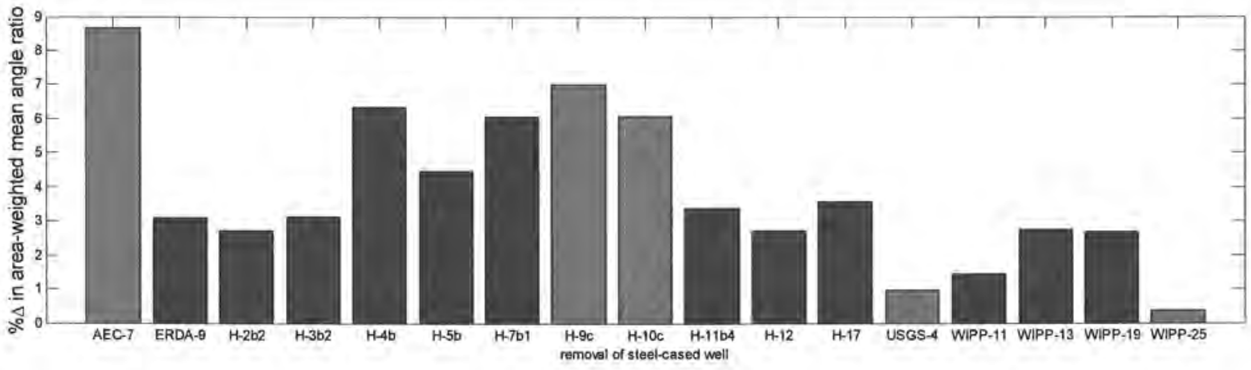


Figure 3-12. Change in mean area-weighted angle ratio (averaged over model domain) upon removal of each steel well from monitoring network (no SNL-6 or SNL-15). Red bars are wells located on convex hull.

In Figure 3-12, AEC-7 has the largest impact on the mean area-weighted angle ratio metric. Excluding the steel cased wells which form the boundary of the triangulation (shown in red) leaves H-4b, H-7b1, and H-5b with the largest impact, with wells USGS-4 and WIPP-25 having the smallest impact, even though they makeup part of the convex hull.

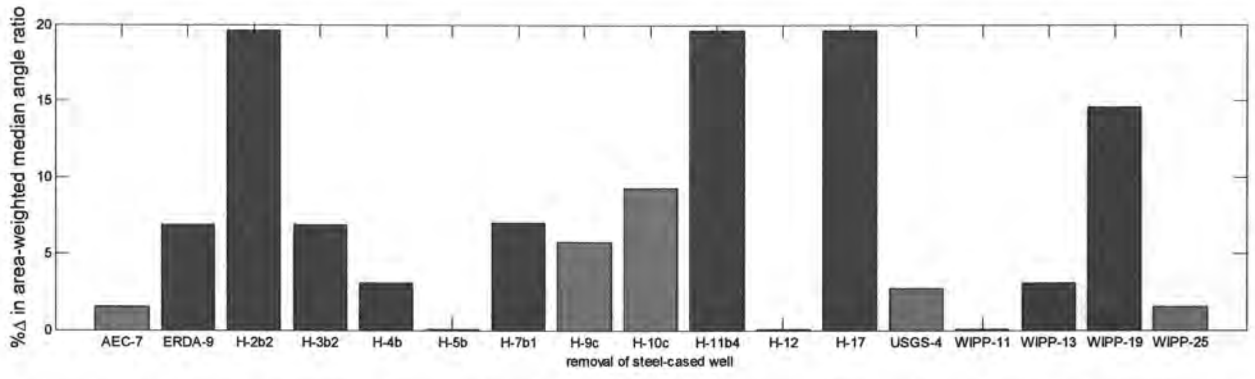


Figure 3-13. Change in median area-weighted angle ratio upon removal of each steel well from monitoring network (no SNL-6 or SNL-15).

In Figure 3-13 H-2b2, H-11b4 and H-17has the approximately the same impact on the median area-weighted angle ratio, but wells H-2b2, H-11b4 and WIPP-19 now also have large percent change values (compared to Figure 3-12). These two bar charts (Figure 3-12 and Figure 3-13) correspond to Figure 3-10 parts a and b for the case of removing one well to the network.

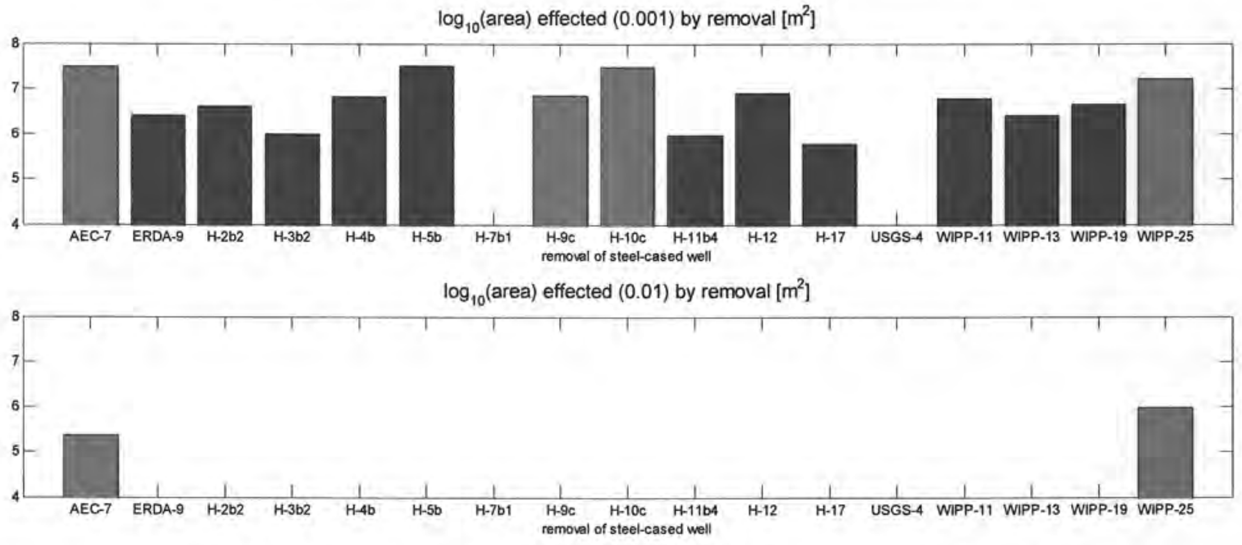


Figure 3-14. Area affected (Δ gradient magnitude ≥ 0.01) by removal of steel well

Figure 3-14 shows the area affected by the predicted change in gradient between the 2007 monitoring network and the reduced network with the corresponding steel-cased well removed. The changed area is defined as the area where the relative change in gradient magnitude is greater than or equal to 0.01 (lower figure) or 0.001 (upper figure). After removing each well, the Delaunay triangulation is recomputed, and the observed gradient is computed for each resulting triangle. The marked difference between the two bar charts in Figure 3-14 indicates that although there is a very large amount of area that would be slightly affected by removing any one steel well (large number of bars in the upper figure), there is very little of the model domain that would be significantly affected by any one well being removed (bottom figure).

Table 3-1. Ranking of steel-cased wells based on triangle gradient estimators. A low numerical rank indicates importance.

	%Δ mean angle ratio		%Δ median angle ratio		average rank
H-17	3.58	7	19.61	1	4
H-10c	6.09	4	9.27	5	4.5
H-11b4	3.38	8	19.61	1	4.5
H-7b1	6.07	5	7.04	6	5.5
H-9c	7.02	2	5.75	9	5.5
H-2b2	2.73	12	19.61	1	6.5
H-4b	6.34	3	3.09	10	6.5
AEC-7	8.65	1	1.51	14	7.5
H-3b2	3.13	9	6.87	7	8
ERDA-9	3.09	10	6.87	7	8.5
WIPP-19	2.70	14	14.61	4	9
H-5b	4.47	6	0.00	16	11
WIPP-13	2.75	11	3.08	11	11
USGS-4	0.96	16	2.74	12	14
H-12	2.72	13	0.00	17	15
WIPP-11	1.44	15	0.00	15	15
WIPP-25	0.36	17	1.51	13	15

These bars represent the areas colored in the figures in the figures in Section 9.0. The individual figures in Section 9.0 show the localized effects of removing a well from the network; the colored areas only immediately surround the well being removed. The effects due to removing wells in areas with small triangles (e.g., inside and near the WIPP LWB) will obviously only propagate out to a small area. Wells that are part of large triangles along the periphery of the domain will affect larger areas when removed.

3.5. Remove Two Steel Wells

Using the same list of "probable" wells from section 2.4.3 (kriging variance reduction), the local gradient estimator analysis of the previous section can be repeated for each of the networks with one of the steel-cased wells already removed. Table 3-2 shows the cumulative effect that removing two steel-cased wells has on the domain-average mean angle ratio (see Figure 3-12 for the corresponding single-well analysis). The percent changes (illustrated in the color image) show that removing any pair of wells including H-10c (row 9) or most wells in a pair with well H-7b1 (row 7, column 4) lead to improvements in the geometric layout of the observation wells, because these wells are involved in several large elongate triangles. These types of improvements are not the goal of this analysis. Well H-9c (row 8) shows the largest decrease in the domain-average angle ratio metric, corresponding to the worst effects on the well network; this well is on the southern edge of the network.

Table 3-3 shows how median triangle size in the network increases (red) or decreases (blue) as pairs of steel-cased wells are removed from the network. Removing other steel-cased wells in conjunction with H-10c (row 9) would decrease average triangle size because the convex hull becomes smaller upon removal of this well.

Table 3-2. Change in domain-average mean angle ratio upon removal of 2 steel-cased wells. Image illustrates percentages given in table, in same row/column order.

		WIPP-25		WIPP-13		H-12		H-7b1	
1	AEC-7	3.7%	1	5.7%	1	5.7%	1	9.5%	1
2	ERDA-9	-1.4%	9	0.3%	9	0.2%	8	3.5%	8
3	H-2b2	-1.7%	11	-0.1%	10	-0.1%	10	3.2%	10
4	H-3b2	-1.4%	8	0.3%	8	0.3%	7	3.5%	7
5	H-4b	1.8%	2	3.4%	2	3.4%	2	6.7%	3
6	H-5b	-0.1%	5	1.6%	5	1.6%	4	4.8%	4
7	H-7b1	1.5%	3	3.2%	4	3.2%	3		
8	H-9c	-10.1%	16	-8.3%	16	-4.1%	16	-4.8%	16
9	H-10c	1.1%	4	3.2%	3	0.4%	6	7.3%	2
10	H-11b4	-1.1%	7	0.6%	7	0.5%	5	3.8%	6
11	H-12	-1.8%	12	-0.1%	11			3.2%	11
12	H-17	-0.9%	6	0.8%	6	-1.2%	12	4.0%	5
13	USGS-4	-3.7%	15	-1.8%	15	-1.8%	15	-3.9%	15
14	WIPP-11	-3.0%	14	-0.8%	13	-1.4%	13	1.9%	13
15	WIPP-13	-1.7%	10			-0.1%	9	3.2%	9
16	WIPP-19	-1.8%	13	-0.1%	12	-0.1%	11	3.1%	12
17	WIPP-25			-1.7%	14	-1.8%	14	1.5%	14

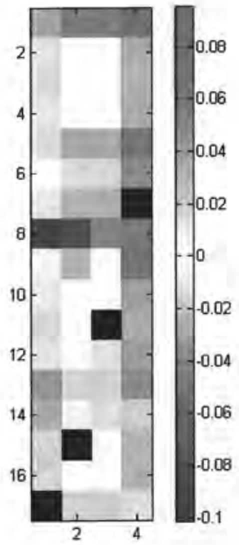
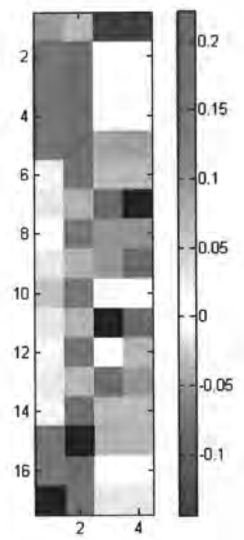


Table 3-3. Change in domain-average median triangle size upon removal of 2 steel-cased wells. Image illustrates percentages given in table, in same row/column order.

		WIPP-25		WIPP-13		H-12		H-7b1	
1	AEC-7	-5.2%	16	-3.4%	16	-14.4%	16	-14.4%	16
2	ERDA-9	15.2%	3	21.9%	2	0.0%	3	0.0%	3
3	H-2b2	15.2%	3	21.9%	2	0.0%	3	0.0%	3
4	H-3b2	15.2%	3	21.9%	2	0.0%	3	0.0%	3
5	H-4b	16.5%	1	22.1%	1	8.5%	1	8.5%	1
6	H-5b	1.7%	8	16.1%	8	-3.4%	10	-3.4%	9
7	H-7b1	-1.7%	12	6.7%	12	-8.3%	14		
8	H-9c	0.0%	11	15.2%	11	-5.2%	12	-5.2%	12
9	H-10c	-1.7%	12	6.7%	12	-5.2%	12	-8.3%	14
10	H-11b4	6.1%	7	15.9%	9	0.0%	3	0.0%	3
11	H-12	-1.7%	12	6.7%	12			-8.3%	14
12	H-17	1.7%	8	15.9%	9	0.0%	3	-3.4%	9
13	USGS-4	-1.7%	12	6.7%	12	-8.3%	14	-5.2%	12
14	WIPP-11	1.7%	8	21.3%	6	-3.4%	10	-3.4%	9
15	WIPP-13	16.5%	1			6.7%	2	6.7%	2
16	WIPP-19	15.2%	3	21.9%	2	0.0%	3	0.0%	3
17	WIPP-25			16.5%	7	-1.7%	9	-1.7%	8



3.6. *Local Gradient Estimator Summary*

The local gradient estimator analysis presented in this section considers the effects of adding a well and removing a well or two, from the perspective of the network and head gradient estimation geometry. The metric which was used to compare potential triangular networks was mainly the ratio of the maximum and minimum angles, with the median triangle size used as a secondary metric.

3.7. Local Gradient Estimator Run Control Summary

The local gradient estimator analysis performed in this section is described here in terms of files, programs, and scripts used. The required files are on the CD and are described in sufficient detail to allow recreation of the results given in the text. All the analysis in this section was done using MATLAB, and the calculation and plotting of results are partially mixed together in the scripts.

3.7.1. Triangles: add a well

The MATLAB script `triangles_add_one.m` (Section 8.3.1) is the main script that performs the calculations for the evaluation of additional locations in terms of Delaunay triangles. This section describes the script's basic behavior. The first lines of this script load in the required data from files (lines 1 to 30). A rectangular array of *x* and *y* locations (UTM NAD27 Zone 13 [m]) are created using `meshgrid()` (lines 32 through 34), which is then compared to the polygon defining the active model domain using `inpolygon()` (line 36), to determine the cells that are inside the active MODFLOW domain. These matrices are then unwrapped into vectors to simplify indexing (line 40).

The main loop of this script (lines 45 through 111) goes over each potential new location (plus one for the base case with no additional monitoring locations), re-triangulating the network (line 59). The results of `delaunay()`is a matrix with three columns corresponding to the three vertices of each triangle, and a row for each triangle. The values in this matrix are integer indices pointing to the values of the *x* and *y* coordinates passed to `delaunay()`. For example, if `tri=delaunay(x,y)`, where `x` and `y` are each a vector of 3 locations, `tri` will be a 1×3 matrix, where the corners of the triangle specified by the first (and only) row of `tri` are obtained addressed like `x(tri(1,[1,2,3])), y(tri(1,[1,2,3]))`. The `geom` matrix stores the results of the geometric calculations for each triangle in the network; rows 1-3 are the lengths of the sides (computed using the Pythagorean theorem – lines 70 to 77), rows 4-6 are the angles between the sides (computed using the cosine law – lines 80 to 87), and row 7 is the area of the triangle (computed using the built-in MATLAB function `polyarea()` – line 90). The interior angle ratio is computed from the maximum and minimum interior angles (line 93). Some summary statistics regarding the entire triangle network are saved into the matrix `Q`; the area-weighted angle ratio average and median are computed, as wells as the average and median triangle size are computed (lines 95 to 109).

After looping over all possible locations, the matrix `Q` contains different average results for each point in the domain that is inside the active MODFLOW flow domain. The results for the existing Culebra network with (Figure 3-5, Figure 3-6, and Figure 3-7) and without SNL-6 and SNL-15 (Figure 3-7, Figure 3-8, and Figure 3-9) were plotted from the `geom` matrix, for the case with no additional wells.

These summarizing results (`Q` matrix) are saved to ASCII file (lines 116 to 123 – see files in `analysis\triangle_metric\output\` directory on CD) and plotted to make color contour maps shown in Figure 3-10, and Figure 3-11.

The MATLAB script `redwhitemap.m` (see Section 8.3.2) is used by the `triangles_add_one.m` script just described, to create a color legend corresponding to blue being negative, red being positive and white being zero, based on a vector of data passed. This script only is used for creating a colormap for plotting figures in MATLAB, but is included here for completeness.

3.7.2. Triangles: removal steel-cased wells

The `triangles_remove_one.m` MATLAB script (Section 8.3.3) does much of the same that the `triangles_add_one.m` script in the previous section did, but it also computes things related to the freshwater head gradient across triangles between wells.

Similar to the previous triangle metric script, the first portion of the script loads data from file (lines 8 to 34), but here a series of nested for loops are used to find the wells on the convex hull (for marking them in the bar chart figures – lines 38 to 46). The main loop of the script re-computes the metrics related to the triangle, removing a different steel-cased well each time through the loop. In addition to geometry metrics related to the triangles (`geom` and `Q` matrices, line 144 through 176), the gradient defined by the freshwater head observed at the three corners of the triangles is also computed using Cramer's rule and saved into the `coeff` matrix (lines 98 to 122).

The gradient estimates are individual values for each triangle in the network (piecewise constant), but to compare the effects of removing a well from the network, which will result in a different network, the values are copied onto a 100 m square grid at each step (lines 125 to 136). This is done by cycling through the triangles in the network (typically about 30 or 40 triangles), each time selecting the cells from the 100 m square grid that are inside the triangle (using `inpolygon()`), assigning the gradient from the triangle to all the cells that fall inside it. The rest of the script is used to plot figures for the analysis report (Figure 3-12, Figure 3-13, and Figure 3-14, and the figures in Section 9.0), using the data computed in the main loop.

The `triangles_remove_two.m` MATLAB script does essentially the same thing as the remove-one script, but takes a list of four "likely to be P&Aed" wells, removing each of these first, then doing the remove-one-well process outlined above. The matrices resulting from this script are made into Table 3-2 and Table 3-3.

4.0 Model Correlation Analysis

In addition to the variance reduction and local gradient estimator approaches to monitoring network design, a third approach is used here to incorporate uncertainty captured in the performance assessment (PA) simulation into the monitoring network design. These calculations also incorporate recent updates in the geologic conceptual model and the influence of these updates on the spatial distribution of transmissivity within the Culebra. These recent updates in the geologic conceptual model have been used to produce the base transmissivity fields used in this study and are summarized in the Culebra T-fields summary report (Kuhlman, 2010b).

4.1. Background

The goal of this portion of the report is to include a third independent metric in the overall optimization that specifically addresses the PA monitoring network design goal of providing head and aquifer transmissivity data for defensible calibration of PA models. Additionally, the approach developed here specifically incorporates PA information in the form of groundwater travel times from the repository area to the boundaries of the WIPP LWB. This approach makes use of the existing ensemble of calibrated transmissivity fields (Hart et al., 2009) such that no additional groundwater flow and/or transport modeling is necessary.

4.2. Calculating Sensitivity Coefficients

The sensitivity of model outputs to changes in model inputs arises in the calibration, uncertainty analysis, and cost optimization of both analytic and numerical models. A model can range in complexity from a linear analytic expression to a complex numerical model. In general, a sensitivity coefficient, *S*, is calculated as the partial derivative of a model output with respect to each model input parameter:

$$S_{ij} = \frac{\partial O_i}{\partial P_j} \tag{10}$$

where S_{ij} is the sensitivity coefficient of the model prediction, *O*, at the i^{th} observation point to the j^{th} model parameter, P_j. S_{ij} is an $n \times m$ matrix (i.e., the Jacobian matrix) with the number of rows equal to the number of model parameters (n) and the number of columns equal to the number of observations (m) (Zheng and Bennett, 2002). S_{ij} is often given in a normalized or dimensionless form, through appropriate scaling factors; this matrix often plays a key role in parameter estimation techniques such as in the conjugate gradient, Newton iteration, or Levenberg-Marquard algorithms.

4.2.1. Sensitivity Equation Method

The expression governing the process which controls how parameters (P_j) are related to outputs (O_i) can sometimes be differentiated using calculus. This method is usually only applicable to simple lumped-parameter or analytic equation models. Although this approach is quite problem-specific, it leads to closed-form expressions for the sensitivity matrix. The form of the sensitivity equations often provides insight to the underlying process without needing to evaluate the problem for specific parameter values. Because the PA model considered here is an ensemble of calibrated MODFLOW models with irregular distribution of parameters and boundary conditions, this analytic sensitivity equation approach is infeasible.

4.2.2. Perturbation Approach

The derivative in Equation (10) can be approximated using finite differences. A small perturbation is made in a single model input (ΔP_j), leading to a set of perturbed model outputs which are differenced with model predictions from a base case $O_i(P_j)$, and normalized by the change in the parameter.

$$S_{ij} = \frac{\partial O_i}{\partial P_j} \approx \frac{O_i(P_j + \Delta P_j) - O_i(P_j)}{\Delta P_j} \tag{11}$$

This is the most generally-applicable and widely-used approach to estimating sensitivity coefficients; for example, this is the approach taken in inverse-modeling codes such as PEST (Doherty, 2002).

If a model has n parameters for which sensitivity information is desired, then at least n+1 model runs must be performed to compute the one-sided finite difference given in Equation (11). For higher-order accuracy, often 2n+1 model runs can be used to estimate derivatives via centered finite differences. For large highly-parameterized models (i.e., thousands of parameters or more), the perturbation approach often leads to unmanageably large computing demands. For the inverse problem, there are many approaches for either reducing the number of parameters which require derivatives; e.g., pilot points and singular value decomposition are both methods used with PEST in the WIPP Culebra PA model calibration (Hart et al., 2009).

When working with an ensemble of independent calibrated models, S_{ij} is computed separately for each realization, and then ensemble sensitivity can be computed by appropriately averaging across the realizations. Although this approach was used to calibrate the PA models and the resulting PEST-computed sensitivity matrices (i.e., Jacobians) are saved in CVS, this approach was not used due to two complications. First, the sensitivities in the MODFLOW model are computed between observed heads and pilot point values (not particle travel times to individual parameter values in the model grid). Second, these sensitivity matrices were only computed at the beginning of the calibration, due to the use of the singular value decomposition, which works with "super pilot points" rather than the pilot points themselves.

4.2.3. Adjoint Sensitivity Approach

An alternate approach to computing S_{ij} using finite differences is the use of the adjoint sensitivity equations, where a system of adjoint equations are derived (similar in form to the diffusion equation) and solved using the same model grid with modified boundary conditions and source terms. The sensitivity coefficients are related directly to the adjoint variable, rather than the main variable (typically head or pressure). Although this method is very problem-specific, it has the advantage of making the number of model runs needed to compute S_{ij} proportional to the number of model predictions or observations (m), rather than the number of model parameters (n) (see e.g., Sykes et al., (1985) (1985)). The adjoint approach was used at WIPP during the CCA, in the GRASPII inverse modeling code (see e.g., RamaRao and Reeves (1990)).

Like the perturbation-based approach, the adjoint method works independently on each realization, requiring appropriate averaging across realizations to develop ensemble-based estimates of parameter sensitivity to model predictions.

4.2.4. Sampling-Based Correlation Approach

In the three sensitivity calculation approaches outlined above, the changes in model predictions, due to perturbing each parameter are kept separate; the derivative in Equation 10 is a partial derivative indicating all other independent variables are held constant while each P_j is varied. Individually perturbing each of the model parameters has the benefit of isolating each parameter's effects on the model predictions, but it demands a large number of forward model runs to fill in the large sensitivity matrix. For the WIPP Culebra PA flow model, we have an existing ensemble of calibrated model realizations, which can be used to statistically investigate the correlation between input parameters and the predictions in a post-mortem sense, after all the model runs are finished. The previous three approaches were for a single realization, requiring averaging to reach an ensemble average; the correlation-based approach uses all the realizations to develop a proxy for sensitivity applicable to the WIPP PA model results.

The correlation-based approach used here begins with an ensemble of 100 calibrated models and the metric for relative importance of one model parameter over another is the parameter's correlation with the model prediction, across the ensemble of model realizations.

Each of the 100 simulations associated with the calibrated T-fields prepared for CRA 2009 PABC (Hart et al., 2009) is a realization where all the parameters are "perturbed" together, rather than individually perturbing parameters by ΔP_j. For a given element in the Culebra flow model, there are 100 values of each parameter (e.g., transmissivity); a histogram of $\log_{10}(K_{\mathrm{eff}})$ in a cell south of the WIPP site, and histogram of the travel time to the WIPP LWB are plotted across all 100 realizations in Figure 4-1.

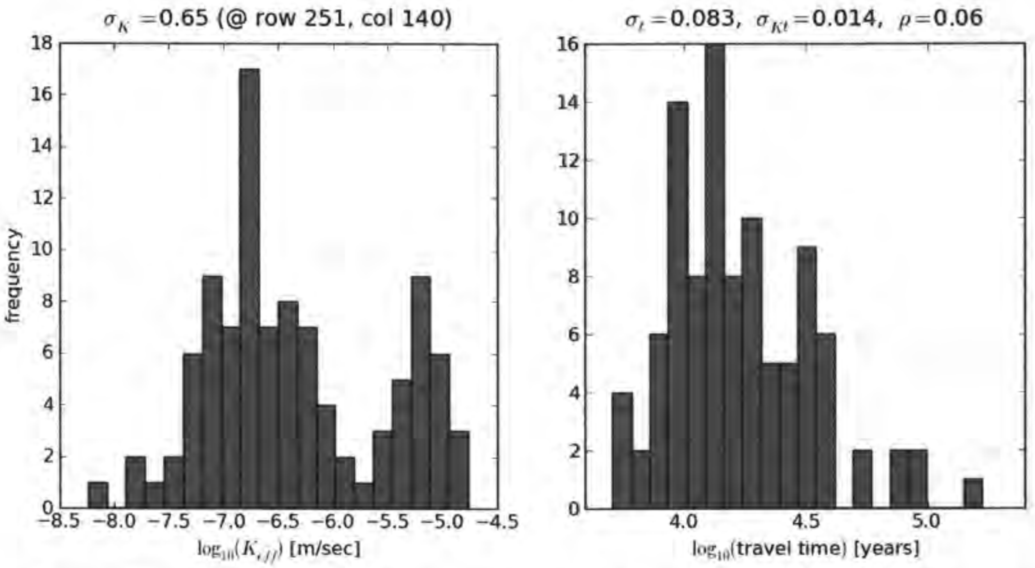


Figure 4-1. Example histograms of a model parameter ($\log_{10}(K_{\mathrm{eff}})$) at a particular cell and model prediction ($\log_{10}$(travel time to WIPP LWB)) across all 100 realizations.

The ensemble correlation approach requires multiple calibrated model realizations, and therefore captures some of the uncertainty captured by the ensemble of models. This is in contrast with the perturbation or adjoint sensitivity approaches which take one calibrated model and use it to estimate parameter sensitivity (essentially assuming a linear approximation of the actual model). To capture the uncertainty given by the ensemble, the perturbation sensitivity approach would have to be performed for each realization of the ensemble – a computationally exasperating process (tens of thousands of individual parameters in each of hundreds of models, with potentially long run-times for each forward run).

McKenna (2004) compared sensitivity coefficients computed using the sampling correlation approach for 100 realizations to those computed with the perturbation sensitivity approach for a single realization, and found that they were similarly but not identically distributed. Although in the sampling-based approach it is not possible to completely differentiate between true and spurious correlations (partial correlation does account for some of this in a statistical sense), the approach is used here based on its computational feasibility and the availability of the 100 realizations.

As opposed to S_{ij}, which is the slope of the linearized relationship between model inputs and outputs, the correlation coefficient, ρ, is a measure of the quality of the linear relationship between two variables (regardless of slope). The correlation coefficient is given by

$$\rho_{PO} = \frac{\frac{1}{n}\sum_{i=1}^{n}(P_i - m_P)(O_i - m_O)}{\sigma_O \sigma_P} = \frac{\sigma_{PO}}{\sigma_O \sigma_P} \tag{12}$$

Where σ_P is the variance of the parameter P, m_P and m_O are the means of P and O respectively, σ_O is the variance of the observation O, and σ_{PO} is the covariance between P and O. ρ indicates the portion of the variance of O which is explained by the variance in P, through an assumed linear relationship (e.g., see Isaaks and Srivastava, (1989), Chapter 3).

When there is more than one free parameter varied at a time, partial correlation is defined as the correlation attributable to a single variable, statistically holding others constant (e.g., see Helton, et al., (2006) §6.4). Partial correlation is demonstrated for the case where a third variable Z is introduced into the problem illustrated in Equation (12);

$$\rho_{PO.Z} = \frac{\rho_{PO} - \rho_{PZ}\rho_{OZ}}{\sqrt{(1-\rho_{PZ}^2)(1-\rho_{OZ}^2)}} \tag{13}$$

where the variables to the right of the dot in the subscript are statistically held constant. This expression reduces the correlation between two variables, by the amount attributed to the spurious correlation between both variables and a third one (here Z). When dealing with more than three variables, there are two primary approaches to computing partial correlation. Conceptually, the simplest is a recursive definition, which is an extension of Equation (13) (e.g., (Spiegel and Stephens, 1999), chapter 15),

$$\rho_{PO.YZ} = \frac{\rho_{PO.Z} - \rho_{PY.Z}\rho_{OY.Z}}{\sqrt{(1-\rho_{PY.Z}^2)(1-\rho_{OY.Z}^2)}} = \frac{\rho_{PO.Y} - \rho_{PZ.Y}\rho_{OZ.Y}}{\sqrt{(1-\rho_{PZ.Y}^2)(1-\rho_{OZ.Y}^2)}} \tag{14}$$

but this recursive approach becomes difficult to compute as the number of variables gets above 4 or 5 (being impossible for hundreds or thousands of variables as in the case for the WIPP model). An alternate definition of Equation (14), in terms of correlation matrices is

$$\rho_{ij.(k\neq i,j)} = \frac{-a_{ij}}{\sqrt{a_{ii}\, a_{jj}}} \tag{15}$$

where $\rho_{ij.(k\neq i,j)}$ is the partial correlation of variables *i* and *j*, accounting for the effects of all other variables; a_{ij} is the matrix inverse of the symmetric correlation matrix *C*, which in the 3×3 case is

$$C = \begin{bmatrix} 1 & \rho_{12} & \rho_{13} \\ \rho_{21} & 1 & \rho_{23} \\ \rho_{31} & \rho_{32} & 1 \end{bmatrix} \tag{16}$$

Often *C* can be poorly scaled and computing partial correlation due to many variables can be numerically unstable, as *C* can be nearly singular and therefore has an ill-defined inverse. The COTS statistical software R includes an implementation (cor2pcor) which computes partial correlation of systems with many variables, utilizing a numerically stable pseudo-inverse approach, automatically scaling the matrices to improve stability. Even though the improved numerical approaches help, the matrix-based approach is intractable for very large problems, because a *n*×*n* matrix must be made (where *n* is the number of active parameters, here over 50,000) in memory; even for single-precision variables this is on the order of a 30-gigabyte matrix. A comparison is made between regular and partial correlation in the results section, using only the area immediately surrounding the WIPP site.

4.3. Model Correlation Results

The calibration of the 100 *T* fields to steady-state and transient heads did not incorporate the groundwater travel time as an estimation variable. The travel time from the center of the WIPP panels (also the location of the Culebra well C-2737) to the WIPP LWB was a separate calculation done after the *T* fields were calibrated; see Figure 4-2 for the travel times and Figure 4-3 for the particle tracks across all 100 realizations.

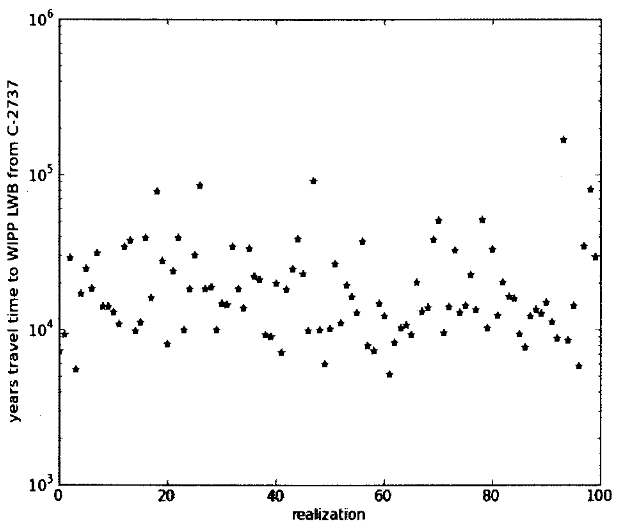


Figure 4-2. Travel times to WIPP LWB for conservative particle (non-dispersive, reactive, with no decay) for 100 realizations used in correlation analysis

The sampling-based sensitivity approach was applied to the results of the 100 calibrated T fields and used to determine the sensitivity of the groundwater travel time to the WIPP boundary with respect to the simulated heads, and effective hydraulic conductivity K_{eff} (the geometric mean of the x- and y-direction hydraulic conductivities),

—— ——— – (17)

where A is the horizontal hydraulic conductivity anisotropy and $K_y=K_xA$. Transmissivity (T) and hydraulic conductivity (K) differ in the Culebra MODFLOW model by a constant thickness, which does not affect correlation calculations. The distribution of K_{eff}, including the mean and standard deviation, across all 100 realizations is plotted in Figure 4-4. The results of these calculations for the K_{eff} are shown in Figure 4-5. Nearly all the wells shown in Figure 4-5 were used in the calibration of the K_{eff} parameter fields (except AEC-7 – see discussion in Section 1.5).

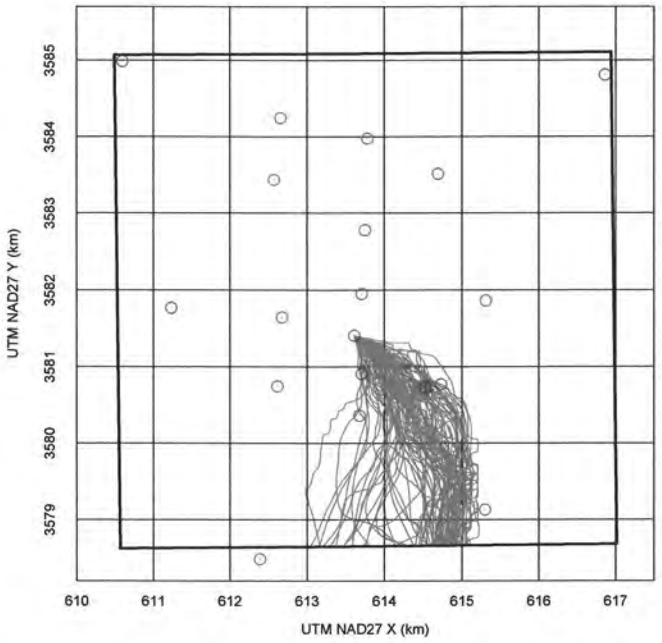


Figure 4-3. Marked water particle tracks from each of the 100 realizations; each track goes from the release point at C-2737 to the WIPP LWB (heavy black square). Green circles are Culebra monitoring wells.

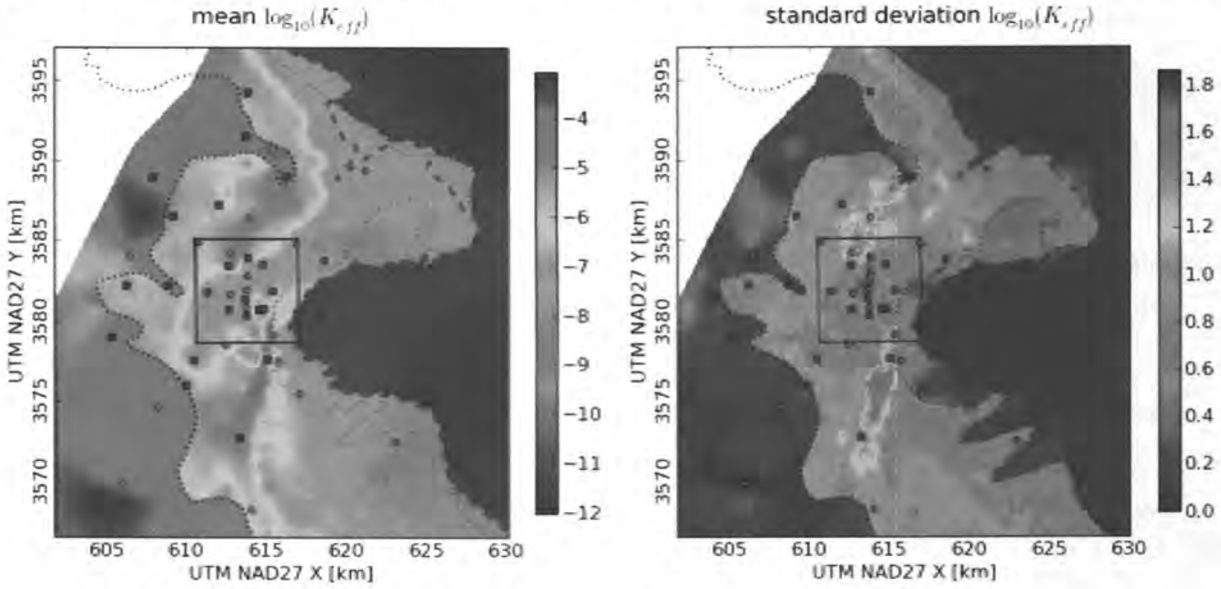
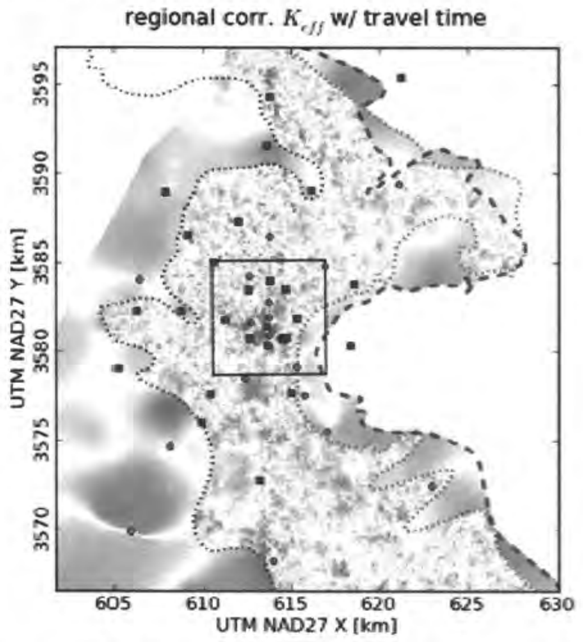


Figure 4-4. Mean and standard deviation of $\log_{10}(K_{eff})$ across all 100 realizations

The correlation results for K_{eff} (see Figure 4-5) show that the magnitudes of the correlation coefficients are not very large in most areas, signifying weak to moderate correlation, both positive and negative, between the travel time to the WIPP boundary and K values used in the model to calculate those travel times. However, the results clearly show regions of relatively higher and lower travel time sensitivity to the two input parameters. The partial correlation statistic (see Figure 4-6) is computed for K_{eff} in each element near the WIPP LWB, accounting for cross-correlation between each element and all other K_{eff} values in the vicinity of WIPP (within 1.5 km of the LWB). Although there are small differences in the distribution of the partial and standard correlation coefficients, the main difference is the absolute value. The partial correlation coefficient is approximately 100 times smaller than the standard correlation coefficient.



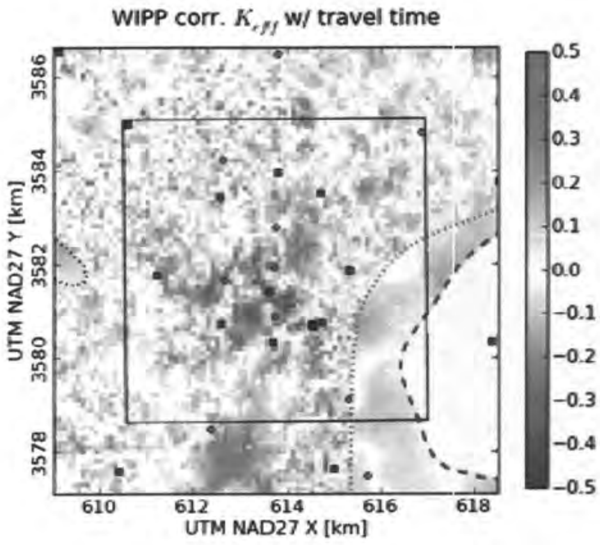


Figure 4-5. Correlation coefficient between $\log_{10}(K_{eff})$ and $\log_{10}$ travel time to WIPP LWB. Steel-cased wells are red circles; fiberglass-cased wells are green squares. Salado dissolution and Rustler halite margins are indicated with dashed lines. Plot on right contains same data plotted on left.

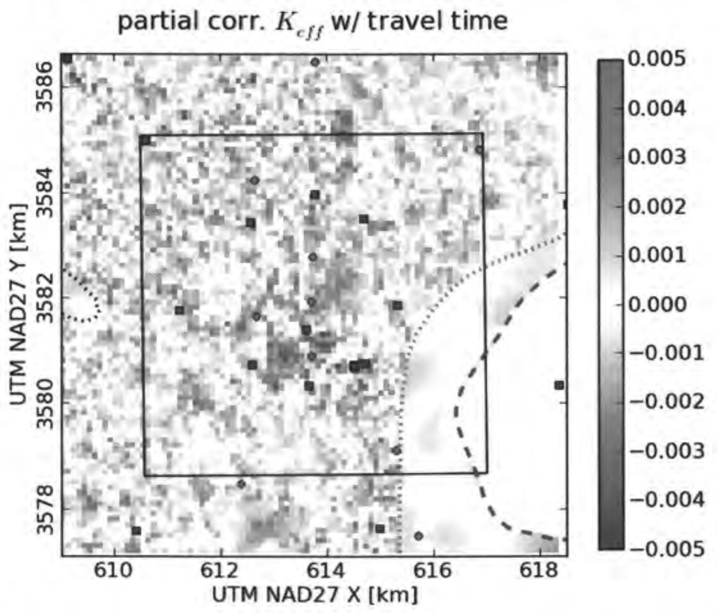


Figure 4-6. Partial correlation between log10(K_{eff}) and $\log_{10}$(travel time) to WIPP LWB

The distribution of model-generated head, across all 100 realizations, is shown in Figure 4-7; here the $\log_{10}$ of the standard deviation is plotted to emphasize the variation in the head across the WIPP LWB. It is interesting to note that visually, areas with the highest variability in K_{eff}

(Figure 4-4) – one of the main inputs to the Culebra flow model – do not correlate with areas of the highest variability in head (Figure 4-7).

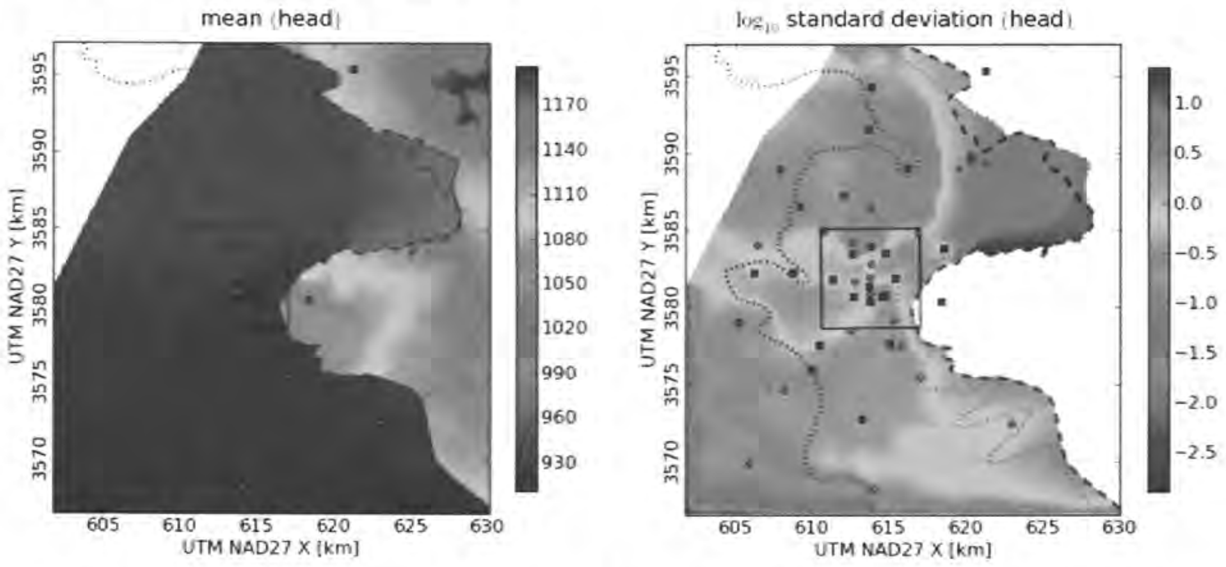
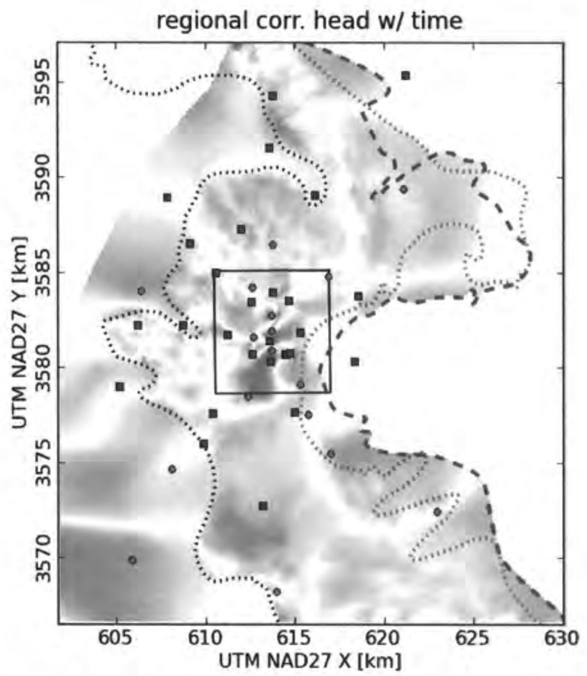


Figure 4-7. Mean and $\log_{10}$ standard deviation of model-simulated head across all 100 realizations

Figure 4-8 shows correlation of $\log_{10}$ model-predicted travel time to model-predicted head (output vs. output); this field is much more smoothly varying than the map of correlation between $\log_{10}$ travel time and K_{eff} (output vs. input). These results are consistent with the difference between model outputs (which must obey the diffusion equation) and the model outputs (which only are forced to have certain geostatistical structure, but are otherwise random).

The partial correlation statistic between model-generated heads and travel times is given in Figure 4-9. Like K_{eff} to travel-time correlation, the partial correlation coefficient is much smaller, but the difference here is only a factor of approximately 30, rather than 100.



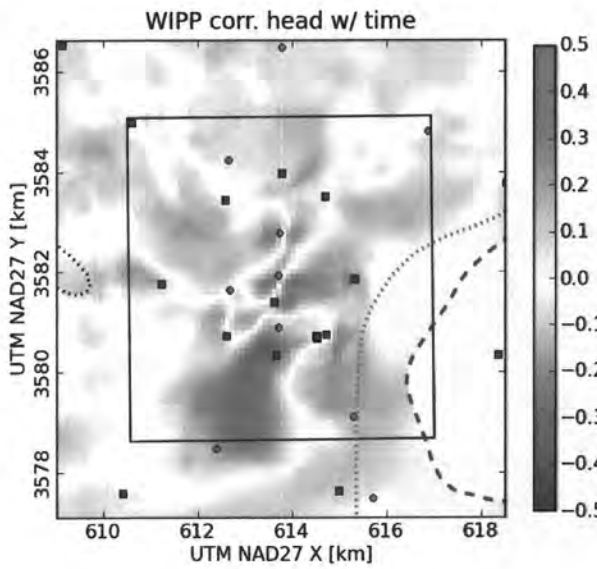


Figure 4-8. Correlation coefficient between model-predicted heads and $\log_{10}$(travel time) to WIPP LWB. Plot on right contains same data plotted on left.

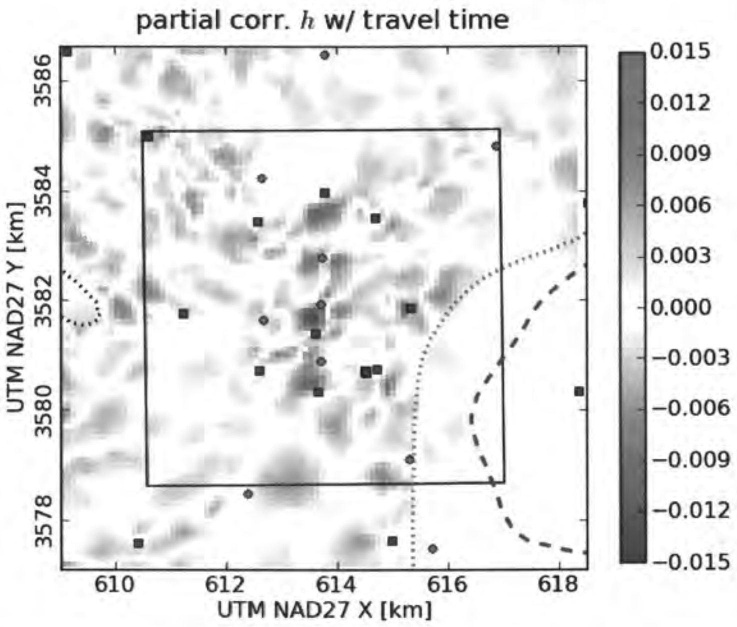


Figure 4-9. Partial correlation coefficient between model-predicted heads and $\log_{10}$(travel time) to WIPP LWB.

The model correlation analysis is the only one of the three given in this report which is not sensitive to the clustering of existing wells. Aside from the fact that the model was calibrated with data collected at the wells, the location of the individual Culebra monitoring wells and model correlation to inputs or output are largely de-coupled. The locations of highest model input/output correlation might occur adjacent to existing monitoring wells.

4.4. Remove One Steel Well

The results of the sampling-based correlation analysis are sampled at locations across the model domain, corresponding to the locations of existing steel-cased wells. The results of this are given in Table 4-1, where the numbers are simply the numerical values sampled from the images of correlation results shown in Figure 4-5 and Figure 4-8.

Table 4-1. Correlation-based analysis results at locations of steel-cased wells. A smaller rank number indicates a higher correlation and therefore assumed importance.

	$\rho\ K_{eff}$	$\lvert\rho\ K_{eff}\rvert$	rank	ρ head	$\lvert\rho$ head$\rvert$	rank	avg rank
ERDA-9	-2.989×10^{-1}	2.989×10^{-1}	1	-2.050×10^{-1}	2.050×10^{-1}	2	1.5
H-3b2	-9.670×10^{-2}	9.670×10^{-2}	6	-1.936×10^{-1}	1.936×10^{-1}	3	4.5
WIPP-19	-9.941×10^{-2}	9.941×10^{-2}	5	1.885×10^{-1}	1.885×10^{-1}	5	5
H-12	1.553×10^{-1}	1.553×10^{-1}	3	-1.462×10^{-1}	1.462×10^{-1}	8	5.5
WIPP-11	1.502×10^{-1}	1.502×10^{-1}	4	-9.576×10^{-2}	9.576×10^{-2}	9	6.5
WIPP-13	2.199×10^{-1}	2.199×10^{-1}	2	-6.001×10^{-2}	6.001×10^{-2}	11	6.5
WIPP-25	-4.109×10^{-2}	4.109×10^{-2}	13	2.250×10^{-1}	2.250×10^{-1}	1	7
USGS-4	-5.631×10^{-2}	5.631×10^{-2}	11	-1.931×10^{-1}	1.931×10^{-1}	4	7.5
AEC-7	7.392×10^{-2}	7.392×10^{-2}	8	7.241×10^{-2}	7.241×10^{-2}	10	9
H-11b4	-5.339×10^{-2}	5.339×10^{-2}	12	1.734×10^{-1}	1.734×10^{-1}	6	9
H-4b	3.852×10^{-2}	3.852×10^{-2}	14	-1.627×10^{-1}	1.627×10^{-1}	7	10.5
H-5b	-7.394×10^{-2}	7.394×10^{-2}	7	1.670×10^{-4}	1.670×10^{-4}	17	12
H-10c	6.302×10^{-2}	6.302×10^{-2}	9	-1.243×10^{-2}	1.243×10^{-2}	16	12.5
H-17	5.662×10^{-2}	5.662×10^{-2}	10	3.295×10^{-2}	3.295×10^{-2}	15	12.5
H-7b1	-1.089×10^{-2}	1.089×10^{-2}	16	5.448×10^{-2}	5.448×10^{-2}	12	14
H-9c	-2.776×10^{-2}	2.776×10^{-2}	15	5.202×10^{-2}	5.202×10^{-2}	13	14
H-2b2	2.715×10^{-3}	2.715×10^{-3}	17	3.768×10^{-2}	3.768×10^{-2}	14	15.5

The results in Table 4-1 are sorted by the average rank between the steel-cased wells for the K_{eff} / $\log_{10}$ travel time correlation (see Figure 4-5) and the head / $\log_{10}$ travel time correlation (see Figure 4-8). A smaller rank number indicates a higher relative correlation in the two cases. Wells with large rank numbers are wells that are located in areas with less correlation between model inputs and outputs.

4.5. Model Correlation Summary

Here, we approximate a true sensitivity analysis using a sampling-based correlation analysis. These sampling-based correlation coefficients are consistent with, but different from, the average sensitivities calculated as numerical derivatives, as was illustrated in (McKenna, 2004). The advantage of this approach to approximating sensitivity is that it is computationally efficient. The sampling-based sensitivity coefficients require an ensemble of calibrated K_{eff} fields, which is computationally burdensome, but they provide an integrated measure of correlation to all of the calibrated K_{eff} fields at once. This approach captures the non-uniqueness of the K_{eff} calibration by using all 100 calibrated fields and also provides a measure of output sensitivity to the input variables at all locations within the domain.

Application of the sampling-based sensitivity approach to the Culebra shows distinct regions of higher and lower correlation to travel time with respect to both calibrated heads and K_{eff}. For travel time sensitivity with respect to heads, the regions of high and low sensitivity are broad and fall mainly within and directly to the south of the WIPP site. Results of travel time sensitivity with respect to K_{eff} show regions of high and low sensitivity that are considerably more localized. The two regions with the greatest absolute correlation for both K_{eff} and model-predicted head are near the C-2737 release point, between the release point and the southern edge of the WIPP LWB, and immediately upstream (north) of the C-2737 release point. These regions of high or low sensitivity can be identified and targeted for additional head monitoring wells and measurements of K_{eff}. Results of the spatial sensitivity calculations are combined with results of other approaches to monitoring well optimization in the following section

4.6. Model Correlation Run Control Summary

4.6.1. Model file checkout and pre-processing run control (Linux)

The model inputs (transmissivity and anisotropy) and outputs (head) were checked out from the `Tfields` and `MiningMod` CVS repositories that are accessible from the PA Linux cluster (`alice.sandia.gov`). The same files exist for each calibrated model realization (see Table 4-2), and they exist in 100 subdirectories with the names `r`*nnn*, where *nnn* is a three-digit number corresponding to the realization name (the numbers range from 001 to 999 and are therefore non-contiguous). The Bash shell script `checkout_model_data.sh` (Section 8.4.1) checks the required files out of CVS (lines 9, 18, 49, and 51), does some manipulation of the directories to simplify the resulting directory structure (lines 30 through 44), and converts the binary MODFLOW head files to ASCII arrays (line 67). Finally the entire file tree of input and output files are zipped up to simplify transfer to Windows from Linux (see lines 72 to 75 – located on the CD in the `analysis\model_correlation` directory inside the `model_files.zip` archive).

Table 4-2. Model files from each calibrated MODFLOW realization

Model File	Description
`r`*nnn*`/modeled_K_field.mod`	calibrated transmissivity field for realization r*nnn*
`r`*nnn*`/modeled_A_field mod`	calibrated anisotropy field for realization r*nnn*
`r`*nnn*`/modeled_head.hed`	model-generated steady-state head for realization r*nnn*
`r`*nnn*`/dtrk.out`	particle tracking results for realization r*nnn*
`r`*nnn*`/{Update,Update2,}`	empty file indicating if the realization originated in the `Update` or `Update2` directories (potentially no file).

The Python script `head_bin2ascii.py` (Section 8.4.2) is used on Linux to convert the binary MODFLOW head files (saved as record-based Fortran unformatted files) to ASCII arrays, based upon the knowledge of the type of data to be expected in the files. Lines 4 through 54 of this Python script define the `FortranFile()` class which is used to encapsulate the functionality needed to read the binary files. Two utility functions (`reshapev2m()` and `floatmatsave()`) are defined in lines 56 through 70. The structure of the MODFLOW binary head files are quite simple; each files is comprised of a single header record and a single array of single-precision head values unwrapped as a vector. The header record contains integers related to the size of the model array subsequently saved, and the head array is saved after that. The Python script reshapes the vector into an array and writes it to an ASCII file in floating point format (line 111).

4.6.2. Partial correlation analysis run control (Windows)

The utility Python script `load_model_data.py` (Section 8.4.3) is called as a library from two other Python scripts to load the 100 realizations of MODFLOW input and output files. This script loops over the 100 r*nnn* subdirectories reading hydraulic conductivity, horizontal anisotropy, travel time to the WIPP LWB, and the model-generated steady-state heads (see Table 4-2). The script then takes the $\log_{10}$ of the K, A and travel times, and defines a logical mask (`wippmask`) for addressing a subset of the model domain only including the WIPP LWB and a buffer of cells surrounding it (lines 58 to 62).

Once the zip archive of ASCII model files is transferred to Windows, the analysis begins with the Python script `export_pcor_inputs.py`, (Section 8.4.4) which loads the results of the 100 realizations (importing the functionality from the `load_model_data.py` script at line 2), saving the results to two large matrices to be processed in R for partial correlation analysis. The matrices saved include the travel time to the WIPP LWB (a single column) concatenate with the head or K_{eff} matrices from a region including the WIPP LWB and a 1,500-m buffer surrounding the LWB, due to a limitation of the approach. A correlation matrix comprised of every model parameter (or head) to every other parameter (or head) is made, the full 307×284=87,188 model cells would result in a correlation matrix with 7,601,921,721 entries (over 56 gigabytes at double precision). The smaller subset of model parameters (WIPP LWB is 64 100-m elements wide and tall + a buffer of 15 elements on each side) results in a large correlation matrix that only has 78,092,569 entries (just under 596 megabytes at double precision). The R script `compute_partial_correlations.R` (Section 8.4.5) simply reads in the matrices saved by the Python script, performs the partial correlation analysis using optimized and numerically stable algorithms (a scaled pseudo-inverse, rather than the simpler – but numerically unstable – matrix inverse), then writes the partial correlation between travel time to the WIPP LWB and either K_{eff} or head in each model cell inside the area surrounding the WIPP site (the last column of the resulting partial correlation matrix – see lines 12 and 19). Output from `export_pcor_inputs.py` and output from `compute_partial_correlations.R` are saved on the CD, along with the intermediate files, in the `analysis\model_correlation\output\` directory.

4.6.3. Correlation analysis run control (Windows)

The Python script `spearman_rank_coefficient.py` (Section 8.4.6) also loads the model data using the `load_model_data.py` module, and also loads the results of the partial correlation calculation done in R (lines 31 to 39). In the loop from lines 41 to 62, the script computes the head vs. travel time and K_{eff} vs. travel time correlations across the 100 realizations at each element in the model domain. The partial correlation results and standard correlation results are then plotted in several forms for figures in the text (Figure 4-1, Figure 4-2, Figure 4-5, Figure 4-6, Figure 4-8, and Figure 4-9), and saved as matrices for later analysis (files located on CD in the `analysis\model_correlation\output\` directory).

5.0 Combining Approaches

This section discusses the combination of the three approaches towards quantifying both the quality of proposed monitoring well locations, and the relative importance of existing steel-cased well locations

5.1. One Additional Monitoring Location

Three different approaches to identifying optimal additional monitoring well locations have been computed. In the case of the geostatistical estimation variance reduction approach, the change in the estimation variance can be computed after adding more wells. However, the results of this approach leads to many locations with high propensity to reduce overall estimation variance and the results of this approach do not uniquely identify one or even a handful of optimal locations for additional wells. To some extent, combining all three of the approaches into a single map reduces this non-uniqueness. Here, the three approaches are combined to provide a combined score, S_c, that identifies the best locations for new wells. The higher the value of the score is, the better that location is for a new well.

The combined score is the sum of the three different fields calculated in the three monitoring approaches scaled appropriately and combined as

$$S_c = \sigma_{OK}^2 + A_r + |r_s| \qquad \textbf{(18)}$$

The three components of S_c are the relative change in the average ordinary kriging variance, σ^2_{OK}, the change in the average triangle interior angle ratio, and the absolute value of the correlation coefficient between travel time to the WIPP boundary and either the estimated transmissivity or head, r_s, each compared relative to the 2007 network. The absolute value of the rank correlation coefficient is used since both positive and negative correlations are of equal importance for locating new monitoring wells. The triangle interior angle ratio is handled differently, because for that metric, negative values are poor places to locate wells. Figure 5-1 shows histograms of the fields that contribute to S_c.

Four different combinations of the input fields are considered, requiring six total input fields. The resulting fields will be comprised of the following four cases:

1. Δ mean kriging variance + Δ mean triangle angle ratio + ρ K_{eff},
2. Δ mean kriging variance + Δ mean triangle angle ratio + ρ head,
3. Δ median kriging variance + Δ median triangle angle ratio + ρ K_{eff},
4. Δ median kriging variance + Δ median triangle angle ratio + ρ head;

either the correlation of travel times to head or Keff are used, and either the mean or median relative kriging and triangle metrics are used.

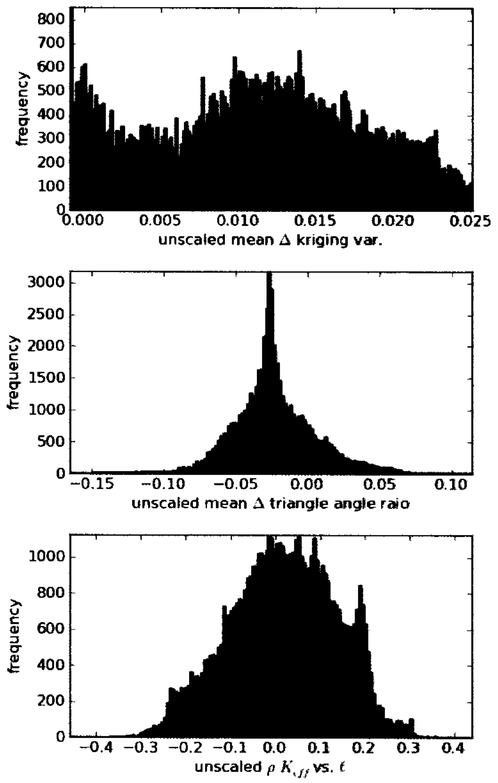


Figure 5-1. Histograms of each component of S_c, before applying scaling.

The three metrics in S_c are already unitless, as reported in their individual previous sections. The results are rescaled here to give them common ranges (a width of unity). The rescaling is accomplished as

$$S_c = \frac{\sigma^2_{OK} - \min(\sigma^2_{OK})}{\max(\sigma^2_{OK}) - \min(\sigma^2_{OK})} + \frac{A_r}{\max(A_r) - \min(A_r)} + \frac{|r_s| - \min(|r_s|)}{\max(|r_s|) - \min(|r_s|)} \tag{19}$$

where the `max()` and `min()` operators define the maximum and minimum values of the different components of the combined score across the entire calculation domain. The triangle metric is handled differently than the others, as it is not shifted to a zero-based origin (no "–min(A_r)" in the numerator); this was done because the negative values of change with respect to the interior angle metric indicate that adding a well at a given location would degrade the quality of the overall average well network.

Histograms of the scaled components to S_c are plotted in Figure 5-2. The top row of plots for the relative change in the kriging variance are simply scaled to the [0,1] interval (they already had a distribution with a minimum value of zero). These distributions are slightly skewed towards 0.0, more so for the change in the median kriging variance. In the second row of plots for the relative change in the triangle angle ratio are scaled to a unit width interval, but they are not shifted (now covering approximately the [-0.6,0.4] interval). These distributions are centrally distributed about a non-zero negative value. The absolute value of the correlation coefficient distribution (bottom row) is now strongly skewed towards 0.0, after taking the absolute value (the original distribution in Figure 5-1 was roughly symmetric about 0.0).

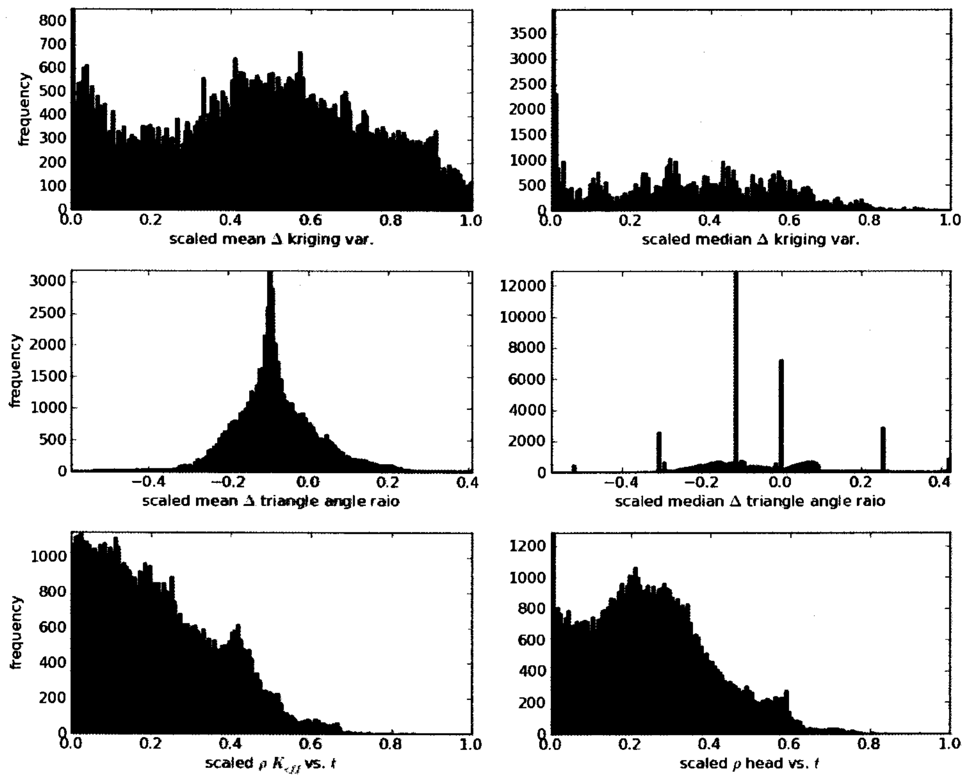


Figure 5-2. Histograms of each component of S_c, after applying scaling

Other than the scaling, one additional change is made to the fields for the mean and median kriging variance. These fields are computed on a two times coarser grid than the triangle angle ratio or the correlation coefficients. The use of this multiplier is to accommodate the long run times for the kriging calculations. The fields resulting from the kriging calculation are copied onto the finer mesh by copying each of the kriging matrix cell's values (without averaging) into the four cells covering the same area in the finer grid. This process is similar to how values were copied from the MODFLOW to SECOTP2D modeling grids in the CRA 2009 PABC calculations (see Kuhlman, (2010a), Appendix A, §1.7).

5.1.1. Results

The theoretical minimum and maximum combined score values for any location in any of the four cases are -0.6 and 2.4 respectively. An image map of the combined score value for case 1 is shown in Figure 5-3. The resulting field is light colored (low score) in most areas surrounding the WIPP LWB and near monitoring wells in the 2007 well network. The resulting image for case 2 is shown in Figure 5-4. The resulting distribution of the case 2 results has lower low values (some negative values, indicated in yellow), and the dark blue location, indicating a good possible location, are more localized than for the means of the same variables (Figure 5-3).

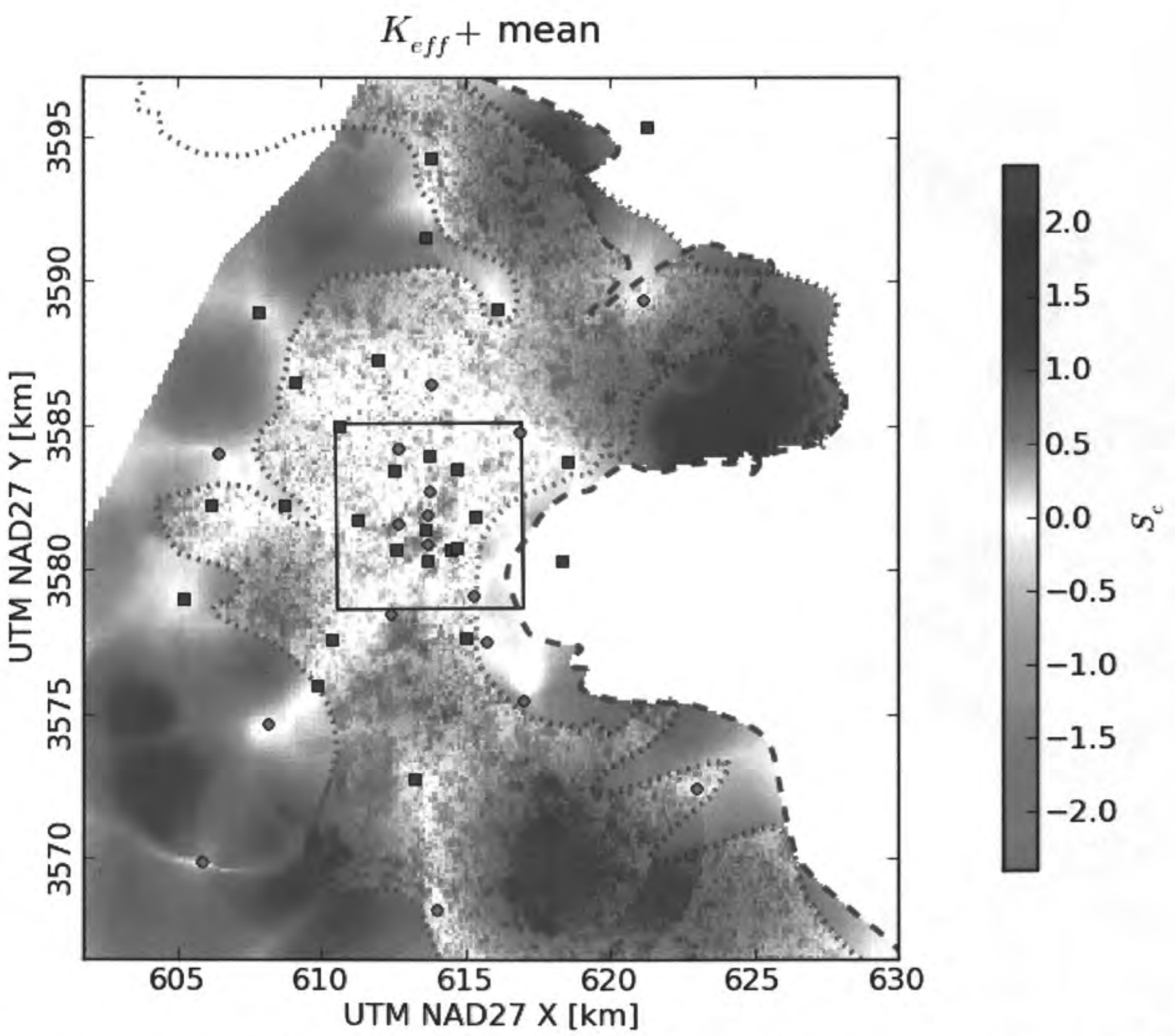


Figure 5-3. Combined scaled results for case 1, using mean Δ kriging variance, mean Δ triangle shape metric, and correlation between K_{eff} and $\log_{10}$ particle travel time. Fiberglass-cased wells are green squares, steel-cased wells are red circles; Salado dissolution and Rustler halite margins are dashed lines; WIPP LWB is black square.

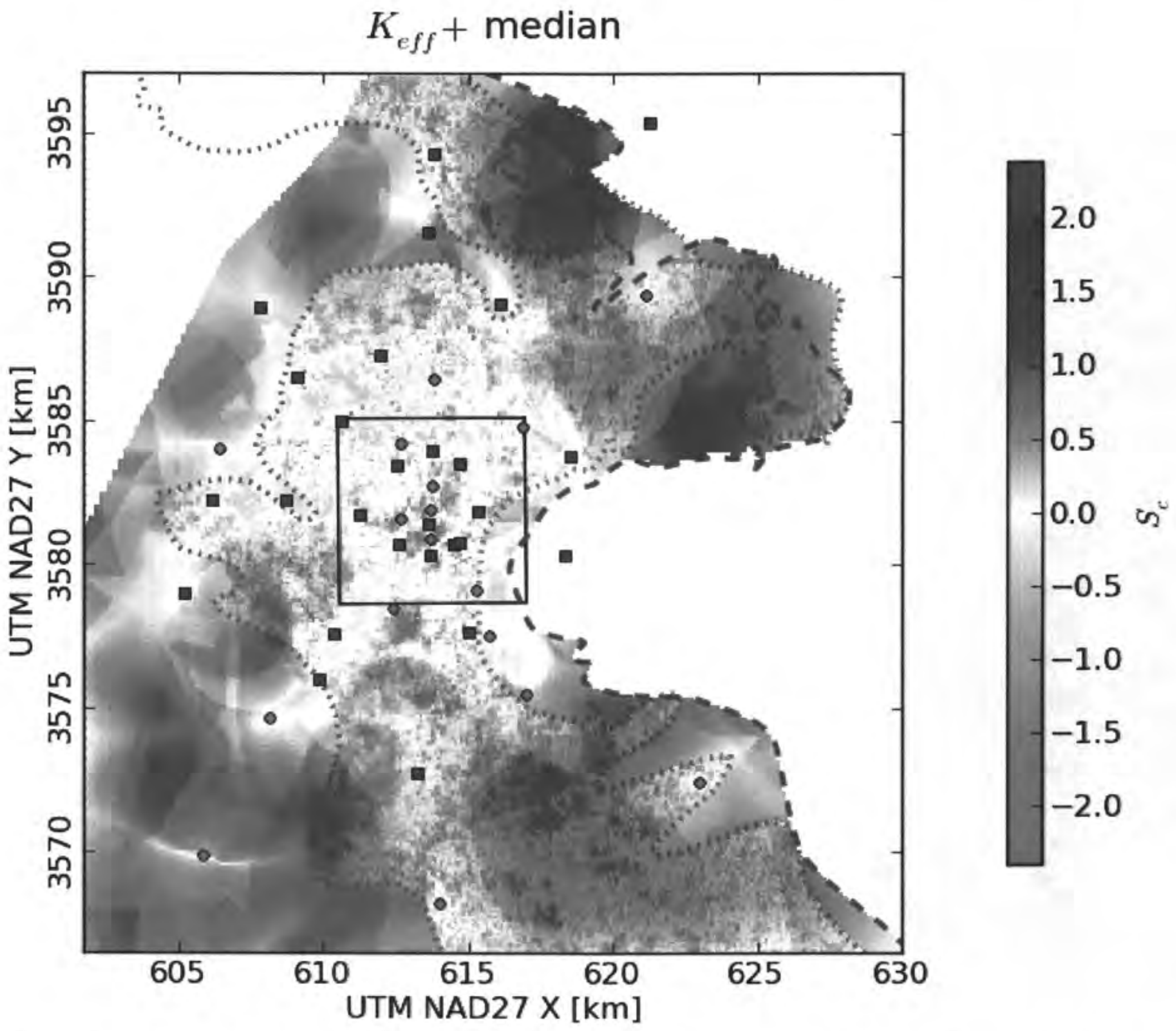


Figure 5-4. Combined scaled results for case 2, using median Δ kriging variance, median Δ triangle shape metric, and correlation between K_{eff} and $\log_{10}$ particle travel time.

The image map showing the results for case 3 is plotted in Figure 5-5. These results are smoother than cases 1 and 2, as was the case for the correlation coefficients that these results contain. There are more isolated possible locations inside the WIPP LWB in case 3 than in cases 1 and 2 (blue areas). The image map showing the results for case 4 is plotted in Figure 5-6. Similar to the differences observed between cases 1 and 2 (Figure 5-3 and Figure 5-4), case 4 has more negative locations (yellow), and the high values outside the WIPP LWB (blue) are more localized than the case considering the mean parameters.

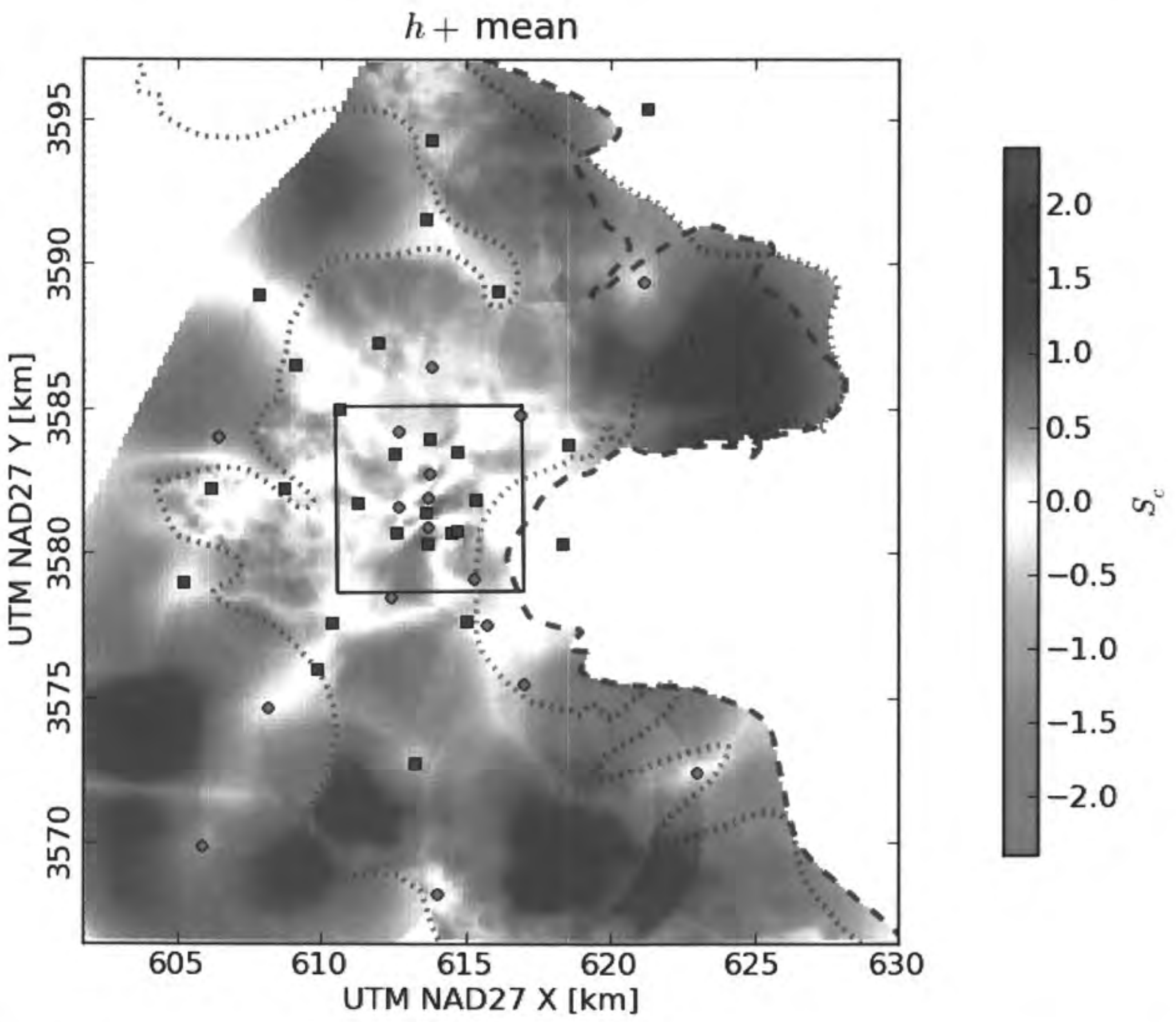


Figure 5-5. Combined scaled results for case 3, using mean Δ kriging variance, mean Δ triangle shape metric, and correlation between modeled head and $\log_{10}$ particle travel time.

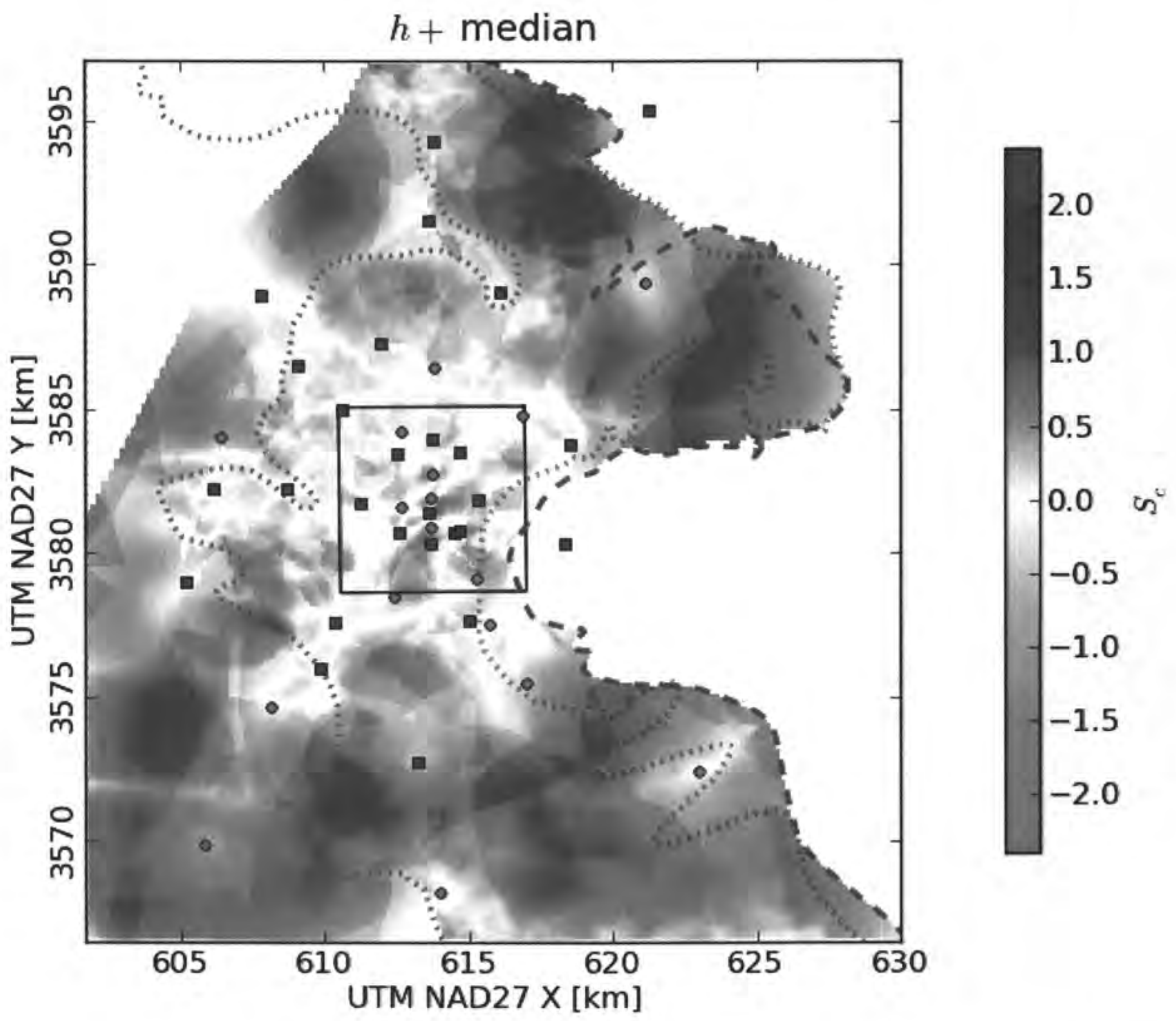


Figure 5-6. Combined scaled results for case 4, using median Δ kriging variance, median Δ triangle shape metric, and correlation between modeled head and $\log_{10}$ particle travel time.

The results of cases 3 and 4 indicate there are areas resulting in relatively high S_c scores inside the WIPP LWB, specifically in the south-central (north of H-4b) and east-central portions (east of the WIPP site buildings). The results in cases 1 and 2 do not indicate any significant high S_c score areas inside the WIPP LWB. This indicates that the methods considered here indicate areas inside the WIPP LWB might be useful for head-monitoring locations regarding head, but additional T values from testing new wells might not be as necessary.

The results outside the WIPP LWB are more focused in cases 2 and 4 (Figure 5-4 and Figure 5-6), where the medians, rather than the means are used. In these cases the two areas with the highest S_c scores (dark blue to purple) are north of the WIPP site between SNL-1 and AEC-7, as well as south of the WIPP site between the DOE Gnome-Coach site (USGS-4) and SNL-12.

In cases 1 and 3, the best new locations for wells would be east of the WIPP LWB, specifically east of SNL-8 and southeast of AEC-7, and south of the WIPP LWB between H-9c and H-10c, also north and west of the DOE Gnome-Coach site (USGS-4).

5.2. Remove One Steel Well

The results of the remove-one-well analyses from the previous sections were plotted together in Figure 5-7. Symbols are scaled according to numerical rank, small rank number correlating to small symbol size.

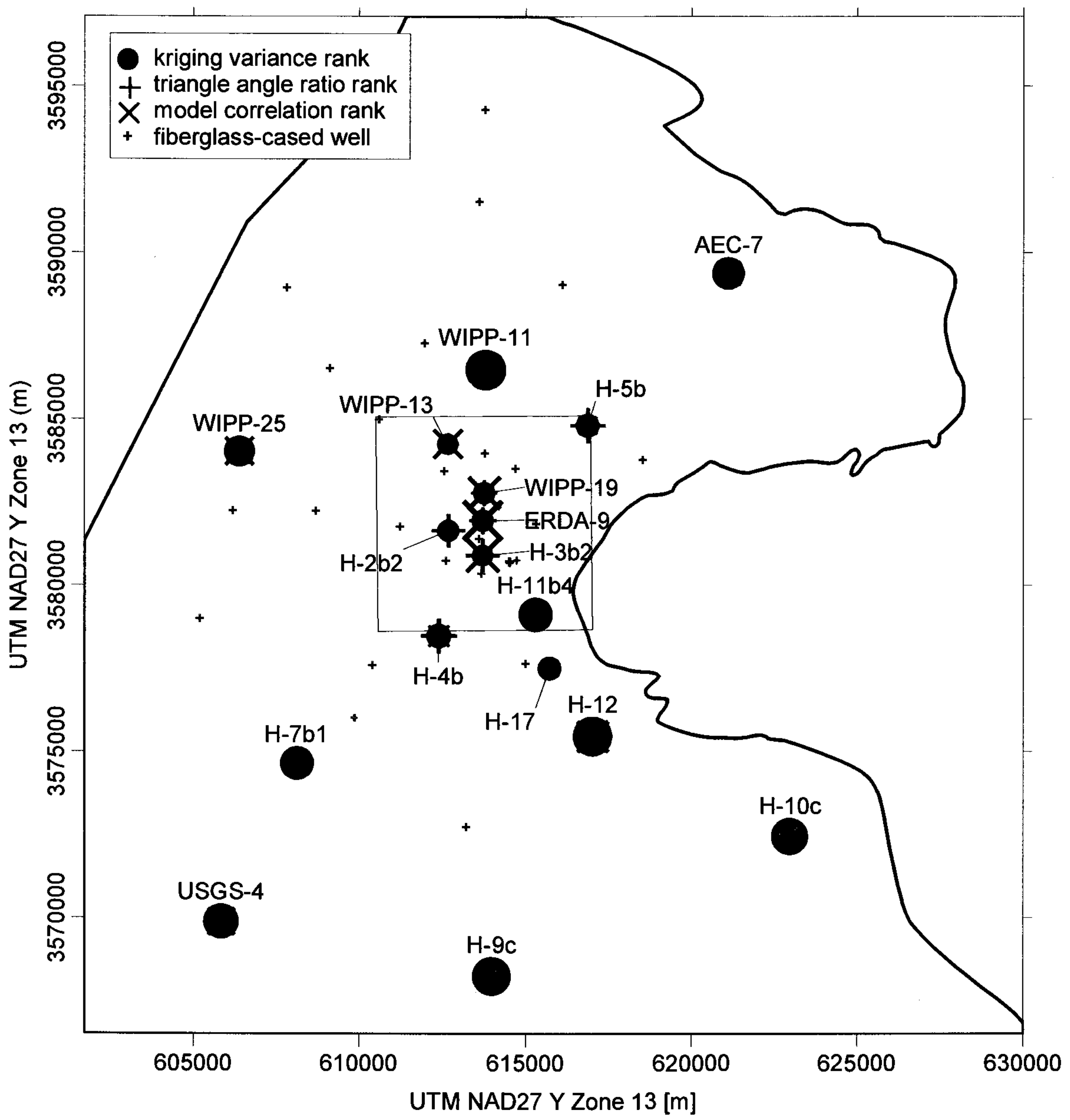


Figure 5-7. Composite plot of steel-cased well rankings from previous sections. Large symbols correspond to greater relative importance for each of the three measures.

Figure 5-7 shows the trend in the kriging variance reduction (filled red circles), where wells inside the WIPP LWB typically have a poor rank, and therefore removing them will have little impact on the kriging variance averaged across the entire model domain (H-11b4 being a slight exception). The results of the triangle gradient estimator maximization process (blue crosses) shows the wells indicated as being most valuable are located in the central and south-east portion of the domain. Aside from the locations inside the WIPP LWB, the wells with high rank regarding the gradient estimator approach are steel wells that are not very near fiberglass-cased wells (H-10c, H-9c, H-12, H-4b, H-5b and AEC-7). The model correlation results are shown as green X's, with the distribution of important steel-cased wells being a more scattered about the MODFLOW model area. These results do not depend on current well locations, aside from the fact that the current wells were used to calibrate the model. ERDA-9 and H-3b2 are in locations where head and Keff are correlated to the model predicted travel time to the WIPP LWB, as would generally be expected. USGS-4, H-12, WIPP-11 and WIPP-25 also have high ranks based on correlation of model results, which are more difficult to explain. The high ranks of these locations are likely due to spurious correlation between the data used in the correlation.

5.2.1. Summary

Three different approaches to monitoring network optimization were used to identify locations where additional wells could improve the network. These three approaches identify: 1) locations where additional wells will reduce the uncertainty in predicting head values at locations without wells; 2) locations where an additional well will allow for maximum improvement in the ability of the existing monitoring well network to identify changes in the magnitude and orientation of the hydraulic gradient by maximizing the quality of local gradient estimators that can be created; and 3) locations where the performance assessment measure of advective travel time to the WIPP boundary is most correlated to the value of head or transmissivity.

These three approaches to monitoring network design all attempt to optimize the network with respect to different objectives. Combining all three of these approaches is done by rescaling each of the raw maps of estimation variance, additional local gradient estimators and sensitivity to have a range (minimum to maximum) of 1.0 and to be unitless. The final combined score maps show the best places to locate additional wells to meet all three objectives when each of the three objectives is given equal weight. The higher the combined score is, the better the location is for a new well. The final combined maps are similar with some minor, but important differences depending on whether or not sensitivity with respect to head or K_{eff} is included in the combined score.

5.3. Method Combination Run Control

The Python script `combine_plot_methods.py` (Section 8.5) loads in the results of the previous three sections, normalizing them to the range $0 \leq x \leq 1$ and summing them up to create composite plots (Figure 5-3 through Figure 5-6) illustrating the optimum location for additional monitoring wells. Histograms of each component before (Figure 5-1) and after (Figure 5-2) scaling are also made for assessing the relative effect each of the three components has on the overall result.

6.0 Conclusions

A set of measurements made in 42 head monitoring wells in the Culebra within and surrounding the WIPP from 2007 were used in this analysis. This set of observations mostly coincided with the freshwater heads used for steady-state calibration of the CRA 2009 PABC MODFLOW model. This head-monitoring network provided the input data for three different approaches to optimizing the monitoring well network. Optimization is interpreted broadly here to include both the identification of new locations where wells could be added to the network to meet some objective and also identification of existing wells that could be removed from the monitoring network as they provide redundant information. The three different approaches to monitoring network optimization examined here are: 1) geostatistical variance reduction; 2) local gradient estimation using combinations of three wells; and 3) sampling-based spatial sensitivity coefficients. In short, the gradient has not changed significantly since the 2004 analysis.

6.1. Summary of Calculations

Geostatistical variance reduction is a fairly common optimization approach (e.g., Rouhani, (1985)) that exploits several properties of the kriging variance to identify new locations where a well could be added to an existing monitoring network to provide the greatest reduction in estimation variance. The same approach can be used to determine existing wells that, upon removal from the monitoring network, provide the smallest increase in the overall estimation variance. Kriging provides an ideal approach to these calculations as the estimation variance calculated through kriging is only a function of the data configuration and not the data values. Therefore, the estimation variance reduction/increase for the addition/removal of a new well can be calculated prior to adding/removing that well from the network. This calculation assumes that the variogram calculated for the head, or residual, values in the network does not change with the addition/removal of a well.

Application of the geostatistical estimation variance calculations to the Culebra network shows that there are many locations where a well can be added to the network that will produce a maximum reduction in the average estimation variance. These locations are all outside of the WIPP site boundaries and the majority of these locations are near the extremities of the MODFLOW model domain. Adding new wells within the WIPP site boundary will not have a significant impact on the estimation variance. The geostatistical estimation variance calculations were also applied to the problem of determining which existing wells to remove from the network. Results for this problem can easily be calculated; however, for removal of more than one well at a time, it is necessary to know what combinations of wells need to be removed to make the problem tractable. Four different base cases were run here and the results show that simultaneous removal of WIPP-13 and another steel-cased well makes an insignificant change in the estimation variance relative to the full 42-well network, while removal of either of other pairs of steel-cased wells has a significant impact (Table 2-5). Averaged across the entire model domain, the removal of wells USGS-4, H-9c, H-10c and AEC-7 would have the largest effect (Figure 2-14). Averaged across the WIPP LWB, removal of wells H-4b, H-5b, H-17 and H-7b1 would have the largest effect (Figure 2-16).

A Delaunay triangulation of the wells in the 2007 monitoring network provides a platform for estimating the quality of triangles as gradient estimators. The interior angle ratio (max angle / min angle) is used as a metric for quantifying the quality of a given arrangement of wells. Local gradient estimators were used to identify the best places to locate additional monitoring wells and the existing wells that could be removed from the network with the smallest impact on the ability of the network to estimate in the gradient.

Results of the calculations to identify locations for additional monitoring wells show that new wells should be located outside of the WIPP site. Additional monitoring wells could optimally be placed north and east of the WIPP LWB, or south between existing wells (Figure 3-10). The well removal calculations were done by removing one well at a time from each of three base case scenarios. Removal of wells in the western portion of the domain, outside the WIPP LWB, has little effect on the quality of the network from the point of view of the triangular gradient estimators (Table 3-1 and Figure 5-7). The removal of steel-cased wells in the southeast or inside the WIPP LWB would have the largest effect on the overall network.

The third approach to monitoring network optimization explored in this report is that of using model correlation to identify locations for new wells where some model output of interest (e.g., travel time) is most sensitive to the transmissivity or head at that location. These correlation coefficients are calculated through a sampling-based technique across 100 calibrated K_{eff} fields. The sampling-based sensitivity coefficients are shown as a map of the sensitivity of the travel time from the repository to the WIPP site boundary with respect to head and transmissivity (Figure 4-5 and Figure 4-8). The results with respect to head show a smoothly varying sensitivity field with large regions of positive and negative correlation between head and travel time. The results with respect to K_{eff} have much more localized regions of positive and negative correlation with travel time being most sensitive to transmissivity at a location directly south of the WIPP site boundary. It is noted that increased knowledge of the spatial variation of the Culebra transmissivity is not a goal of the long-term monitoring network, but transmissivity is an input to the T field calibration process used as input to further PA calculations.

As a final step, the results of the geostatistical estimation variance calculations, the local gradient estimation and the spatial sensitivity coefficients were combined into two "combined score" maps. These maps show, on a normalized scale, the best locations to locate new monitoring wells. In general, these areas are outside of the WIPP site.

6.2. Reexamination of Monitoring Goals

The different purposes, goals and factors that must be taken into account in the design of the Culebra long-term monitoring network were stated in Section 1.2. These goals come from a variety of sources, mainly the state and federal regulatory bodies with WIPP oversight and the ability of the network to provide needed inputs to PA models. Practical factors impacting network design require that the total number of wells in the monitoring network be minimized and that certain wells be retained in the network. The monitoring network should also serve as a vehicle to provide new information to the hydrologic and geologic conceptual models.

The first monitoring network goal is to allow for *determination of the direction and rate of groundwater flow across the WIPP site*. Triangular gradient estimators were developed to meet this goal (Section 3.0). Independently obtained head measurements cannot by themselves determine the direction and magnitude of the hydraulic gradient. For a confined aquifer with a mainly two-dimensional flow pattern, head measurements at three separate locations are necessary to determine the orientation and magnitude of the gradient. Small equilateral triangles are typically the best for estimating gradients over an area from point head measurements, assuming the observed heads result in a gradient large enough to measure over the ambient noise in the system.

The second monitoring goal is to *provide data needed to infer causes of changes in water levels.* Detecting water level change can be done in a single well and an implicit requirement to meet this goal is that there are enough wells in key locations both within and around the WIPP site to detect any water level changes. Checking for the adequate distribution of wells in and around the WIPP site is accomplished using a geostatistical variance reduction approach (Section 2.0). These calculations identify where additional wells are needed and which existing wells can be removed from the network. After a change in water level is detected, the cause of that change must be inferred. There must be enough wells in the proper configuration to infer the cause of a change. The geostatistical variance reduction and three-point estimator approaches to monitoring network design provide networks that maintain enough well density with the proper configurations to infer causes of changes.

The third goal is that the *monitoring network must provide spatially distributed head data adequate to allow both defensible boundary conditions to be inferred for Culebra flow models and defensible calibration of those models.* This goal is related to the previous one in that a network that provides enough wells with the spatial distribution and configuration to detect and infer causes of changes in water levels should also provide the data necessary to infer boundary conditions and calibrate Culebra flow models. Therefore both the geostatistical variance reduction and the gradient estimator approaches and the data gaps and redundancies that they identify apply to this goal as well. Additionally, a third approach to monitoring network design based on model correlation analysis was developed to explicitly incorporate the results of calibrated groundwater flow models directly into the monitoring network design. The set of calibrated groundwater models used as the basis of this third approach incorporates the latest geologic and hydrologic conceptual models. This approach to monitoring network design defines areas along the boundaries and within the groundwater flow model where the model results are most sensitive to the calibrated values of head and transmissivity. Regions of high sensitivity are targeted for future well locations.

In addition to meeting these three goals, a number of other factors were considered in the design of the monitoring network. These included preserving the locations of existing fiberglass and steel-cased wells, identifying wells that provide redundant information, incorporating current hydrologic and geologic conceptual models and identifying locations where questions in the conceptual models can be addressed and/or locations where the groundwater flow models used in PA calculations are correlated to the local values of head and transmissivity. Both the geostatistically-based variance reduction approach and the three-point estimator approach to monitoring network design explicitly considered minimization of the number of wells in the monitoring network through removal of existing wells. Tradeoffs between the minimization of the wells in the network and the ability of the network to provide information on changes in heads were examined. The monitoring network design done here was focused on optimization approaches that are readily quantified into different objective functions. Meeting certain, less easily quantified, factors such as locations where conceptual model questions can be addressed is more difficult and the monitoring networks designed here did not explicitly address this factor.

The results of the calculations done to meet the monitoring goals and the other factors are combined into a series of maps (Figure 5-3 through Figure 5-6) that show the best locations for adding wells to the monitoring network. A map has also been created showing which existing steel-cased wells are the most and least important to maintain within the monitoring network (Figure 5-7).

7.0 Bibliography

American Society of Civil Engineers, 1990. Review of Geostatistics in Geohydrology. *Journal of Hydraulic Engineering*, 116(5), pp.612-58.

Armstrong, M., 1984. Problems with Universal Kriging. *Mathematical Geology*, 16(1), pp.101-08.

Bazaraa, M.S., Sherali, H.D. and Shetty, C.M., 1993. *Nonlinear programming*. 2nd ed. New York: John Wiley and Sons.

Chiles, J.-P. and Delfiner, P., 1999. *Geostatistics: Modeling Spatial Uncertainty*. New York: Wiley Interscience.

Cole, B.E. and Silliman, S.E., 1996. Estimating the Horizonatal Gradient in Heterogeneous Unconfined Aquifers: Comparison of Three-Point Schemes. *Ground Water Monitoring & Remediation*, 16(2), pp.84-91.

Conover, W.J., 1980. *Practical Nonparametric Statistics*. 2nd ed. New York: John Wiley and Sons.

Conwell, P.M., Silliman, S.E. and Zheng, L., 1997. Design of a Piezometer Network for the Estimation of the Sample Variogram of the Hydrualic Gradient: The Role of the Instrument. *Water Resources Research*, 33(11), pp.2489-94.

Deutsch, C.V. and Journel, A.G., 1998. *GSLIB: Geostatistical Software Library and User's Guide*. 2nd ed. New York: Oxford University Press.

DOE, 2009. *Waste Isolation Pilot Plant Groundwater Protection Program Plan*. Carlsbad, NM. DOE/WIPP-06-3339.

Doherty, J., 2000. *PEST Manual*. Brisbane, Australia: Watermark Numerical Computing.

Goovaerts, P., 1998. *Geostatistics for Natural Resources Evaluation*. New York: Oxford University Press.

Hart, D.B., 2010. *WIPP PA Validation Document for MODFLOW-2000 Version 1.6: addenda for direct solver*. Carlsbad, NM: Sandia National Laboratories. ERMS 552422.

Hart, D.B., Beauheim, R.L. and McKenna, S.A., 2009. *Analysis Report for Task 7 of AP-114: Calibration of Culebra Transmissivity Fields*. Carlsbad, NM: Sandia National Laboratories. ERMS 552391.

Hart, D.B., Holt, R.M. and McKenna, S.A., 2008. *Analysis Report for Task 5 of AP-114: Generation of Revised Base Transmissivity Fields*. Carlsbad, NM: Sandia National Laboratories. ERMS 549597.

Helton, J.C., Johnson, J.D., Sallaberry, C.J. and Storlie, C.B., 2006. *Survey of Sampling-Based Methods for Uncertainty and Sensitivity Analysis*. Albuquerque, NM: Sandia National Laboratories. SAND2006-2901.

Isaaks, E.H. and Srivastava, R.M., 1989. *An Introduction to Applied Geostatistics*. New York: Oxford University Press.

Johnson, P.B., 2009. *Potentiometric Surface, Adjusted to Equivalent Freshwater Heads, of the Culebra Dolomite Member of the Rustler Formation near the WIPP Site*. Carlsbad, NM: Sandia National Laboratories. ERMS 548162.

Kitanidis, P.K., 1997. *Introduction to Geostatistics: Applications in Hydrogeology*. New York: Cambridge University Press.

Kuhlman, K.L., 2008. *Analysis Plan for Optimization and Minimization of the Culebra Monitoring network for the WIPP, AP-111 Revision 1*. Carlsbad, NM: Sandia National Laboratories.

Kuhlman, K.L., 2010a. *Analysis Report for the CRA-2009 PABC Culebra Flow and Transport Calculations*. Carlsbad, NM: Sandia National Laboratories. ERMS 552951.

Kuhlman, K.L., 2010b. *Development of Culebra T Fields for CRA 2009 PABC*. Carlsbad, NM: Sandia National Laboratories. ERMS 553276.

McKenna, S.A., 2004. *Analysis Report AP-111, Culebra Water Level Monitoring Network Design*. Carlsbad, NM: Sandia National Laboratories. ERMS 540477.

McKenna, S.A. and Wahi, A., 2006. Local Hydraulic Gradent Estimator Analysis of Long-Term Monitoring Networks. *Ground Water*, 44(5), pp.723-31.

Meigs, L.C. et al., eds., 2000. Interpretation of Tracer Tests Performed in the Culebra Dolomite at the Waste Isolation Pilot Plant Site. Carlsbad, NM: Sandia National Laboratories. SAND97-3109.

Menke, W., 1984. *Geophysical Data Analysis: Discrete Inverse Theory*. 1st ed.

R Development Team, 2009. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

RamaRao, B.S. and Reeves, M., 1990. *Theory and Verification for the GRASP II Code for Adjoint-Sensitivity Analysis of Steady-State and Transient Groundwater Flow*. Albuquerque, NM: Sandia National Laboratories. SAND89-7143.

Rouhani, S., 1985. Variance Reduction Analysis. *Water Resources Research*, 21(6), pp.837-46.

Silliman, S.E. and Frost, C., 1998. Monitoring Hydraulic Gradient Using Three-Point Estimator. *Journal of Environmental Engineering*, 124(6), pp.517-23.

Silliman, S.E. and Mantz, G., 2000. The Effect of Measurement Error on Estimating the Hydraulic Gradient in Three Dimensions. *Ground Water*, 38(1), pp.114-20.

Spiegel, M.R. and Stephens, L.J., 1999. *Theory and Problems of Statistics*. 3rd ed. McGraw-Hill.

Sykes, J.F., Wilson, J.L. and Andrews, R.W., 1985. Sensitivity Analysis for Steady-State Groundwater Flow Using Adjoint Operators. *Water Resources Research*, 21(3), pp.359-71.

Tuckfield, R.C., Shine, E.P., Hiergesell, R.A., Denham, M.E., Reboul, S. and Beardsley, C., 2001. *Using Geosceince and Geostatistics to Optimize Groudnwater Monitoring Networks at the Savannah River Site*. WSRC-MS-2001-00145.

U.S. EPA, 1996. Title 40 CFR Part 194: Criteria for the Certification and Recertification of the Waste. pp.5223-45.

U.S, n.d.

Warrick, A.W. and Myers, D.E., 1984. Optimization of sampling locations for variogram calculations. *Water Resources Research*, 23(3), pp.496-500.

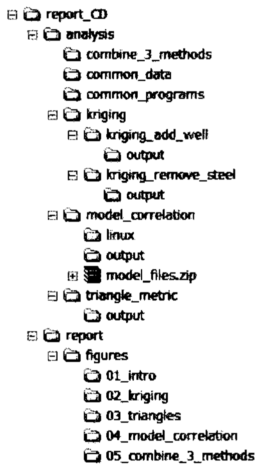
Zheng, C. and Bennett, G.D., 2002. *Applied Contaminant Transport Modeling*. 2nd ed. Wiley Interscience.

8.0 Run Control Script Listings

This appendix lists the source code for the scripts written for and used in this analysis report, and documents them to allow their reasonable verification and future use, according to NP 19-1. The scripts listed in this section neither model physical phenomena nor solve differential equations that model physical phenomena. Rather they are utility codes that process inputs and summarize outputs for other modeling codes (i.e., KT3D). The scripts are heavily commented (green text) to allow the flow of the execution to be easily followed.

8.1. Listing of Files Included on CD

The following directory listing (Table 8-1) corresponds to the directory tree given after it in Figure 8-1.



```
report_CD
  analysis
    combine_3_methods
    common_data
    common_programs
    kriging
      kriging_add_well
        output
      kriging_remove_steel
        output
    model_correlation
      linux
      output
      model_files.zip
    triangle_metric
      output
  report
    figures
      01_intro
      02_kriging
      03_triangles
      04_model_correlation
      05_combine_3_methods
```

Figure 8-1. Directory Tree of CD

Table 8-1. CD Directory listing

```
C:\report_CD>dir /S /TC
 Volume in drive C is DriveC
 Volume Serial Number is 542A-10F7

 Directory of C:\report_CD

04/10/2010  10:50 AM    <DIR>          .
04/10/2010  10:50 AM    <DIR>          ..
04/07/2010  12:42 PM    <DIR>          analysis
04/07/2010  12:42 PM    <DIR>          report
               0 File(s)              0 bytes

 Directory of C:\report_CD\analysis

04/07/2010  12:42 PM    <DIR>          .
04/07/2010  12:42 PM    <DIR>          ..
04/07/2010  12:51 PM    <DIR>          combine_3_methods
04/07/2010  12:44 PM    <DIR>          common_data
04/07/2010  12:50 PM    <DIR>          common_programs
04/07/2010  12:43 PM    <DIR>          kriging
04/07/2010  12:43 PM    <DIR>          model_correlation
04/07/2010  12:43 PM    <DIR>          triangle_metric
               0 File(s)              0 bytes

 Directory of C:\report_CD\analysis\combine_3_methods

04/07/2010  12:51 PM    <DIR>          .
```

Table 8-1. CD Directory listing

```
04/07/2010  12:51 PM    <DIR>          ..
04/11/2010  12:10 PM            10,136 combine_plot_methods.py
04/11/2010  04:11 PM               997 composite_remove_one_steel.dat
               2 File(s)         11,133 bytes

 Directory of C:\report_CD\analysis\common_data

04/07/2010  12:44 PM    <DIR>          .
04/07/2010  12:44 PM    <DIR>          ..
04/10/2010  01:46 PM             1,776 2007_well_data.dat
04/10/2010  01:46 PM             1,823 2007_well_data_for_trend.dat
04/10/2010  01:46 PM             1,606 2007_well_data_for_triangles.dat
04/10/2010  01:46 PM             2,210 2007_well_data_with_names.dat
04/10/2010  01:46 PM               379 2007_well_names.dat
04/10/2010  01:46 PM               331 2007_well_names_for_triangles.dat
04/10/2010  01:46 PM             1,694 base_data.dat
04/10/2010  01:46 PM             5,415 h2_200711.bln
04/10/2010  01:46 PM             5,799 h3_200711.bln
04/10/2010  01:46 PM         1,395,622 model_cells_100_inside_totalbdry.dat
04/10/2010  01:46 PM           350,196 model_cells_200_inside_totalbdry.dat
04/10/2010  01:46 PM           156,766 model_cells_300_inside_totalbdry.dat
04/10/2010  01:46 PM            87,626 model_cells_400_inside_totalbdry.dat
04/10/2010  01:46 PM            56,668 model_cells_500_inside_totalbdry.dat
04/10/2010  01:46 PM            40,040 model_cells_600_inside_totalbdry.dat
04/10/2010  01:46 PM                64 model_domain_specs.dat
04/10/2010  01:46 PM               105 modflow_boundary.bln
04/10/2010  01:46 PM               189 no-flow-area-only.bln
04/10/2010  01:46 PM             8,532 total_boundary.bln
04/10/2010  01:46 PM               105 wipp_boundary.bln
              20 File(s)      2,116,946 bytes

 Directory of C:\report_CD\analysis\common_programs

04/07/2010  12:50 PM    <DIR>          .
04/07/2010  12:50 PM    <DIR>          ..
04/07/2010  12:45 PM           157,184 KT3D.EXE
04/07/2010  12:50 PM             1,807 redwhitemap.m
               2 File(s)        158,991 bytes

 Directory of C:\report_CD\analysis\kriging

04/07/2010  12:43 PM    <DIR>          .
04/07/2010  12:43 PM    <DIR>          ..
04/07/2010  12:49 PM    <DIR>          kriging_add_well
04/07/2010  12:50 PM    <DIR>          kriging_remove_steel
               0 File(s)              0 bytes

 Directory of C:\report_CD\analysis\kriging\kriging_add_well

04/07/2010  12:49 PM    <DIR>          .
04/07/2010  12:49 PM    <DIR>          ..
04/07/2010  12:52 PM               882 generate_model_cell_masks.m
04/07/2010  12:44 PM            11,658 krig_plus_one.py
04/07/2010  12:45 PM               327 kt3d_driver.bat
04/10/2010  01:47 PM    <DIR>          output
04/07/2010  12:46 PM                92 shared_data.py
               4 File(s)         12,959 bytes

 Directory of C:\report_CD\analysis\kriging\kriging_add_well\output

04/10/2010  01:47 PM    <DIR>          .
04/10/2010  01:47 PM    <DIR>          ..
04/10/2010  01:47 PM           262,416 addone_mod_results_corrcoef.dat
04/10/2010  01:47 PM           268,008 addone_mod_results_max.dat
04/10/2010  01:47 PM           263,199 addone_mod_results_mean.dat
04/10/2010  01:47 PM           263,393 addone_mod_results_median.dat
04/10/2010  01:47 PM           271,115 addone_mod_results_stdev.dat
04/10/2010  01:47 PM           262,416 addone_wipp_results_corrcoef.dat
```

Table 8-1. CD Directory listing

```
04/10/2010  01:47 PM           267,024 addone_wipp_results_max.dat
04/10/2010  01:47 PM           266,686 addone_wipp_results_mean.dat
04/10/2010  01:47 PM           262,416 addone_wipp_results_median.dat
04/10/2010  01:47 PM           268,976 addone_wipp_results_stdev.dat
04/10/2010  01:47 PM                26 base_stats.out
04/11/2010  03:43 PM           196,812 X.dat
04/11/2010  03:42 PM           218,680 Y.dat
              13 File(s)      3,071,167 bytes

 Directory of C:\report_CD\analysis\kriging\kriging_remove_steel

04/07/2010  12:50 PM    <DIR>          .
04/07/2010  12:50 PM    <DIR>          ..
04/07/2010  12:52 PM             1,696 krig_remove_one_steel.py
04/07/2010  12:52 PM             2,381 krig_remove_two_steel.py
04/10/2010  01:47 PM    <DIR>          output
04/09/2010  11:58 AM            64,512 remove_one_well_results2_2010.xls
04/09/2010  11:59 AM           164,352 remove_two_well_results3.xls
               4 File(s)        232,941 bytes

 Directory of C:\report_CD\analysis\kriging\kriging_remove_steel\output

04/10/2010  01:47 PM    <DIR>          .
04/10/2010  01:47 PM    <DIR>          ..
04/10/2010  01:50 PM             1,671 model_results_one.dat
04/10/2010  01:48 PM             4,161 remove_two_model.csv
04/10/2010  01:48 PM             4,131 remove_two_wipp.csv
04/10/2010  01:50 PM             1,604 wipp_results_one.dat
               4 File(s)         11,567 bytes

 Directory of C:\report_CD\analysis\model_correlation

04/07/2010  12:43 PM    <DIR>          .
04/07/2010  12:43 PM    <DIR>          ..
04/11/2010  02:02 PM               719 compute_partial_correlations.R
04/11/2010  01:57 PM               613 export_pcor_inputs.py
04/11/2010  02:13 PM    <DIR>          linux
04/11/2010  01:58 PM             1,975 load_model_data.py
04/11/2010  11:51 AM       120,751,461 model_files.zip
04/11/2010  01:58 PM    <DIR>          output
04/07/2010  12:55 PM             7,611 spearman_rank_coefficient.py
               5 File(s)    120,762,379 bytes

 Directory of C:\report_CD\analysis\model_correlation\linux

04/11/2010  02:13 PM    <DIR>          .
04/11/2010  02:13 PM    <DIR>          ..
04/11/2010  11:51 AM             2,169 checkout_model_data.sh
04/11/2010  11:51 AM             3,714 head_bin2ascii.py
               2 File(s)          5,883 bytes

 Directory of C:\report_CD\analysis\model_correlation\output

04/11/2010  01:58 PM    <DIR>          .
04/11/2010  01:58 PM    <DIR>          ..
04/11/2010  02:52 PM         1,084,380 corr_head_vs_time.dat
04/11/2010  02:52 PM         1,072,200 corr_keff_vs_time.dat
04/11/2010  01:58 PM        10,670,500 head_trav.dat
04/11/2010  02:12 PM           171,444 hpc.out
04/11/2010  03:59 PM         1,126,533 keff_mean.out
04/11/2010  01:58 PM         9,786,982 keff_trav.dat
04/11/2010  03:59 PM         1,046,563 keff_var.out
04/11/2010  02:12 PM           185,720 kpc.out
               8 File(s)     25,144,322 bytes

 Directory of C:\report_CD\analysis\triangle_metric

04/07/2010  12:43 PM    <DIR>          .
```

Table 8-1. CD Directory listing

```
04/07/2010  12:43 PM    <DIR>          ..
04/10/2010  03:52 PM    <DIR>          output
04/07/2010  12:53 PM             6,230 triangles_add_one.m
04/07/2010  12:54 PM            13,051 triangles_remove_one.m
04/07/2010  12:54 PM             7,735 triangles_remove_two.m
               3 File(s)         27,016 bytes

 Directory of C:\report_CD\analysis\triangle_metric\output

04/10/2010  03:52 PM    <DIR>          .
04/10/2010  03:52 PM    <DIR>          ..
04/10/2010  03:53 PM         1,395,622 triangles_add_one_mean.dat
04/10/2010  03:53 PM         1,395,622 triangles_add_one_median.dat
               2 File(s)      2,791,244 bytes

 Directory of C:\report_CD\report

04/07/2010  12:42 PM    <DIR>          .
04/07/2010  12:42 PM    <DIR>          ..
04/09/2010  10:38 AM    <DIR>          figures
               0 File(s)              0 bytes

 Directory of C:\report_CD\report\figures

04/09/2010  10:38 AM    <DIR>          .
04/09/2010  10:38 AM    <DIR>          ..
04/09/2010  10:39 AM    <DIR>          01_intro
04/09/2010  10:39 AM    <DIR>          02_kriging
04/09/2010  10:39 AM    <DIR>          03_triangles
04/09/2010  10:40 AM    <DIR>          04_model_correlation
04/09/2010  10:40 AM    <DIR>          05_combine_3_methods
               0 File(s)              0 bytes

 Directory of C:\report_CD\report\figures\01_intro

04/09/2010  10:39 AM    <DIR>          .
04/09/2010  10:39 AM    <DIR>          ..
04/09/2010  10:38 AM            13,001 fig01_fiber_vs_steel_well_locations.srf
               1 File(s)         13,001 bytes

 Directory of C:\report_CD\report\figures\02_kriging

04/09/2010  10:39 AM    <DIR>          .
04/09/2010  10:39 AM    <DIR>          ..
04/10/2010  01:03 PM             3,798 krig_add_one_plotting.m
04/09/2010  12:01 PM           983,627 may2007_variogram_modela.srf
04/09/2010  10:53 AM        16,893,066 perturbation_spread_of_variograms.srf
04/10/2010  11:40 AM           111,616 piecewise_linear_trend.xls
04/09/2010  11:57 AM            15,469 remove_one_steel_well.srf
04/09/2010  10:53 AM            28,672 trend_surface_remove_one_results.xls
               6 File(s)     18,036,248 bytes

 Directory of C:\report_CD\report\figures\03_triangles

04/09/2010  10:39 AM    <DIR>          .
04/09/2010  10:39 AM    <DIR>          ..
04/09/2010  10:40 AM             2,072 random_points_triangle_explanation.mat
04/09/2010  10:42 AM             4,384 three_point_estimator_fig.m
04/09/2010  10:42 AM           539,206 three_point_estimator_log10r.tif
04/09/2010  12:00 PM         1,150,768 three_triangle_metrics.fig
04/09/2010  10:40 AM         1,317,464 triangulation_explanation.fig
               5 File(s)      3,013,894 bytes

 Directory of C:\report_CD\report\figures\04_model_correlation

04/09/2010  10:40 AM    <DIR>          .
04/09/2010  10:40 AM    <DIR>          ..
```

Table 8-1. CD Directory listing

```
 Directory of C:\report_CD\report\figures\05_combine_3_methods

04/09/2010  10:40 AM    <DIR>          .
04/09/2010  10:40 AM    <DIR>          ..
               0 File(s)              0 bytes

     Total Files Listed:
              81 File(s)    175,409,691 bytes
              65 Dir(s)  147,501,948,928 bytes free
```

8.2. Kriging Variance Minimization Scripts

The following scripts were used in the kriging variance minimization (see Section 2.0).

8.2.1. R script `plot_linear_fit_summary.R`

The following R script computes the linear fit surface (Equation 1, in Section 2.1) and the related summary statistics given in Figure 2-3 and Table 2-1 using built-in statistical functions.

```
# this R script computes the best fit linear model through the freshwater
# head data and plots some summary statistics included as figures in the
# analysis report.

# load in data
wells <- read.table('../../common_data/2007_well_data_for_trend.dat')
row.names(wells) <- read.table('../../common_data/2007_well_names.dat')$V1
names(wells) <- c('x','y','fwh','res','casing','flag')
attach(wells)

# don't select SNL-6 and SNL-15 (they have -999 in res column)
# and don't use redundant H-19 wells
mask <- flag == 1
wells.lm <- lm(fwh[mask]~x[mask]+y[mask])
summary(wells.lm)
par(mfrow=c(2,2))
plot(wells.lm)
```

8.2.2. Python script `remove_one_variogram_effects.py`

The following Python script computes the best-fit trend surface through the dataset after individually removing each steel-cased well. The two outputs from this script are the effects of removing a well on the best-fit linear surface (see Figure 2-4) and the resulting smaller-by-one datasets used to compute experimental variograms via Surfer in Figure 2-6.

```
import numpy as np
import os

modelDat = np.loadtxt(r'..\..\common_data\model_domain_specs.dat')

# use midpoint of model domain for origin of surface fitting
# to improve condition number of matrices in least-squares fitting
xmid = (modelDat[2,0] + modelDat[1,0])/2.0
ymid = (modelDat[2,1] + modelDat[1,1])/2.0

fh = open(r'..\..\common_data\2007_well_names.dat','r')
names = [line.rstrip() for line in fh]
fh.close()

wellDat = np.loadtxt(r'..\..\common_data\2007_well_data_for_trend.dat',dtype=np.float64)

# data columns: X,Y,FWH,res,casing,flag
# FWH :: may 2007 freshwater head
# res :: residual computed using R
# casing :: 1=steel, 0=fiberglass/pvc
# flag :: 0= do not use in trend analysis, 2=do not use at all,
#          1= use in both trend & variogram analysis

fh = open('trend_surface_remove_one_results.csv','w')
fh.write('well,sum squared error,condition number,rank,R^2,A,B,C,gradient,angle\n')

trendWells = wellDat[wellDat[:,5]==1]
trendNames = [name for (i,name) in enumerate(names) if wellDat[i,5]==1]
ntwells = trendWells.shape[0]
trendNames.append('base_case')

# additional wells used in variogram analysis, but not in trend analysis H-19b{2,3,4,5,6,7}
varioWells = wellDat[wellDat[:,5]==0]
varioNames = [name for (i,name) in enumerate(names) if wellDat[i,5]==0]
nvwells = varioWells.shape[0]

for i in xrange(ntwells+1):
    if i==ntwells or np.abs(trendWells[i,4] - 1.0) < 0.01:

        # make a mask that is all true
        mask = trendWells[:-1,0] > 1.0

        # set the current steel well to false
        if i < ntwells:
            mask[i] = False

        tX = trendWells[mask,0] - xmid
        tY = trendWells[mask,1] - ymid
        tH = trendWells[mask,2]

# using numpy recompute linear trend & compute residuals
# compute statistics about change removing each well has on estimated surface
# relative change in angle, slope & offset of surface
# write wells & residuals to file

        trendA = np.concatenate((tX[:,None],tY[:,None],np.ones((tX.shape[0],1))),axis=1)

        x,residues,rank,singulars = np.linalg.lstsq(trendA,tH)
        # residues is "squared Euclidian norm"

        cond = np.max(singulars)/np.min(singulars)
        # coefficient of determination
        rsq = 1.0 - residues/np.sum((tH - np.mean(tH))**2)
        # rsq = 1 - SS_err / SS_tot

        # write summary of fit as a line in file
        fh.write(','.join(str(z) for z in (trendNames[i],residues[0],cond,rank,rsq[0])) +',')
        fh.write('%.7e,%.7e,' % tuple(x[0:2])) # A and B
        fh.write('%.7e,' % (x[2] - x[0]*xmid - x[1]*ymid,)) # C corrected to original coords
        fh.write(','.join(str(z) for z in (np.sqrt(x[0]**2 + x[1]**2),
                                            np.arctan2(x[1],x[0])/np.pi*180.0))+'\n')

        tHpred = np.dot(trendA,x)
        outdata = np.concatenate((trendWells[mask,0:2],
                                  tHpred[:,None],(tHpred-tH)[:,None]),axis=1)

        # write all trend data to separate file for variogram analysis in Surfer
        np.savetxt('trend_results_'+trendNames[i]+'.dat',outdata,fmt='%.2f')

fh.close()
```

8.2.3. Python script `krig_plus_one.py`

The following Python script drives the GSLIB kriging program `kt3d.exe` during the kriging variance minimization process for adding one well (where it is called as a program). The script is also imported as a library in the remove-one and remove-two kriging variance reduction scripts. This script imports `shared_data.py` (line 8) to act as a container for storing shared variables, and uses the MS-DOS batch script `kt3d_driver.bat` (line 70) to manage directories and executables related to kt3d execution.

```
import os
import threading
from time import sleep
import numpy as np
from scipy.stats import rankdata
from math import ceil

import shared_data as sh

# this script is part of AP-111
# this python script adds an observation point at points
# in the model domain, each time calling KT3D.exe to krig the
# current network along with this additional observation.

# the kriging variance is read in and some statistics are saved
# for comparison and plotting.

def krig(ii,jj,xx,yy,nxx,nyy,x00,y00,dxx,dyy,base=False,addone=True):
    """ write KT3D input file, call kt3d.exe,
    and read in results for summarizing in global array."""

    # write kt3d parameter file
    d = '%03d_%03d' % (ii,jj)
    os.popen('mkdir ' + d)
    fname = os.path.join(d,'KT3D.PAR')
    fpar = open(fname,'w')

    fpar.write("""Parameters for KT3D\n*******************\n\nSTART OF PARAMETERS:
data.dat                           \\file with data
1   2   0   4   0                  \\   columns for X, Y, Z, var, sec var
-1.0e21   1.0e21                    \\   trimming limits
0                                   \\option: 0=grid, 1=cross, 2=jackknife
xvk.dat                             \\file with jackknife data
1   2   0   3    0                  \\   columns for X,Y,Z,vr and sec var
0                                   \\debugging level: 0,1,2,3
kt3d.dbg                            \\file for debugging output
kriged.out                            \\file for kriged output
%(nxx)d  %(x00)g  %(dxx)g              \\nx,xmn,xsiz
%(nyy)d  %(y00)g  %(dyy)g              \\ny,ymn,ysiz
1    0.5    1.0                     \\nz,zmn,zsiz
1    1      1                       \\x,y and z block discretization
0    44                             \\min, max data for kriging
44                                  \\max per octant (0-> not used)
40000.0  40000.0  1.0               \\maximum search radii
90.0   0.0   0.0                    \\angles for search ellipsoid
1     0.0                           \\0=SK,1=OK,2=non-st SK,3=exdrift
0 0 0 0 0 0 0 0 0                   \\drift: x,y,z,xx,yy,zz,xy,xz,zy
0                                   \\0, variable; 1, estimate trend
extdrift.dat                        \\gridded file with drift/mean
1                                   \\  column number in gridded file
1    3.0                            \\nst, nugget effect
3   40.0    90.0     0.0   0.0           \\it,cc,ang1,ang2,ang3
          7500.0 7500.0  10.0         \\a_hmax, a_hmin, a_vert\n""" % vars())
    fpar.close()

    # write data back to file, adding new point to end
    finput = open(os.path.join(d,'data.dat'),'w')
    finput.write('data for kriging data + 1 new well \n5 \nX \nY \nfwh \nres \ncasing \n')
    finput.write(sh.data)
    if base == False and addone == True:
        # add one data point (5 columns, tab delimited)
        finput.write('%8.1f\t%9.1f\t 100.00 \t1.00\t0 ' % (xx,yy))
    finput.close()

    # run KT3D via MS-DOS batch script
    output = os.popen('kt3d_driver.bat ' + d)
    for line in output:
        pass
    failure = output.close()
    if failure:
        print '*** KT3D failed ***',ii,jj
    else:
        print '(%03i,%03i) ' % (ii,jj),

    ## read in and calculate summary statistics on kriging variance
    # output from kt3d is a vector, reshape it into a matrix
    var = np.reshape(np.loadtxt(os.path.join(d,'kriged.out'),
                            skiprows=4,usecols=(1,)),(nyy,nxx))

    if base == True:
        sh.base_case_mod =  (var[mod_m[0]:mod_m[1], mod_n[0]:mod_n[1]])
        sh.base_case_wipp = (var[wipp_m[0]:wipp_m[1], wipp_n[0]:wipp_n[1]])
```

```
            # write base case kriging variance as a matrix for contouring
            np.savetxt('kriged_base_case_var_img.dat',sh.base_case_mod,fmt='%.3f')

        var_mod =  var[mod_m[0]:mod_m[1], mod_n[0]:mod_n[1]]
        var_wipp = var[wipp_m[0]:wipp_m[1], wipp_n[0]:wipp_n[1]]

        # use lock when writing to global variable to prevent thread collisions
        with threading.Lock():
            # compute statistics for sub-block corresponding to model domain,
            # masked by the cells which are inside the area of interest

            # change in aoi-wide standard deviation
            sh.mod_results[ii,jj,0] = (sh.base_case_mod[aoimask].std() -
                                       var_mod[aoimask].std())/
                                       sh.base_case_mod[aoimask].std()
            # change in aoi-wide average
            sh.mod_results[ii,jj,1] = (np.average(sh.base_case_mod[aoimask]) -
                                       np.average(var_mod[aoimask]))/
                                       np.average(sh.base_case_mod[aoimask])

            # change in aoi-wide median
            sh.mod_results[ii,jj,2] = (np.median(sh.base_case_mod[aoimask]) -
                                       np.median(var_mod[aoimask]))/
                                       np.median(sh.base_case_mod[aoimask])

            # correlation coefficient between cases
            sh.mod_results[ii,jj,3] = 1.0 - np.corrcoef(sh.base_case_mod[aoimask].flatten(),
                                                  var_mod[aoimask].flatten())[0,1]

            # change in max variance
            sh.mod_results[ii,jj,4] = (sh.base_case_mod[aoimask].max() -
                                     var_mod[aoimask].max())

            # same statistics for land-withdrawl boundary sub-block
            sh.wipp_results[ii,jj,0] = (sh.base_case_wipp[wippmask].std() -
                                        var_wipp[wippmask].std())/
                                        sh.base_case_wipp[wippmask].std()
            sh.wipp_results[ii,jj,1] = (np.average(sh.base_case_wipp[wippmask]) -
                                        np.average(var_wipp[wippmask]))/
                                        np.average(sh.base_case_wipp[wippmask])
            sh.wipp_results[ii,jj,2] = (np.median(sh.base_case_wipp[wippmask]) -
                                        np.median(var_wipp[wippmask]))/
                                        np.median(sh.base_case_wipp[wippmask])
            sh.wipp_results[ii,jj,3] = 1.0 - np.corrcoef(sh.base_case_wipp[wippmask].flatten(),
                                                   var_wipp[wippmask].flatten())[0,1]
            sh.wipp_results[ii,jj,4] = (sh.base_case_wipp[wippmask].max() -
                                     var_wipp[wippmask].max())


##################################################
# common stuff that is useful for adding or removing wells from the network
# included below, imported elsewhere.

# coordinates of wipp land-withdrawl boundary (McKenna 2004) AP-111, p 13
# averaged to be a N-S square for simple array addressing (it is nearly square anyway)
fh = open(r'..\common_data\wipp_boundary.dat')
wipp_file = [l.rstrip() for l in fh]
fh.close()
coords = []

# corners listed in file in order :NE,SE,SW,NW,NE (NE repeated to close loop)
for line in wipp_file[:-1]:
    coords.append([float(z) for z in line.split()])

wipp_x = ((coords[2][0] + coords[1][0])/2.0, (coords[0][0] + coords[3][0])/2.0)
wipp_y = ((coords[2][1] + coords[3][1])/2.0, (coords[0][1] + coords[1][1])/2.0)

# output grid (the model domain) specifications, number of elements reduced by
# a multiplier to make the run-time feasible
mult = 4.0
fh = open(r'..\common_data\model_domain_specs.dat')
moddata = [l.rstrip() for l in fh]
fh.close()

# always rounds up when determining number of elements
nx,ny = [int(ceil(float(z)/mult)) for z in moddata[0].split()]
x0,y0 = [float(z) for z in moddata[1].split()]
x1,y1 = [float(z) for z in moddata[2].split()]
dx,dy = [float(z)*mult for z in moddata[3].split()]
```

```
# make vectors
x = np.array([x0 + i*dx for i in range(nx)])
y = np.array([y0 + i*dy for i in range(ny)])

# check that everything adds up correctly to fill the domain
assert abs(x[-1] - x1) < dx, abs(y[-1] - y1) < dy

# saved from matlab, this array has 1.00E+0 for cells inside area of interest and
# 0.00E+0 for cells outside
size = mult*100
intmask = np.loadtxt(r'..\common_data\model_cells_%(size)d_inside_totalbdry.dat' % vars())
aoimask = intmask > 0.99

print 'model domain',intmask.sum(),'true out of',np.size(intmask)

# make outerproduct matricies for Matlab plotting
X = np.outer(np.ones(ny),x)
Y = np.outer(y,np.ones(nx))

# indicies for this grid corresponding to the wipp lwb
wipp_n = (int((wipp_x[0] - x0)/dx), int((wipp_x[1] - x0)/dx))
wipp_m = (int((wipp_y[0] - y0)/dy), int((wipp_y[1] - y0)/dy))
sh.base_case_wipp = np.zeros((wipp_m[1]-wipp_m[0], wipp_n[1]-wipp_n[0]))

wippmask = aoimask[wipp_m[0]:wipp_m[1], wipp_n[0]:wipp_n[1]]
wippcheck = np.zeros(np.shape(wippmask))
wippcheck[wippmask] = 1.0
print 'WIPP boundary',wippcheck.sum(),'true out of',np.size(wippmask)

# indicies corresponding to the model domain
mod_n = (int((x0 - x0)/dx), int((x0 - x0)/dx) + nx)
mod_m = (int((y0 - y0)/dy), int((y0 - y0)/dy) + ny)
sh.base_case_mod = np.zeros((mod_m[1]-mod_m[0], mod_n[1]-mod_n[0]))

format = '%.5e'

# read observed data as one long string
fh = open(r'..\common_data\2007_well_data.dat','r')
sh.data = fh.read().strip() # strip off ending / beginning whitespace
fh.close()

# make sure data file ends in a newline
if sh.data[-1] != '\n':
    sh.data = sh.data + '\n'


# ####################################################
# only run from here below if called as a program (rather than
# imported as a library)

if __name__ == '__main__':

    # global arrays to write results into
    sh.mod_results =  np.ones((ny,nx,5))
    sh.wipp_results = np.ones((ny,nx,5))

    krig(0,0,0.0,0.0,nx,ny,x0,y0,dx,dy,base=True)
    f = open('base_stats.out','w')
    # mean, median, std dev
    f.write(' '.join([str(x) for x in (sh.mod_results[0,0,1],sh.mod_results[0,0,2],
                                       sh.mod_results[0,0,0],'\n')]))
    f.write(' '.join([str(x) for x in (sh.wipp_results[0,0,1],sh.wipp_results[0,0,2],
                                       sh.wipp_results[0,0,0],'\n')]))
    f.close()

    for j in xrange(nx):
        print ' '
        for i in xrange(ny):
            # don't do calculation if point is outside area of interest
            if aoimask[i,j] == True:
                while True:
                    # limit the number of threads (8 processors)
                    if threading.activeCount() <= 8:
                        threading.Thread(target=krig,
                                         args=(i,j,X[i,j],Y[i,j],nx,ny,
                                               x0,y0,dx,dy,False)).start()
                        break
                    else:
                        sleep(0.015)

    # wait for all the worker threads to finish before writing output
    while True:
        if threading.activeCount() > 1:
            sleep(1.0)
        else:
            break

    # write output in Matlab-friendly matrix format
    names = ['stdev','mean','median','corrcoef','max']

    for i,name in enumerate(names):
        print 'writing', name,i
        np.savetxt('addone_mod_results_' + name + '.dat',  sh.mod_results[:,:,i],fmt=format)
        np.savetxt('addone_wipp_results_' + name + '.dat', sh.wipp_results[:,:,i],fmt=format)

    np.savetxt('X.dat',X,fmt='%.1f')
    np.savetxt('Y.dat',Y,fmt='%.1f')
```

8.2.4. Python script `shared_data.py`

The following short Python script is used to allow data to be saved and shared in a common module (see line 8 of `krig_plus_one.py`, line 7 of `krig_remove_one_steel.py`).

```
""" this is just for putting global data in, so
it can be seen between modules"""

pass
```

8.2.5. MS-DOS batch script `kt3d_driver.bat`

The following MS-DOS batch script is called by the Python scripts (see line 70 of `krig_plus_one.py`) that drive kt3d, and is actually responsible for calling `kt3d.exe`, first creating a temporary directory and copying the executable and input files into that directory. This allows the scripts to be threaded and have more than one copy of kt3d running at a time.

```
echo off
rem kriging plus one driver script
rem this batch file copies the executable into a working directory
rem runs it (it expects a standard input filename KT3D.PAR)
rem and deletes the executable
copy /B /Y KT3D.EXE %1
copy /A /Y response %1
chdir %1
KT3D.EXE < response
del /F KT3D.EXE response
chdir ..\
```

8.2.6. MATLAB script `generate_model_cell_masks.m`

The following MATLAB script generates ASCII matrix files representing the model grid, indicating whether each cell is inside or outside the active MODFLOW region (relying on the MATLAB built-in command `inpolygon()` to do most of the work). The text files generated by this script are read in by krig_plus_one.py (line 176 of Section 8.2.1).

```
% this Matlab script exports arrays representing whether a cell
% from the model grid is inside or outside the area of interest
% for use in python scripts

clear
totalbdry = load('..\common_data\total_boundary.dat');

% model grid (for 100x100 elements - the base size)
grid = load('..\common_data\model_domain_specs.dat');
nx = grid(1,1);     ny = grid(1,2);
xmin = grid(2,1);  ymin = grid(2,2);
```

```
xmax = grid(3,1);  ymax = grid(3,2);
dx = grid(4,1);    dy = grid(4,2);
clear grid;

for mult = 1:6
    [X,Y] = meshgrid(xmin:mult*dx:xmax, ymin:mult*dy:ymax);
    % create logical mask
    INSIDE = inpolygon(X,Y,totalbdry(:,1),totalbdry(:,2));
    % convert to real to save (Matlab can't write logical values to ASCII)
    inside = +INSIDE;
    filename = ['model_cells_',sprintf('%d',100*mult),'_inside_totalbdry.dat'];
    save('-ASCII',filename,'inside')
end
```

8.2.7. Python script `krig_remove_one_steel.py`

The following Python script imports the main `krig()` routine from `krig_plus_one.py` (see Section 8.2.1), but instead of adding more locations and re-kriging, a single steel-cased well is removed from the existing dataset and the remaining set is re-kriged.

```
import sys
import numpy as np

# most of the functionality is defined in krig_plus_one; import to re-use code
sys.path.append(r'..\kriging_add_well')
import krig_plus_one as k
import shared_data as shared

# this python script removes an observation point, each time calling
#  KT3D.exe to krig the remaining network

fh = open(r'..\common_data\2007_well_names.dat','r')
names = [line.rstrip() for line in fh]
fh.close()

# fifth column is casing type (1=steel, 0=fiberglass)
fh = open(r'..\common_data\2007_well_data.dat','r')
wells = [line.rstrip().split() for line in fh]
fh.close()

shared.mod_results = np.zeros((len(wells)+1,1,5))
shared.wipp_results = np.zeros((len(wells)+1,1,5))

# base case, for computing percentage change
shared.data = '\n'.join('\t'.join(w) for w in wells)
print 'base_case',
k.krig(len(wells),0,0.0,0.0,k.knx,k.kny,k.kx0,k.ky0,k.dx,k.dy,base=True,addone=False)

fm = open('model_results_one.dat','w')
fw = open('wipp_results_one.dat','w')

stnames = []

# remove one steel cased well
for i,well in enumerate(wells):
    if int(well[4]) == 1:
        stnames.append(names[i])

        # make a copy and delete current well from copy
        cwells = list(wells)
        del cwells[i]
        shared.data = '\n'.join('\t'.join(w) for w in cwells)

        print names[i],
        k.krig(i,0,0.0,0.0,k.knx,k.kny,k.kx0,k.ky0,k.dx,k.dy,base=False,addone=False)
        fm.write(', '.join([str(x) for x in shared.mod_results[i,0,:]]))
        fm.write(', ' + names[i] + '\n')
        fw.write(', '.join([str(x) for x in shared.wipp_results[i,0,:]]))
        fw.write(', ' + names[i] + '\n')

fw.close()
fm.close()
```

8.2.8. Python script `krig_remove_two_steel.py`

The following Python script is analogous to that in Section 8.2.6, except a list of most-likely-to-be-removed steel-cased wells are first removed before removing a second steel-cased wells and re-kriging the results. The main functionality of this routine is imported from the Python script `krig_plus_one.py` (see Section 8.2.1).

```
import sys
import numpy as np

# most of the functionality is defined in krig_plus_one; import to re-use code
sys.path.append(r'..\kriging_add_well')
import krig_plus_one as k
import shared_data as shared

# this python script removes an observation point, each time calling
#  KT3D.exe to krig the remaining network

fh = open(r'..\common_data\2007_well_names.dat','r')
names = [line.rstrip() for line in fh]
fh.close()

# fourth column is casing type (1=steel, 0=fiberglass)
fh = open(r'..\common_data\2007_well_data.dat','r')
wells = [line.rstrip().split() for line in fh]
fh.close()

# perform the "remove one well" analysis for the networks modulo the
# following "likely to not be replaced" wells
firstWell = ['WIPP-25','WIPP-13','H-12','H-7b1']

shared.mod_results = np.zeros((len(wells)+1,1,5))
shared.wipp_results = np.zeros((len(wells)+1,1,5))

# same base-case used throughout to allow comparison
shared.data = '\n'.join('\t'.join(w) for w in wells)
print 'base case',
k.krig(len(wells),0,0.0,0.0,k.knx,k.kny,k.kx0,k.ky0,k.dx,k.dy,base=True,addone=False)

fm = open('model_results_two.dat','w')
fw = open('wipp_results_two.dat','w')

stnames = []

# remove one of the first steel cased wells
for first in firstWell:
    # find index in list
    ifirst = names.index(first)

    # make a local copy of well list
    cwells = list(wells)

    # remove first steel well
    del cwells[ifirst]

    # cycle through remaining steel wells
    for i,well in enumerate(wells):
        if int(well[4]) == 1:
            if ifirst != i:

                stnames.append(names[i])

                # make another copy of list, removing second well
                ccwells = list(cwells)
                del ccwells[i]

                # collapse back into string
                shared.data = '\n'.join('\t'.join(w) for w in ccwells)

                print names[ifirst],names[i],
                k.krig(i,0,0.0,0.0,k.knx,k.kny,k.kx0,k.ky0,k.dx,k.dy,base=False,addone=False)
                fm.write(', '.join([str(x) for x in shared.mod_results[i,0,:]])+',')
                fm.write(', '.join((names[ifirst],names[i])) + '\n')
                fw.write(', '.join([str(x) for x in shared.wipp_results[i,0,:]])+',')
                fw.write(', '.join((names[ifirst],names[i])) + '\n')

fw.close()
fm.close()
```

8.3. Triangle Metric Maximization Scripts

The following scripts were used in the local gradient estimation or triangle metric maximization portion of the analysis (see Section 3.0).

8.3.1. MATLAB script `triangles_add_one.m`

The following MATLAB script computes and plots figures related to the triangle interior angle ratio metric. Each location in the model domain is added to the current network and the statistics are re-computed.

```
clear
% this Matlab script asses the benefit of adding a new well, where
% locations on the MODFLOW model grid are used as potential locations.
% This approach is geometry-based only;

% the ratio min(angle)/max(angle) is used as a metric for the "quality" of
% a triangle.  More equilateral (ratio=1) triangles would be better.

addpath '..\common_programs\'
wells = load('..\common_data\2007_well_data_for_triangles.dat');
margin = load('..\common_data\composite_23_margin.dat');
noflow = load('..\common_data\no_flow_boundary.dat');
totalbdry = load('..\common_data\total_boundary.dat');
WIPP = load('..\common_data\wipp_boundary.dat');

% default qhull options, except QbB, which scales domain to unit box (since
% UTM coordinates are numerically large and can lead to significant
% roundoff error)
triopts = {'Qt','QbB','Qc','Qz'};

xt=wells(:,1);
yt=wells(:,2);
nw = size(xt,1);

% model grid
grid = load('..\common_data\model_domain_specs.dat');
nx = grid(1,1);     ny = grid(1,2);
xmin = grid(2,1);  ymin = grid(2,2);
dx = grid(4,1);    dy = grid(4,2);
clear grid;

[X,Y] = meshgrid(linspace(xmin,xmin+nx*dx,nx), ...
    linspace(ymin,ymin+ny*dy,ny));
D = numel(X);

INSIDE = reshape(inpolygon(X,Y,totalbdry(:,1),totalbdry(:,2)),D,1);
INSIDE(D+1) = 1;

% observation points in a long x,y vector
Z(1:D,1:2) = [reshape(X,D,1),reshape(Y,D,1)];

Q = zeros(D,5);
numt = zeros(D,1);

for jj=1:D+1

    % only points between no-flow and h2/h3 halite boundaries are
    % candidate sites, skip the others
    if INSIDE(jj)

        if jj==D+1
            x=xt;
            y=yt;
        else
            x = [xt; Z(jj,1)];
            y = [yt; Z(jj,2)];
        end

        tri = delaunay(x,y,triopts);

        nt = size(tri,1);
        geom = zeros(size(tri,1),7);  % 3 sides, 3 angles, area, # pts inside

        % calculate geometric things related to triangles
        % lengths from Pythagorean theorem
        % angles from cosine law
        % area from Matlab built-in fcn

        % length of side a (2->3)
        geom(1:nt,1) = sqrt((x(tri(:,3)) - x(tri(:,2))).^2 + ...
            (y(tri(:,3)) - y(tri(:,2))).^2);
        % length of side b (3->1)
        geom(1:nt,2) = sqrt((x(tri(:,1)) - x(tri(:,3))).^2 + ...
            (y(tri(:,1)) - y(tri(:,3))).^2);
        % length of side c (1->2)
        geom(1:nt,3) = sqrt((x(tri(:,2)) - x(tri(:,1))).^2 + ...
            (y(tri(:,2)) - y(tri(:,1))).^2);

        % angle 1 between sides b & c in radians
        geom(1:nt,4) = acos((sum(geom(:,2:3).^2,2) - geom(:,1).^2)./ ...
            (2.0*prod(geom(:,2:3),2)));
        % angle 2 between sides a & c in radians
        geom(1:nt,5) = acos((sum(geom(:,1:2:3).^2,2) - geom(:,2).^2)./ ...
            (2.0*prod(geom(:,1:2:3),2)));
        % angle 3 between sides b & a in radians
        geom(1:nt,6) = acos((sum(geom(:,1:2).^2,2) - geom(:,3).^2)./ ...
            (2.0*prod(geom(:,1:2),2)));

        % area of triangle - use MATLAB built-in function
        geom(1:nt,7) = polyarea(x(tri(:,1:3)),y(tri(:,1:3)),2);

        % compute triangle comparison criterias
        ang_ratio = min(geom(:,4:6),[],2)./max(geom(:,4:6),[],2);

        % area-weighted angle ratio
        Q(jj,1) = sum(ang_ratio(:).*geom(:,7))/(nt*sum(geom(1:nt,7)));

        % non-weighted angle ratio average
        Q(jj,2) = sum(ang_ratio(:))/nt;

        % area-weighted angle ratio median
        Q(jj,3) = median(ang_ratio(:).*geom(:,7))/sum(geom(1:nt,7));
        numt(jj) = nt;

        % mean triangle area
        Q(jj,4) = sum(geom(:,7))/nt;

        % median triangle area
        Q(jj,5) = median(geom(:,7));
    end
end

% reset values outside area of interest to not-a-number
% so they are not plotted.

% save results for use in final 3-way combination of results
out = reshape(squeeze((Q(1:end-1,1)-Q(D+1,1))./Q(D+1,1)),ny,nx);
out(~INSIDE(1:end-1)) = -999;
save('triangles_add_one_mean.dat','out','-ASCII');
out = reshape(squeeze((Q(1:end-1,3)-Q(D+1,3))./Q(D+1,3)),ny,nx);
out(~INSIDE(1:end-1)) = -999;
save('triangles_add_one_median.dat','out','-ASCII');
clear out;

Q(~INSIDE(1:end-1),1:end) = NaN;
numt(~INSIDE(1:end-1)) = NaN;

scrnsz = get(0,'ScreenSize');

%% plot results
figure()
ceil(max(numt)-min(numt))
contourf(X,Y,reshape(numt(1:D),ny,nx),3);
colorbar;
daspect([1,1,1]);
hold on
tri = delaunay(xt,yt,triopts);
triplot(tri,xt,yt,'g','LineWidth',0.5);
plot(xt,yt,'or','LineWidth',2)
```

```
140   plot(margin(:,1),margin(:,2),'-m','LineWidth',2)
      plot(noflow(:,1),noflow(:,2),'--k','LineWidth',2)
142   plot(WIPP(:,1),WIPP(:,2),'-k','LineWidth',1.5)
      xlabel('NAD27 UTM x Zone 13 [m]','FontSize',14)
144   ylabel('NAD27 UTM y Zone 13 [m]','FontSize',14)
      title('Total number of triangles in network')
146   % measured from inside of figure window (no borders or toolbars included)
      % position -> [left, bottom, width, height]
148   set(gcf,'Position',[10,50,(scrnsz(4)-120)*0.95,scrnsz(4)-120])
      % make file printed at screen size, rather than bad default
150   set(gcf, 'PaperPositionMode', 'auto');
      print('-dmeta','triangles_add1_total_number.emf')
152
      type2 = {'scaled_mean_angle','unscaled_mean_angle','median_angle','mean_area','median_area'};
154   cblab = {'%\Delta area-weighted mean angle ratio', ...
               '%\Delta mean angle ratio', ...
156            '%\Delta area-weighted median angle ratio', ...
               '%\Delta mean triangle area', '%\Delta median triangle area'};
158   txt = {'a','','b','','',''};
      white = 0.0;
160   for ii=1:5
          clf;
162       data = squeeze((Q(1:D,ii) - Q(D+1,ii))./Q(D+1,ii));
          contourf(X,Y,reshape(data,ny,nx),20);
164       colormap(redwhitemap(data,white));
          cb = colorbar;
166       set(get(cb,'ylabel'),'string',cblab{ii},'FontSize',14);
          daspect([1,1,1]);
168       hold on
          tri = delaunay(xt,yt,triopts);
170       triplot(tri,xt,yt,'g','LineWidth',2)
          plot(xt,yt,'or','LineWidth',2)
172       plot(margin(:,1),margin(:,2),'-m','LineWidth',2)
          plot(noflow(:,1),noflow(:,2),'--k','LineWidth',2)
174       plot(WIPP(:,1),WIPP(:,2),'-k','LineWidth',1.5)
          xlabel('NAD27 UTM X Zone 13 [m]','FontSize',14)
176       ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',14)
          set(gcf,'Position',[10,50,(scrnsz(4)-120)*0.85,scrnsz(4)-120])
178       set(gcf, 'PaperPositionMode', 'auto');
          text(6.05E5,3.594E6,txt{ii},'FontSize',24,'FontWeight','bold');
180       brighten(0.5);
          print('-dmeta', ['triangles_add1_',type2{ii},'.emf'])
182   end
```

8.3.2. MATLAB function `redwhitemap.m`

The following MATLAB script is a function for computing the red-white-blue color maps used in the plotting of figures in this section; see line 161 of `triangles_add_one.m` in section 8.3.1.

```
function [ map ] = redwhitemap( data, white )
%REDWHITEMAP create a specific color map from
% blue = min to red=max with wite at a specific number

mindata = min(min(data));
maxdata = max(max(data));

nlevels = 64;

map = zeros(nlevels,3);

if mindata >= white
    % ** all data will be colored red (no blue or white)

    mindata = white;

    % compute color at midpoint of each bin, rather than at max or min
    xn = mindata + (0.5:1.0:(nlevels-0.5))*(maxdata - mindata)/nlevels;

    % white -> red
    map(xn >= white,1) = 1;    % red
    map(xn >= white,2) = (maxdata - xn(xn >= white))/(maxdata - white);  % green
    map(xn >= white,3) = map(xn >= white,2);  % blue


elseif maxdata <= white
```

```
    % ** all data will be colored blue (no red or white)

    maxdata = white;

    % compute color at midpoint of each bin, rather than at max or min
    xn = mindata + (0.5:1.0:(nlevels-0.5))*(maxdata - mindata)/nlevels;

    % blue -> white
    map(xn < white,1) = (xn(xn < white) - mindata)/(white - mindata);  % red channel
    map(xn < white,2) = map(xn < white,1);  % green
    map(xn < white,3) = 1;       % blue

else
    % ** data will be blue, red, and white

    % compute color at midpoint of each bin, rather than at max or min
    xn = mindata + (0.5:1.0:(nlevels-0.5))*(maxdata - mindata)/nlevels;

    % blue -> white
    map(xn < white,1) = (xn(xn < white) - mindata)/(white - mindata);  % red channel
    map(xn < white,2) = map(xn < white,1);  % green
    map(xn < white,3) = 1;       % blue

    % white -> red
    map(xn >= white,1) = 1;    % red
    map(xn >= white,2) = (maxdata - xn(xn >= white))/(maxdata - white);  % green
    map(xn >= white,3) = map(xn >= white,2);  % blue

end
end
```

8.3.3. MATLAB script `triangles_remove_one.m`

The following MATLAB script computes and plots the triangle interior angle ratio metric after individually removing each of the steel-cased wells from the network.

```
clear
% This matlab script looks at the effects that removing one of the
% steel-cased (without replacement) would have on the estimation of the
% gradient, using linear interpolation across Delauny triangles as the
% estimator.

% Load data
addpath '..\common_programs\';

% well datat (x,y,fwh,res,casing type)
wells = load('..\common_data\2007_well_data_for_triangles.dat');
names = textread('..\common_data\2007_well_names_for_triangles.dat','%s');

% the majority of this analysis should be done without SNL-6 and SNL-15,
% but some figures in text use them for comparison
RHmask = wells(:,4) > -990; % exclude SNL-6 and SNL-15
wells = wells(RHmask,:);
names = names(RHmask,:);
% RHmask = ones(size(wells,1),1);

margin = load('..\common_data\composite_23_margin.dat');
noflow = load('..\common_data\no_flow_boundary.dat');
totalbdry = load('..\common_data\total_boundary.dat');
wipp = load('..\common_data\wipp_boundary.dat');

nearwipp = [min(wipp(:,1))-750.0,max(wipp(:,1))+750.0, ...
            min(wipp(:,2))-750.0,max(wipp(:,2))+750.0];

% wells that make a convex hull around the dataset of all wells
hull = convhull(wells(:,1),wells(:,2));

steelwells = wells(wells(:,5)==1,1:3);  % fifth column indicates casing type
fiberwells = wells(wells(:,5)==0,1:3);
stnames =    {names{wells(:,5)==1}};

% wells on the hull that are also steel-cased
j=1;
for i=1:size(steelwells,1)
    for k=1:size(hull,1)
```

```
            if sqrt((steelwells(i,1) - wells(hull(k),1))^2 + ...
                    (steelwells(i,2) - wells(hull(k),2))^2) < 1
                sthull(j) = i;
                j=j+1;
            end
        end
    end

    xt=wells(:,1);
    yt=wells(:,2);
    ht=wells(:,3);
    nw = size(xt,1);
    nst = size(steelwells,1);
    Q = zeros(nst+1,3);

    % calculation grid (not MODFLOW grid) is minimal grid which includes
    % convex hull around data
    xmin = min(xt);  ymin = min(yt);
    xmax = max(xt);  ymax = max(yt);
    dx = 100.0;       dy = 100.0;  % note: using 100x100 is slow.

    [X,Y] = meshgrid(xmin:dx:xmax, ymin:dy:ymax);
    nx = size(X,2);
    ny = size(X,1);

    INSIDE = inpolygon(X,Y,totalbdry(:,1),totalbdry(:,2));

    npts = numel(X);

    % direction and magnitude of gradient in each cell, for
    % scenario of removing each steel-casing well + base case

    GRAD = zeros(npts,nst+1,2);

    % effects of removing one steel-casing well + base case for comparison
    for jj=1:nst+1

        if jj < nst+1
            % set of x,y,h without steel casing well jj
            x = [fiberwells(:,1);steelwells(1:jj-1,1);steelwells(jj+1:nst,1)];
            y = [fiberwells(:,2);steelwells(1:jj-1,2);steelwells(jj+1:nst,2)];
            h = [fiberwells(:,3);steelwells(1:jj-1,3);steelwells(jj+1:nst,3)];
        else
            x= xt;
            y=yt;
            h=ht;
        end

        tri = delaunay(x,y);
        nt = size(tri,1);

        D = zeros(size(tri,1),1);
        coeff = zeros(size(tri,1),4);
        grad =  zeros(size(tri,1),2); % angle and magnitide of hydraulic gradient
        geom =  zeros(size(tri,1),8);  % 3 sides, 3 angles, area, # pts inside

        % compute equation for line through 3 points
        % value of determinant used in denominator of Cramer's rule
        D(1:nt) = x(tri(:,1)).*y(tri(:,2)) + x(tri(:,2)).*y(tri(:,3)) + ...
            y(tri(:,1)).*x(tri(:,3)) - x(tri(:,3)).*y(tri(:,2)) - ...
            x(tri(:,1)).*y(tri(:,3)) - x(tri(:,2)).*y(tri(:,1));

        % a (coefficient on x)
        coeff(1:nt,1) = (h(tri(:,1)).*y(tri(:,2)) + y(tri(:,1)).*h(tri(:,3)) + ...
            h(tri(:,2)).*y(tri(:,3)) - h(tri(:,3)).*y(tri(:,2)) - ...
            h(tri(:,2)).*y(tri(:,1)) - h(tri(:,1)).*y(tri(:,3)))./D;

        % b (coefficient on y)
        coeff(1:nt,2) = (x(tri(:,1)).*h(tri(:,2)) + h(tri(:,1)).*x(tri(:,3)) + ...
            x(tri(:,2)).*h(tri(:,3)) - x(tri(:,3)).*h(tri(:,2)) - ...
            x(tri(:,2)).*h(tri(:,1)) - x(tri(:,1)).*h(tri(:,3)))./D;

        % c (constant coefficient)
        coeff(1:nt,3) = (x(tri(:,1)).*y(tri(:,2)).*h(tri(:,3)) + ...
            y(tri(:,1)).*h(tri(:,2)).*x(tri(:,3)) + ...
            x(tri(:,2)).*y(tri(:,3)).*h(tri(:,1)) - ...
            x(tri(:,3)).*y(tri(:,2)).*h(tri(:,1)) - ...
            x(tri(:,2)).*y(tri(:,1)).*h(tri(:,3)) - ...
            x(tri(:,1)).*y(tri(:,3)).*h(tri(:,2)))./D;

        % compute angle and magnitude of hydraulic gradient
    grad(1:nt,1) = atan2(coeff(:,2),coeff(:,1));
    grad(1:nt,2) = sqrt(sum(coeff(:,1:2).^2,2));

    % map results from "vector" triangles to "raster" grid
    for kk=1:nt
        % result is a logical vector, indicating if the cell is in (T) or
        % out (F) side this current triangle
        IN = reshape(inpolygon(X,Y,x(tri(kk,1:3)),y(tri(kk,1:3))),npts,1);

        % sum(IN) = number of cells inside the triangle
        % ones(sum(IN))*kk = column vector of the counter kk
        % coeff(...,1:2) = x & y gradient repeated for every cell inside
        %                  that triangle, copied to correct locations in GRAD

        GRAD(IN,jj,1:2) = coeff(ones(sum(IN),1)*kk,1:2);
    end

    % calculate geometric things related to triangles
    % lengths from Pythagorean theorem
    % angles from cosine law
    % area from Matlab built-in fcn

    % length of side a (2->3)
    geom(1:nt,1) = sqrt((x(tri(:,3)) - x(tri(:,2))).^2 + ...
        (y(tri(:,3)) - y(tri(:,2))).^2);
    % length of side b (3->1)
    geom(1:nt,2) = sqrt((x(tri(:,1)) - x(tri(:,3))).^2 + ...
        (y(tri(:,1)) - y(tri(:,3))).^2);
    % length of side c (1->2)
    geom(1:nt,3) = sqrt((x(tri(:,2)) - x(tri(:,1))).^2 + ...
        (y(tri(:,2)) - y(tri(:,1))).^2);

    % angle 1 between sides b & c in radians
    geom(1:nt,4) = acos((sum(geom(:,2:3).^2,2) - geom(:,1).^2)./ ...
        (2.0*prod(geom(:,2:3),2)));
    % angle 2 between sides a & c in radians
    geom(1:nt,5) = acos((sum(geom(:,1:2:3).^2,2) - geom(:,2).^2)./ ...
        (2.0*prod(geom(:,1:2:3),2)));
    % angle 3 between sides b & a in radians
    geom(1:nt,6) = acos((sum(geom(:,1:2).^2,2) - geom(:,3).^2)./ ...
        (2.0*prod(geom(:,1:2),2)));

    % area of triangle - use MATLAB built-in function
    geom(1:nt,7) = polyarea(x(tri(:,1:3)),y(tri(:,1:3)),2);

    % compute goodness triangle criteria
    ang_ratio = min(geom(:,4:6),[],2)./max(geom(:,4:6),[],2);

    % area-weighted mean angle ratio
    Q(jj,1) = sum(ang_ratio(:).*geom(:,7))/(nt*sum(geom(1:nt,7)));

    % area-weighted median angle ratio
    Q(jj,2) = median(ang_ratio(:).*geom(:,7))/sum(geom(1:nt,7));

    % median triangle area
    Q(jj,3) = median(geom(:,7));

end

if sum(RHmask) == 44
    % plot figures showing distribution of metrics for 2007 Culebra network
    figure();
    trisurf(tri,x,y,ones(size(x),1),log10(geom(:,7)));
    view(2)
    axis('image')
    xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
    ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
    cb = colorbar;
    set(get(cb,'ylabel'),'string','log_{10}(triangle area [m^2])','FontSize',12);
    text(6.06E5,3.593E6,'a','FontSize',20,'FontWeight','bold')
    brighten(0.25);
    print('-dmeta','triangles_2007_network_log10_area.emf')

    trisurf(tri,x,y,ones(size(x),1),ang_ratio);
    view(2)
    axis('image')
    xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
    ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
    cb = colorbar;
    set(get(cb,'ylabel'),'string','interior angle ratio','FontSize',12);
    text(6.06E5,3.593E6,'b','FontSize',20,'FontWeight','bold')
    brighten(0.25);
    print('-dmeta','triangles_2007_network_angratio_area.emf')

    trisurf(tri,x,y,ones(size(x),1),log10(grad(:,2)));
    view(2)
    axis('image')
    xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
    ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
    cb = colorbar;
    set(get(cb,'ylabel'),'string','log_{10}(gradient magnitude)','FontSize',12);
    text(6.06E5,3.593E6,'c','FontSize',20,'FontWeight','bold')
    brighten(0.25);
    print('-dmeta','triangles_2007_network_gradmag_area.emf')

    figure()
    subplot(131)
    plot(log10(geom(:,7)),ang_ratio,'o')
    xlabel('log_{10}(triangle area [m^2])')
    ylabel('interior angle ratio')
    axis([5,8,0,1])
    text(7.5,0.9,'a','FontSize',22,'FontWeight','bold')
    subplot(132)
    plot(log10(grad(:,2)),ang_ratio,'+')
    xlabel('log_{10}(gradient magnitude)')
    ylabel('interior angle ratio')
    axis([-5,0,0,1])
    text(-0.6,0.9,'b','FontSize',22,'FontWeight','bold')
    subplot(133)
    plot(log10(grad(:,2)),log10(geom(:,7)),'*')
    ylabel('log_{10}(triangle area [m^2])')
    xlabel('log_{10}(gradient magnitude)')
    axis([-5,0,5,8])
    text(-0.6,7.7,'c','FontSize',22,'FontWeight','bold')
    print('-dmeta','scatter_plots_2007_network_metrics.emf')

    figure()
    triplot(tri,x,y)
    axis('image')
    hold on
    plot(fiberwells(:,1),fiberwells(:,2),'bs', ...
         'MarkerSize',6,'MarkerFaceColor','b');
    plot([steelwells(1:i-1,1);steelwells(i+1:nst,1)], ...
          [steelwells(1:i-1,2);steelwells(i+1:nst,2)],'ro', ...
         'MarkerSize',6,'MarkerFaceColor','r');
    plot(margin(:,1),margin(:,2),'-m','LineWidth',2);
    plot(noflow(:,1),noflow(:,2),'--k','LineWidth',2);
    plot(wipp(:,1),wipp(:,2),'-k','LineWidth',2);
    midx = sum(x(tri(:,1:3)),2)/3.0;
    midy = sum(y(tri(:,1:3)),2)/3.0;
    quiver(midx,midy,-coeff(:,1),-coeff(:,2),2.5,'k','LineWidth',2)
    xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
    ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
    text(6.06E5,3.594E6,'a','FontSize',22,'FontWeight','bold')
    print('-dmeta','vector_plots_2007_network.emf')
else
    % plot figures showing distribution of metrics for 2007 Culebra network (no SNL-6 or SNL-15)
    figure();
    trisurf(tri,x,y,ones(size(x),1),log10(geom(:,7)));
    view(2)
    axis('image')
    xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
    ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
    cb = colorbar;
    set(get(cb,'ylabel'),'string','log_{10}(triangle area [m^2])','FontSize',12);
    text(6.06E5,3.593E6,'a','FontSize',20,'FontWeight','bold')
    brighten(0.25);
    print('-dmeta','triangles_noSNL15-6_network_log10_area.emf')

    trisurf(tri,x,y,ones(size(x),1),ang_ratio);
    view(2)
    axis('image')
    xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
    ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
    cb = colorbar;
    set(get(cb,'ylabel'),'string','interior angle ratio','FontSize',12);
    text(6.06E5,3.593E6,'b','FontSize',20,'FontWeight','bold')
    brighten(0.25);
    print('-dmeta','triangles_noSNL15-6_network_angratio_area.emf')

    trisurf(tri,x,y,ones(size(x),1),log10(grad(:,2)));
    view(2)
```

```
        axis('image')
        xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
        ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
        cb = colorbar;
        set(get(cb,'ylabel'),'string','log_{10}(gradient magnitude)','FontSize',12);
        text(6.06E5,3.593E6,'c','FontSize',20,'FontWeight','bold')
        brighten(0.25);
        print('-dmeta','triangles_noSNL15-6_network_gradmag_area.emf')

        figure()
        subplot(131)
        plot(log10(geom(:,7)),ang_ratio,'o')
        xlabel('log_{10}(triangle area [m^2])')
        ylabel('interior angle ratio')
        axis([5,8,0,1])
        text(7.5,0.9,'a','FontSize',22,'FontWeight','bold')
        subplot(132)
        plot(log10(grad(:,2)),ang_ratio,'+')
        xlabel('log_{10}(gradient magnitude)')
        ylabel('interior angle ratio')
        axis([-5,0,0,1])
        text(-0.6,0.9,'b','FontSize',22,'FontWeight','bold')
        subplot(133)
        plot(log10(grad(:,2)),log10(geom(:,7)),'*')
        ylabel('log_{10}(triangle area [m^2])')
        xlabel('log_{10}(gradient magnitude)')
        axis([-5,0,5,8])
        text(-0.6,7.7,'c','FontSize',22,'FontWeight','bold')
        print('-dmeta','scatter_plots_noSNL15-6_network_metrics.emf')

        figure()
        triplot(tri,x,y)
        axis('image')
        hold on
        plot(fiberwells(:,1),fiberwells(:,2),'bs', ...
            'MarkerSize',6,'MarkerFaceColor','b');
        plot([steelwells(1:i-1,1);steelwells(i+1:nst,1)], ...
             [steelwells(1:i-1,2);steelwells(i+1:nst,2)],'ro', ...
            'MarkerSize',6,'MarkerFaceColor','r');
        plot(margin(:,1),margin(:,2),'-m','LineWidth',2);
        plot(noflow(:,1),noflow(:,2),'--k','LineWidth',2);
        plot(wipp(:,1),wipp(:,2),'-k','LineWidth',2);
        midx = sum(x(tri(:,1:3)),2)/3.0;
        midy = sum(y(tri(:,1:3)),2)/3.0;
        quiver(midx,midy,-coeff(:,1),-coeff(:,2),0.5,'k','LineWidth',2)
        xlabel('NAD27 UTM X Zone 13 [m]','FontSize',12)
        ylabel('NAD27 UTM Y Zone 13 [m]','FontSize',12)
        text(6.06E5,3.594E6,'b','FontSize',22,'FontWeight','bold')
        print('-dmeta','vector_plots_noSNL15-6_network.emf')
    end

    % save results to file for making tables
    stnames
    out = abs(100.0*(Q(1:nst,1:3)-Q(ones(nst,1)*(nst+1),1:3))./Q(ones(nst,1)*(nst+1),1:3));
    save('triangles_remove_one_well.dat','out','-ASCII');

    figure()
    scrnsz = get(0,'ScreenSize');

    % change in mean interior angle-ratio of network
    figure()
    ylab = {'%\Delta in area-weighted mean angle ratio', ...
            '%\Delta in area-weighted median angle ratio', ...
            '%\Delta in median triangle area'};

    fname = {'mean_angle','median_angle','median_area'};

    for i=1:3
        clf
        tmp=abs(100.0*(Q(1:nst,i)-Q(nst+1,i))./Q(nst+1,i));
        bar(1:nst,tmp,'r');
        hold on;
        tmp(sthull(:)) = 0.0;
        bar(1:nst,tmp,'b');
        xlabel('removal of steel-cased well');
        ylabel(ylab{i},'fontsize',13);
        set(gcf,'PaperType','tabloid')
        set(gca,'XTickMode','manual');
        set(gca,'XTick',1:nst);
        set(gca,'XTickLabel',stnames);
        set(gcf,'PaperPositionMode','auto')
    set(gcf,'Position',[10,50,0.85*scrnsz(3),0.33*scrnsz(4)])
    print('-dmeta',['triangles_remove1_',fname{i},'_compare.emf'])
end

% relative difference between gradient without steel well and base case
BASE = GRAD(1:npts,ones(nst,1)*(nst+1),1:2);
DIFF = (GRAD(1:npts,1:nst,1:2) - BASE)./BASE;
mag = sqrt(sum((GRAD(:,1:nst,1:2) - GRAD(:,ones(nst,1)*(nst+1),1:2)).^2,3));

% area "effected" by removal of well (square meters)
clf;
subplot(211)
tol = 1.0E-3;
mask = mag > tol;
mask(~INSIDE) = false;
count = zeros(nst,1);
for j=1:nst
    count(j) = sum(mag(:,j) > tol);
end
tmp = dx*dy*(count);
bar(1:nst,log10(tmp),'r');
hold on;
tmp(sthull(:)) = 0.0;
bar(1:nst,log10(tmp),'b');
xlabel('removal of steel-cased well');
title('log_{10}(area) effected (0.001) by removal [m^2]','fontsize',13);
axis([0,nst+1,4,8]);
set(gcf,'PaperType','tabloid')
set(gca,'XTickMode','manual');
set(gca,'XTick',1:nst);
set(gca,'XTickLabel',stnames);
subplot(212)
tol = 1.0E-2;
mask = mag > tol;
mask(~INSIDE) = false;
count = zeros(nst,1);
for j=1:nst
    count(j) = sum(mag(:,j) > tol);
end
tmp = dx*dy*(count);
bar(1:nst,log10(tmp),'r');
hold on;
tmp(sthull(:)) = 0.0;
bar(1:nst,log10(tmp),'b');
xlabel('removal of steel-cased well');
title('log_{10}(area) effected (0.01) by removal [m^2]','fontsize',13);
axis([0,nst+1,4,8]);
set(gcf,'PaperType','tabloid')
set(gca,'XTickMode','manual');
set(gca,'XTick',1:nst);
set(gca,'XTickLabel',stnames);

set(gcf,'PaperPositionMode','auto')
set(gcf,'Position',[10,50,0.85*scrnsz(3),0.5*scrnsz(4)])
print('-dmeta','triangles_remove1_effected_logarea_compare.emf')

DIFF(~INSIDE,:,:) = NaN;

% change in gradient magnitude upon removal of steel well
clf;
DIFF2 = GRAD(1:npts,1:nst,1:2) - BASE;  % not normalized
LEN = sqrt(DIFF2(1:npts,1:nst,1).^2 + DIFF2(1:npts,1:nst,2).^2);
tmp = sum(LEN(mask),1)./count;
bar(1:nst,tmp,'r');
tmp(sthull(:)) = 1.0;
bar(1:nst,tmp,'b');
set(gca,'YScale','log')
hold on;
xlabel('removal of steel-cased well');
ylabel('\Delta in gradient magnitude from well removal','fontsize',13);
set(gcf,'PaperType','tabloid')
set(gca,'XTickMode','manual');
set(gca,'XTick',1:nst);
set(gca,'XTickLabel',stnames);
set(gcf,'PaperPositionMode','auto')
set(gcf,'Position',[10,50,0.85*scrnsz(3),0.33*scrnsz(4)])
print('-dmeta','triangles_remove1_gradmag_compare.emf')

% change in mean gradient angle upon removal of steel well
clf;
angle = abs(atan2(DIFF2(1:npts,1:nst,2), DIFF2(1:npts,1:nst,1)));
```

```
tmp = sum(angle(mask),1)./count;
bar(1:nst,tmp,'r');
hold on;
tmp(sthull(:))=0.0;
bar(1:nst,tmp,'b');
xlabel('removal of steel-cased well');
ylabel('|\Delta in gradient direction| from well removal','fontsize',13);
v=axis();
axis([v(1:2),0,pi])
set(gca,'YTick',0:pi/4:pi);
set(gca,'YTickLabel','0|45|90|135|180');
set(gcf,'PaperType','tabloid')
set(gca,'XTickMode','manual');
set(gca,'XTick',1:nst);
set(gca,'XTickLabel',stnames);
set(gcf,'PaperPositionMode','auto')
set(gcf,'Position',[10,50,0.85*scrnsz(3),0.33*scrnsz(4)])
print('-dmeta','triangles_remove1_gradang_compare.emf')

gradHSVimage = zeros(ny,nx,3);

% largest magnitude change seen in any figure (for consistent scaling)
maxmag = max(max(log10(sqrt(DIFF(:,:,1).^2 + DIFF(:,:,2).^2))));

figure();
for i=1:nst

    % easier to re-compute than save
    x = [fiberwells(:,1);steelwells(1:i-1,1);steelwells(i+1:nst,1)];
    y = [fiberwells(:,2);steelwells(1:i-1,2);steelwells(i+1:nst,2)];

    % wells that make a convex hull for the dataset less one well
    localhull = convhull(x(:),y(:));
    LOCINSIDE = inpolygon(X,Y,x(localhull),y(localhull));

    clf()
    mag = sqrt(reshape(DIFF(:,i,1).^2,ny,nx) + reshape(DIFF(:,i,2).^2,ny,nx));
    rex = reshape(DIFF(:,i,1),ny,nx);
    rey = reshape(DIFF(:,i,2),ny,nx);
    angle = abs(atan2(rey,rex));

    % clear results outside the convex hull of the reduced dataset.
    mag(~LOCINSIDE) = NaN;
    angle(~LOCINSIDE) = NaN;

    % map angle onto hue and log10(magnitude) onto brightness (assume full
    % saturation)

    % data range: 0 <= theta <= +pi
    blue = 0.6534;  % red is 1.0; scale range from blue to red
    gradHSVimage(:,:,1) = (1.0 - blue)*angle./pi + blue;
    gradHSVimage(:,:,2:3) = 1.0;  % full saturation / brightness

    logmag = log10(mag);
    minmag = log10(tol);

    % reset values lower than tolerance to tolerance
    logmag(logmag < minmag) = minmag;
    gradALPHA = (logmag - minmag)./(maxmag - minmag);

    h = image(hsv2rgb(gradHSVimage));
    set(h,'XData',X(1,:)); % assign coorinates to pixels to allow
    set(h,'YData',Y(:,1)); % overlays to be plotted over image
    set(h,'AlphaData',gradALPHA);  % make "no-change" areas clear
    axis xy   % flip y-axis from image convention to plot convention

    daspect([1,1,1]);
    hold on;
    title(stnames{i},'fontsize',15);
    xlabel('NAD27 UTM x Zone 13 [m]');
    ylabel('NAD27 UTM y Zone 13 [m]');

    tri = delaunay(x,y);
    triplot(tri,x,y,'-g','LineWidth',1/3);

    hold on
    plot(fiberwells(:,1),fiberwells(:,2),'bs', ...
        'MarkerSize',9,'MarkerFaceColor','b');
    plot([steelwells(1:i-1,1);steelwells(i+1:nst,1)], ...
        [steelwells(1:i-1,2);steelwells(i+1:nst,2)],'ro', ...
        'MarkerSize',9,'MarkerFaceColor','r');
    plot(steelwells(i,1),steelwells(i,2),'ko', ...
        'LineWidth',2.5,'MarkerEdgeColor','r', ...
        'MarkerSize',14,'MarkerFaceColor','k');

    plot(margin(:,1),margin(:,2),'-m','LineWidth',2);
    plot(noflow(:,1),noflow(:,2),'--k','LineWidth',2);
    plot(wipp(:,1),wipp(:,2),'-k','LineWidth',2);

    if strcmp(stnames{i},'H-2b2') || strcmp(stnames{i},'ERDA-9') || ...
            strcmp(stnames{i},'H-3b2') || strcmp(stnames{i},'WIPP-19')
        % for on-site wells, zoom in to WIPP LWB area
        axis(nearwipp);
    else
        axis([xmin,xmax,ymin,ymax]);
    end
    set(gcf,'PaperPositionMode','auto')
    set(gcf,'PaperType','usletter')
    set(gcf,'Position',[10,50,(scrnsz(4)-120)*0.85,scrnsz(4)-120])
    print('-dmeta',['triangles_grad_change_',stnames{i},'.emf']);
end

```

8.3.4. MATLAB script `triangles_remove_two.m`

The following MATLAB script computes and plots the triangle interior angle metric upon removal of two steel wells from the well network.

```
clear
% This matlab script looks at the effects that removing one of the
% steel-cased (without replacement) would have on the estimation of the
% gradient, using linear interpolation across Delauny triangles as the
% estimator.

firstWell = {'WIPP-25','WIPP-13','H-12','H-7b1'};
nfst = size(firstWell,2);

% Load data
addpath '..\common_programs\';

% well datat (x,y,fwh,res,casing type)
wells = load('..\common_data\2007_well_data_for_triangles.dat');
names = textread('..\common_data\2007_well_names_for_triangles.dat','%s');

RHmask = wells(:,4) > -990; % exclude SNL-6 and SNL-15
wells = wells(RHmask,:);
names = names(RHmask,:);

margin = load('..\common_data\composite_23_margin.dat');
noflow = load('..\common_data\no_flow_boundary.dat');
totalbdry = load('..\common_data\total_boundary.dat');

% wells that make a convex hull around the dataset of all wells
hull = convhull(wells(:,1),wells(:,2));

steelwells = wells(wells(:,5)==1,1:3);  % fifth column indicates casing type
fiberwells = wells(wells(:,5)==0,1:3);
stnames =   {names{wells(:,5)==1}};

% wells on the hull that are also steel-cased
j=1;
for i=1:size(steelwells,1)
    for k=1:size(hull,1)
        if sqrt((steelwells(i,1) - wells(hull(k),1))^2 + ...
                (steelwells(i,2) - wells(hull(k),2))^2) < 1
            sthull(j) = i;
            j=j+1;
        end
    end
end

xt=wells(:,1);
yt=wells(:,2);
ht=wells(:,3);
nw = size(xt,1);
```

```
nst = size(steelwells,1);
Q = zeros(nst+1,nfst,2);
Q = NaN;

stnames{nst+1} = 'BASE-CASE';

% calculation grid (not MODFLOW grid) is minimal grid which includes
% convex hull around data
xmin = min(xt);  ymin = min(yt);
xmax = max(xt);  ymax = max(yt);
dx = 100.0;       dy = 100.0;  % note: using 100x100 is slow.

[X,Y] = meshgrid(xmin:dx:xmax, ymin:dy:ymax);
nx = size(X,2);
ny = size(X,1);

INSIDE = inpolygon(X,Y,totalbdry(:,1),totalbdry(:,2));

npts = numel(X);

% direction and magnitude of gradient in each cell, for
% scenario of removing each steel-casing well + base case

GRAD = ones(npts,nst+1,2);

% effects of removing one steel-casing well + base case for comparison
for mm=1:nfst
    for jj=1:nst+1

        % two steel wells must be different
        if ~strcmp(firstWell{mm},stnames{jj})

            if jj < nst+1
                % set of x,y,h without steel casing well jj or mm
                x = [fiberwells(:,1);steelwells(~strcmp(stnames(1:nst),stnames{jj}) & ...
                                                ~strcmp(stnames(1:nst),firstWell{mm}),1)];
                y = [fiberwells(:,2);steelwells(~strcmp(stnames(1:nst),stnames{jj}) & ...
                                                ~strcmp(stnames(1:nst),firstWell{mm}),2)];
                h = [fiberwells(:,3);steelwells(~strcmp(stnames(1:nst),stnames{jj}) & ...
                                                ~strcmp(stnames(1:nst),firstWell{mm}),3)];
            else
                x = xt;
                y = yt;
                h = ht;
            end

            tri = delaunay(x,y);
            nt = size(tri,1);

            D = zeros(size(tri,1),1);
            coeff = zeros(size(tri,1),4);
            grad = zeros(size(tri,1),2); % angle and magnitide of hydraulic gradient
            geom = zeros(size(tri,1),8);  % 3 sides, 3 angles, area, # pts inside

            %% compute equation for line through 3 points
            % value of determinant used in denominator of Cramer's rule
            D(1:nt) = x(tri(:,1)).*y(tri(:,2)) + x(tri(:,2)).*y(tri(:,3)) + ...
                y(tri(:,1)).*x(tri(:,3)) - x(tri(:,3)).*y(tri(:,2)) - ...
                x(tri(:,1)).*y(tri(:,3)) - x(tri(:,2)).*y(tri(:,1));

            % a (coefficient on x)
            coeff(1:nt,1) = (h(tri(:,1)).*y(tri(:,2)) + y(tri(:,1)).*h(tri(:,3)) + ...
                h(tri(:,2)).*y(tri(:,3)) - h(tri(:,3)).*y(tri(:,2)) - ...
                h(tri(:,2)).*y(tri(:,1)) - h(tri(:,1)).*y(tri(:,3)))./D;

            % b (coefficient on y)
            coeff(1:nt,2) = (x(tri(:,1)).*h(tri(:,2)) + h(tri(:,1)).*x(tri(:,3)) + ...
                x(tri(:,2)).*h(tri(:,3)) - x(tri(:,3)).*h(tri(:,2)) - ...
                x(tri(:,2)).*h(tri(:,1)) - x(tri(:,1)).*h(tri(:,3)))./D;

            % c (constant coefficient)
            coeff(1:nt,3) = (x(tri(:,1)).*y(tri(:,2)).*h(tri(:,3)) + ...
                y(tri(:,1)).*h(tri(:,2)).*x(tri(:,3)) + ...
                x(tri(:,2)).*y(tri(:,3)).*h(tri(:,1)) - ...
                x(tri(:,3)).*y(tri(:,2)).*h(tri(:,1)) - ...
                x(tri(:,2)).*y(tri(:,1)).*h(tri(:,3)) - ...
                x(tri(:,1)).*y(tri(:,3)).*h(tri(:,2)))./D;

            % compute angle and magnitude of hydraulic gradient
            grad(1:nt,1) = atan2(coeff(:,2),coeff(:,1));
            grad(1:nt,2) = sqrt(sum(coeff(:,1:2).^2,2));
            grad(1:nt,3) = max(h(tri(:,1:3)),[],2) - min(h(tri(:,1:3)),[],2);

            % map results from "vector" triangles to "raster" grid
            for kk=1:nt
                % result is a logical vector, indicating if the cell is in (T) or
                % out (F) side this current triangle
                IN = reshape(inpolygon(X,Y,x(tri(kk,1:3)),y(tri(kk,1:3))),npts,1);

                % sum(IN) = number of cells inside the triangle
                % ones(sum(IN))*kk = column vector of the counter kk
                % coeff(...,1:2) = x & y gradient repeated for every cell inside
                %                  that triangle, copied to correct locations in GRAD

                GRAD(IN,jj,1:2) = coeff(ones(sum(IN),1)*kk,1:2);
            end
            %% calculate geometric things related to triangles
            % lengths from Pythagorean theorem
            % angles from cosine law
            % area from Matlab built-in fcn

            % length of side a (2->3)
            geom(1:nt,1) = sqrt((x(tri(:,3)) - x(tri(:,2))).^2 + ...
                (y(tri(:,3)) - y(tri(:,2))).^2);
            % length of side b (3->1)
            geom(1:nt,2) = sqrt((x(tri(:,1)) - x(tri(:,3))).^2 + ...
                (y(tri(:,1)) - y(tri(:,3))).^2);
            % length of side c (1->2)
            geom(1:nt,3) = sqrt((x(tri(:,2)) - x(tri(:,1))).^2 + ...
                (y(tri(:,2)) - y(tri(:,1))).^2);


            % angle 1 between sides b & c in radians
            geom(1:nt,4) = acos((sum(geom(:,2:3).^2,2) - geom(:,1).^2)./ ...
                (2.0*prod(geom(:,2:3),2)));
            % angle 2 between sides a & c in radians
            geom(1:nt,5) = acos((sum(geom(:,1:2:3).^2,2) - geom(:,2).^2)./ ...
                (2.0*prod(geom(:,1:2:3),2)));
            % angle 3 between sides b & a in radians
            geom(1:nt,6) = acos((sum(geom(:,1:2).^2,2) - geom(:,3).^2)./ ...
                (2.0*prod(geom(:,1:2),2)));

            % area of triangle - use MATLAB built-in function
            geom(1:nt,7) = polyarea(x(tri(:,1:3)),y(tri(:,1:3)),2);

            % compute goodness triangle criteria
            ang_ratio = min(geom(:,4:6),[],2)./max(geom(:,4:6),[],2);

            % area-weighted angle ratio
            Q(jj,mm,1) = sum(ang_ratio(:).*geom(:,7))/sum(geom(1:nt,7));

            % median triangle area
            Q(jj,mm,2) = median(geom(1:nt,7));
        else
            Q(jj,mm,1:2) = NaN;
        end
    end
end

scrnsz = get(0,'ScreenSize');

BASE = Q(ones(1,nst)*(nst+1),:,1:2);
plt = (Q(1:nst,:,1:2) - BASE)./BASE;

figure()
h1 = imagesc(plt(1:nst,:,1));
axis('image')
%colormap(redwhitemap(reshape(plt(:,:,1),numel(plt(:,:,1)))));
colorbar()
figure()
h2 = imagesc(plt(1:nst,:,2));
axis('image')
%colormap(redwhitemap(reshape(plt(:,:,2),numel(plt(:,:,2)))));
colorbar()
```

8.4. Model Parameter Correlation Maximization Scripts

The first two scripts are run in Linux, then the following Python and R scripts are run in Windows and are used to compute the correlation and partial correlation results used in the analysis.

8.4.1. Bash shell script `checkout_model_data.sh`

The following Linux Bash shell script is run to check the Culebra MODFLOW model inputs and results needed out from CVS, convert the binary head output files to ASCII, perform directory manipulations and zip the results into a single file for transfer to Windows XP.

```
#!/bin/bash

# this Bash script is run in Linux and checks out the model files
# required to perform the model correlation analysis.

repo=/nfs/data/CVSLIB

# check out the list of the final 100 fields used from AP-144
cvs -d ${repo}/MiningMod checkout Inputs/keepers

# move it into the current directory
mv Inputs/keepers .
rm -rf Inputs

# checkout model inputs from Tfields repository in CVS (AP-114 Task 7)
for d in `cat keepers`; do
  # checkout transmissivity and anisotropy fields
  cvs -d ${repo}/Tfields checkout Outputs/${d}/modeled_{K,A}_field.mod
done

# modify the path of "updated" T-fields, so they are all at the
# same level in the directory structure (to make these agree w/ mining mod repository)

if [ -a keepers_short ]; then
    # delete any pre-existing files here,
    # since file is concatenated to in next loop
    rm keepers_short
fi

for d in `cat keepers`; do
  bn=`basename ${d}`
  # test whether it is a compound path
  if [ ${d} != ${bn} ]; then
      dn=`dirname ${d}`
      mv ./Outputs/${d}/ ./Outputs/

      # put an empty file in the directory to indicate
      # what the directory was previously named
      touch ./Outputs/${bn}/${dn}
  fi

  # create a keepers list without directories
  echo ${bn} >> keepers_short
done

# get output files from MiningMod CVS repository
for d in `cat keepers_short`; do
    # checkout particle tracking results (R0 is no mining replicate)
    cvs -d ${repo}/MiningMod checkout Outputs/R0/${d}/dtrk.out
    # checkout binary heads
    cvs -d ${repo}/MiningMod checkout Outputs/R0/${d}/modeled_head.bin

    # move files into existing directories
    mv Outputs/R0/${d}/{dtrk.out,modeled_head.bin} Outputs/${d}/
done

# remove intermediate directories
rm -rf Outputs/R0
rm -rf Outputs/Update
rm -rf Outputs/Update2

# convert binary MODFLOW head output to ascii for use in AP-111 analysis
for d in `cat keepers_short`; do
```

```
    cd Outputs/${d}
    ln -sf ../../head_bin2ascii.py .
    python head_bin2ascii.py
    rm ./head_bin2ascii.py
    rm ./modeled_head.bin
    cd ../..
done

# zip results up for transfer to windowz
cd Outputs
zip -r model_files.zip r???
mv model_files.zip ../
```

8.4.2. Python script `head_bin2ascii.py`

The following Python script is run in Linux to convert the binary MODFLOW head output files to ASCII format, for transfer to Windows XP for further analysis. This script is called by the Bash shell script `checkout_model_data.sh` that checks the data out of CVS and does the looping over the directories.

```
import struct
from sys import argv,exit

class FortranFile(file):
    """ modified from May 2007 Enthought-dev mailing list post by Neil Martinsen-Burrell"""

    def __init__(self,fname, mode='r', buf=0):
    file.__init__(self, fname, mode, buf)
    self.ENDIAN = '<'  # little endian
        self.di = 4  # default integer (could be 8 on 64-bit platforms)

    def readReals(self, prec='f'):
        """Read in an array of reals (default single precision) with error checking"""
        # read header (length of record)
    l = struct.unpack(self.ENDIAN+'i',self.read(self.di))[0]
    data_str = self.read(l)
    len_real = struct.calcsize(prec)
    if l % len_real != 0:
        raise IOError('Error reading array of reals from data file')
    num = l/len_real
    reals = struct.unpack(self.ENDIAN+str(num)+prec,data_str)
        # check footer
    if struct.unpack(self.ENDIAN+'i',self.read(self.di))[0] != l:
        raise IOError('Error reading array of reals from data file')
    return list(reals)

    def readInts(self):
        """Read in an array of integers with error checking"""
    l = struct.unpack('i',self.read(self.di))[0]
    data_str = self.read(l)
    len_int = struct.calcsize('i')
    if l % len_int != 0:
        raise IOError('Error reading array of integers from data file')
    num = l/len_int
    ints = struct.unpack(str(num)+'i',data_str)
    if struct.unpack(self.ENDIAN+'i',self.read(self.di))[0] != l:
        raise IOError('Error reading array of integers from data file')
    return list(ints)

    def readRecord(self):
        """Read a single fortran record (potentially mixed reals and ints)"""
        dat = self.read(self.di)
        if len(dat) == 0:
            raise IOError('Empy record header')
    l = struct.unpack(self.ENDIAN+'i',dat)[0]
    data_str = self.read(l)
        if len(data_str) != l:
            raise IOError('Didn''t read enough data')
        check = self.read(self.di)
        if len(check) != 4:
            raise IOError('Didn''t read enough data')
    if struct.unpack(self.ENDIAN+'i',check)[0] != l:
        raise IOError('Error reading record from data file')
    return data_str
```

```
def reshapev2m(v,nx,ny):
    """Reshape a vector that was previously reshaped in C-major order from a matrix,
    back into a C-major order matrix (here a list of lists)."""
    m = [None]*ny
    n = nx*ny
    for i,(lo,hi) in enumerate(zip(xrange(0, n-nx+1, nx), xrange(nx, n+1, nx))):
        m[i] = v[lo:hi]
    return m

def floatmatsave(filehandle,m):
    """Writes array to open filehandle.
    Outer list is rows, inner lists are columns."""

    for row in m:
        f.write(''.join([' %9.4f' % col for col in row]) + '\n')

# open file and set endian-ness
try:
    infn,outfn = argv[1:3]
except:
    print '2 command-line arguments not given, using default in/out filenames'
    infn = 'modeled_head.bin'
    outfn = 'modeled_head.hed'

ff = FortranFile(infn)

# currently this assumes a single-layer MODFLOW model (or at least only one layer of output)

# format of MODFLOW header in binary layer array
fmt = '<2i2f16s3i'
# little endian, 2 integers, 2 floats,
#    16-character string (4 element array of 4-byte strings), 3 integers

while True:
    try:
        # read in header
        h = ff.readRecord()

    except IOError:
        # exit while loop
        break

    else:
        # unpack header
        kstp,kper,pertim,totim,text,ncol,nrow,ilay = struct.unpack(fmt,h)

        # print status/confirmation to terminal
        print kstp,kper,pertim,totim,text,ncol,nrow,ilay

        h = ff.readReals()

ff.close()

f = open(outfn,'w')

floatmatsave(f,reshapev2m(h,ncol,nrow)[::-1])
f.close()
```

8.4.3. Python script `load_model_data.py`

The following Python script is not called by itself, but instead is used as a library in two other Python scripts. This script loads the model input (transmissivity and anisotropy fields) and model output (head) from each of the 100 calibrated MODFLOW realizations.

```
import numpy as np
from os.path import join
from glob import glob

datadir = '../../../common_data/'
fh = open(datadir + 'model_domain_specs.dat','r')
nx,ny =      [int(x) for x in fh.readline().strip().split()]
xmin,ymin = [float(x) for x in fh.readline().strip().split()]
xmax,ymax = [float(x) for x in fh.readline().strip().split()]
fh.close()
```

```
dt = np.float64  # "double precision"

# number of fields and elements in each
numf = 100
numel = nx*ny
ndata = 3
tcorr = np.zeros((ndata+1,numel),dtype=dt)
hcorr = np.zeros((ndata,numel),dtype=dt)
trav = np.zeros((numf,),dtype=dt)

dpi = 160
figsize = (14,6)

kdat = np.zeros((numf,numel),dtype=dt)
adat = np.zeros((numf,numel),dtype=dt)
hdat = np.zeros((numf,numel),dtype=dt)

# loop over all the directories, read input
for i,d in enumerate(glob('r???')):
    print i,d
    # x (row) log10 hydraulic conductivity
    kdat[i,:] = np.loadtxt(join(d,'modeled_K_field.mod'),dtype=dt)
    # log10 ratio y/x (col/row) for conductivity
    adat[i,:] = np.loadtxt(join(d,'modeled_A_field.mod'),dtype=dt)

    # log10 travel time to LWB is first column, last row
    fn = open(join(d,'dtrk.out'),'r')
    trav[i] = float(fn.readlines()[-1].split()[0])
    fn.close()

    # read in modflow head (saved in file as a matrix already)
    hdat[i,:] = np.loadtxt(join(d,'modeled_head.hed'),dtype=dt)[::-1,:].reshape((numel,))

kdat = np.log10(kdat)
adat = np.log10(adat)
trav = np.log10(trav)

hdat[hdat == -999] = np.NaN

tflat = trav.flatten()
kdat[kdat < -15] = np.NaN
keff = kdat + 0.5*adat

print 'min log effective k:',np.nanmin(keff)
print 'max log effective k:',np.nanmax(keff)

# define a mask that selects the WIPP LWB area + a buffer of cells around it
wippmask = np.zeros((307,284),dtype='bool') # false boolean array
buffer = 15
wippmask[121-buffer:185+buffer,88-buffer:152+buffer] = True
wippmask.shape = (307*284,)

print 'successfully loaded model data'
```

8.4.4. Python script `export_pcor_inputs.py`

The following Python script calls the library `load_model_data.py` to read in the model data, then exports the K_{eff} and head data for an area surrounding the WIPP LWB for use in the following R script that does the partial correlation analysis.

```
import numpy as np
from load_model_data import *

# save large matrix: nrows = 100
# ncols = # elements (here (64 + (buffer * 2))**2 + 1 for travel time)

# save imported data for use in R
# for partial correlation analysis

# perform outer difference, then only use upper triangle of tensor

np.savetxt('keff_trav.dat',
           np.concatenate((keff[:,wippmask],trav[:,None]),axis=1),
           fmt='%.7f')
```

```
np.savetxt('head_trav.dat',
          np.concatenate((hdat[:,wippmask],trav[:,None]),axis=1),
          fmt='%.7f')

print 'saved data for partial correlation analysis in R'
```

8.4.5. R script `compute_partial_correlations.R`

The following R script loads in the data exported by `export_pcor_inputs.py`, computes the partial correlation of K_{eff} and head in each cell to travel times and head to travel times, accounting for the effects K_{eff} or head in all other cells.

```
# read in the matrix that has realizations as rows (100) and parameters as columns
# (k or h at model cells and travel time as last column)

k <- read.table('keff_trav.dat')
library(corpcor)

# this takes a lot of RAM (> 2GB)
pc <- pcor.shrink(k)

# write all rows, last column to file (partial correlation of each k  to travel time
# holding effects of all other k values constant)
write.table(pc[,dim(pc)[1]],'kpc.out',row.names=FALSE,col.names=FALSE)

h <- read.table('head_trav.dat')
pc <- pcor.shrink(h)

# write all rows, last column to file (partial correlation of each k  to travel time
# holding effects of all other k values constant)
write.table(pc[,dim(pc)[1]],'hpc.out',row.names=FALSE,col.names=FALSE)
```

8.4.6. Python script `spearman_rank_coefficient.py`

The following Python script computes correlation statistics between the results of the Culebra model calibration (particle tracking times to the WIPP LWB) and the Culebra model input files, creating plots of the results for the report.

```
import numpy as np
from os.path import join
from glob import glob
import matplotlib
matplotlib.use('Agg') # to improve memory usage
import matplotlib.pyplot as plt
import matplotlib.colors as colors

# save code for loading data in separate module
from load_model_data import *

def finish_fig(extents):
    '''Add common things to figures'''
    plt.hold = True
    plt.xlabel('UTM NAD27 X [km]')
    plt.axis(extents)
    locs,labels = plt.xticks()
    plt.xticks(locs,(locs/1000.0).astype('|S3'))
    plt.ylabel('UTM NAD27 Y [km]')
    locs,labels = plt.yticks()
    plt.yticks(locs,(locs/1000.0).astype('|S4'),rotation=90)
    plt.plot(wipp[:,0],wipp[:,1],'k-',linewidth=1)
    plt.plot(h2[:,0],h2[:,1],'g--',linewidth=2)
    plt.plot(h3[:,0],h3[:,1],'r:',linewidth=2)
    plt.plot(salado[:,0],salado[:,1],'k:',linewidth=2)
    plt.plot(wells[fiberg,0],wells[fiberg,1],'gs',markersize=4)
    plt.plot(wells[~fiberg,0],wells[~fiberg,1],'ro',markersize=4)
    plt.axis('image')
    plt.axis(extents)

# load in partial-correlation data exported from R
pck = np.zeros((307*284,),dtype=dt)
```

```
pch = np.zeros((307*284,),dtype=dt)

pch[wippmask] = np.loadtxt('hpc.out',dtype=dt)
pck[wippmask] = np.loadtxt('kpc.out',dtype=dt)

pck.shape = (307,284)
pch.shape = (307,284)

print 'successfully loaded partial correlation data'

for n in xrange(numel):
    if n % 10000 == 0:
        print n

    # A = travel time from C-2737 to WIPP LWB (global)
    # B = head at same cell as property (local)
    # Tx vs A/B
    # Ty vs A/B
    # Teff vs A/B
    # A vs B

    data1 = [kdat[:,n],  kdat[:,n] + adat[:,n], kdat[:,n] + 0.5*adat[:,n]]
    hflat = hdat[:,n].flatten()

    for i,d in enumerate(data1):
        dflat = d.flatten()
        tcorr[i,n] = np.corrcoef(dflat,tflat)[0,1]
        hcorr[i,n] = np.corrcoef(dflat,hflat)[0,1]

    tcorr[ndata,n] = np.corrcoef(hflat,tflat)[0,1]

# blank out no-flow area
tcorr[np.isnan(kdat[0,:])[None,:]] = np.NaN
hcorr[np.isnan(kdat[0,:])[None,:]] = np.NaN

# clean up some temporary things
del data1
del hflat
del dflat

wipp = np.loadtxt(datadir+'wipp_boundary.dat')
h2 = np.loadtxt(join(datadir,'h2_200711.dat'),delimiter=',')
h3 = np.loadtxt(join(datadir,'h3_200711.dat'),delimiter=',')
salado = np.loadtxt(join(datadir,'mrgn_dissolution.dat'),skiprows=5)
wells = np.loadtxt(datadir+'2007_well_data.dat')
fiberg = wells[:,4] == 0.0

# regional left,right,bottom,top
regext = (xmin,xmax,ymin,ymax)

# wipp area left,right,bottom,top
wippext = (wipp[:,0].min() - 1500.0, wipp[:,0].max() + 1500.0,
           wipp[:,1].min() - 1500.0, wipp[:,1].max() + 1500.0)

cmap = colors.LinearSegmentedColormap.from_list('bwr',('blue','white','red'))
norm1 = colors.Normalize(vmin=-1,vmax=+1)
norm2 = colors.Normalize(vmin=-0.5,vmax=+0.5)
normsm1 = colors.Normalize(vmin=-0.015,vmax=+0.015)
normsm2 = colors.Normalize(vmin=-0.005,vmax=+0.005)

plt.figure(1)
plt.semilogy(10.0**trav,'k*')
plt.xlabel('realization')
plt.ylabel('years travel time to WIPP LWB')
plt.savefig('travel_times.png')
plt.close(1)

fmt = '%.5e'
fn = ['_kx_','_ky_','_keff_']
nn = ['K_x', 'K_y','K_{eff}']

# plot comparisons of partial and regular Keff correlation inside WIPP
plt.figure(1,figsize=figsize,dpi=dpi)
plt.subplot(121)
plt.imshow(tcorr[2,:].reshape((ny,nx)),interpolation='nearest',
           cmap=cmap,norm=norm2,extent=regext)
plt.colorbar(shrink=0.8)
finish_fig(wippext)
plt.title('corr. $K_{eff}$ w/ travel time')
plt.subplot(122)
plt.imshow(pck.reshape((ny,nx)),interpolation='nearest',
           cmap=cmap,norm=normsm2,extent=regext)
plt.colorbar(shrink=0.8)
finish_fig(wippext)
plt.title('partial corr. $K_{eff}$ w/ travel time')
plt.savefig('Keff_partial_travel_time_corr.png')
plt.close(1)

# plot comparisons of partial and regular head correlation inside WIPP
plt.figure(1,figsize=figsize,dpi=dpi)
plt.subplot(121)
plt.imshow(tcorr[ndata,:].reshape((ny,nx)),interpolation='nearest',
           cmap=cmap,norm=norm2,extent=regext)
plt.colorbar(shrink=0.8)
finish_fig(wippext)
plt.title('corr. $h$ w/ travel time')
plt.subplot(122)
plt.imshow(pch.reshape((ny,nx)),interpolation='nearest',
           cmap=cmap,norm=normsm1,extent=regext)
plt.colorbar(shrink=0.8)
finish_fig(wippext)
plt.title('partial corr. $h$ w/ travel time')
plt.savefig('h_partial_travel_time_corr.png')
plt.close(1)

# write results (reshaped into matrix form)
for j,f in enumerate(fn):
    np.savetxt('corr'+f+'vs_time.dat', tcorr[j,:].reshape((ny,nx)),fmt=fmt)

    plt.figure(1,figsize=figsize,dpi=dpi)
    plt.subplot(121)
    plt.imshow(tcorr[j,:].reshape((ny,nx)),interpolation='nearest',
               cmap=cmap,norm=norm2,extent=regext)
    finish_fig(regext)
    plt.title('regional corr. $' + nn[j] + '$ w/ travel time')
    plt.subplot(122)
    plt.imshow(tcorr[j,:].reshape((ny,nx)),interpolation='nearest',
               cmap=cmap,norm=norm2,extent=regext)
    plt.colorbar(shrink=0.8)
    finish_fig(wippext)
    plt.title('WIPP corr. $' + nn[j] + '$ w/ travel time')
    plt.savefig(f[1:] + 'travel_time_corr.png')
    plt.close(1)

    np.savetxt('corr'+f+'vs_head.dat', hcorr[j,:].reshape((ny,nx)),fmt=fmt)

    plt.figure(2,figsize=figsize,dpi=dpi)
    plt.subplot(121)
    plt.imshow(hcorr[j,:].reshape((ny,nx)),interpolation='nearest',
               cmap=cmap,norm=norm1,extent=regext)
    finish_fig(regext)
    plt.title('regional corr. $' + nn[j] + '$ w/ head')
    plt.subplot(122)
    plt.imshow(hcorr[j,:].reshape((ny,nx)),interpolation='nearest',
               cmap=cmap,norm=norm1,extent=regext)
    plt.colorbar(shrink=0.8)
    finish_fig(wippext)
    plt.title('WIPP corr. $' + nn[j] + '$ w/ head')
    plt.savefig(f[1:] + 'heads_corr.png')
    plt.close(2)

np.savetxt('corr_head_vs_time.dat', tcorr[ndata,:].reshape((ny,nx)),fmt=fmt)

plt.figure(4,figsize=figsize,dpi=dpi)
plt.subplot(121)
plt.imshow(tcorr[ndata,:].reshape((ny,nx)),interpolation='nearest',
           cmap=cmap,norm=norm2,extent=regext)
finish_fig(regext)
plt.title('regional corr. head w/ time')
plt.subplot(122)
plt.imshow(tcorr[ndata,:].reshape((ny,nx)),interpolation='nearest',
           cmap=cmap,norm=norm2,extent=regext)
plt.colorbar(shrink=0.8)
finish_fig(wippext)
plt.title('WIPP corr. head w/ time')
plt.savefig('heads_vs_travel_time_corr.png')
plt.close(4)

# compute variance across all realizations for output and each parameter
print 'travel time to WIPP LWB:\tmean:%.8e\tstd:%.8e\n' % \
    (trav.sum()/100.0,np.sqrt(np.var(trav)))
```

```
data = [kdat, kdat+adat, kdat+0.5*adat]
dnam = ['kx_', 'ky_', 'keff_']
dnnm = ['$\\log_{10}(K_x', '$\\log_{10}(K_y', '$\\log_{10}(K_{eff}']

for j,(dat,nam) in enumerate(zip(data,dnam)):
    std = np.sqrt(np.var(dat,axis=0).reshape((ny,nx)))
    std[std < 1.0E-10] = 0.0
    mean = (dat.sum(axis=0)/100.0).reshape((ny,nx))
    np.savetxt(nam+'var.out',std,fmt=fmt)
    np.savetxt(nam+'mean.out',mean,fmt=fmt)
    plt.figure(3,figsize=figsize,dpi=dpi)
    plt.subplot(121)
    plt.imshow(mean,interpolation='nearest',extent=regext)
    plt.colorbar(shrink=0.8)
    finish_fig(regext)
    plt.title('mean ' + dnnm[j] + ')$')
    plt.subplot(122)
    plt.imshow(std,interpolation='nearest',extent=regext, norm=colors.Normalize(vmin=0.0))
    plt.colorbar(shrink=0.8)
    finish_fig(regext)
    plt.title('$\\log_{10}$ standard deviation ' + dnnm[j] + ')$')
    plt.savefig(nam + 'avg_std.png')
    plt.close(3)
```

8.5. Combination of Three Methods Scripts

8.5.1. Python script `combine_plot_methods.py`

The following Python script combines the results of the three individual methods, plots the figures in the text, and samples the results at steel-cased well locations to create the table in the text.

```
import numpy as np
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
import matplotlib.colors as colors
from os.path import join
from itertools import chain

# weights used for recombination
w = (0.5,1.0,1.0)

def normalize_field(f):
    """pass a field with NaN in places outside active modflow region"""
    fmin = np.nanmin(f)
    fmax = np.nanmax(f)
    return (f-fmin)/(fmax-fmin)

def normalize_triangle(f):
    fmin = np.nanmin(f)
    fmax = np.nanmax(f)
    return f/(fmax-fmin)

def spread_field(fsm,factor=2):
    """map a field that is a subset of the 307x284 field onto the large field"""
    flarge = np.empty((fsm.shape[0]*factor,fsm.shape[1]*factor),dtype=fsm.dtype)
    for j,row in enumerate(fsm):
        drow = list(chain(*[(x,x) for x in row]))
        flarge[2*j,:] = drow
        flarge[2*j+1,:] = drow
    return flarge[0:-1,:]

def finish_fig(extents):
    '''Add common things to figures'''
    plt.hold = True
    plt.xlabel('UTM NAD27 X [km]')
    plt.axis(extents)
    locs,labels = plt.xticks()
    plt.xticks(locs,(locs/1000.0).astype('|S3'))
    plt.ylabel('UTM NAD27 Y [km]')
    locs,labels = plt.yticks()
    plt.yticks(locs,(locs/1000.0).astype('|S4'),rotation=90)
    plt.plot(wipp[:,0],wipp[:,1],'k-',linewidth=1)
    plt.plot(h2[:,0],h2[:,1],'g--',linewidth=2)
    plt.plot(h3[:,0],h3[:,1],'r:',linewidth=2)
    plt.plot(salado[:,0],salado[:,1],'b:',linewidth=2)
    plt.plot(wells[fiberg,0],wells[fiberg,1],'gs',markersize=4)
    plt.plot(wells[steel,0],wells[steel,1],'ro',markersize=4)
    plt.axis('image')
    plt.axis(extents)

datadir = join('..','common_data')

fh = open(join(datadir,'model_domain_specs.dat'),'r')
nx,ny =     [int(x) for x in fh.readline().strip().split()]
xmin,ymin = [float(x) for x in fh.readline().strip().split()]
xmax,ymax = [float(x) for x in fh.readline().strip().split()]
fh.close()

wipp = np.loadtxt(join(datadir,'wipp_boundary.dat'))
h2 = np.loadtxt(join(datadir,'h2_200711.dat'),delimiter=',')
h3 = np.loadtxt(join(datadir,'h3_200711.dat'),delimiter=',')
salado = np.loadtxt(join(datadir,'mrgn_dissolution.dat'),skiprows=5)
wells = np.loadtxt(join(datadir,'2007_well_data.dat'))

fhwn = open(join(datadir,'2007_well_data_with_names.dat'),'r')
# names are last column of each row, but not including 2 wells in CH region
steel_well_names = [x.rstrip().split()[-1] for x in fhwn if x.split()[-2] == '1']
fhwn.close()

fiberg = wells[:,4] == 0.0
steel = wells[:,4] == 1.0

wellij = np.zeros((steel.sum(),2),'int')
wellij[:,0] = np.floor((wells[steel,0] - xmin)/100.0)
wellij[:,1] = np.floor((ymin - wells[steel,1])/100.0)

# regional left,right,bottom,top
regext = (xmin,xmax,ymin,ymax)

# wipp area left,right,bottom,top
wippext = (wipp[:,0].min() - 1500.0, wipp[:,0].max() + 1500.0,
           wipp[:,1].min() - 1500.0, wipp[:,1].max() + 1500.0)

fs = (18,9)

# read in mean/medain kriging + 1 results
# these are on a mesh with 1/4 as many elements (1/2 as many in each direction)
# and therefore must be mapped onto the MODFLOW grid

kmean = np.loadtxt(join('..','kriging_add_well','addone_mod_results_mean.dat'))
kmedian = np.loadtxt(join('..','kriging_add_well','addone_mod_results_median.dat'))

kmean[kmean==1] = np.NaN      # blank out areas outside MODFLOW active areas
kmedian[kmedian==1] = np.NaN

nkmean = normalize_field(kmean)
nkmedian = normalize_field(kmedian)

kmean = spread_field(kmean)[::-1,:]
kmedian = spread_field(kmedian)[::-1,:]

nkmean = spread_field(nkmean)[::-1,:]  # flip wrt y
nkmedian = spread_field(nkmedian)[::-1,:]

# read in the results of the add-one analysis for triangles
tmean = np.loadtxt(join('..','triangle_add_well','triangles_add_one_mean.dat'))
tmedian = np.loadtxt(join('..','triangle_add_well','triangles_add_one_median.dat'))

tmean[tmean==-999] = np.NaN      # blank out areas outside MODFLOW active areas
tmedian[tmedian==-999] = np.NaN

ntmean = normalize_triangle(tmean)[::-1,:]
ntmedian = normalize_triangle(tmedian)[::-1,:]

# read in correlation results (handling the NaN in the file)
fhk = open(join('..','model_correlation','CRA2009_model',
                        'final_100_fields','corr_keff_vs_time.dat'),'r')
kl = []
for line in fhk:
    kl.append([float(x) for x in
               line.strip().replace('1.#QNANe+00','-999').split()])
fhk.close()
kcorr = np.array(kl)
del kl

fhh = open(join('..','model_correlation','CRA2009_model',
                           'final_100_fields','corr_head_vs_time.dat'),'r')
hl = []
for line in fhh:
    hl.append([float(x) for x in
                line.strip().replace('1.#QNANe+00','-999').replace('-1.#IND0e+00','1.0E-
16').split()])
fhh.close()
hcorr = np.array(hl)
del hl

# set -999 substituted above back to NaN
kcorr[kcorr==-999]=np.NaN
kcorr[np.isnan(tmean[::-1,:])]=np.NaN
hcorr[hcorr==-999]=np.NaN
hcorr[np.isnan(tmean[::-1,:])]=np.NaN

nkcorr = normalize_field(np.abs(kcorr))  # should already be flipped
nhcorr = normalize_field(np.abs(hcorr))

# ####################
# compute the remove-one analysis for correlation results from sampling-based correlation
analysis
fh = open('corr_remove_one_steel.dat','w')

fh.write('\t'.join(steel_well_names) + '\n')
np.savetxt(fh,kcorr[wellij[:,1],wellij[:,0]][None,:],fmt='%.6e',delimiter='\t')
np.savetxt(fh,hcorr[wellij[:,1],wellij[:,0]][None,:],fmt='%.6e',delimiter='\t')
fh.close()

# plot up histograms of distribution in each field
plt.figure(1,figsize=(12.5,10))
bins = 150

plt.subplot(321)
plt.hist(nkmean[~np.isnan(nkmean)].flatten(),bins=bins)
plt.ylabel('frequency')
plt.xlabel(r'scaled mean $\Delta$ kriging var.')
plt.axis('tight')

plt.subplot(322)
plt.hist(nkmedian[~np.isnan(nkmedian)].flatten(),bins=bins)
plt.xlabel(r'scaled median $\Delta$ kriging var.')
plt.axis('tight')

plt.subplot(323)
plt.hist(ntmean[~np.isnan(ntmean)].flatten(),bins=bins)
plt.ylabel('frequency')
plt.xlabel(r'scaled mean $\Delta$ triangle angle raio')
plt.axis('tight')

plt.subplot(324)
plt.hist(ntmedian[~np.isnan(ntmedian)].flatten(),bins=bins)
plt.xlabel(r'scaled median $\Delta$ triangle angle raio')
plt.axis('tight')

plt.subplot(325)
plt.hist(nkcorr[~np.isnan(nkcorr)].flatten(),bins=bins)
plt.ylabel('frequency')
plt.xlabel(r'scaled $\rho$ $K_{eff}$ vs. $t$')
plt.axis('tight')

plt.subplot(326)
plt.hist(nhcorr[~np.isnan(nhcorr)].flatten(),bins=bins)
plt.xlabel(r'scaled $\rho$ head vs. $t$')
plt.axis('tight')

plt.subplots_adjust(hspace=0.3)
plt.savefig('histograms_of_distributions.png')
plt.close(1)

# plot up histograms of original (unscaled) distribution in each field
plt.figure(1,figsize=(12.5,10))
bins = 150

plt.subplot(321)
plt.hist(kmean[~np.isnan(kmean)].flatten(),bins=bins)
plt.axis('tight')
plt.ylabel('frequency')
plt.xlabel(r'unscaled mean $\Delta$ kriging var.')

plt.subplot(322)
plt.hist(kmedian[~np.isnan(kmedian)].flatten(),bins=bins)
plt.axis('tight')
plt.xlabel(r'unscaled median $\Delta$ kriging var.')

plt.subplot(323)
plt.hist(tmean[~np.isnan(tmean)].flatten(),bins=bins)
plt.axis('tight')
plt.ylabel('frequency')
plt.xlabel(r'unscaled mean $\Delta$ triangle angle raio')

plt.subplot(324)
plt.hist(tmedian[~np.isnan(tmedian)].flatten(),bins=bins)
plt.axis('tight')
plt.xlabel(r'unscaled median $\Delta$ triangle angle raio')

plt.subplot(325)
plt.hist(kcorr[~np.isnan(kcorr)].flatten(),bins=bins)
plt.axis('tight')
plt.ylabel('frequency')
plt.xlabel(r'unscaled $\rho$ $K_{eff}$ vs. $t$')

plt.subplot(326)
plt.hist(hcorr[~np.isnan(hcorr)].flatten(),bins=bins)
plt.axis('tight')
plt.xlabel(r'unscaled $\rho$ head vs. $t$')

plt.subplots_adjust(hspace=0.3)
plt.savefig('histograms_of_original_distributions.png')
plt.close(1)

cmap = colors.LinearSegmentedColormap.from_list('rwg',('red','orange','white','blue','purple'))
nrm = colors.Normalize(vmin=-2.4,vmax=2.4)
#cmap = 'jet'
out = np.zeros((307,284,4))

## combine results linearly with multipliers
# mean and median + keff and head = 4 results
plt.figure(1)
#out[:,:,0] = np.sqrt((w[0]*nkcorr)**2 + (w[1]*nkmean)**2 + (w[2]*ntmean)**2)
out[:,:,0] = w[0]*nkcorr + w[1]*nkmean + w[2]*ntmean
plt.imshow(out[:,:,0],interpolation='nearest',extent=regext,cmap=cmap,norm=nrm)
cb = plt.colorbar(shrink=0.8)
cb.set_label('$S_c$')
plt.title('$K_{eff} +$ mean')
finish_fig(regext)
plt.savefig('combined_results_map_Keff_mean.png')
plt.close(1)

plt.figure(1)
#out[:,:,1] = np.sqrt((w[0]*nhcorr)**2 + (w[1]*nkmean)**2 + (w[2]*ntmean)**2)
out[:,:,1] = w[0]*nhcorr + w[1]*nkmean + w[2]*ntmean
plt.imshow(out[:,:,1],interpolation='nearest',extent=regext,cmap=cmap,norm=nrm)
cb = plt.colorbar(shrink=0.8)
cb.set_label('$S_c$')
plt.title('$h +$ mean' )
finish_fig(regext)
plt.savefig('combined_results_map_h_mean.png')
plt.close(1)

plt.figure(1)
#out[:,:,2] = np.sqrt((w[0]*nkcorr)**2 + (w[1]*nkmedian)**2 + (w[2]*ntmedian)**2)
out[:,:,2] = w[0]*nkcorr + w[1]*nkmedian + w[2]*ntmedian
plt.imshow(out[:,:,2],interpolation='nearest',extent=regext,cmap=cmap,norm=nrm)
cb = plt.colorbar(shrink=0.8)
cb.set_label('$S_c$')
plt.title('$K_{eff} +$ median' )
finish_fig(regext)
plt.savefig('combined_results_map_Keff_median.png')
plt.close(1)

plt.figure(1)
#out[:,:,3] = np.sqrt((w[0]*nhcorr)**2 + (w[1]*nkmedian)**2 + (w[2]*ntmedian)**2)
out[:,:,3] = w[0]*nhcorr + w[1]*nkmedian + w[2]*ntmedian
plt.imshow(out[:,:,3],interpolation='nearest',extent=regext,cmap=cmap,norm=nrm)
cb = plt.colorbar(shrink=0.8)
cb.set_label('$S_c$')
plt.title('$h +$ median' )
finish_fig(regext)
plt.savefig('combined_results_map_h_median.png')
plt.close()
```

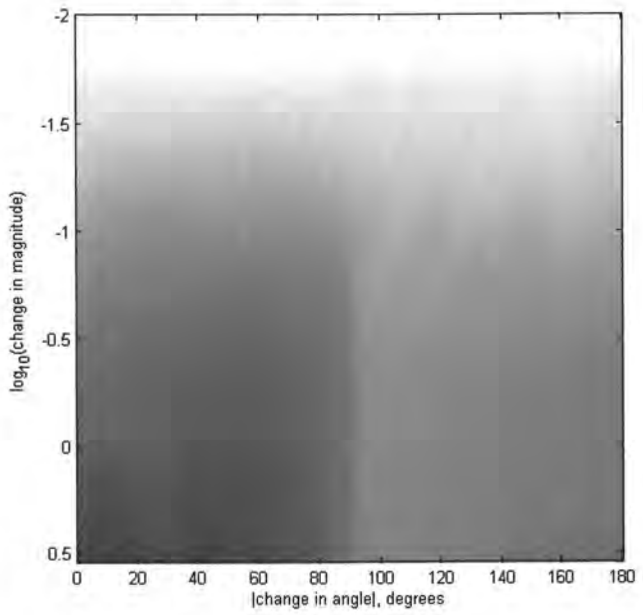
```
# ########################################
# save table of results at steel-cased wells
fh = open('composite_remove_one_steel.dat','w')

fh.write('\t'.join(steel_well_names) + '\n')
for j in [0,1,2,3]:
    np.savetxt(fh,out[wellij[:,1],wellij[:,0],j][None,:],fmt='%.6e',delimiter='\t')
fh.close()
```

9.0 Remove One Steel Well Figures

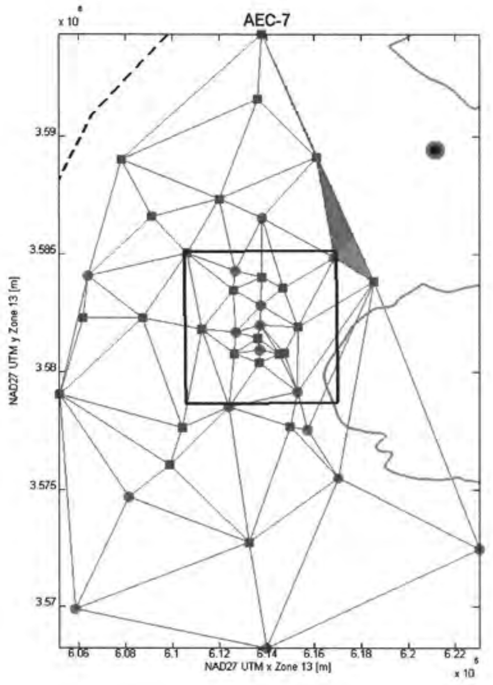
The following set of 18 figures (1 colormap and 17 map plots) shows the computed impact on the estimated local gradient from removal of a single steel-cased well from the Culebra monitoring network. Two different types of changes are being illustrated; changes in both gradient and magnitude are represented by the hue and saturation of the color, respectively. White areas in the figures correspond to areas where the change in the predicted gradient is less than the threshold value (0.01), while the deeply colored areas indicate a large change in the magnitude of the gradient. Hot (red) colors indicate the magnitude of the change in angle, with cool colors (blue) indicating no change in direction (both factors are illustrated in the 2D color map below).

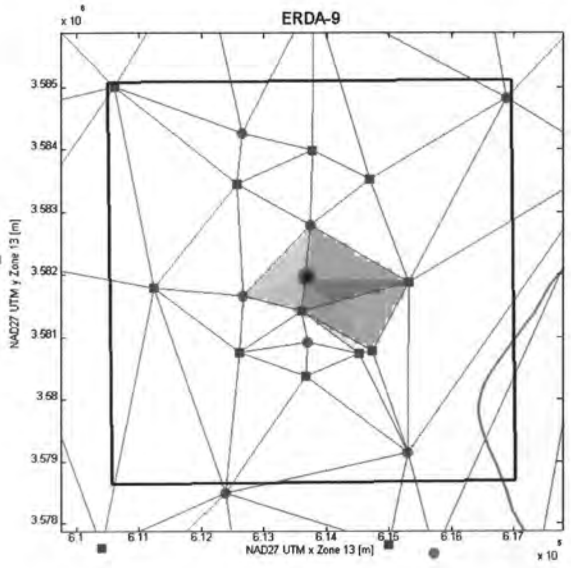
For example, if removing a well causes the gradient to change in magnitude only by the maximum amount, the region would be filled with dark blue. Likewise, if the gradient direction change completely (180 degrees) upon removing the well, but the gradient magnitude only changed slightly, the region would be filled with a faint red or pink color. Magenta indicates a change of 90 degrees, halfway between red and blue.

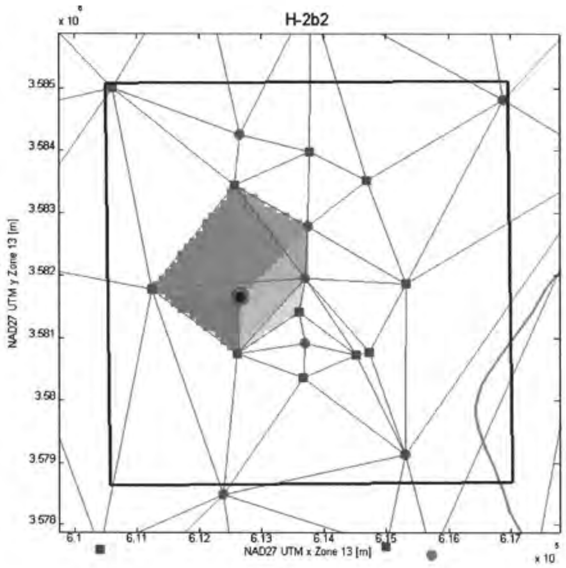


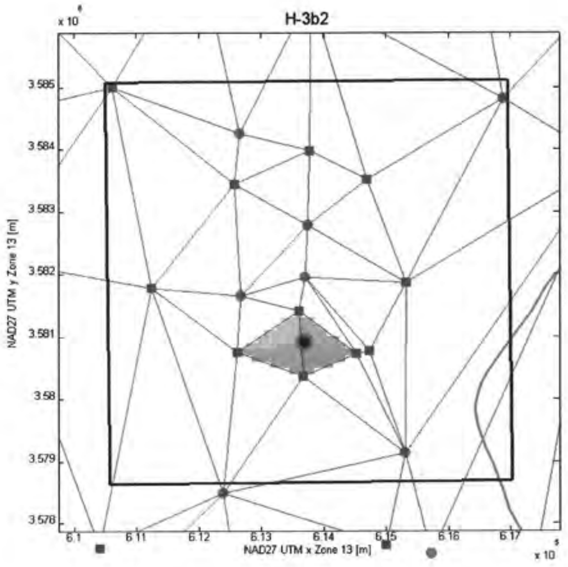
Each figure shows the localized effects on the gradient magnitude estimate, due to removing a steel-cased well from the network. The colors (representing changes above the threshold) are only found in the triangles directly connected to the removed point.

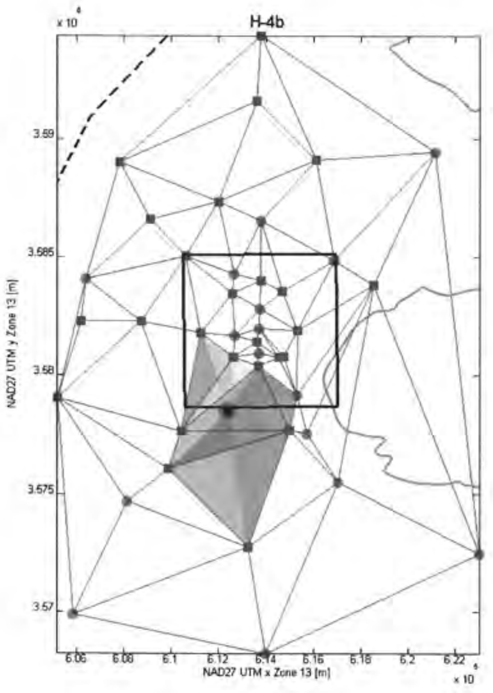
For wells that have no effect outside the WIPP LWB, the plot area is reduced to this smaller area. The Delaunay triangles corresponding to the reduced monitoring network are plotted on the figure; the original triangles are not shown.

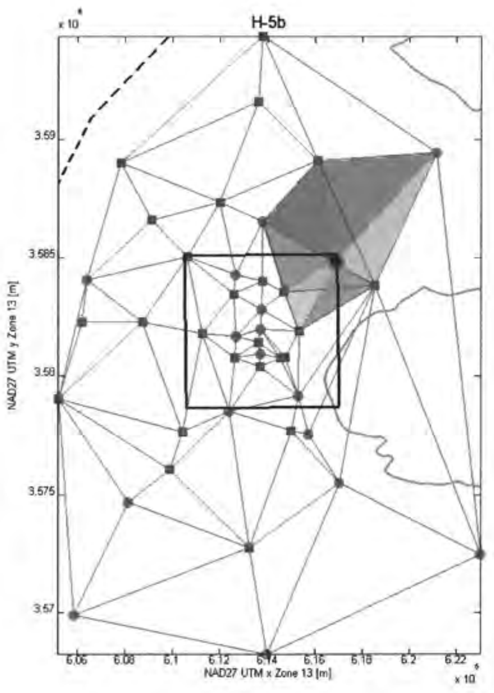

AEC-7
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06
6.08
6.1
6.12
6.14
6.16
6.18
6.2
6.22
x 10^5
NAD27 UTM x Zone 13 [m]

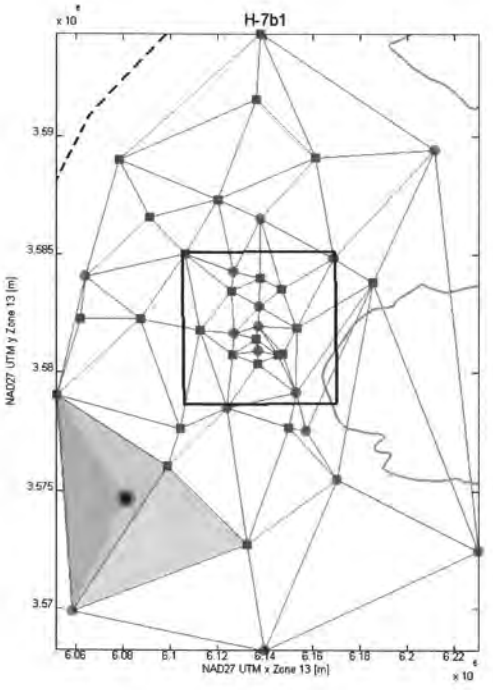

ERDA-9
x 10^6
3.585
3.584
3.583
3.582
3.581
3.58
3.579
3.578
NAD27 UTM y Zone 13 [m]
6.1
6.11
6.12
6.13
6.14
6.15
6.16
6.17
x 10^5
NAD27 UTM x Zone 13 [m]

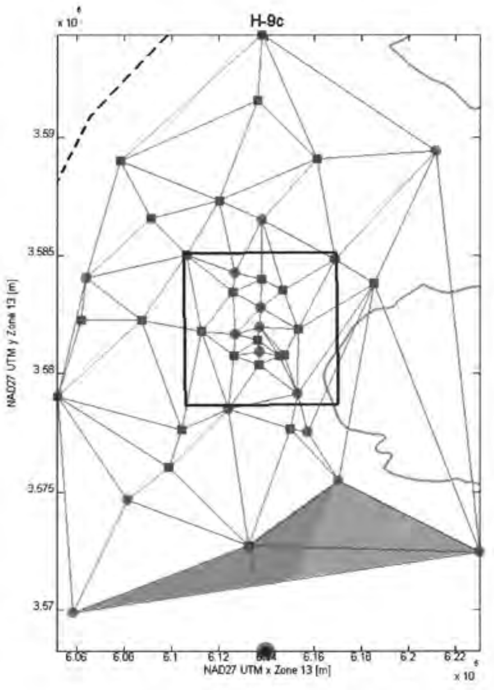

H-2b2
x 10^6
3.585
3.584
3.583
3.582
3.581
3.58
3.579
3.578
NAD27 UTM y Zone 13 [m]
6.1
6.11
6.12
6.13
6.14
6.15
6.16
6.17
x 10^5
NAD27 UTM x Zone 13 [m]

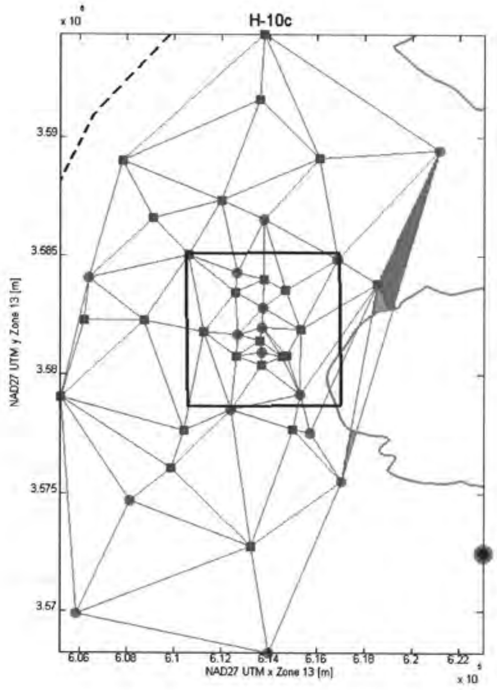

H-3b2
x 10^6
3.585
3.584
3.583
3.582
3.581
3.58
3.579
3.578
NAD27 UTM y Zone 13 [m]
6.1
6.11
6.12
6.13
6.14
6.15
6.16
6.17
x 10^5
NAD27 UTM x Zone 13 [m]

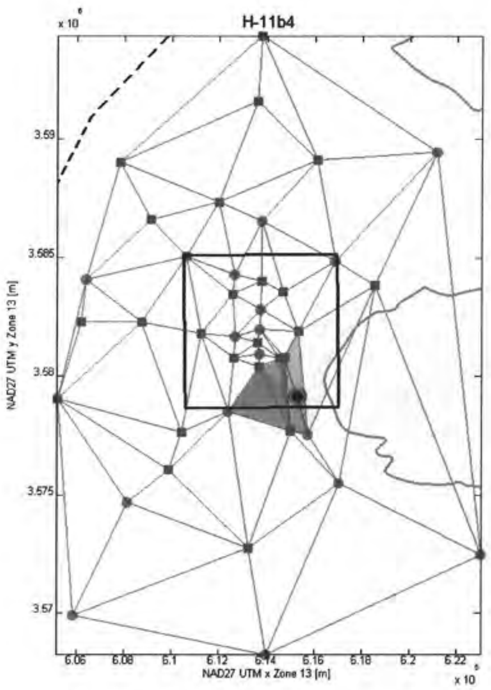

H-4b
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06 6.08 6.1 6.12 6.14 6.16 6.18 6.2 6.22
NAD27 UTM x Zone 13 [m]
x 10^5

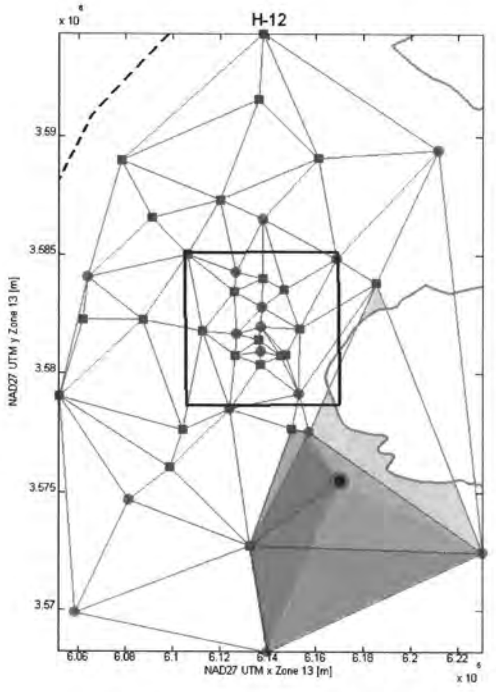

H-5b
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06 6.08 6.1 6.12 6.14 6.16 6.18 6.2 6.22
NAD27 UTM x Zone 13 [m]
x 10^5

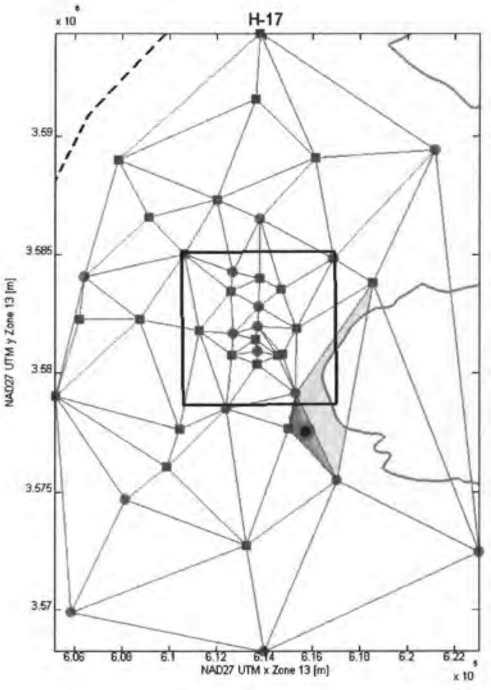

H-7b1
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06 6.08 6.1 6.12 6.14 6.16 6.18 6.2 6.22
NAD27 UTM x Zone 13 [m]
x 10^5

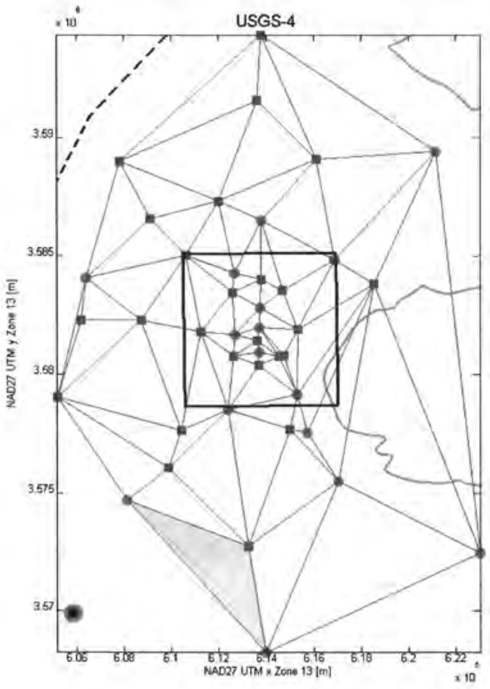

H-9c
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06 6.08 6.1 6.12 6.14 6.16 6.18 6.2 6.22
NAD27 UTM x Zone 13 [m]
x 10^5

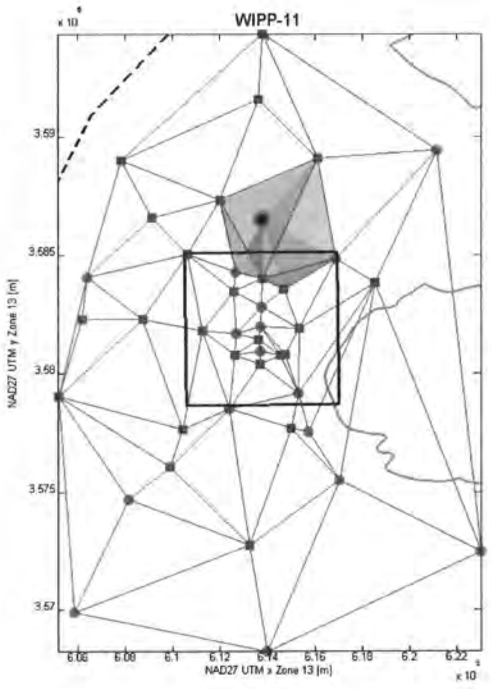

H-10c
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06
6.08
6.1
6.12
6.14
6.16
6.18
6.2
6.22
x 10^5
NAD27 UTM x Zone 13 [m]

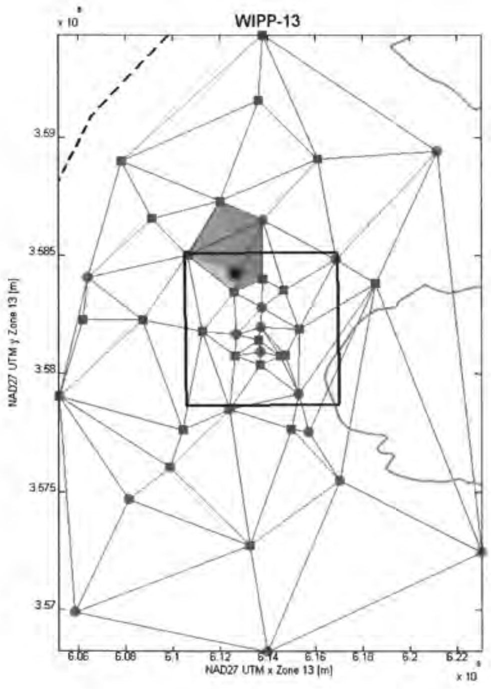

H-11b4
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06
6.08
6.1
6.12
6.14
6.16
6.18
6.2
6.22
x 10^5
NAD27 UTM x Zone 13 [m]

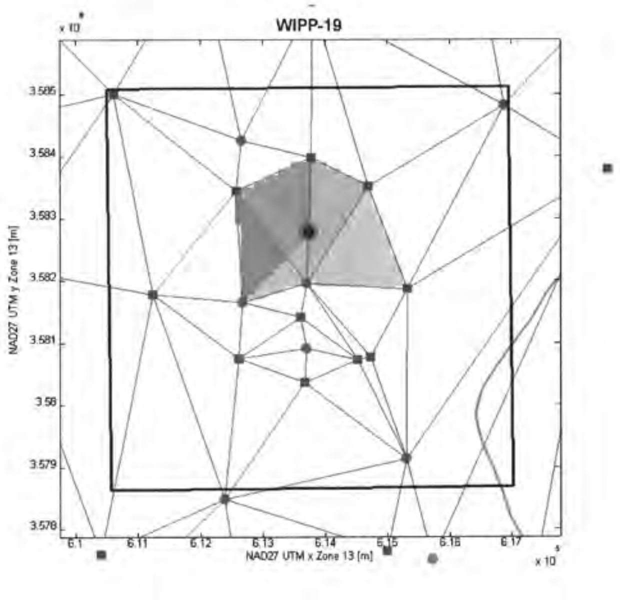

H-12
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06
6.08
6.1
6.12
6.14
6.16
6.18
6.2
6.22
x 10^5
NAD27 UTM x Zone 13 [m]

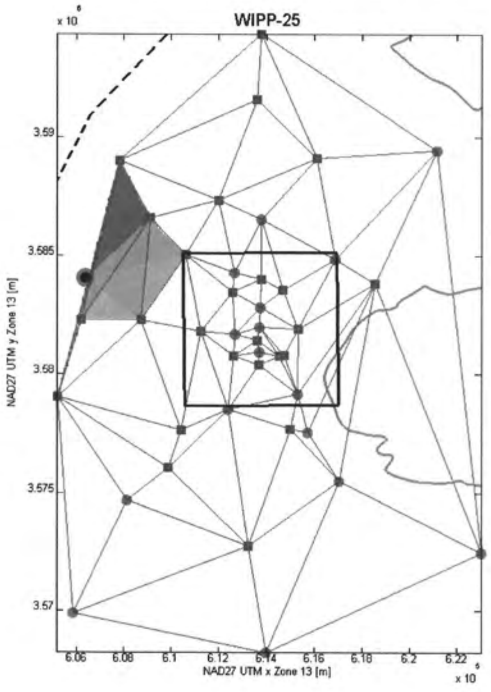

H-17
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06
6.08
6.1
6.12
6.14
6.16
6.18
6.2
6.22
x 10^5
NAD27 UTM x Zone 13 [m]


USGS-4
NAD27 UTM x Zone 13 [m]
NAD27 UTM y Zone 13 [m]


WIPP-11
NAD27 UTM x Zone 13 [m]
NAD27 UTM y Zone 13 [m]


WIPP-13
NAD27 UTM x Zone 13 [m]
NAD27 UTM y Zone 13 [m]


WIPP-19
NAD27 UTM x Zone 13 [m]
NAD27 UTM y Zone 13 [m]


WIPP-25
x 10^6
3.59
3.585
3.58
3.575
3.57
NAD27 UTM y Zone 13 [m]
6.06
6.08
6.1
6.12
6.14
6.16
6.18
6.2
6.22
x 10^5
NAD27 UTM x Zone 13 [m]