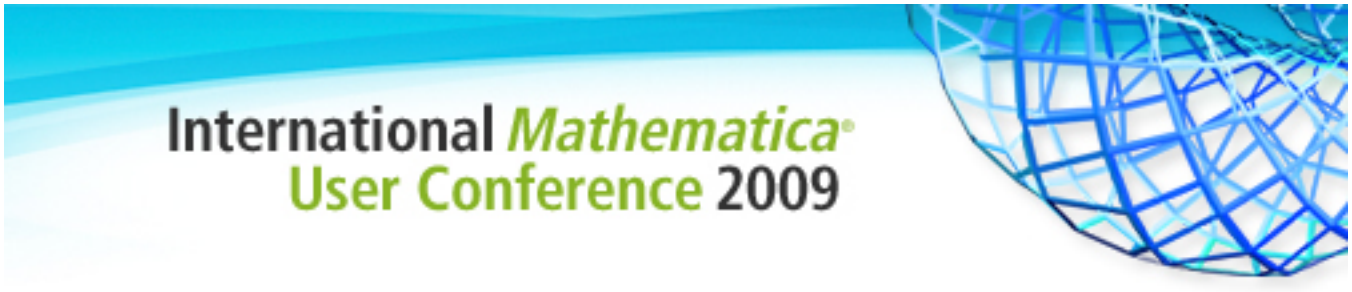
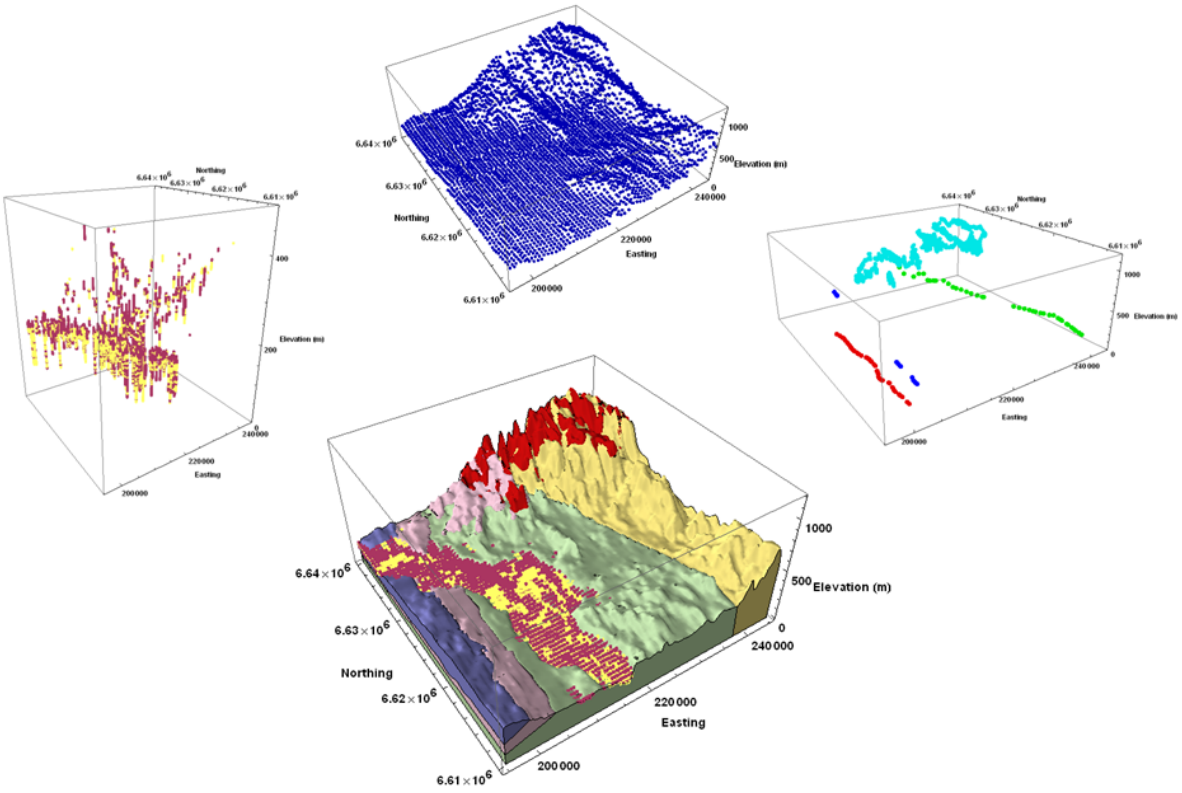


Catchment Scale 3D Geological Models from Sparse Data Sets

Bryce F. J. Kelly

The University of New South Wales, Australia
Connected Waters Initiative
National Centre for Groundwater Research and Training


Northing
Easting
Elevation (m)
6.64×10^6
6.63×10^6
6.62×10^6
6.61×10^6
200000
220000
240000
0
500
1000

Introduction

Three dimensional models of regional geological structures and the properties of each unit are important for numerous earth science applications. Such models are essential for :

- defining the distribution of base metals,
- delineating hydrocarbon traps,
- determining the volume of groundwater contamination plumes, and
- creating the framework for hydrogeological water balance models.

This presentation illustrates how to build 3D geological models using *Mathematica*. The data processing steps are demonstrated on a catchment scale model that was constructed to define the hydrogeological framework for a regional groundwater investigation in an environmentally stressed aquifer in Australia.

Geological Data Sets

The 3D geological models are built from sparse data sets including :

- borehole lithological logs,
- geological maps,
- geophysical surveys, and
- field measurements of geological surfaces (defined by their strike and dip).

Sample points are often irregularly distributed in space.

Mathematica provides a single environment to edit and sort the data, grid the sparse data, interpolate surfaces and then view the 3D conceptual geological model.

Gridding Irregularly Spaced 2D Data

The Need for Sparse Data Interpolation

The final 3D geological models are built using the function **RegionPlot3D**. To plot the region that corresponds to the geological feature of interest an approximate function over the x,y domain needs to be defined for each surface used to delineate the limits of the region.

An approximate function for a surface is determined using the **Interpolation** function. The data passed to **Interpolation** need to be on a regular grid (also called a mesh or rectangular grid).

Two sparse data regular gridding functions are defined: nearest neighbor and inverse distance. Both of these algorithms are refinements of the functions published by Haneberg (2004).

Defining the Dimensions of the Region of Interest

3D geological models are built using a combination of 2D and 3D grids.

The domain dimensions are defined here and used throughout the calculations below.

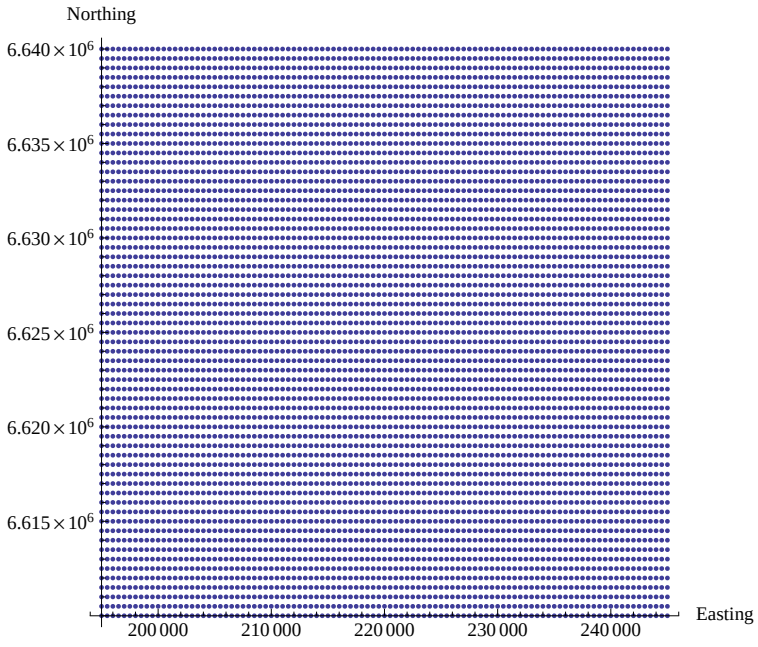
```
xmin = 195 000; xmax = 245 000; xspace = 500;
ymin = 6 610 000; ymax = 6 640 000; yspace = 500;
zmin = 0; zmax = 1200; space = 5;
```

The dimensions of the 2D grid used for many of the calculations are also defined.

```
grid2D = Flatten[Table[{x, y}, {x, xmin, xmax, xspace},
    {y, ymin, ymax, yspace}], 1];
Dimensions[grid2D]
```

```
{6161, 2}
```

```
ListPlot[grid2D, AxesOrigin → {xmin, ymin},
 PlotStyle → PointSize[0.008], AspectRatio → 1,
 AxesLabel → {"Easting", "Northing"},
 ImageSize → {350, 350}]
```



2D Nearest Neighbor Gridding of Irregularly Spaced Data

Some of the geological data sets are dense and need little interpolation. For these data sets nearest neighbor gridding is used.

The function **NearestNeighborGrid2D** defined below interpolates/extrapolates 2D data sets onto a regular grid and then the function **Interpolation** is applied to the grid list to define an approximate function at all locations over the domain. An approximate function is returned by **NearestNeighborGrid2D**.

Inputs for the new function **NearestNeighborGrid2D** are the 2D surface data, $\{\{x_1,y_1,z_1\},\{x_2,y_2,z_2\}, \ldots\}$, and the regular grid list, $\{\{x_1,y_1\},\{x_2,y_2\}, \ldots\}$.

```
NearestNeighborGrid2D[datain_, gridin_] := Module[
  {nnfunction, nn2Dmap, nn2Ddatagrid, nn2Dfunction},
  nnfunction = Nearest[
    datain[[All, {1, 2}]] → datain[[All, 3]]];
  nn2Dmap = Map[nnfunction, gridin][[All, 1]];
  nn2Ddatagrid =
   Partition[Flatten[Riffle[gridin, nn2Dmap]], 3];
  nn2Dfunction = Interpolation[nn2Ddatagrid,
    Method → "Hermite", InterpolationOrder → 3];
  Return[nn2Dfunction]]
```

2D Inverse Distance Gridding of Irregularly Spaced Data: Defined

Inverse distance gridding is commonly used to grid geological data (Yamamoto 1998).

The value at the grid node, $z_{x,y}$, is determined via:

$$z_{x,y} = \frac{\sum_{i=1}^{k} \frac{z_i}{d_i^p}}{\sum_{i=1}^{k} \frac{1}{d_i^p}}$$

where d_i is the Euclidean distance between the node and the i-th data point, k is the number of neighboring points, and p is the power of distance.

2D Inverse Distance Gridding of Irregularly Spaced Data: The Function

Haneberg (2004) published a *Mathematica* algorithm for gridding irregularly spaced data using inverse distance gridding. Below is a modified inverse distance algorithm that takes advantage of the functions **Nearest** and **EuclideanDistance**, and uses no **Do** loops for faster calculation times.

Inputs for the new function **InverseDistanceGrid2D** are the 2D surface data, {{x_1,y_1,z_1},{x_2,y_2,z_2}, ... }, the number of nearest points to use, the power of distance (typically 1, 2 or 3), and the regular grid list, {{x_1,y_1},{x_2,y_2}, ... }.

Within the **InverseDistanceGrid2D** module the data are interpolated/extrapolated onto a regular grid and then the function **Interpolation** is applied to the grid list to define an approximate function at all locations over the domain. An approximate function is returned by **InverseDistanceGrid2D**.

```
InverseDistanceGrid2D[datain_, k_, p_, gridin_] :=
 Module[{nfunction, xyz, xy,
   z, d, zestimate, zgrid, zfunction},
  nfunction = Nearest[
    datain[[All, {1, 2}]] → datain[[All]]];
  xyz = Map[nfunction[{#, #}, k] &, gridin];
  xy = xyz[[All, All, {1, 2}]];
  z = xyz[[All, All, 3]];
  d =
   Table[EuclideanDistance[gridin[[i]], xy[[i, j]]],
    {i, Length[gridin]}, {j, Length[xy[[1]]]}];
  zestimate = Apply[Plus, z / d^p, 1] /
    Apply[Plus, 1 / d^p, 1];
  zgrid = Table[{gridin[[i, 1]], gridin[[i, 2]],
     zestimate[[i]]}, {i, Length[gridin]}];
  zfunction = Interpolation[zgrid,
    Method → "Spline", InterpolationOrder → 3];
  Return[zfunction]]
```

Building the 3D Facies Model

Import the Digital Elevation Model (DEM)

Digital Elevation Models (DEM) are available from many government geological organizations.

The USGS provides DEM data for the whole globe.

The DEM of the Maules Creek catchment is imported and then reduced in size for the demonstration.

```
SetDirectory["F:\\BryceKellyMIUC2009"];
dem = Drop[Import["maulesdem.csv"], 1];
subdem =
  Table[Extract[dem, i], {i, 1, Length[dem], 50}];
Dimensions[dem]
Dimensions[subdem]
Clear[dem];
```

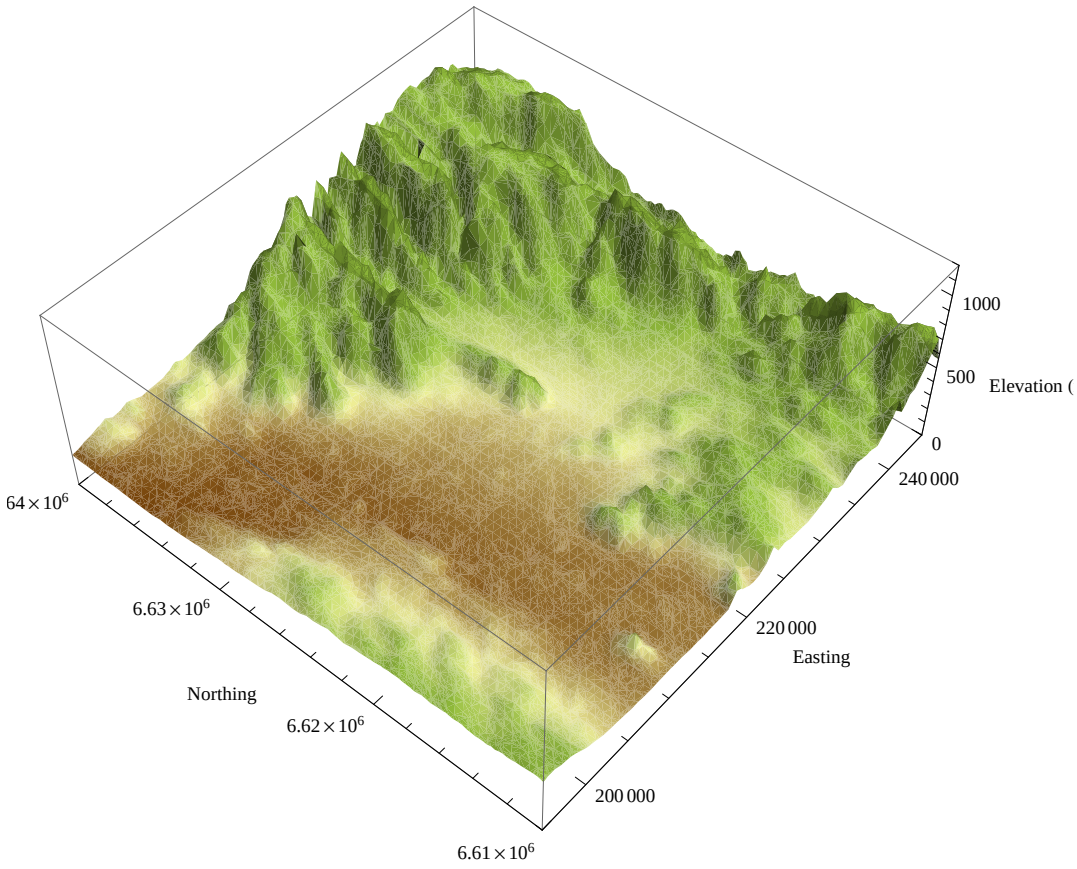
```
{503 949, 3}
```

```
{10 079, 3}
```

View the DEM in 3D

The DEM is plotted below using the function **ListPlot3D.** In the image the dark brown areas are the fertile vertosol soils where irrigated crops are grown. In the south east corner coal mines are being developed. The dimensions of the catchment are approximately 40 km by 30 km.

```
ListPlot3D[subdem,
 PlotRange → {{xmin, xmax}, {ymin, ymax}, {zmin, zmax}},
 Mesh → None,
 ColorFunction → (Blend[{
       {0, RGBColor[0.356863, 0.180392, 0]},
       {0.07,
        RGBColor[0.737255, 0.737255, 0.478431]},
       {0.15, RGBColor[0.443137, 0.572549, 0.180392]}},
      #3] &),
 AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 ViewPoint → {-1.75, -1.5, 2.5},
 ImageSize → {500, 450}]
```



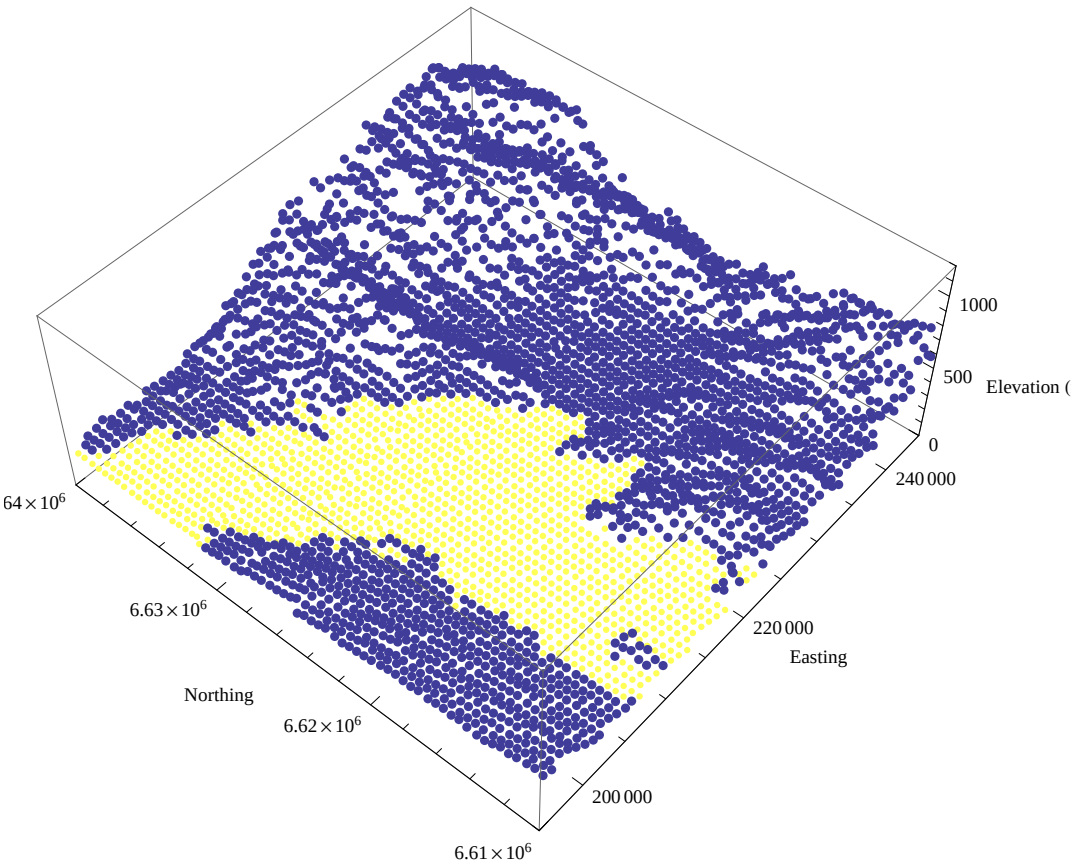
Sort the DEM into Rock and Alluvial Zones

The DEM needs to be sorted into areas that represent the rock and the alluvium. This is done by taking advantage of the fact that alluvial areas are relatively flat and at low elevations. The model is divided into two domains and the data are then sorted based on the elevation.

Alternative methods for sorting the data are variance, median and gradient filters. In this terrain the simple elevation cutoff is acceptable.

There are thin unconsolidated sediments in the eastern portion of the catchment. This demonstration focuses on the alluvial sediments in the main paleochannel.

```
rock = {};
alluvium = {};
Do[If[subdem[[i, 1]] ≤ 210 000 && subdem[[i, 3]] < 245,
  AppendTo[alluvium, subdem[[i]]], If[210 000 < subdem[[i, 1]] < 220 000 && subdem[[i, 3]] < 280,
   AppendTo[alluvium, subdem[[i]]], If[220 000 < subdem[[i, 1]] && subdem[[i, 3]] < 280,
    AppendTo[alluvium, subdem[[i]]], AppendTo[rock, subdem[[i]]]]]], {i, Length[subdem]}]

rockpointplot =
  ListPointPlot3D[rock, PlotRange → {{xmin, xmax}, {ymin, ymax}, {zmin, zmax}}, PlotStyle → PointSize[0.01]];
alluviumpointplot = ListPointPlot3D[alluvium, PlotRange → {{xmin, xmax}, {ymin, ymax}, {zmin, zmax}},
   PlotStyle → {Lighter[Yellow], PointSize[0.01]}];
Show[rockpointplot, alluviumpointplot, AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 ImageSize → {500, 450}, ViewPoint → {-1.75, -1.5, 2.5}]
```



Import the Bedrock Picks from the Boreholes

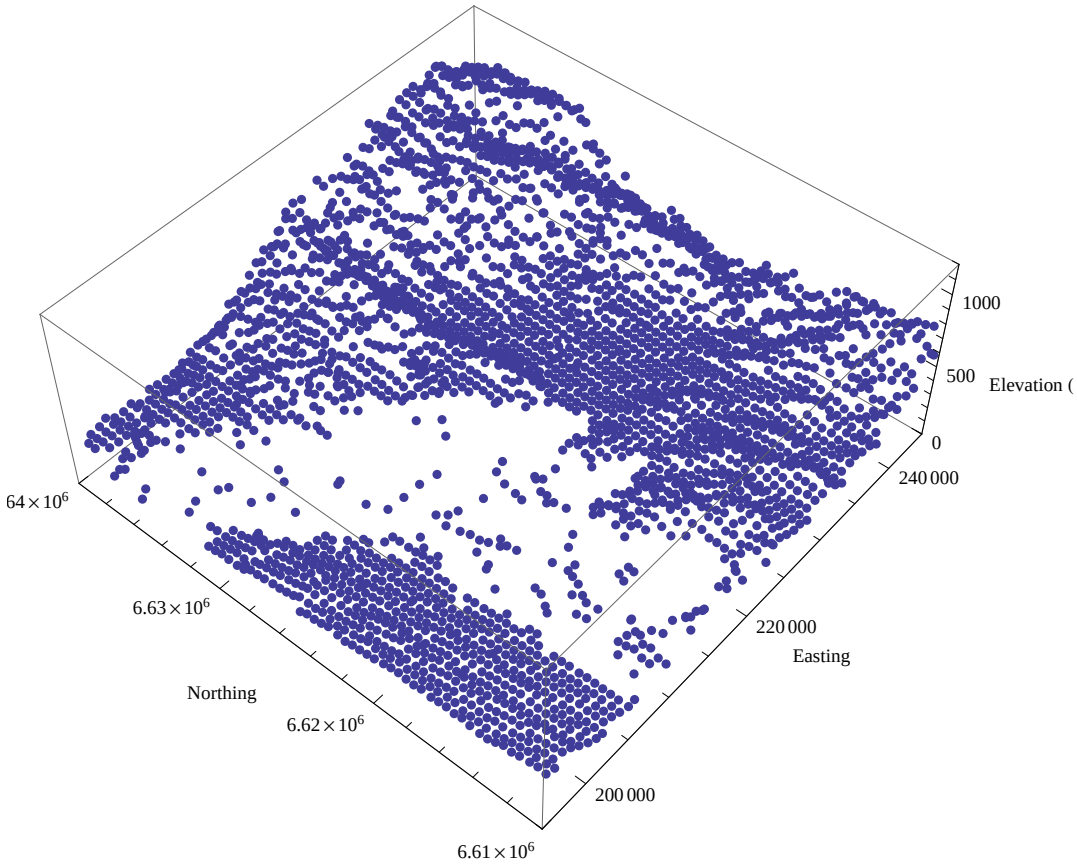
Whenever a borehole is drilled in New South Wales (NSW), Australia, a record of the lithology is kept by the NSW State Government.

Some of these boreholes intersect the bedrock. These borehole rock top picks are combined with the DEM rock data.

These points define the extent of the bedrock surface, the bottom of the alluvium (unconsolidated sediments).

```
boreholerockpoints =
  Drop[Import["maulesbedrocktops.csv"], 1][[
   All, {2, 3, 7}]];
rocksurfacepoints = Join[rock, boreholerockpoints];
```

```
ListPointPlot3D[rocksurfacepoints,
 PlotRange → {{xmin, xmax}, {ymin, ymax}, {zmin, zmax}},
 PlotStyle → PointSize[0.01],
 AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 ImageSize → {500, 450},
 ViewPoint → {-1.75, -1.5, 2.5}]
```



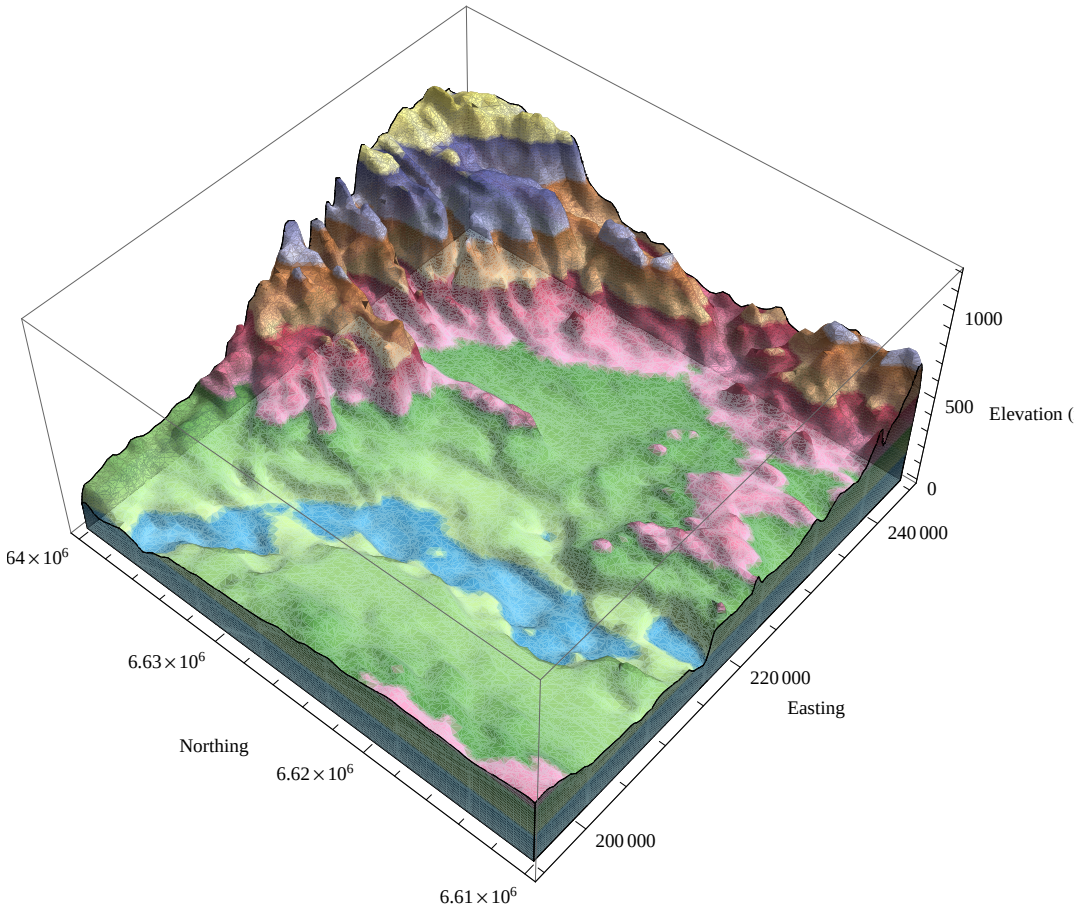
Grid the Rock Surface Data

To define the base of the aquifer the rock surface data are gridded using the function **InverseDistanceGrid2D**.

We can then highlight the pre-Tertiary paleochannel (the base of the alluvial aquifer) by using the function **RegionPlot3D**.

The predicate is: z < rockfunction[x, y]

```
ksearch = 3; power = 2;
rockfunction = InverseDistanceGrid2D[
    rocksurfacepoints, ksearch, power, grid2D];
```



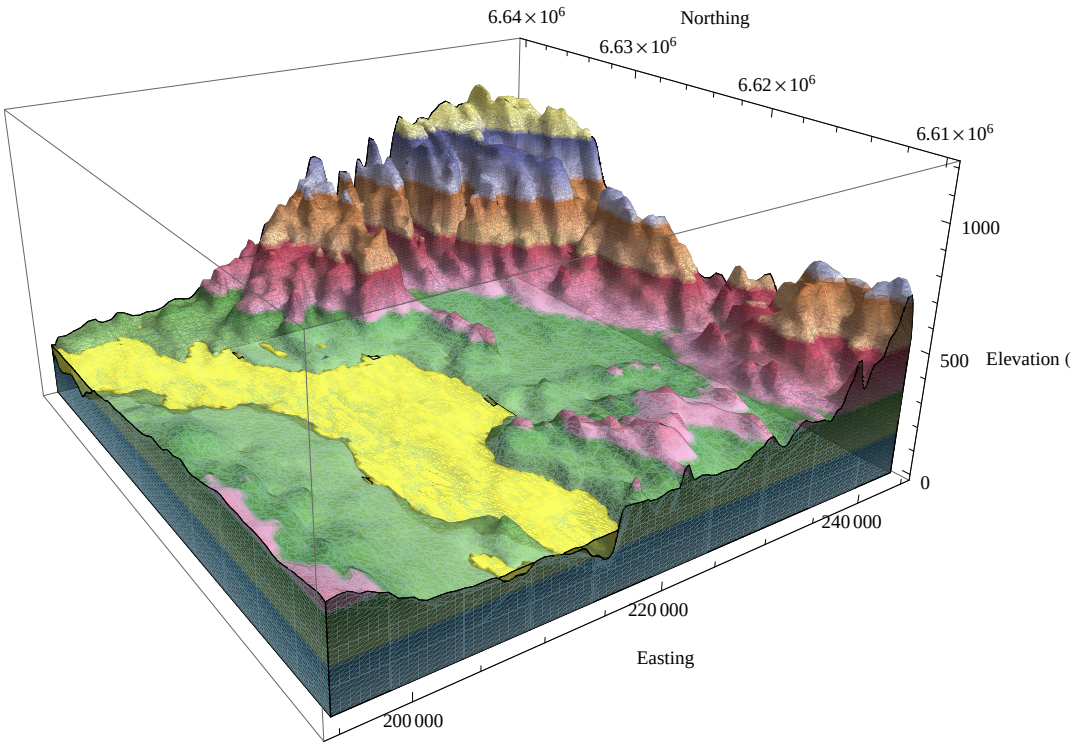
A RegionPlot3D Image of the 3D Surface Model

The sorted alluvial data set is gridded using the new function **NearestNeighborGrid2D**.

```
alluviumfunction =
  NearestNeighborGrid2D[alluvium, grid2D];
```

Now that each surface is defined by a continuous approximate function over the x,y domain a 3D plot of the rock and alluvium can be calculated using **RegionPlot3D**.

The predicate for the alluvium is: z < alluviumfunction[x, y] && z > rockfunction[x, y]

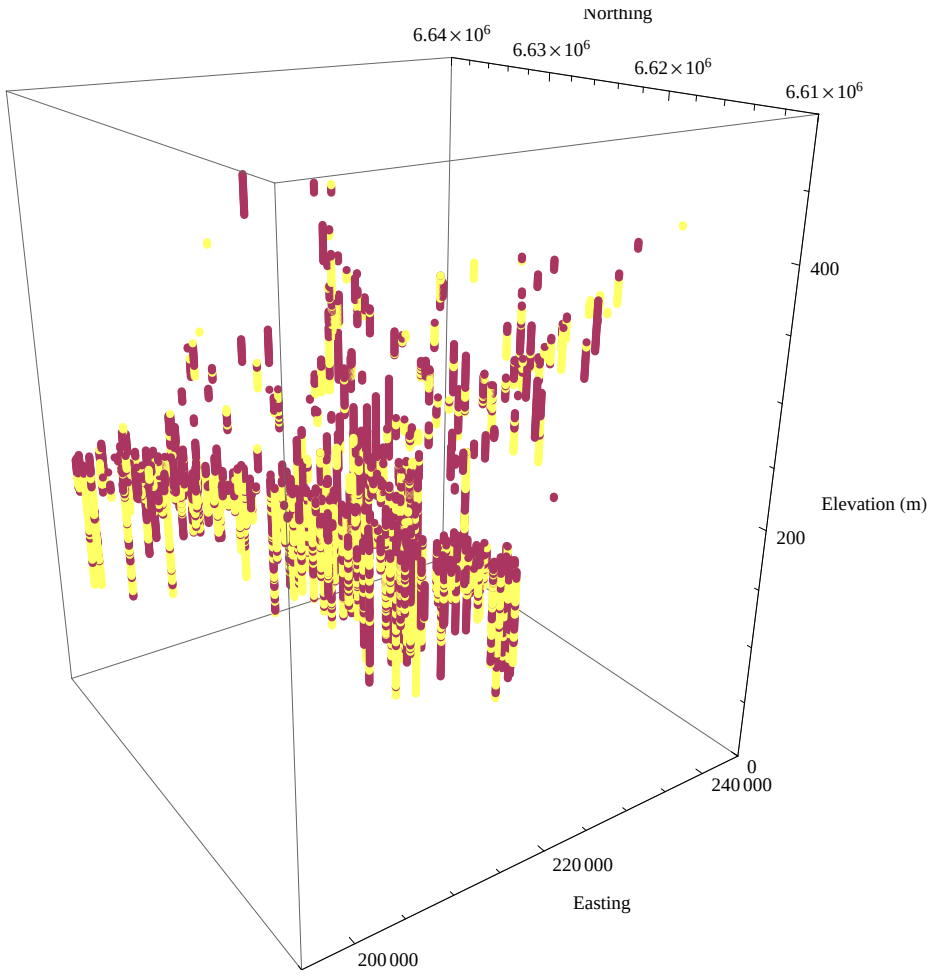


Import the Borehole Facies Data

The next step is to fill in the alluvial region with the facies data.

For this example the clay is one class (purple) and the high water yielding sands and gravels are the second class (yellow).

```
faciesdata = Drop[Import["maulesfacies.csv"], 1];
clay = Cases[faciesdata, {_, _, _, 1}];
sandgravel = Cases[faciesdata, {_, _, _, 2}];
ListPointPlot3D[
 {clay[[All, {1, 2, 3}]], sandgravel[[All, {1, 2, 3}]]},
 PlotRange → {{xmin, xmax}, {ymin, ymax}, {zmin, 500}},
 PlotStyle → {Directive[RGBColor[0.666667,
     0.207843, 0.380392], PointSize[0.01]],
   Directive[Lighter[Yellow, 0.4], PointSize[0.01]]},
 AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 ViewPoint → {-1.0, -1.5, 0.75},
 BoxRatios → {1, 1, 1.25}, ImageSize → {450, 450} ]
```

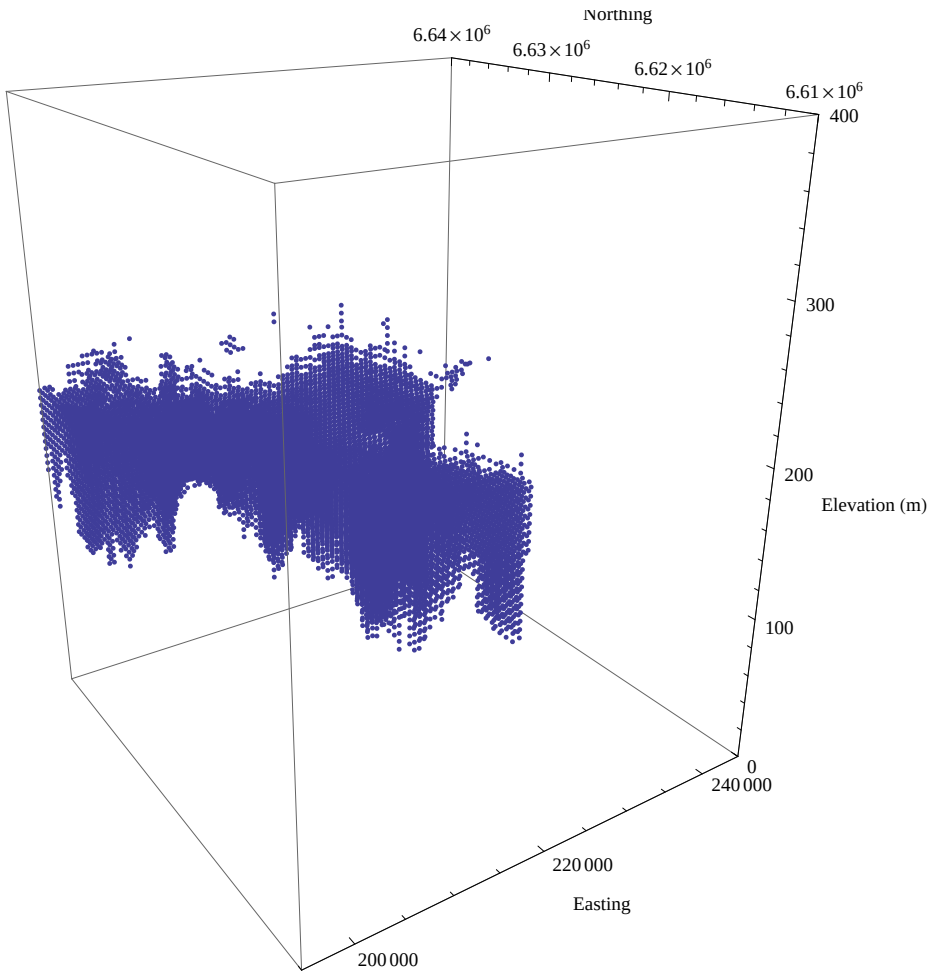


Defining the Dimensions of the 3D Grid

The regular 3D grid list, {{x_1,y_1,z_1},{x_2,y_2,z_2}, ... }, is defined and the nodes limited to those that fall within the limits of the alluvium.

```
grid3D = Flatten[Table[{x, y, z}, {x, xmin, 225 000, 500},
    {y, ymin, ymax, 500}, {z, zmin, 400, 5}], 2];
sort[{x_, y_, z_}] := If[z > rockfunction[x, y] &&
    z < alluviumfunction[x, y], 1, 0];
inout = Map[sort, grid3D];
inoutlist =
  Partition[Flatten[Riffle[grid3D, inout]], 4];
insidenodes = Cases[inoutlist, {_, _, _, 1}][[
   All, {1, 2, 3}]];
```

```
ListPointPlot3D[insidenodes,
 PlotRange → {{xmin, xmax}, {ymin, ymax}, {zmin, 400}},
 AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 PlotStyle → PointSize[0.006],
 ViewPoint → {-1.0, -1.5, 0.75},
 BoxRatios → {1, 1, 1.25}, ImageSize → {450, 450}]
```



3D Nearest Neighbor Gridding of Borehole Facies Data

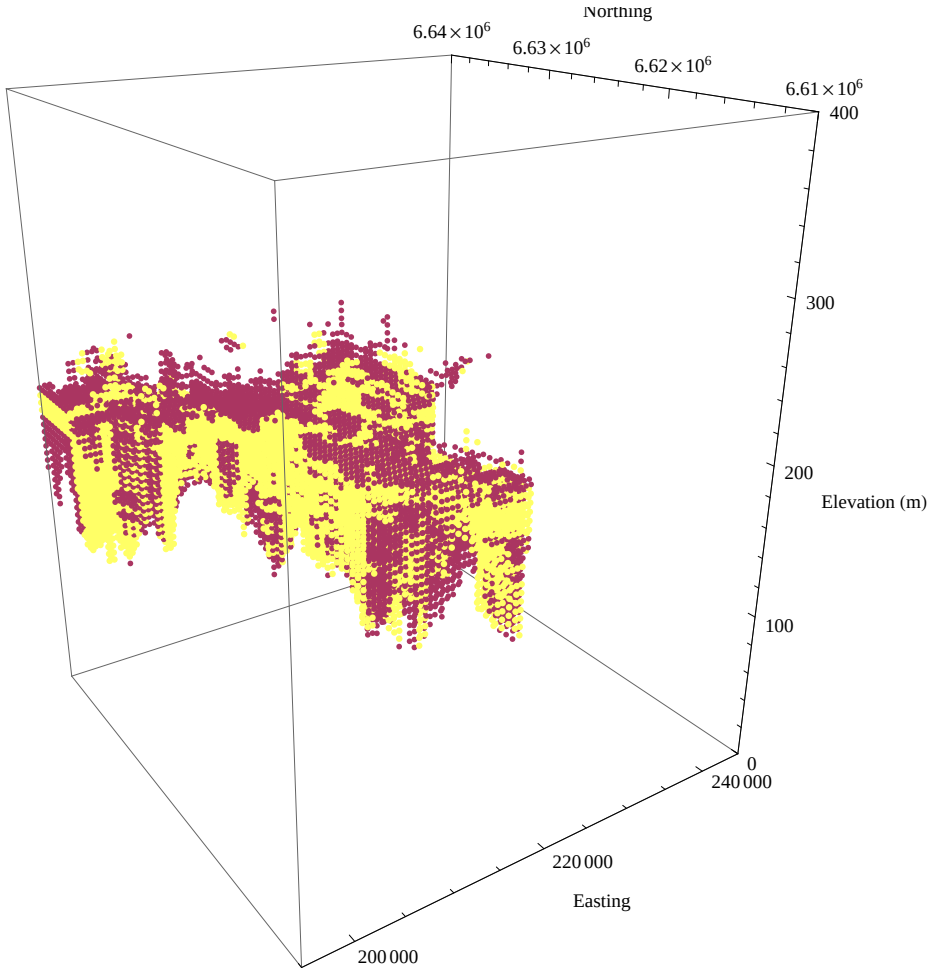
The facies class assigned to each grid node is estimated from the borehole facies data using 3D nearest neighbor gridding.

This simple categorical classification technique has proven to be a robust method in many sedimentary environments (Tartakovsky, 2007).

Gridding the facies data enhances the visual continuity of the sand/gravel and clay lenses.

```
faciesfunction = Nearest[
   faciesdata[[All, {1, 2, 3}]] → faciesdata[[All, 4]]];
faciesNNlist = Map[faciesfunction, insidenodes][[
   All, 1]];
faciesNNgrid3Dinside = Partition[
   Flatten[Riffle[insidenodes, faciesNNlist]], 4];
```

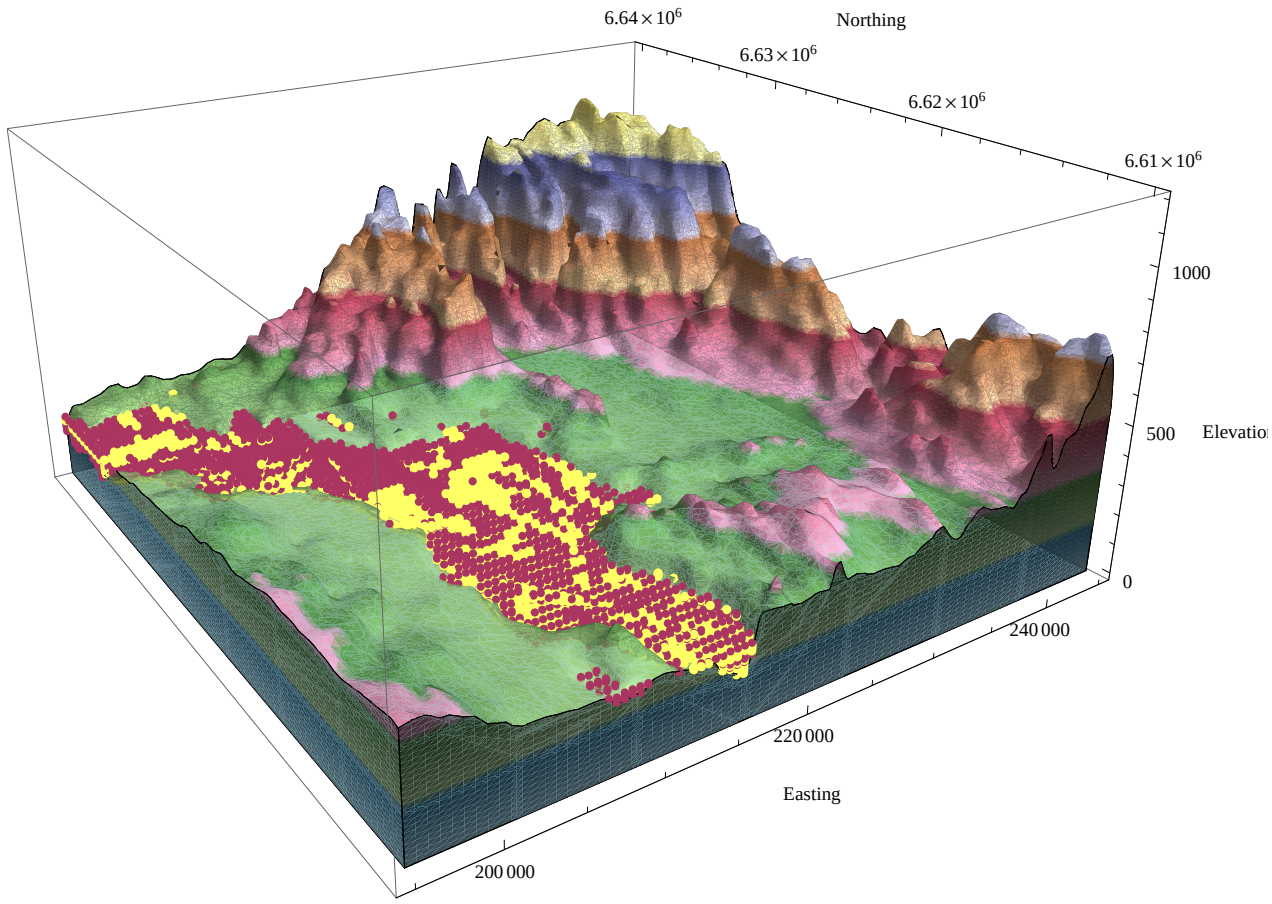
```
clayNN = Cases[faciesNNgrid3Dinside, {_, _, _, 1}];
sandgravelNN =
  Cases[faciesNNgrid3Dinside, {_, _, _, 2}];
faciespoints3D = ListPointPlot3D[
  {clayNN[[All, {1, 2, 3}]],
   sandgravelNN[[All, {1, 2, 3}]]},
  PlotRange → {{xmin, xmax}, {ymin, ymax}, {zmin, 400}},
  PlotStyle →
   {Directive[RGBColor[0.666667, 0.207843, 0.380392],
     PointSize[0.007]], Directive[
     Lighter[Yellow, 0.4], PointSize[0.008]]},
  BoxRatios → {1, 1, 1.25}, AxesLabel →
   {"Easting", "Northing", "Elevation (m)"},
  ViewPoint → {-1.0, -1.5, 0.75},
  ImageSize → {450, 450}]
```



3D Geological Facies Model

Displayed below is the final facies model. The facies data points fill the paleovalley. This model can be exported into groundwater flow modeling programs.

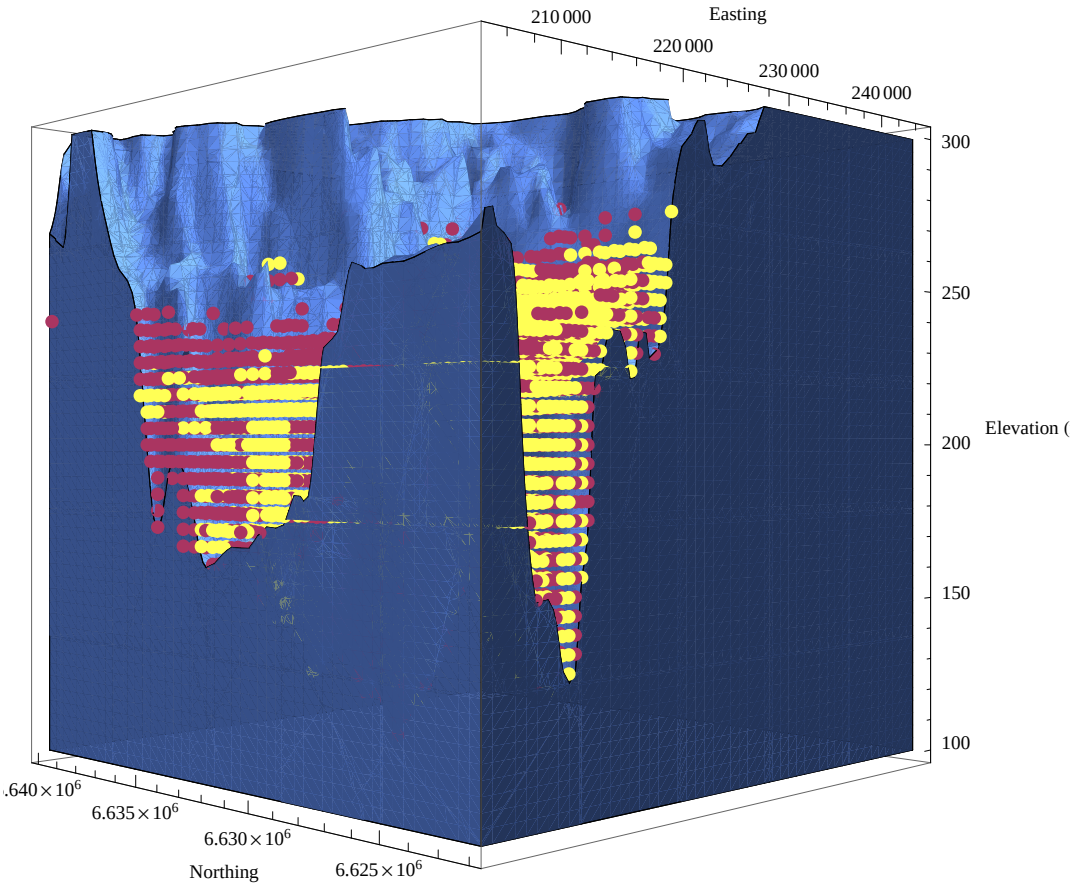
```
Show[rockregionplot3D, faciespoints3D,
 ViewPoint → {-1.0, -1.5, 0.75},
 ImageSize → {600, 550} ]
```



Slice the 3D Facies Model

The 3D facies model can be sliced to examine the facies distribution along lines of interest.

```
xslice = 205 000; yslice = 6 622 000;
ztop = 300; zbottom = 100;
faciesslice =
  ListPointPlot3D[{clayNN[[All, {1, 2, 3}]],
    sandgravelNN[[All, {1, 2, 3}]]}, PlotRange →
    {{xslice, xmax}, {yslice, ymax}, {zbottom, ztop}},
   PlotStyle → {Directive[RGBColor[0.666667,
       0.207843, 0.380392], PointSize[0.015]],
     Directive[Lighter[Yellow], PointSize[0.015]]}];
rockslice = RegionPlot3D[z < rockfunction[x, y],
   {x, xslice, xmax}, {y, yslice, ymax},
   {z, zbottom, ztop}, Mesh → None, PlotPoints → 50,
   PlotStyle → RGBColor[0.392157, 0.576471, 1]];
Show[rockslice, faciesslice,
 AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 Lighting → "Neutral", ViewPoint → {-2, -2, 0},
 ImageSize → {500, 500} ]
```


210 000
Easting
220 000
230 000
240 000
300
250
200
150
100
Elevation (
.640 × 10^6
6.635 × 10^6
6.630 × 10^6
6.625 × 10^6
Northing

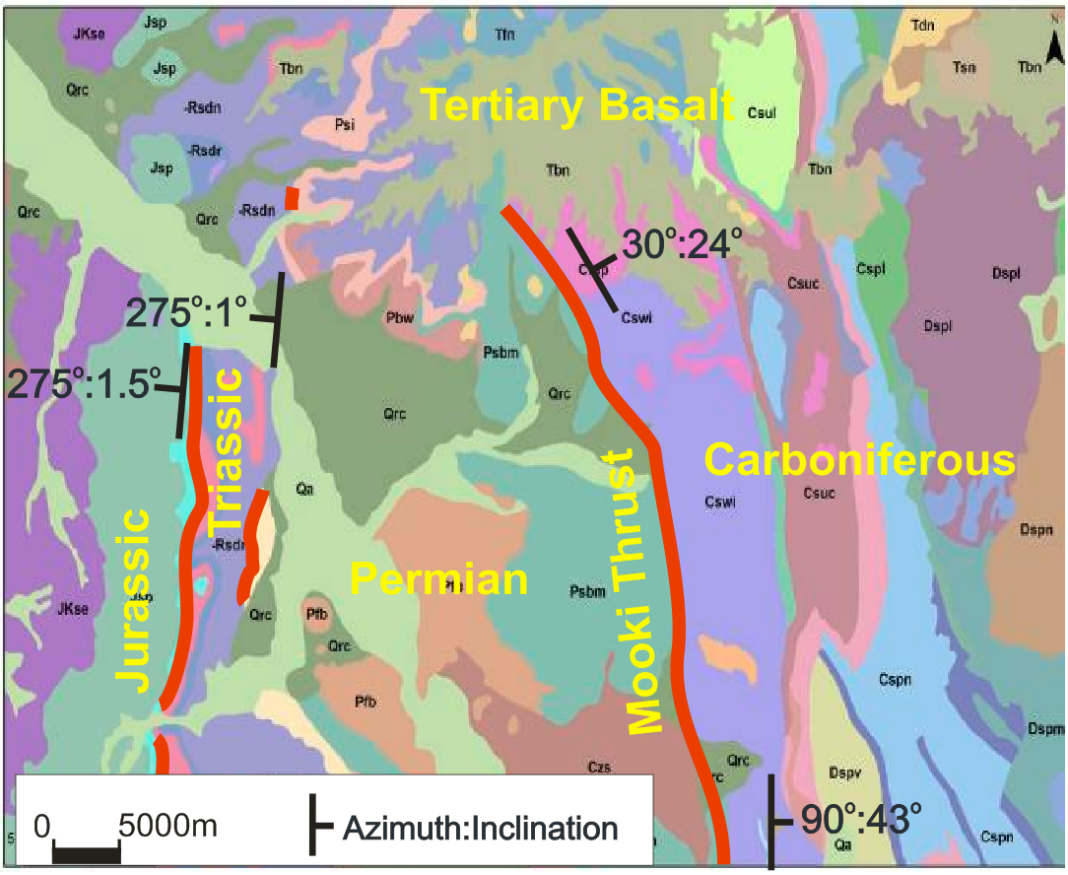
Building the Geological Structural Model

Maules Creek Geological Map

Below is the geological map of Maules Creek.

The major geological periods are labeled in yellow on the map. The red lines show portions of the map where details of the strike and dip of the contact surface have been obtained. The outline of the Tertiary basalt was also digitized.

From this sparse data set a 3D geological structural model based on the geological period boundaries will be constructed.



Import the Geological Structural Data

At selected locations on the upper surface of geological structures the dip inclination and dip azimuth are recorded. The structures can be geological period boundaries, the top of formations, or fault surfaces.

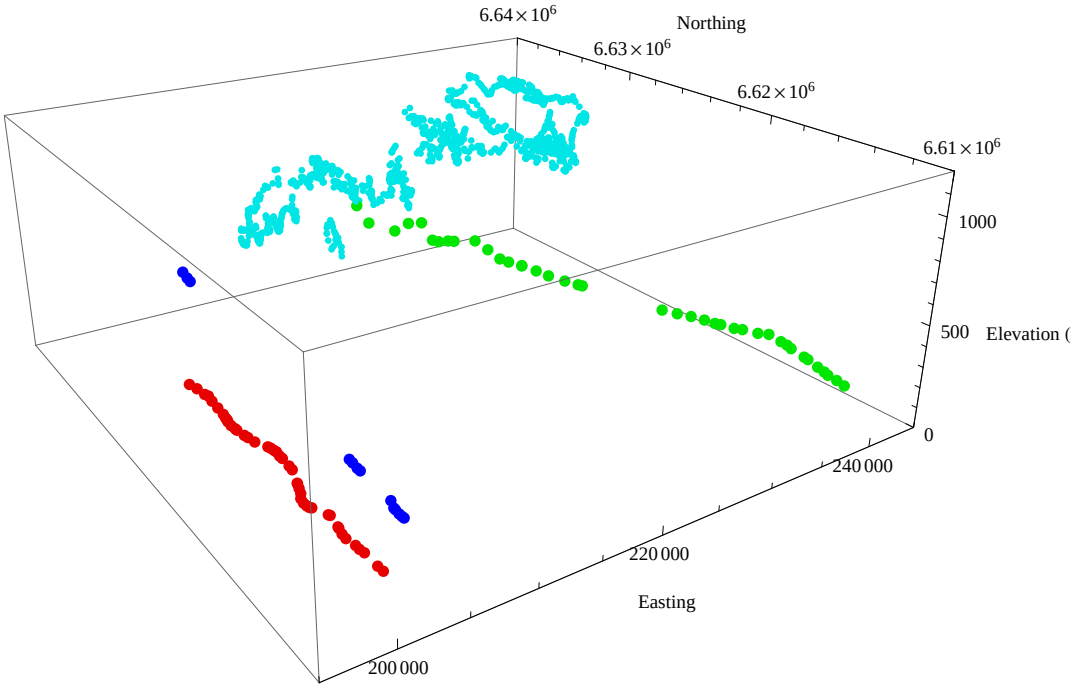
These data sets provide sparse details on the shape of the geological surfaces that define the volume of the geological elements of interest.

For the period boundaries and fault surface data sets at each point the coordinates (x,y,z), azimuth and inclination are loaded. Azimuth values range from 0 to 360, north is 0, east is 90. Inclination values range from 0 to 90, where 0 is horizontal and 90 is vertically down.

The base of the Tertiary basalt is defined by projecting the digitized map points onto the DEM. Each point representing the outline of the Tertiary basalt boundary is then defined by the triplet {x,y,z}.

Below the structural data for the Maules Creek catchment are imported and displayed.

```
topTriassic = Drop[Import["TopTriassic.csv"], 1];
topPermian = Drop[Import["TopPermian.csv"], 1];
faultMookiThrust = Drop[Import["MookiThrust.csv"], 1];
bottomTertiaryBasalt =
  Drop[Import["BottomTertiaryBasalt.csv"], 1];
pointsTertiaryBasalt =
  bottomTertiaryBasalt[[All, {2, 3, 4}]];
ListPointPlot3D[{topTriassic[[All, {2, 3, 4}]],
  topPermian[[All, {2, 3, 4}]],
  faultMookiThrust[[All, {2, 3, 4}]],
  pointsTertiaryBasalt}, PlotRange →
  {{xmin, xmax}, {ymin, ymax}, {zmin, zmax}}, PlotStyle →
  {Directive[Darker[Red, 0.1], PointSize[0.012]],
   Directive[Blue, PointSize[0.012]],
   Directive[Darker[Green, 0.1], PointSize[0.012]],
   Directive[Darker[Cyan, 0.1], PointSize[0.007]]},
 AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 ViewPoint → {-1.0, -1.5, 0.75},
 ImageSize → {500, 400}]
```


6.64×10^6
Northing
6.63×10^6
6.62×10^6
6.61×10^6
1000
500
Elevation (
0
240 000
220 000
Easting
200 000

Projection of Field and Map Data

Starting at the measurement point (x_0,y_0,z_0), the n-th point (x_n,y_n,z_n) projected along the linear line in the down dip direction is calculated using the trigonometric equations:

$$x_n = x_0 + \text{Cos[inclination] Sin[azimuth] n s},$$
$$y_n = y_0 + \text{Cos[inclination] Cos[azimuth] n s},$$
$$z_n = z_0 - \text{Sin[inclination]},$$

where s is the size of the interval between each point.

A set of points is projected both up and down dip from the field or digitized data location.

The function **GeoPointProjectLine** is defined and applied to each of the structural data sets.This process is described in more detail in de Kemp (1998), Bistacchi et al. (2008) and Zanchi et al. (2009).

```
GeoPointProjectLine[geodatain_, stepin_, countin_] :=
 Module[
  {step, projectdown, projectup, projectiontable},
  projectdown = Flatten[N[Table[{
       geodatain[[i, 2]] + (Cos[geodatain[[i, 5]] °] *
           Sin[geodatain[[i, 6]] °]) stepin n,
       geodatain[[i, 3]] + (Cos[geodatain[[i, 5]] °] *
           Cos[geodatain[[i, 6]] °]) stepin n,
       geodatain[[i, 4]] - Sin[geodatain[[i, 5]] °]
         stepin n},
      {i, Length[geodatain]}, {n, 0, countin}]], 1];
  projectup = Flatten[N[Table[{
       geodatain[[i, 2]] - (Cos[geodatain[[i, 5]] °] *
           Sin[geodatain[[i, 6]] °]) stepin n,
       geodatain[[i, 3]] - (Cos[geodatain[[i, 5]] °] *
           Cos[geodatain[[i, 6]] °]) stepin n,
       geodatain[[i, 4]] + Sin[geodatain[[i, 5]] °]
         stepin n},
      {i, Length[geodatain]}, {n, 0, countin}]], 1];
  projectiontable = Join[projectup, projectdown];
  Return[projectiontable]]
```

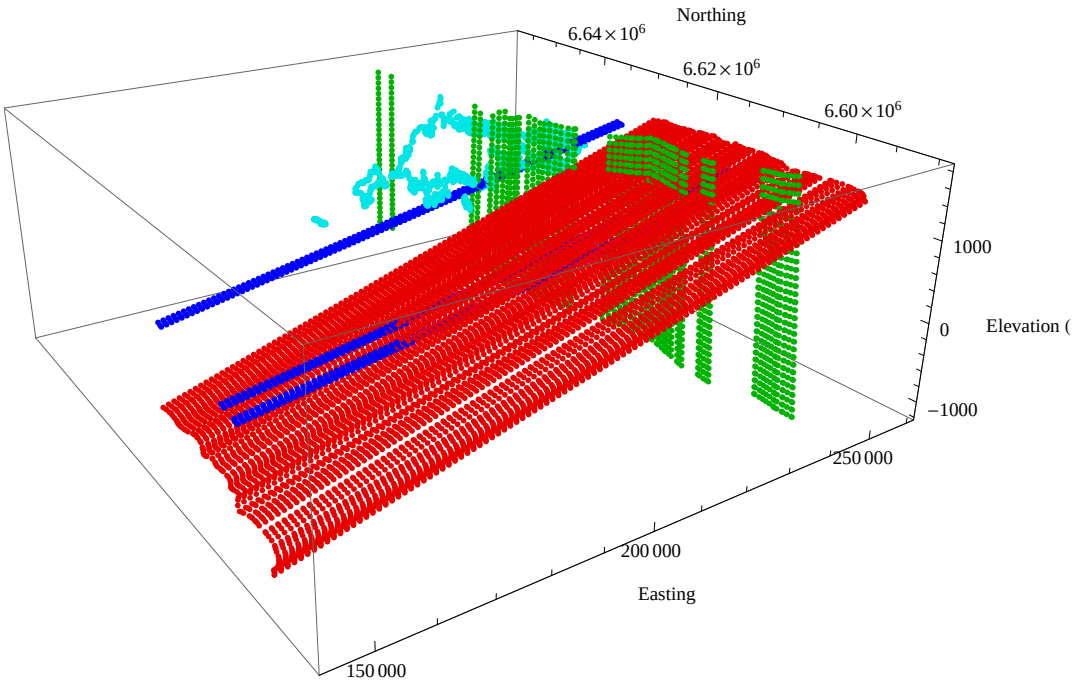
The Projected Data

Below is a 3D plot of the projected data after the function **GeoPointProjectLine** has been applied to the sparse geological structural data sets.

Data points are projected beyond the domain of interest to control the shape of the grids to be calculated.

```
step = 1000; count = 50;
linesTriassic =
  GeoPointProjectLine[topTriassic, step, count];
linesPermian = GeoPointProjectLine[
   topPermian, step, count];
linesMookiThrust = GeoPointProjectLine[
   faultMookiThrust, 200, 15];
```

```
ListPointPlot3D[{linesTriassic, linesPermian,
  linesMookiThrust, pointsTertiaryBasalt},
 PlotRange → {{xmin, xmax}, {ymin, ymax}}, AxesLabel →
  {"Easting", "Northing", "Elevation (m)"}, PlotStyle →
  {Directive[Darker[Red, 0.1], PointSize[0.006]],
   Directive[Blue, PointSize[0.006]],
   Directive[Darker[Green, 0.3], PointSize[0.006]],
   Directive[Darker[Cyan, 0.1], PointSize[0.006]]},
 ViewPoint → {-1.0, -1.5, 0.75},
 ImageSize → {500, 450}]
```



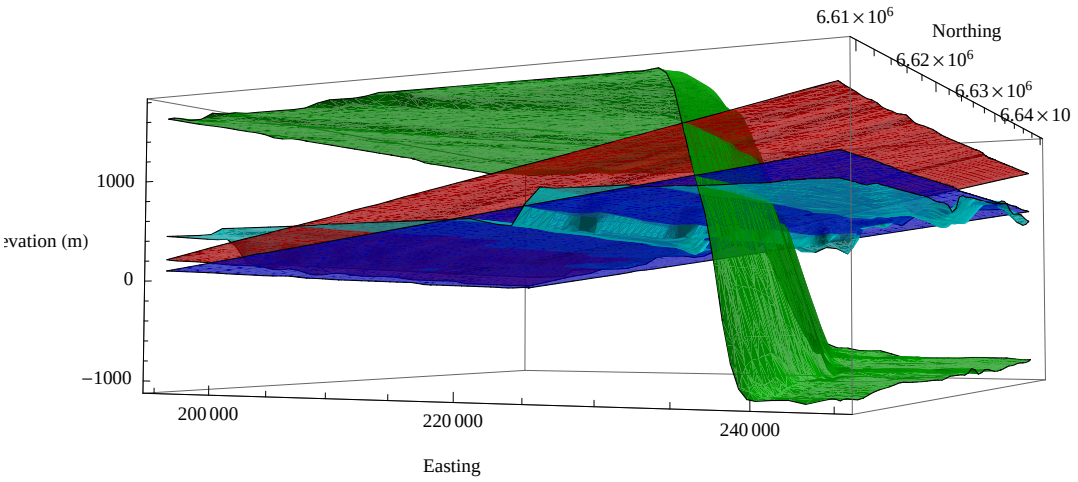
Grid the Projected Data

Each projected data set can now be gridded using the new functions **NearestNeighborGrid2D** and **InverseDistanceGrid2D**.

The Tertiary basalt data are gridded using nearest neighbor, while the projected data sets that define the structural tops of the Jurassic and Permian, along with the Mooki Thrust are gridded using inverse distance.

```
ksearch = 3; power = 2;
grid2Dc = Partition[Flatten[Table[{x, y},
      {x, xmin, xmax, 1000}, {y, ymin, ymax, 1000}]], 2];
fPermian = InverseDistanceGrid2D[linesPermian,
   ksearch, power, grid2Dc];
fTriassic = InverseDistanceGrid2D[
   linesTriassic, ksearch, power, grid2Dc];
fMookiThrust = InverseDistanceGrid2D[
   linesMookiThrust, ksearch, power, grid2Dc];
fTertiaryBasalt = NearestNeighborGrid2D[
   pointsTertiaryBasalt, grid2Dc];
```

```
Plot3D[{fTriassic[x, y], fPermian[x, y],
  fMookiThrust[x, y], fTertiaryBasalt[x, y]},
 {x, xmin, xmax}, {y, ymin, ymax},
 PlotStyle → {Darker[Red, 0.1], Blue,
   Darker[Green, 0.3], Darker[Cyan, 0.1]},
 PlotPoints → 50, Mesh → None,
 AxesLabel → {"Easting", "Northing", "Elevation (m)"},
 ViewPoint → {1, -2, -0.1}, ImageSize → {500, 300},
 Lighting → {{"Directional", GrayLevel[0.95],
    {220 000, 6 620 000, 5000}}, {"Directional",
    GrayLevel[0.8], {350 000, 6 630 000, -500}},
   {"Directional", GrayLevel[0.8],
    {210 000, 6 660 000, -5000}}}]
```



Constructing the Geological Regions

Each region of the geological model is defined using the function **RegionPlot3D**.

Using a set of relational and logical operators for the predicate the geological history is replicated.

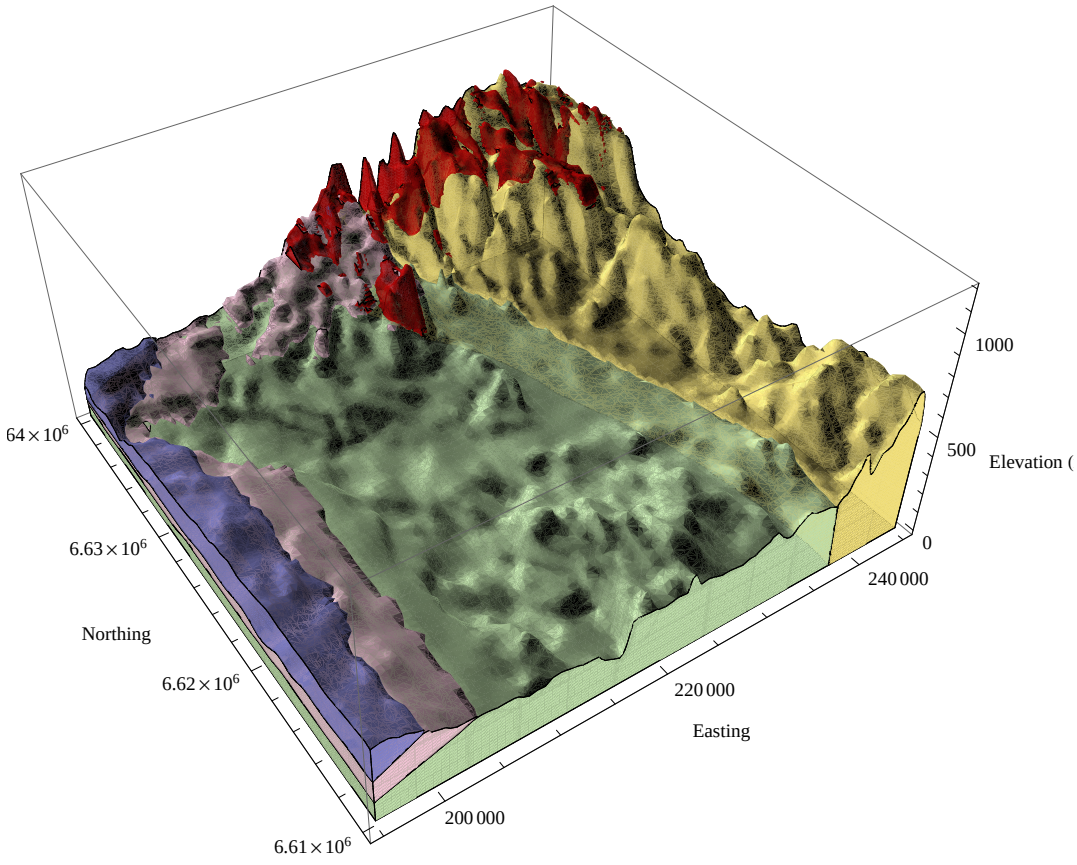
```
regionTertiaryBasalt = RegionPlot3D[
   z > fTertiaryBasalt[x, y] && z < rockfunction[x, y],
   {x, 215 000, xmax}, {y, 6 630 000, ymax}, {z, zmin,
    zmax}, Mesh → None, PlotPoints → 75, PlotStyle →
    RGBColor[0.729412, 0.0392157, 0.0392157]];

regionJurassic = RegionPlot3D[z > fTriassic[x, y] &&
    z < rockfunction[x, y] && z < fMookiThrust[x, y],
   {x, xmin, xmax}, {y, ymin, ymax}, {z, zmin, zmax},
   Mesh → None, PlotPoints → 75,
   PlotStyle → RGBColor[0.470588, 0.470588, 0.737255]];

regionTriassic = RegionPlot3D[z < fTriassic[x, y] &&
    z > fPermian[x, y] && z < rockfunction[x, y] &&
    z < fTertiaryBasalt[x, y] && z < fMookiThrust[x, y],
   {x, xmin, xmax}, {y, ymin, ymax}, {z, zmin, zmax},
   Mesh → None, PlotPoints → 75, PlotStyle →
    Lighter[RGBColor[0.823529, 0.576471, 0.670588]]];

regionPermian = RegionPlot3D[
   z < fPermian[x, y] && z < rockfunction[x, y] &&
    z < fTertiaryBasalt[x, y] && z < fMookiThrust[x, y],
   {x, xmin, xmax}, {y, ymin, ymax}, {z, zmin, zmax},
   Mesh → None, PlotPoints → 75, PlotStyle →
    Lighter[RGBColor[0.556863, 0.733333, 0.443137]]];

regionMookiThrust =
  RegionPlot3D[z > fMookiThrust[x, y] &&
    z < rockfunction[x, y] && z < fTertiaryBasalt[x, y],
   {x, xmin, xmax}, {y, ymin, ymax}, {z, zmin, zmax},
   Mesh → None, PlotPoints → 75,
   PlotStyle → RGBColor[0.894118, 0.827451, 0.454902]];
```

3D Geological Structural Model

The above calculated, but not displayed, geological period region images are combined into the single 3D geological structural model shown below.

Each geological period is represented as a solid color. The color key used below is: Tertiary basalt (red), Jurassic (blue), Triassic (pink), Permian (green), and Carboniferous (yellow).

```
structuralmodel =
 Show[regionJurassic, regionTriassic, regionPermian,
  regionMookiThrust, regionTertiaryBasalt,
  AxesLabel → {"Easting", "Northing", "Elevation (m)"},
  BoxRatios → {1, 1, 0.5},
  ViewPoint → {-1.0, -1.5, 1.5},
  ImageSize → {500, 450},
  Lighting → {{"Directional", GrayLevel[0.95],
     {220 000, 6 600 000, 500}}, {"Directional",
     GrayLevel[0.8], {150 000, 6 630 000, 500}},
    {"Directional", GrayLevel[0.8],
     {210 000, 6 660 000, 5000}}}]
```



3D Geological Structural and Facies Model

The last step is to combine the facies and structural models into a single 3D geological model.

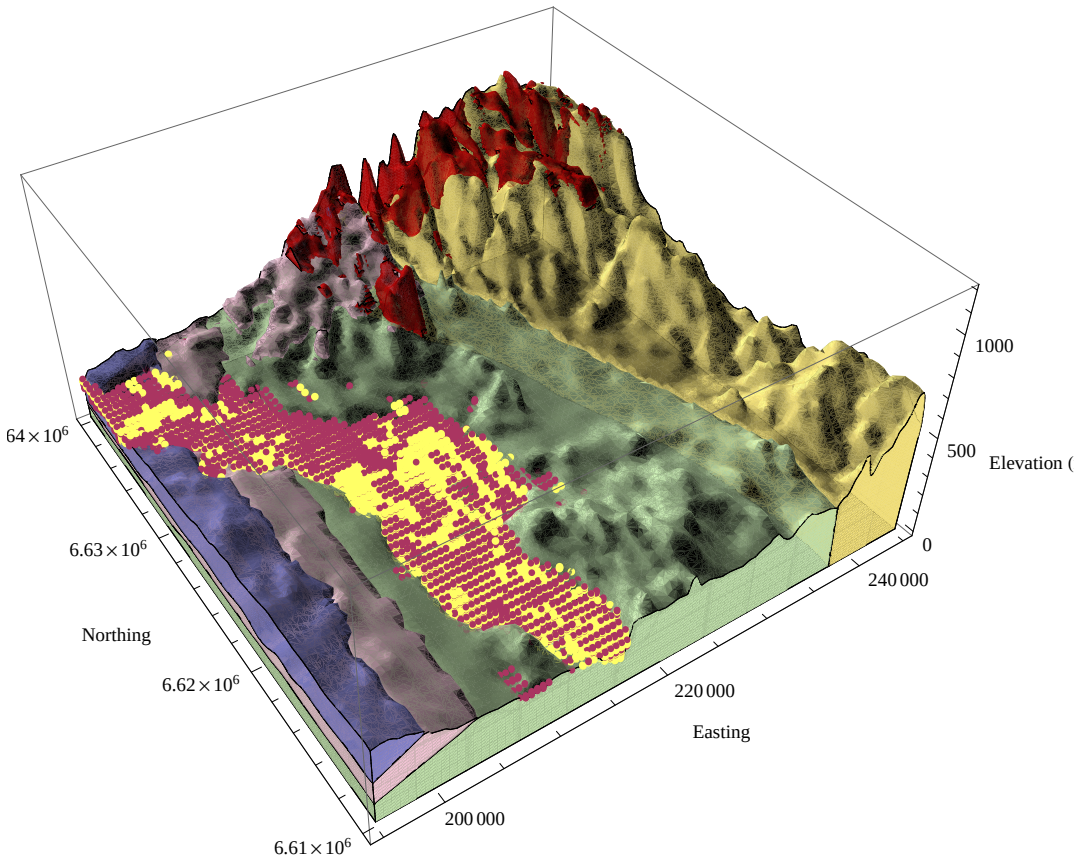
These models are the starting point for further geological investigations. They can be used to:

- present our conceptualization of the regional geological history,
- determine the volume of the aquifer,
- predict the lithology at new borehole locations, and
- provide the framework for regional water balance modeling.

As more details are obtained about the catchment the models can be easily updated or refined.

Finally, collating the information into 3D images allows for the easy communication of geological information to the wider community.

```
Show[structuralmodel, faciespoints3D]
```



Acknowledgments

The financial support of the Australian Government National Water Commission, Australian Research Council, Cotton Catchment Communities CRC and the Namoi Catchment Management Authority is acknowledged.

I would also like to thank Dr B. Giambastiani for her assistance with preparing the data sets used in this presentation, and Professor I. Acworth for the coordination and administration of the grants.

References

Bistacchi A., Massironi M., Dal Piaz G.V., Dal Piaz G., Monopoli B., Schiavo A., Toffolon G. (2008) 3D fold and fault reconstruction with an uncertainty model: An example from an Alpine tunnel case study. *Computers and Geosciences,* 34(4): 351-372.

de Kemp E.A. (1998) Three-dimensional projection of curvilinear geological features through direction cosine interpolation of structural field observations. *Computers and Geosciences,* 24(3): 269-284.

Haneberg, W.C. (2004) *Computational Geosciences with Mathematica*. Springer-Verlag Berlin Heidelberg. ISBN 3-540-40245-4.

Tartakovsky D.M., Wohlberg B., Guadagninni A. (2007) Nearest-neighbor classification for facies delineation. *Water Resources Research,* 43 DOI 10.1029/2007WR005968.

Yamamoto, J.K. (1998) A review of numerical methods for the interpolation of geological data. *An. Acad. Bras. Ci.*, 70(1): 91-116.

Zanchi A., Francesca S., Stefano Z., Simone S., Graziano G. (2009) 3D reconstruction of complex geological bodies: Examples from the Alps. *Computers and Geosciences,* 35(1): 49-69.