2.4.5 The SAIGUP Model

Most commercial simulators use a combination of an 'input language' and a set of data files to describe and set up a simulation model of a reservoir. However, although the principles for the input description has much in common, the detail syntax is obviously unique to each simulator. Herein, we will mainly focus on the ECLIPSE input format, which has emerged as an industry standard for describing static and dynamic properties of a reservoir system, from the reservoir rock, via production and injection wells and up to connected top-side facilities. ECLIPSE input decks use keywords to signify and separate the different data elements that comprise a full model. These keywords define a detailed language that can be used to specify how the data elements should be put together, and modify each other, to form a full spatio-temporal model of a reservoir. In the most general form, an ECLIPSE input file consists of eight sets of keywords are organized into eight sections that must come in a prescribed order. However, some of the sections are optional and may not always be present. The order of the keywords within each section is arbitrary, except in the section that defines wells and gives operating schedule, etc. Altogether, the ECLIPSE format consists of thousands of keywords, and describing them all is far beyond the scope of this presentation.

In the following, we will instead briefly outline some of the most common keywords that are used in the GRID section that describes the reservoir geometry and petrophysical properties. The purpose is to provide you with a basic understanding of the required input for simulations of real-life reservoir models. Our focus is mainly on the ingredients of a model and not on the specific syntax. For brevity, we will therefore not go through all Matlab and MRST statements used to visualize the different data elements. All details necessary to reproduce the results can be found in the script `rocks/showSAIGUP.m`.

As an example of a realistic representation of a shallow-marine reservoir, we will use a simulation model taken from the SAIGUP study [40]. The SAIGUP models mainly focus on shoreface reservoirs in which the deposition of sediments is caused by variation in sea level, so that facies are forming belts in a systematic pattern (river deposits create curved facies belts, wave deposits create parallel belts, etc). Sediments are in general deposited when the sea level is increasing. No sediments are deposited during decreasing sea levels; instead, the receding sea may affect the appearing shoreline and cause the creation of a barrier.

Assuming that the archive file `SAIGUP.tar.gz` that contains the model realization has been downloaded as described in Section 1.5, we used MATLAB's `untar` function to extract the data set and place it in a standardized path relative to the root directory of MRST:

```
untar('SAIGUP.tar.gz', fullfile(ROOTDIR, 'examples', 'data', 'SAIGUP'))
```

This will create a new directory that contains seventeen data files that comprise the structural model, various petrophysical parameters, etc:

```
028_A11.EDITNNC      028.MULTX  028.PERMX  028.SATNUM       SAIGUP.GRDECL
028_A11.EDITNNC.001  028.MULTY  028.PERMY  SAIGUP_A1.ZCORN
028_A11.TRANX        028.MULTZ  028.PERMZ  SAIGUP.ACTNUM
028_A11.TRANY        028.NTG    028.PORO   SAIGUP.COORD
```

The main file is `SAIGUP.GRDECL`, which lists the sequence of keywords that specifies how the data elements found in the other files should be put together to make a complete model of the reservoir rock. The remaining files represent different keywords: the grid geometry is given in files `SAIGUP_A1.ZCORN` and `SAIGUP.COORD`, the porosity in `028.PORO`, the permeability tensor in the three `028.PERM*` files, net-to-gross properties in `028.NTG`, the active cells in `SAIGUP.ACTNUM`, transmissibility multipliers that modify the flow connections between different cells in the model are given in `028.MULT*`, etc. For now, we will rely entirely on MRST's routines for reading Eclipse input files; more details about corner-point grids and the Eclipse input format will follow later in the book, starting in Chapter 3.

The `SAIGUP.GRDECL` file contains seven of the eight possible sections that may comprise a full input deck. The `deckformat` module in MRST contains a comprehensive set of input routines that enable the user to read the most important keywords and options supported in these sections. Here, however, it is mainly the sections describing static reservoir properties that contain complete and useful information, and we will therefore use the much simpler function `readGRDECL` from MRST core to read and interprets the GRID section of the input deck:

```
grdecl = readGRDECL(fullfile(ROOTDIR, 'examples', ...
   'data',  'SAIGUP','SAIGUP.GRDECL'));
```

This statement parses the input file and stores the content of all keywords it recognizes in the structure `grdecl`:

```
grdecl =
    cartDims: [40 120 20]
       COORD: [29766x1 double]
       ZCORN: [768000x1 double]
      ACTNUM: [96000x1 int32]
       PERMX: [96000x1 double]
       PERMY: [96000x1 double]
       PERMZ: [96000x1 double]
       MULTX: [96000x1 double]
       MULTY: [96000x1 double]
       MULTZ: [96000x1 double]
        PORO: [96000x1 double]
         NTG: [96000x1 double]
      SATNUM: [96000x1 double]
```

The first four data fields describe the grid, and we will come back to these in Chapter 3.3.1. In the following, we will focus on the next eight data fields, which contain the petrophysical parameters. We will also briefly look at the last data field, which delineates the reservoir into different (user-defined) rock types that can used to associated different rock-fluid properties.

MRST uses the strict SI conventions in all of its internal calculations. The SAIGUP model, however, is provided using the Eclipse 'METRIC' conventions

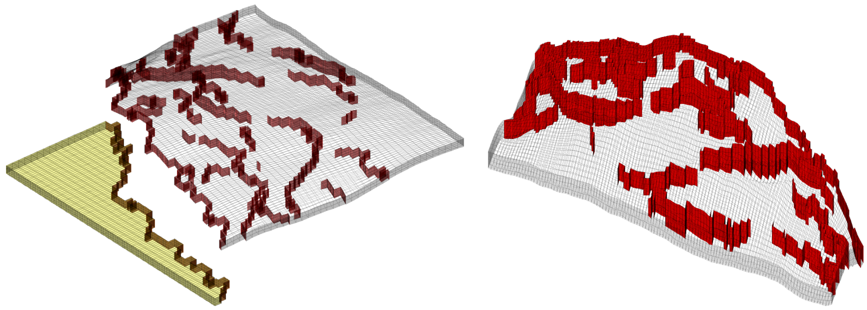


Fig. 2.12. The structural SAIGUP model. The left plot shows the full model with faults marked in red and inactive cells marked in yellow, whereas the right plot shows only the active parts of the model seen from the opposite direction.

(permeabilities in mD and so on). We use the functions `getUnitSystem` and `convertInputUnits` to assist in converting the input data to MRST's internal unit conventions.

```
usys   = getUnitSystem('METRIC');
grdecl = convertInputUnits(grdecl, usys);
```

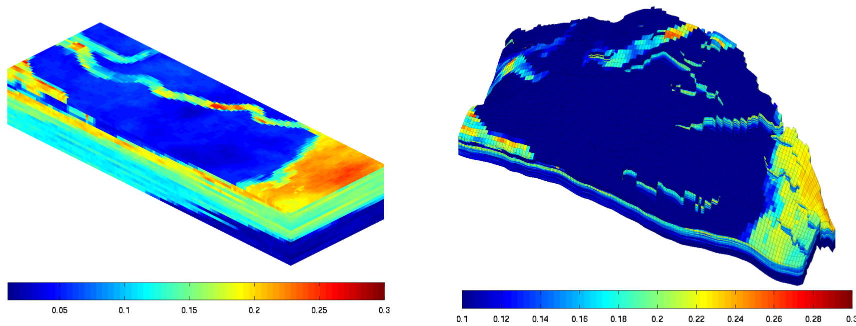
Having converted the units properly, we generate a space-filling grid and extract petrophysical properties

```
G = processGRDECL(grdecl);
G = computeGeometry(G);
rock   = grdecl2Rock(grdecl, G.cells.indexMap);
```

The first statement takes the description of the grid geometry and constructs an unstructured MRST grid represented with the data structure outlined in Section 3.4. The second statement computes a few geometric primitives like cell volumes, centroids, etc., as discussed on page 75. The third statement constructs a rock object containing porosity, permeability, and net-to-gross.

For completeness, we first show a bit more details of the structural model in Figure 2.12. The left plot shows the whole $40 \times 120 \times 20$ grid model[1], where we in particular should note the disconnected cells marked in yellow that are not part of the active model. The relatively large fault throw that disconnects the two parts is most likely a modelling artifact introduced to clearly distinguish the active and inactive parts of the model. A shoreface reservoir is bounded by faults and geological horizons, but faults also appear inside the reservoir as the right plot in Figure 2.12 shows. Faults and barriers will typically have a pronounced effect on the flow pattern, and having an accurate representation is important to produce reliable flow predictions.

[1] To not confuse the reader, we emphasize that only the active part of the model is read with the MRST statements given above. How to also include the inactive part, will be explained in more details in Chapter 3.




```
p = reshape( grdecl.PORO , G.cartDims );
slice(p, 1, 1, 1); view(-135,30), shading flat,
axis equal off,
set(gca,'ydir','reverse','zdir','reverse')
colorbar('horiz'); caxis([0 .01 0.3]);
```

```
args = {'EdgeAlpha'; 0.1; 'EdgeColor'; 'k'};
plotCellData(G, rock.poro, args{:});
axis tight off; set(gca,'DataAspect',[1 1 0.1]);
view(-65,55); zoom(1.4); camdolly(0,-0.2,0)
colorbar('horiz'); caxis([0.1 0.3])
```

Fig. 2.13. Porosity for the SAIGUP model. The left plot shows porosity as generated by geostatistics in logical ijk space. The right plot shows the porosity mapped to the structural model shown in Figure 2.12.

The petrophysical parameters for the model were generated on a regular $40 \times 120 \times 20$ Cartesian grid, as illustrated in the left plot of Figure 2.13, and then mapped onto the structural model, as shown in the plot to the right. A bit simplified, one can view the Cartesian grid model as representing the rock body at geological 'time zero' when the sediments have been deposited and have formed a stack of horizontal grid layers. From geological time zero and up to now, geological activity has introduced faults and deformed the layers, resulting in the structural model seen in the left plot of Figure 2.13.

Having seen the structural model, we continue to study the petrophysical parameters. The grid cells in our model are thought to be larger than the laminae of our imaginary reservoir and hence each grid block will generally contain both reservoir rock (with sufficient permeability) and impermeable shale. This is modelled using the net-to-gross ratio, `rock.ntg`, which is shown in Figure 2.14 along with the horizontal and vertical permeability. The plotting routines are exactly the same as for the porosity in Figure 2.13, but with different data and slightly different specification of the colorbar. From the figure, we clearly see that the model has a large content of shale and thus low permeability along the top. However, we also see high-permeable sand bodies that cut through the low-permeable top. In general, the permeabilities seem to correlate well with the sand content given by the net-to-gross parameter.

Some parts of the sand bodies are partially covered by mud that strongly reduces the vertical communication, most likely because of flooding events. These mud-draped surfaces occur on a sub-grid scale and are therefore modelled through a multiplier value (`MULTZ`) between zero and one that can be used to manipulate the effective communication (the transmissibility) between a given cell (i, j, k) and the cell immediately above $(i, j, k+1)$. For complete-

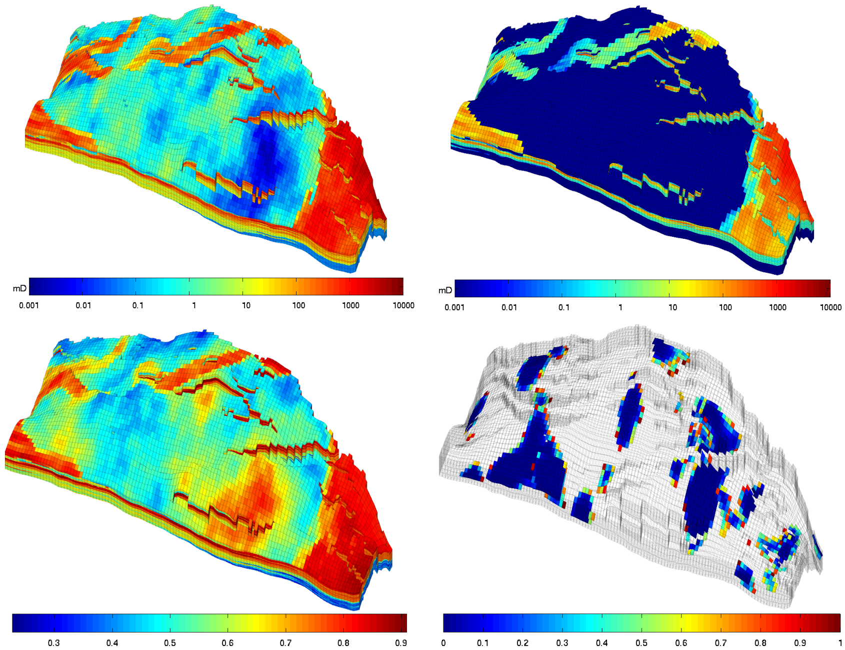


Fig. 2.14. The upper plots show the horizontal (left) and vertical permeability (right) for the SAIGUP model, using a logarithmic color scale. The lower plots show net-to-gross (left) and vertical multiplier values less than unity (right).

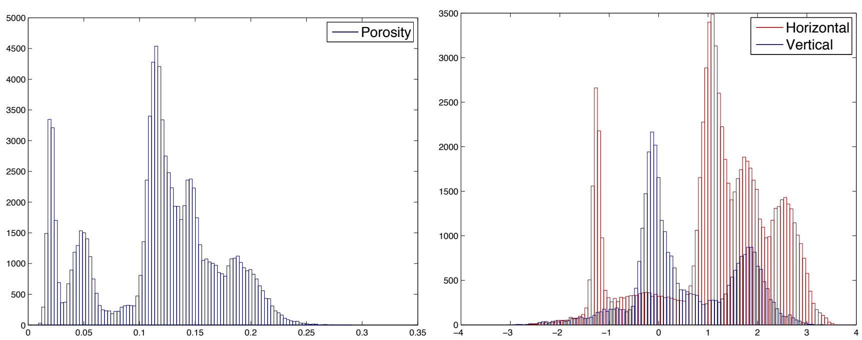
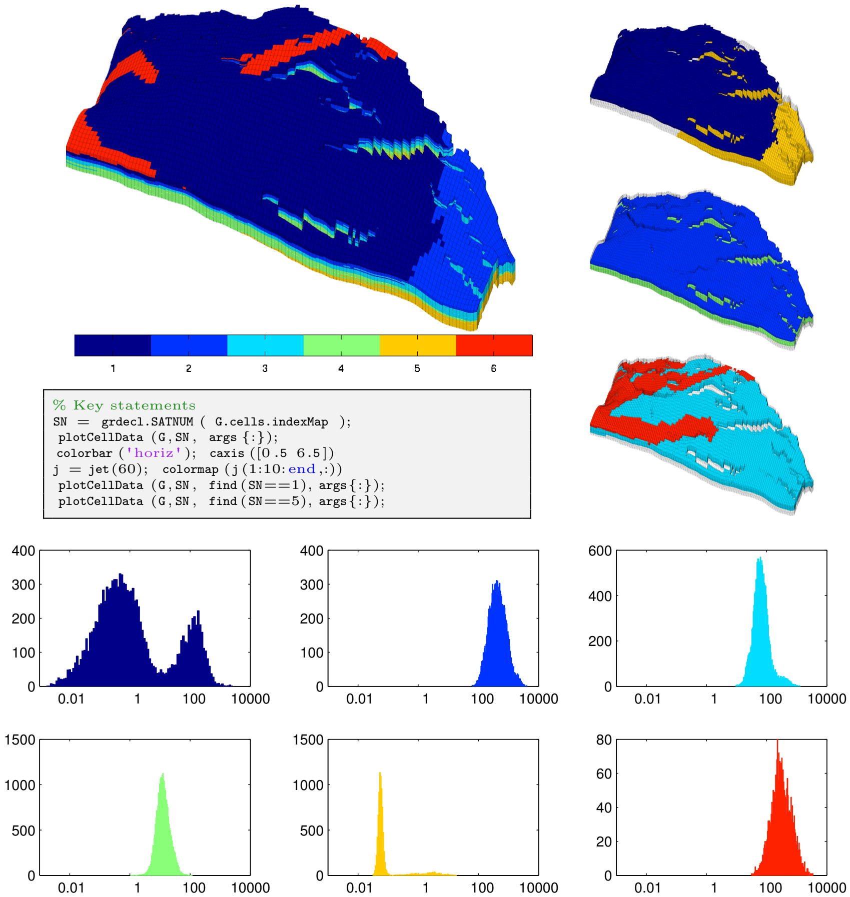


Fig. 2.15. Histogram of the porosity (left) and the logarithm of the horizontal and vertical permeability (right).

ness, we remark that the horizontal multiplier values (`MULTX` and `MULTY`) play a similar role for vertical faces, but are equal one in (almost) all cells for this particular realization.

To further investigate the heterogeneity of the model, we next look at histograms of the porosity and the permeabilities, as we did for the SPE 10 example (the MATLAB statements are omitted since they are almost iden-




```
% Key statements
SN = grdecl.SATNUM ( G.cells.indexMap );
 plotCellData (G,SN, args{:});
 colorbar ('horiz'); caxis ([0.5 6.5])
j = jet(60); colormap (j(1:10:end,:))
 plotCellData (G,SN, find(SN==1), args{:});
 plotCellData (G,SN, find(SN==5), args{:});
```

Fig. 2.16. The upper-left plot shows the rock type distribution for the SAIGUP model. The right column shows the six rock types grouped in pairs; from top to bottom, rock types number 1 and 5, 2 and 4, and 3 and 6. The bottom part of the figure shows histograms of the lateral permeability in units [mD] for each of the six rock types found in the SAIGUP model.

tical). In Figure 2.15, we clearly see that the distributions of porosity and horizontal permeability are multi-modal in the sense that five different modes can be distinguished, corresponding to the five different facies used in the petrophysical modelling.

It is common modelling practice that different rock types are assigned different rock-fluid properties (relative permeability and capillary functions), more details about such properties will be given later in the book. In the Eclipse input format, these different rock types are represented using the SATNUM keyword. By inspection of the SATNUM field in the input data, we see that the model contains six different rock types as depicted in Figure 2.16.

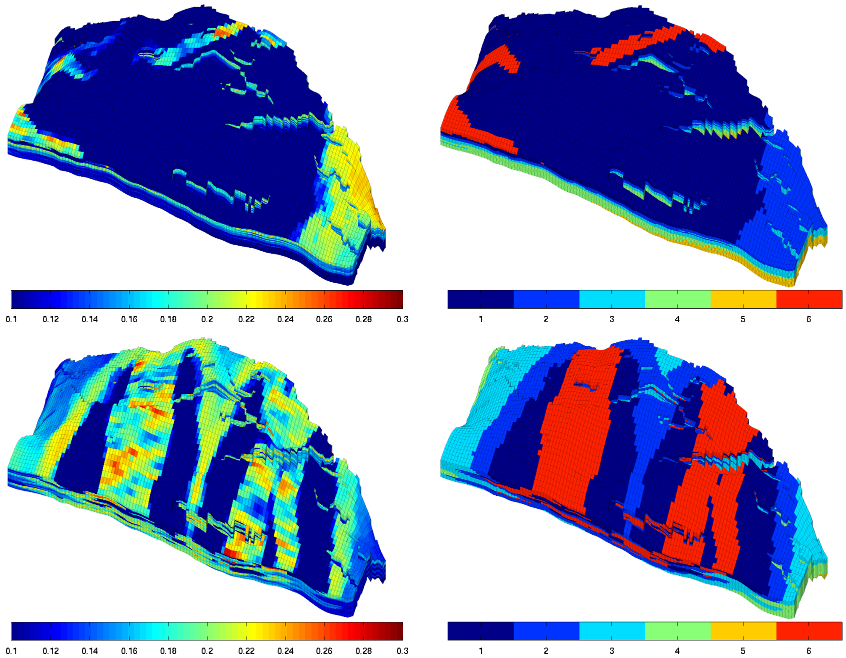


Fig. 2.17. Comparison of porosity (left) and the distribution of rock types (right) for two different SAIGUP realizations.

For completeness, the figure also shows the permeability distribution inside each rock type. Interestingly, the permeability distribution is multi-modal for at least two of the rock types.

Finally, to demonstrate the large difference in heterogeneity resulting from different depositional environment, we compare the realization we have studied above with another realization. In Figure 2.17 we show porosities and rock-type distributions. Whereas our original realization seems to correspond to a depositional environment with a flat shoreline, the other realization corresponds to a two-lobed shoreline, giving distinctively different facies belts. The figure also clearly demonstrates how the porosity (which depends on the grain-size distribution and packing) varies with the rock types. This can be confirmed by a quick analysis:

```
for i=1:6,  pavg(i) = mean(rock.poro(SN==i));
            navg(i) = mean(rock.ntg(SN==i)); end
```

```
pavg = 0.0615    0.1883    0.1462    0.1145    0.0237    0.1924
navg = 0.5555    0.8421    0.7554    0.6179    0.3888    0.7793
```

In other words, rock types two and six are good sands with high porosity, three and four have intermediate porosity, whereas one and five correspond to less quality sand with a high clay content and hence low porosity.