This document describes the workflow for generating a prior covariance matrix and prior mean for use in data-space inversion by MARE2DEM, including uncertainty quantification via ‘randomize-then-optimize’ (RTO). This workflow uses three code repositories:

1. modelViewing
2. dataViewing
3. CovarianceConstructor

In part 1, modelViewing, a 2D slice through a 3D subsurface resistivity model is extracted. A portion of this slice is then marked for inversion (the ‘reservoir’) and imported into MARE2DEM format. Next, this model (which includes only the region near the receivers and transmitter) must be extended to the lateral edges of the model using Mamba2D. This part is the only non-automated portion of the workflow. All of the preceding steps occur using Matlab scripts and functions.

In part 2, the 3D data to be inverted are rotated and relocated to the 2D profile. This portion is still under construction but should be done in a few days.

In part 3, a series of scripts and functions written in Julia are run to take the MARE2DEM model produced by part 1 and generate a prior model mean, standard deviation, and correlation length. Together, these comprise the prior. The covariance matrix doesn’t explicitly include the prior mean, since the former is a matrix and the latter is a vector, but both will be used to generate prior models during RTO.

Part 1—modelViewing

1. Obtain the modelViewing code repository from <https://github.com/danblatter/modelViewing>.
2. Call extract\_KI\_slice.m
   1. You can also call extract\_emgeo\_slice.m. Each function has an example at the top showing how to use it
   2. This script produces an output file called fout\_##.dat, where fout is the file path and name you provide, while ## is the rotation angle in degrees that you specify
3. Run getKItopSealBottomReservoir2Dsurfaces.jl.
   1. Make sure you read in the file you produced in step 2 (change the file path in the first “readdlm” command of this script). Change the output filenames as you see fit.
   2. The output of this code is a series of (x,z) nodes saved to text files defining the top and bottom of the invertible region of the model. These are obtained by interpolating from 2D surfaces defined over the 3D model to the desired profile.
   3. The easiest way to run this script is install Julia, VScode, and the Julia plugins for VScode. Then you simply open the modelViewing window in VScode, view getKItopSealBottomReservoir2Dsurfaces.jl in the central pane, and press the “run” button (shaped like a triangle) in the upper right
4. Run makeLBLgrid.m.
   1. This requires the Matlab portion of MARE2DEM. I have plans to turn all three code repositories plus the Matlab portion of MARE2DEM into one large “CovarianceConstructor” repository. At the moment, however, you need these Matlab MARE2DEM codes (MARE2DEM/mare2dem\_matlab/) somewhere on your computer and added to your Matlab path.
   2. In addition, you must copy the files contained in the folder “MARE2DEM\_Matlab\_filesIhadToModify” in this repository and paste them in their respective places within MARE2DEM/mare2dem\_matlab (overwriting the originals).
      1. m2d\_gridToM2D.m resides in: MARE2DEM/mare2dem\_matlab/a\_util/conversion/
      2. m2d\_writeResistivity.m resides in: MARE2DEM/mare2dem\_matlab/a\_util/io
      3. Mamba2D.m resides in: MARE2DEM/mare2dem\_matlab/a\_util/mamba
   3. Make sure and change the filenames that occupy to the upper three blocks of code in makeLBLgrid.m to match the desired input and output files. This includes the file output in #2 above as well as the surfaces output in #3 above.
   4. This step should plot the 2D slice you extracted in #2, surrounded by a half-space. It should also output two files: a filename.0.resistivity file and a filename.poly file, where filename is the filename you provided to makeLBLgrid.m
5. Use Mamba2D to extend the model described in the .0.resistivity and .poly files to the lateral edges of the model.
   1. Only the region between the surfaces produced in #3 above should be invertible
   2. Exactly how detailed the 1D model extensions should be is subjective, but should be consistent across all 2D profiles/angles and model years. I think a best practice should be to first identify several key geologic units within the non-inverted/constant portion of the model; then define the depths that bound these units and record these depth values in a spreadsheet along with the average resistivity of the geologic unit above them. Use Mamba’s “add horizontal segments” feature to add these layers in and define the resistivities of the units.
   3. Don’t forget to mark the horizontal extension of the invertible section of the model as invertible (“free parameter”) since we won’t know what resistivity the inversion will settle on at the edges of the region of interest, so we’ll have to invert for the entire horizontal extension. Mesh this region as well.
   4. Once finished, use Mamba to write the fully meshed inversion model, which will produce filename.0.resistivity and filename.poly files; don’t forget to change ‘filename’ or you will overwrite the output from #4 above

Part 2—dataViewing

This part of the code is under construction but should be finished soon.

Part 3—CovarianceConstructor

1. Obtain the CovarianceConstructor repository from <https://github.com/danblatter/CovarianceConstructor>
2. Run getModelGridCentroids.jl
   1. This calls MARE2DEM and therefore must be run from a terminal. To run this you must have MARE2DEM/mare2dem\_source downloaded, compiled, and working. This is not an easy thing to accomplish! I am, however, assuming you have already done so.
   2. To run getModelGridCentroids.jl, open a terminal window and cd to your CovarianceConstructor folder. Then run mpirun -n 5 oversubscribe Julia getModelGridCentroids.jl
   3. Make sure before running the above command you have changed getModelGridCentroids.jl to reflect the desired input and output filenames described in the first few lines of code
3. Run CovarianceConstructor\_KI.jl
   1. There are a lot of options to select before running this script, but you should by now have all the input files you need.
   2. The centroid locations and their resistivity values are contained in the files you produced in #2 above
   3. Set kernel equal to “GaspariCohn.” Other kernels are available, but a non-stationary kernel is isotropic and requires knowing the correlation length everywhere. Other stationary kernels aren’t compact and result in non-sparse covariance matrices
      1. Set the anisotropic correlation lengths [x; z] as you see fit
   4. wellX should be the point along the transformed axis/2D profile where the well you’re using to generate a mean and standard deviation model is located. This should, ideally, be the injection well. It’s location along the profile is determined in Part 2 (still being implemented)
   5. Determine how many geologic surfaces to use in defining the prior. Use only surfaces that extend throughout the region of interest. In effect, don’t use surfaces that terminate within the invertible portion of the model. The code assumes that each surface is defined everywhere within the invertible portion of the model and is always deeper than the surface above it and always shallower than the surface below it. The code always uses the surfaces produced in #3 of Part 1 above, since these define the invertible zone. You can add your own surfaces between these two, however. I refer to them henceforth as “interior surfaces”.
      1. I produced interior surfaces by using the true resistivity model in Mamba. Meaning, I selected points by hand that seemed to delineate important geological units/transitions. Of course, in practice one would use seismic data to produce such surfaces
      2. More interior surfaces means more/stronger prior information will be supplied to the inversion.
      3. If you add interior surfaces, follow the format of the commented code and insert them from shallowest to deepest between s\_top and s\_bot in the array called ‘H’.
   6. The output of running this script will be (a) a bunch of figures visually representing the prior; and (b) a sparse covariance matrix B that can be passed to MARE2DEM as part of an RTO inversion