# Tests

The folder from which these tests are run should be a subfolder of that in which the driver executables exist. Type the command:

runtests.bat

Then see file *diff\_record.dat*.

An exceedingly brief description of the tests is in the table below. In all cases the “standard” output file is the name of the output file with an extra extension of “.std”.

# Model Post-Processing

|  |  |  |
| --- | --- | --- |
| **command** | **Output files** | **Notes** |
| driver1 < driver1a.in | driver1a.out | MF6 DIS heads file |
| driver1 < driver1b.in | driver1b.out | MFUSG unstructured heads file |
| driver1 < driver1c.in | driver1c.out | MFUSG unstructured flow; noncompact storage |
| driver1 < driver1d.in | driver1d.out | MF6 DIS structured flow file |
| driver1 < driver1e.in | driver1e.out | MF6 DISV heads file |
| driver1 < driver1f.in | driver1f.out | MF6 DISV flow file |
| driver1 < driver1g.in | driver1g.out | complex MF6 DIS flow (incl MVR) |
| driver1 < driver1h,in | umodel\_usg\_wel.contents | MFUSG unstructured |
| driver2 < driver2a.in | heads\_interp1.dat  heads\_interp1\_sgl.dat | MF6 steady state single layer;  Single precision counterpart to heads file. |
| driver2 < driver2b.in | coast\_heads\_wells.dat | modflow 15 layers; 36 times |
| driver2 < driver2c.in | lock\_heads\_wells.dat | modflow 1 layer; 114 times |
| driver3 < driver3a.in | coast\_heads\_wells\_time\_interp.dat | modflow 15 layers; 36 times |
| driver3 < driver3b.in | lock\_heads\_wells\_time\_interp.dat | modflow 1 layer; 114 times |
| driver4 < driver4a.in | hd1h.fac  hd1h.bln  hd1h\_wells\_sim.dat | MF6; DIS grid |
| driver4 < driver4b.in | vdl.fac  vdl\_interp.bln  vdl\_well\_heads.dat | MF6; DISV grid |
| driver5 < driver5a.in | sop\_flow\_contents.dat  sop\_chd\_flows.dat | MF6 DIS grid; flows to/from CHD |
| driver5 < driver5b.in | inflows.dat | MF6 DIS; complex flow type |
| driver5 < driver5c.in | rchflow.dat | MF6 DISV; recharge |
| driver5 < driver5d.in | umodel\_wellflow.dat  umodel\_wellflow\_interp.dat | MFUSG CLN budget |
| driver5 < driver5e.in | nfseg.cbb.contents  nfseg\_rech.dat | MFNWT |

# Model Pre-Processing

|  |  |  |
| --- | --- | --- |
| driver6 < driver6a.in | factors.dat  interpolated.ref | 2D kriging with spatially-varying variogram |
| driver7 < driver7a.in | factors1.dat  interpolated1.ref | 2D kriging with automatic adaption to local pilot point spatial density |
| driver8 < driver8a.in | factors3da.dat  factors3da.dat | 3D simple kriging in two zones with exponential variogram and log interpolation |
| driver8 < driver8b.in | factors3db.dat  interptabb.dat | 3D ordinary kriging with Gaussian variogram; high rake and non-log interpolation |
| driver9 < driver9a.in | covid2d.mat | 2D covariance matrix formation with spatially varying variograms |
| driver9 < driver9b.in | covid2d\_unif.mat | 2D covariance matrix formation with zonally-uniform variograms; no need to undertake SVD |
| driver10 < driver10a.mat | cov3d.mat | 3D covariance matrix formation with spatially varying variograms |
| driver10 < driver10b.mat | cov3de\_unif.mat | 3D covariance matrix formation with zonally-uniform variograms; no need to undertake SVD |
| driver11 < driver11a.in | fac\_linear.dat  targ1.dat | Overlaying a polylinear structural feature on a model grid; interpolation using natural numbers. |
| driver11 < driver11b.in | fac\_poly.dat  targ2.dat | Overlaying a polygonal structural feature on a model grid; interpolation using logged numbers. |
| driver12 < driver12a.in | interp\_ipd.ref | 2D inverse power of distance interpolation with spatially-varying interpolation parameters. |
| driver13 < driver13a.in | interptaba\_ipd.dat | 3D inverse power of distance, anisotropic interpolation with spatially varying interpolation parameters. |
| driver13 < driver13b.in | interptabb\_ipd.dat | 3D inverse power of distance, anisotropic interpolation with uniform interpolation parameters. |
| driver14 < driver14a.in | random1\_2d.csv | 2D stochastic field generation using spatial convolution. |
| driver15 < driver15a.in | random1\_3d.csv | 3D stochastic field generation using spatial convolution. |