**Initialization and updating of objects for Forward Solvers**

**ModelOperator**: depends on **grid**, **metricElements**, and indirectly on **ModelParameter**

***creat****e*: sets **grid**, **metricElements**, allocates arrays; note that in ModelOperator\_MF, metric elements is created – and the creator for this object automatically sets the metricELements arrays.

***SetEquation****s* : sets curl-curl operator coefficients

***SetCond*** : sets SigmaE (edge conductivity) using **ModelParameter,** which is not itself stored in the modelOperator object; every time modelParameter changes, SetCond must be called (possibly through calls to setCond in higher-level objects).

Note that **MetricElements** MUST be set ModelParameter before *SetCond* can be called.

Once set routines Amult (curl-curl operator, scaled by Vedge + diagonal dependent on conductivty), divC, grad, div, multAib, multCurlT can be used (NOT divCgrad, which is used onlh in divergence correction, at least in ModelOperator\_MF).

**ModelParameter**: depends on **grid,** and has a pointer to **metricElements**

metric elements must be created first—through creation of ModelOperator

THUS at present: 1) read in model file, creating grid and modelParameter; (2) instantiate ModelOperator, which creates MetricElements, and a pointer to this object. (3) set MetricELements (call model\_parameter%setMetric( model\_operator%metric )).

Perhaps this sequence could be changed, but idea is to have a single MetricElements object, which both ModelOperator and ModelParameter both point to.

**Preconditioner:** depends on **ModelOperator**, on *Period* and indirectly on **ModelParameter**

***create:*** initializes using ModelOperator – still needs to be set

***setPreConditoner:*** completes setup of preconditoner – ModelParameter must be set (and ModelOperator%SetCond called), and Period mut be provided to complete this step (in general—for some cases like Preconditoner\_None, this setup may not be required).

**DiverenceCorrection:** depends indirectly on **ModelOperator** and **ModelParameter**.

***create***: sets a pointer to **Solver** (PCG), which provides access to ModelOperator, including the edge conductivities defined by ModelParameter.

***setCond***: this calls divCorSetup (procedure in ModelOperator) and setPreconditoner (for PCG preconditioner pointed to by Solver) to complete setup of the divergence correction. This must be called after conductivity is updated in ModelOperator, and certainly before first use of DivergenceCorrection (create is NOT enough!)

**ForwardSolver:** depends on **ModelOperator**, on *Period* and indirectly on **ModelParameter**

***create:*** sets a pointer to **Solver**, which provides access to ModelOperator, including the edge conductivities defined by ModelParameter. Also initializes solver control parameters and diagnostics.

***setCond:*** call to update conductivity—required before first use. This will call ModOp%setCond, PreCond%setCond, and if appropriate DivCorr%setCond, updating all required objects for new model parameter. Of course the individual setCond routines can be called separately, but this allows control through interaction only with the ForwardSolver.

***setPeriod:*** call this to change/set period – this sets period/omega in Solver object also. omega is a parameter is Solver base class, so can always set Solver%omega = omega by hand, but there is presently no explicit method for setting period/omega in Solver – perhaps need to fix this?