# Parflow & PFWRF Software Tasks

This is a set of working notes for SGS as he was getting far too many little sheets of paper floating around with PF & PFWRF ideas and tasks. This includes short term and very long term bugs/feature enhancements that have been discussed. Obviously needs a bit of work to prioritize the tasks.

1. Setup PF.WRF test problem.
2. Use WRF namelist to pass PF control values.  
     
   Mostly completed, need to add a couple of values (DZ).  
     
   Completed.
3. Active/Inactive cell efficiency issues.  
     
   Completed. All the easy conversions have been done. Some “loop over everything” still exist but would require most extensive code changes to fix.
4. Link to new *hypre.*Completed. Using latest hypre library.
5. Parallel Guassian permeability setup is broken.  
     
   Completed. Memory overwrite issue.
6. Input/problem setup.  
     
   Currently setup requires creating WRF and matching PF input decks. Would be better to have more unified/automated tools to do this. Modeled after the PF ARPS setup.
7. Memory usage issues (PF is piggy).  
     
   Currently temporary space is being allocated at setup for each module causing temporary space to be the sum of required temporary space. Need to make temporary space so that PF only allocates what is needed at any given point in the simulation. E.G. PGS is allocating a temporary vector but it is only needed for setup; the temporary space does not need to be kept around the entire execution time.  
     
   Solution is to allocate and free temporary vectors on each module invocation rather than in init and destruction phases.  
     
   Completed.
8. Need to move repository to CSM.
9. Output Silo files.  
   Completed.
10. Insure++ tests.  
      
    Run PF through insure++.  
      
    Currently doing this.
11. Nightly regression tests.  
      
    Need to setup an automated regression test suite. Need to have CSM repo working first. Run on tux cluster and ???.
12. Insure++ regression tests.  
      
    Make insure++ part of the regression tests to catch programming errors and memory leaks/overwrites.
13. Setup bug/feature request tracking system.  
      
    Would be nice to replace this file with a tracking system.  
      
    Very low priority.
14. Current tests don’t work.  
      
    The current test setup does not work correctly; several of the tests can pass when they should not.  
      
    Solution is to make sure output is cleaned and is unique for each test.
15. Temporal subcycling in PF.WRF  
      
    PF is being called with timesteps that are smaller than needed.
16. Water balance?  
      
    There was some discussion on 1/5/09 about adding some routines to check water balance. SGS is not sure exactly what this means. Andy seemed to consider this very important.
17. PFWRF on Ra.  
      
    Reed has PF working. WRF in progress.  
      
    High Priority.
18. Replace TCL interface.  
      
    TCL is not exactly the “hot thing” anymore. Consider replacing TCL with Python/Ruby or perhaps even R could be used? What would potential users be most familiar with and what would be the most useful. Could even think about using SciLab.
19. PF compilation flags.  
      
    Currently using generic optimizations, what about SSE and other flags?
20. PF on GPU?

Would be fun to put PF on a GPU, especially if we could get by with single precision, which is what WRF seems to be using.

1. PFAMR.  
     
   Just an easy thing like port PF to SAMRAI ☺
2. Port new PF to MS Windows.
3. Particle code integration with PF?  
     
   At a meeting Reed talked about how he was running PF and then running his particle code; could this be integrated? In the “olden” times we discussed adding various particle/particle like schemes to PF but never did it. Andy had some code he was using for doing some of the PF heterogeneity studies, is that the same code?
4. Document Octree structure  
     
   Capture knowledge of octree flags into the header files.
5. Add octree/indicator field tests.  
     
   Add tests for indicator field octree generation for different processor topologies. One where indicator field overlaps processor boundaries etc.
6. Add inactive cells to silo output format if possible.  
     
   Does Silo allow “inactive” cells so the inactive regions are not displayed in Visit?
7. Octree efficiency  
     
   On first inspection it was not clear if the octree’s being generated had good use of full/empty nodes; looked like it was generating trees that were too full. Perhaps a post processing phase to cleanup?
8. Looping to some level in the octree.  
     
   The grgeom loops will go down to lowest level in octree. It may make sense to stop at some block size and loop over active/inactive cells vs going down to 1x1 level in the octree.
9. Make octree non-uniform refinement?  
     
   The octree is perhaps a bit more oriented towards 3D things, not the 2D+ stuff that is currently being done for large problems. There might be some way to tweak things to make it more efficient.
10. Octree based storage.  
      
    Currently the octree is used only for looping, perhaps use it as storage as well. This could eliminate keeping the inactive cells around at all (currently we are just not looping over them).