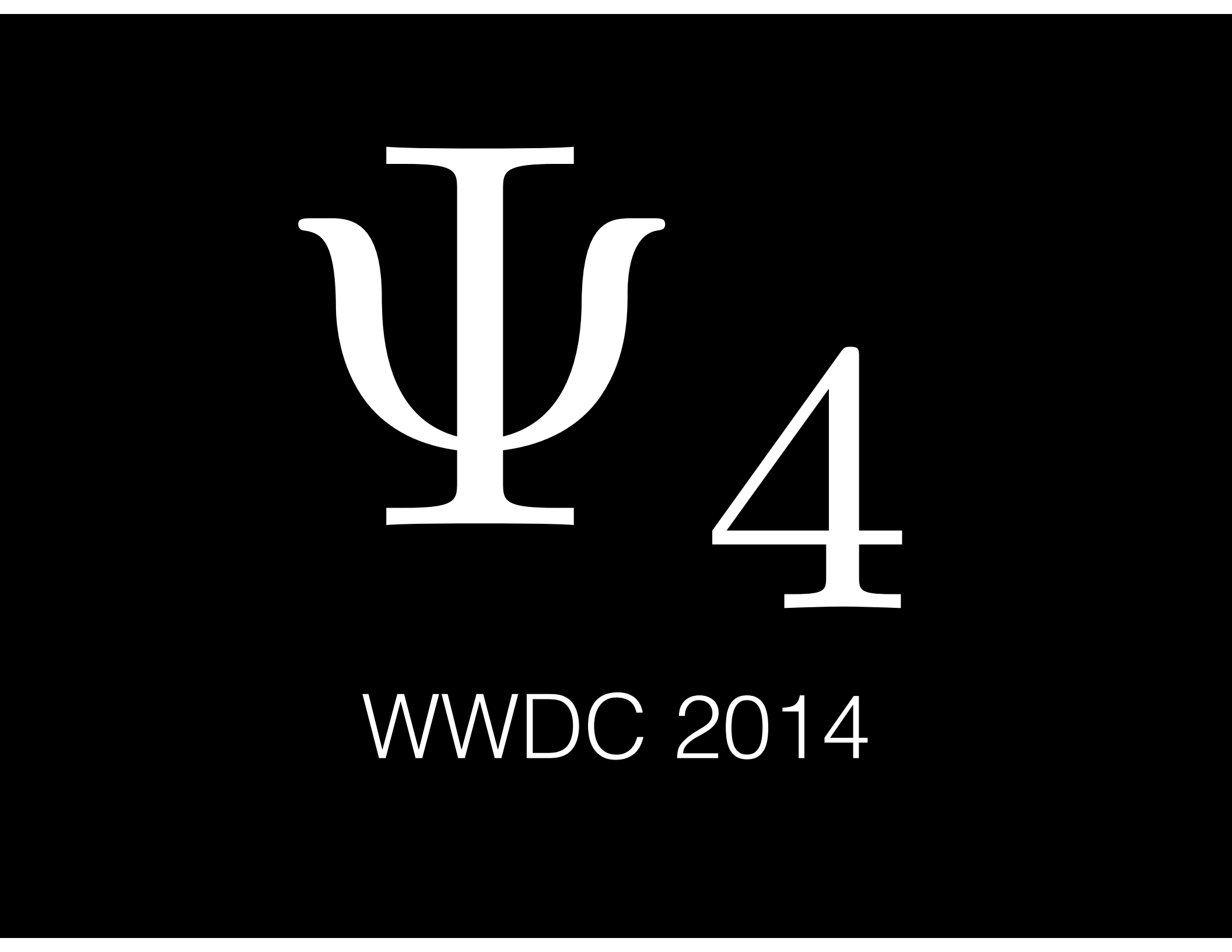
Ψ4
WWDC 2014

Recent VT Developments

- PCM capabilities (Roberto di Remigio and Luca Frediani @ U. Tromsø and Andy Simmonett)
- libPANACHE (Ben Pritchard and Andy Simmonett)
  - Based on lib3index and libmints
  - DF, CD with interfaces to PSI4 and GAMESS
- Local-CC with PAOs, PNOs, and OSVs (Harley McAlexander)
- Perturbed frozen virtual natural orbitals (Ashutosh Kumar)
- CMake Updates (Roberto di Remigio)
- EFP Capabilities (Andy Simmonett and Ben Pritchard)
- Incremental Scheme (Joachim Friedrich @ U. Chemnitz)

PCM in PSI4

- Library libpcm defines PCM class:
  - Constructor initializes PCMSolver library, which prepares cavity and extracts input from PSI4.
  - PSI4 computes electrostatic integrals at each cavity tessera and contracts with current density using in-lined functor.
  - PCMSolver takes electrostatic potentials and returns apparent surface charges to PSI4.
  - PSI4 contracts surface charges against electrostatic integrals to obtain PCM potential (another functor) which is added to the Fock matrix.

PCM in PSI4

- Implementation in libscfsolver/hf.cc yielded immediate functionality in all Hartree-Fock and DFT codes
- Additional effort required for CC codes, but energies now available at PTE, PTES, and PTED levels.
- New CMake infrastructure inserts PCMSolver as an external module automagically.

EFP *via* PSI4/GAMESS

- Simple data-driven interface:
  - GAMESS produces the potential for a given structure
  - PSI4 extracts the potential and MOs from the GAMESS output (yes, really) and inserts them into the PSI4 Hamiltonian and wave function object, respectively.
  - Subsequent “indirect” coupled-cluster linear response computation use to obtain dynamic optical properties in the presence of the solvent.

CMake Infrastructure

- Infrastructure by Andy and Ryan has opened many doors for making PSI4 easier to build, maintain, and extend.
- In order to overcome some difficulties with the PCMsolver integration and to take advantage of newer CMake capabilities, Roberto di Remigio has revised the CMake system on a fork.

CMake Infrastructure

- Improved Fortran/C++ name mangling
- Regression testing via CTest:
  - Execute/exclude tests by name or label/group
  - Additional testing capabilities possible: unit testing, valgrind testing, more.
- CDash interface provides integrated information on builds on various servers.