The CASTEP project: reflections on the first ten years

Keith Refson
STFC Rutherford Appleton Laboratory
Chilton, Didcot, OXON OX11 0QX

30 Sep 2010

Synopsis

- Materials Modelling by Quantum Mechanics
- The CASTEP project
- Programming and Physics
- A critical retrospective
- Summary

Materials Modelling by Quantum Mechanics

Motivation

> *The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known, and the difficulty is only that the application of these laws leads to equations much too complicated to be soluble.*

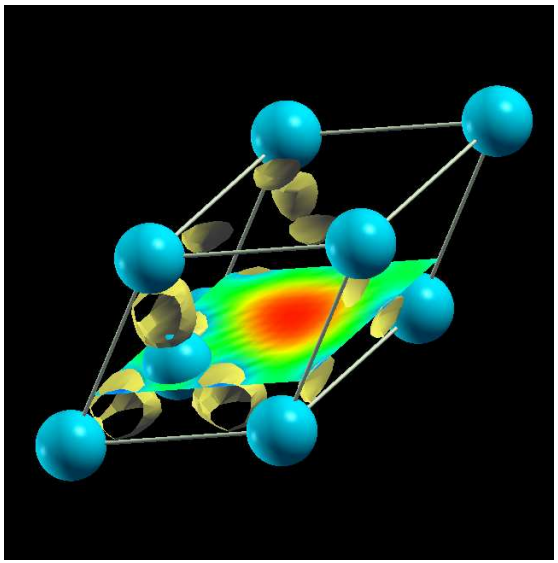
P.A.M. Dirac, Proceedings of the Royal Society **A123**, 714 (1929)

> *Nobody understands quantum mechanics.*

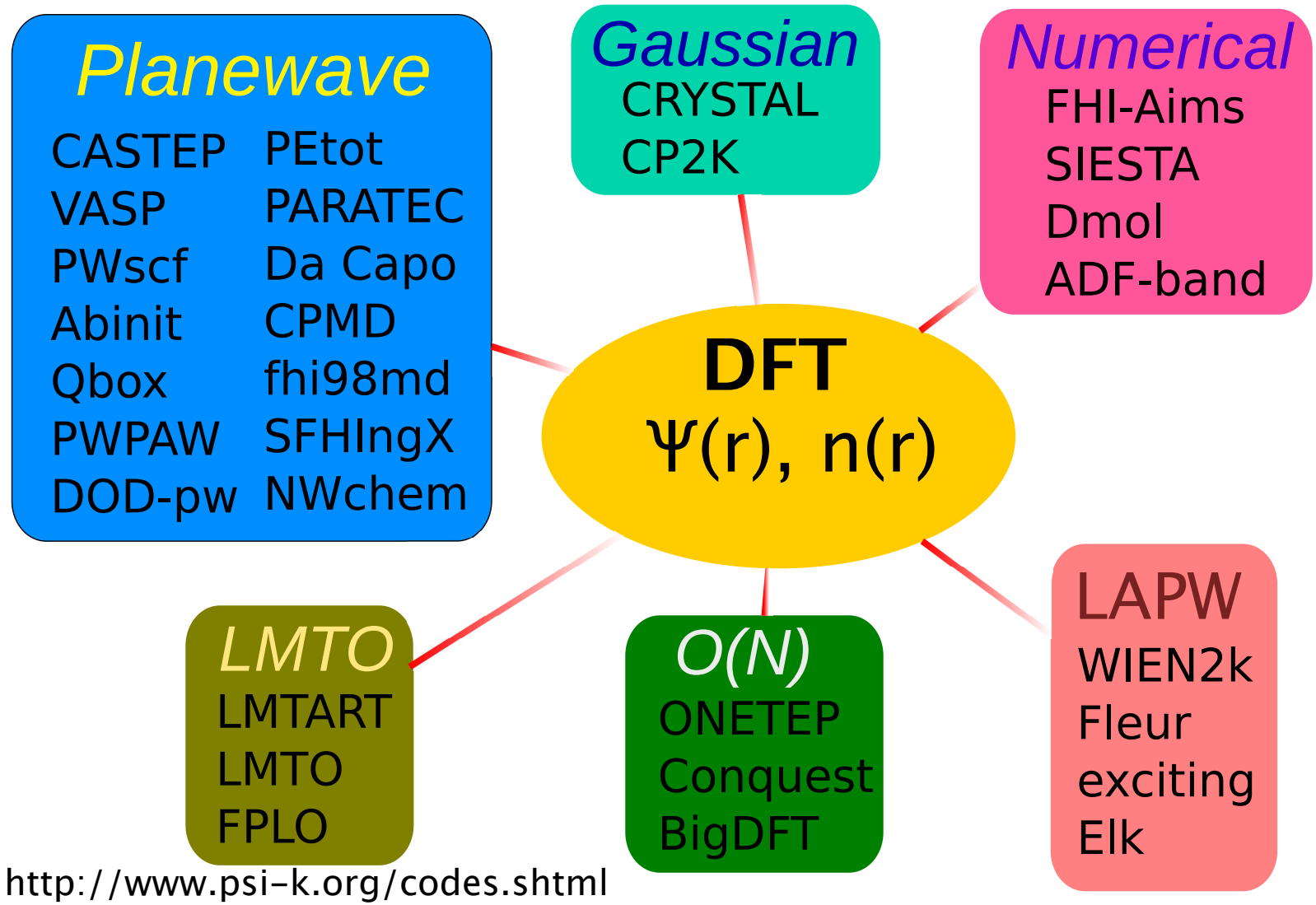
R. P. Feynman

DFT modelling in a plane-wave basis

- Simulation cell contains ions and electrons
- Periodic boundary conditions applied
- Electron density represented on grid.
- Basis coefficients of orbitals stored on recip-space grid.
- Use FFTs to evaluate terms.
- Kohn-Sham equations solved for ground state
- PW basis set is efficient, accurate and robust.
- PW basis requires pseudopotentials.
- Mature technology good algorithms.
- Robust in use.



Some ab initio codes



http://www.psi-k.org/codes.shtml

The CASTEP project

History of CASTEP

- Original code named CASTEP developed by Mike Payne in 1990s
- Commercialised by Molecular Simulations late 1990s
- By late 1990s unstructured developments and merges left codebase in unmodifiable state.
- 1999: New CASTEP project founded to write replacement
- 2000: Castep Developer Group formed (Matt Probert, Chris Pickard Stewart Clark, Matt Segall, Phil Hasnip, Phil Lindan, Mike Payne)
- 2000: Original design specification written; implementation begins
- Dec 2001: First release (2.0) of new code at Durham workshop
- 2002: Keith Refson joins CDG
- 2009 Jonathan Yates joins CDG
- 2002-2010 New functionality added every year
- 2010 CASTEP 5.5 released
- Free academic license for UK researchers/Commercial distribution by Accelrys

Goals of the CASTEP project

- Full-featured materials modelling code based on plane-wave basis and pseudopotentials
- A comprehensive set of "properties" calculations such as spectroscopy
- Engineered for extensibility as development platform
- Robust in use; good I/O interface and error reporting
- Portable to wide range of platforms and compilers
- Parallel HPC code targetted at cluster and supercomputer services

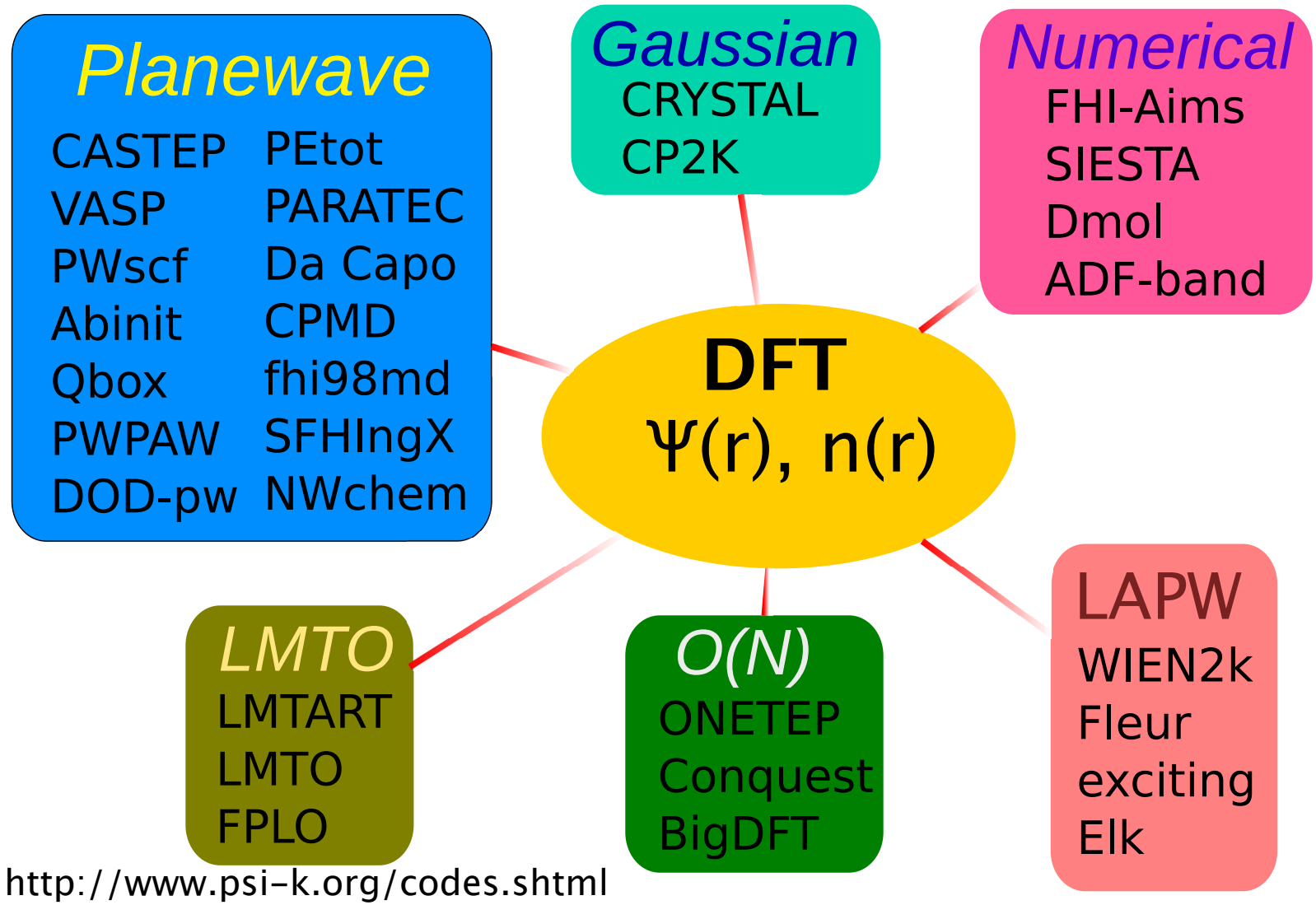
The choice of Fortran

Several criteria influence language choice

- Must support high-level programming of physics
- Expertise of developer community
- Portability
- Compilers must be widely available
- HPC criterion restricts language choice to C, C++, Fortran must interface to MPI
- Execution speed/efficiency is vital
- Maintainability

The CDG decided Fortran 90 was best fit for new CASTEP project.

Fortran in Materials Modelling



http://www.psi-k.org/codes.shtml

The Kohn-Sham equations

Density Functional Theory – Hohenberg Kohn and Sham

$$\left[\frac{-\hbar^2}{2m}\nabla^2 + V_{\text{I-E}}(r) + V_{\text{H}}(r) + V_{\text{XC}}(r)\right]\psi_i(r) = \epsilon_i\psi_i(r)$$

- $\frac{-\hbar^2}{2m}\nabla^2\psi_i(r)$ is kinetic energy of electrons.
- $V_{\text{H}}(r)$ is mean ("Hartree") potential of other electrons.
- $\hat{V}_{\text{I-E}}(r) = \frac{Z_I e}{|\mathbf{r}-\mathbf{R_I}|}$ is Coulomb interaction of electrons with nucleus
- $\hat{V}_{\text{XC}}(r)$ is *exchange-correlation* potential due to Pauli exclusion principle
- $\psi_i(\mathbf{r_i})$ is wavefunction of single electron in effective potential of other electrons and nuclei. Represented in plane-wave basis - many coefficients
- Solve for partial eigenvalue spectrum using iterative method, eg **block Davidson**. Need to evaluate *gradient*

$$\left[\frac{-\hbar^2}{2m}\nabla^2 + \hat{V}_{\text{I-E}}(r) + \hat{V}_{\text{H}}(r) + \hat{V}_{\text{XC}}(r)\right]\psi_i(r)$$

Programming and Physics

$$\left[\frac{-\hbar^2}{2m}\nabla^2 + \hat{V}_{\text{I-E}}(r) + \hat{V}_{\text{H}}(r) + \hat{V}_{\text{XC}}(r)\right]\psi_i(r)$$

```
subroutine hamiltonian_apply_slice(slice,local_pot,nl_d,H_slice,eigenvalues,ek)
  use constants
  use wave
  use pot
  use nlpot

  type(wavefunction_slice),                    intent(inout) :: slice
  type(potential),                             intent(inout) :: local_pot
  real(kind=dp), dimension(:,:,:,:,:), intent(in)            :: nl_d
  type(wavefunction_slice),                    intent(inout) :: H_slice
  real(kind=dp), dimension(:),optional,intent(out)           :: eigenvalues
  real(kind=dp), dimension(:),optional,intent(out)           :: ek

  call wave_initialise(H_slice,slice%kpoint,slice%spin,'Z')

  call pot_apply(local_pot,slice,H_slice)

  call nlpot_apply_and_add(slice,nl_d,H_slice)

  call wave_add_kinetic_energy(slice,ek,H_slice)

end subroutine hamiltonian_apply_slice
```

The wavefunction_slice type


```
type, public :: wavefunction_slice
  ! A portion of a wavefunction, i.e. a group of bands at a single kpt and spin, plus meta-dat

  character(len=32)                                    :: name
  complex(kind=dp), dimension(:,:), allocatable :: coeffs
  complex(kind=dp), dimension(:,:), allocatable :: realspace_coeffs
  real(kind=dp),    dimension(:,:), allocatable :: realspace_coeffs_at_gamma
  logical,          dimension(:),     allocatable :: have_realspace
  integer                                              :: realspace_band_start
  integer                                              :: nbands_max
  integer                                              :: nbands
  integer                                              :: realspace_nbands_max
  integer                                              :: realspace_nbands
  integer                                              :: kpoint          ! kpoint index within parent
  integer                                              :: spin            ! spin index within parent wa
  complex(kind=dp), dimension(:,:), allocatable :: beta_phi
  real(kind=dp),    dimension(:,:), allocatable :: beta_phi_at_gamma
  logical                                              :: have_beta_phi
  logical                                              :: have_gamma
  integer                                              :: waves_at_kp

  integer                                              :: total_nbands        ! Total no. bands
  logical                                              :: bands_distributed
  integer,          dimension(:,:), allocatable    :: node_band_index ! mapping for all nodes in
end type wavefunction_slice
```


endslide

Software Engineering Aspects

- Architecture and modular structure designed first, then ...
- "Spec" - API specification written from design
- F90 modules and derived types used for data encapsulation
- System-dependent and COMMS/MPI calls hidden in wrapper modules.
- Overloaded functions used for code clarity with related types
- Code must adhere to "Spec" or spec must be revised
- Coding guidelines mandate or advise on use of various F90 features, such as array passing conventions
- Language standard was initially F90, now F95+TR15581
- Extensive use of LAPACK and optimized BLAS for performance
- MPI for parallel communications

Language features

CASTEP makes extensive use of F95 language features to support design principles

- Architectural modularity maps to F90 modules
- Data hiding using F90 modules
- parameterised types (selected_real_kind)
- data abstraction using derived types
- Interfaces for overloading same operation on related types
- F90 memory management using ALLOCATABLE arrays
- Originally used POINTERs in derived types, now ALLOCATABLEs
- Array expressions, assignment and intrinsics
- implicit none
- Deprecated features including COMMON, EQUIVALENCE, specifics, goto forbidden by coding guidelines

Critial Assessment

CASTEP Features

- Plane-wave basis set and pseudopotential method.
- Parallelised over FFT grid, k-points and bands using MPI.
- Optimised for massively parallel scaling on HPC resources.

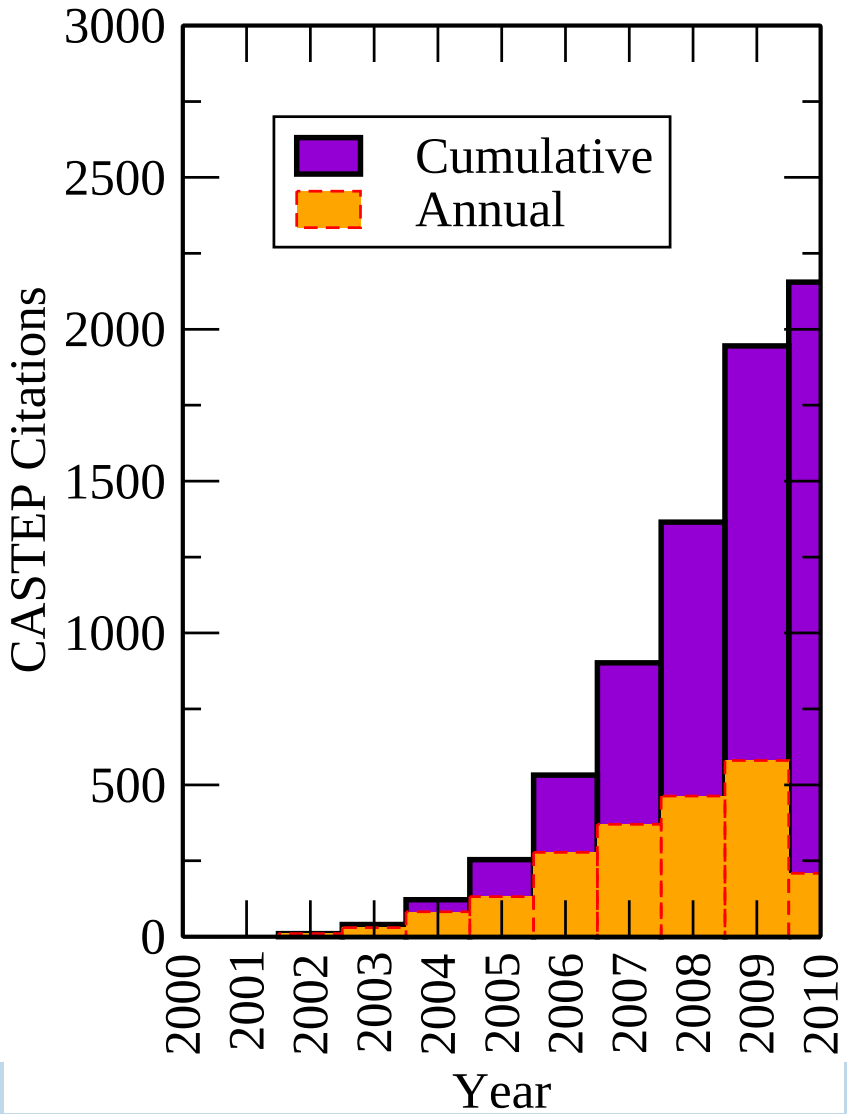
- Ultrasoft and norm-conserving pseudopotentials
- XC-functionals LDA, PBE/RPBE, PW91, WC, PBEsol, HF, SX, PBE0, B3LYP
- Dispersion forces using SEDC (Grimme scheme)
- Electronic band structure
- Built-in pseudopotential generator
- Molecular dynamics in NVE, NVT and NPH ensembles
- BFGS optimisation of ions and cell
- Internal co-ordinates optimisation
- Damped MD.
- Quantum path integral MD

- Transition-State Searching
- DFPT and FD lattice dynamics
- Quasi-harmonic free energy
- IR and Raman intensities
- Dielectric Permittivity
- Born Effective Charges
- PAW for core-sensitive properties
- NMR Chemical Shifts
- EPR Hyperfine Interaction Tensors
- Electric Field Gradient Tensor
- Mulliken population analysis
- Hirshfeld population analysis
- Wannier functions
- Berry phase polarization

Citation report

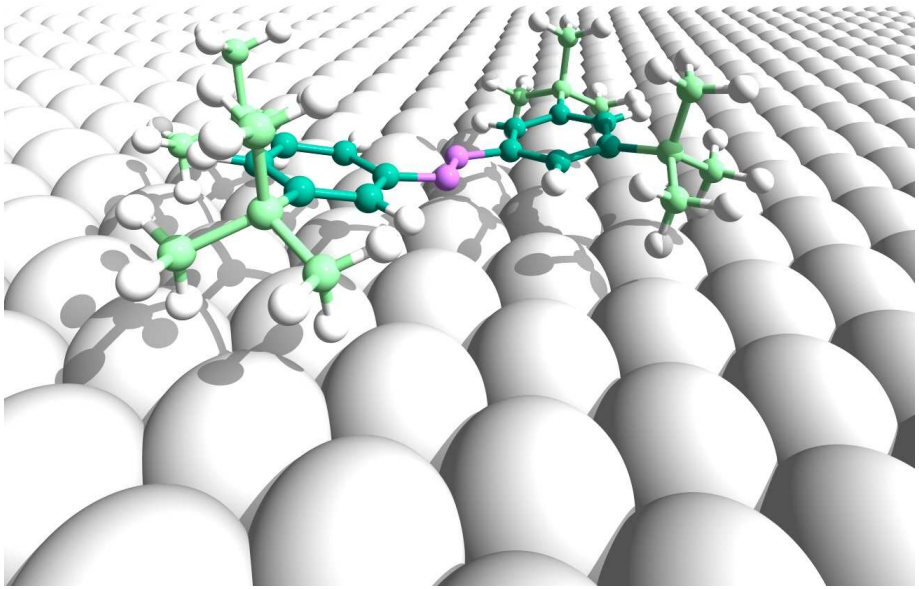
2155 literature citations by May 2010 91 UK citations in 2009

- 2 JACS
- 3 Phys Chem Chem Phys
- 2 Chem Phys Phys Chem
- 4 J. Chem. Phys.
- 11 J Phys. Chem.
- 2 Geochim. Cosmochim. Acta
- 5 Physical Review Letters
- 1 Geophysical Research Letters
- 6 Applied Physics Letters
- 12 Physical Review B
- 5 J. Appled Phys
- 7 J. Phys. Condens. Matter

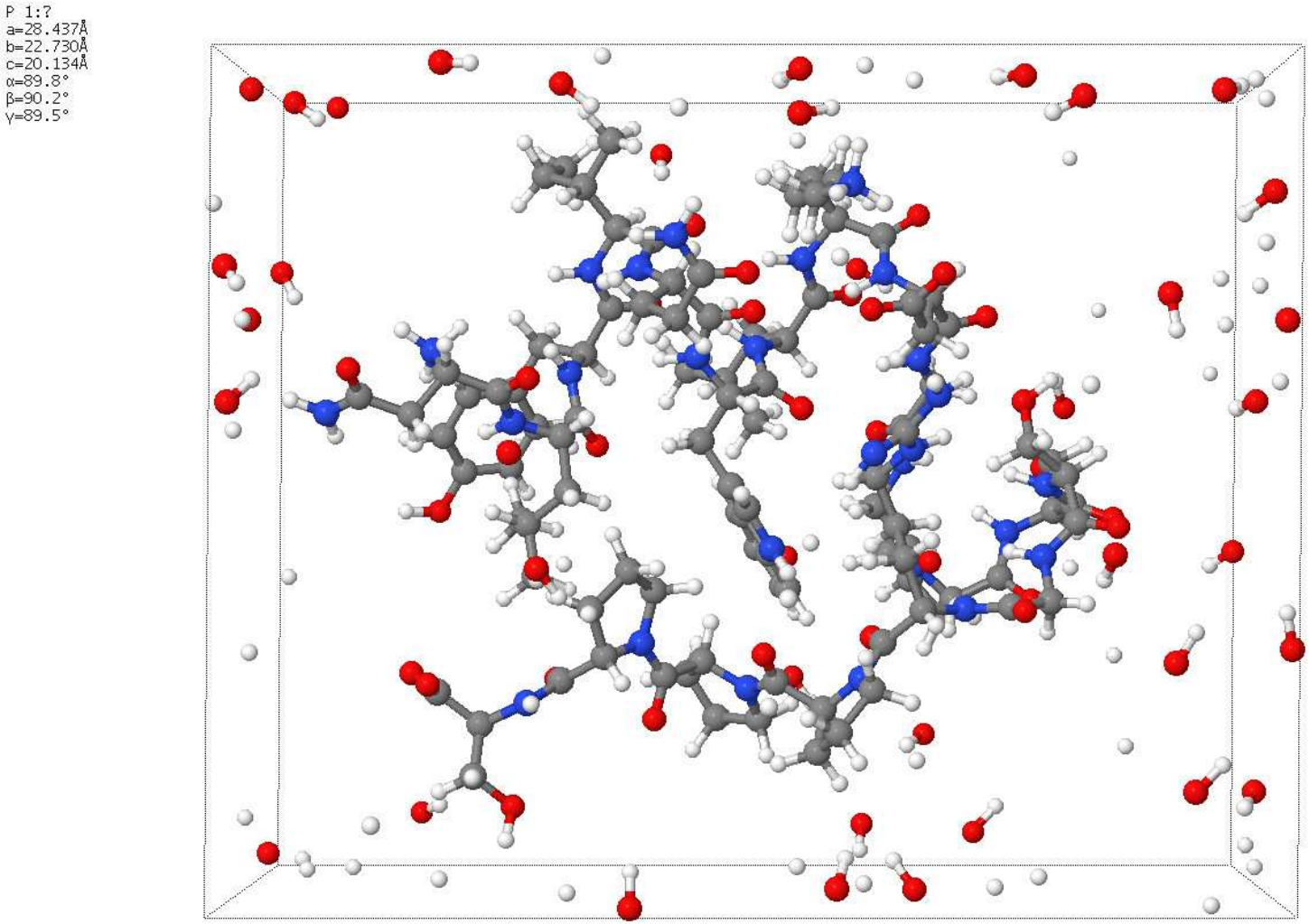


Azobenzene as a molecular switch

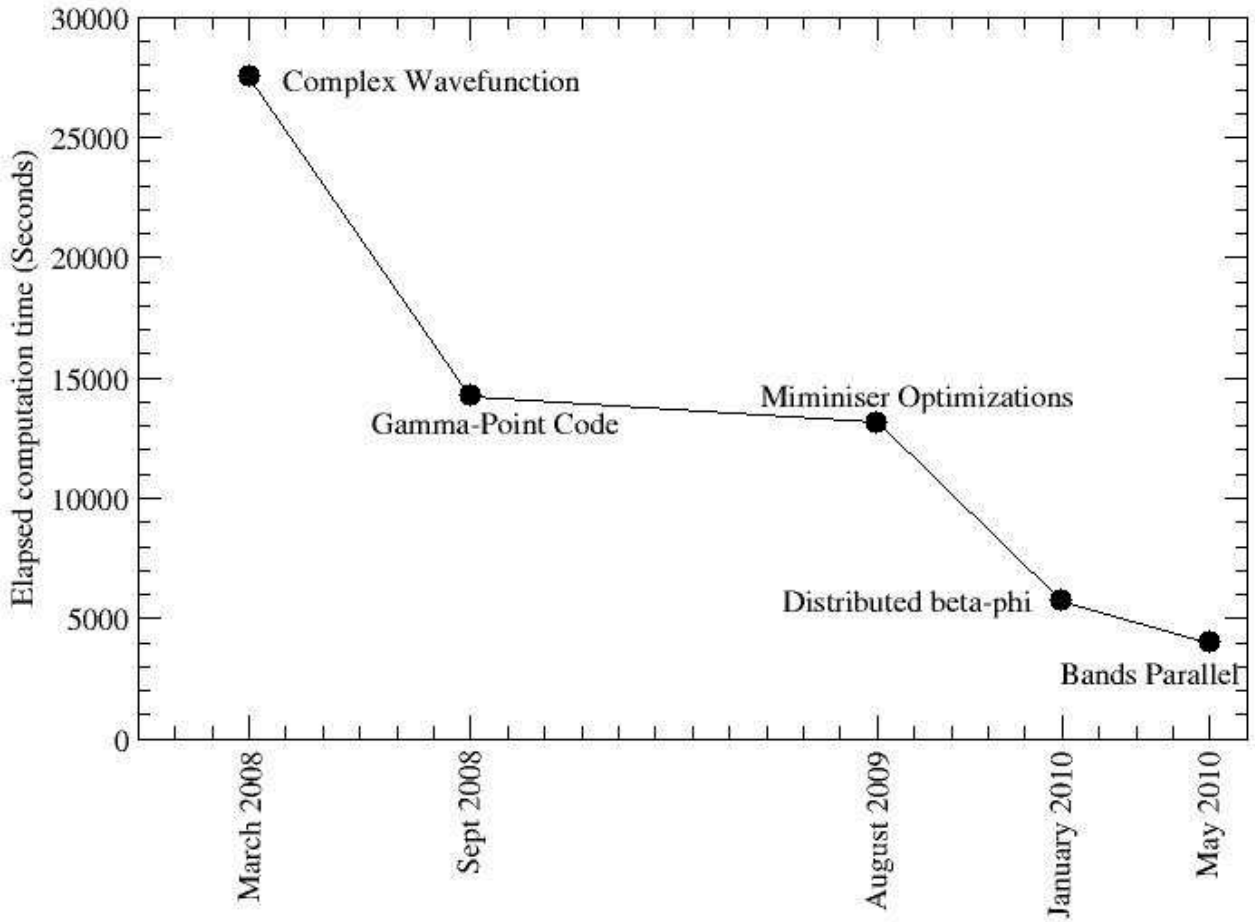
- Fritz Haber Institut (Berlin) Group [PCCP 111, 1-9 (2010)]
- TBA (tetrabutyl azobenzene) on Au(111) surface
- Used DFT-D in CASTEP
- Structural, energetic and vibrational calculations to characterize role of



HPC Capability: Peptide in water 1280 atoms



HECToR Performance



Retrospective assessment of design architecture

Most design decisions paid off

- Much new functionality implemented at high level. DFPT for vibrational properties, IR and raman spectroscopy, TD-DFT, NMR spectroscopy.
- Profound intermediate-level changes, such as “gamma point” optimizations were made without disrupting high-level code.

but some mistakes

- Could have made data encapsulation more complete - derived types do not carry enough metadata for complete flexibility
- Architecture has evolved but would benefit from enhanced type structure
- Excessive use of module variable data

Co-evolution of Fortran and CASTEP

- Original language was F90.
- Updates to F95 - very minor changes
- TR 15581: Updated all types to use allocatables rather than pointers
- Waiting impatiently for F2003 I/O and ALLOCATE system error message handling
- Waiting impatiently for F2003 argument and environment routines

Unsatisfactory aspects

- Still using F77-style interfaces to system libraries LAPACK, BLAS, MPI as fully portable, widely-available F90 versions not available.
- Automatic arrays must be avoided as unusable due to unmanageable stack overflow problems
- Memory management using Allocatables makes memory profiling hard.
- No way of gracefully handling memory exceeded errors on any common O/S. (OOM Killer)
- Compilation cascade and long compile times frustrate development cycle.
- The "same-compiler" requirement for library compatibility - maintenence burden
- Argument association rules requiring rank matching give rise to unnecessary overload duplication.
- In case of MPI this is a catastrophe. We must still use "pass 1st element" method for layered comms calls.
- Compilers are all buggy. CASTEP has found bugs in Intel, Compaq, Alpha, SGI, Cray, PGI, Pathscale, Sun, g95, gfortran, IBM, NAG, Lahey, Salford.

Non-standard intrinsic modules for non-standard system interface.

Summary

The CASTEP project has been a great success
Synergy of Fortran and Physics can be very productive
Fortran is the dominant language in a very high-impact field of science - computational materials modelling

Acknowledgements EPSRC for funding
HECToR for computer time and support
STFC E-Science (SCARF) for computer resources
The CDG (Stewart Clark, Matt Probert, Chris Pickard, Jonathan Yates, Phil Hasnip, Matt Segall and Mike Payne)