Multiscale QM/MM modelling of catalytic systems with ChemShell†

You Lu, a Kakali Sen, b Chin Yong, a David S. D. Gunn, a John A. Purton, a Jingcheng Guan, b Alec Desmoutier, b Jamal Abdul Nasir, b Xingfan Zhang, b Lei Zhu, b Jing Hou, b Joe Jackson-Masters, c Sam Watts, c Rowan Hanson, c Harry N. Thomas, c Omal Jayawardena, † Andrew J. Logsdail, b Scott M. Woodley, b Hans M. Senn, d Paul Sherwood, e C. Richard A. Catlow, d b Alexey A. Sokol d b and Thomas W. Keal d * a

Hybrid quantum mechanical/molecular mechanical (QM/MM) methods are a powerful computational tool for the investigation of all forms of catalysis, as they allow for an accurate description of reactions occurring at catalytic sites in the context of a complicated electrostatic environment. The scriptable computational chemistry environment ChemShell is a leading software package for QM/MM calculations, providing a flexible, high performance framework for modelling both biomolecular and materials catalysis. We present an overview of recent applications of ChemShell to problems in catalysis and review new functionality introduced into the redeveloped Python-based version of ChemShell to support catalytic modelling. These include a fully guided workflow for biomolecular QM/MM modelling, starting from an experimental structure, a periodic QM/MM embedding scheme to support modelling of metallic materials, and a comprehensive set of tutorials for biomolecular and materials modelling.

Introduction

Modelling catalytic processes is a challenge for computational chemistry, as it requires an accurate characterisation of reactivity influenced by a usually complex chemical environment. This type of problem is, however, well suited to a hybrid multiscale approach, where the reaction site is treated using an accurate quantum mechanical (QM) description of the electronic structure, while the environment is accounted for through a more approximate, but less computationally expensive method. A major advantage of the hybrid approach is the ability to use a high level of theory in the central QM region, as illustrated by the work of Piccini et al., 1 who achieved chemical accuracy in the calculation of rate constants for catalytic reactions in zeolites. The environment can be described either using a relatively inexpensive level of QM theory, or with a classical molecular mechanics (MM) forcefield.

Hybrid QM/MM methods offer a good balance of computational efficiency and modelling accuracy and have become established as a powerful tool for catalytic science. QM/MM methods are available in a range of computational chemistry software packages, for which the implementations broadly fall into two categories. In the first category are extensions to existing software packages, for which the implementations broadly fall into two categories. In the first category are extensions to existing electronic structure or classical molecular dynamics (MD) packages, with examples including AM1,2 AMBER,3,4 CHARMM,5 CP2K,6 deMon2k,7 Gaussian,8,9 GROMACS,10 NAMD,11 and ORCA,12 among others. QM/MM methods are implemented in these packages either through incorporating the required additional features directly into the software, or through interfacing with other software that can perform the additional calculations. Advantages of this category include the potential for efficient integration with a specific codebase, and a familiar starting point for existing users of the software, but at the cost of being unable to use features from alternative codes which might be relevant to a given scientific problem.

The second category of QM/MM implementations is exemplified by the ChemShell computational chemistry environment,13-15 which is well established as a flexible scripting platform for chemical calculations with a focus on hybrid multiscale methods. Recognising that most computational chemistry software...
is specialised in either QM or MM calculations, ChemShell implements a code-coupling framework that acts as a wrapper around external QM and MM packages and combines their outputs into a full QM/MM simulation. The user is therefore able to mix and match their preferred external codes according to their practical experience and specific features they wish to use. This attractive feature has made the second category an increasingly popular option for QM/MM methods development. Other computational chemistry programs taking a similar approach include ASE, 16 Cuby, 17 Fromage, 18 Janus, 19 LICHEM, 20 MiMiC, 21 PUPIL, 22 QMCube, 23 QMMM, 24 and Spicy. 25

ChemShell has a modular design to ensure that it is robust when working with a potentially large number of different combinations of external packages. The modular framework in the latest release of the code is illustrated in Fig. 1. At the lowest level in the framework are objects defined in ChemShell for storing chemical data and methods for manipulating the data. Molecular and materials systems are defined using Fragment objects, which contain details of the system’s geometry, periodicity, charges and other atomic attributes. This information can either be specified directly via Python commands or imported from common structural data formats such as XYZ or PDB. The output of chemical calculations, such as energies and gradients of a system, are stored in Result objects.

At the next level, calculations are defined using Theory objects. The nature of the calculation is defined by specifying either an external QM or MM package, with relevant additional options to describe the calculation parameters (for example, the functional and basis set for a density functional theory (DFT) calculation using a QM package, or the forcefield definition for an MM package). Hybrid methods can also be specified using ChemShell’s QM/MM driver, which combines the user’s choice of QM and MM theories. In the hybrid case the QM/MM driver will then also handle the boundary between the QM and MM regions of the system, based on a user-provided list of atoms that define the QM region. Full details of the standard QM/MM schemes implemented in ChemShell can be found in previous articles. 13-15

Once a Theory has been defined, it can then be used for one or more high-level chemical tasks, which are also defined as Python objects. The simplest Task is a single-point calculation, where the energy, and optionally also the gradient, will be returned. More complicated Tasks include geometry optimisation (using the DL-FIND geometry optimisation library26 distributed with ChemShell) and molecular dynamics. These Tasks invoke the chosen Theory whenever an energy or gradient is required, and so are completely flexible as to how the underlying Theory has been defined.

Finally, at the top level, specialised guided Workflows have been implemented to provide functionality that combines multiple Tasks and other features in the code to achieve a specific goal. Examples of ChemShell Workflows include the cluster cutting process to set up solid-state embedding models, and the new biomolecular solvation workflow, which we describe later in this article.

All ChemShell calculations are controlled via an input script. These scripts were originally written in the Tcl programming language, but in recent years a major redevelopment of ChemShell was undertaken, 15 and the new software is scriptable using the Python programming language, which has become established as the most widely used scripting language for scientific applications. The open source, Python-based version of ChemShell (“Py-ChemShell”) is intended to succeed the original Tcl-based package and is the only version now under active development, although Tcl-ChemShell continues to be maintained as a legacy software resource.

A strong focus of the ChemShell project is on scalability to high performance computing (HPC) systems, including through task-farming parallelisation. 27 An advanced task-farming framework is provided, which allows many chemical tasks to be performed in parallel, in addition to making use of parallelism in the external codes it calls, resulting in highly scalable QM/MM calculations. 15 The Py-ChemShell build system is also designed with HPC-specific builds in mind, with automated build configurations provided for a range of HPC platforms including ARCHER2, the UK’s national supercomputing service. These configuration scripts can be easily adapted to support other HPC systems.

In this review, we present a survey of recent ChemShell applications and describe new software developments in
Py-ChemShell up to and including version 21.0, released in December 2021. Over this period there have been many underpinning improvements to the code, as well as new interfaces to the quantum-mechanical packages ORCA,28 FHI-aims,29 Molpro,30 Gaussian,31 and CP2K,32 the semi-empirical QM codes MNDO,33 and DFTB+,34 and the classical molecular dynamics package NAMD.35 There have also been significant new high-level functionalities, which we describe in the next sections, including a comprehensive guided workflow for biomolecular QM/MM calculations, further refinements to the covalent and ionic solid-state embedding models, and a new periodic QM/MM embedding scheme targeting metallic systems, each accompanied by an illustrative example system.

QM/MM simulations of biomolecules

ChemShell is widely used for modelling complex biomolecular systems, especially in the investigation of enzyme-catalysed reaction mechanisms, where a quantum-mechanical description of the active site is necessary to characterise the catalytic processes, while a hybrid approach is needed to efficiently calculate the full protein system. Metalloenzymes are particularly well-suited to QM/MM calculations, as they usually require at least a DFT level of theory to adequately describe their reactivity at the metal site. The versatile cytochrome P450 superfamily of haem enzymes (Fig. 2) are among the most studied systems using QM/MM simulations with ChemShell, and continue to be a very active topic of research.36–51 In recent work Kalita et al. used a combination of molecular dynamics simulations and QM/MM to explore oxygen binding and the protonation steps leading to formation of compound I in two archetypal P450 systems.52 The team used these techniques also to understand the behaviour of a bioengineered CYP450PME variant that catalyses a reaction (C–H amination) that natural P450s do not.53 Lu et al. used ChemShell to illuminate the key role of a neighbouring residue in the oxidation of ethanol by CYP2E1, which results in a preferred mechanism involving an unusual initial hydrogen abstraction from the ethanol O–H bond.54

ChemShell has been used to investigate many other haem-containing proteins, including recent studies by Liu and co-workers on the latex clearing protein LcpK30, a b-type cytochrome,55 the inactivation mechanism of neuronal nitric oxide synthase,56 and a proton-coupled electron transfer process in coprohaem decarboxylase, which is involved in haem biosynthesis.57 Other recent works have looked at catalase-peroxidases (KatGs),58 indoleamine-2,3-dioxygenase,59 and YfeX, a promising candidate for carbine transfer biocatalysis.60

The wider field of iron-containing enzymes also continues to be a fruitful target for QM/MM studies.61–73 In recent work Tian et al. have studied the unique non-haem iron enzyme ergothioneine synthase, using QM/MM to identify oxygen binding modes, determine the preferred spin state for reactivity, and clarify the sequence of sulfoxidation and C–S bond formation in its catalytic cycle.74 The 2-oxoglutarate (2OG) dependent non-haem iron oxygenases catalyse a wide variety of oxidative processes and have been the target of multiple recent studies.75–86 Chaturvedi et al. used ChemShell QM/MM calculations to investigate the reaction mechanism of PHF8, which catalyses demethylation of methylated lysine in histones, and is a target for cancer treatments.87 The simulations revealed that rearrangement of 2OG at the iron centre is an important part of the mechanism, which may help inform the design of inhibitors.

More complex active sites containing multiple metal centres are also amenable to QM/MM simulations. Several recent studies have investigated reactivity of non-haem diiron enzymes.88–91 Song et al. used ChemShell to model PhnZ, a mixed-valence diiron oxygenase, which catalyses the conversion of organophosphonic acids to inorganic phosphate via a mechanism involving the substrate bonding to one Fe(II) site and oxygen to the other.92 [NiFe]-hydrogenases have also been explored93,94 which are of great interest in the development of the hydrogen economy due to their ability to catalyse H2 oxidation at ambient conditions. The radical S-adenosylmethionine (SAM) enzyme family, which are dependent on a [4Fe–4S] cofactor, also continue to attract attention for their ability to catalyse a wide range of redox reactions.95 Feng et al. used QM/MM calculations to provide insight into how the reactivity of the SAM enzyme Dph2 is affected by the electronic state of the [4Fe–4S] cluster.96 More complicated still are the double-cubane cluster proteins,97 featuring an [Fe8S9] complex, and the iron-molybdenum cofactor

![Fig. 2](Fig_2.png)
Copper enzymes catalyse a wide variety of reactions, often utilising multinuclear copper centres. Wu et al. used QM/MM calculations in ChemShell to investigate the form of the active species in two binuclear copper monooxgenases, which catalyse stereospecific C–H hydroxylation of precursors to hormones and neurotransmitters, resulting in a full description of the catalytic cycle. Sen et al. have investigated copper-containing nitrite reductases, which reduce nitrite to nitric acid as part of the global nitrogen cycle. These enzymes contain two copper centres, one of which catalyses the reduction reaction while the other is involved in electron transfer. A combination of ChemShell QM/MM simulations and serial crystallography, which provides a “movie” of the reduction process in a protein crystal, yield complementary information on the system, resulting in a full description of the reaction mechanism.

The flexibility of QM/MM approaches to describe a wide range of metal cofactors is demonstrated by recent ChemShell applications on magnesium-dependent tyrosine synthase and ald-alu-epimerase, xylotise α-1,3-xylotyltransferase (where magnesium was modelled in place of manganese), nickel-containing quercetinase, tungsten-dependent benzoyl-coenzyme A reductase and zinc-dependent leukotriene A4 hydrolase, matrix metalloproteinase-11 and matrix metalloproteinase-9. Zhao et al. used ChemShell to find the most favourable pathway for catalysis of N–N bond formation by a family of zinc-binding cupin proteins.

While most recent ChemShell protein studies have targeted metalloenzymes, QM/MM simulations are equally applicable to other types of enzymes. Cantù Reinhard et al. investigated the case of nogalamycin monoxygenase (NMO), which, unlike most other monoxygenases, is able to catalyse the oxidation of a substrate without either a metal or organic cofactor. QM/MM calculations revealed paths for activation of the substrate by triplet O2, giving insight into the NMO reaction mechanism. Enzymes can also contain a mixture of metal and non-metal active sites, as in the case of cyclooxygenase-2, which has a haem-containing peroxidase active site as well as a cyclooxygenase active site. The peroxidase active site is responsible for generating a tyrosyl radical, which initiates the reaction at the cyclooxygenase active site. The subsequent deoxygenation of the substrate arachidonic acid at the cyclooxygenase active site can be efficiently characterised using QM/MM methods as the peroxidase site does not need to be included in the QM region at this stage.

Environmental chemistry is an increasingly prominent topic for QM/MM simulations, which have been used to investigate biodegradation of pesticides and plastics and other hazardous pollutants. Feng et al. used ChemShell to study the degradation of polyethylene terephthalate (PET), comparing the common features of reactions catalysed by the enzymes IsPETase and IsMHETase and analysing the structural characteristics that affect catalytic activity, pointing the way to the design of more efficient enzymes for plastic recycling.

It is well-established that intrinsic and external electric fields can significantly affect catalytic processes, and these are the focus of several recent studies using ChemShell QM/MM calculations in combination with electric fields modelled using the TITAN code. Yadav et al. used these methods to show that a single-site difference between two cytochrome P450 peroxigenases creates a local electric field that changes the stability of intermediate species in the catalytic cycle, resulting in a different reaction mechanism and products. Ramanan et al. looked at the effect of applying external electric fields on arginine demethylation catalysed by a 2OG-oxygenase, revealing that they promote C–H activation and inhibit N–H activation pathways.

QM/MM simulations of biomolecules are not limited to proteins, and several recent ChemShell studies have focussed on DNA, RNA, and their components. Biocatalysis applications involving DNA repair enzymes are an active topic in this area. Naydenova et al. used ChemShell to investigate the catalytic mechanism of uracil DNA glycosylase, which initiates excision of uracil DNA lesions. QM/MM simulations revealed the key role of an active-site histidine residue and alternative reaction pathways that explain why mutation reduces, but does not completely suppress, enzymatic activity.

Other complex biomolecular targets in recent work include membrane enzymes, protein–protein interactions, and the bacterial ribosome.

QM/MM methods are now widely applied to the study of excited electronic states, in tandem with electronic structure software packages capable of time-dependent density functional theory or wavefunction-based excited state calculations. Light-sensitive proteins that have been simulated with ChemShell in recent studies include a wide range of rhodopsins, photosystem II, calcium-regulated photoproteins, the fluorescent protein, the phototoxic protein KillerRed, bacterial phytocromes, and cyanobacteriochromes. Excited state calculations are typically used to simulate experimental UV-vis spectra, and so are useful for characterising any molecular system containing a chromophore, including catalytic intermediates.

The vast majority of biomolecular applications use a form of DFT for the QM region, but QM/MM approaches can also accommodate higher-level methods where necessary. Nutho et al. used ChemShell QM/MM calculations to generate potential energy profiles up to the LCCSD(T)/MM level to help establish the reaction mechanism for inhibition of zika virus protease by peptidyl aldehydes. Douglas-Gallardo et al. used ChemShell QM/MM calculations and high-level projector-embedding calculations with Molpro to provide corrections to free energy barriers for carbon dioxide formation by the RuBisCO enzyme, calculated using semi-empirical QM/MM molecular dynamics, to increase the accuracy of the reaction
profiles obtained for different protonation states of key amino acid residues. Free energies may also be estimated at the DFT/MM level using QM/MM free energy perturbation, as in the study of Zheng et al. on HCV protease.

Biomolecular workflows in Py-ChemShell

The 21.0 release of Py-ChemShell introduces QM/MM functionality for biomolecular modelling into the redeveloped version of ChemShell for the first time. In the original Tcl-based version of ChemShell, it was expected that the user would set up and equilibrate the biomolecular system of interest using a classical MD package before importing it into ChemShell to perform QM/MM calculations. It is still possible to follow this workflow in Py-ChemShell, but this requires a high level of familiarity with modelling best practices. We have therefore implemented a more comprehensive guided workflow within ChemShell, starting from the beginning with an experimental crystal structure, and guiding the user with a set of best practices for system setup, solvation and equilibration, resulting in a model that is ready for QM/MM simulations. The aim of this workflow is to reduce the prior knowledge required for new users while also offering sufficient flexibility to experienced users, who may also benefit from the efficiency of a more automated setup process. In the following sections we summarise the steps in the workflow with reference to the example of P450cam, a well-studied cytochrome P450 enzyme that catalyses the hydroxylation of camphor, as shown in Fig. 2. The workflow we describe is illustrated by the flowchart in Fig. 3.

System preparation

The preparation of a protein for QM/MM calculations begins with a structure obtained using X-ray crystallography or other experimental techniques. The structural data may be read from a PDB file on disk, or ChemShell can download a structure directly from the Protein Data Bank if the user supplies a PDB ID. For P450cam the initial structure (PDB reference 1DZ8) is shown in Fig. 2A.

When ChemShell imports a raw PDB file, it performs some initial processing, such as choosing a single atomic position if alternatives are given (based on a consideration of mean occupancy and, if that is not conclusive, B-factors, where a lower B-factor indicates reduced thermal motion and so less uncertainty in the specified atomic coordinates). It does not, however, attempt to remediate any missing amino acid residues, and the user should make sure that the protein chain to be modelled is complete before proceeding.

Following the import of the PDB file, the structure usually needs to be manipulated further into a form suitable for simulation. This may include selection of a specific chain to be simulated, removal of unwanted ions or molecules, and addition of relevant ligands if they are not already present in the structure. ChemShell provides a number of powerful atom selection tools to assist with these steps.

Fig. 3 Flowchart of the workflow for biomolecular solvation and QM/MM calculations in Py-ChemShell. The underlying machinery shown in the blue box is automated, but may be controlled by the user via setting of options at each stage.
At this point, the protein model is ready for ChemShell's guided solvation workflow. This routine automates the remaining setup stages, solvation of the system, and equilibration using classical MD through an interface to NAMD. The first task in the workflow is to add hydrogen atoms to the protein model, which are not observed by X-ray crystallography, and so not included in the PDB structure. To address this, ChemShell provides a wrapper to call the PDB2PQR utility program, which automates the addition of hydrogen atoms. PDB2PQR also assigns partial charges to the model using values from the forcefield selected by the user, and calls PROPKA to predict the pKa values of titratable groups in the protein, so that protonation states can be assigned.

Once the missing hydrogen atoms, protonation states, and partial charges have been assigned, the next stage is to solvate the protein model. By default, ChemShell will place the protein model in a pre-equilibrated cubic TIP3P water box, with an appropriate cavity created for the protein by deleting water molecules that occupy the same volume. A cubic box is used in preparation for the periodic MD simulations that will be run to equilibrate the full protein system. The size of the box can be adjusted as required.

In order to run the MD simulations with NAMD, a Protein Structure File (PSF) must be generated. ChemShell provides a wrapper for the PSFGEN utility in NAMD to create the PSF file. To improve the automation of this stage the ChemShell wrapper will automatically apply patches corresponding to common additional species, such as the haem group in P450cam. The user may also supply their own patches for more unusual species, as they would with a standalone PSFGEN run.

Finally, before running the MD, the workflow neutralises the solvated system through addition of counterions to the solvent environment. By default, sodium ions are added to neutralise a negatively charged system, or chloride ions to neutralise a positively charged system. The final periodic solvated P450cam system ready for equilibration is shown in Fig. 2B.

MD equilibration

The MD stage of the workflow automates a three-step equilibration process for the solvated protein system. In the initial implementation of the workflow, NAMD is used as the classical MD driver.

In the first step of the process, the solvent environment and neutralising counterions are allowed to relax around the protein and any ligands or cofactors, which are fixed at their initial positions. The aim of this MD step is to remove any solvent inhomogeneities around the protein created by the initial solvation. The relaxation is accomplished through an initial partial energy minimisation, followed by an MD simulation under NVT conditions at 300 K. Default values for the minimisation and MD timescales are provided based on previous experience (as given in Fig. 3), but can be adjusted by the user as desired.

In the second step, the protein is allowed to relax in the solvent environment, albeit with backbone constraints in place to retain the overall structure at this stage. Again, an initial energy minimisation is performed, to enable the protein to swiftly reconfigure, followed by an MD simulation. In this step, the MD simulation is run using an NPT ensemble, so that the water density can equilibrate to natural conditions.

At this stage the user must make a choice about whether species such as cofactors or ligands are fixed or free to move in the rest of the MD simulations. This may be an enforced decision, most commonly because a full set of forcefield parameters is not readily available for the species at hand, which therefore needs to be kept fixed. It may also be desirable to fix the species in an initial configuration if it is known to be the correct initial configuration for a reaction to take place. In our P450cam example, this is the case for the camphor molecule, whose binding position and orientation should be kept during MD as they are correct for the subsequent H-abstraction reaction. While fixing parts of the system introduces a degree of artificiality into the MD equilibration, it should be noted that if the fixed species are within the QM region in the subsequent QM/MM calculations they will be fully optimised at that stage.

The third step is a final “production” MD simulation, again under NPT conditions but without any constraints on the protein backbone. This simulation provides the trajectory from which snapshots will be taken as a starting point for QM/MM calculations.

Py-ChemShell provides a set of analysis tools for MD trajectories that can be used to check if equilibration has completed, for example by looking at whether the RMSD of the protein backbone has converged. The temperature and energies along the trajectory can also be plotted. Once the MD simulation stage has been successfully completed, a separate ChemShell routine takes snapshots at regular intervals along the trajectory for the following QM/MM stage.

QM/MM calculations

Snapshots from the final MD simulation are the starting point for biomolecular QM/MM calculations in ChemShell. However, once the initial equilibration is complete, it is preferable to avoid the use of periodic boundary conditions as this would restrict the range of QM codes that could be used and is relatively computationally expensive, as well as potentially introducing artefacts due to periodic protein–protein interactions. Instead, ChemShell provides routines to cut the protein and a shell of solvent molecules out of the MD snapshots, resulting in a non-periodic model that is suitable for subsequent QM/MM calculations, as illustrated for the example of P450cam in Fig. 2C. This model will be saved to a PQR format file, along with the assigned partial charges from the initial setup stages.

At this point, the model is ready for QM/MM calculations. The QM region now needs to be defined, and ChemShell’s atom selection tools can again be used to build the QM atom list from the relevant chemical groups. An example minimal QM region for P450cam is shown in Fig. 2D, consisting of the core haem group, cysteinate side chain, oxygen ligand, and camphor molecule. The chosen QM theory should also be defined in the usual way for QM/MM calculations. An integer value must be specified for the charge of the QM region, which is most
commonly obtained by summing the MM partial charges corresponding to the QM region.

For the MM part of a biomolecular QM/MM calculation, DL_POLY\textsuperscript{196} is used to evaluate MM energies and gradients, as it provides a flexible, general purpose forcefield engine for systems set up either using the guided workflow or imported from elsewhere. The Python-based version of ChemShell interfaces to the open-source version 5 release of DL_POLY, which scales well to very large QM/MM systems.\textsuperscript{15} For biomolecular calculations using CHARMM forcefields,\textsuperscript{197} the forcefield parameters are managed by a wrapper to the DL_FIELD program,\textsuperscript{198} which is used to convert CHARMM forcefield data automatically into DL_POLY format. User-defined parameter sets outside the standard CHARMM forcefields can also be provided to DL_FIELD, e.g. for the camphor molecule in the case of P450cam.

Once the biomolecular QM/MM model has been fully specified, the calculations are run as with any other QM/MM system using the modular ChemShell framework. Tasks can now be defined and run, making use of high-level ChemShell features, such as the range of geometry optimisation techniques provided by DL-FIND to characterise reaction mechanisms,\textsuperscript{26} including the nudged elastic band and dimer methods.

In the present release of Py-ChemShell (\textit{v}21.0), the full workflow is designed to be used with CHARMM forcefields. However, the user may not wish to use the complete workflow: the import mechanism via DL_FIELD is separate to the solvation stage, and if the initial CHARMM MD runs have been run elsewhere, these snapshots can be directly imported to QM/MM calculations in ChemShell. An alternative workflow is supported for AMBER forcefields,\textsuperscript{199,200} which can be imported directly into Py-ChemShell for QM/MM calculations. In this case, it is assumed that the system has been set up and equilibrated externally, for example in AMBER, and snapshots taken from there can be imported into ChemShell. Conversion tools are also provided for importing existing QM/MM setups that have been built in Tcl-ChemShell. An online case study tutorial, which goes step-by-step through the ChemShell commands used to set up the P450cam model and run QM/MM calculations on it, is available on the ChemShell website.\textsuperscript{201}

Future releases of Py-ChemShell will provide a full guided workflow for AMBER forcefields, following the same lines as for CHARMM forcefields. The workflow will also be extended to support solvation and equilibration of DNA strands in a similar manner to proteins.

**Solid state embedding**

Although QM/MM methods have been more commonly used in the area of biocatalysis, they are equally applicable to solid state modelling. ChemShell provides both covalent and ionic embedded cluster models for heterogeneous catalysis applications. The covalent embedding model is extensively used for studies of porous catalytic materials, with a number of recent ChemShell investigations targeting microporous zeolites\textsuperscript{202–205} and mesoporous MCM-41.\textsuperscript{206,207} Nastase \textit{et al.} used QM/MM simulations of the zeolites H-ZSM-5 and H-Y to study a crucial step in the conversion of methanol to hydrocarbons where the intermediate dimethyl ether is formed.\textsuperscript{208} The results highlighted the importance of the strength of the Brønsted acidic character at the active site on the stability of dimethyl ether, which may help to explain the superior catalytic performance of H-ZSM-5. Abdul Nasir \textit{et al.} investigated the effect of water and ammonia solvent molecules on the selective catalytic reduction of NO\textsubscript{x} species by the copper-exchanged zeolite Cu-CHA.\textsuperscript{209} The results showed that solvation of the active sites has a strong influence on the energy profile of the catalytic cycle, with differing effects on the oxidative and reductive parts. Liu \textit{et al.} used QM/MM methods to investigate the structure and stability of vanadium-doped active sites in Al-MCM-41, and their potential use as heterogeneous catalysts for the oxidation of furans.\textsuperscript{210} Besides catalysis applications, covalent QM/MM embedding methods in ChemShell have also been recently applied to the excited state properties of organic materials\textsuperscript{211–216} and the adsorption of hydrogen onto amorphous solid water.\textsuperscript{217}

The ionic QM/MM embedded cluster approach can be used to model both bulk defects and surface reactivity. In recent ChemShell studies it has been used to characterise native point defects in Ga\textsubscript{n+1}\textsuperscript{218} to create optimised models of rutile TiO\textsubscript{2} surfaces,\textsuperscript{219} to study oxygen vacancies in TiO\textsubscript{2} using DFT and high-level wavefunction methods,\textsuperscript{220} to improve interatomic potential descriptions of CeO\textsubscript{2},\textsuperscript{221} to investigate bulk and surface vacancies in MnO with potential catalytic applications to the transformation of CO\textsubscript{2},\textsuperscript{222} and to study the defect properties of Cu in ZnO, an important industrial catalyst for the synthesis of methanol.\textsuperscript{223} Sainna \textit{et al.} used ChemShell to study the reaction of glycerol over a catalytic MgO surface to form methanol.\textsuperscript{224} The results indicated that the most favourable route to methanol production involves a disproportionation reaction on the surface of MgO, while water is unlikely to play an important part in the process.

The nature of the QM/MM model employed changes significantly depending on whether a covalent or ionic embedded cluster approach is required, and we describe below separately the capabilities of ChemShell for each approach. Both approaches, however, start from the construction of a finite cluster model cut from a periodic bulk or slab model of the system. The finite cluster QM/MM model has several significant advantages over a periodic representation of the system, including avoidance of unphysical periodic interactions of the local defect with itself, straightforward handling of charged systems, and the ability to use a wide range of non-periodic QM software packages, improving the computational efficiency of the simulations. However, the creation of a cluster model necessarily involves the loss of the periodic electrostatic environment. To compensate for this, a set of point charges is arranged outside the cluster and fitted to reproduce the bulk electrostatic potential and field within the active region of the cluster. ChemShell provides a guided workflow for automated construction of the clusters and point charge fitting, following the procedure detailed in ref. 13.

**Covalent embedded cluster model**

For covalent or semi-covalent systems such as zeolites, which can be described using forcefields with covalent bonding terms,
a similar QM/MM approach can be used as in the modelling of biomolecules.225 Link atoms are added to the QM region where bonds are broken at the boundary with the MM region, and the charge shift scheme is applied to avoid over-polarisation of the QM region. In the case of zeolites, an additional charge adjustment to the MM region is required because the natural choice of the QM region, groups of silicon or aluminium atoms and their connected oxygen atoms, will not have a zero charge.13 This is an automated procedure in all versions of ChemShell for the case of aluminosilicate systems. In the Python-based version of ChemShell, the adjustment routine has been generalised so that other systems can be treated in the same way, with charge modifications calculated using user-defined bond contributions from each pair of atoms at the boundary.

A typical embedded cluster model of a zeolite is shown in Fig. 4. This example, with a cluster radius of 21.2 Å, consists of 2157 atoms. In ChemShell the standard additive electrostatic embedding scheme is generally used for zeolite calculations. As with electrostatic embedding calculations on biomolecules, any QM software package capable of supporting point charges in the QM Hamiltonian may be used. Typically calculations are performed using DFT with a hybrid exchange–correlation functional. In Py-ChemShell, the MM region is usually calculated with GULP,226 using the Hill–Sauer aluminosilicate forcefield.227,228

A zeolite case study tutorial is available on the ChemShell website,229 detailing the setup of siliceous and aluminosilicate embedded cluster models, geometry optimisation of the zeolite system, calculation of ionisation potentials and deprotonation energies, and calculation of vibrational frequencies when methanol is adsorbed in the zeolite framework.

**Ionic embedded cluster model**

The QM/MM embedding model must be modified for ionically bonded systems such as transition metal oxides, where a link atom approach at the boundary is no longer appropriate. In Py-ChemShell, we follow the ionic embedding procedure established in the original Tcl-based version of ChemShell,230 where pseudopotentials are applied to atoms in a boundary region to localise the electron density within the inner QM region.

QM/MM calculations using the ionic embedding model typically use polarisable embedding, where the QM and MM parts of the model are polarised by each other. In ChemShell, the polarisable MM region is represented by shell model potentials, where shell positions are relaxed under the electrostatic influence of the QM region. As with electrostatic embedding, the QM region is in turn polarised by the MM environment, so the calculation must be iterated self-consistently until the polarisation effects have converged.

In v21.0 of Py-ChemShell, the ionic embedding model is supported by NWChem,231 GAMESS-UK232 and FHI-aims, which contain the necessary mechanisms for handling pseudopotentials at the boundary sites. The polarisable MM region is calculated using shell model potentials implemented in GULP.233

A typical example ionic QM/MM embedded cluster model of a CO₂ molecule adsorbed onto MgO is shown in Fig. 5. In order to model the surface, a hemispherical cluster of radius 30 Å has been cut from a periodic slab model of MgO, with external point charges fitted to the electrostatic potential and electric field in a central sampling region of radius 15 Å, which is also the region allowed to relax when the cluster is optimised. Typically, 20–30 atoms around the central adsorption site are included in the QM region. The MM region is usually modelled using DFT with a hybrid exchange–correlation functional. Large-core effective core potentials are placed on the Mg₂⁺ cations in the boundary between the QM and MM regions. The MM region is calculated using a shell model potential parameterised for MgO.233

![Fig. 4 ChemShell embedded cluster model of the zeolite H-ZSM-5.](image1)

Yellow atoms = Si, red = O, purple = Al, white = H, black = external point charges. A typical QM region of 67 atoms is shown with a ball and stick representation.

![Fig. 5 ChemShell embedded cluster model of CO₂ adsorbed onto an MgO surface.](image2)

Red spheres = O, green = Mg, cyan = C, black = external point charges.
Two ionic embedding case study tutorials are available on the ChemShell website. The first takes the user through the process of setting up a cluster representing an MgO surface to study surface reactivity, including addition of an adsorbed CO₂ molecule, and calculating vibrational frequencies and ionisation potentials of the system. The second demonstrates how a bulk MgO cluster can be set up and doped with Li, with the aim of calculating the barrier for electron migration, calculation of vibrational frequencies of the transition state, and calculation of electron paramagnetic resonance properties.

Periodic QM/MM embedding

The non-periodic models described above can be applied to a wide range of systems across biomolecular and materials modelling, but there are some problems for which they are not suitable. Adsorption on metallic surfaces is one example of this: while taking a QM/MM approach would offer the same advantages in principle over pure QM methods as for other surface studies, the delocalised electronic structure of the metal surface is not properly described by a finite QM cluster, and so the standard non-periodic embedding models described above cannot be applied.

To enable systems of this kind to be modelled, we have implemented the “QM/Me” embedding scheme of Meyer and Reuter in Py-ChemShell. The QM/Me method is fully described in the original paper so here we only highlight the differences with the non-periodic embedded cluster model implemented in ChemShell.

Like the previous methods, QM/Me follows a hybrid embedding approach, with the QM region containing part of the metal surface and the adsorbate in order to capture adsorbate–adsorbate and substrate–adsorbate interactions accurately. Importantly, in QM/Me the QM region is calculated with periodic boundary conditions (using a slab model) so that the band structure of the metallic system is correctly described. This region is embedded in a much larger bath-like MM region, which is used to describe longer range elastic substrate–substrate interactions, and is again calculated with periodic boundary conditions. However, the two calculations cannot simply be added together, as the QM calculation would also describe substrate–substrate interactions to some extent and so these will be overcounted. To avoid this problem, the QM region is calculated twice: once including both surface and adsorbate and once for the surface alone. The second QM calculations is then subtracted from the first, removing the surface–surface interactions while retaining the desired description of the adsorption. The system is then fully described as:

\[ E^{QM/Me}(\mathbf{R}) = [E^{QM}(\mathbf{R}_{slab} \cup \mathbf{R}_{ads}) - E^{QM}(\mathbf{R}_{slab})] + E^{Me}(\mathbf{R}_{slab} \cup \mathbf{R}_{env}) \]

where \( \mathbf{R} \) are the coordinates of the full system including the adsorbate species \( \mathbf{R}_{ads} \), the slab model \( \mathbf{R}_{slab} \), and the remainder of the MM environment \( \mathbf{R}_{env} \). \( E^{QM} \) is the overall system energy, \( E^{QM}(\mathbf{R}_{slab}) \) the energy of the two QM calculations with and without the adsorbate, and \( E^{Me} \) is the energy of the classical environment.

The ChemShell implementation of QM/Me follows the original scheme, while also being fully integrated into the modular ChemShell framework so that in principle any interfaced QM and MM packages can be used, providing they are capable of calculations with periodic boundary conditions. Our initial implementation released in Py-ChemShell v21.0 is intended for use with the interface to CP2K for the periodic QM calculations, and GULP for the MM region.

Fig. 6 shows an example of a QM/Me model used to optimise O₂ adsorbed onto a Pd[100] surface. In this case, the MM bath contained 500 Pd atoms and was calculated with the modified embedded atom method with second nearest neighbour (MEAM-2nn). DFT with the PBE exchange–correlation functional has been used for the QM region. The basis set employed was TZV2P-MOLOPT-GTH for oxygen and SZV-MOLOPT-SR-GTH for palladium and the pseudopotential was GTH-PBE. In this demonstrative case the QM region contained a 3-layer slab of 48 Pd atoms for which a 2 x 4 x 1 Monkhorst-Pack mesh was used for K-points. The QM/Me scheme, including definition of the two QM calculations, is fully automated using the “QM/Me” option in Py-ChemShell.

In addition to the QM/Me method, Py-ChemShell v21.0 has new functionality to support periodic calculations in general, in particular with the ability to optimise unit cells with DL-FIND. Future releases of ChemShell will extend the QM/Me method to other QM codes (FHI-aims, CASTEP and CRYSTAL), and implement other periodic QM/MM embedding schemes within the same framework.

Conclusions and outlook

In this review we have surveyed the wide range of biomolecular and materials QM/MM applications in biomolecular and solid state catalysis that have recently been studied using ChemShell and outlined the capabilities of the latest release of the developed Python-based version of the software. The 2021 release
of Py-ChemShell includes the full functionality required for biomolecular QM/MM workflows, guiding users from the initial download of an experimental protein structure, through setup and equilibration with classical molecular dynamics, and then on to full QM/MM calculations using the flexible, modular ChemShell framework. The latest release of Py-ChemShell also implements the periodic QM/Me scheme for surface-adsorbate systems, complementing the established finite cluster embedding models for materials modelling. The addition of QM/Me extends the range of materials that can be studied to include metallic systems, where a delocalised QM description is required.

The Python-based version of ChemShell is under very active development. The next release of the software is planned for mid-2023 and will include a generic molecular dynamics module, enabling MD with any of the QM/MM schemes described in this paper and any combination of QM and MM codes supported by ChemShell. It will also feature routines for harmonic IR and Raman spectroscopic signatures and anharmonic corrections based on the vibrational self-consistent field (VSCF) approach, as well as additional interfaces to external QM packages.

Looking further ahead, several new development projects are underway, targeting different aspects of QM/MM functionality in ChemShell. A major scientific focus of the method developments is the emerging research area of hybrid catalysis, which involves catalysts that combine elements of the usual categories of heterogeneous, homogeneous and biomolecular catalysts. New features in development targeting these systems include advanced multilayer QM/MM models, including high-level wavefunction-in-DFT embedding and solvent boundary potentials, and enhanced sampling methods to fully characterise the behaviour of solvated catalysts. Support for free energy methods in Py-ChemShell will be enabled through a planned interface to the PLUMED library. Free energy methods previously available in Tcl-ChemShell (such as umbrella sampling and QM/MM free energy perturbation) will be supported together with additional methods such as metadynamics, to provide the user with a range of options for obtaining free energies. Beyond this, adaptive QM/MM methods will be introduced to allow solvent molecules to enter or leave the QM region, which will provide a dynamic description of solvent around a catalytic centre.

Another key area for future development will be the integration of machine learning (ML) approaches for interatomic potentials into embedding models for materials modelling. ML methods have also been recently introduced into DL-FIND to accelerate geometry optimisation by Kästner and co-workers. Complementing the scientific method developments, the underlying ChemShell software infrastructure is being enhanced to meet the needs of new computing platforms. As part of the UK’s ExCALIBUR exascale computing initiative, ChemShell is being retooled for the exascale computing era, with a particular focus on complex QM/MM workflows coupling to highly scalable molecular and periodic DFT software to target problems in computational catalysis. ChemShell is also being developed to manage multiscale embedding workflows involving emerging quantum computing technologies in combination with conventional high performance computing.

Together these developments provide a flexible, scalable toolkit for QM/MM simulations that is applicable to all types of catalysis: heterogeneous, homogeneous and bio-catalysis. To download ChemShell and access the online documentation, tutorials and forum, please visit chemshell.org.

Author contributions

The software developments described in this review are the result of several projects. The biomolecular QM/MM features in Py-ChemShell v21.0 were developed and validated by YL, KS, CY, HMS, PS and TWK. Solid state embedding model enhancements and validation in Py-ChemShell v19.0–21.0 were carried out by YL, DSDG, JG, AD, JAN, XZ, LZ, QH, AJL, CRAC, AAS and TWK. The covalent and ionic embedding case study tutorials were created by JJ-M, SW, RH, HNT, OJ, TWK and AJL. Periodic QM/MM embedding and related development work in Py-ChemShell v21.0 was carried out by YL, DSDG, JAP and TWK as part of the SAINT project led by SMW. The manuscript was drafted by TWK, YL and KS, and was reviewed and edited by all the authors.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

TWK, CY, HMS, PS, CRAC and AAS acknowledge funding for the biomolecular and solid state embedding developments described in this review from EPSRC grant EP/R001847/1. SMW, CRAC and TWK acknowledge funding for the periodic QM/MM developments from EPSRC grant EP/P022235/1. The SAINT project team are grateful to Jörg Meyer and Matt Watkins for discussions concerning the QM/Me method and CP2K. Additional support for solid state embedding developments was provided by CoSeC, the Computational Science Centre for Research Communities, in collaboration with the Materials Chemistry Consortium, one of the UK’s High End Computing consortia, and the UK’s Collaborative Computational Project 5 (CCP5), which are funded by EPSRC grants EP/R029431/1 and EP/M022617/1. We acknowledge use of computing resources provided by STFC Scientific Computing Department’s SCARF consortia, and the ARCHER2 UK National Supercomputing Service (https://www.archer2.ac.uk). AJL acknowledges funding by the UKRI Future Leaders Fellowship program (MR/T018372/1). AJL and TWK thank CCP5 for supporting several student summer projects at Cardiff University to create online case study tutorials for Py-ChemShell.

Notes and references

1 G. Piccini, M. Alessio and J. Sauer, Ab Initio Calculation of Rate Constants for Molecule–Surface Reactions with
over aluminium-doped zinc oxide using an efficient mass-
29 V. Blum, R. Gehrke, F. Hanke, P. Havu, V. Havu, X. Ren, K. Reuter and M. Scheffler, Ab initio molecular simulations
39 X. Wang, J. Shi and Y. Liu, Oxidative Rearrangement Mechanism of Pentalenolactone F Catalyzed by Cyto-
41 H. Su, B. Wang and S. Shaik, Quantum-Mechanical/ Molecular-Mechanical Studies of CYP11A1-Catalyzed Bio-
46 J. Wei and Y. Liu, Mechanistic Insights into the P450 TleB-
Catalyzed Unusual Intramolecular C–N Bond Formation


53 S. Kalita, S. Shaik and K. D. Dubey, MD simulations and QM/MM calculations reveal the key mechanistic elements which are responsible for the efficient C–H amination reaction performed by a bioengineered P450 enzyme, *Chem. Sci.*, 2021, 12, 14507–14518.


A. Gheorghiu, P. V. Coveney and A. A. Arabi, The influence of base pair tautomerism on single point mutations in aqueous DNA, Interface Focus, 2020, 10, 20190120.


This article is licensed under a Creative Commons Attribution 3.0 Unported Licence.
Perspective


