Computational Studies on Neighbour Exclusion in a Series of Diacridine and Di-(terpy)Pt(II) Thiolate Complexes:

Combined Molecular Dynamics and Fragment Molecular Orbital Approaches

UNSW
AUSTRALIA

Thesis submitted in partial fulfilment of the requirements for the admission to the degree of Bachelor of Science (Advanced)

KEIRAN ROWELL

Supervisor: Dr. Graham E. Ball

Co-supervisors: A/Prof. Laurence P. G. Wakelin, Dr. Donald S. Thomas

SCHOOL OF CHEMISTRY

31st October 2014

Abstract

The phenomenon of neighbour exclusion (that molecules cannot intercalate into adjacent base-pair steps in DNA) has been recognised since the 70s, however the cause(s) of neighbour exclusion have not been definitively settled. Furthermore, macroscopic measurements on the complexes formed by DNA and linked dimeric intercalators have been previously interpreted as being able to violate neighbour exclusion. Thus molecular dynamics (MD) and fragment molecular orbital (FMO) methods are used in this study to provide firmer theoretical grounding to the interpretation of neighbour exclusion and the possibility of violating it.

MD simulations were performed *via* AMBER's ff12SB force-field on a series of linked dimeric diacridine and di-(terpy)Pt(II) thiolate intercalators inserted into a range of pyridine-purine base-pair steps with the linker either spanning one or two base-pairs, yielding 151 separate 10 ns trajectories. The (terpy)Pt(II) thiolate moiety was parameterised for MD *via* DFT scans and MDC-q partial charge calculations. Post-processing on the MD trajectories was performed *via* the MMPBSA module, however these values were found to be of limited accuracy in representing the entropic and solvation terms in the system. RI-SCS-MP2/6-31G* FMO calculations were performed on structures representative of each trajectory's dominant conformation, which were generated *via* a clustering analysis and minimisation protocol.

Structures with the linker spanning two base-pairs were found to be viable according to MD and were in better agreement with experimental results, contradicting previous interpretations based upon static space-filling models. Proposals of steric or helix unwinding causes of neighbour exclusion are not supported by this work. Additionally, the mean electrostatic repulsion between adjacent charged chromophores (*circa* 4 kcal/mol) was found to be a significant contributor to, but not the sole cause of, neighbour exclusion.

Acknowledgements

Firstly I'd like to acknowledge the privilege of having not one but three terrific supervisors and their invaluable guidance in helping grasp all the different facets of this work: Graham for the *ab intio* methods, Larry for the molecular biology, and Don for the molecular dynamics. Their unending supply of good-humour, enthusiasm and sage advice made this thesis possible.

To Chris my fellow colleague and modeller, with whom I've slogged through the minefields of incomprehensible variables, unintelligible manuals, and incompatible nomenclature. Whose presence, whether physical or virtual, turned the disheartening moments into hilarity and the routine ones into a joy.

To Peter the vinyl pusher for his unrelenting kindness, the cherished coffee breaks, impeccable music suggestions, and just generally indulging me in the fantasy that one can be a geeky theorist while still maintaining affectations of being a with it hepcat (Big Science indeed!).

To my various friends in and out of uni who are too numerous to properly thank. Liam for helping it seem like I have the faintest clue about physics, Brad similary for computing and for the stress decompression of rant-filled late night strolls. Max for the much needed zany factor, how I ever ended up buffooning around directionless on stage owes a great deal to his raw enthusiasm. Emrys for his glorious bombast, thirst for impromptu science lessons in cosy watering holes, and wonderful literary gifts in return.

To all my friends from my courses throughout the years, my undergrad was peppered with moments I'll never forget.

I want to thank all the awesome academics and teachers I've had the pleasure learning under, who've given me every opportunity to dive deeper into the science I love.

Lastly I'd like to thank my wonderful family for being the most supportive and pleasant people one could hope for. Their care and perspective kept me sane, and I hope they didn't have to put up with too much because of it. Sadly they can now utterly trounce me at every board game I own.

Glossary

6-31G*: Pople basis set with 6 GTOs representing the core orbitals, 3+1 GTOs representing the valence orbitals, and polarisation functions on all non-H atoms

9AA: 9-Aminoacridine; a DNA intercalating monomer. The methylene linked dimers are referred to as C-*n*

ADF: Amsterdam Density Functional; a modelling suite. Includes STOs and ZORA for DFT

AM1-BCC: Austin Model One with Base Charge Correction; the method used by Amber to assign partial charges to atoms in novel organic molecules.

AMBER: Assisted Model Building and Energy Refinement; an MD simulation package

BDA: Bond Detached Atom; the division point between FMO fragments used to ensure that every fragment is a closed-shell system

C-*n*: Dimeric di-9-aminoacridine ligands with a linker length of *n* methylenes

C3NC3: Dimeric di-9-aminoacridine ligand with a dipropylamine linker

D-*n*: Dimeric dithiolato-linked (2,2':6',2''-terpyridine)platinum(II) ligands with linker length of *n* methylenes

DFT: Density Functional Theory

ECP/MCP: Effective/Model Core Potential; a method in which core electrons in a basis-set are replaced with a more efficiently computed psuedo-potential

ff12SB: AMBER's 2012 protein and DNA force-field

Facio: A graphical interface aiding in the preparation and analysis of FMO calculations

FMO: Fragment Molecular Orbital; a method by which large systems are fragmented into chemically consistent subunits and system properties are computed from fragment dimers

GAFF: Generalised Amber Force Field; the default parameter set used in AMBER for novel organic molecules

Gaussian: A computational chemistry program specialising in electronic structure calculations

GAMESS: General Atomic and Molecular Electronic Structure System; an *ab initio* quantum chemistry package

GTO: Gaussian-Type Orbitals; computationally efficient linear combinations of GTOs are used to approximate orbitals

HF: Hartree-Fock; a method to approximate the wave-function of a system using a Slater

determinant (where each orbital is expressed in a determinant form, which guarantees anti-symmetrisation of the wave-function). Usually the lowest QM level of theory in a calculation.

LANL2DZ: Los Alamos National Laboratory 2 Double Zeta ECP

M06: Truhlar's 2006 hybrid Minnesota Functional

M-4: Monomer (butane-l-thiolato)-(2,2,':6',2''-terpyridine)platinum(II) ligand. The dimers are refered to as D-*n*

MD/MM: Molecular Dynamics/Molecular Mechanics

MMBPSA: Molecular Mechanics Poisson-Boltzmann Surface Area; a MD trajectory post-processing module which uses a Hess's law like relation to calculate $\Delta G_{solvation}$

MP2: Møller–Plesset perturbation theory to the 2^{nd} order; allows inclusion of electron correlation effects

NAB: Nucleic Acid Builder, an AMBER module for generating DNA structures

NE: Neighbour Exclusion; the principle that chromophores cannot occupy adjacent intercalation sites

PCM: Polarisable Continuum Model; QM solvation with a polarisable dielectric medium

PIE: Pair Interaction Energy; the energy of interaction between two fragments in a system calculated *via* FMO

PIEDA: Pair Interaction Energy Decomposition Analysis; divides pair inceration energies into chemically intuitive contributions

PME: Particle Mesh Ewald; allows infinite 'tiling' of water-box to reproduce bulk properties

RI: Resolution of Identity; accelerates correlated calculations by transforming 4-centre integrals to 3- and 2- centre integrals through the use of a large auxiliary basis-set

QM: Quantum Mechanical

SCF: Self-Consistent Field; orbitals are solved iteratively in the 'mean field' of other electrons

SCS: Spin-Component Scaling; same-spin and opposite-spin interactions given different scaling factors

STO: Slater-Type Orbitals; a more physically accurate but less computationally efficient representation of orbitals

TIP3P: Transferable Intermolecular Potential – Three Point-charges; a popular water model

TZP/TZ2P: Triple-zeta basis sets with one and two polarisation functions respectively

Zeta: The number of basis functions used to represent a valence orbital, double (DZ) *etc.*

ZORA: Zeroth Order Regular Approximation; corrects for relativistic effects on orbitals

Table of Contents

Chapter 1: Introduction

1.1. Intercalation and Its Therapeutic Significance

DNA intercalation is the process of insertion of molecules with planar aromatic moieties (chromophores) between base-pairs of the DNA helix. Intercalation is a non-covalent interaction which is favourable due to electrostatic attraction between the (usually) positively charged intercalator and the negatively charged DNA, aromatic stacking overlap upon insertion, and shielding of the hydrophobic chromophores from the surrounding solvent. Intercalators are usually similar in size to nucleobases, and their intercalative mode is influenced by the length of the major axis of the chromophore, as well as the position of the chromophore's substituents.[1] Intercalators which possess major axes of the length of a base-pair's span or smaller typically intercalate with their major axis parallel to the Watson-Crick pairing of nucleobases, and those with longer major axes 'spear' perpendicularly into the DNA helix. Upon intercalation the DNA helix typically unwinds and extends in length to accommodate the inserted molecule.

Intercalation was first identified in the 1960's[2] and has been an active area of study and application since.[3] Intercalating compounds have found widespread biological use: in molecular biology contexts[4] (such as fluorescent staining with ethidium bromide) or in therapeutic applications, particularly as chemotherapeutic compounds; often operating as transcription inhibitors[5,6] or topoisomerase I or II poisons.[7,8] As topoisomerase poisons, the intercalator stabilises double-stranded breaks in DNA formed by topoisomerase during DNA repair/replication and hence prevents religation of the DNA strand, eventually triggering apoptosis. This class of drug shows high efficacy, and gains selectivity for cancerous cells due

to their upregulation of topoisomerases and more rapid replication rate, which increases the likelihood of forming stabilised complexes. Clinical intercalators have gone through many generations: progressing from mono-intercalators to linked bis-intercalating dimers and then poly-intercalating agents. Variations on chromophore moiety and linker functional groups have also been synthesised in order to influence their method of binding and increase their potency.[9,10] As such, a deeper understanding of the structural, dynamic, and energetic details of intercalation would be informative to the design of newer generations of chemotherapeutic agents.

1.2. Neighbour Exclusion

The neighbour exclusion principle is an empirically derived principle stating that intercalators cannot occupy adjacent base-pair steps on a DNA strand *(see Figure 1 below)*.

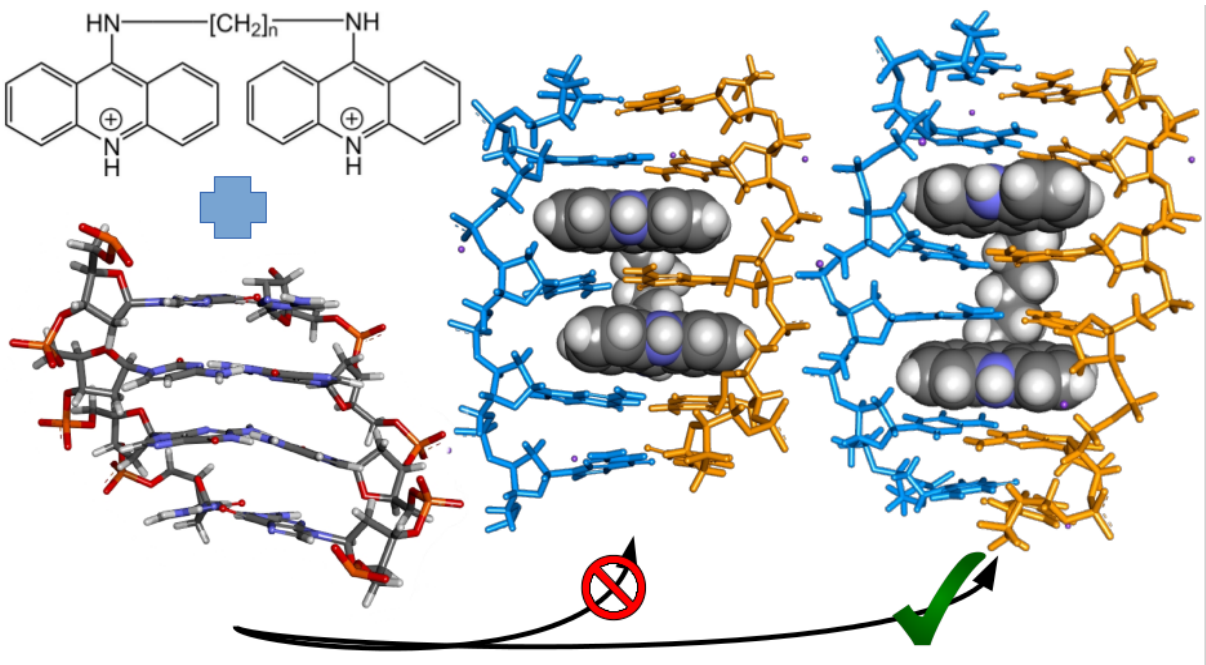


Figure 1: A representation of a bis-intercalator (top left) *inserting into DNA* (bottom left) *forming either a 1 base-pair sandwich* (*1bps*, middle) *or 2 base-pair sandwich* (*2bps*, right). *According to NE the 1bps should be forbidden but there are experimental indications of cases where ligands are bis-intercalated while their linkers are only long enough to form a 1bps.*

Neighbour exclusion (NE) has direct, measurable implications to the binding modes of intercalators. In the case of mono-intercalators this means that saturation will occur when only half the sites are occupied, *i.e.* when there is a 1:2 ratio between the drug and possible intercalation sites rather than a 1:1 ratio. For bis-intercalators (dimeric compounds in which chromophores are tethered by a linker 'chain'), whether one or both of the chromophores intercalate (defined as monofunctional or bifunctional binding respectively) is determined by whether the linker is able to span across the perpendicular axis length of either one base-pair or two base-pairs, forming a one base-pair sandwich (1bps) or a two base-pair sandwich (2bps) complex respectively.

1.3. Reported Violation of Neighbour-exclusion

While neighbour exclusion is seen to occur in the majority of intercalative complexes, there are also reported bis-intercalators which appear to exhibit bifunctional intercalation in such a way that they violate neighbour exclusion by forming a 1bps.[11–13] By measuring macroscopic properties such as the DNA helix's unwinding angle and extension length upon intercalation it was determined that the bis-intercalators studied must be binding bifunctionally. In addition, the linker length of these compounds were considered to be too short to accommodate the formation of a 2bps, and therefore a 1bps must be forming in violation of the neighbour exclusion principle. However these conclusions were based upon static space-filling model geometries, and do not account for the dynamic and flexible nature of DNA-chromophore interactions. Molecular Dynamics (MD) simulations should provide a more representative model for these compounds' binding modes without making the assumptions that structural parameters are static and at an equilibrium value.

In particular this work will provide theoretical studies and analysis on the results of

Wakelin *et al.*[11,12,14] on the diacridine series and (2,2':6',2''-terpyridine)platinum(II) thiolate series of bis-intercalators with simple methylene linkers, henceforth referred to as **C-*n*** and ***D*-*n*** respectively where *n* is the number of carbons in the linker chain (*see Figure 2 below*).

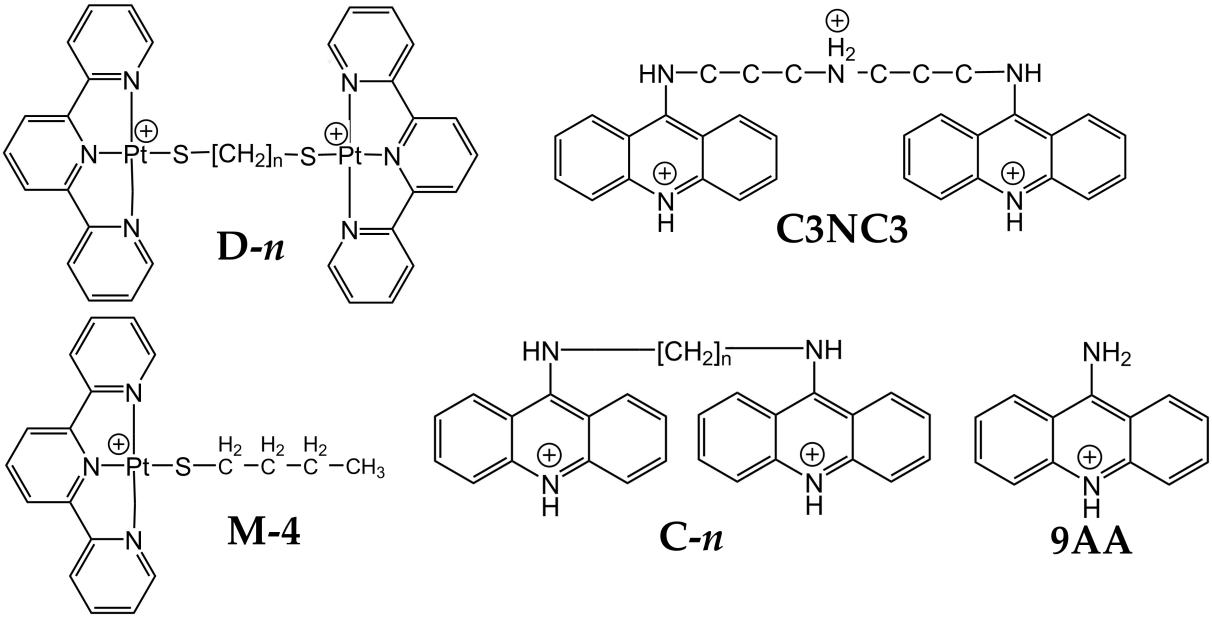


Figure 2: Skeletal formulae and abbreviations for ligands studied in this work. (2,2':6',2''-terpyridine)platinum(II) thiolate dimers (***D*-n**, top left), *corresponding* (*1-butane-thiolato*)-*(2,2':6',2''-terpyridine)platinum(II) monomer* (***M*-*4***, bottom left), *methylene linked diacridines* (***C*-n**, bottom middle), *corresponding 9-aminoacridine monomer* (***9AA,*** bottom right), *and dipropyl amine linked diacridine* (***C3NC3***, top right).

1.4. Theories on the Causes of Neighbour Exclusion

Despite decades of study and application of intercalation, the cause of neighbour exclusion has not been definitively settled. Conformational arguments have been proposed as its cause, including that upon intercalation the DNA's structure is altered such that the adjacent intercalation site is sterically inaccessible,[4,9] or that the formation of a intercalation site requires a specific mixed C2'-*endo* and C3'-*endo* conformation of the sugar-phosphate backbone (*see S1.3. DNA Sugar Pucker Definition*) which can occur at most every second base-pair step.[15,16] Early modelling work on a $(GC)_4$ eight nucleotide strand also indicated that the energy to create a stretch in two adjacent base-pair steps to accommodate intercalators was

greater than the sum of the energy required to stretch each site individually, potentially causing adjacent intercalation to be energetically prohibitive.[17] Additionally, vibrational entropy losses which occur upon violation of neighbour exclusion have been predicted to be a possible cause.[18] Neighbour exclusion violation with charged chromophores involves bringing repulsive charges closer together than in a 2bps and may have an associated electrostatic penalty discouraging the formation of a 1bps. The unwinding of the helix upon intercalation has also been proposed as potentially causing neighbour exclusion as it increases the van der Waals overlap of the bases adjacent to an intercalation site, increasing their stacking stabilisation. As a consequence, breaking of the base stacking at positions adjacent to intercalated sites becomes more difficult, hindering adjacent intercalation.[19] Polyelectrolyte effects have also been proposed as the cause for neighbour exclusion,[20,21] stipulating they alone can explain the anti-cooperativity (*i.e.* binding of intercalators discourages further binding) in experimental binding curves. Polyelectrolyte theory states that DNA is subjected to stresses resulting from repulsion between its anionic groups and between the counter-ions in the condensation sheath on its surface. Upon binding the helix is extended, separating the anionic sections which reduces electrostatic repulsions, and counter-ions are released into the bulk which provides a driving force for intercalation. However each new intercalation reduces these repulsive forces less than the one before it, lessening this driving force and leading to intercalative binding being anti-cooperative. Thus under the scheme of polyelectrolyte theory it is not necessary to postulate a site exclusion model to explain the experimental binding curves.

1.5. Rationale and Aims of This Project

While there has been a considerable amount of theoretical studies on the insertion mechanism

in intercalation[22–25] there is a relative paucity of studies investigating the mechanism which neighbour exclusion. Rao and Kollman's 1987 MM/MD study on the dual intercalation of 9-aminoacridine[18] and Prabhakara and Harvey's 1988 study on actinomycin D's intercalation in Z- and B-DNA,[17] remain some of the most widely referenced computational studies in relation to neighbour exclusion. However due to the limitations in computational power and theoretical methods at the time of these studies only a small number of systems could be simulated, and only on a picosecond timescale, limiting the amount of conformational space able to be sampled.

Due to the phenomenal increase in computer power in the intervening time, coupled with the rapid development of modelling methods and improvement of force-fields, these systems are now able to be treated in a much more physically accurate way and in greater depth. In particular, the porting of MD software to run on GPU hardware has made system sizes and timescales which would have previously required large computer-clusters now computable on consumer level graphics cards.[26] In addition fragment based methods have opened up large systems such as macromolecules to electronic structure methods which would previously have been practically inaccessible due to the prohibitive scaling factors of quantum mechanical (QM) methods.[27]

As such the specific aims of this project were to use modern computational methods to:

1. Provide a more detailed theoretical underpinning to the structural limitations of the binding of a series of diacridine (**C-*n***) and di-(terpy)Pt(II) thiolate (**D-*n***) bis-intercalators to DNA and hence determine whether neighbour exclusion is being violated in these systems.

2. Investigate at what linker length a two base-pair sandwich is able to be formed, and

what implication that has for the experimentally observed transition from monofunctionality to bifunctionality with increasing linker length.

3. Determine if there are any notable sequence effects on the way these intercalators interact with DNA.

4. Calculate Gibbs energies of binding by free energy methods and see if trends in preferences for grooves or sequences can be determined.

5. Calculate entropic effects of binding *via* normal-mode analysis to see if adjacent binding of intercalators has an associated entropic penalty and hence favours neighbour exclusion.

6. Use fragment based electronic structure methods to get quantum mechanical energies of the interaction between intercalators and the components of the bound DNA, and hence obtain a more detailed picture of the stabilising and destabilising forces present.

1.6. Choice of Methodologies

1.6.1. Sequences and Ligands Studied

In order to avoid DNA end-effects (such as fraying at then ends of the strands, or varying electrostatic potential) potentially affecting the conformation of the intercalation site, the sequence was extended by 5 base-pairs on either side of the intercalation site, yielding a 14-mer in the case of a 2bps and a 13-mer for the 1bps. These ends were kept constant for all sequences, thus from the 5' to the 3' end all sequences read: CGATG-[intercalation sequence]-CATCG. A range of intercalation sequences were chosen to be investigated to see if there are sequence specific effects on the nature of intercalation as each nucleobase can cause structural changes in the DNA strand, but also has different stacking and electron donor abilities.[28] The

diacridines and the 9-aminoacridine monomers show little selectivity to particular sequences[12,29] while in contrast the (terpy)Pt(II) thiolate monomer shows a marked selectivity for G·C base-pairs steps, however this appears to be eliminated in the corresponding dimers.[12] This lack of selectivity means there are combinatorially very many probable intercalation sites, but in general intercalators tend to have more favourable insertion at pyrimidine-purine base-pair steps. This can be rationalised by the decrease in overlap of the adjacent nucleobases compared to in pyrimidine-pyrimidine or in purine-purine base-pair steps. Subsequently, pyrimidine-purine steps have decreased van der Waals stabilisation energy and a lower energy barrier to breaking the stacking interactions and forming an intercalation site. Hence the pyrimidine-purine base-pair steps of CGCG, CACA, and TATA were studied for 2bps complexes and CGC, CAC, TAT, TGT were studied for 1bps complexes. For brevity all complexes will be referred to by only the intercalation site sequence, the CGATG 'top' and CATGC 'tail' can be taken as implied.

The transition to bifunctionality in the dimers was determined experimentally to be at **C-6** in the diacridine case[11] (with **C-5**'s functionality being ambiguous) and at **D-5** in the di-(terpy)Pt(II) thiolates (with **D-4**'s functionality similarly ambiguous).[14] Unfortunately a clearly monofunctional (terpy)Pt(II) thiolate ligand was not observed as **D-3** and below could not be isolated experimentally.[14] Thus the ligands **C-4** to **C-6** were modelled in this study as 1bps and 2bps in all sequences, with **C-7**, **C-8** and **C3NC3** being studied as only 2bps due to the longer linker length in these latter diacridines. **D-4** to **D-7** were studied as both 2bps and 1bps for all sequences. In addition, the dual intercalation of **9AA** and **M-4** monomers were studied in each 2bps and 1bps intercalation sequence for comparison as cases where there is no linker influencing the mode of binding. Intercalation with the linker chain occupying the minor or major groove was studied in all cases. Each DNA sequence was also simulated

unintercalated for comparison as a native structure. In total this yielded 151 unique combinations of DNA-intercalator complexes which were studied *via* both MD and fragmentation molecular orbital (FMO) calculations.

1.6.2. Combining the Molecular Dynamics and Fragment Molecular Orbital Methods

The molecular mechanics (MM) framework is an entirely parameterised method in which all atoms/bonds in the systems are parameterised from a library based on atom type, hybridisation, position in moiety *etc.*, which are collectively referred to as the force-field. All energy potentials are in a simple functional form, typically as a harmonic potential with respect to some reference equilibrium geometry, thus enabling rapid evaluation of the energy of a particular configuration. In molecular dynamics the system is put into motion and its trajectory evolved according to the constraints of the force-field and mechanisms of classical mechanics such as Newton's laws of motion. In this way the important chemical interactions of a system can be studied over a time frame, providing a picture of not only its structural but also its dynamical properties.

For macromolecular systems, and in particular biomolecules, MD methods are the most commonly employed theoretical methods due to: 1) the size of biomolecules being prohibitively large for other modelling methods, 2) the greater reliance of biomolecules on conformational and dynamic features for structure and function, 3) the propensity for the structure and activity of biomolecules to be determined by non-covalent interactions as opposed to covalent interactions in other chemical systems, and 4) their construction by repetition of chemically consistent subunits (*e.g.* amino-acids, nucleotides). However the results given by MD methods are limited by the robustness of the parameterisation of its force-field. As such the AMBER force-field was chosen as it was specifically developed to

model nucleic acids[30] and has also been explicitly validated for stacking interactions with reference to QM calculations.[31,32]

However, while MD may provide good structural and dynamic details with respect to QM calculations and experimental structures, by its nature it cannot reproduce effects which were not included in its parameterisation. This includes the inability to break or form bonds during a trajectory, as well as being only able to provide qualitative, relative energy estimates. QM/MM methods have enjoyed wide popularity to augment MD as they allow the definition of an 'active site' treated with QM methods which can properly model complex chemical interactions while the bulk of the system is treated with a force-field.[33] While these methods have shown successful application to systems like proteins which have a small active site requiring QM treatment, QM methods are generally feasible for systems of around 100 atoms. In the case of modelling DNA bis-intercalators one would need to apply QM methods not only to both chromophores but also their flanking base-pairs in order to gain an improvement in the treatment of the intercalation site's interactions (*e.g.* stacking interactions), hence leading to a QM section of several hundreds of atoms. This would be unfeasible for high-accuracy QM methods, and more tractable QM calculations of lower accuracy are unlikely to be able to properly represent dominant intercalation forces such as dispersion. QM/MM methods would also dramatically curtail the length of MD simulations possible, as updating the QM section becomes the slowest part of the calculation. In addition, QM/MM methods have not yet been fully implemented on GPUs, forcing the calculations onto CPUs which would decrease the computing speed by orders of magnitude.

As an alternative, fragment based methods can be used on the geometries acquired from MD trajectories. Fragment based methods work on the assumption that most chemical interactions are localised and that large systems can be divided up into chemically consistent

subunits, maintaining delocalised properties (such as those arising from conjugated systems) within a single fragment. In the case of biomolecules the choice of fragments is usually obvious, amino-acids for proteins and nucleotides for nucleic acids. This fragmentation approach avoids the size-scaling issues which limit QM methods to systems of around 100 atoms, and trivially parallelises the calculation of the system, allowing it to be divided amongst many CPUs (such as those in large scale computer clusters) with little loss in efficiency. The Fragment Molecular Orbital method[34] (FMO) was chosen for this project (*see Figure 3 below*). FMO has the advantage of a rather natural fragmentation scheme, avoiding the need to introduce extra atoms as 'caps' on fragmented bonds. FMO allows the use of these uncapped fragments by applying heterolytic bond fragmentation, assigning two electrons and one proton from a bond to a fragment such that it results in two closed-shell and neutral fragments. In addition, the system's monomers and dimers are solved to self-consistency in a Coulomb “bath” of the entire system's electrostatic potential and as such the electron density remains accurate without ever having to do a full molecular orbital calculation on the entire system.[35] Dimers which have significant separation are assumed to only have long range interactions and thus most dimers are treated *via* only a Coulomb operator. Thus FMO allows high level QM calculations to be tractably applied to large systems with very little loss of accuracy compared to unfragmented calculations.[36] FMO also provides the advantage that through its dimer calculations the specific interactions of each subunit with each other (in this case nucleobase-chromophore interactions) can be explicitly computed and hence the important stabilising and destabilising interactions investigated on a detailed energetic level.

In this way the use of a combination of MD and FMO on realistic DNA-intercalator systems can provide atomistic simulations of trajectories on nanosecond length scales *via* the MD calculations, as well as high-accuracy QM energy calculations which provide a detailed

energetic break-down of all the inter-residue interactions in the system.

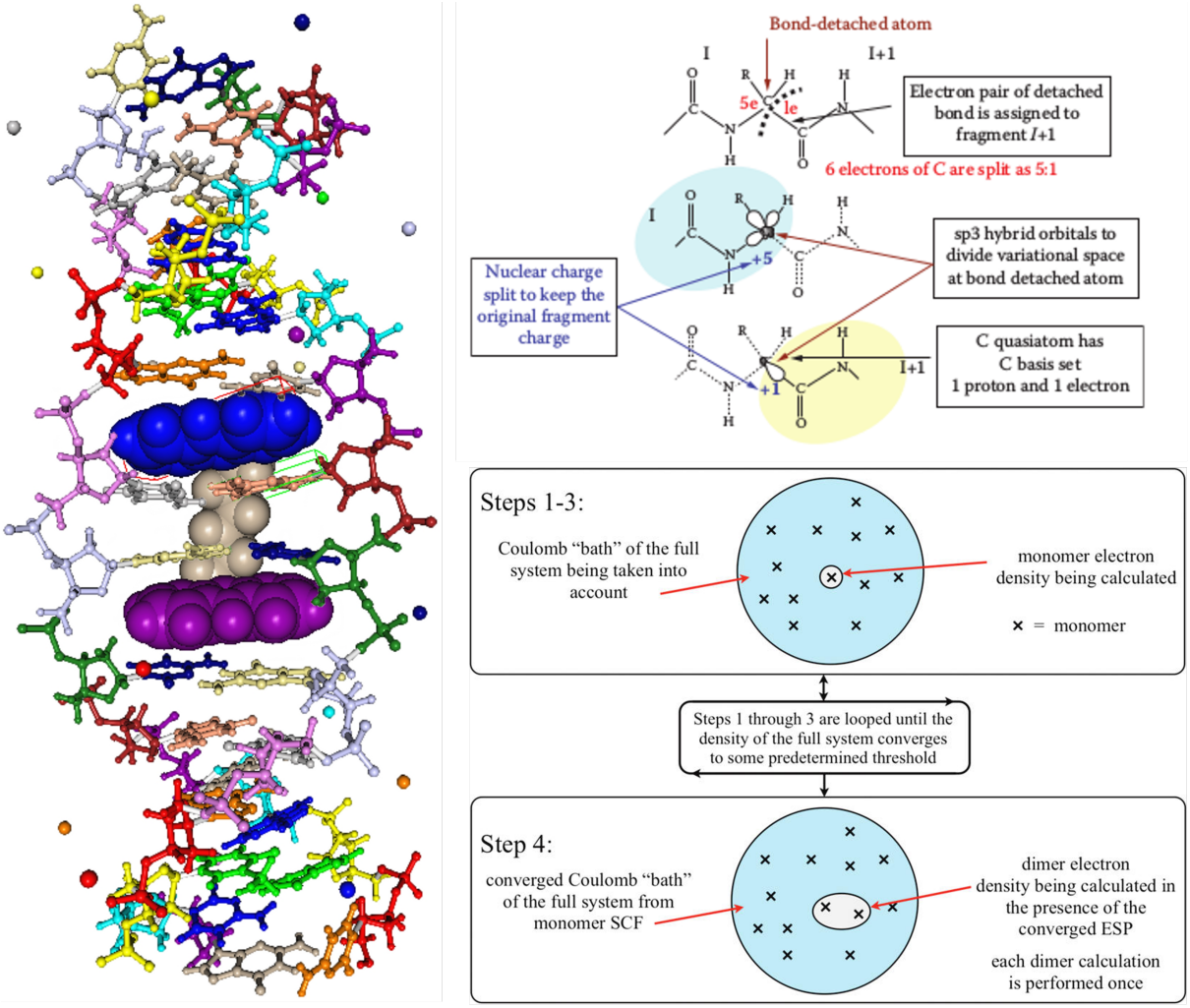


Figure 3: Diagrams detailing the FMO approach. Left: *The system is divided up into chemically consistent fragments (each fragment represented in a different colour*). Top right: *the bonds are fractioned not by capping but by "bond-detached atoms" (BDA) and the* sp^3 *bond is corrected for* via *an orbital projector (diagram from ref. 80).* Bottom right: *All fragments are solved in a "Coulomb bath" of the entire system, first the monomers are iterated to self-consistency then the dimers are calculated pair-wise (diagram from ref. 27).*

The efficiency of these methods allows a large number of systems to be modelled, enabling a series of ligands and a range of intercalation sequences to be studied using practical amounts of computer time. This will provide theoretical grounding for interpreting the often ambiguous experimental results on the bis-intercalation of the DNA discussed earlier, as well as insight into the interactions dictating the formation of either a 1bps or a 2bps.

Chapter 2: Methodology

2.1. Molecular Dynamics

All MD calculations were performed using the AMBER14 and AmberTools14 software packages.[37,38] DNA structures were created in the Nucleic Acid Builder (NAB) module with the canonical B-type double-helix. Intercalated complexes were created with the xLEaP module by manually inserting the ligands into the requisite DNA base-pair steps. These initial structures were then minimised in 20 lots of 5000 cycles, beginning with 500 kcal mol^{-1} $Å^{-2}$ harmonic restraints on both the DNA and intercalator and then easing off the restraints, first those on the DNA and then those on the intercalator, prior to one final unrestrained minimisation. These structures were then subject to equilibration using 22 lots of 10 ps dynamics at 293 K using the Langevin thermostat with a collision frequency of 3 ps^{-1}, again starting with 500 kcal mol^{-1} $Å^{-2}$ restraints and decreasing over each lot, prior to a final 14 lots of unrestrained dynamics. Finally, the 10 ns of production dynamics were done in 100 lots of 100 ps, again at 293 K using the Langevin thermostat with a collision frequency of 3 ps^{-1} (*refer to S2.2. for full input files of the MD protocol*s). All production MD simulations were unrestrained. All dynamics were performed at constant pressure using the Particle Mesh Ewald (PME)[26] method with periodic boundary conditions and a cut-off of 10 Å. The SHAKE algorithm was used to constrain hydrogen bonds. A 1 fs time-step was used for all dynamics. All MD simulations were performed in explicit solvent using a truncated octahedral water-box whose faces were at least 14 Å away from the solute and, enough Na^+ counter-ions to neutralise the system. TIP3P water[39] was used for the explicit solvent molecules, alongside the respective Joung/Cheatham ion parameters for the counter-ions.[40]

The AMBER ff12SB force-field was used for standard atom types, while non-standard

atoms were defined from the General AMBER Force Field[41] (GAFF, Ver. 1.7) after being parameterised *via* standard procedures using the Antechamber module with the AM1-BCC partial charge method.[42,43] The (terpy)Pt(II) thiolate intercalators required manual parameterisation, which is discussed below.

2.2. Parameterisation of (terpy)Pt(II) Thiolate Ligands

The AMBER force-field does not contain the requisite values for the (terpy)Pt(II) thiolate moiety, *i.e.* the atomic partial charges and the parameters which define the change in energy with respect to change in bond/angle/dihedral values. Therefore new parameters were derived for these systems in order to perform MD simulations with the di-(terpy)Pt(II) thiolate intercalators. Atom types were manually defined according to GAFF specifications.[41] The (terpy)Pt(II) thiolate chromophore with a methyl group on the sulfur was geometry optimised. The ADF modelling suite[44–46] was used for the optimisation due to its use of more realistic Slater-Type Orbitals (STO) as basis functions, as well as its good handling of relativistic effects on electron orbitals, which are particularly pronounced for Pt and its neighbouring elements. The M06/TZP (STOs) level of theory with enforced Cs symmetry, no frozen core, and scalar zeroth order regular approximation (ZORA) correction[47] for relativistic effects was used for the optimisation. The equilibrium MD parameter values were taken from this geometry optimised structure. The Pt non-bonded parameters were taken from a review on the van der Waals radii of Pt and Pd in MM modelling.[48]

A single-point calculation was then performed on the optimised geometry (*see Figure 4 below*) at the same level of theory but with a TZ2P basis set. Partial charges were then derived from this single-point calculation using the Multipole Derived Charges method[49] to the quadrupole level (MDC-q). This in essence works by assigning partial charges

on atoms such that the calculated molecular multipole distributions are reproduced by these assigned atomic point-charges. While all assignments of partial charges are necessarily arbitrary, as partial charges are not QM observables, MDC-q reproduces the molecular dipoles and quadrupoles exactly and thus realistically reflects the electronic distribution of the entire molecule.

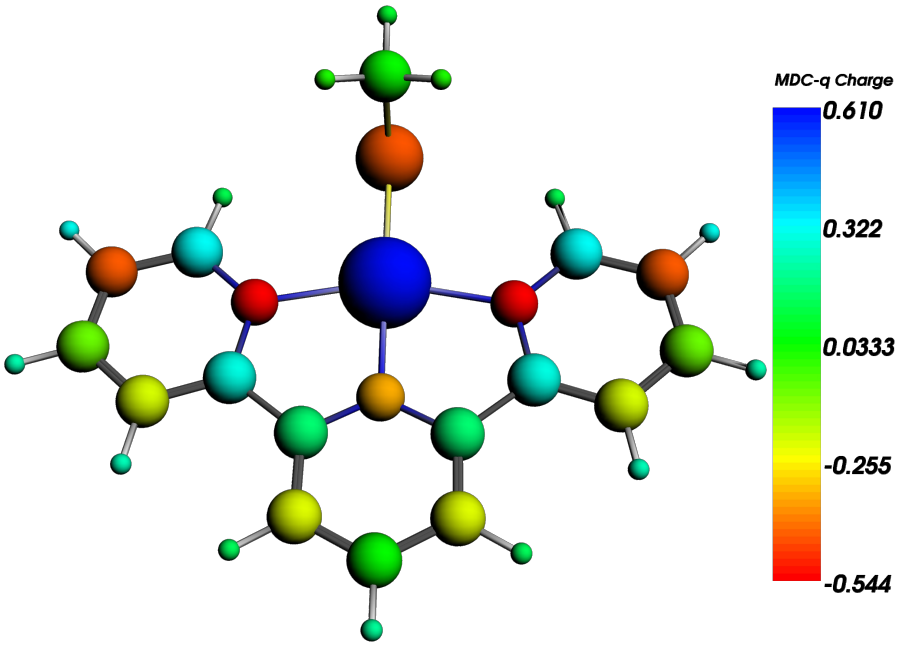


Figure 4: Optimised geometry and MDC-q partial charges of (terpy)Pt(II) thiolate monomer with methyl chain. The level of theory used was: M06/TZ2P//M06/TZP (STOs) + scalar ZORA + no frozen core.

Force constants were derived from energy changes during relaxed scans along the relevant bonds, angles, and dihedrals. As the relaxed scans involve geometry optimisations and energy calculations at each step along the scan, a more computationally efficient method was used for the scans than was used for the geometry optimisation. Relaxed scans were performed using the Gaussian 09 (Rev D.01) program,[50] using the M06/6-31G** level of theory with the LANL2DZ Effective Core Potential (ECP) for the Pt's core orbitals. Step sizes in the scans were 0.025 Å for bonds, 2º for angles, and 0.5º for dihedrals. Curves were then fitted to the energy profiles of the scans; parabolic for bonds and angles, and sinusoidal for dihedrals in

accordance with the functional form of the AMBER force-field.[41] The coefficients of these curves were then used to determine the force constants. The harmonic approximation of bonds and angles becomes invalid at significant distortions from equilibrium (the approximation capturing less of the physical profile of a bond than, say, a Morse potential) and as such the curves were fit over a range 0.35 Å for bonds, 30º for angles, and 15º for dihedrals, with the exception of the N-Pt-S-C dihedral which was fit over a full 360º scan. This was done as the N-Pt-S-C dihedral could have a large degree of conformational flexibility, being unencumbered by constraints of aromatic planarity present in other dihedrals, and as such is likely important for determining the overall dynamical conformation of the di-(terpy)Pt(II) thiolate dimers. This curve was fit using a genetic algorithm developed by Sergi Ruiz[51] which allowed multiple parameters to be varied at once, as well as construction of a fitted curve by addition of multiple sinusoidal curves, thus allowing a better fit to the DFT data.

2.3. Free Energy and Entropic Calculations

Free energies were calculated from the 10 ns of MD using the Molecular Mechanics Poisson-Boltzmann (MMPBSA.py) module with the Poisson-Boltzmann (PB) solvation method and entropic corrections *via* harmonic normal-mode calculations in NAB. Normal-mode analysis was performed on each frame occurring at 12.5 ps intervals. The ionic strength was set to 10 mM in all calculations to match experimental conditions. The 1-trajectory approach (where receptor, ligand, and complex geometries are all taken from one MD run of the complex) was employed initially with a linear PB solver, however this was found to be inaquate for studies on these systems. Separate MD runs were therefore preformed on free ligands and free DNA sequences as input to calculations using the 3-trajectory approach on all systems (*see Figure*

11 in section 3.4). In addition, the non-linear PB method was employed as this provides a better treatment of the solvation for highly charged systems such as DNA, and the internal dielectric constant inside the DNA strand was increased to 4.

2.4. Fragment Molecular Orbital Calculations

FMO calculations were performed with the GAMESS package[52,53] using the Facio (ver. 18.7.4) interface[54] to aid generation of input files. The fragments were generated by fractioning along all the nucleobase-glycosidic C-N and phospodiester C-O bonds in the DNA. The intercalating dimers were fractioned along the first available sp^3 C-C bonds closest to the attachment point of the chromophore in the linker, so that the interactions of each of the chromophores could be examined separately. The double-counting of energies with bonds connected through bond detached atoms (BDA) was corrected for using the process of subtracting the BDA energies of model ethane systems as detailed in Fedorov & Kitaura's 2007 paper.[55]

Second-order Møller-Plesset perturbation theory (MP2) was used for all FMO calculations. MP*n*-theory is a perturbative method where the Hamiltonian of the zeroth order wave-function (the one-electron Fock matrix formed as the basis for most QM calculations) is perturbed *via* a correction term to represent the *n*-th order Hamiltonian by evaluating up to the *n*-th order wave-functions of a system. Evaluation of the wave-function to the 2nd order is the first point the effects of electron-electron correlation on orbitals is included and thus makes MP2 one of the most efficient methods for including the effects of correlation in calculations. The Hartree-Fock (HF) or DFT methods do not include the electron correlation effects in dispersion forces, vital for representing stacking interactions, and hence are inappropriate for the systems studied in this project. While the applicability of DFT methods to systems with

dispersion interactions can be improved with empirical dispersion correction factors,[56] MP2's inclusion of dispersion is explicitly calculated from the theory and thus remains the most accurate choice in explicitly correlated *ab initio* methods before moving to the costly coupled-cluster methods, whose large scaling of computational cost with system size makes them currently prohibitive for anything but modestly sized systems.[32,57] To speed up the correlated components of the calculations the Resolution of Identity approximation (RI) was used, which transforms costly 4-centre integrals into more rapidly evaluated 3- and 2-centre integrals by use of a large auxiliary basis set which covers the interaction space of the interacting orbitals.[58] In fact, when using RI-MP2 with FMO the dimer self-consistent field (SCF) portion of the calculation becomes the most time consuming component.[59] Spin-component scaling[60] (SCS) was applied to all correlated calculations. SCS is a method in which the interaction energy between opposite-spin and same-spin electrons are scaled by separate factors (6/5 and 1/3 respectively according to Stefan Grimme's SCS scheme)[60] in order to compensate for deficiencies in default MP2's treatment of certain interactions energies such as overestimation of stacking energies.[61]

The RI-SCS-MP2(full)/6-31G* level of theory was used for all FMO calculations, with cc-pVDZ as the auxiliary basis set. A similar methodology has recently been benchmarked on model DNA systems, including variations on fragmentation schemes and electrostatic potential approximations, and it was found to be suitable in reproducing DNA's stacking and hydrogen bonding interactions.[62] Pair interaction energy decomposition analysis (PIEDA) was used to divide the pair-interaction energies (PIEs) between fragments into its more chemically intuitive electrostatic, dispersional, exchange-repulsion (analogous to steric repulsion), charge-transfer, and solvation components, allowing for a more mechanistic understanding of which particular forces have significant effects during intercalation.[55,63] A

comparable level of theory combined with the PIEDA method has previously been identified as suitable for evaluating DNA complex interactions.[64] A TZP model core potential[65,66] (MCP) was used for the core electrons of the Pt atoms. In addition, the Direct Inversion of the Iterative Subspace (DIIS) method with dampening was used for systems containing Pt atoms to obviate potential convergence issues. The Polarizable Continuum Model (PCM[1(2)])[63] method was used for implicit water solvation, and GAMESS's in-built van der Waals radii were used in constructing the solute cavity, with the Pt radii being set at 1.7 Å.[48] The number of tessera constituting the solvation surface was increased to 240 and the dispersion, cavitation, and induced-charge compensation methods were used for the PCM calculations.

2.5. Analysis and Custom Code

2.5.1. Cluster Analysis and Selection of a Representative Frame

Due to the prohibitive computational expense of performing FMO calculations on even a small subset of the structures generated *via* MD trajectories, a molecular structure representative of the complex's dominant conformation for each DNA-ligand system was selected *via* a clustering and minimisation protocol (*see Figure 5 below*).

The entire 10 ns of MD was subjected to k-means clustering using the kclust program from the MMTSB toolset.[67] This method groups the MD structures into clusters based upon structural similarity, and generates a centroid for each cluster. The centroid is the average of the atomic positions of all structures inside the cluster. Various values for the clustering radius (the maximum root mean square variation in distance a frame in a cluster can have from the cluster's centroid) were tried and a radius of 3 Å was found to give sensible clusters with notable conformational differences. Radius values below 3 Å gave many clusters which were only trivially different from each other, and values above 3 Å tended to group the entire 10 ns

trajectory into one cluster, providing no information on conformational variation during the trajectory. This process was automated *via* a shell script (*see S3.3. generate_clusters_3A.sh*).

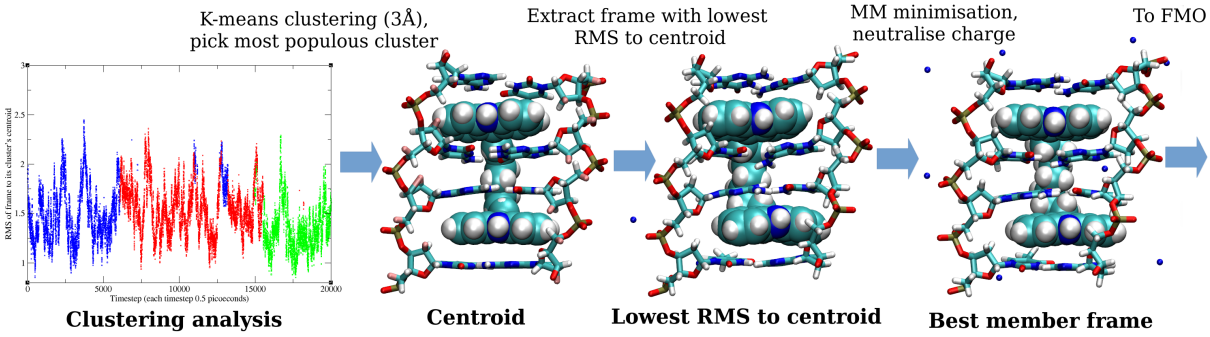


*Figure 5: Flow diagram of the clustering and minimisation process used to generate a representative geometry from MD for use in FMO calculations. The intercalation site of CACA with **C**-7 in the minor groove is displayed.*

The centroids of the clusters cannot be used for FMO calculations as this averaging of atomic positions can generate unphysical structures. Instead, the MD frame with the lowest root mean square variation in atom locations compared to the centroid of the most populous cluster was chosen as representative of the complex's dominant conformation, referred to the as the 'best member' frame. As each individual MD frame can contain transient bad contacts, aromatic non-planarity, *etc.*, the best member frame was subjected to a brief 250 cycle minimisation to ensure that geometrical properties were in local minima while avoiding deviating from the global conformation of the centroid. This ensured that aromatic systems were planar, allowing proper treatment of stacking interactions for FMO calculations, and that distortions in the intercalation site were due to valid conformational constraints rather than temporary variations during MD. All solvation and counter-ions were stripped from the minimised best member frame as the location of counter-ions is highly mobile during MD. The Na+ counter-ions were then re-added to the structure *via* the xLeAP module such that they were placed on a 1 Å resolution Coulombic potential grid so that they provided the maximum amount of charge neutralisation, a requirement for attaining good FMO energies from DNA systems.[62] This minimised best member structure with counter-ions was generated *via* a shell

script developed for this project (*see S3.4. minimise_bestmember.sh*) and thus one structure representative of each complex's MD trajectory was generated for use in FMO calculations.

2.5.2. Generating Queryable and Plottable Interaction Energies

The FMO procedure generates pair interaction energies (PIEs) between each fragment pair in the system, *i.e.* $\frac{n(n-1)}{2}$ PIEs for *n* fragments, and as such 151 systems with *circa.* 80 fragments each generates large amounts of data which needs to be made searchable in order to be properly analysed. Accordingly a program was developed in Python which allows easy searching and manipulation of the FMO/PIEDA data generated in this project (*see S3.1. PIEDA_mat.py, all code developed for this project written by K. Rowell*). The program searches a set of specified GAMESS output files and assigns each file to a unique complex 'dictionary' which stores the complex's information. Because all files were named according to a naming scheme containing the intercalation site sequence, ligand, and groove position of the linker, PIEDA_mat.py can then automatically reconstruct the duplex sequence and find the chromophore and linker locations in the fragment sequence. Each fragment is created as a fragment 'object' and thus can store information such as their type and charge, as well as a topology of which fragments are stacked or paired with each other. The 2D pair interaction energy (PIE) matrix generated by GAMESS for each complex is then made searchable and short 'rules' were written to query particular complex types, selecting for certain properties based on the information stored in each complex's 'dictionary' *(see S3.1.1. PIEDA_mat.py Usage Examples*). The relevant PIEs are then stored in an easily sortable text file.

In addition to making the FMO data manipulatable, a method to rapidly plot these interactions was required. To this end PIEDA_plot.py was written in the matplotlib[68] Python library to automatically plot data generated from PIEDA_mat.py simply by providing

PIEDA_plot.py's output text file as an argument. In this way the analysis of the vast amount of PIEs was made programmable, automatable, and completely customisable. In addition, validation of this searching and plotting method was provided by comparison of the output of PIEDA_plot.py and Facio's inbuilt PIE visualiser (*see Figure 6 below*).

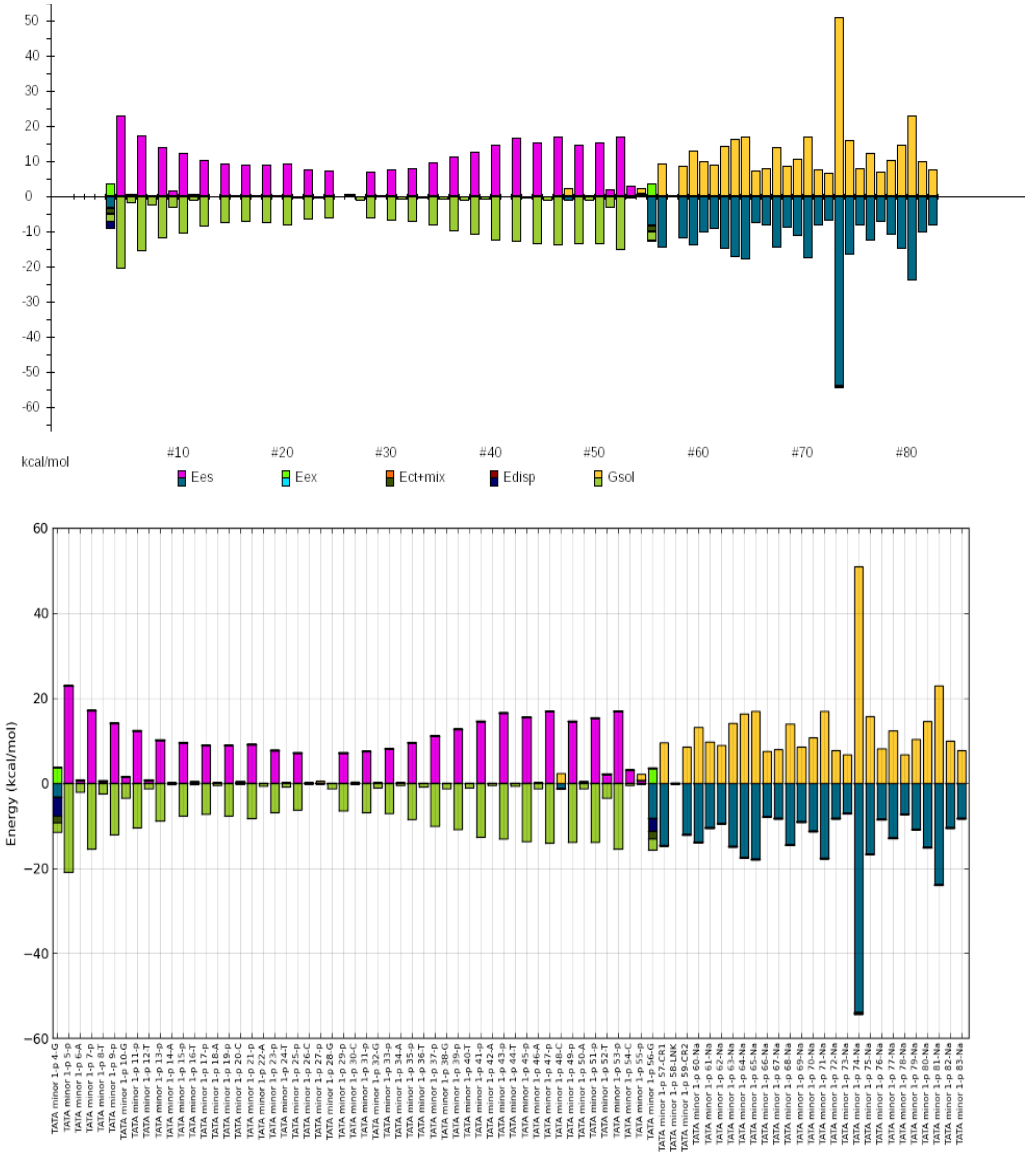


Figure 6: A comparison between plots generated by Facio's inbuilt visualiser (top) *and PIEDA_mat.py + PIEDA_plot.py* (bottom) *demonstrating that PIEDA_mat.py properly assigns all energy values and types to their respective fragments. The PIEDA_mat.py + PIEDA_plot.py procedure has the advantages of being able to be programmed for queries of particular information and is able to search a range of complexes at once. Pictured is the PIEs of the 1st fragment with all other fragments in the complex made by* ***C****-4 binding from minor groove of the TATA intercalation site.*

Chapter 3: Results

3.1. Classical Force-Field Parameters for the (terpy)Pt(II) Thiolate Moiety

Since AMBER lacks in-built force-field terms for Pt moieties, new ones were derived. Initial attempts to construct force-field parameters for di-(terpy)Pt(II) thiolate compounds by working 'by analogy' and using previous MM parameters from analogous bonds and angles on similar systems[69,70] were unsuccessful, failing to reproduce the same structure as DFT calculations upon MM minimisation. Previous MM parameters of a molecule with the same (terpy)Pt(II) thiolate moiety were found in the literature,[71] in which equilibrium values were taken from a related crystal structure[72] and the relevant force constants were estimated with reference to experimental measurements of vibrational frequencies of analogous bonds.[73] These parameters gave a reasonable minimum energy conformation upon MM minimisation, but were found to be unsuitable for dynamics. The approximations used in this approach, such as the use of identical force constants for the Pt-N and Pt-S bonds, made them inadequate for accurate MD calculations. Instead equilibrium geometries and custom force constant parameters were derived from DFT calculations (*see section 2.2*). The DFT (M06/TZP + ZORA) optimised geometry achieved excellent agreement with the related crystal structure,[72] with a mean absolute error between the two structures of 0.013 Å for bonds and 0.72º for angles (*see S4.1.3. for full comparison of bond and angle values*).

MM minimisation using these DFT derived force-field parameters gave a structure in agreement with DFT optimisations, and during MD simulation produced none of the errors due to bad geometry contacts which occurred when the previous parameters were used. The harmonic force constants gave a good approximation to the DFT energy potentials (*see Figure 7 below*) and thus are reliable classical parameters for the moiety's dynamical properties.

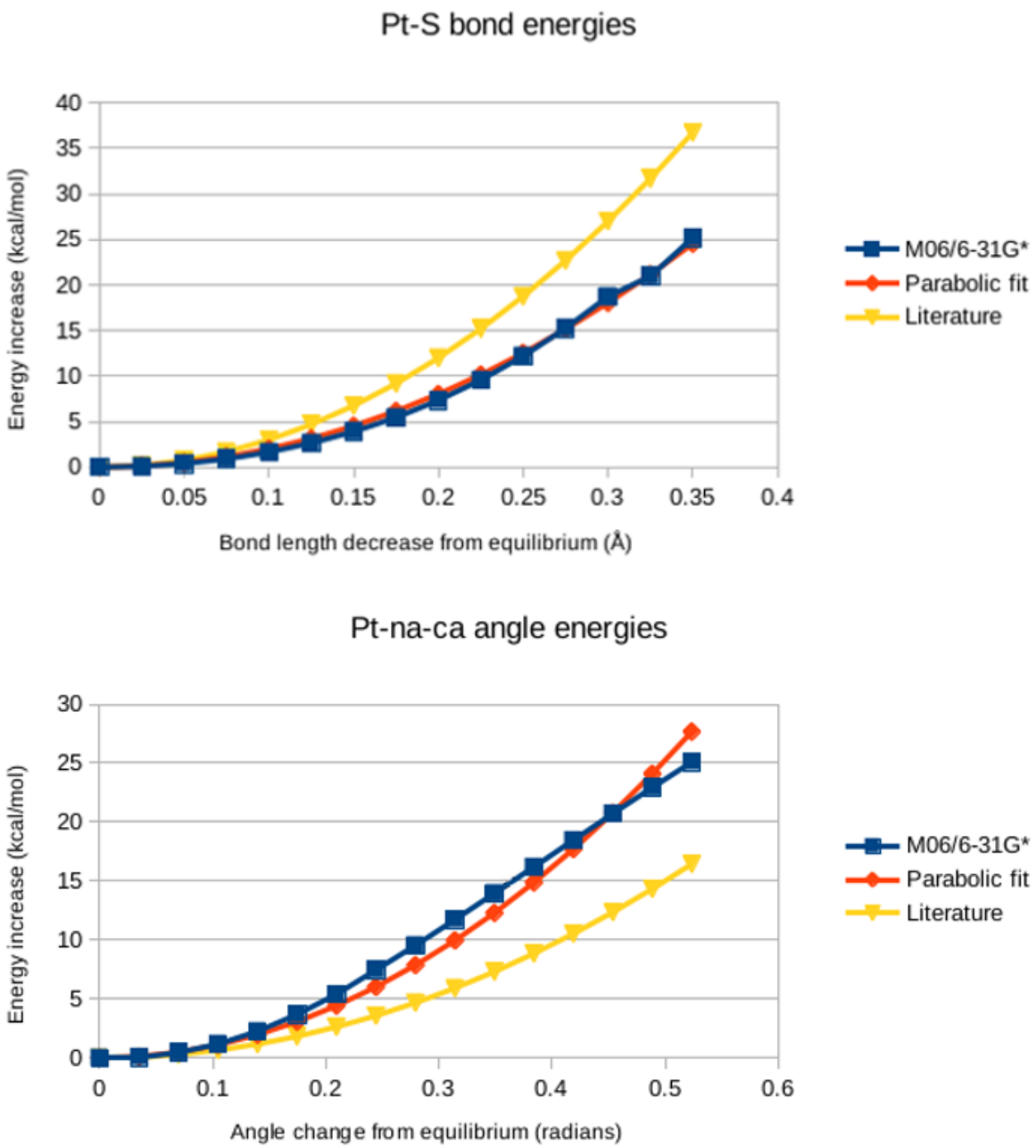


Figure 7: Comparisons the energy profiles of an example bond (top) *and angle* (bottom) *scan. The blue line represents the DFT value, the red line is the profile using the parameters from a parabolic fit to the DFT values, and the yellow line is using the parameter values from reference* 71 (see supplementary DVD for complete parameter fits).

These new parameters for the bonds and angles involving Pt were found to integrate well with the rest of atom types defined *via* AMBER's GAFF force-field[41] (GAFF nomenclature in parentheses), which for the terpyridine moiety involved the sp^2 hybridised nitrogen atoms

with three substituents (na) and the sp^2 hybridised aromatic carbon atoms (ca). The sulfur atom was defined using the thiolate atom type (ss). Table 1 below presents a summary of the relevant force-field parameters and comparison to the literature values. (S*ee S4: Parameter Files for relevant .mol2 unit file and .frcmod parameter file*).

	Classical force constants			***Classical force constants***	
Bond type	**DFT values[a]**	**Literature values[71]**	**Dihedral type**	**DFT values[a]**	**Literature values[71,b]**
PT-na	254.89	300	PT-na-cp-ca	38.53	-
PT-ss	200.40	300	na-PT-na-ca	40	-
Angle type			ca-na-PT-ss	23.49	-
PT-na-ca	100.90	60	ca-na-cp-ca	49.54	-
PT-na-cp	119.12	60	*Bond force constants in kcal mol^{-1} $Å^{-2}$*		
PT-ss-c3	49.79	60	*Angle force constants in kcal mol^{-1} $radians^{-2}$*		
na-PT-ss	87.02	60	*[a]Values derived from M06/LANL2DZ,6-31G** scans*		
na-PT-na	114.63	Excluded	*[b]Used 40 kcal mol^{-1} $radians^{-2}$ planarity term instead*		

Table 1: Summary and comparison of the classical force-field parameters derived from relaxed DFT scans and literature values based on vibration frequencies of analogous bonds and angles. (Refer to S4.1. for complete input files of molecule and force constants definitions).

The MDC-q partial charges derived from the M06/TZ2P singe-point energy calculation (*see Figure 4 in section 2.2*) were found to be appropriately scaled to the other atomic partial charges in the force-field used (*see S4.1.2. for complete partial charges*). Additionally, the MDC-q charges are calculated at a higher and more physically representative level of theory than the AM1-BCC method used by AMBER for generating partial charges of novel compounds, and hence are suitable for use in MD simulations.

3.2. Stability of Intercalation of Complexes

The MD trajectories of the bis-intercalators were analysed to determine whether both chromophores remained stably intercalated during a 10 ns unrestrained MD simulation in explicit solvent. While bis-intercalators are known to spontaneously unintercalate and re-

intercalate one chromophore at a time, allowing them to 'walk' along the DNA sequence, that process occurs on a far longer timescale, typically in the millisecond range. As such observation within the 10 ns of the ejection of a chromophore from the DNA and subsequent monofunctional binding was taken as evidence that the bifunctional binding mode of that complex was not stable. In order to account for the possibility that instability may be an artifact due to the manually constructed bis-intercalated structure containing unfavourable steric repulsions due to too short inter-atomic distances, the initial trajectories which were unstable had their starting structures rebuilt with a new binding geometry devoid of steric clashes and were then resubmitted to MD. The results of these trajectories are summarised below, Table 2 for the diacridines and Table 3 for the (terpy)Pt(II) thiolate dimers.

	C-4 major	C-4 minor	C-5 major	C-5 minor	C-6 major	C-6 minor
TATA (2bps)	**2**	**2**	**2**	**2**	**2**	**2**
CACA (2bps)	**2**	**2**	**2**	**2**	**2**	**3**
CGCG (2bps)	**2**	**2**	**2**	**2**		**3**
TAT (1bps)	**4**				**2**	
TGT (1bps)	**2**					**2**
CAC (1bps)	**2**					
CGC (1bps)	**2**	**4**	**2**	**2**	**2**	
	C-7 major	C-7 minor	C-8 major	C-8 minor	C3NC3 major	C3NC3 minor
TATA (2bps)					**2**	
CACA (2bps)		**2**			**2**	
CGCG (2bps)						

Table 2: Stability of diacridines over 10ns MD, for all intercalation sites, with the linker in the major and minor grooves. Light green: *stable over whole run.* Orange: *tenuous/distorted intercalation.* Light blue: *one chromophore unintercalates during MD. The number inside the boxes reflects the number of MD runs performed.*

It can be seen for the diacridines that a linker chain of only four carbons is generally insufficient to form a 2bps, and that **C-5**'s binding mode is enigmatic (in line with the ambiguous experimental binding data).[11] All ligands with linkers possessing 6 or more carbons were stable over the 10 ns. In contrast almost all ligands could form 1bps from the

major and minor grooves, with the exceptions being the anomalous complexes of **C-4** in TAT with linker in the major groove and **C-4** in CGC with linker in the minor groove.

These trajectories does not reproduce the neighbour exclusion phenomenon, indicating that either MD does not incorporate the physical effect(s) underlying neighbour exclusion or that 1bps are structurally feasible but energetically or entropically unfavourable compared to their 2bps or mono-intercalated counterparts.

	D-4 major	D-4 minor	D-5 major	D-5 minor	D-6 major	D-6 minor	D-7 major	D-7 minor
TATA (2bps)		**2**		**2**				
CACA (2bps)		**2**		**2**				
CGCG (2bps)		**2**		**2**				
TAT (1bps)								
TGT (1bps)								
CAC (1bps)								
CGC (1bps)								

Table 3: Stability of di-(terpy)Pt(II) thiolate ligands over 10ns of MD, for all intercalation sites, with linker in the major and minor grooves. Light green: *stable over whole run.* Orange: *tenuous/distorted intercalation.* Light blue: *one chromophore unintercalates during MD. The number inside the boxes reflects the number of MD runs performed.*

In the case of the (terpy)Pt(II) thiolate dimers, **D-4** to **D-7**, it can be seen that all ligands have linkers long enough to form a 2bps, as the C-S bonds joining the methylene chain to the chromophores are longer than the respective C-N bonds in the diacridines. However minor groove complexes were found to be unfavourable for the shorter linker length **D-4** and **D-5** dimers. This major groove preference was previously predicted and rationalised due to the shape of the terpyridine moiety maximising stacking overlap when the linker is in the major groove and crystallographic evidence of similar monomers binding from the major groove.[12] Again for the di-(terpy)Pt(II) thiolate ligands all 1bps complexes were stable for the entire 10 ns.

The 28 other complexes formed by dual intercalation of the **9AA** and **M-4** monomers

in each sequence, major and minor groove, were also simulated for 10 ns of MD and all complexes, whether 2bps or 1bps, were stable for the entire run. Thus the monomers have no inherent hinderance to forming 2bps in the MD trajectories, and it is the limited linker length in some dimers which forces them mono-intercalated binding over the 10 ns trajectory.

3.3. Intercalation Sites and Structural Considerations

Since MD models indicated **C-6** ligands were capable of forming 2bps (previously ruled out based on space-filling models) it was of interest to examine the structural properties of the intercalation sites with these ligands, renderings of which can be seen in Figure 8 below.

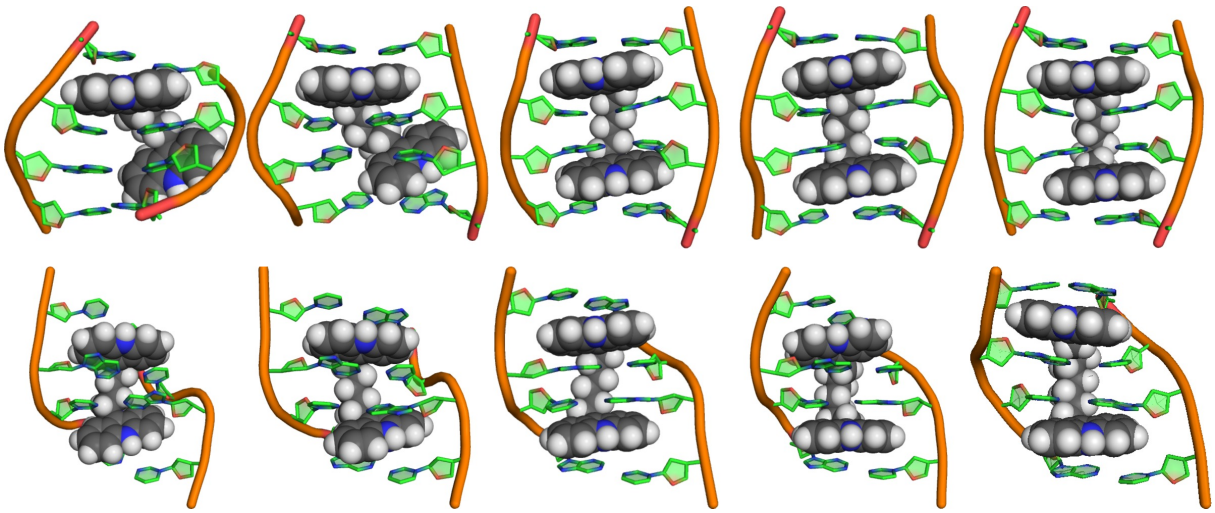


Figure 8: Renders of minimised best member frames from MD of various diacridine 2bps in the CGCG intercalation site. Top row: ***C-4*** *increasing along the row to* ***C-8****, linker in the major groove.* Bottom row: ***C-4*** *increasing along the row to* ***C-8****, linker in the minor groove. The trend of a less distorted intercalation site with increasing linker length can be observed.*

On inspection of the ligand and its flanking base-pairs it can be seen that ligands with smaller linker chains form monofunctional or very distorted bifunctional complexes. With **C-5** complexes, only one chromophore maintains fully inserted intercalation throughout the trajectory, while the other is partially intercalated and relatively mobile during the entire run. **C-6** adopted stable bis-intercalated geometries, however its linker chain is in the fully extended staggered conformation and its chromophores are generally 'splayed out' to increase

the reach of the two chromophores, rather than lying flat inside the base-pair step. There is also associated buckling of the sandwiched base-pairs, again lessening the distance **C-6** has to span to form a 2bps. **C-7** and **C-8** on the other hand appear to comfortably form a 2bps, with their chromophores tending to lie more flat in the helix. In addition slack can be observed in the linker chain of **C-8,** which is evident in the disorder of the position of the linker atoms in the centroids of these complexes.

It should be noted that this trend is consistent with electric dichroism measurements on the roll angle (the angle the intercalator makes with the helical axis, *see S1.1.*). **C-4** has a measured roll angle of 16°, presumably an average value due to one intercalated chromophore with no roll and an unintercalated chromophore at about 32°, **C-5** and **C-6** which show noticeable intercalation site distortion during MD have average roll angles of 11° and 10°, while **C-7** and **C-8** which display 'relaxed' intercalation from MD are both measured to have no roll with respect to the surrounding base-pairs.[11] To analyse this distortion the structural values of the complexes were determined *via* the 3DNA program.[72] Table 4 and Table 5 below provide a comparison between the intercalation site geometries of the complexes formed by **C-6** and **C-8** respectively in CGCG's major groove (*see S5: 3DNA Structural Values for all **C-6** and **C-8** 2bps values, and the supporting DVD for 3DNA output on all complexes*).

CGCG **C-6** major:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GC/GC	-15.29	-18.03	39.09	3	-7.86	-9.51	-5.33	-151	C4'-*exo*	-125.5	C1'-*exo*
CG/CG	0.93	-3.03	-0.15	7.05	-87.45	-15.98	7.48	-118.2	C4'-*exo*	-81	C1'-*exo*
GC/GC	-14.1	1.89	24.98	2.71	20.09	12.47	9.19	-113.6	O4'-*endo*	-120	C4'-*exo*
CG/CG	14.9	4.2	10.77	7.17	14.5	-28.42	3.04	-114.7	C1'-*exo*	-131.4	C4'-*exo*
GC/GC	-9.19	2.09	37.17	3.05	-3.86	13.68	-2.5	-74.7	C1'-*exo*	-104.6	C4'-*exo*

*Table 4: Structural values of the complex with **C-6** in CGCG's major groove. Shown are the five base-pair steps surrounding the intercalation site.*

CGCG **C-8** major:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GC/GC	4.81	-7.49	41.6	3.15	8.43	-11.01	6.09	-104.8	C1'-*exo*	-109	C2'-*endo*
CG/CG	15.68	-2.68	9.09	7.26	10.97	43.51	2.21	-115.3	C3'-*endo*	-78.9	C1'-*exo*
GC/GC	-6.36	2.53	16.08	2.78	26.56	-22.48	8.47	-136	C4'-*exo*	-113.8	O4'-*endo*
CG/CG	21.13	-5.71	5.89	7.28	43.86	-29.71	6.46	-117.4	C4'-*exo*	-108.1	C1'-*exo*
GC/GC	-0.23	-2.48	40.51	3.05	1.56	9.76	1.09	-79.9	C1'-*exo*	-116.3	C4'-*exo*

*Table 5: Structural values of the complex with **C-8** in CGCG's major groove. Shown are the five base-pair steps surrounding the intercalation site.*

In contrast to the 2bps complexes, the diacridines' 1bps geometries show little distortion at the intercalation site regardless of linker length as the linker has less distance to span (*see Figure 9 below*). The 1bps geometries do not agree with the experimental findings that **C-4** binds monofunctionally, while **C-5** and **C-6** have a significant roll angle with respect to the DNA axis. The are only a select few signs of distortion in **C-4** 1bps complexes (TAT with linker in the major groove, CGC with linker in the minor groove), and in general **C-4** 1bps complexes have chromophore roll angles well below the experimental measurement of 16°, while all other 1bps complexes show no significant chromophore roll angle.

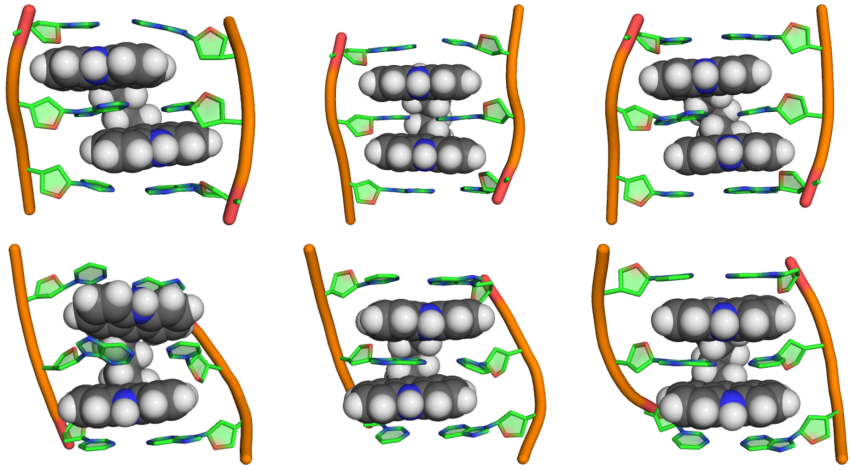


Figure 9: Renders of minimised best member frames from MD of various diacridine 1bps complexes in the CAC intercalation site. Top: *Linker in major groove **C-4, C-5, C-6**.* Bottom: *Linker in minor groove **C-4, C-5, C-6.***

The case of the (terpy)Pt(II) thiolate dimers is similar, with the notable difference that major

groove binding tends to give less distorted intercalation sites than minor groove binding (*see Figure 10 below*).

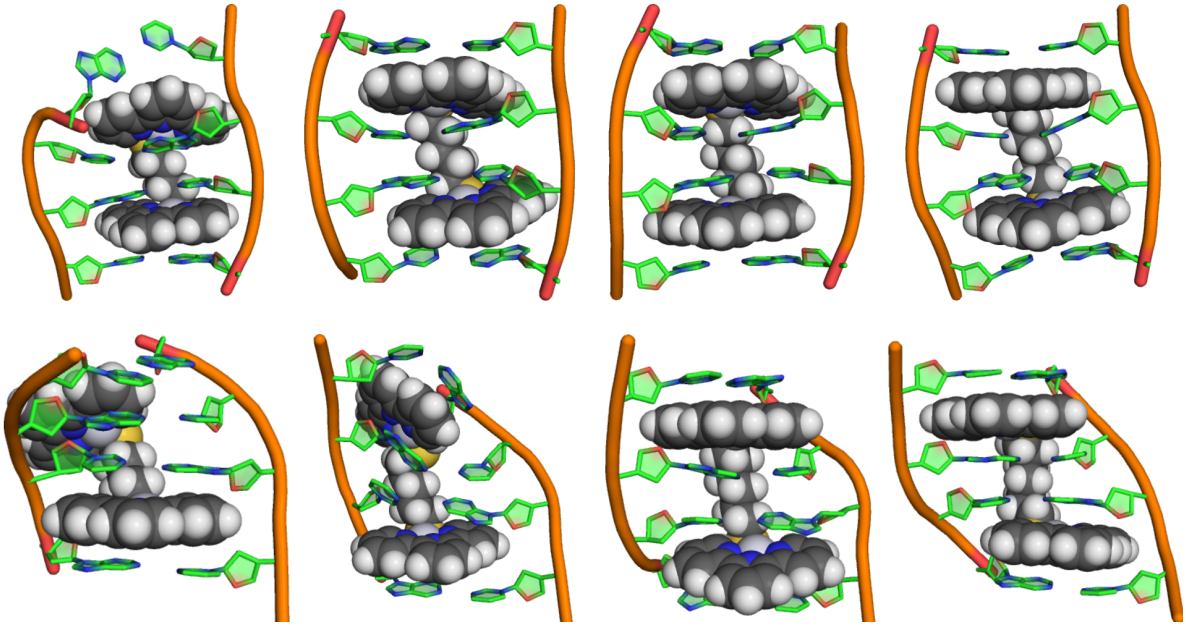


Figure 10: Renders of minimised best member frames from MD of various (terpy)Pt(II) thiolate dimer 2bps complexes in the TATA intercalation site. Top row: ***D-4*** *increasing along the row to* ***D-7****, linker in the major groove.* Bottom row: ***D-4*** *increasing along the row to* ***D-7****, linker in the minor groove.*

D-4 and **D-5** have linker lengths which cause significant structural distortions from the minor groove. In particular the TATA sequence seems to be the least favourable sequence to form a minor groove bound 2bps, the complex with **D-4** is monofunctional and the complex with **D-5** has a bound conformation with one chromophore twisted almost parallel to the DNA backbone. Also of note is that when binding from the major groove, Pt appears to locate itself above the O6 of the guanine in all guanine containing intercalation sites (CGCG, CGC, CACA, CAC), which was suggested as a possible stabilisation mechanism for major groove binding of Pt containing ligands.[12] Typical Pt-O6 interatomic distances when binding from the major groove are in the range of 3.2-3.5 Å, whereas when binding from the minor groove the O6 is inaccessible and the Pt-O6 interatomic separations are all greater than 5 Å.

3.4. Free Energies and Entropies of Binding

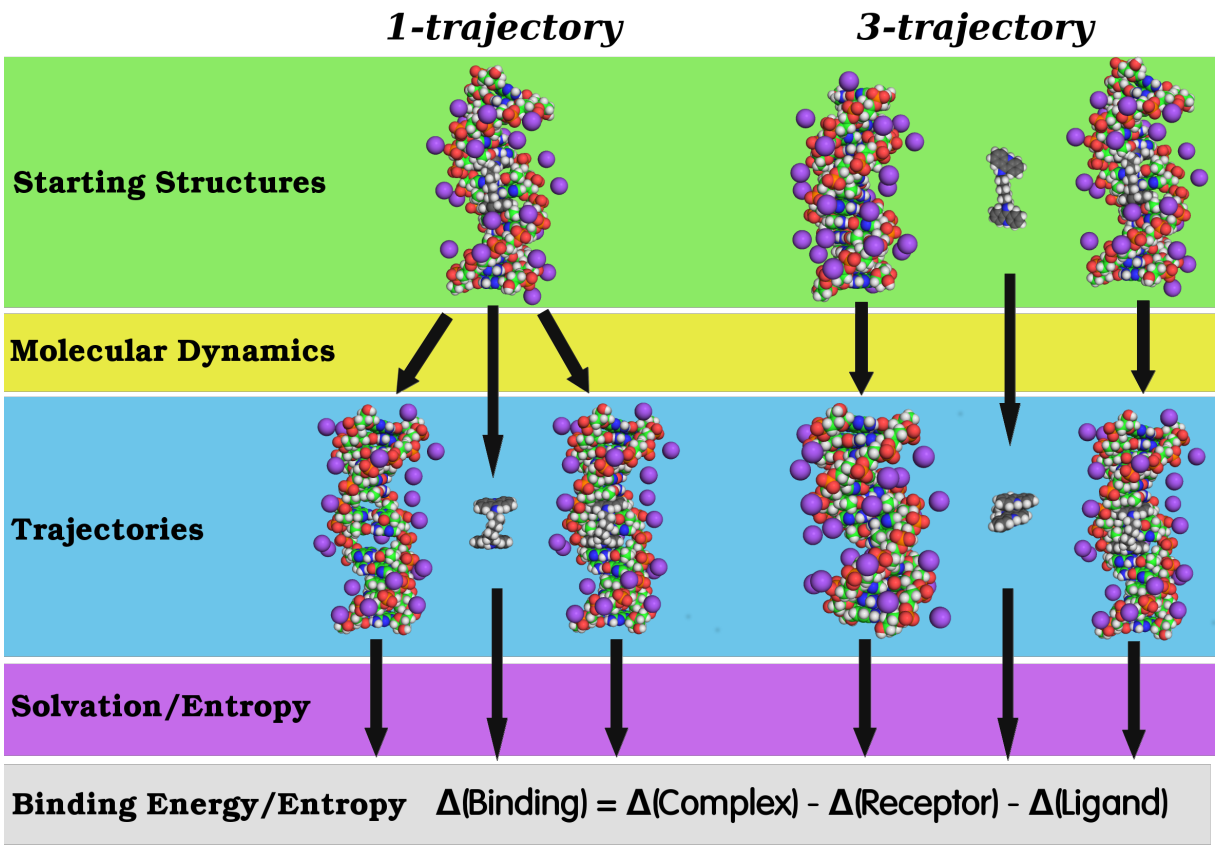


Figure 11: A flow diagram showing the difference between the one 1-trajectory (left) *and 3-trajectory* (right) *approaches in MMPBSA. In the one 1-trajectory approach the receptor, ligand and complex coordinates used in solvation and entropy calculations are taken from a single MD simulation of the complex. In contrast the 3-trajectory approach uses separate MD simulations for the individual contributions of the receptor, ligand, and complex.*

Initial attempts to acquire reliable free energy calculations *via* the default 1-trajectory approach of MMPBSA were unsuccessful because the thermodynamic cycle of intercalation is not properly represented by this single trajectory approach where the 'free' ligand and receptor contributions are taken from the coordinates of the ligand and receptor atoms in the complexed trajectory (*see Figure 11 above*). This means than in the 1-trajectory approach any structural changes upon binding are assumed to be negligible, which while somewhat defensible in the case of 'lock-and-key' binding of ligands to largely rigid proteins is an invalid assumption in the case of DNA intercalation. As the 1-trajectory approach does not properly account for the conformational rearrangements of these systems, conclusions from

the MMPBSA values, even qualitative ones, would likely be spurious.

The 3-trajectory approach should better accommodate these structural changes by taking receptor, ligand, and complex geometries from separate simulations, however this introduces the issue that the internal energy terms (*e.g* bond terms) do not cancel between the trajectories which results in a large amount of deviation in the calculated values.[74] This is seen in the large standard deviations (*circa* 30 kcal/mol) in 3-trajectory derived average Gibb's free energies of binding which masks the differences in average energies between the complexes (in the range of 10-20 kcal/mol) and hence prevents making any definitive assessment of sequence or groove selectivity of ligands. While MMPBSA values are commonly used in protein binding studies, their inability to provide reliable comparisons to experiment[75]and general poor performance[32] with intercalator systems has previously been noted.

The difference in entropy change upon binding between a 1bps and its respective 2bps was calculated *via* 3-trajectory normal-mode analysis however no discernable trend can be observed in the data, with the entropic difference between the 1bps and respective 2bps complexes fluctuating between +4 kcal/mol and -6 kcal/mol (*see Figure 12 below*).

Thus no consistent entropic determinant of binding functionality could be calculated, but this may be more due the shortcomings of using MM harmonic potentials in normal mode analysis to capture the entropies of the system accurately enough to discern subtle differences, than to entropy differences between 2bps and 1bps being negligible in reality. Full QM frequency calculations of the complexes would provide far more reliable entropy values, but would also require complete FMO geometry optimisation of the complexes and is currently prohibitively expensive to perform for systems of this size and number.

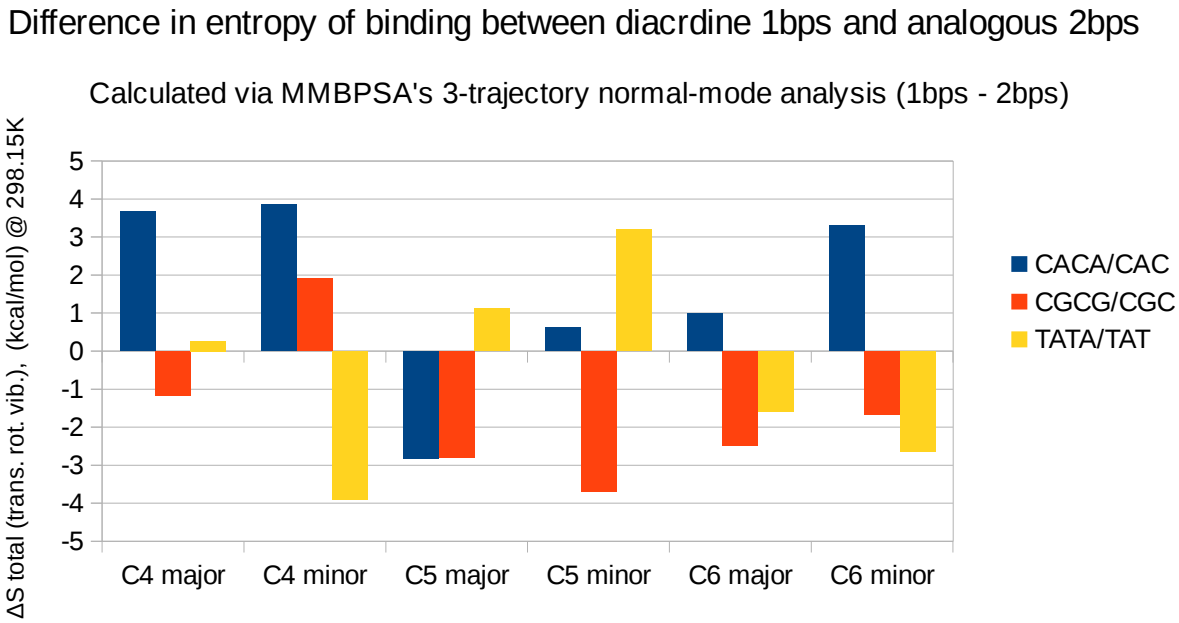


Figure 12: Differences in entropy change upon binding between each diacridine 1bps complex and its respective 2bps complex as calculated via *MMPBSA's normal-mode analysis. The classical normal-mode analysis in unable to determine any consistent trends in energy difference.*

3.5. Electronic Effects on Intercalation

3.5.1. Electrostatic Repulsion Between Chromophores

The change in electrostatic repulsion between the two positively charged chromophores upon moving from a 2bps to the respective 1bps was determined from the FMO results (*see Figure 13 below*). For the **9AA** complexes the 1bps had a higher intrinsic electrostatic repulsion between the two chromophores by an average of 9.7 kcal/mol compared to the analogous 2bps. When the solvation shielding is taken into account the total inter-chromophore repulsions are reduced to an average of 4 kcal/mol, with some variation across different complexes as the solvation energy varies according to each complex's unique solute cavity surface.

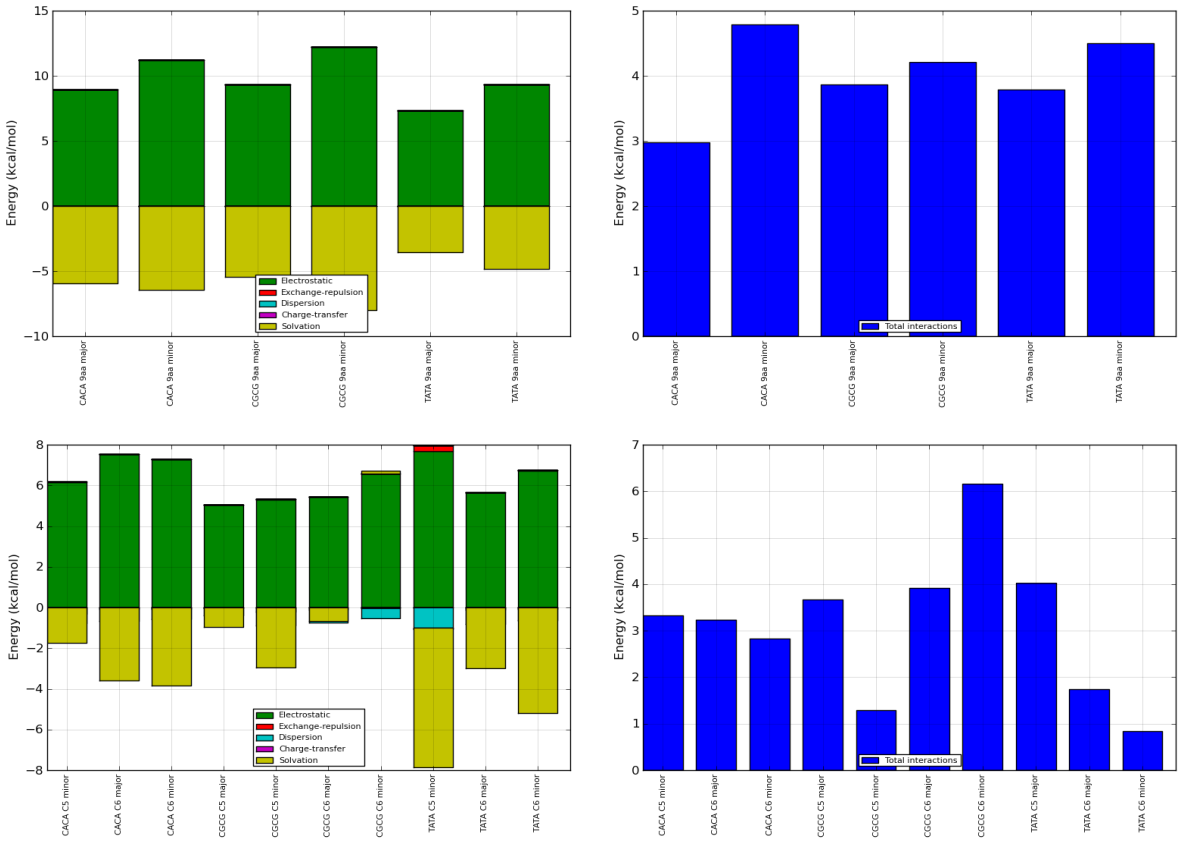


Figure 13: FMO results showing the difference *in PIEs between the two chromophores when moving from a 2bps to a 1bps.* Top: *dual* ***9AA*** *complexes.* Bottom: *diacridines (excluding those which do not form a stably intercalated complex).* Left: *PIEs separated into the various components (electrostatic, exchange-repulsion* etc.*).* Right: *total PIEs.*

3.5.2. Stacking Interactions in Unwound Base-pairs

According to the theory elaborated in reference 19 the local unwinding and subsequent increase in nucleobase overlap for the base-pair steps adjacent to the intercalation site will cause an increase in that adjacent base-pair step's stacking energy. This was investigated by comparing the total interaction energy of the base-pairs in the base-pair steps adjacent to the intercalation site with their equivalent in the unintercalated DNA sequences (*see Figure 14 below*). While there is significant variation in the FMO calculated energies, as each data point from the chart represents a single unique conformation of these systems, if unwinding

increased the adjacent base-pair steps' stabilisation significantly it should result in a shift the entire sets interaction energies to more stabilising (negative) PIEs.

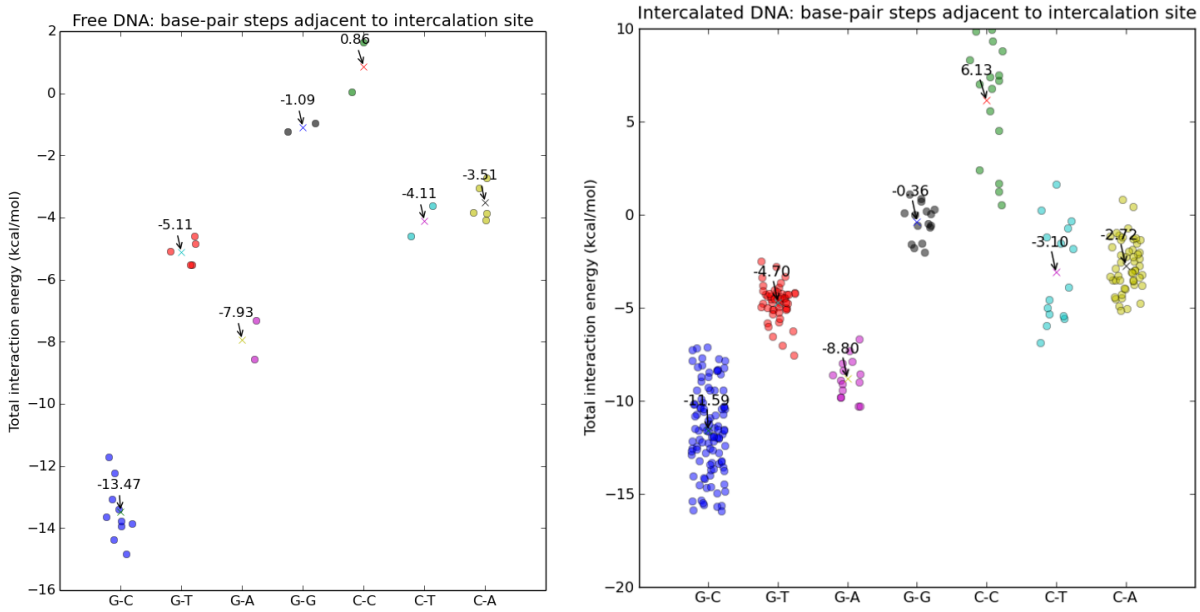


Figure 14: Total stacking energies between each base-base combination present in the base-pair steps adjacent to the intercalation site. Left: *Free DNA PIE values between bases with no intercalation.* Right: *PIE values of bases in base-pair steps adjacent to intercalation site.*

While the unintercalated DNA stacking values suffer somewhat form under sampling of some base-base combinations (*e.g.* C-C *vs.* G-C) due to the choice of intercalation sequences, the average values maintain the same ranking order as reference QM values, with the exception of the two similar values of C-T *vs.* C-A. (*see Table 6 below*).

Stacked bases	G-C	G-T	G-A	G-G	C-C	C-T	C-A
RI-SCS-MP2/6-31G* mean PIE values; free DNA	-13.47	-5.11	-7.93	-1.09	0.86	-4.11	-3.51
RI-SCS-MP2/6-31G* mean PIE values; intercalated DNA	-11.59	-4.7	-8.8	-0.36	6.13	-3.1	-2.72
Literature CBS(T) values of base-base stacking with a 36° helical twist	-10.8	-5.67	-9.14	-3.54	-1.62	-4.69	-4.96

Table 6: Base-base stacking energies taken from the FMO data of this study, and CBS(T) values from reference 28. CBS(T) is a high accuracy QM method which approximates the value of coupled-cluster calculations with single, doubles and estimated triples (CCSD(T)) excitations, extrapolated to the complete basis-set limit (CBS). This is achieved by adding the difference between CCSD(T) and MP2 values at a moderate basis-set onto the MP2/CBS value. Thus CBS(T) is calculated via: $\Delta E_{CBS}^{CCSD(T)} = \Delta E_{CBS}^{MP2} + \left(\Delta E_{moderate\,basis}^{CCDS(T)} - \Delta E_{moderate\,basis}^{MP2}\right)$

Exact quantitive agreement between the FMO and reference base-base energies should not be expected as the FMO energies are obtained using a smaller basis-set and structures directly from the minimised best member frame of each complexes trajectory, whereas the reference energies are from idealised QM optimised DNA structures with an exact 36° helical twist.

An increase in total stacking energy upon unwinding is not observed. After intercalation, the average PIE for each base-base combination remains similar, typically destabilising by *circa* 1 kcal/mol, with the exception of C-C stacking which becomes *circa* 5 kcal/mol less favourable upon unwinding. This increase in the C-C's stacking energy can be explained by an increase in bases' unfavourable electrostatic interactions upon further overlap of their repulsive functional groups.[28] This lack of change in base-base PIEs is consistent with previous *ab intio* scans of the of base stacking energy with respect to twist angle that show that while the intrinsic (vacuum) electrostatic stacking energy does show a strong dependence on twist angle this difference is nullified in solvated calculations. This is because the water's solvation energy term stabilises unfavourable stacking twists and destabilises more favourable arrangements, almost entirely cancelling any twist angle dependence.[31] Thus it is unlikely that any local unwinding caused by intercalation would significantly stabilise the adjacent base-pair step in such a way to cause neighbour exclusion.

3.6. C3NC3 and Its Hydrogen Bonding Potential

The dipropyl amine linker of the **C3NC3** ligand is the only positively charged linker in this study and as such the modelling can reveal the unique interactions of the charged linker with the intercalation site (such as hydrogen bonds) and the effect the sequence's nucleotide composition has on the binding mode of **C3NC3**.

C3NC3 with the linker in the major groove is found to form hydrogen bonds to

guanine in the CGCG and CACA intercalation sites, whereas from the minor groove guanine hydrogen acceptors are not accessible. The stabilisation provided by hydrogen bonding is evident in the clustering analysis. For example with **C3NC3** when the linker is in the minor groove of CGCG the centroid of the most populous cluster shows a high amount of disorder, whereas from the major groove the centroid shows almost no disorder due to the linker adopting a stable and rigid conformation during MD because of the clear hydrogen bond between the linker's amine and the guanine's O6 (*see Figure 15 below*).

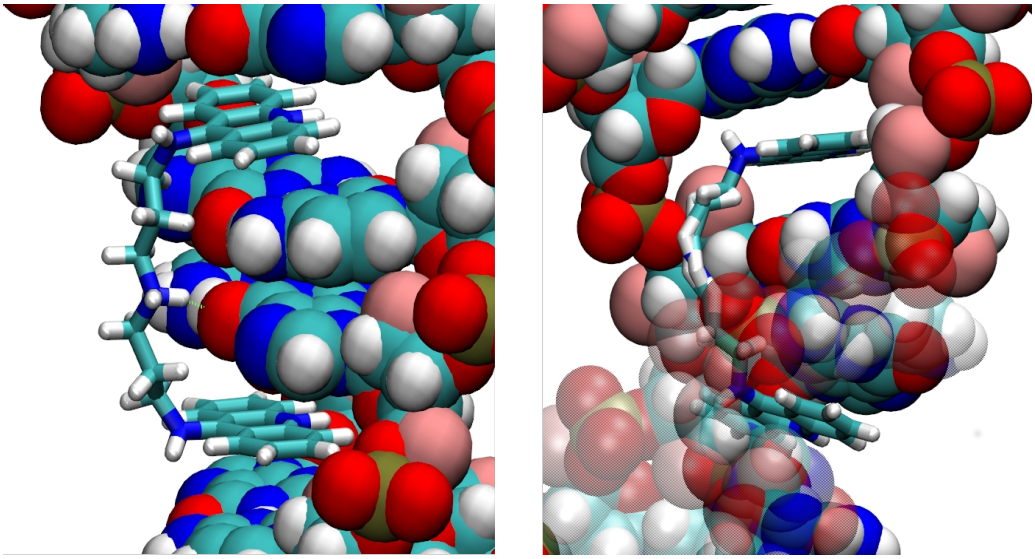


*Figure 15: Centroids of the most populous clusters of **C3NC3** with the linker in the major* (left) *and minor* (right) *grooves of CGCG. The major groove structure is made rigid by a stabilising hydrogen bond. This hydrogen bond is lacking in the minor groove, causing the linker to be highly mobile during MD and hence yielding a disordered centroid.*

The hydrogen bonds are also evident in the FMO data. The characteristic stabilising electrostatic, dispersion and charge-transfer interactions between fragments which are hydrogen bonded (total interaction *circa* -35 kcal/mol) can be seen between the linker and intercalation site guanines when the linker is in the major groove. In contrast, when the linker is in the minor groove the linker-guanine PIEs are destabilising because the linker is interacting repulsively with the positive partial charges on the guanine’s exocyclic amine group (*see Figure 16 below*).

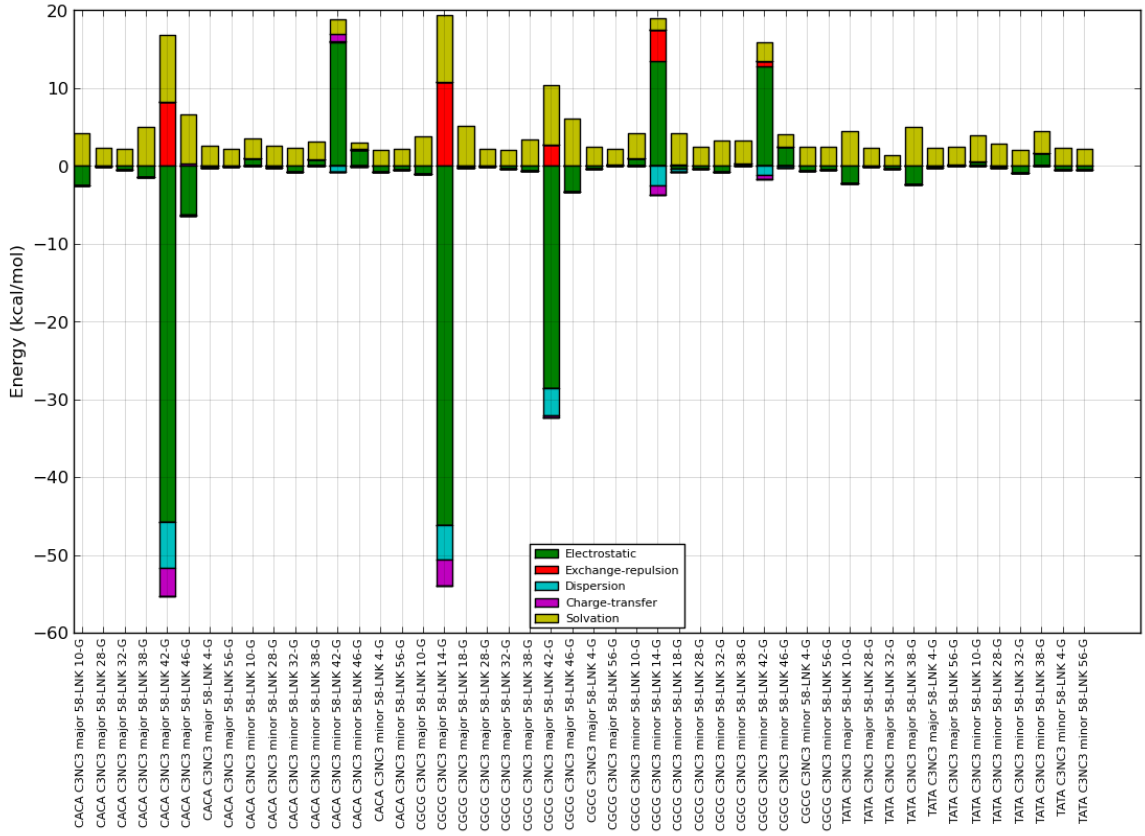


*Figure 16: PIEDA_mat.py FMO energy plot of the PIEs between the linker fragment of **C3NC3** and all guanines in the respective complexes. The different characteristics of hydrogen bonding interactions (large negative energies), electrostatic clashes (significant positive energies) and distantly separated fragments (little interaction) can be observed.*

In comparison, interactions between **C3NC3**'s linker and adenines in the DNA strand never reach above 10 kcal/mol, furthermore the positive and negative contributions typically cancel to give a total interaction of magnitude of less than 3 kcal/mol. In the case of the of **C3NC3** interacting with cytosine, the O2 in the minor groove provides a fair degree of electrostatic stabilisation to the positive linker (PIEs *circa* -20 kcal/mol), whereas in the major groove its exocyclic amine provides some inherent electrostatic repulsion (PIEs *circa* 5 kcal/mol). but this repulsion is largely cancelled by the charge shielding from the solvent. Thymine appears to provide stabilising interactions with **C3NC3** whether it is in the major or the minor groove due to the fact that it contains two ketone groups, with its O4 being accessible from the major groove and its O2 from the minor. These stabilising PIEs are typically smaller than with guanine or cytosine, being around -8 kcal/mol, with the exception of the when **C3NC3** binds

from the minor groove of TATA, whose minimised best member frame shows orientation of the linker's amine proton directly to a thymine's O2 (interatomic distance 1.8 Å) and the FMO data has a corresponding PIE of -17.9 kcal/mol between the linker and that thymine fragment.

Chapter 4: Discussion

4.1. Implications for Proposed Neighbour Exclusion Violation

It is clear from the MD data that it is possible to form 2bps with all diacridine compounds which were found experimentally to be unambiguously bifunctional (*i.e.* **C-6** to **C-8**). In addition, the geometries of each 2bps are also more in keeping with the structural constraints (*e.g* roll angle) determined by experiment than the 1bps geometries. Accordingly the binding data of methylene linked diacridines should not be interpreted as indicating they violate the neighbour exclusion principle, but in fact that they are forced to adopt measurably strained conformations because of it.

The di-(terpy)Pt(II) thiolate ligands (**D**-4 to **D-7**) were also found to form a stable 2bps for all sequences, though with more of a noticable preference for binding from the major groove over the minor groove. Roll angles were not measured for the (terpy)Pt(II) thiolate compounds so structural constraints can't be used to discriminate between the viability of the 2bps and 1bps complexes' geometries, but again the assertion that they must be forming a 1bps in violation of neighbour exclusion is not supported. That 1bps complexes could be forming cannot be ruled out by this study. However, in light of the reinterpretation of the binding of the diacridines, wide experimental support that neighbour exclusion is generally applicable, and absence of conclusive evidence that the di-(terpy)Pt(II) thiolate intercalators are in violation of the principle, it is likely that 2bps are the physically adopted structures.

4.2. Evaluation of Various Theories on Neighbour Exclusion

Firstly it should be addressed that the lengths of DNA sequences used in these studies are appropriate for representing the neighbour exclusion phenomenon. As pointed out in reference 18 there is spectroscopic evidence showing neighbour exclusion occurs in pentanucleotide strands, and as such the neighbour exclusion phenomenon cannot be explained as a global rearrangement phenomenon only present in large stretches of DNA. Hence there must be locally determining factors causing exclusion over relatively small stretches of DNA.

Attempts to explain neighbour exclusion by steric arguments are not supported by this work. 1bps geometries show very stable intercalation and are not sterically penalised by the MD's representation of bond potentials. No occlusion of the adjacent sites is observed throughout the dynamics. Additionally, geometries formed by single intercalation show no indication that untenable DNA structures would form if the DNA lattice was saturated with an intercalator at every base-pair step. The notion that mixed C2'-*endo*, C3'-*endo* puckering is required also finds no support, as the MD trajectories oscillate between a variety of sugar-puckers which is also seen in NMR solution structures of other intercalators.[76]

The average 4 kcal/mol increase in electrostatic repulsion between charged chromophores upon adjacent intercalation will encourage the formation of a 2bps over a 1bps when possible. However, in consideration of the binding data demonstrating **C-4** remains monofunctional rather than forming a geometrically accessible 1bps, it seems unlikely that this increase in electrostatic repulsion between adjacent chromophores can force monofunctionality by outweighing the electrostatic stabilisation that would be gained by intercalating inside the vast field of negative charge present in DNA. Furthermore, there is crystallographic evidence of charged intercalators stacking on the outside of intercalated

dinucleotide steps, bringing the chromophores' charges in the same proximity of each other as would occur in a 1bps.[16] With only two negative phosphate groups on the DNA strand, this should represent an ideal case for chromophore-chromophore repulsions to dominate and the fact that they do not is evidence that the electrostatic repulsion is not great enough to cause neighbour exclusion on its own. In addition, electrostatic repulsion fails to explain the occurence of neighbour exclusion in uncharged chromophores.

Unfortunately despite strong indications of the importance of both entropic and polyelectrolyte effects to neighbour exclusion it is difficult to draw conclusive evaluations on either effects from the current methodology. Entropy differences between 1bps and 2bps complexes calculated *via* harmonic normal-mode analysis return no consistent result, implying that the difference in entropies is not large enough to be preserved upon use of the assumptions inherent to classical normal-mode analysis. However noticable stiffening is observed throughout MD trajectories when adjacent intercalation is modelled, and it seems likely that this loss in vibrational entropy is an entropic dissuasion from violating neighbour exclusion, albeit one which cannot be adequately quantified under the current framework.

Similarly, theoretical assessment of polyelectrolyte effects remains difficult as proper treatment of solvation and ionic effects are notoriously difficult to model. Solvation is most often approximated in QM calculations by a single polarisable 'continuum' substance which provides the solvation energy and charge stabilisation that a real solvent would. However, effective modelling of the polyelectrolyte effect would require realistic representation of an entire condensation sheath of counter-ions and the desolvation energies of taking counter-ions into the bulk solvent. QM calculations using explicit solvent are currently unfeasibly demanding computationally and MD investigations of polyelectrolyte effects may not yield reliable data as it is generally outside the purview of the problems the force-fields were

developed for, and would involve many separate trajectories over a wide range of counter-ion concentrations.

There is also no support in the FMO data for the theory that neighbour exclusion arises not from adjacently intercalated structures being unviable, but instead because helical unwinding makes the insertion process inaccessible at adjacent sites. Considering the accuracy achieved by MP2 calculations on DNA once SCS has been applied, and the agreement with benchmark-level QM calculations, it can be concluded that intercalation has little effect on the stacking stability of the adjacent base-pair steps.

Chapter 5: Further Work and Conclusions

5.1. Further Work

Attaining reliable entropic values remains a great concern to accurate assessment of the factors involved in neighbour exclusion. Non-QM and semi-empirical approaches are found to not be high enough levels of theory to treat entropies of DNA systems, however frequency calculations are not implemented in fragment based methods and so scaling factors re-emerge as a serious limitation to the type of calculations that can be performed. Dispersion corrected DFT presents itself as the least computationally intensive of the methods which have had success in reproducing vibrational spectra of nucleotides, however tractibility remains in the realm of a few nucleobases; serious truncations of the systems studied and increases in computational time available would be required.[77]

Polyelectrolyte modelling remains a field of active development,[78] and while methods going beyond the mean field approximation of the Poisson-Boltzmann equation have been reported,[79] in general they have not been implemented in standard computational chemistry

packages and remain the domain of specialist research groups. The application of these methods has been to model systems and hence gauging their efficacy with atomistic representations of DNA strands is difficult. Thus evaluating the polyelectrolyte effect on DNA intercalation processes would likely require development and validation of a new protocol specific to these systems.

The basis-set used for FMO in this study is of modest size and it would be of interest to calculate more accurate PIEs of this systems for use in quantitive comparisons between the various interactions. Since the accuracy of MM geometries may provide more of a limit to the accuracy of the energies obtained than what would be gained by increasing the basis-set size used for single-point energy calculations, FMO optimisation of the best member frames would be prudent. However the energy gradient of MP2-FMO in polarizable continuum model solvation is not yet available and so geometry optimisations would be limited to applying either the vacuum MP2-FMO or solvated Hartree-Fock (HF-FMO/PCM) methods. Nevertheless comparison of QM and MM structures and their corresponding energetic values would provide insight into possible emergent forces in intercalation that MM models may be limited in representing.

In consideration of the unusual indication in the experimental binding data for **C-5** that there is a transition from monofunctionally to bifunctionally as the drug to DNA ratio increases, it would be interesting to carry out MD simulations with progressively higher saturations of a DNA lattice with **C-5** to see if there are structural changes to the saturated DNA which may allow a change in the functionality of binding. That the MD trajectories show **C-5** as having one chromophore partially inserted seems to indicate that relatively small changes in the structure of the DNA strand might alter the distance the linker needs to span by enough to allow fully inserted bifunctional intercalation.

Finally it hoped that work will inspire further attempts to investigate the solution dynamics of DNA-intercalator complexes, and it is hoped that additional experimental data can be acquired to provide further clarification of the mechanism(s) of neighbour exclusion.

5.2. Conclusions

These modelling results do not support the assertion that bifunctional diacridine and di-(terpy)Pt(II) thiolate dimers with shorter linker lengths are unable to span across two base-pairs and are therefore in violation of the neighbour exclusion in principle. The results of atomistic level molecular dynamics simulations instead indicate that the structures measured experimentally are not forming in violation of neighbour exclusion, but are adopting sterically strained structures because of the intercalators' adherence to the neighbour exclusion principle.

The observed monofunctionality of the short linker length dimers was also reproduced in the molecular dynamic trajectories, as one of the chromophores unintercalates during the 10 ns trajectory as a consequence of steric strain arising from the linker. In addition, the previously proposed preference of di-(terpy)Pt(II) thiolate dimers for binding from the major groove was confirmed *via* simulation, with strong overlap between the chromophore's Pt atom and the guanine’s O6 indicative of a possible stabilising interaction causing this preference. The combined molecular dynamics/fragment molecular orbital analysis of the diacridine ligand with dipropyl amine linker (**C3NC3**) revealed hydrogen bonding from the linker's amine to the guanine’s O6 when intercalating from the major groove (total inter-fragment stabilisation *circa* -35 kcal/mol), and to cytosine's O2 when intercalating from the minor groove (total inter-fragment stabilisation *circa* -20 kcal/mol).

The molecular dynamics trajectories show no support for proposals that neighbour exclusion is a steric phenomenon, neither occlusion of the adjacent site nor a requirement for

a specific sugar-pucker arrangement was observed. The polyelectrolyte and entropic effects on DNA intercalation were unable to be reproduced to a high enough level of accuracy using classical molecular dynamics methods to be evaluated. However stiffening of the DNA strand and presumed loss in vibrational entropy was observed upon adjacent intercalation. Additionally, polyelectrolyte theory is consistent with the anti-cooperativity in experimental intercalator binding curves. As such these remain viable mechanisms which may contribute to neighbour exclusion.

Fragment molecular orbital calculations determined that when the positively charged chromophores were intercalated adjacently rather than separated by one base-pair there was an increase in intrinsic electrostatic repulsion between the chromophores on the order of 10 kcal/mol which corresponds to a increase in total repulsion of *circa* 4 kcal/mol once solvation effects were taken into account. Thus electrostatic repulsion will be a contributing factor to neighbour exclusion of charged intercalators, however crystallographic evidence of charged chromophore stacking in proximity akin to that which would occur upon neighbour exclusion violation demonstrate that this repulsion cannot be attributed as the sole cause of neighbour exclusion. There was no significant increase in base-base interactions in unwound base-pair steps adjacent to intercalated sites, casting doubt on the proposal that it is the unwinding of adjacent base-pair sites upon intercalation which causes neighbour exclusion.

Thus the results of this modelling put into doubt the assertion that diacridine and di-(terpy)Pt(II) thiolate intercalators are able violate the neighbour exclusion principle, and reject many proposed mechanisms by which the principle might arise. However, further study is needed to provide a satisfactory explanation for cause(s) of neighbour exclusion and why the principle has applicability to such a wide range of intercalator systems.

References

(1) Wright, R. G.; Wakelin, L. P. G.; Fieldes, A.; Acheson, R. M.; Waring, M. J. *Biochemistry* **1980**, *19*, 5825–5836.

(2) Lerman, L. S. *J. Mol. Biol.* **1961**, *3*, 18–30.

(3) Mukherjee, A.; Sasikala, W. D. *Drug–DNA Intercalation: From Discovery to the Molecular Mechanism*; 1st ed.; Elsevier, 2013; Vol. 92, pp. 1–62.

(4) Ihmels, H.; Otto, D. *Top. Curr. Chem.* **2005**, 161–204.

(5) Gniazdowski, M.; Denny, W. A.; Nelson, S. M.; Czyz, M. *Curr. Med. Chem.* **2003**, *10*, 909–924.

(6) Leung, C.-H.; Chan, D. *Med. Res. Rev.* **2013**, *33*, 823–846.

(7) Pommier, Y.; Leo, E.; Zhang, H.; Marchand, C. *Chem. Biol.* **2010**, *17*, 421–433.

(8) Ketron, A.; Denny, W.; Graves, D.; Osheroff, N. *Biochemistry* **2012**, *51*, 1730–1739.

(9) Rescifina, A.; Zagni, C.; Varrica, M. G.; Pistarà, V.; Corsaro, A. *Eur. J. Med. Chem.* **2014**, *74*, 95–115.

(10) Wakelin, L. P. G. *Med. Res. Rev.* **1986**, *6*, 275–340.

(11) Wakelin, L. P. G.; Romanos, M.; Chen, T. K.; Glaubiger, D.; Canellakis, E. S.; Waring, M. J. *Biochemistry* **1978**, *17*, 5057–5063.

(12) McFadyen, W. D.; Wakelin, L. P. G.; Roos, I. A.; Hillcoat, B. L. *Biochem. J.* **1987**, *242*, 177–183.

(13) Atwell, G.; Stewart, G.; Leupin, W.; Denny, W. *J. Am. Chem. Soc.* **1985**, *107*, 4335–4337.

(14) McFadyen, W. D.; Wakelin, L. P. G.; Roos, I.; Hillcoat, B. L. *Biochem. J.* **1986**, *238*, 757–763.

(15) Miller, K. J.; Pycior, J. F. *Biopolymers* **1979**, *18*, 2683–2719.

(16) Wang, A.; Nathans, J.; Marel, G. Van der. *Nature* **1978**, *246*, 471.

(17) Prabhakaran, M.; Harvey, S. C. *Biopolymers* **1988**, *27*, 1239–1248.

(18) Rao, S. N.; Kollman, P. A. *Proc. Natl. Acad. Sci. U. S. A.* **1987**, *84*, 5735–5739.

(19) Williams, L.; Egli, M.; Gau, Q. In *Structure & Function Volume 1*; 1992; pp. 107–124.

(20) Friedman, R. A.; Manning, G. S. *Biopolymers* **1984**, *23*, 2671–2714.

(21) Friedman, R. A.; Shahin, M.; Zuckerbraun, S. *J. Biomol. Struct. Dyn.* **1991**, *8*, 977–988.

(22) Neidle, S.; Jenkins, T. *Methods Enzymol.* **1991**, *203*.

(23) Varvaresou, A.; Iakovou, K. *J. Mol. Model.* **2011**, *17*, 2041–2050.

(24) Wilhelm, M.; Mukherjee, A.; Bouvier, B.; Zakrzewska, K.; Hynes, J. T.; Lavery, R. *J. Am. Chem. Soc.* **2012**, *134*, 8588–8596.

(25) Sasikala, W. D.; Mukherjee, A. *Phys. Chem. Chem. Phys.* **2013**, *15*, 6446–6455.

(26) Götz, A. W.; Williamson, M. J.; Xu, D.; Poole, D.; Le Grand, S.; Walker, R. C. *J. Chem. Theory Comput.* **2012**, *8*, 1542–1555.

(27) Gordon, M. S.; Fedorov, D. G.; Pruitt, S. R.; Slipchenko, L. V. *Chem. Rev.* **2012**, *112*, 632–672.

(28) Sponer, J.; Jurecka, P.; Marchan, I.; Luque, F. J.; Orozco, M.; Hobza, P. *Chemistry* **2006**, *12*, 2854–2865.

(29) Denny, W. A.; Atwell, G. J.; Baguley, B. C.; Wakelin, L. P. G. *J. Med. Chem.* **1985**, 1568–1574.

(30) Cheatham, T. E.; Case, D. A. *Biopolymers* **2013**, *99*, 969–977.

(31) Sponer, J.; Sponer, J. E.; Mládek, A.; Jurečka, P.; Banáš, P.; Otyepka, M. *Biopolymers* **2013**, *99*, 978–988.

(32) Šponer, J.; Riley, K. E.; Hobza, P. *Phys. Chem. Chem. Phys.* **2008**, *10*, 2581–2583.

(33) Senn, H. M.; Thiel, W. *Angew. Chem. Int. Ed. Engl.* **2009**, *48*, 1198–1229.

(34) Kitaura, K.; Ikeo, E.; Asada, T. *Chem. Phys. Lett.* **1999**, 701–706.

(35) Gordon, M. S.; Mullin, J. M.; Pruitt, S. R.; Roskop, L. B.; Slipchenko, L. V.; Boatz, J. A. *J. Phys. Chem. B* **2009**, *113*, 9646–9663.

(36) Fedorov, D. G.; Ishida, T.; Uebayasi, M.; Kitaura, K. *J. Phys. Chem. A* **2007**, *111*, 2722–2732.

(37) D.A. Case, V. Babin, J.T. Berryman, R.M. Betz, Q. Cai, D.S. Cerutti, T.E. Cheatham, III, T.A. Darden, R.E. Duke, H. Gohlke, A.W. Goetz, S. Gusarov, N. Homeyer, P. Janowski, J. Kaus, I. Kolossváry, A. Kovalenko, T.S. Lee, S. LeGrand, T. Luchko, R. Luo, B. , X. W. and P. A. K. AMBER14, 2014.

(38) Salomon-Ferrer, R.; Case, D. A.; Walker, R. C. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2013**, *3*, 198–210.

(39) Price, D. J.; Brooks, C. L. *J. Chem. Phys.* **2004**, *121*, 10096–10103.

(40) Joung, I.; III, T. C. *J. Phys. Chem. B* **2009**, *113*, 13279–13290.

(41) Wang, J.; Wolf, R. *J. Comput. Chem.* **2004**, *25*, 1157–1174.

(42) Jakalian, A.; Bush, B. *J. Comput. Chem.* **2000**, *21*, 132–146.

(43) Jakalian, A.; Jack, D. B.; Bayly, C. I. *J. Comput. Chem.* **2002**, *23*, 1623–1641.

(44) E.J. Baerends, T. Ziegler, J. Autschbach, D. Bashford, A. Bérces, F.M. Bickelhaupt, C. Bo, P. M.; Boerrigter, L. Cavallo, D.P. Chong, L. Deng, R.M. Dickson, D.E. Ellis, M. van Faassen, L. Fan, T. H.; Fischer, C. Fonseca Guerra, M. Franchini, A. Ghysels, A. Giammona, S.J.A. van Gisbergen, A. W. G.; J.A. Groeneveld, O.V. Gritsenko, M. Grüning, S. Gusarov, F.E. Harris, P. van den Hoek, C.R. Jacob, H.; Jacobsen, L. Jensen, J.W. Kaminski, G. van Kessel, F. Kootstra, A. Kovalenko, M.V. Krykunov, E. van; Lenthe, D.A. McCormack, A. Michalak, M. Mitoraj, S.M. Morton, J. Neugebauer, V.P. Nicu, L.; Noodleman, V.P. Osinga, S. Patchkovskii, M. Pavanello, P.H.T. Philipsen, D. Post, C.C. Pye, W.; Ravenek, J.I. Rodríguez, P. Ros, P.R.T. Schipper, G. Schreckenbach, J.S. Seldenthuis, M. Seth, J. G.; Snijders, M. Solà, M. Swart, D. Swerhone, G. te Velde, P. Vernooijs, L. Versluis, L. Visscher, O. V.; F. Wang, T.A. Wesolowski, E.M. van Wezenbeek, G. Wiesenekker, S.K. Wolff, T.K. Woo, A. L. Y. ADF2013, 2013.

(45) Te Velde, G.; Bickelhaupt, F. M.; Baerends, E. J.; Fonseca Guerra, C.; van Gisbergen, S. J. A.; Snijders, J. G.; Ziegler, T. *J. Comput. Chem.* **2001**, *22*, 931–967.

(46) Guerra, C. F.; Snijders, J. G.; Velde, G.; Baerends, E. J. *Theor. Chem. Acc.* **1998**, *99*, 391–403.

(47) Van Lenthe, E.; Ehlers, A.; Baerends, E.-J. *J. Chem. Phys.* **1999**, *110*, 8943.

(48) Hambley, T. W. *Inorg. Chem.* **1998**, *37*, 3767–3774.

(49) Swart, M. *J. Comput. Chem.* **2001**, *22*, 79–88.

(50) Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery Jr., J. A.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, M. J.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dapprich, S.; Daniels, A. D.; Farkas, Ö.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. Gaussian 09, 2009.

(51) Ruiz, S. GA_minimum.py, 2013.

(52) Schmidt, M.; Baldridge, K. *J. Comput. Chem.* **1993**, *14*, 1347–1363.

(53) Gordon, M. S.; Schmidt, M. W. In *Theory and Applications of Computational Chemistry: the first forty years*; Dykstra, C. E.; Frenking, G.; Kim, K. S.; Scuseria, G. E., Eds.; Elsevier: Amsterdam, 2005; pp. 1167–1189.

(54) Suenaga, M. *J. Comput. Chem. Japan* **2008**, *7*, 33–54.

(55) Fedorov, D.; Kitaura, K. *J. Comput. Chem.* **2007**, *28*, 222–237.

(56) Grimme, S. *J. Comput. Chem.* **2004**, *25*, 1463–1473.

(57) Reha, D.; Kabela, M.; Ryjacek, F.; Sponer, J.; Sponer, J.; Elstner, M.; Sandor, S.; Hobza, P. *J. Am. Chem. Soc.* **2002**, *124*, 3366–3376.

(58) Weigend, F.; Häser, M.; Patzelt, H.; Ahlrichs, R. *Chem. Phys. Lett.* **1998**, *294*, 143–152.

(59) Ishikawa, T.; Kuwata, K. *Chem. Phys. Lett.* **2009**, *474*, 195–198.

(60) Antony, J.; Grimme, S. *J. Phys. Chem. A* **2007**, *111*, 4862–4868.

(61) Rafal A. Bachorz, a Florian A. Bischoff, Sebastian Höfener, Wim Klopper, Philipp Ottiger, Roman Leist, Jann A. Freyb, S. L. *Phys. Chem. Chem. Phys.* **2008**, *10*, 2581–2583.

(62) Fukuzawa, K.; Watanabe, C.; Kurisaki, I.; Taguchi, N.; Mochizuki, Y.; Nakano, T.; Tanaka, S.; Komeiji, Y. *Comput. Theor. Chem.* **2014**, *1034*, 7–16.

(63) Fedorov, D. G.; Kitaura, K. *J. Phys. Chem. A* **2012**, *116*, 704–719.

(64) Fedorov, D. G.; Nagata, T.; Kitaura, K. *Phys. Chem. Chem. Phys.* **2012**, *14*, 7562–7577.

(65) MCP+NOSeC-V-TZP = MCP relativistic (6611/64/3111/5) [inner and valence shell] from: H. Mori, K. Ueno-Noto, Y. Osanai, T. Noro, T. Fujiwara, M. Klobukowski, E. M. *Chem. Phys. Lett* **2009**, *476*, 317–322.

(66) MCP+NOSeC-V-TZP = MCP relativistic (6611/64/3111/5) [valence correlated set] from: Y. Osanai, T. Noro, E. Miyoshi, M. Sekiya, T. K. *J. Chem. Phys.* **2004**, *120*, 6408–6413.

(67) Feig, M.; Karanicolas, J.; Brooks III, C. L. MMTSB Tool Set, 2001.

(68) Hunter, J. D. *Comput. Sci. Eng.* **2007**, *9*, 90–95.

(69) Williams, K. M.; Rowan, C.; Mitchell, J. *Inorg. Chem.* **2004**, *43*, 1190–1196.

(70) Baruah, H.; Wright, M. W.; Bierbach, U. *Biochemistry* **2005**, *44*, 6059–6070.

(71) Juranic, N.; Likic, V.; Kostic, N. M.; Macurat, S. *Inorg. Chem.* **1995**, *34*, 938–944.

(72) Jennette, K. W.; Gill, T.; Sadownick, A.; Lippard, S. J. *J. Am. Chem. Soc.* **1975**, *213*, 6159–6168.

(73) Ferraro, J. R. *Low-Frequency Vibrations of Inorganic and Coordination Compounds*; Plenum Press: New York, 1971; p. 309.

(74) Sponer, J.; Spacková, N. *Methods* **2007**, *43*, 278–290.

(75) Vargiu, A. V.; Magistrato, A. *Inorg. Chem.* **2012**, *51*, 2046–2057.

(76) Serobian, A.; Thomas, D. S.; Ball, G. E.; Denny, W. A.; Wakelin, L. P. G. *Biopolymers* **2014**, *101*, 1099–1113.

(77) Fornaro, T.; Biczysko, M.; Monti, S.; Barone, V. *Phys. Chem. Chem. Phys.* **2014**, *16*, 10112–10128.

(78) Li, N. K.; Fuss, W. H.; Yingling, Y. G. *Macromol. Theory Simulations* **2014**, n/a–n/a (Early View Publication).

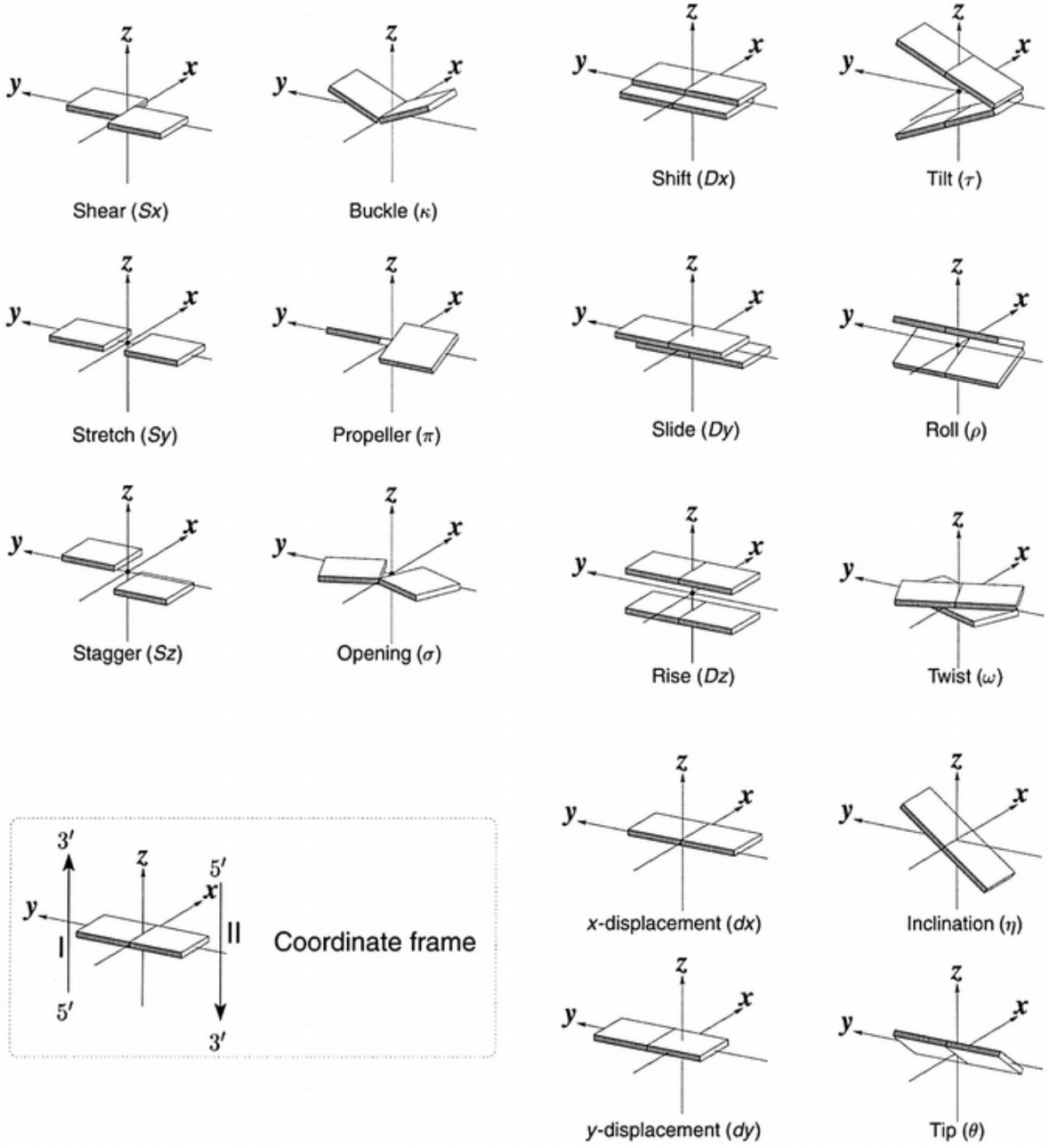
(79) Baeurle, S.; Charlot, M.; Nogovitsin, E. *Phys. Rev. E* **2007**, *75*, 011804.

(80) Fedorov, D.; Kitaura, K. *The Fragment Molecular Orbital Method: Practical Applications to Large Molecular Systems*; CRC Press: Boca Raton, Florida, 2009; p. 2888.

Supplementary

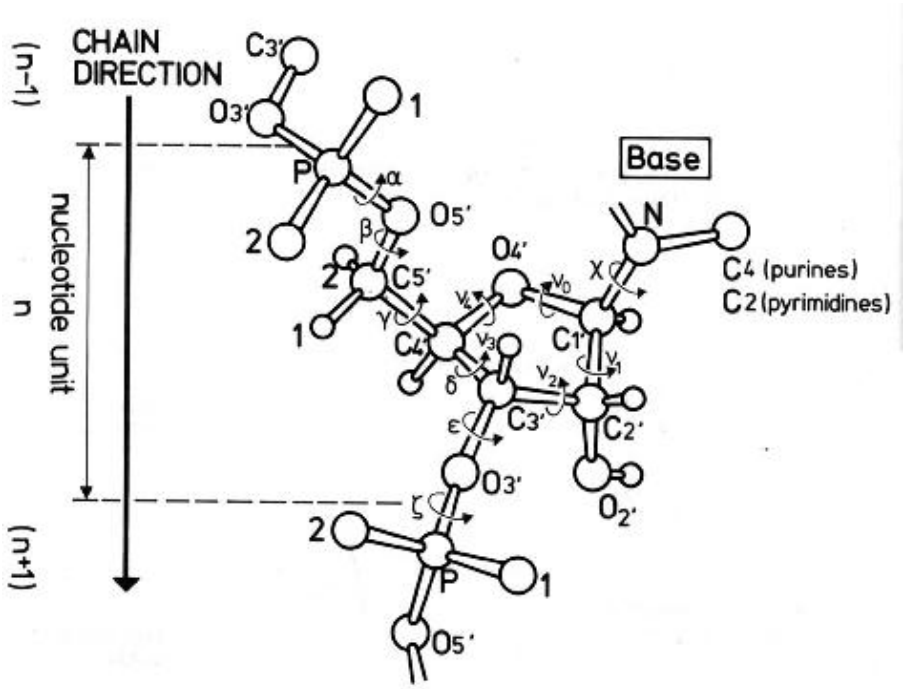
S1: Reference Diagrams

S1.1. DNA Base-pair Structural Parameters



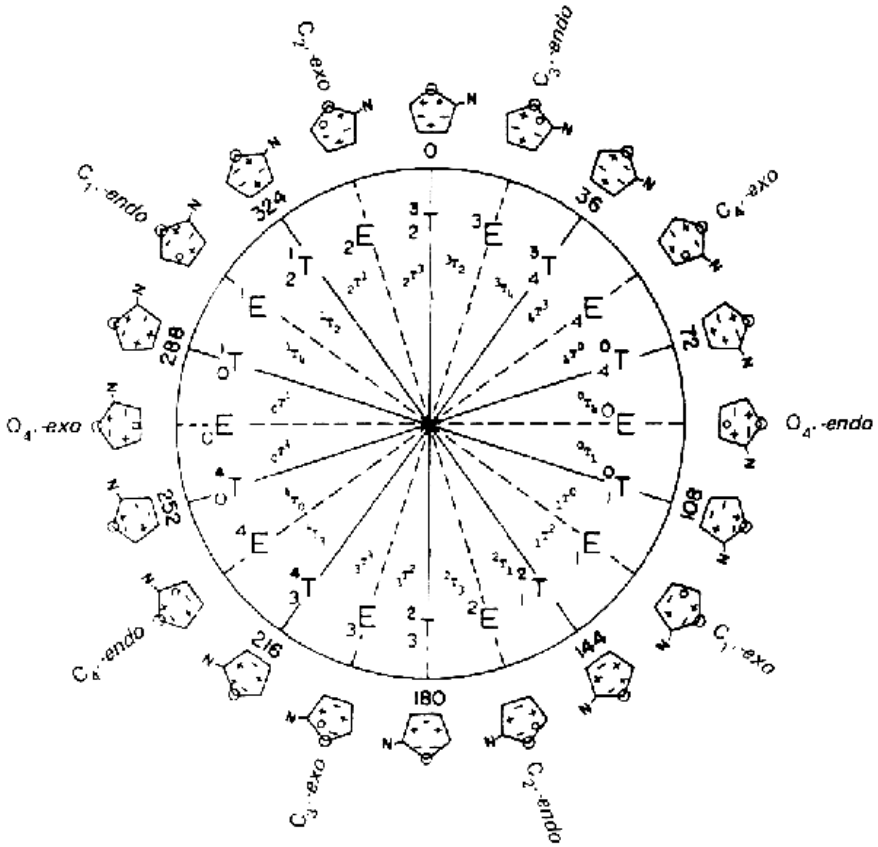
Source: Lu, X-J; Olson, W. K; *Nuc. Acids Res.*, **2003**, *31*, 17, 5108-5121

S1.2. DNA Backbone Torsion Angle Definitions



Source: Saenger, W. *Principles of Nucleic Acid Structure*, Springer, New York, **1984**

S1.3. DNA Sugar Pucker Definition



Source: Saenger, W. *Principles of Nucleic Acid Structure*, Springer, New York, **1984**

S2: Computational Details

S2.1. Resources, Documentation and Storage

Molecular Dynamics were performed on a main CentOS workstation using two GeForce GTX780 GPUs as well as on three Tesla C2050 GPUs. DFT calculations were largely performed on UNSW's Krypton CPU cluster while the majority of the FMO calculations were done on the Australian National Computational Infrastructure's (NCI) Raijin High Performance Computing cluster. Clustering and other analysis was performed on a CentOS workstation using an intel i7-4790 CPU.

Project work was documented with the UNSW's eNotebook system *via* weekly progress logs. Important data and files were stored and shared on UNSW's School of Chemistry intranet, allowing separate back-up of key information (including binary files of all trajectories) as well as easy sharing access amongst the research group. Access to this share drive can be granted upon request.

All minimised best member frames, 3DNA values, MMPBSA output, MD unit libraries, FMO output files, important spreadsheets, and relevant scripts are accessible on the supplementary DVD submitted with this thesis.

S2.2. Molecular Dynamics Details

S2.2.1. Minimisation Protocol

The system built in xLeAP is minimised for 20 cycles, the restrains easing off DNA first then then intercalator, with minimisation 19 and 20 unrestrained.

```
# general minimization
&cntrl
        imin = 1,
        maxcyc = 1000, ncyc = 5000,
        ntc = 2, tol = 0.00001,
        cut = 10.0,
        ntpr = 500,
        ntb = 1,
```

```
        ntr = 1,
&end
Restrain DNA residues
500.0
RES 1 28
END
Hold mln residues
500.0
RES 31
END
END
```

S2.2.2. Equilibration Protocol

The minimised structure is then subject to 22 lots of dynamics to equilibrate it before final production dynamics. The constraints are eased off the DNA then the intercalator, with the system allowed to equilibrate unrestrained for 12 lots of 10 ps.

```
# initial dynamics 10ps constant volume
&cntrl
        imin = 0,
        ntx = 1, irest = 0,
        ntt = 3, tempi = 293.0, temp0 = 293.0, gamma_ln = 3.0,
        ntwr = 1000, ntwx = 1000,
        ntc = 2, tol = 0.00001,
        cut = 10.0,
        ntpr = 500,
        ntb = 1,
        ntr = 1,
        nstlim = 10000,
        ig = -1,
&end
Restrain DNA residues
500.0
RES 1 28
END
Hold mln residues
500.0
RES 31
END
END
```

S2.2.3. Production Protocol

```
# constant pressure dynamics 100ps
&cntrl
        imin = 0,
        ntx = 7, irest = 1,
        ntt = 1, temp0 = 293.0, tautp = 2.0,
        ntwr = 500, ntwx = 500, ntwv = 500,
        ntc = 2, tol = 0.00001,
        cut = 10.0,
        ntpr = 500,
        ntb = 2,
        ntp = 1,
```

```
        nstlim = 100000,
&end
```

S2.3. Fragment Molecular Orbital Details

S2.3.1. Example FMO Input File With BDA Correction and Pt MCP

```
!*** FMO 4.3 (Gamess) INPUT generated by Facio 18.7.4 ***

 $CONTRL RUNTYP=ENERGY NPRINT=-5 ISPHER=1 MAXIT=50 PP=MCP $END
 $MP2 CODE=RIMP2 SCSPT=SCS NACORE=0 $END
 $RIMP2 USEDM=.FALSE. $END
 $AUXBAS CABNAM=CCD $END
 $SYSTEM MWORDS=1000 $END
 $GDDI NGROUP=1 $END
 $INTGRL NINTIC=-191000000 $END
 $SCF dirscf=.t. diis=.t. damp=.t. NPUNCH=0 $END
!*** NumCPU : 16  MemPerNode : 30000MB
 $PCM SOLVNT=WATER IEF=-10 ICOMP=2 ICAV=1 IDISP=1 IFMO=2 $END
 $PCMCAV RADII=VANDW RIN(867)=1.7 RIN(901)=1.7 $END
 $TESCAV NTSALL=240 $END
 $FMOPRP
 r0bda(1)=
 1.49646183 1.41676992 1.49072633 1.43915427 1.48821840 1.42264191
1.49630378
 ...
 [RBDA radii array]
 e0bda(1)=
  -14.8260710066   -0.2640762215   -0.0783606224   -0.0425744234
 ...
 [EBDA radii array]
 NAODIR=200
    NGRFMO(1)=1, 1, 0, 0, 0,   0, 0, 0, 0, 0
    IPIEDA=1
    NPRINT=9
    NPCMIT=2
 $END
 $FMO
      SCFTYP(1)=RHF
      MODGRD=10
      MODMUL=0
      MAXCAO=5
      MAXBND=53
      NLAYER=1
      MPLEVL(1)=2
      NFRAG=77
      ICHARG(1)=  -1,  0, -1,  0, -1,  0, -1,  0, -1,  0,
                  -1,  0, -1,  0, -1,  0, -1,  0, -1,  0,
                  -1,  0, -1,  0,  0,  0, -1,  0, -1,  0,
                  -1,  0, -1,  0, -1,  0, -1,  0, -1,  0,
                  -1,  0, -1,  0, -1,  0, -1,  0, -1,  0,
                   0,  0,  0,  1,  1,  1,  1,  1,  1,  1,
                   1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
                   1,  1,  1,  1,  1,  1,  1
      FRGNAM(1)=       Frag1,     Frag2,     Frag3,     Frag4,     Frag5,
                       Frag6,     Frag7,     Frag8,     Frag9,    Frag10,
                      Frag11,    Frag12,    Frag13,    Frag14,    Frag15,
                      Frag16,    Frag17,    Frag18,    Frag19,    Frag20,
                      Frag21,    Frag22,    Frag23,    Frag24,    Frag25,
```

```
                    Frag26,    Frag27,    Frag28,    Frag29,    Frag30,
                    Frag31,    Frag32,    Frag33,    Frag34,    Frag35,
                    Frag36,    Frag37,    Frag38,    Frag39,    Frag40,
                    Frag41,    Frag42,    Frag43,    Frag44,    Frag45,
                    Frag46,    Frag47,    Frag48,    Frag49,    Frag50,
                    Frag51,    Frag52,    Frag53,       CR1,       CR2,
                    Frag56,    Frag57,    Frag58,    Frag59,    Frag60,
                    Frag61,    Frag62,    Frag63,    Frag64,    Frag65,
                    Frag66,    Frag67,    Frag68,    Frag69,    Frag70,
                    Frag71,    Frag72,    Frag73,    Frag74,    Frag75,
                    Frag76,    Frag77
     INDAT(1)= 0
     [Atom fragment assigning array]
 $END
 $FMOHYB
 6-31G*      15   5
   1 0  -0.065034   0.288264   0.000000   0.000000   0.604412
         0.290129   0.000000   0.000000   0.319045  -0.017106
        -0.017106   0.057934   0.000000   0.000000   0.000000
   0 1  -0.065040   0.288293   0.569832   0.000000  -0.201456
         0.290147   0.300783   0.000000  -0.106342   0.049598
        -0.017106  -0.008771   0.000000  -0.027223   0.000000
   0 1  -0.065039   0.288293  -0.284916  -0.493490  -0.201455
         0.290145  -0.150392  -0.260486  -0.106340  -0.000427
         0.032923  -0.008771   0.033353   0.013612   0.023576
   0 1  -0.065039   0.288293  -0.284916   0.493490  -0.201455
         0.290145  -0.150392   0.260486  -0.106340  -0.000427
         0.032923  -0.008771  -0.033353   0.013612  -0.023576
   0 1   1.010938  -0.011975   0.000000   0.000000   0.000000
        -0.054085   0.000000   0.000000  -0.000000  -0.003174
        -0.003174  -0.003174   0.000000   0.000000   0.000000
 MINI         5   5
   1 0  -0.104883   0.308874   0.000000   0.000000   0.521806
   0 1  -0.104883   0.308874   0.491961   0.000000  -0.173934
   0 1  -0.104883   0.308876  -0.245980  -0.426050  -0.173933
   0 1  -0.104883   0.308876  -0.245980   0.426050  -0.173933
   0 1   0.988209   0.063992   0.000000   0.000000   0.000000
 $END
 $FMOBND
        -9        11  6-31G*      MINI
 ...
 [FMO bond array]
$DATA
 FMO calculation :
TGT_D7_minor_bestmember_minimised_stripped_facio_ready.pdb
 C1
 H.1-1    1
      N31 6

 O.1-1    8
      N31 6
      d 1
      1 0.800 1.0

 C.1-1    6
      N31 6
      d 1
      1 0.800 1.0
```

```
N.1-1    7
     N31 6
     d 1
     1 0.800 1.0

P.1-1   15
     N31 6
     d 1
     1 0.550 1.0

S.1-1   16
     N31 6
     d 1
     1 0.650 1.0

Na.1-1  11
     N31 6
     d 1
     1 0.175 1.0

Pt.1-1   78
     MCP READ
      s    6
        1          4353.6878000    -0.0469513
        2           139.5193500     0.2266760
        3            30.9355800    -0.6130067
        4             5.8779624     1.1648153
        5             1.2448131    -1.0271067
        6             0.5358210    -0.3296848
      s    6
        1          4353.6878000    -0.0143291
        2           139.5193500     0.0670164
        3            30.9355800    -0.1828176
        4             5.8779624     0.3859127
        5             1.2448131    -0.5457005
        6             0.5358210    -0.1831346
      s    1
        1             0.1305835     1.0000000
      s    1
        1             0.0475191     1.0000000
      s    1
        1             0.0158397     1.0000000
      p    6
        1          1713.5934000    -0.0130998
        2           349.8466800    -0.0594017
        3            42.1347060     0.2447558
        4             7.6185763    -0.5727409
        5             1.2898238     0.8234821
        6             0.4791653     0.3095040
      p    4
        1             6.7373960    -0.0209380
        2             1.4520090     0.1995040
        3             0.2509840    -0.7386610
        4             0.0985220    -0.3743490
      p    1
        1             0.0463831     1.0000000
      d    3
        1           306.2975000     0.0097280
        2            80.9556880     0.0706847
```

```
        3             9.4601578    -0.2633012
      d    1
        1             1.2659134     1.0000000
      d    1
        1             0.4624143     1.0000000
      d    1
        1             0.1566139     1.0000000
      f    5
        1           102.3983310    -0.0170710
        2            27.9995760    -0.0965490
        3             9.0445860    -0.1757830
        4             1.0296650     0.5832330
        5             0.3574190     0.5337880

 $END
 $MCP
  MCP for Pt
  Pt
  6  4 14  3 10  2  7  1  5
     18.00(5F14.7)
     1.0624653     1.1038967     0.3547309   449.1375900     3.5730819
     0.1786484
   384.5536700    15.6073700     4.1781419  5928.6796000    18.6671820
     2.8804116
  0  0  0  1  1  1
  5814.9514000  1033.0078000   247.6446000    56.0098000
425544.0700000 33380.2200000  5246.6529000  1168.4378000  2241.3567000
   222.7974200    96.9323420   174.0781500    29.3002110    12.0994690
     6.9856132     3.0045536     1.2427011     0.4891653
     0.0189936     0.1082595     0.3995376     0.6011699    -0.0050129
     0.0409057     0.0448034     0.0105335    -0.0367406    -0.0091572
     0.0000000     0.0000000     0.0000000     0.0000000
     0.0077545     0.0441989     0.1631185     0.2454386     0.0648315
    -0.5290285    -0.5794381     0.0093821    -0.0327243    -0.0081562
     0.0079250     0.0019688     0.0000000     0.0000000
     0.0037325     0.0212743     0.0785141     0.1181373     0.0371101
    -0.3028201    -0.3316750    -0.3031404     1.0573419     0.2635313
    -0.0143755    -0.0035713     0.0026151     0.0006235
     0.0019846     0.0113117     0.0417465     0.0628145     0.0111101
    -0.0906589    -0.0992975    -0.3278627     1.1435724     0.2850234
    -1.2703583    -0.3155920     0.0016123     0.0003844
   898.3178800   207.6116400    42.6051580
  6973.6794000  1372.2397000   400.7567100   138.1012200    62.0942420
    24.6659190    10.9457520     4.5044779     1.6508689     0.5868666
     0.0224692     0.1536589     0.4701039     0.4828654     0.0321246
     0.0404450    -0.0108658    -0.0096750     0.0000000     0.0000000
     0.0125901     0.0860990     0.2634111     0.2705617    -0.5125044
    -0.6452449    -0.0316673    -0.0281969     0.0075613     0.0051594
     0.0063498     0.0434240     0.1328515     0.1364579    -0.3923648
    -0.4939887     0.6818571     0.6071315     0.0143535     0.0097939
   162.6000100    25.7311000
   660.2177700   181.3076500    64.3192440    24.7263870    21.2189930
     8.4399992     3.2570605
     0.0381244     0.2345310     0.5391087     0.3700907     0.0086085
     0.0192533     0.0095097
     0.0199264     0.1225820     0.2817753     0.1934348    -0.2945949
    -0.6588769    -0.3254350
     6.9399548
   112.0603200    34.7540200    12.9107350     4.8728255     1.6855876
```

```
$END
$FMOXYZ
     1      H           20.116       14.901       -3.158
...
 [List of atom types and locations]
```

S3: Code Written for This Project

All code written by K. Rowell over the course of his honours thesis.

S3.1. PIEDA_mat.py

```
###############################################################################
# Written by Keiran Rowell for his Honours project in Chemistry at UNSW.
#   Largely written during the second half of 2014.
###############################################################################
#This version I added a few checks so free dna sequences are handled when searching
import sys
import os
import re
#gamess_files_dir='/media/21a34721-cb72-4dba-92d3-
27847c51e266/Ashley/facio_files/minimised_bestmember_pdbs/FMO_input/'
gamess_files_dir='/mnt/crashley/Ashley/FMO_calculations/RI-SCS_MP2_6-
31Gd_PCM_PIEDA/output_files/'
gamess_out_files= [f for f in os.listdir(gamess_files_dir) if f.endswith('.out')]
complexes_list = []
#-----------------------------------Functions-------------------------------#-
class Fragment(object):
    "fragments from the FMO calculations"
    def __init__(self, frag_type=None, frag_num=None, frag_charge=None, frag_mer=None,
frag_complex=None, frag_job=None, stacked_next=None, stacked_prev=None, frag_paired=None,
stacked_next_leading=None, stacked_next_lagging=None, stacked_prev_leading=None,
stacked_prev_lagging=None):
        self.frag_name = str(frag_num) + '-' + frag_type
        self.frag_type = frag_type
        self.frag_num = frag_num
        self.frag_charge = frag_charge
        self.frag_mer = frag_mer
        self.frag_complex = frag_complex
        self.frag_job = frag_job
        self.stacked_next = stacked_next
        self.stacked_prev = stacked_prev
        self.frag_paired = frag_paired
        #Leading and lagging used for the Chromos since they stack with 4 bases.
        self.stacked_next_leading = stacked_next_leading
        self.stacked_next_lagging = stacked_next_lagging
        self.stacked_prev_leading = stacked_prev_leading
        self.stacked_prev_lagging = stacked_prev_lagging

def make_frag(frag_type, frag_num, frag_charge, frag_mer, frag_complex, frag_job):
    fragment = Fragment(frag_type, frag_num, frag_charge, frag_mer, frag_complex, frag_job)
    return fragment
def get_complex_details(file):
    complex = {}
    job_name = file
    complex['job_name'] = job_name
    split_job_name = job_name.split("_")
    #assign details to complex's dict
    if split_job_name[1] == 'dual':
        complex['inter_seq'] = split_job_name[0]
        complex['ligand'] = split_job_name[2]
        complex['groove'] = split_job_name[3]
        complex['lig_amount'] = 2 #not sure if this is the right way to implement this check
    elif split_job_name[1] == 'bestmin':
        complex['inter_seq'] = split_job_name[0]
        complex['ligand'] = 'free_dna'
```

```
        complex['groove'] = None
        complex['lig_amount'] = 0
    else:
        complex['inter_seq'] = split_job_name[0]
        complex['ligand'] = split_job_name[1]
        complex['groove'] = split_job_name[2]
        complex['lig_amount'] = 1
    if complex['ligand'] == 'free_dna':
         complex['complex_name'] = complex['inter_seq'] + '_' + complex['ligand']
    else:
        complex['complex_name'] = complex['inter_seq'] + '_' + complex['ligand'] + '_' +
complex['groove']
    #very that it's a valid DNA sequence
    if not re.match("^[ACTG]+$", complex['inter_seq']):
        print("Bad complex name, make sure the first field contains only valid DNA characters
(A,T,C,G)")
        quit()
    if len(complex['inter_seq']) == 4:
        complex['mer'] = 14
    elif len(complex['inter_seq']) == 3:
        complex['mer'] = 13
    else:
        print("Sequence not 13-mer or 14-mer, I don't know how to handle this")
        quit()

    return complex
def build_leading_strand(complex):
    #The standard 5 b.p. sequences all the DNA strands have been topped and tailed with
    seq_top = "CGATG"
    seq_tail = "CATCG"
    #Append top and tail to make leading strand
    leading_strand_seq = list()
    leading_strand_seq = seq_top + complex['inter_seq'] + seq_tail
    complex['leading'] = leading_strand_seq

    return complex
def build_lagging_strand(complex):
    lagging_strand_seq = ""
    leading_strand_seq = complex['leading']

    for base in leading_strand_seq:
        if base == "A":
            lagging_strand_seq += "T"
        elif base == "C":
            lagging_strand_seq += "G"
        elif base == "T":
                    lagging_strand_seq += "A"
        elif base == "G":
            lagging_strand_seq += "C"
    complex['lagging'] = lagging_strand_seq

    return complex
def build_duplex(complex):
    reversed_lagging = complex['lagging'][::-1]
    complex['duplex_seq'] = complex['leading'] + reversed_lagging
    return complex
def determine_number_sodiums(complex):
    if complex['ligand'] == 'C3NC3':  #All other arrangements lead to two charges, this one
three.
        complex['num_sodiums'] = complex['mer'] * 2 - 5
    elif complex['ligand'] == 'free_dna':
        complex['num_sodiums'] = complex['mer'] * 2 - 2
    else:
        complex['num_sodiums'] = complex['mer'] * 2 - 4
    return complex
def determine_number_ligand_frags(complex):
    if complex['lig_amount'] == 2:
        complex['num_lig_frags'] = 2
    elif complex['ligand'] == 'free_dna':
        complex['num_lig_frags'] = 0
    else:
        complex['num_lig_frags'] = 3  #single ligands in my case are bisintercalators, broken
up into three fragments.
```

```
    return complex
def get_frag_charges_and_labels(complex):
    """Takes in the name of a file (job_name) and searches the GAMESS .out for the Fragment
statistics section. Populates a list with the fragment labels (complex['frag_labels']) and
charges (complex['frag_charges']) present in the .out file, rather than relying on searching
for the matching .inp file, or it being echoed."""
    job_name = complex['job_name']

    raw_frag_stats_list = []
    clean_frag_stats_list = []
    target_file = gamess_files_dir + job_name

    with open(target_file) as input_data:
        for line in input_data:
            if line.startswith('     I  NAME     Q NAT0 NATB  NA  NAO LAY MUL SCFTYP
NOP     MOL    CONV'): #Line just after Fragment statistics header
                break
        for line in input_data:
            if line.startswith(' Close fragment pairs, distance relative to vdW radii'):
                break
            raw_frag_stats_list.append(line.replace('=','').split()) #Remove = seperator
    input_data.close()
    clean_frag_stats_list = filter(None, raw_frag_stats_list)

    frag_labels_list = []
    frag_charges_list = []
    for i in range(len(clean_frag_stats_list)):
        frag_labels_list.append(clean_frag_stats_list[i][1])
        frag_charges_list.append(clean_frag_stats_list[i][2])
    complex['frag_labels'] = frag_labels_list
    complex['frag_charges'] = frag_charges_list
    return complex
def populate_fragment_list(complex):
    complex['fragments'] = []
    #Need to grab chromophore locations from labels list
    #My definition is CR1 is the first chromo encounter moving along the 'tip' (CGATG)
    for j in range(len(complex['frag_labels'])):
        if complex['frag_labels'][j] == 'CR1':
            CR1_index = j
        if complex['frag_labels'][j] == 'CR2':
            CR2_index = j
    #If I have forgotten to label CR1 and CR2 this will cause errors, but that will point out
my mistake.

    #Populate fragment list of complex with fragment objects, should make this a separate
function
    for i in range( (2*len(complex['duplex_seq'])) + complex['num_lig_frags'] +
complex['num_sodiums']):
        #print complex['frag_charges'][i]
        if i < (2*len(complex['duplex_seq'])):
            if i % 2 == 0:
                new_fragment = make_frag('p', i+1, complex['frag_charges'][i], complex['mer'],
complex['complex_name'], complex['job_name'])
            else:
                if complex['duplex_seq'][i/2] == "A":
                    new_fragment = make_frag('A', i+1, complex['frag_charges'][i],
complex['mer'], complex['complex_name'], complex['job_name'])
                elif complex['duplex_seq'][i/2] == "C":
                    new_fragment = make_frag('C', i+1, complex['frag_charges'][i],
complex['mer'], complex['complex_name'], complex['job_name'])
                elif complex['duplex_seq'][i/2] == "T":
                    new_fragment = make_frag('T', i+1, complex['frag_charges'][i],
complex['mer'], complex['complex_name'], complex['job_name'])
                elif complex['duplex_seq'][i/2] == "G":
                    new_fragment = make_frag('G', i+1, complex['frag_charges'][i],
complex['mer'], complex['complex_name'], complex['job_name'])
        elif  i >= (2*len(complex['duplex_seq'])) and i < ( (2*len(complex['duplex_seq'])) +
complex['num_lig_frags'] ):
            if i == CR1_index:
                new_fragment = make_frag('CR1', i+1, complex['frag_charges'][i],
complex['mer'], complex['complex_name'], complex['job_name'])
            elif i == CR2_index:
```

```
                new_fragment = make_frag('CR2', i+1, complex['frag_charges'][i],
complex['mer'], complex['complex_name'], complex['job_name'])
            else:
                new_fragment = make_frag('LNK', i+1, complex['frag_charges'][i],
complex['mer'], complex['complex_name'], complex['job_name'])
        else:
            new_fragment = make_frag('Na', i+1, complex['frag_charges'][i], complex['mer'],
complex['complex_name'], complex['job_name'])
        complex['fragments'].append(new_fragment)

    #Sanity check to make sure you've created the right number of units.
    if len(complex['fragments']) != len(complex['frag_charges']):
        print("Something went very wrong, the number of fragment objects doesn't match the
.out file's charge list!")
        quit()
    return complex
def get_stacked(complex):
    #Step through to find the chromophore fragments
    for fragment in complex['fragments']:
        if fragment.frag_type == 'CR1':
            CR1_frag = fragment
        if fragment.frag_type == 'CR2':
            CR2_frag = fragment

    if complex['ligand'] != 'free_dna':
        if complex['mer'] == 14 and complex['ligand']:
            for i in range(1,56,2):
                #First define all the end fragment sections
                if i == 1:
                    complex['fragments'][i].stacked_prev = None
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 27:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = None
                elif i == 29:
                    complex['fragments'][i].stacked_prev = None
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 55:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = None
                #Then rules for the bases stacked with the chromophores
                elif i == 11:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = CR1_frag
                elif i == 13:
                    complex['fragments'][i].stacked_prev = CR1_frag
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 15:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = CR2_frag
                elif i == 17:
                    complex['fragments'][i].stacked_prev = CR2_frag
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 39:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = CR2_frag
                elif i == 41:
                    complex['fragments'][i].stacked_prev = CR2_frag
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 43:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = CR1_frag
                elif i == 45:
                    complex['fragments'][i].stacked_prev = CR1_frag
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                #Finally the 'standard' inter-strand stacks
                else:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
            #Deal with the 4 stacking of chromophores by defining major and minor strand
            CR1_frag.stacked_next_leading = complex['fragments'][13]
            CR1_frag.stacked_next_lagging = complex['fragments'][45]
            CR1_frag.stacked_prev_leading = complex['fragments'][11]
```

```
            CR1_frag.stacked_prev_lagging = complex['fragments'][43]
            CR2_frag.stacked_next_leading = complex['fragments'][17]
            CR2_frag.stacked_next_lagging = complex['fragments'][41]
            CR2_frag.stacked_prev_leading = complex['fragments'][15]
            CR2_frag.stacked_prev_lagging = complex['fragments'][39]

        elif complex['mer'] == 13:
            for i in range(1,52,2):
                #First define all the end fragment sections
                if i == 1:
                    complex['fragments'][i].stacked_prev = None
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 25:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = None
                elif i == 27:
                    complex['fragments'][i].stacked_prev = None
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 51:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = None
                #Then rules for the bases stacked with the chromophores
                elif i == 11:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = CR1_frag
                elif i == 13:
                    complex['fragments'][i].stacked_prev = CR1_frag
                    complex['fragments'][i].stacked_next = CR2_frag
                elif i == 15:
                    complex['fragments'][i].stacked_prev = CR2_frag
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                elif i == 37:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = CR2_frag
                elif i == 39:
                    complex['fragments'][i].stacked_prev = CR2_frag
                    complex['fragments'][i].stacked_next = CR1_frag
                elif i == 41:
                    complex['fragments'][i].stacked_prev = CR1_frag
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
                #Finally the 'standard' inter-strand stacks
                else:
                    complex['fragments'][i].stacked_prev = complex['fragments'][i-2]
                    complex['fragments'][i].stacked_next = complex['fragments'][i+2]
            #Deal with the 4 stacking of chromophores by defining major and minor strand
            CR1_frag.stacked_next_leading = complex['fragments'][13]
            CR1_frag.stacked_next_lagging = complex['fragments'][41]
            CR1_frag.stacked_prev_leading = complex['fragments'][11]
            CR1_frag.stacked_prev_lagging = complex['fragments'][39]
            CR2_frag.stacked_next_leading = complex['fragments'][15]
            CR2_frag.stacked_next_lagging = complex['fragments'][39]
            CR2_frag.stacked_prev_leading = complex['fragments'][13]
            CR2_frag.stacked_prev_lagging = complex['fragments'][37]
        else:
            print "%s is not a 14- or 13-mer, can't deal" % complex['complex_name']

    return complex

def get_paired(complex):
    #Get the fragment which would correspond to the Watson-Crick paired base
    if complex['mer'] == 14:
        for i in range(1,56,2):
            complex['fragments'][i].frag_paired = complex['fragments'][56-i]
    elif complex['mer'] == 13:
        for i in range(1,52,2):
            complex['fragments'][i].frag_paired = complex['fragments'][52-i]
    else:
       print "%s is not a 14- or 13-mer, can't deal" % complex['complex_name']
    return complex
def get_twobody_matrix(complex):
    """Takes in the name of a file (job_name) and searches the GAMESS .out for the final 'Two-
body FMO properties' section. Returns a 2D array (list) of the interaction energies."""
    job_name = complex['job_name']
```

```
    twobody_matrix = []
    twobody_header_count = 0
    target_file = gamess_files_dir + job_name
    with open(target_file) as input_data:
        for line in input_data:
            if line.startswith('    I    J DL  Z    R   Q(I->J)  EIJ-EI-EJ dDIJ*VIJ    total
Ees      Eex    Ect+mix    Edisp    Gsol'):
                twobody_header_count += 1
                if twobody_header_count > 1 :      #Hacky currenly, would like to do reverse
search from bottom of file but difficult with incrementers
                    input_data.next()  #Need to remove the ---- header, just step silently
along one more iteration
                    break
        for line in input_data:
            if line.startswith('\n'):     #Continue reading until the newline space at the end
of the matrix
                break
            twobody_matrix.append(line.split())   #Concise way to add all these lines to the
matrix
    complex['matrix'] = twobody_matrix

    return complex
def get_IFIE(frag_a, frag_b, energy_request, complexes_list):
    """ Given two fragments follwed by an arguement of one or more energy types (Ees+Eex,
Etot, etc.),
        will return a list of the interaction energy between the two in the format:
        [COMPLEX_NAME, FRAG_A, FRAG_B, ENERGYTYPES, ENGVAL1, ENGVAL2, ..., ENGVALN]     """
    #Sanity check that calling two fragments in the same complex
    if frag_a.frag_job != frag_b.frag_job:
        print("Can't get IFIE on these fragments %s, %s: not from the same job!")  %
(frag_a.frag_name, frag_b.frag_name)#Should probably be an exception
        return None

    #Split up requested energies so that the matrix can be searched.
    queried_energies = energy_request.split('+')
    energy_positions_dict = {'Etot':8, 'Ees':9, 'Eex':10, 'Ect':11, 'Edisp':12, 'Gsol':13}
    energy_pos_vals = []
    for energy in queried_energies:
        energy_pos_vals.append(energy_positions_dict[energy])
    IFIE_line = []
    for complex in complexes_list:
        if complex['job_name'] == frag_a.frag_job:
            target_complex = complex
    target_complex_matrix = target_complex['matrix']
    for line in target_complex_matrix:
        if (eval(line[0]) == frag_a.frag_num and eval(line[1]) == frag_b.frag_num) or
(eval(line[0]) == frag_b.frag_num and eval(line[1]) == frag_a.frag_num):  #I think I need to
clean this up by looking at the ordering
            IFIE_line = [frag_a.frag_complex, frag_a.frag_name, frag_b.frag_name,
energy_request]
            for position in energy_pos_vals:
                IFIE_line.append(line[position])
    return IFIE_line
#Should add a toggle hear to give full frag details or not
def print_IFIE(frag_a, frag_b, energy_request, complexes_list):
    IFIE_line = get_IFIE(frag_a, frag_b, energy_request, complexes_list)
    print str(IFIE_line).translate(None,"'").strip("[").strip("]") #remove the list
demarcations for easier reading/parsing
#---------------------------------------------
Main-----------------------------------------------#
#I have even more separating out into functions to do.
for file in gamess_out_files:
    complex = get_complex_details(file)
#   print("Adding %s ...") % complex['job_name']
    complex = build_leading_strand(complex)
    complex = build_lagging_strand(complex)
    complex = build_duplex(complex)
    complex = determine_number_sodiums(complex)
    complex = determine_number_ligand_frags(complex)
    complex = get_frag_charges_and_labels(complex)
#   print complex  #before you get the horrendous amount of numbers from the matrix
    complex = populate_fragment_list(complex)
    complex = get_stacked(complex)
```

```
    complex = get_paired(complex)
    complex = get_twobody_matrix(complex)
    complexes_list.append(complex)

#print('\n')
#print("The complexes loaded in are:")
#for complex in complexes_list:
#    print complex['job_name']
#----------------------------------------Searching
logic----------------------------------------#
#This way I can just write seperate pieces of logic and just supply them as an argument
logic_code = sys.stdin.read()
exec logic_code
```

S3.1.1. PIEDA_mat.py Usage Examples

These short rules should be supplied to PIEDA_mat.py as the 1st argument, and are run *via* 'exec' in PIEDA_mat.py

Find C3NC3 linker-guanine interactions:

```
for complex in complexes_list:
     if complex['ligand'] == 'C3NC3':
        for fragment_a in complex['fragments']:
            if fragment_a.frag_type == 'LNK':
                frag_a = fragment_a
                for fragment_b in complex['fragments']:
                    if fragment_b.frag_type == 'G':
                        frag_b = fragment_b
                        print_IFIE(frag_a, frag_b,
'Ees+Eex+Edisp+Ect+Gsol', complexes_list)
```

Find base-base stacking interactions in base-pair steps adjacent to chromophores:

```
for complex in complexes_list:
    for fragment in complex['fragments']:
        if fragment.frag_type == 'CR1':
            #Get the two adjacent and on the outside of CR1
            print_IFIE(fragment.stacked_prev_leading,
fragment.stacked_prev_leading.stacked_prev, 'Etot', complexes_list)
            print_IFIE(fragment.stacked_next_lagging.stacked_next,
fragment.stacked_next_lagging, 'Etot', complexes_list)
        if fragment.frag_type == 'CR2':
            print_IFIE(fragment.stacked_next_leading,
fragment.stacked_next_leading.stacked_next, 'Etot', complexes_list)
            print_IFIE(fragment.stacked_prev_lagging.stacked_prev,
fragment.stacked_prev_lagging, 'Etot', complexes_list)
```

Find 9aa chromophore-chromophore interactions:

```
for complex in complexes_list:
    if complex['ligand'] == '9aa':
        for fragment_a in complex['fragments']:
```

```
            if fragment_a.frag_type == 'CR1':
                frag_a = fragment_a
                for fragment_b in complex['fragments']:
                    if fragment_b.frag_type == 'CR2':
                        frag_b = fragment_b
                        print_IFIE(frag_a, frag_b,
'Ees+Eex+Edisp+Ect+Gsol', complexes_list)
```

S3.2. PIEDA_plot.py

```
#Written by Keiran Rowell (z3374843) for use with PIEDA_mat.py for analysis of FMO data.
import os
import sys
import matplotlib.pyplot as plt
import numpy as np

interactions_list = []

energies_list = sys.argv[1]

with open(energies_list) as input_data:
    for line in input_data:
        interactions_list.append(line.strip('\n').split(', '))

#Get the types of energy reported in the first line
#assuming from PIEDA_mat and all report same energy types
raw_energy_types = interactions_list[0][3]
energy_types = raw_energy_types.split('+')

print energy_types

Etot_index, Ees_index, Eex_index, Edisp_index, Ect_index, Gsol_index = (None,)*6

#Get whatever the ordering is in this list
for i in range(len(energy_types)):
    if energy_types[i] == "Etot":
        Etot_index = i+4
    if energy_types[i] == "Ees":
        Ees_index = i+4
    if energy_types[i] == "Eex":
        Eex_index = i+4
    if energy_types[i] == "Edisp":
        Edisp_index = i+4
    if energy_types[i] == "Ect":
        Ect_index = i+4
    if energy_types[i] == "Gsol":
        Gsol_index = i+4

complexes = []
if Etot_index is not None:
    Etot_engs_list = []
if Ees_index is not None:
    Ees_engs_list = []
if Eex_index is not None:
    Eex_engs_list = []
if Edisp_index is not None:
    Edisp_engs_list = []
if Ect_index is not None:
    Ect_engs_list = []
if Gsol_index is not None:
    Gsol_engs_list = []

#Negative values so bars don't cancel
if Etot_index is not None:
    Etot_engs_neg_list = []
if Ees_index is not None:
    Ees_engs_neg_list = []
if Eex_index is not None:
    Eex_engs_neg_list = []
if Edisp_index is not None:
    Edisp_engs_neg_list = []
```

```
if Ect_index is not None:
    Ect_engs_neg_list = []
if Gsol_index is not None:
    Gsol_engs_neg_list = []

for interaction in interactions_list:

#Terse printing which doesn't give fragment numbers and types
    #complexes.append( interaction[0].split('_')[0] + ' ' + interaction[0].split('_')[1] + ' '
+ interaction[0].split('_')[-1])
#Setting needed for 1bps 1bps differnce
    #complexes.append( interaction[0].split('_')[0] + ' ' + interaction[0].split('_')[2] + ' '
+ interaction[0].split('_')[-1])
#Verbose printing which does give all the details
    complexes.append( interaction[0].split('_')[0] + ' ' + interaction[0].split('_')[1] + ' '
+ interaction[0].split('_')[2] + ' ' + interaction[1] + ' ' + interaction[2])
    if Etot_index is not None:
        if float(interaction[Etot_index]) > 0:
            Etot_engs_list.append(float(interaction[Etot_index]))
            Etot_engs_neg_list.append(0)
        else:
            Etot_engs_list.append(0)
            Etot_engs_neg_list.append(float(interaction[Etot_index]))
    if Ees_index is not None:
        if float(interaction[Ees_index]) > 0:
            Ees_engs_list.append(float(interaction[Ees_index]))
            Ees_engs_neg_list.append(0)
        else:
            Ees_engs_list.append(0)
            Ees_engs_neg_list.append(float(interaction[Ees_index]))
    if Eex_index is not None:
        if float(interaction[Eex_index]) > 0:
            Eex_engs_list.append(float(interaction[Eex_index]))
            Eex_engs_neg_list.append(0)
        else:
            Eex_engs_list.append(0)
            Eex_engs_neg_list.append(float(interaction[Eex_index]))
    if Edisp_index is not None:
        if float(interaction[Edisp_index]) > 0:
            Edisp_engs_list.append(float(interaction[Edisp_index]))
            Edisp_engs_neg_list.append(0)
        else:
            Edisp_engs_list.append(0)
            Edisp_engs_neg_list.append(float(interaction[Edisp_index]))
    if Ect_index is not None:
        if float(interaction[Ect_index]) > 0:
            Ect_engs_list.append(float(interaction[Ect_index]))
            Ect_engs_neg_list.append(0)
        else:
            Ect_engs_list.append(0)
            Ect_engs_neg_list.append(float(interaction[Ect_index]))
    if Gsol_index is not None:
        if float(interaction[Gsol_index]) > 0:
            Gsol_engs_list.append(float(interaction[Gsol_index]))
            Gsol_engs_neg_list.append(0)
        else:
            Gsol_engs_list.append(0)
            Gsol_engs_neg_list.append(float(interaction[Gsol_index]))

#Convert lists to arrays because that's how matplotlib likes it and allows stacked columns
if Etot_index is not None:
    Etot_engs = np.array( Etot_engs_list )
if Ees_index is not None:
    Ees_engs = np.array( Ees_engs_list )
if Eex_index is not None:
    Eex_engs = np.array( Eex_engs_list )
if Edisp_index is not None:
    Edisp_engs = np.array( Edisp_engs_list )
if Ect_index is not None:
    Ect_engs = np.array( Ect_engs_list )
if Gsol_index is not None:
    Gsol_engs = np.array( Gsol_engs_list )
#And their negative counter-parts
```

```
if Etot_index is not None:
    Etot_neg_engs = np.array( Etot_engs_neg_list )
if Ees_index is not None:
    Ees_neg_engs = np.array( Ees_engs_neg_list )
if Eex_index is not None:
    Eex_neg_engs = np.array( Eex_engs_neg_list )
if Edisp_index is not None:
    Edisp_neg_engs = np.array( Edisp_engs_neg_list )
if Ect_index is not None:
    Ect_neg_engs = np.array( Ect_engs_neg_list )
if Gsol_index is not None:
    Gsol_neg_engs = np.array( Gsol_engs_neg_list )

x_pos = np.arange(len(complexes))

#Aesthetic parameters
bar_width = 0.75
opacity = 1.0
pos_offset = 0
neg_offset = 0

#Creating the stacked bars
if Etot_index is not None:
    pEtot = plt.bar(x_pos, Etot_engs, bar_width, color='b', alpha=opacity, label='Total
interactions')
    pos_offset += Etot_engs
if Ees_index is not None:
    pEes = plt.bar(x_pos, Ees_engs, bar_width, color='g', alpha=opacity,
label='Electrostatic', bottom=pos_offset)
    pos_offset += Ees_engs
if Eex_index is not None:
    pEex = plt.bar(x_pos, Eex_engs, bar_width, color='r', alpha=opacity, label='Exchange-
repulsion', bottom=pos_offset)
    pos_offset += Eex_engs
if Edisp_index is not None:
    pEdisp = plt.bar(x_pos, Edisp_engs, bar_width, color='c', alpha=opacity,
label='Dispersion', bottom=pos_offset )
    pos_offset += Edisp_engs
if Ect_index is not None:
    pEct = plt.bar(x_pos, Ect_engs, bar_width, color='m', alpha=opacity, label='Charge-
transfer', bottom=pos_offset )
    pos_offset += Ect_engs
if Gsol_index is not None:
    pGsol = plt.bar(x_pos, Gsol_engs, bar_width, color='y', alpha=opacity, label='Solvation',
bottom=pos_offset )

#And their negative counterparts
if Etot_index is not None:
    pEtot = plt.bar(x_pos, Etot_neg_engs, bar_width, color='b', alpha=opacity)
    neg_offset += Etot_neg_engs
if Ees_index is not None:
    pEes = plt.bar(x_pos, Ees_neg_engs, bar_width, color='g', alpha=opacity,
bottom=neg_offset)
    neg_offset += Ees_neg_engs
if Eex_index is not None:
    pEex = plt.bar(x_pos, Eex_neg_engs, bar_width, color='r', alpha=opacity,
bottom=neg_offset)
    neg_offset += Eex_neg_engs
if Edisp_index is not None:
    pEdisp = plt.bar(x_pos, Edisp_neg_engs, bar_width, color='c', alpha=opacity,
bottom=neg_offset )
    neg_offset += Edisp_neg_engs
if Ect_index is not None:
    pEct = plt.bar(x_pos, Ect_neg_engs, bar_width, color='m', alpha=opacity, bottom=neg_offset
)
    neg_offset += Ect_neg_engs
if Gsol_index is not None:
    pGsol = plt.bar(x_pos, Gsol_neg_engs, bar_width, color='y', alpha=opacity,
bottom=neg_offset )

plt.xticks(x_pos+(bar_width/2), complexes, fontsize=8, rotation='vertical')
plt.ylabel('Energy (kcal/mol)')
plt.legend(loc=8,prop={'size':8})
```

```
plt.grid(color='k', linestyle='-', linewidth=0.10)

plt.show()
```

S3.3. generate_clusters_3A.sh

```
#!/bin/bash

#Automating the clustering procedure as outline by Ross Walker
#(http://ambermd.org/tutorials/basic/tutorial3/section6.htm)

#Setting up variables:
#3 works best empirically for my system.
RADIUS_VALUE=3

#truncate full path to current directory
CONTAINING_DIR=${PWD##*/}
#usually the .prmtop and dir follow same naming convertion. Alter if not,
NAME=$CONTAINING_DIR
echo "Clustering of trajectory of $NAME"
#Alter to relevant directory with all cluster scripts. Make sure MMTSB is accesible
SCRIPTS_DIR=/home/keiran/scripts
echo "Using scripts from $SCRIPTS_DIR"

#Perform job:
#gunzip required .prmtop files:
if [ -f "${NAME}_wat.prmtop.gz" ]
then
  gunzip ${NAME}_wat.prmtop.gz
fi
if [ -f "${NAME}_vac.prmtop.gz" ]
then
   gunzip ${NAME}_vac.prmtop.gz
fi

#Check for .prmtop
if [ ! -f "${NAME}_wat.prmtop" ]
then
  echo "${NAME}_wat.prmtop does not exist or is misnamed"
  exit
fi
if [ ! -f "${NAME}_vac.prmtop" ]
then
  echo "${NAME}_vac.prmtop does not exist or is misnamed"
  exit
fi

#Create the binops file to work with later.
#Should add exit status if no md23.mdrcrd(.gz)
if [ -f "md23.mdcrd.gz" ]
then
   cpptraj ${NAME}_wat.prmtop < ${SCRIPTS_DIR}/mdcrdgz_to_binpos.ptraj > mdcrd_to_binpos.out
elif [ -f "md23.mdcrd" ]
then
   cpptraj ${NAME}_wat.prmtop < ${SCRIPTS_DIR}/mdcrd_to_binpos.ptraj > mdcrd_to_binpos.out
else
   echo "No md23.mdcrd file! Aborting."
   exit
fi

#make directories to work in.
mkdir clustering
cd clustering
mkdir PDBfit

#generate PDBs for each frame (20,000 for 10ns)
/usr/local/amber12/bin/ptraj ../${NAME}_vac.prmtop < ${SCRIPTS_DIR}/extract_pdb.ptraj

#sane numbering, adding leading 0s
cd ./PDBfit
${SCRIPTS_DIR}/fix_numbering_pdb.csh
rm complex_clust.pdb   #An un-numbered pdb causes segfault
```

```
#Standard k-means clustering
rm ../clustfils
ls -1 . > ../clustfils
kclust -mode rmsd -centroid -cdist -heavy -lsqfit -radius ${RADIUS_VALUE} -maxerr 1
-iterate ../clustfils > ../Centroids_3A

#extract centroids
cd ..
awk -f ${SCRIPTS_DIR}/extract_centroids.awk Centroids_3A | tee Centroids_3A.stats
```

S3.4. minimise_bestmember.sh

```
#!/bin/bash

#WIP. This script will take a frame from clustering analysis and submitting to a brief MM
minimisation (steepest descent) in order remove transient unfavourable interactions.
#Alongside minimising the bestmember it also handles: stripping solvation, adding counter-ions
and converting naming scheme to ready it for facio

cd ./clustering
pop_clust=$(sort -n -r -k3 Centroids_3A.stats | head -n 1 | awk '{print $1}')
echo "The most populous cluster is $pop_clust"
abs_frame=$(sort -n -k2 centroid0${pop_clust}.member.dat | head -n 1 | awk '{print $1}')
echo "The besmember frame is at frame number $abs_frame"

#200 frames be mdcrd and the first 22 are equilibriation
div_num=$(($abs_frame/200))
crd_file_num=$(($div_num + 23))
echo "It is in the original mdcrd file number $crd_file_num"
mod_num=$(($abs_frame%200))
frame_num=$mod_num
echo "In that mdcrd file it is frame $mod_num"
echo "Extracting frame..."
cd ..

SCRIPTS_DIR=/home/keiran/scripts
CONTAINING_DIR=${PWD##*/}
NAME=$CONTAINING_DIR

sed -e s/CRDNUM/"$crd_file_num"/g -e s/FRAMENUM/"$frame_num"/g -e s/NAME/"$NAME"/g  < $
{SCRIPTS_DIR}/extract_frame_from_clust_template.trajin > $
{SCRIPTS_DIR}/extract_frame_from_clust_edited.trajin

#Check whether gziped or not. Under construction
if [[ ! -f md${crd_file_num}.mdcrd.gz ]]
then
  sed -i s/.gz//g ${SCRIPTS_DIR}/extract_frame_from_clust_edited.trajin
  if [[ ! -f md${crd_file_num}.mdcrd ]]
  then
     echo "The coordinate file md${crd_file_num}.mdcrd appears to be missing"
     exit
  fi
fi

/usr/local/amber12/bin/ptraj ${NAME}_wat.prmtop < $
{SCRIPTS_DIR}/extract_frame_from_clust_edited.trajin

mkdir ./clustering/minimisation
cp "extracted_${NAME}_frame_${crd_file_num}_${frame_num}.restrt.${frame_num}"
./clustering/minimisation
#have to put the frame number after the extension due to quirk in ptraj
cp ${NAME}_wat.prmtop ./clustering/minimisation

cd ./clustering/minimisation
cp "extracted_${NAME}_frame_${crd_file_num}_${frame_num}.restrt.${frame_num}" "min0.restrt"
cp ${SCRIPTS_DIR}/min_steepdesc_50cyc.in .
cp ${SCRIPTS_DIR}/repeat_min .
sed -i s/NAME/"${NAME}_wat"/g repeat_min

num_repeats=5
./repeat_min $(($num_repeats -1 )) #Andre's repeat scripts all use a gt comparison
```

```
ambpdb -p ${NAME}_wat.prmtop < min${num_repeats}.restrt > ${NAME}_bestmember_minimised.pdb

cp ${SCRIPTS_DIR}/trajin_strip_solvation.in .
sed -i s/SOLVATEDFILE/"${NAME}_bestmember_minimised"/g trajin_strip_solvation.in
sed -i s/STRIPPEDFILE/"${NAME}_bestmember_minimised_stripped"/g trajin_strip_solvation.in
cpptraj ${NAME}_wat.prmtop < trajin_strip_solvation.in

FMO_facio_input_conversion.sh ${NAME}_bestmember_minimised_stripped.mol2

crashley=`echo $crashley`
PDB_DIR=${crashley}/facio_files/minimised_bestmember_pdbs
cp ${NAME}_bestmember_minimised_stripped_facio_ready.pdb $PDB_DIR
#should append a label onto my minimised pdbs to say how many cycles they have been minimised
for.
```

S3.4.1. extract_frame_from_clust_template.trajin

```
trajin mdCRDNUM.mdcrd.gz FRAMENUM FRAMENUM
center :1-29 origin mass
image origin center familiar
trajout extracted_NAME_frame_CRDNUM_FRAMENUM.restrt restart
```

S3.4.2. trajin_strip_solvation.in

```
trajin SOLVATEDFILE.pdb
strip :WAT
strip :Na+
trajout STRIPPEDFILE.mol2 mol2
```

S3.5. FMO_facio_input_conversion.sh

```
#!/bin/bash
SCRIPTS_DIR=/home/keiran/scripts
units_dir=/home/keiran/units

file="$1"
basename=${file%.mol2}

#Variables for name sanity checking
type=unknown
mer=unkown
sequence=unknown
unit=unkown
groove=unknown

num_fields=$(echo "$file" | awk -F '_' '{print NF}' )
if [ $num_fields -eq 4 ]
  then
    echo "Likely is free DNA"
    type=free_dna
    sequence=$(echo "$file" | awk -F '_' '{print $1}' )
    unit=none
    groove=none
elif [ $num_fields -eq 6 ]
  then
    echo "DNA with drug"
    type=dna_drug
    sequence=$(echo "$file" | awk -F '_' '{print $1}' )
    unit=$(echo "$file" | awk -F '_' '{print $2}' )
    groove=$(echo "$file" | awk -F '_' '{print $3}' )
elif [ $num_fields -eq 7 ]
  then
    if [[ ! "$(echo "$file" | awk -F '_' '{print $1}' )" -eq "dual" ]]
        then
          type=invalid
          echo "If the complex is dual intercalated, 2nd field separated by underscore should
be "dual". Otherwise file naming issue"
          exit
    fi
    echo "DNA with 2 monointercalators"
    type=dna_2mono
    sequence=$(echo "$file" | awk -F '_' '{print $1}' )
```

```
    unit=$(echo "$file" | awk -F '_' '{print $3}' )
    groove=$(echo "$file" | awk -F '_' '{print $4}' )
else
  echo "I'm confused! The input file isn't named correctly, wrong amount of fields separated
by underscores."
  type=invalid
  exit
fi

#Regex to make sure valid dna sequence
if [[ $sequence =~ ^[ACTG]+$ ]]
then
  echo "Valid sequence"
else
  type=invalid
  echo "Invalid sequence, please make sure file name is correct"
  exit
fi

if [[ ${#sequence} -eq 4 ]]
  then mer=14
elif [[ ${#sequence} -eq 3 ]]
  then mer=13
else
  type=invalid
fi

echo "type is $type"
echo "sequence  name is $sequence"
echo "unit name is $unit"
echo "groove side is $groove"
echo "mer is ${mer}-mer"

#Arithmetic to determine the number of counter-ions to add
if [[ $mer -eq 14 ]]
  then
    num_ions=26
elif [[ $mer -eq 13 ]]
  then
    num_ions=24
else
  type=invalid
  echo "This seems to be the wrong type of -mer! Please check file name."
  exit
fi

if [ "$type" == "free_dna" ];
  then
    :
elif [ "$type" == "dna_drug" ];
  then
    if [ "$unit" == "C3NC3" ] #Check that it's not the +3 ligand rather than all the other +2
ligands
      then
         num_ions=$(($num_ions-3))
    else
      num_ions=$(($num_ions-2))
    fi
elif [ "$type" == "dna_2mono" ];
  then
    num_ions=$(($num_ions-2))
else
  type=invalid
  echo "Problem getting complex type, please check naming conventions."
fi

if [ "$type" == "invalid" ];
  then
  echo "Somewhere along the way the complex type became invalid, check your naming
conventions."
fi

if [ "$unit" == "C3NC3" ]
```

```
  then
  unit="C3"  #Sadly the Amber programs seem to cap at 3 letter unit
fi

if [ "$unit" == "9aa" ]
  then
  unit="9a"
fi

if [ "$unit" == "9AA" ]
  then
  unit="9a"
fi

#See what the program has decided
echo "Complex -mer is ${mer}-mer"
echo "Complex type is $type"
echo "Therefore adding $num_ions Na+ counter-ions"

#Fix hydrogen naming incopmatibilites and replace MOL with unit type
cp $file ${basename}_fixed.mol2
#Below was need when I was using .pdb files instead of .mol2 files
#sed -e s/MOL/"${unit}L"/g -e s/\'HO5/HO5\'/g -e s/\'HO3/HO3\'/g -e s/\'H5\'/H5\'\'/g -e
s/\'H2\'/H2\'\'/g  < "$file" > "${basename}_fixed.mol2"

#Create leap template for adding ions in tleap, then add those counter-ions
sed -e s#UNITS_DIR#"$units_dir"#g -e s/UNIT/"${unit}L"/g -e s/BASENAME/"$basename"/g -e
s/NUMIONS/$num_ions/g < ${SCRIPTS_DIR}/leap_addions_template.cmd > $
{SCRIPTS_DIR}/leap_addions_edited.cmd
tleap -f ${SCRIPTS_DIR}/leap_addions_edited.cmd

#Fix more naming incompatibles with facio
sed -e s/HO5\'/\ H5\'/g -e s/HO3\'/\ H3\'/g -e s/\'H5/\ H5/g -e s/\'H2/\ H2/g -e s/\ Pt/Pt\ /g
< "${basename}_ions.pdb" > "${basename}_facio_ready.pdb"
```

S3.5.1. leap_addions_template.cmd

```
source leaprc.ff12SB
source leaprc.gaff
loadamberparams frcmod.ionsjc_tip3p
loadoff UNITS_DIR/UNIT.lib
loadamberparams UNITS_DIR/UNIT.frcmod
dna = loadmol2 BASENAME.mol2
addions dna Na+ NUMIONS
savepdb dna BASENAME_ions.pdb
quit
```

S4: Parameter Files

S4.1. New (terpy)Pt(II) Thiolate Parameters

Below is the force-field parameter file *terpy-Pt_thiol.frcmod*, which contains all derived parameters for the (terpy)Pt(II) thiolate moiety. Energetic parameters were fit to M06/6-31G* relaxed geometry scans in Gaussian. Equilibrium bond lengths were taken from a M06/TZP + scalar-ZORA geometry optimisation in ADF. *terpy-Pt_thiol.mol2* contains atom definitions, topology and partial charges for this moiety. The partial charges were calculated in ADF using the same level of theory but increasing to a TZ2P basis set. Please refer to *complete_terpy-Pt_fits_to_DFT_scans.ods* on the supplementary disc for fit data and plots.

S4.1.1. terpy-Pt_thiol.frcmod

```
# (terpyridine)-Pt-thiol parameters developed by Keiran Rowell for his 2014 Honours
project at UNSW. Used in conjunction with MDC-q partial charges from a TZ2P basis
set.

MASS
PT 195.01        0.000       Atom no. Would need suitable polarizability, however
using ff12SB, not a polarisable force-field.

Force constants: M06/6-31G*|LANL2DZ relaxed scan. Equilibrium values: M06/TZP-STO
all electron basis set with scalar-ZORA relativistic correction, Cs symmetry
enforced. Methyl substituent for thiol. Henceforth designated with " "

BOND
PT-na   254.89   2.060       " "
PT-ss   200.40   2.348       " "

ANGLE
PT-na-ca  100.897     126.900   " "
PT-na-cp  119.123     113.000   " "
PT-ss-c3   49.793     100.200   " "
na-PT-na  114.631     079.800   " "
na-PT-ss   87.017     100.200   " "
na-cp-cp   73.000     108.790   same as cp-ca-na, antechamber defined
cp-na-cp   65.880     120.960   same as ca-na-cp  antechamber defined

DIHE
PT-na-cp-ca   1   38.529       180.000           2.000      " "
PT-na-cp-cp   1   38.529       180.000           2.000      same as PT-na-cp-ca
na-PT-na-ca   1   40.000       180.000           2.000      " "
na-PT-na-cp   1   40.000       180.000           2.000      same as na-PT-na-cp
ca-na-PT-ss   1   23.487       180.000           2.000      " "
cp-na-PT-ss   1   23.487       180.000           2.000      same as ca-na-PT-ss
ca-na-cp-ca   1   49.535       180.000           2.000      " "
ca-na-cp-cp   1   49.535       180.000           2.000      same as ca-na-cp-ca
cp-na-cp-ca   1   49.535       180.000           2.000      same as ca-na-cp-ca
cp-cp-na-cp   1   49.535       180.000           2.000      same as ca-na-cp-ca
na-PT-ss-c3   1   00.900       360.000          -2.000      Genetic algorithm  fit
to M06/6-31G* torisional scan. Script courtesy of Sergio Ruiz @ Universtat de
Barcelona
na-PT-ss-c3   1   00.100       180.000           4.000       2nd function from
genetic algorithm for better fit


IMPROPER
PT-ca-na-cp         1.1          180.0         2.0          Using default value
PT-cp-na-cp         1.1          180.0         2.0          Using default value
ca-h4-ca-na         1.1          180.0         2.0          Using default value
ca-ca-ca-ha         1.1          180.0         2.0          General improper
torsional angle (2 general atom types)
ca-cp-ca-ha         1.1          180.0         2.0          General improper
torsional angle (2 general atom types)
ca-cp-cp-na         1.1          180.0         2.0          Using default value

NONBON
  PT          1.7800  0.2000            Hambley T., Inorg Chem, 1998, vol. 37, 3767
```

S4.1.2. terpy-Pt_thiol.mol2

```
@<TRIPOS>MOLECULE
LIG
   35    39     1     0     0
SMALL
```

No Charge or Current Charge

```
@<TRIPOS>ATOM
      1 N          -0.0140    1.1780    3.8690 na        1 LIG      -0.341740
      2 N1         -0.1280    2.8160    5.8960 na        1 LIG      -0.543651
      3 C          -0.1970    3.7180    0.8670 ca        1 LIG       0.322587
      4 C1         -0.0320    1.4900    1.5380 cp        1 LIG       0.301196
      5 C2          0.0410    0.5650    2.6860 cp        1 LIG       0.172441
      6 C3          0.0410    0.5650    5.0510 cp        1 LIG       0.172441
      7 C4         -0.0320    1.4900    6.1990 cp        1 LIG       0.301196
      8 C5         -0.1970    3.7180    6.8700 ca        1 LIG       0.322587
      9 C6         -0.1710    3.3520   -0.4660 ca        1 LIG      -0.439469
     10 H          -0.2830    4.7490    1.1980 h4        1 LIG       0.130806
     11 C7         -0.0070    1.0740    0.2250 ca        1 LIG      -0.207415
     12 C8          0.1580   -0.8140    2.6610 ca        1 LIG      -0.217942
     13 C9          0.1580   -0.8140    5.0770 ca        1 LIG      -0.217942
     14 C10        -0.0070    1.0740    7.5120 ca        1 LIG      -0.207415
     15 C11        -0.1710    3.3520    8.2040 ca        1 LIG      -0.439469
     16 H1         -0.2830    4.7490    6.5400 h4        1 LIG       0.130806
     17 C12        -0.0780    2.0140   -0.7900 ca        1 LIG      -0.082635
     18 H2         -0.2290    4.1140   -1.2290 ha        1 LIG       0.314549
     19 H3          0.0640    0.0210   -0.0070 ha        1 LIG       0.253540
     20 S          -0.3500    5.5190    3.8690 ss        1 LIG      -0.443473
     21 C13         0.2150   -1.4910    3.8690 ca        1 LIG       0.034109
     22 H4          0.2070   -1.3560    1.7280 ha        1 LIG       0.144232
     23 H5          0.2070   -1.3560    6.0090 ha        1 LIG       0.144232
     24 C14        -0.0780    2.0140    8.5280 ca        1 LIG      -0.082635
     25 H6          0.0640    0.0210    7.7450 ha        1 LIG       0.253540
     26 H7         -0.2290    4.1140    8.9670 ha        1 LIG       0.314549
     27 H8         -0.0600    1.6980   -1.8240 ha        1 LIG       0.242042
     28 H9          0.3080   -2.5680    3.8690 ha        1 LIG       0.215504
     29 Pt         -0.1570    3.1790    3.8690 PT        1 LIG       0.610232
     30 N2         -0.1280    2.8160    1.8420 na        1 LIG      -0.543651
     31 H10        -0.0600    1.6980    9.5620 ha        1 LIG       0.242042
     32 H11         1.4560    7.0760    3.8690 h1        1 LIG       0.094293
     33 H12         1.9280    5.6210    4.7570 h1        1 LIG       0.009655
     34 C15         1.4160    5.9890    3.8690 c3        1 LIG       0.031205
     35 H13         1.9280    5.6210    2.9810 h1        1 LIG       0.009655
@<TRIPOS>BOND
     1    1    5 ar
     2    1    6 ar
     3    1   29 1
     4    2    7 ar
     5    2    8 ar
     6    2   29 1
     7    3    9 ar
     8    3   10 1
     9    3   30 ar
    10    4    5 1
    11    4   11 ar
    12    4   30 ar
    13    5   12 ar
    14    6    7 1
    15    6   13 ar
    16    7   14 ar
    17    8   15 ar
    18    8   16 1
    19    9   17 ar
    20    9   18 1
    21   11   17 ar
    22   11   19 1
    23   12   21 ar
    24   12   22 1
    25   13   21 ar
```

```
    26   13   23 1
    27   14   24 ar
    28   14   25 1
    29   15   24 ar
    30   15   26 1
    31   17   27 1
    32   20   29 1
    33   20   34 1
    34   21   28 1
    35   24   31 1
    36   29   30 1
    37   32   34 1
    38   33   34 1
    39   34   35 1
@<TRIPOS>SUBSTRUCTURE
     1 LIG         1 TEMP              0 ****  ****    0 ROOT
```

S4.1.3. Comparison Between DFT and Crystal Structure Values

Bond type	Crystal Structure	MO6/TZP value
Coordination sphere		
Pt-S	2.303	2.348
Pt-N1	2.023	2.06
Pt-N2	1.968	2.006
Pt-N3	2.03	2.06
Terpy ligand		
N1-C1	1.347	1.33
N1-C5	1.368	1.363
N2-C6	1.339	1.333
N2-C10	1.338	1.333
N3-C11	1.389	1.363
N3-C15	1.344	1.333
C1-C2	1.364	1.383
C2-C3	1.375	1.379
C3-C4	1.393	1.386
C4-C5	1.367	1.378
C5-C6	1.476	1.476
C6-C7	1.378	1.384
C7-C8	1.384	1.386
C8-C9	1.386	1.386
C9-C10	1.369	1.384
C10-C11	1.475	1.476
C11-C12	1.376	1.378
C12-C13	1.371	1.386
C13-C14	1.383	1.379
C14-C15	1.371	1.383

Angle type	Crystal Structure	MO6/TZP value
Coordination sphere		
N1-Pt-N2	80.6	79.8
N2-Pt-N3	80.8	79.8
N1-Pt-S	100.4	100.2
N3-Pt-S	98.1	100.2
N1-Pt-N3	161.4	159.6
N2-Pt-S	178.9	179.4
Terpy ligand		
C1-N1-C5	118.4	120
C1-N1-Pt	128.4	126.9
C5-N1-Pt	113.2	113
C15-N3-C11	118.7	120
C15-N3-Pt	128.5	126.9
C11-N3-Pt	112.7	113
C3-C4-C5	119.9	119.5
C4-C5-N1	121	120.5
C4-C5-C6	123.4	123.4
N1-C5-C6	115.6	116.1
N2-C6-C7	119.1	118.6
N2-C6-C5	112.6	113.5
C5-C6-C7	128.3	127.9
C6-C7-C8	118.3	118.3
C7-C8-C9	120.8	121.3
C8-C9-C10	119	118.3
C10-N2-C6	123.9	125
C10-N2-Pt	118.1	117.5
C6-N2-Pt	118	117.5
N1-C1-C2	122.7	121.7
C1-C2-C3	119.4	119
C2-C3-C4	118.6	119.3
N2-C10-C9	118.9	118.6
N2-C10-C11	113	113.5
C9-C10-C11	128.1	127.9
C12-C11-N3	119.7	120.5
C12-C11-C10	125	123.4
N3-C11-C10	115.3	116.1
C11-C12-C13	120.8	119.5
C12-C13-C14	119.2	119.3
C13-C14-C15	119.1	119
N3-C15-C14	122.5	121.7

Mean absolute errors:

Bonds (Å)	0.013
Angles (deg)	0.72

Experimental structures taken from:
Jennette, K *et al.*, *JACS,* **1975**, 213, 6159–6168.

Calculated values taken from ADF optimisation:
M06/TZP, no frozen core, scalar ZORA correction.

S5: 3DNA Structural Values

Refer to the supplementary DVD folder '3DBA_values_of_minimised_bestmember_frames' for 3DNA output of the best member frames of all complexes in this study. Below is the intercalation site values for all 2bps complexes with the **C-6** and the **C-8** ligands.

CGCG_C6_major:

Step	Buckle	Propeller	Twist	Rise	Inc.	Tip	Roll	Strand I: χ	Pucker	Strand II: χ	Pucker
GC/GC	-15.29	-18.03	39.09	3	-7.86	-9.51	-5.33	-151	C4'-exo	-125.5	*C1'-exo*
CG/CG	0.93	-3.03	-0.15	7.05	-87.45	-15.98	7.48	-118.2	C4'-exo	-81	*C1'-exo*
GC/GC	-14.1	1.89	24.98	2.71	20.09	12.47	9.19	-113.6	O4'-endo	-120	*C4'-exo*
CG/CG	14.9	4.2	10.77	7.17	14.5	-28.42	3.04	-114.7	C1'-exo	-131.4	*C4'-exo*
GC/GC	-9.19	2.09	37.17	3.05	-3.86	13.68	-2.5	-74.7	C1'-exo	-104.6	*C4'-exo*

CGCG_C6_minor:

Step	Buckle	Propeller	Twist	Rise	Inc.	Tip	Roll	Strand I: χ	Pucker	Strand II: χ	Pucker
GC/GC	2.95	-4.62	31.65	3.3	-5.18	-7.82	-2.85	-132.9	O4'-endo	-108.3	C1'-exo
CG/CG	8.42	-2.87	18.03	7.29	-33.25	-8.95	-11.84	-129.9	O4'-endo	-88.8	C3'-exo
GC/GC	0.6	-7.31	42.15	3.27	-13.22	-4.79	-9.68	-77.8	C1'-exo	-112.1	C2'-endo
CG/CG	5.58	-16.34	16.53	7.14	-16.9	23.98	-5.32	-110.1	O4'-endo	-80.7	C1'-exo
GC/GC	2.75	-4.19	27.65	3.53	5.79	6.43	2.79	-108.3	C1'-exo	-139.3	O4'-endo

CGCG_C8_major:

Step	Buckle	Propeller	Twist	Rise	Inc.	Tip	Roll	Strand I: χ	Pucker	Strand II: χ	Pucker
GC/GC	4.81	-7.49	41.6	3.15	8.43	-11.01	6.09	-104.8	C1'-exo	-109	C2'-endo
CG/CG	15.68	-2.68	9.09	7.26	10.97	43.51	2.21	-115.3	C3'-endo	-78.9	C1'-exo
GC/GC	-6.36	2.53	16.08	2.78	26.56	-22.48	8.47	-136	C4'-exo	-113.8	O4'-endo
CG/CG	21.13	-5.71	5.89	7.28	43.86	-29.71	6.46	-117.4	C4'-exo	-108.1	C1'-exo
GC/GC	-0.23	-2.48	40.51	3.05	1.56	9.76	1.09	-79.9	C1'-exo	-116.3	C4'-exo

CGCG_C8_minor:

Step	Buckle	Propeller	Twist	Rise	Inc.	Tip	Roll	Strand I: χ	Pucker	Strand II: χ	Pucker
GC/GC	1.15	-0.5	26.37	2.95	8.69	-15.95	4.1	-144.3	C4'-exo	-134.9	O4'-endo
CG/CG	20.95	-1.26	21.35	7.58	10.12	9.29	3.82	-127.5	C3'-endo	-121.4	O4'-endo
GC/GC	-18.53	5.57	29.89	3.08	-5.65	7.47	-2.94	-127	C2'-endo	-117.7	C1'-exo
CG/CG	3.25	11.7	20.87	7.24	-27.85	-19.75	-11.42	-123.7	C1'-exo	-138.3	O4'-endo
GC/GC	-7.45	-4.75	35.31	3.45	-4.79	1.87	-2.91	-111.9	C2'-endo	-131.8	C4'-exo

TATA_C6_major:

Step	Buckle	Propeller	Twist	Rise	Inc.	Tip	Roll	Strand I: χ	Pucker	Strand II: χ	Pucker
GT/AC	-0.2	-8.05	36.06	3.25	-7.2	-7.99	-4.5	-111.7	C2'-endo	-118.4	C2'-endo
TA/TA	4.13	-4.47	18.51	6.98	12.82	18.76	4.35	-116.6	C1'-exo	-84.7	C1'-exo

AT/AT	-11.45	13.21	18.9	3.04	23.48	-2.07	8.16	-86.9	C4'-endo	-122.3	O4'-endo
TA/TA	3.64	1.39	5.38	7.07	-27.9	0.26	-2.85	-125.3	C4'-exo	-90.8	C1'-exo
AC/GT	-5.71	2.77	32.52	3.06	-5.42	6.83	-3.06	-84.6	C1'-exo	-118.7	O4'-endo

TATA_C6_minor:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GT/AC	-4.75	-8.49	31.97	3.07	1.72	-9.83	0.96	-116.1	O4'-endo	-108.6	C1'-exo
TA/TA	8.44	-0.56	8.7	7.39	20.68	18.07	3.4	-116.5	C1'-exo	-77.6	C1'-exo
AT/AT	-8.52	-6.64	34.41	3.15	-20.05	3.04	-12.33	-84.2	C1'-exo	-143.8	C4'-exo
TA/TA	2.8	-12.18	33.3	7.13	-32.6	19.01	-21.6	-117.7	C2'-endo	-93.7	C1'-exo
AC/GT	-19.85	-1.01	18	2.78	1.33	1.71	0.42	-107	C2'-endo	-111.7	C2'-endo

TATA_C8_major:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GT/AC	3.07	-11.59	35.39	3.33	-15.35	-3.41	-9.55	-115.4	C2'-endo	-127.8	C1'-exo
TA/TA	6.6	-1.19	16.32	6.89	2.07	19.93	0.61	-140.5	C4'-exo	-79.6	C1'-exo
AT/AT	-4.21	11.8	16.97	3.07	29.27	-2.93	9.47	-95.2	C2'-endo	-112	O4'-endo
TA/TA	13.8	-5.48	9.66	6.85	7.52	-7.33	1.28	-124.5	O4'-endo	-109.5	O4'-endo
AC/GT	11.78	-6.05	35.55	3.35	-6.96	3.9	-4.28	-71.5	C1'-exo	-136.7	O4'-endo

TATA_C8_minor:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GT/AC	6.56	1.1	22.36	3.29	-14.84	-0.28	-5.88	-114.9	C1'-exo	-144.5	O4'-endo
TA/TA	19.36	1.82	34.18	7.39	-13.12	-0.72	-7.83	-140.6	C4'-exo	-107.9	C2'-endo
AT/AT	-8.52	15.54	21.32	3.37	-20.01	8.95	-7.78	-113.1	C2'-endo	-110.4	C1'-exo
TA/TA	8.35	11.45	22.97	7.38	-13.9	-22.39	-5.94	-134.6	O4'-endo	-128.4	C2'-endo
AC/GT	-18	-12.32	26.05	3.22	2.6	3.47	1.17	-110.8	C1'-exo	-128.5	C4'-exo

CACA_C6_major:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GC/GC	-10.01	-8.98	29.34	3.15	1.09	-2.35	0.55	-132.5	C1'-exo	-105.1	C2'-endo
CA/TG	-2.29	0.5	15.54	6.68	-7.87	-12.94	-2.18	-117.2	C1'-exo	-87.4	C2'-endo
AC/GT	-1.4	-12.88	27.08	3.07	17.23	-1.19	8.31	-79.7	C3'-exo	-130.5	C4'-exo
CA/TG	2.61	-19.8	28.74	6.79	-4.45	2.27	-2.22	-119.1	O4'-endo	-97.8	C1'-exo
AC/GT	-12.82	-8.16	30.16	3.15	1.64	3.06	0.86	-106.1	C1'-exo	-129.2	C1'-exo

CACA_C6_minor:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GC/GC	1.59	-13.46	26.52	2.97	8.23	-0.82	3.8	-102.1	C1'-exo	-133.6	O4'-endo
CA/TG	18.03	-6.66	25.28	7.12	-17.93	-11.97	-8.22	-111.6	O4'-endo	-102.7	C2'-endo

AC/GT	-9.96	-7.95	41.83	3.49	-34.32	-0.62	-27.39	-95.1	C1'-exo	-112.3	C2'-endo
CA/TG	-1.19	3.04	9.35	7.11	34.95	2.43	6.53	-134.4	O4'-endo	-81.1	C1'-exo
AC/GT	-13.34	-0.57	24.46	2.94	13.43	0.59	5.79	-99	C1'-exo	-122	O4'-endo

CACA_C8_major:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GC/GC	-7.93	-11.75	30.89	3.13	2.91	8.17	1.56	-137.1	O4'-endo	-114.8	C2'-endo
CA/TG	-0.5	-0.1	24.72	6.63	-11.59	-23.23	-5.31	-142.1	O4'-endo	-116.9	C3'-exo
AC/GT	-7.38	1.36	23.93	2.95	12.31	7.23	5.21	-89.4	C1'-exo	-129.4	C4'-exo
CA/TG	8.96	-3.04	11.38	7.14	-4.39	-31.27	-0.97	-128.1	C4'-exo	-117.6	O4'-endo
AC/GT	-10.03	-5.4	40.2	3.36	1.29	10.37	0.89	-90.8	C1'-exo	-111.6	O4'-endo

CACA_C8_minor:

Step	**Buckle**	**Propeller**	**Twist**	**Rise**	**Inc.**	**Tip**	**Roll**	**Strand I: χ**	**Pucker**	**Strand II: χ**	**Pucker**
GC/GC	-3.18	-1.43	27.03	3	10.15	5.13	4.8	-111	C2'-endo	-117.9	O4'-endo
CA/TG	9.22	-7.71	20.92	7	-10.67	8.85	-3.95	-130.6	O4'-endo	-87.7	C1'-exo
AC/GT	1.42	3.72	37.45	3.71	-23.75	-2.66	-16.09	-84.7	C1'-exo	-137.6	O4'-endo
CA/TG	-1.25	-0.09	19.49	6.93	12.32	-8.64	4.27	-125.4	C1'-exo	-77.6	C1'-exo
AC/GT	-8.05	-19.84	33.45	3.14	3.93	3.73	2.27	-85.5	C1'-exo	-131.2	O4'-endo